KeyStone Architecture
Packet Accelerator (PA)

User Guide

Literature Number: SPRUGS4A
July 2012

Release History

Release	Date	Description/Comments
SPRUGS4A	July 2012	• Added Clock Control section (Page 2-2) • Added Packet Accelerator Queue Interface Section. (Page 2-2) • Added Programming the Packet Accelerator with the Low-Level Driver section. (Page 2-2) • Added the L4 Classify Engine Architecture section. (Page 2-4) • Added the L3 Classify Engine Architecture section. (Page 2-4) • Added the L2 Classify Engine Architecture section. (Page 2-4) • Added the Modify/Multiroute Engine Architecture section. (Page 2-5) • Removed the PDSP, LUT1, LUT2, and Timer Architecture sections. (Page 2-6) • Updated Receive classification dataflow example. (Page 3-2) • Updated transmit checksum generation dataflow example. (Page 3-3) • Added Power Management Section (Page 2-6) • Corrected the number of receive channels from 26 to 24 (Page 1-2)
SPRUGS4	November 2010	Initial Release

Contents

List of Tables

List of Figures

Preface

About This Manual

The packet accelerator (PA) is one of the main components of the network coprocessor (NETCP) peripheral. The PA works together with the security accelerator (SA) and the gigabit Ethernet switch subsystem to form a network processing solution. The purpose of PA in the NETCP is to perform packet processing operations such as packet header classification, checksum generation, and multi-queue routing.

Notational Conventions

This document uses the following conventions:

- Commands and keywords are in **boldface** font.
- Arguments for which you supply values are in *italic* font.
- Terminal sessions and information the system displays are in `screen font`.
- Information you must enter is in **`boldface screen font`**.
- Elements in square brackets ([ ]) are optional.

Notes use the following conventions:

Note—Means reader take note. Notes contain helpful suggestions or references to material not covered in the publication.

The information in a caution or a warning is provided for your protection. Please read each caution and warning carefully.

CAUTION—Indicates the possibility of service interruption if precautions are not taken.

WARNING—Indicates the possibility of damage to equipment if precautions are not taken.

Related Documentation from Texas Instruments

Gigabit Ethernet (GbE) Switch Subsystem for KeyStone Devices User Guide	SPRUGV9
Interrupt Controller (INTC) for KeyStone Devices User Guide	SPRUGW4
Multicore Navigator for KeyStone Devices User Guide	SPRUGR9
Network Coprocessor (NETCP) for KeyStone Devices User Guide	SPRUGZ6
Security Accelerator (SA) for KeyStone Devices User Guide	SPRUGY6

Trademarks

Chapter 1

Introduction

1.1 Purpose of the Peripheral

The packet accelerator (PA) is one of the main components of the network coprocessor (NETCP) peripheral. The PA works together with the security accelerator (SA) and the gigabit Ethernet switch subsystem to form a network processing solution. The purpose of PA in the NETCP is to perform packet processing operations such as packet header classification, checksum generation, and multi-queue routing.

1.2 Features

The NETCP has the following hardware features:

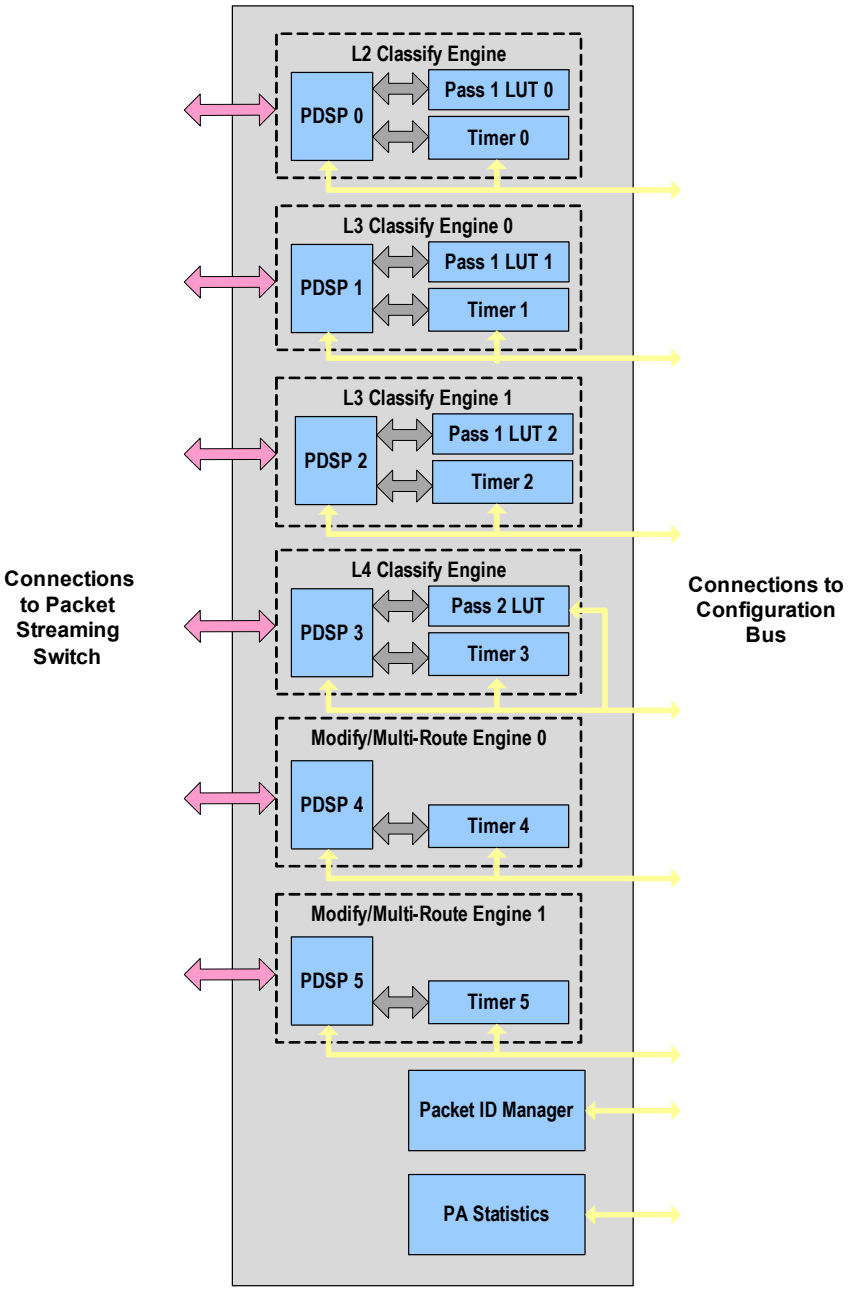
- Provides high performance packet DMA controller
  - 9 transmit channels
  - 24 receive channels
- Provides packet accelerator for packet lookup and modification operations
  - Contains 6 independent packed data structure processors (PDSP)
  - Provides 3 1st pass lookup table (LUT1) accelerators
    - Supports up to 64 entries
    - All lookup operations occur at line rate
    - Lookup result based on field match scoring (based on RFC4301)
  - Provides 1 2nd pass lookup table accelerator
    - Supports up to 8K entries
    - All lookup operations occur at line rate
    - Lookup can occur while add or delete operation is ongoing

1.3 Functional Block Diagram

Figure 1-1 shows the packet accelerator (PA) functional block diagram. The PA provides the following hardware modules:

- Layer 2 Classification Engine
- (2) Layer 3 Classification Engines
- Layer 4 Classification Engine
- (2) Modification/Multiroute Engines
- Packet ID Manager
- PA Statistics Module

Figure 1-1 Packet Accelerator Functional Block Diagram



Chapter 2

Architecture

This chapter describes the details of the packet accelerator architecture.

