TMS320TCI648x DSP
Peripheral Component Interconnect (PCI)

User's Guide

Literature Number: SPRUE69B
March 2006–Revised July 2010

Preface **9**
1 Overview **10**
2 PCI Architecture **12**
2.1 Address Decoder 12
2.2 Slave State Machine 12
2.3 Slave Back End PCI Interface 13
2.4 Master Back End PCI Interface 13
2.5 Master State Machine 13
2.6 Output Multiplexer 13
2.7 Configuration Registers 13
2.8 Error Handler 13
2.9 Back End Registers Interface 13
3 PCI Signals **14**
3.1 PCI Pin Description 14
3.2 Connecting a Local PCI to an External PCI Device 15
4 Clocks **17**
5 Byte Addressing **18**
6 Slave Memory Map **19**
6.1 Configuring Slave Window Registers 19
6.2 Slave Access Address Translations 20
7 Slave Operations **22**
7.1 Slave Configuration Operations 22
7.2 Slave Memory Operations 23
8 Master Memory Map **26**
8.1 Configuring Master Windows Using Address Substitution Registers 0 to 31 (PCIADDSUB*n*) 27
8.2 Master Address Translation 27
9 Master Operations **30**
9.1 Master Configuration Operations 30
9.2 Master I/O Operations 31
9.3 Master Memory Operations 32
10 Exceptions, Status Reporting, and Interrupts **33**
10.1 PCI Exceptions 33
10.2 Status Reporting 34
10.3 PCI Interrupts 35
11 PCI Reset Information **37**
11.1 PCI Pin Reset 37
11.2 C64x+ Megamodule Local Reset 37
11.3 PCI Register Reset Values 37
12 PCI Configuration **38**
12.1 Programming the PCI Configuration Space Registers 38
12.2 Programming the PCI Back End Registers 38
12.3 PCI Configuration Hook Registers 39

12.4 PCI I2C EEPROM Autoinitialization 39
13 PCI Registers 42
13.1 PCI Configuration Registers 45
13.2 PCI Back End Configuration Registers 59
13.3 DSP-To-PCI Address Translation Registers 94
13.4 PCI Hook Configuration Registers 95
Appendix A Revision History 121

List of Figures

1 PCI Block Diagram 12
2 PCI Signals 14
3 PCI to External PCI device 15
4 Slave Window Configuration 19
5 PCI-to-DSP Address Translation 21
6 Master Window Configuration 27
7 PCI Address Substitution Register (0 to 31) 27
8 DSP-to-PCI Address Translation 28
9 Example of DSP-to-PCI Address Translation 28
10 Signal Connections for I2C EEPROM Boot Mode 41
11 Vendor ID/Device ID Register (PCIVENDEV) 45
12 Command/Status Register (PCICSR) 46
13 Class Code/Revision ID Register (PCICLREV) 48
14 BIST/Header Type/Latency Timer/Cacheline Size Register (PCICLINE) 49
15 Base Address Register 0 (PCIBAR0) 50
16 Base Address Register 1 (PCIBAR1) 51
17 Base Address Register 2 (PCIBAR2) 52
18 Base Address Register 3 (PCIBAR3) 53
19 Base Address Register 4 (PCIBAR4) 54
20 Base Address Register 5 (PCIBAR5) 55
21 Subsystem Vendor ID/Subsystem ID Register (PCISUBID) 56
22 Capabilities Pointer Register (PCICPBPTR) 57
23 Max Latency/Min Grant/Interrupt Pin/Interrupt Line Register (PCILGINT) 58
24 Status Set Register (PCISTATSET) 59
25 Status Clear Register (PCISTATCLR) 61
26 Host Interrupt Enable Set Register (PCIHINTSET) 62
27 Host Interrupt Enable Clear Register (PCIHINTCLR) 63
28 Back End Application Interrupt Enable Set Register (PCIBINTSET) 64
29 Back End Application Interrupt Enable Clear Register (PCIBINTCLR) 66
30 Vendor ID/Device ID Mirror Register (PCIVENDEVMIR) 67
31 Command/Status Mirror Register (PCICSRMIR) 68
32 Class Code/Revision ID Mirror Register (PCICLREVMIR) 70
33 BIST/Header Type/Latency Timer/Cacheline Size Mirror Register (PCICLINEMIR) 71
34 Base Address Mask Register 0 (PCIBAR0MSK) 72
35 Base Address Mask Register 1 (PCIBAR1MSK) 73
36 Base Address Mask Register 2 (PCIBAR2MSK) 74
37 Base Address Mask Register 3 (PCIBAR3MSK) 75
38 Base Address Mask Register 4 (PCIBAR4MSK) 76
39 Base Address Mask Register 5 (PCIBAR5MSK) 77
40 Subsystem Vendor ID/Subsystem ID Mirror Register (PCISUBIDMIR) 78
41 Capabilities Pointer Mirror Register (PCICPBPTRMIR) 79
42 Max Latency/Min Grant/Interrupt Pin/Interrupt Line Mirror Register (PCILGINTMIR) 80
43 Slave Control Register (PCISLVCNTL) 81
44 Slave Base Address 0 Translation Register (PCIBAR0TRL) 83
45 Slave Base Address 1 Translation Register (PCIBAR1TRL) 84
46 Slave Base Address 2 Translation Register (PCIBAR2TRL) 85
47 Slave Base Address 3 Translation Register (PCIBAR3TRL) 86