2.1 Clock Control

The packet accelerator (PA) has one clock, which it receives from the network coprocessor (NETCP). This clock is used to operate all of the PA logic, plus the common NetCP logic, such as the packet DMA controller and the packet streaming switch; therefore, the PA clock domain must also be enabled when using the security accelerator (SA) or the gigabit Ethernet (GbE) switch. To reduce power consumption, this clock is disabled by default; therefore, this clock must be enabled before using the PA. For more information about this clock, including the operating frequency, see the *Network Coprocessor (NETCP) for KeyStone Devices User Guide*, and the device-specific data manual. For more information about power management for the PA, please see Section 2.10 "Power Management" on page 2-6.

2.2 Packet Accelerator Transmit Queue Interface

When sending packets to the packet accelerator (PA) from the Host, packets must be sent through the multicore navigator queue interface. There are 6 hardware TX queues that are used to sent packets to the PA for processing, with each queue directly mapped to a specific engine in the PA. When a descriptor (with linked packet) is pushed into one of the TX queues, the PA packet DMA automatically transfers the packet to the specific engine in the PA. The mapping between the queues and the PA hardware processing engines are shown in Table 2-1.

Table 2-1 Queue to PA Engine Mapping

Queue	PA Engine
640	L2 Classification Engine
641	L3 Classification Engine 0
642	L3 Classification Engine 1
643	L4 Classification Engine
644	Modify/Multiroute Engine 0
645	Modify/Multiroute Engine 1
End of Table 2-1	

For more information about the packet DMA and queues, please see the multicore navigator user guide.

2.3 Programming the Packet Accelerator with the Low-Level Driver

To ease the task of programming the packet accelerator (PA) by abstracting many of the hardware details, a low level driver (LLD) software package has been generated for use with the PA. Included with the PA LLD are firmware images that must be loaded onto the PDSPs before using the PA. Due to interdependencies between the firmware and the PA LLD, all users must use the PA LLD. Failure to use the PA LLD will result in undefined behavior. The PA LLD provides a set of APIs that can be broken into the following 3 categories:

- System APIs - provide basic system functions for the PA
- Configuration APIs - primarily used to configure the PA
- Utility Functions - format commands for use by the PA

The PA LLD APIs are briefly described in Table 2-2. PA LLD APIs are occasionally added and changed, so please see the PA LLD documentation for the most complete and up to date information about the PA LLD APIs.

Table 2-2 SA LLD APIs

API	Description
SYSTEM FUNCTIONS	
Pa_getBufferReq	Returns the memory requirements for the PA driver.
Pa_create	Creates the PA driver instance.
Pa_startCfg	Adds PA configuration.
Pa_close	Deactivates the PA driver instance.
Pa_control	Performs system-level control and configuration.
Pa_resetControl	Controls the reset state of the Sub-system.
Pa_downloadImage	Downloads a PDSP firmware image to a sub-system when the packet processing module is in reset.
Pa_getHandleRefCount	Returns the number of reference channels linked to the LUT1 handle.
Pa_getVersion	Returns the PA LLD version information.
Pa_getVersionStr	Returns the PA LLD version string.
Pa_requestStats	Requests sub-system statistics.
Pa_formatStatsReply	Formats the statistics reply from the PA.
Pa_requestUsrStats	Queries user-defined statistics.
Pa_getTimestamp	Returns the 48-bit system timestamp.
CONFIGURATION FUNCTIONS	
Pa_addMac	Adds a mac address to the L2 table.
Pa_addIp	Adds an IP address to the L3 table.
Pa_addPort	Adds a destination port to the L4 (LUT2) table.
Pa_addSrio	Adds a SRIO entry to the L2 table.
Pa_setCustomLUT1	Performs the global configuration for level 3 (LUT1) custom lookups.
Pa_AddCustomLUT1	Adds a custom lookup entry to the lookup tables (LUT1).
Pa_setCustomLUT2	Performs the global configuration for level 4 (LUT2) custom lookups.
Pa_addCustomLUT2	Adds a custom lookup to the LUT2 lookup tables.
Pa_delHandle	Deletes a MAC, SRIO, IP, or Custom LUT1 handle.
Pa_delL4Handle	Deletes a UDP, TCP, GTPU, or CustomLUT2 handle.
Pa_forwardResult	Examines the reply of the sub-system to a command.
Pa_configExceptionRoute	Configures the routing of packets based on a exception condition such as MAC broadcast, multicast, or error packet.
Pa_configCmdSet	Configures the command set which consists of a list of commands.
Pa_configMultiRouteSet	Configures the multi-route group which consists of packet multi-route entries.
Pa_configCrcEngine	Configures the specified CRC engine.
Pa_configUsrStats	Configures the user-defined statistics operation.
Pa_configTimestamp	Configures the PA timer which is used to generate 48-bit timestamp.
UTILITY FUNCTIONS	
Pa_formatTxRoute	Formats the commands to add checksums and route a Tx packet.
Pa_formatRoutePatch	Formats the commands to route a packet and blind patch.
Pa_formatTxCmd	Formats a list of commands to be executed on the packets to be transmitted over the network.
End of Table 2-2	

2.4 L2 Classify Engine Architecture

This section discusses the architecture of the L2 classify engine. The L2 classify engine is used primarily for the classification of packets with MAC or other custom L2 headers. The L2 classify engine consists of a PDSP with accompanying firmware, a look up table (LUT1) and a timer that can be used by the PDSP for various timing tasks. The firmware image that runs on the PDSP is provided by the PA LLD (user-written firmware is not supported) and must be downloaded into the PDSP instruction RAM before attempting to use the PA. The Pa_downloadImage API is provided to help with this task. The PDSP firmware classifies data packets by submitting lookup requests to the LUT1 module, and the LUT1 returns the results of the lookup based on the contents of the entries stored in its lookup table. Based on the results from the LUT1 module, the PDSP will direct the packet to the next destination, or drop the packet if it did not match any of the LUT1 entries. The L2 classify engine LUT1 can store up to 64 entries. The LUT1 does not contain a register interface so entries must be added and removed by the PDSP through configuration packets. Configuration and data packets can be sent to the L2 classify engine through queue 640. Data packets can also arrive from other locations such as the gigabit Ethernet switch via the packet streaming switch.

2.5 L3 Classify Engine Architecture

This section discusses the architecture of the L3 classify engine. The PA contains 2 L3 classify engines. The L3 classify engine 0 is used primarily for the classification of packets with IP or other custom L3 headers. The L3 classify engine1 is used primarily for classifying inner IP headers in IPsec tunnels. The L3 classify engine consists of a PDSP with accompanying firmware, a look up table (LUT1) and a timer that can be used by the PDSP for various timing tasks. The firmware image that runs on the PDSP is provided by the PA LLD (user-written firmware is not supported) and must be downloaded into the PDSP instruction RAM before attempting to use the PA. The Pa_downloadImage API is provided to help with this task. The PDSP firmware classifies data packets by submitting lookup requests to the LUT1 module, and the LUT1 returns the results of the lookup based on the contents of the entries stored in its lookup table. Based on the results from the LUT1 module, the PDSP will direct the packet to the next destination, or drop the packet if it did not match any of the LUT1 entries. The L3 classify engine LUT1 can store up to 64 entries. The LUT1 does not contain a register interface so entries must be added and removed by the PDSP through configuration packets. Configuration and data packets can be sent to the L3 classify engine 0 through queue 641 and L3 classify engine 1 through queue 642. Data packets can also arrive from other locations such as the L2 classify engine via the packet streaming switch.

2.6 L4 Classify Engine Architecture

This section discusses the architecture of the L4 classify engine. The L4 classify engine is used primarily for the classification of packets with UDP, TCP, or other custom L4 headers such as GTPU. The L4 classify engine consists of a PDSP with accompanying firmware, a look up table (LUT2) and a timer that can be used by the PDSP for various timing tasks. The firmware image that runs on the PDSP is provided by the PA LLD (user-written firmware is not supported) and must be downloaded into the PDSP instruction RAM before attempting to use the PA. The Pa_downloadImage API is provided to help with this task. The PDSP firmware classifies data packets by submitting lookup requests to the LUT2 module, and the LUT2 returns the results of the lookup based on the contents of the entries stored in its lookup table. Based on the results from the LUT2 module, the PDSP will direct the packet to the next destination, such as the packet DMA for delivery to the Host, or it can drop the packet if it did not

match any of the LUT2 entries. The L4 classify engine LUT2 can store up to 8192 entries. Although the LUT2 does not contain a register interface adding entries through the register interface is not supported. All entries must be added and removed by the PDSP through configuration packets. Configuration and data packets can be sent to the L4 classify engine through queue 643. Data packets can also arrive from other locations such as the L3 classify engines via the packet streaming switch.

2.7 Modify/Multiroute Engine Architecture

This section discusses the architecture of the modify/multiroute engines. The PA contains 2 modify/multiroute engines, which are used for various tasks such as generating CRC checksums, multirouting packets, providing PA statistics data, and many other tasks. Both modify/multiroute engines contain a PDSP with accompanying firmware, and a timer that can be used by the PDSP for various timing tasks. The firmware image that runs on the PDSP is provided by the PA LLD (user-written firmware is not supported) and must be downloaded into the PDSP instruction RAM before attempting to use the PA. The Pa_downloadImage API is provided to help with this task. Packets can be sent to the modify and multiroute engines through queue 644 and queue 645. The PA LLD will determine the whether the packet should be directed to modify/multiroute engine 0 or modify/multiroute engine 1, and this information will be passed back through the APIs.

2.8 Adding Entries to the Classification Engines

This section will discuss adding entries to the classification engines. Before the classification engines can be used to classify packets, entries must be added to the classification engines' lookup tables. Entries are configurable, and are registered by sending a configuration packet to the desired configuration engine. All configuration packets must be generated using the PA LLD, which contains several APIs for adding entries (see Table 2-1 or the PA LLD documentation for more information). The PA also provides the ability to link entries between classification engines. This allows the ability to specify a specific combination of L2/L3/L4 addresses. For example, a UDP address in the L4 classify engine can be linked to an IP address in the L3 classify engine, which can then be linked to a MAC address in the L2 classify engine. The PA LLD provides many code examples for adding entries to the lookup tables, but a high level overview is provided in Procedure 2-1.

Procedure 2-1 Adding Entries to the Classification Engines

Step	Action
1	Configure the header fields that the classification engine should use to classify the packet
2	Configure the routing information that the classification engine will use to route the packet after classification
3	Optionally, configure the link to an entry in a preceding classification engine
4	Use the PA LLD to generate a configuration packet (e.g. Pa_addIp)
5	Link the configuration packet to a descriptor to prepare the packet to be sent to the PA
6	Set the protocol specific information words to communicate to the classification engine that this is a configuration packet
7	Optionally set the descriptor software information words
8	Push the configuration packed onto the transmit queue for the desired classification engine (this information is provided by the PA LLD when the configuration packet is generated)
9	The PA classification engine will send a configuration response packet to indicate whether or not the entry was successfully added to the classification engine. The user can wait for the response to proceed, or the user can go on with other tasks until the configuration response arrives.

10 Check that the entry was successfully added to the classification engine and register it with the PA LLD by using the Pa_forwardResult API to allow the PA LLD to examine the response packet.

2.9 DMA Event Support

All DMA events for the packet accelerator (PA) are handled thought the packet streaming switch and the packet DMA controller in the network coprocessor (NETCP). For more information about the packet streaming switch, see the *Network Coprocessor (NETCP) for KeyStone Devices User Guide*. For more information about the and the packet DMA controller, see the *Network Coprocessor (NETCP) for KeyStone Devices User Guide* or the *Multicore Navigator for KeyStone Devices User Guide*.

2.10 Power Management

The packet accelerator (PA) power is managed through the network coprocessor (NETCP) power domain and through disabling the clock that operates the PA logic. The PA shares its power domain with the other modules in the NETCP, and therefore, can not be powered down independently of NETCP. The clock for the PA is disabled by default, and can be controlled independently of the other modules in the NETCP. For more information about power management for the PA, see the *Network Coprocessor (NETCP) for KeyStone Devices User Guide* or the device-specific data manual.

Chapter 3

Data Flow Examples

This chapter provides examples of the data flow within the packet accelerator (PA), and interaction with other modules in the network coprocessor (NETCP).

3.1 Overview

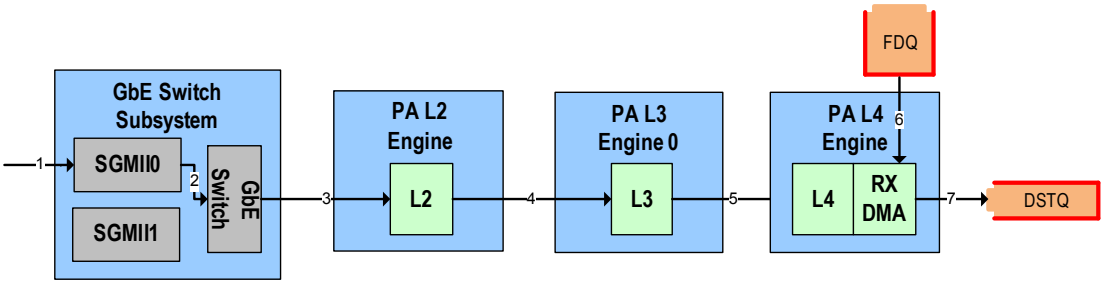
The following sections cover examples of the flow of data packets through the hardware in the packet accelerator (PA) and network coprocessor (NETCP). Since the PA is a highly configurable, the data flow is dependent on how the PA has been setup. This section covers the following examples:

- Receive Classification Example
- Transmit Checksum Generation Example

3.2 Receive Classification Example

The following section covers an example of receive packet routing in the PA for a non-encrypted data packet. The packet originates from the gigabit Ethernet (GbE) switch subsystem, and will show one example of routing the packet through the PA to the host.