48 Slave Base Address 4 Translation Register (PCIBAR4TRL) 87
49 Slave Base Address 5 Translation Register (PCIBAR5TRL) 88
50 Base Address *n* Mirror Register (PCIBAR*n*MIR) 89
51 Master Configuration/IO Access Data Register (PCIMCFGDAT) 90
52 Master Configuration/IO Access Address Register (PCIMCFGADR) 91
53 Master Configuration/IO Access Command Register (PCIMCFGCMD) 92
54 Master Configuration Register (PCIMSTCFG) 93
55 PCI Address Substitution *n* Register (PCIADDSUB*n*) 94
56 PCI Vendor ID and Device ID Program Register (PCIVENDEVPRG) 95
57 PCI Command and Status Program Register (PCICMDSTATPRG) 96
58 PCI Class Code and Revision ID Program Register (PCICLREVPRG) 97
59 PCI Subsystem Vendor ID and Subsystem ID Program Register (PCISUBIDPRG) 98
60 Max Latency and Min Grant Program Register (PCIMAXLGPRG) 99
61 LRESET Register (PCILRSTREG) 100
62 Configuration Done Register (PCICFGDONE) 101
63 Base Address Mask Register 0 Program Register (PCIBAR0MPRG) 102
64 Base Address Mask Register 1 Program Register (PCIBAR1MPRG) 103
65 Base Address Mask Register 2 Program Register (PCIBAR2MPRG) 104
66 Base Address Mask Register 3 Program Register (PCIBAR3MPRG) 105
67 Base Address Mask Register 4 Program Register (PCIBAR4MPRG) 106
68 Base Address Mask Register 5 Program Register (PCIBAR5MPRG) 107
69 Base Address Register 0 Program Register (PCIBAR0PRG) 108
70 Base Address Register 1 Program Register (PCIBAR1PRG) 109
71 Base Address Register 2 Program Register (PCIBAR2PRG) 110
72 Base Address Register 3 Program Register (PCIBAR3PRG) 111
73 Base Address Register 4 Program Register (PCIBAR4PRG) 112
74 Base Address Register 5 Program Register (PCIBAR5PRG) 113
75 Base Address Translation Register 0 Program Register (PCIBAR0TRLPRG) 114
76 Base Address Translation Register 1 Program Register (PCIBAR1TRLPRG) 115
77 Base Address Translation Register 2 Program Register (PCIBAR2TRLPRG) 116
78 Base Address Translation Register 3 Program Register (PCIBAR3TRLPRG) 117
79 Base Address Translation Register 4 Program Register (PCIBAR4TRLPRG) 118
80 Base Address Translation Register 5 Program Register (PCIBAR5TRLPRG) 119
81 Base Enable Program Register (PCIBASENPRG) 120