Figure 3-1 Receive Classification Example



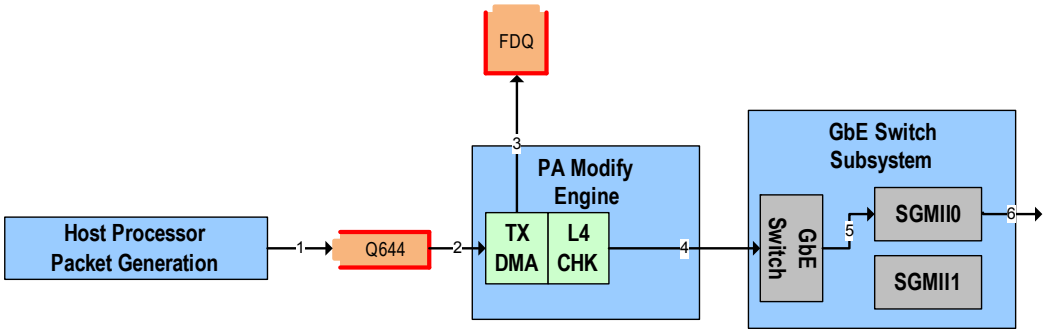
This example assumes that the L2, L3, and L4 classify engines are setup to match source and destination addresses for each header, and that the packet that arrives has MAC, IPv4, and UDP headers.

1. SGMII0 receives a packet from the wire.
2. The packet is routed into port 1 of the gigabit Ethernet Switch.
3. The packet is routed out of port 0 of the GbE switch and transferred to the PA L2 classify engine. The PA L2 classify engine matches the MAC addresses and Ethertype based on the entries in the L2 classify engine lookup table.
4. The packet is routed to the PA L3 classify 0 engine, where the L3 classify engine 0 matches the IPv4 addresses based on the entries in the L3 classify engine 0 lookup table.
5. The packet is routed to the PA L4 classify engine, where the L4 classify engine matches the UDP addresses in the packet based on the entries in the L4 classify engine lookup table.
6. The NETCP receive packet DMA uses the configured receive flow to pop a descriptor from the free descriptor queue and fills the linked data buffer with packet data.
7. After the data transfer completes, the NETCP receive packet DMA pushes the descriptor onto the destination queue.

3.3 Transmit Checksum Generation Example

The following section covers an example of transmit packet routing in the PA for a non-encrypted data packet. This shows one example of routing transmit packets through the PA to the gigabit Ethernet (GbE) switch subsystem. The packet originates from the host, and will be sent to the modify/multiroute engine for an UDP checksum before being sent out over the network.

Figure 3-2 Transmit Checksum Generation Example



This example assumes that the that the packet has MAC, IPv4, and UDP headers, and shows the modify/multiroute engine 0 being used to generate the UDP checksum.

1. Host processor generates a packet and pushes it into transmit queue 644 for modify/multiroute engine 0.
2. The NETCP transmit packet DMA pops the descriptor from the transmit queue, and transfers the packet data to modify/multiroute engine 0.
3. After the transfer completes, the NETCP transmit packet DMA pushes the transmit descriptor onto the transmit completion queue. When the PA modify/multiroute engine 0 receives the packet, it calculates an L4 checksum and inserts it into the packet.
4. The modify/multiroute engine transfers the packet to port 0 of the gigabit Ethernet switch.
5. The gigabit Ethernet switch transfers the packet out port 1 to SGMII0.
6. SGMII0 transmits the packet over the wire.

Chapter 4

Registers

This chapter describes the registers available in the packet accelerator (PA) and its submodules. For clarity, the registers for each module and submodule are described separately. Provided for each register is a bit field description and a memory offset address. The offset address values provided are relative to the associated base address of the PA module.

See the *Network Coprocessor (NETCP) User Guide* in "Related Documentation from Texas Instruments" on page ø-viii for the base address of the packet accelerator module relative to the NETCP.

Table 4-1 lists the packet accelerator registers.

Table 4-1 Packet Accelerator Memory Map

Address Offset[1]	Size	Region	Section
00000h	128	Mailbox slots	Section 4.1
00400h	16	Packet ID Allocation Registers	Section 4.2
00500h	64	Stage 2 LUT Registers	Section 4.3
01000h	32	PDSP 0 Control/Status Registers	Section 4.4
01100h	32	PDSP 1 Control/Status Registers	Section 4.4
01200h	32	PDSP 2 Control/Status Registers	Section 4.4
01300h	32	PDSP 3 Control/Status Registers	Section 4.4
01400h	32	PDSP 4 Control/Status Registers	Section 4.4
01500h	32	PDSP 5 Control/Status Registers	Section 4.4
03000h	256	PDSP 0 Timer Registers	Section 4.5
03100h	256	PDSP 1 Timer Registers	Section 4.5
03200h	256	PDSP 2 Timer Registers	Section 4.5
03300h	256	PDSP 3 Timer Registers	Section 4.5
03400h	256	PDSP 4 Timer Registers	Section 4.5
03500h	256	PDSP 5 Timer Registers	Section 4.5
06000h	256	Statistics Counting Block Registers	Section 4.6
End of Table 4-1			

1. The actual addresses of these registers are device specific. See your device-specific data manual to verify the PA register addresses.

4.1 PDSP Mailbox Slots

This section describes the PDSP Mailbox Slots available in the packet accelerator (PA). The register address offsets listed in Table 4-2 are relative to the PDSP Mailbox Slots memory region. To determine the base address of PDSP Mailbox Slots region relative to the PA memory map, please see Table 4-1.

The PDSP mailbox slots are for use by system masters in the PA to pass information to the PDSPs. Non-zero writes to the mailbox slots will cause the PDSP to be notified on the input status pin the cycle following the write. Any data stored in the mailbox slots can be read by the PDSPs.

Table 4-2 PDSP Mailbox Slots

Address Offset	Status Bit Routing
00h	PDSP 0 Mailbox Registers
10h	PDSP 1 Mailbox Registers
20h	PDSP 2 Mailbox Registers
30h	PDSP 3 Mailbox Registers
40h	PDSP 4 Mailbox Registers
50h	PDSP 5 Mailbox Registers
60h - 7Fh	Reserved

4.2 Packet ID Allocation Registers

This section describes the Packet ID Allocation Registers available in the packet accelerator (PA). The register address offsets listed in Table 4-3 are relative to the Packet ID Allocation memory region. To determine the base address of Packet ID Allocation memory region relative to the PA memory map, please see Table 4-1.

Table 4-3 Packet ID Allocation Registers

Address Offset[1]	Register	Section
0400h	Revision Register	Section 4.2.1
0404h	Soft Reset Register	Section 4.2.2
0408h	Range Limit Register	Section 4.2.3
040Ch	IdValue Register	Section 4.2.4
End of Table 4-3		

1. The actual addresses of these registers are device specific. See your device-specific data manual to verify the register addresses.

4.2.1 Packet ID Revision Register (PID_REV)

The Revision Register contains the major and minor revisions for the module.

Table 4-4 Packet ID Allocation Module Revision Register (PID_REV)

Bits	Field	Type	Reset	Description
31-30	Reserved	R	0h	Reserved.
29-16	PID_MODID	R	TBD	Module ID field.
15-11	PID_REVRTL	R	0h	RTL revision. Will vary depending on release.
10-8	PID_REVMAJ	R	1h	Major revision.
7-0	PID_REVMIN	R	0h	Minor revision.
End of Table 4-4				

4.2.2 Packet ID Allocation Module Soft Reset Register (PID_SOFT_RESET)

The Soft Reset Register is written in order to clear all previous ID allocations.

Table 4-5 Packet ID Allocation Module Soft Reset Register (PID_SOFT_RESET)

Bits	Field	Type	Reset	Description
31-0	PID_TRIGGER	W	0h	Writing anything to this field causes all internal statistics to be cleared.
End of Table 4-5				

4.2.3 Packet ID Allocation Module Range Limit Register (PID_RANGE_LIM)

Writing the range limit register sets the highest allowed packet ID value. No packet ID above this value can be allocated. (Previously allocated values above this limit could still be freed.)

Table 4-6 Packet ID Allocation Module Range Limit Register (PID_RANGE_LIM)

Bits	Field	Type	Reset	Description
31-10	Reserved	R	0h	Reserved.
9-0	RANGE_LIMIT	RW	3FFh	Contains the value of the highest allowable ID returned by allocation.
End of Table 4-6				

4.2.4 Packet ID Allocation Module ID Value Register (PID_IDVAL)

Writing the range limit register sets the highest allowed packet ID value. No packet ID above this value can be allocated. (Previously allocated values above this limit could still be freed.)

Table 4-7 Packet ID Allocation Module IdValue Register (PID_IDVAL)

Bits	Field	Type	Reset	Description
31-11	Reserved	R	0h	Reserved.
10-0	PKT_ID_VAL	RW	0h	Packet ID Value. When read, this register returns the value of a newly allocated packet ID, or returns 400h when no packet ID is available. When written, frees the supplied packet ID.
End of Table 4-7				

4.3 Stage 2 Look Up Table (LUT2) Control Registers Region

This section describes the Stage 2 Look Up Table (LUT2) available in the packet accelerator (PA). The register address offsets listed in Table 4-8 are relative to the LUT2 memory region. To determine the base address of the LUT2 memory region relative to the PA memory map, please see Table 4-1.

Table 4-8 LUT2 Control Registers Region memory Map

Address Offset	Register	Section
00h	Revision Register	Section 4.3.1
04h	Soft Reset Register	Section 4.3.2
08h – 1Fh	Reserved	Reserved
20h	Add Data 0 Register	Section 4.3.3
24h	Add Data 1 Register	Section 4.3.4
28h	Add Data 2 Register	Section 4.3.5
2Ch	Add Data 3 Register	Section 4.3.6
30h	Add/Delete Key Register	Section 4.3.7
30h	Add/Delete Control Register	Section 4.3.8
End of Table 4-8		

4.3.1 LUT2 Revision Register (REV)

The Revision Register contains the major and minor revisions for the module.

Table 4-9 LUT2 Revision Register (REV)

Bits	Field	Type	Reset	Description
31-30	Reserved	R	0h	Reserved
29-16	LUT2_MODID	R	TBD	Module ID field
15-11	LUT2_REVRTL	R	0h	RTL revision. Will vary depending on release.
10-8	LUT2_REVMAJ	R	1h	Major revision.
7-0	LUT2_REVMIN	R	0h	Minor revision.
End of Table 4-9				

4.3.2 LUT2 Soft Reset Register (LUT2_SOFT_RESET)

The Soft Reset Register is written in order to clear the contents of the LUT.

Table 4-10 LUT2 Soft Reset Register (LUT2_SOFT_RESET)

Bits	Field	Type	Reset	Description
31-0	TRIGGER	W	0h	Writing anything to this field causes the LUT to be cleared. This operation takes 2 clock cycles to complete.
End of Table 4-10				

4.3.3 Add Data 0 Register (ADD_DATA0)

The Add Data 0 Register is written in order to initialize bits 31-0 of the 128 bit data value which is to be added to the table for the specified key.

Table 4-11 Add Data 0 Register (ADD_DATA0)

Bits	Field	Type	Reset	Description
31-0	DATA0	RW	0h	This is the data which will be placed in bits 31-0 of the data portion of the added table entry.
End of Table 4-11				

4.3.4 Add Data 1 Register (ADD_DATA1)

The Add Data 1 Register is written in order to initialize bits 63-32 of the 128 bit data value which is to be added to the table for the specified key.

Table 4-12 Add Data 1 Register (ADD_DATA1)

Bits	Field	Type	Reset	Description
31-0	DATA1	RW	0h	This is the data which will be placed in bits 63-32 of the data portion of the added table entry.
End of Table 4-12				

4.3.5 Add Data 2 Register (ADD_DATA2)

The Add Data 2 Register is written in order to initialize bits 95-64 of the 128 bit data value which is to be added to the table for the specified key.

Table 4-13 Add Data 2 Register (ADD_DATA2)

Bits	Field	Type	Reset	Description
31-0	DATA2	RW	0h	This is the data which will be placed in bits 95-64 of the data portion of the added table entry.
End of Table 4-13				

4.3.6 Add Data 3 Register

The Add Data 3 Register is written in order to initialize bits 127-96 of the 128 bit data value which is to be added to the table for the specified key.

Table 4-14 Add Data 3 Register (ADD_DATA3)

Bits	Field	Type	Reset	Description
31-0	DATA3	RW	0h	This is the data which will be placed in bits 127-96 of the data portion of the added table entry.
End of Table 4-14				

4.3.7 Add/Delete Key Register (ADD_DEL_KEY)

The Add/Delete Key Register is written in order to initialize the key value that will be associated with the added or deleted data entry. For a delete, this key will be used to locate the entry in the table which is to be deleted using the normal lookup process.

Table 4-15 Add/Delete Key Register (ADD_DEL_KEY)

Bits	Field	Type	Reset	Description
31-0	KEY	RW	0h	This is the key which will be matched against during the search operation.
End of Table 4-15				

4.3.8 Add/Delete Control Register (ADD_DEL_CONTROL)

The Add/Delete control register is used to initiate an add or delete operation into the table.