List of Tables

1 PCI Pin Description 14
2 PCI Base Addresses 19
3 PCI Master Windows 26
4 Byte Enables and AD[1:0] Encodings 31
5 PCI Exceptions 33
6 PCI Interrupts 35
7 Back End PCI Configuration Registers 39
8 I2C EEPROM Memory Layout 40
9 PCI Registers 42
10 Bit Field Type Encodings 44
11 Vendor ID/Device ID Register (PCIVENDEV) Field Descriptions 45
12 Command/Status Register (PCICSR) Field Descriptions 46
13 Class Code/Revision ID Register (PCICLREV) Field Descriptions 48
14 BIST/Header Type/Latency Timer/Cacheline Size Register (PCICLINE) Field Descriptions 49
15 Base Address Register 0 (PCIBAR0) Field Descriptions 50
16 Base Address Register 1 (PCIBAR1) Field Descriptions 51
17 Base Address Register 2 (PCIBAR2) Field Descriptions 52
18 Base Address Register 3 (PCIBAR3) Field Descriptions 53
19 Base Address Register 4 (PCIBAR4) Field Descriptions 54
20 Base Address Register 5 (PCIBAR5) Field Descriptions 55
21 Subsystem Vendor ID/Subsystem ID Register (PCISUBID) Field Descriptions 56
22 Capabilities Pointer Register (PCICPBPTR) Field Descriptions 57
23 Max Latency/Min Grant/Interrupt Pin/Interrupt Line Register (PCILGINT) Field Descriptions 58
24 Status Set Register (PCISTATSET) Field Descriptions 59
25 Status Clear Register (PCISTATCLR) Field Descriptions 61
26 Host Interrupt Enable Set Register (PCIHINTSET) Field Descriptions 62
27 Host Interrupt Enable Clear Register (PCIHINTCLR) Field Descriptions 63
28 Back End Application Interrupt Enable Set Register (PCIBINTSET) Field Descriptions 64
29 Back End Application Interrupt Enable Clear Register (PCIBINTCLR) Field Descriptions 66
30 Vendor ID/Device ID Mirror Register (PCIVENDEVMIR) Field Descriptions 67
31 Command/Status Mirror Register (PCICSRMIR) Field Descriptions 68
32 Class Code/Revision ID Mirror Register (PCICLREVMIR) Field Descriptions 70
33 BIST/Header Type/Latency Timer/Cacheline Size Mirror Register (PCICLINEMIR) Field Descriptions 71
34 Base Address Mask Register 0 (PCIBAR0MSK) Field Descriptions 72
35 Base Address Mask Register 1 (PCIBAR1MSK) Field Descriptions 73
36 Base Address Mask Register 2 (PCIBAR2MSK) Field Descriptions 74
37 Base Address Mask Register 3 (PCIBAR3MSK) Field Descriptions 75
38 Base Address Mask Register 4 (PCIBAR4MSK) Field Descriptions 76
39 Base Address Mask Register 5 (PCIBAR5MSK) Field Descriptions 77
40 Subsystem Vendor ID/Subsystem ID Mirror Register (PCISUBIDMIR) Field Descriptions 78
41 Capabilities Pointer Mirror Register (PCICPBPTRMIR) Field Descriptions 79
42 Max Latency/Min Grant/Interrupt Pin/Interrupt Line Mirror Register (PCILGINTMIR) Field Descriptions 80
43 Slave Control Register (PCISLVCNTL) Field Descriptions 81
44 Slave Base Address 0 Translation Register (PCIBAR0TRL) Field Descriptions 83
45 Slave Base Address 1 Translation Register (PCIBAR1TRL) Field Descriptions 84
46 Slave Base Address 2 Translation Register (PCIBAR2TRL) Field Descriptions 85
47 Slave Base Address 3 Translation Register (PCIBAR3TRL) Field Descriptions 86