Table 4-16 Add/Delete Control Register (ADD_DEL_CONTROL)

Bits	Field	Type	Reset	Description
31	GO	RW	0h	Go bit. Writing this bit will cause the add/delete operation to start. Once started, the operation cannot be cancelled. This bit will remain asserted until the add or delete operation completes.
30	ADD_DEL	RW	0h	Add/delete control bit. This bit is encoded as follows: 1 = Add entry 0 = Delete entry
29-0	Reserved	R	0h	Reserved.
End of Table 4-16				

4.4 PDSP Control/Status Register Region

This section describes the PDSP Control/Status Registers available in the packet accelerator (PA). There are 6 PDSPs in the PA, each with its own set of identical control/status registers. The register address offsets listed in Table 4-17 are relative to the PDSP Control/Status memory region. To determine the base address of the PDSP Control/Status memory region for each PDSP relative to the PA memory map, please see Table 4-1.

The PDSP control/status registers region contains control and status registers for the PDSP. The control/status registers region memory map is as follows:

Table 4-17 PDSP Control/Status Register Region

Address Offset	Register	Section
00h	PDSP Control Register	Section 4.4.1
04h	PDSP Status Register	Section 4.4.2
08h	PDSP Wakeup Enable Register	Section 4.4.3
0Ch	PDSP Cycle Count	Section 4.4.4
10h	PDSP Stall Count	Section 4.4.5
20h	PDSP Constant Table Block Index Register 0	Section 4.4.6
24h	PDSP Constant Table Block Index Register 1	Section 4.4.7
28h	PDSP Constant Table Programmable Pointer Register 0	Section 4.4.8
2Ch	PDSP Constant Table Programmable Pointer Register 1	Section 4.4.9
30h-FFh	Reserved	Reserved
End of Table 4-17		

4.4.1 PDSP Control Register (PDSP_CONTROL)

Table 4-18 PDSP Control Register (PDSP_CONTROL) (Part 1 of 2)

Bits	Field	Type	Reset	Description
31-16	PCOUNTER_RST_VAL	RW	0h	Program Counter Reset Value. This field controls the address where the PDSP will start executing code from after it is taken out of reset*
15	PDSP_STATE	R	0h	Run State. This bit indicates whether the PDSP is currently executing an instruction or is halted. 0 = PDSP is halted and host has access to the instruction RAM and debug registers regions. 1 = PDSP is currently running and the host is locked out of the instruction RAM and debug registers regions This bit is used by an external debug agent to know when the PDSP has actually halted when waiting for a HALT instruction to execute, a single step to finish, or any other time when the PDSP_ENABLE has been cleared.
14-8	Reserved	R	0h	Reserved.
7-4	Reserved	R	0h	Reserved.
3	COUNTER_ENABLE	RW	0h	PDSP Cycle Counter Enable. Enables PDSP cycle counters 0 = Counters not enabled 1 = Counters enabled
2	PDSP_SLEEPING	RW	0h	PDSP Sleep Indicator. This bit indicates whether or not the PDSP is currently asleep. 0 = PDSP is not asleep 1 = PDSP is asleep If this bit is written to a 0, the PDSP will be forced to power up from sleep mode.

Table 4-18 PDSP Control Register (PDSP_CONTROL) (Part 2 of 2)

Bits	Field	Type	Reset	Description
1	PDSP_ENABLE	RW	0h	Processor Enable. This bit controls whether or not the PDSP is allowed to fetch new instructions 0 = PDSP is disabled 1 = PDSP is enabled If this bit is de-asserted while the PDSP is currently running and has completed the initial cycle of a multi-cycle instruction (LBxO, SBxO, SCAN, etc.), the current instruction will be allowed to complete before the PDSP pauses execution. Otherwise, the PDSP will halt immediately. Because of the unpredictability/timing sensitivity of the instruction execution loop, this bit is not a reliable indication of whether or not the PDSP is currently running. The PDSP_STATE bit should be consulted for an absolute indication of the run state of the core. When the PDSP is halted, it's internal state remains coherent therefore this bit can be reasserted without issuing a software reset and the PDSP will resume processing exactly where it left off in the instruction stream.
0	SOFT_RST_N	R	0h => 1h	Soft Reset. When this bit is cleared, the PDSP will be reset. This bit is set back to 1 on the next cycle after it has been cleared.
End of Table 4-18				

4.4.2 PDSP Status Register

This register stores the PDSP program counter delayed by 1 cycle.

Table 4-19 PDSP Status Register (PDSP_STATUS)

Bits	Field	Type	Reset	Description
31-16	Reserved	R	0h	Reserved.
15-0	PCOUNTER	R	0h	Program Counter. This field is a registered (1 cycle delayed) reflection of the PDSP program counter[1]
End of Table 4-19				

1. The PC is an instruction address where each instruction is a 32 bit word. This is not a byte address and to compute the byte address just multiply the PC by 4 (PC of 2 = byte address of 8h, or PC of 8 = byte address of 20h).

4.4.3 PDSP Wakeup Enable Register (PDSP_WAKE_EN)

This register allows the status input from R31 to wakeup the core.

Table 4-20 PDSP Wakeup Enable Register (PDSP_WAKE_EN)

Bits	Field	Type	Reset	Description
31-0	BITWISE_ENABLES	RW	0h	Wakeup Enables. This field is ANDed with the incoming R31 status inputs (whose bit positions were specified in the STMAP parameter) to produce a vector which is unary ORed to produce the STATUS_WAKEUP source for the core. Setting any bit in this vector will allow the corresponding status input to wake up the core when it is asserted high. The PDSP should set this enable vector prior to executing a SLP (sleep) instruction to ensure that the desired sources can wake up the core.
End of Table 4-20				

4.4.4 PDSP Cycle Count Register (PDSP_CYCLECOUNT)

This register counts the number of cycles for which the PDSP has been enabled.

Table 4-21 PDSP Cycle Count Register (PDSP_CYCLECOUNT)

Bits	Field	Type	Reset	Description
31-0	CYCLECOUNT	RW	0h	This value is incremented by 1 for every cycle during which the PDSP is enabled and the counter is enabled (both bits PDSP_ENABLE and COUNTER_ENABLE set in the PDSP control register). Counting halts while the PDSP is disabled or counter is disabled, and resumes when re-eneabled. Counter clears the "counter enable" bit in the PDSP control register when the count reaches FFFFFFFFh. (Count does not wrap). The register can be read at any time. The register can be cleared when the counter or PDSP is disabled. Clearing this register also clears the PDSP Stall Count Register.
End of Table 4-21				

4.4.5 PDSP Stall Count Register (PDSP_STALLCOUNT)

This register counts the number of cycles for which the PDSP has been enabled, but unable to fetch a new instruction. It is linked to the Cycle Count Register (0Ch) such that this register reflects the stall cycles measured over the same cycles as counted by the cycle count register. Thus the value of this register is always less than or equal to cycle count.