48 Slave Base Address 4 Translation Register (PCIBAR4TRL) Field Descriptions 87
49 Slave Base Address 5 Translation Register (PCIBAR5TRL) Field Descriptions 88
50 Base Address *n* Mirror Register (PCIBAR*n*MIR) Field Descriptions 89
51 Master Configuration/IO Access Data Register (PCIMCFGDAT) Field Descriptions 90
52 Master Configuration/IO Access Address Register (PCIMCFGADR) Field Descriptions 91
53 Master Configuration/IO Access Command Register (PCIMCFGCMD) Field Descriptions 92
54 Master Configuration Register (PCIMSTCFG) Field Descriptions 93
55 PCI Address Substitution *n* Register (PCIADDSUB*n*) Field Descriptions 94
56 PCI Vendor ID and Device ID Program Register (PCIVENDEVPRG) Field Descriptions 95
57 PCI Command and Status Program Register (PCICMDSTATPRG) Field Descriptions 96
58 PCI Class Code and Revision ID Program Register (PCICLREVPRG) Field Descriptions 97
59 PCI Subsystem Vendor ID and Subsystem ID Program Register (PCISUBIDPRG) Field Descriptions 98
60 Max Latency and Min Grant Program Register (PCIMAXLGPRG) Field Descriptions 99
61 LRESET Register (PCILRSTREG) Field Descriptions 100
62 Configuration Done Register (PCICFGDONE) Field Descriptions 101
63 Base Address Mask Register 0 Program Register (PCIBAR0MPRG) Field Descriptions 102
64 Base Address Mask Register 1 Program Register (PCIBAR1MPRG) Field Descriptions 103
65 Base Address Mask Register 2 Program Register (PCIBAR2MPRG) Field Descriptions 104
66 Base Address Mask Register 3 Program Register (PCIBAR3MPRG) Field Descriptions 105
67 Base Address Mask Register 4 Program Register (PCIBAR4MPRG) Field Descriptions 106
68 Base Address Mask Register 5 Program Register (PCIBAR5MPRG) Field Descriptions 107
69 Base Address Register 0 Program Register (PCIBAR0PRG) Field Descriptions 108
70 Base Address Register 1 Program Register (PCIBAR1PRG) Field Descriptions 109
71 Base Address Register 2 Program Register (PCIBAR2PRG) Field Descriptions 110
72 Base Address Register 3 Program Register (PCIBAR3PRG) Field Descriptions 111
73 Base Address Register 4 Program Register (PCIBAR4PRG) Field Descriptions 112
74 Base Address Register 5 Program Register (PCIBAR5PRG) Field Descriptions 113
75 Base Address Translation Register 0 Program Register (PCIBAR0TRLPRG) Field Descriptions 114
76 Base Address Translation Register 1 Program Register (PCIBAR1TRLPRG) Field Descriptions 115
77 Base Address Translation Register 2 Program Register (PCIBAR2TRLPRG) Field Descriptions 116
78 Base Address Translation Register 3 Program Register (PCIBAR3TRLPRG) Field Descriptions 117
79 Base Address Translation Register 4 Program Register (PCIBAR4TRLPRG) Field Descriptions 118
80 Base Address Translation Register 5 Program Register (PCIBAR5TRLPRG) Field Descriptions 119
81 Base Enable Program Register (PCIBASENPRG) Field Descriptions 120

Read This First

About This Manual

This document describes the peripheral component interconnect (PCI) module in TMS320TCI648x devices. For details on the PCI interface, see the PCI Specification revision 2.3.

Notational Conventions

This document uses the following conventions.