Table 4-22 PDSP Stall Count Register (PDSP_STALLCOUNT)

Bits	Field	Type	Reset	Description
31-0	STALLCOUNT	R	0h	This value is incremented by 1 for every cycle during which the PDSP is enabled and the counter is enabled (both bits PDSP_ENABLE and COUNTER_ENABLE set in the PDSP control register), and the PDSP was unable to fetch a new instruction for any reason. Counting halts while the PDSP is disabled or the counter is disabled, and resumes when re-eneabled. The register can be read at any time. The register is cleared when PDSP Cycle Count Register is cleared.
End of Table 4-22				

4.4.6 PDSP Constant Table Block Index Register 0 (PDSP_BLK_IDX0)

This register is used to set the block indices which are used to modify entries 24 and 25 in the PDSP Constant Table. This register can be written by the PDSP whenever it needs to change to a new base pointer for a block in the State/Scratchpad RAM. This function is useful since the PDSP is often processing channelized streams which require it to change contexts. The PDSP can use this register to avoid requiring excessive amounts of code for repetitive channelized functions. The format of this register is as follows:

Table 4-23 PDSP Constant Table Block Index Register 0 (PDSP_BLK_IDX0)

Bits	Field	Type	Reset	Description
31-20	Reserved	R	0h	Reserved.
19-16	C25_BLK_INDEX	RW	0h	PDSP Constant Entry 25 Block Index. This field sets the value that will appear in bits 11-8 of entry 25 in the PDSP Constant Table.
15-4	Reserved	R	0h	Reserved.
3-0	C24_BLK_INDEX	RW	0h	PDSP Constant Entry 24 Block Index. This field sets the value that will appear in bits 11-8 of entry 24 in the PDSP Constant Table.
End of Table 4-23				

4.4.7 PDSP Constant Table Block Index Register 1 (PDSP_BLK_IDX1)

This register functions the same as the PDSP Constant Table Block Index Register 1 but controls entries 26 and 27 in the PDSP Constant Table. This register is formatted as follows:

Table 4-24 PDSP Constant Table Block Index Register 1 (PDSP_BLK_IDX1)

Bits	Field	Type	Reset	Description
31-20	Reserved	R	0h	Reserved.
19-16	C27_BLK_INDEX	RW	0h	PDSP Constant Entry 27 Block Index. This field sets the value that will appear in bits 11-8 of entry 27 in the PDSP Constant Table.
15-4	Reserved	R	0h	Reserved.
3-0	C26_BLK_INDEX	RW	0h	PDSP Constant Entry 26 Block Index. This field sets the value that will appear in bits 11-8 of entry 26 in the PDSP Constant Table.
End of Table 4-24				

4.4.8 PDSP Constant Table Programmable Pointer Register 0 (PDSP_POINTER0)

This register allows the PDSP to set up the 256-byte page index for entries 28 and 29 in the PDSP Constant Table which serve as general purpose pointers which can be configured to point to any locations inside the session router address map. This register is useful when the PDSP needs to frequently access certain structures inside the session router address space whose locations are not hard coded such as tables in scratchpad memory. This register is formatted as follows:

Table 4-25 PDSP Constant Table Programmable Pointer Register 0 (PDSP_POINTER0)

Bits	Field	Type	Reset	Description
31-16	C29_POINTER	RW	0h	PDSP Constant Entry 29 Pointer. This field sets the value that will appear in bits 23-8 of entry 29 in the PDSP Constant Table
15-0	C28_POINTER	RW	0h	PDSP Constant Entry 28 Pointer. This field sets the value that will appear in bits 23-8 of entry 28 in the PDSP Constant Table
End of Table 4-25				

4.4.9 PDSP Constant Table Programmable Pointer Register 1 (PDSP_POINTER1)

This register functions the same as the PDSP Constant Table Programmable Pointer Register 0 but allows the PDSP to control entries 30 and 31 in the PDSP Constant Table. This register is formatted as follows:

Table 4-26 PDSP Constant Table Programmable Pointer Register 1 (PDSP_POINTER1)

Bits	Field	Type	Reset	Description
31-16	C31_POINTER	RW	0h	PDSP Constant Entry 29 Pointer. This field sets the value that will appear in bits 23-8 of entry 31 in the PDSP Constant Table
15-0	C30_POINTER	RW	0h	PDSP Constant Entry 28 Pointer. This field sets the value that will appear in bits 23-8 of entry 30 in the PDSP Constant Table
End of Table 4-26				

4.5 PDSP Timer Registers

This section describes the PDSP timer registers available in the packet accelerator (PA). There are 6 PDSP timers in the PA, each with its own set of identical registers. The register address offsets listed in Table 4-27 are relative to the PDSP timer memory region. To determine the base address of each PDSP timer memory region relative to the PA memory map, please see Table 4-1.

The timers are used for various timing tasks within the PA.

Table 4-27 PDSP Timer Registers

Address Offset	Register	Section
00h	Control Timer Register	Section 4.5.1
04h	Load Timer Register	Section 4.5.2
08h	Value Timer Register	Section 4.5.3
0Ch	Timer Interrupt Register	Section 4.5.4
10h-FF	Reserved	Reserved
End of Table 4-27		

4.5.1 Control Timer Register (CNTL_TIM)

The Control Timer Register controls the 16-bit count down timer module. Bit field descriptions for the Control Timer Register can be found in Table 4-28.

Table 4-28 Control Timer Register (CNT_TIM) (Part 1 of 2)

Bits	Field	Type	Reset	Description
31-16	Reserved	R	0h	Reserved.
15	PRESCALE_EN	RW	0h	Prescaler Enable. 0 = Prescaler Disabled 1 = Prescaler Enabled
14-6	Reserved	R	0h	Reserved.
5-2	PRESCALE	RW	Fh	Timer Prescale Value. These bits are a power-of-two prescaler values. The timer divides the frequency of the timer reference clock by a factor of 2 to 8192. 0000 = Divide by 2 0001 = Divide by 4 0010 = Divide by 8 0011 = Divide by 16 0100 = Divide by 32 0101 = Divide by 64 0110 = Divide by 128 0111 = Divide by 256 1000 = Divide by 512 1001 = Divide by 1024 1010 = Divide by 2048 1011 = Divide by 4096 1100 = Divide by 8192 1101-1111 = Disable timer

Table 4-28 Control Timer Register (CNT_TIM) (Part 2 of 2)

Bits	Field	Type	Reset	Description
1	MODE	RW	0h	Timer Mode. 0 = Sets the timer in one shot mode. After the counter expires, the START bit is set to 0, stopping the counter. 1 = Sets the timer in auto-load mode. After the counter expires, the counter is reloaded with the value in the LOAD_TIM register and the timer resumes counting.
0	START	RW	0h	Timer Start. 0 = Stops the 16-bit timer 1 = Starts the 16-bit timer
End of Table 4-28				

4.5.2 Load Timer Register (LOAD_TIM)

The Load Timer Register contains the 16-bit count down value for timer module. Bit field descriptions for the Load Timer Register can be found in Table 4-29.

Table 4-29 Load Timer Register (LOAD_TIM)

Bits	Field	Type	Reset	Description
31-16	Reserved	R	0h	Reserved.
15-0	LOAD_TIM	RW	0h	Load Timer Value. This field is the 16-bit count down value for the timer module.
End of Table 4-29				

4.5.3 Value Timer Register (VALUE_TIM)

The Value Timer Register contains the current 16-bit count value of the timer. Bit field descriptions for the Value Timer Register can be found in Table 4-30.

Table 4-30 Value Timer Register (VALUE_TIM)

Bits	Field	Type	Reset	Description
31-16	Reserved	R	0h	Reserved.
15-0	VALUE	R	0h	Current Timer Value. This field contains the current 16-bit count value of the timer.
End of Table 4-30				

4.5.4 Timer Interrupt Register (TIMER_IRQ)

The Timer Interrupt Register indicates the registered interrupt when the 16-bit count down timer expires. Bit field descriptions for the Timer Interrupt Register can be found in Table 4-29.

Table 4-31 Timer Interrupt Register (TIMER_IRQ)

Bits	Field	Type	Reset	Description
31-16	Reserved	R	0h	Reserved.
15-1	Reserved	R	0h	Reserved.
0	TIMER_IRQ	RW	0h	Timer Interrupt. Indicates the registered interrupt when the 16-bit counter expires. I is also used as a software mechanism for clearing the interrupt. Reading a 0 = the interrupt is inactive Reading a 1 = the interrupt is active Writing a 0 = no effect Writing a 1 = clears the active interrupt
End of Table 4-31				

4.6 Packet Accelerator Statistics Register Region

This section describes the packet accelerator statistics registers available in the Packet Accelerator (PA). The register address offsets listed in Table 4-32 are relative to the packet accelerator statistics memory region. To determine the base address of the packet accelerator statistics memory region relative to the PA memory map, please see Table 4-1.

Table 4-32 Packet Accelerator Statistics Register Region Memory Map

Address Offset	Register	Section
00h	Revision Register	Section 4.6.1
04h	Soft Reset Register	Section 4.6.2
08h	Increment Flags Register	Section 4.6.3
0Ch	Statistics Capture Register	Section 4.6.4
20h	Statistic 0 Register	Section 4.6.5
24h	Statistic 1 Register	Section 4.6.5
...	...	...
20h + N*4	Statistic N Register	Section 4.6.5
...	...	...
98h	Statistic 30 Register	Section 4.6.5
9Ch	Statistic 31 Register	Section 4.6.5
A0h-FFh	Reserved	Reserved
End of Table 4-32		

4.6.1 Revision Register

The Revision Register contains the major and minor revisions for the module.

Table 4-33 Revision Register

Bits	Field	Type	Reset	Description
31-30	Reserved	R	0h	Reserved.
29-16	STATS_MODID	R	TBD	Module ID field.
15-11	STATS_REVRTL	R	0h	RTL revision. Will vary depending on release.
10-8	STATS_REVMAJ	R	1h	Major revision.
7-0	STATS_REVMIN	R	0h	Minor revision.
End of Table 4-33				

4.6.2 Soft Reset Register

The Soft Reset Register is written in order to clear the contents of all statistics.

Table 4-34 Soft Reset Register

Bits	Field	Type	Reset	Description
31-0	STATS_TRIGGER	W	0h	Writing anything to this field causes all internal statistics to be cleared.
End of Table 4-34				

4.6.3 Increment Flags Register

Writing the unit Increment Flags register causes the increment process to begin. For each flag set in this register, the corresponding statistic is increment by one.

Table 4-35 Increment Flags Register

Bits	Field	Type	Reset	Description
31-0	INC_FLAGS	W	0h	Each flag corresponds to a statistic to increment. Bit 0 corresponds to the Statistic 0 register. The amount to increment is 1.
End of Table 4-35				

4.6.4 Statistics Capture Register

Writing the unit Statistics Capture register causes the values of all 32 statistics to be captured and written to the statistics register. In addition, for all of the flags in the value that are set to 1, the corresponding internal statistic value is cleared after the capture.

Table 4-36 Statistics Capture Register

Bits	Field	Type	Reset	Description
31-0	STATS_CLR_ FLAGS	W	0h	Each flag corresponds to a statistic. Bit 0 corresponds to the Statistic 0 register. Writing any value to this register causes all 32 statistic values to be captured. For any bit that is set to 1, the corresponding statistic is also cleared after being captured.
End of Table 4-36				

4.6.5 Statistic N Register

The statistics register contains the captured statistic count for each statistic in the block. These values remain static and are only updated when a capture is initiated by writing the statistics capture register.

Table 4-37 Statistics N Register

Bits	Field	Type	Reset	Description
31-0	STATS_REG	R	0h	Holds the 32-bit value of the statistic.
End of Table 4-37				

Index

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license from TI to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of TI information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

TI products are not authorized for use in safety-critical applications (such as life support) where a failure of the TI product would reasonably be expected to cause severe personal injury or death, unless officers of the parties have executed an agreement specifically governing such use. Buyers represent that they have all necessary expertise in the safety and regulatory ramifications of their applications, and acknowledge and agree that they are solely responsible for all legal, regulatory and safety-related requirements concerning their products and any use of TI products in such safety-critical applications, notwithstanding any applications-related information or support that may be provided by TI. Further, Buyers must fully indemnify TI and its representatives against any damages arising out of the use of TI products in such safety-critical applications.

TI products are neither designed nor intended for use in military/aerospace applications or environments unless the TI products are specifically designated by TI as military-grade or "enhanced plastic." Only products designated by TI as military-grade meet military specifications. Buyers acknowledge and agree that any such use of TI products which TI has not designated as military-grade is solely at the Buyer's risk, and that they are solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI products are neither designed nor intended for use in automotive applications or environments unless the specific TI products are designated by TI as compliant with ISO/TS 16949 requirements. Buyers acknowledge and agree that, if they use any non-designated products in automotive applications, TI will not be responsible for any failure to meet such requirements.

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

Products		**Applications**	
Audio	www.ti.com/audio	Automotive and Transportation	www.ti.com/automotive
Amplifiers	amplifier.ti.com	Communications and Telecom	www.ti.com/communications
Data Converters	dataconverter.ti.com	Computers and Peripherals	www.ti.com/computers
DLP® Products	www.dlp.com	Consumer Electronics	www.ti.com/consumer-apps
DSP	dsp.ti.com	Energy and Lighting	www.ti.com/energy
Clocks and Timers	www.ti.com/clocks	Industrial	www.ti.com/industrial
Interface	interface.ti.com	Medical	www.ti.com/medical
Logic	logic.ti.com	Security	www.ti.com/security
Power Mgmt	power.ti.com	Space, Avionics and Defense	www.ti.com/space-avionics-defense
Microcontrollers	microcontroller.ti.com	Video and Imaging	www.ti.com/video
RFID	www.ti-rfid.com		
OMAP Mobile Processors	www.ti.com/omap		
Wireless Connectivity	www.ti.com/wirelessconnectivity		
	TI E2E Community Home Page		e2e.ti.com

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265