- Hexadecimal numbers are shown with the suffix h. For example, the following number is 40 hexadecimal (decimal 64): 40h.
- Registers in this document are shown in figures and described in tables.
  - Each register figure shows a rectangle divided into fields that represent the fields of the register. Each field is labeled with its bit name, its beginning and ending bit numbers above, and its read/write properties below. A legend explains the notation used for the properties.
  - Reserved bits in a register figure designate a bit that is used for future device expansion.

Related Documentation From Texas Instruments

The following documents describe the TMS320C6000™ devices and related support tools. Copies of these documents are available on the Internet at www.ti.com. *Tip:* Enter the literature number in the search box provided at www.ti.com.

SPRU189 — *TMS320C6000 DSP CPU and Instruction Set Reference Guide.* Describes the CPU architecture, pipeline, instruction set, and interrupts for the TMS320C6000 digital signal processors (DSPs).

SPRU198 — *TMS320C6000 Programmer's Guide.* Describes ways to optimize C and assembly code for the TMS320C6000™ DSPs and includes application program examples.

SPRU301 — *TMS320C6000 Code Composer Studio Tutorial.* Introduces the Code Composer Studio™ integrated development environment and software tools.

SPRU321 — *Code Composer Studio Application Programming Interface Reference Guide.* Describes the Code Composer Studio™ application programming interface (API), which allows you to program custom plug-ins for Code Composer.

SPRU871 — *TMS320C64x+ Megamodule Reference Guide.* Describes the TMS320C64x+ digital signal processor (DSP) megamodule. Included is a discussion on the internal direct memory access (IDMA) controller, the interrupt controller, the power-down controller, memory protection, bandwidth management, and the memory and cache.

TCI648x DSP Peripheral Component Interconnect (PCI)

1 Overview

The PCI module supports the following features:

- PCI Local Bus Specification (revision 2.3) compliant
- Single function PCI interface provided
- 32-bit address/data bus width
- Operation up to 66 MHz
- Optimized burst behavior supported for system cache line sizes of 16, 32, 64 and 128 bytes

The PCI operates as a PCI slave device for configuration cycles and memory cycles. It also acts as a PCI master device for configuration cycles, IO cycles, and memory accesses to other devices.

As a slave, the PCI includes the following features:

- Response to accesses as a 32-bit agent with medium DEVSEL_N timing (single wait state)
- Direct support of the Memory Read, Memory Read Multiple, Memory Read Line, Memory Write, Configuration Read and Configuration Write transactions
- Aliases Memory Write and Invalidate to the Memory Write command
- Support of variable length burst transfers up to a cache line for Memory Read Line transactions
- Support of unlimited length burst transfers for Memory Read Multiple and Memory Write transactions
- Support of single data phase transfers with disconnect for Memory Read, Configuration Read and Configuration Write transactions
- Support of both immediate or timeout forced delayed transactions for Memory Read, Memory Read Line, and Memory Read Multiple transactions
- Support of posting of Memory Write transactions
- Support of up to six base address registers (BAR0-BAR5)
- Support of programmable cache line size of 4, 8, 16, 32, 64, or 128 bytes
- Ports provided to set configuration space registers to specific values after reset

As a master, the PCI includes the following features:

- Transaction initiation as a 32-bit agent
- Support of the Configuration Read, Configuration Write, IO Read, IO Write, Memory Read, Memory Read Line, Memory Read Multiple, Memory Write, and Memory Write and Invalidate PCI Bus commands
- Support of bursts transfers of up to 256 data phases for Memory Read Line, Memory Read Multiple, and Memory Write transactions
- Support of single data phase transfers for Memory Read transactions
- Automatic selection between Memory Read, Memory Read Line, and Memory Read Multiple based on the requested transaction length and the cache line size
- Assertion of the $\overline{\text{PIRDY}}$ signal one clock cycle after the $\overline{\text{PFRAME}}$ signal is asserted. For optimal performance, it does not insert wait states during a burst.

The PCI Module does not support:

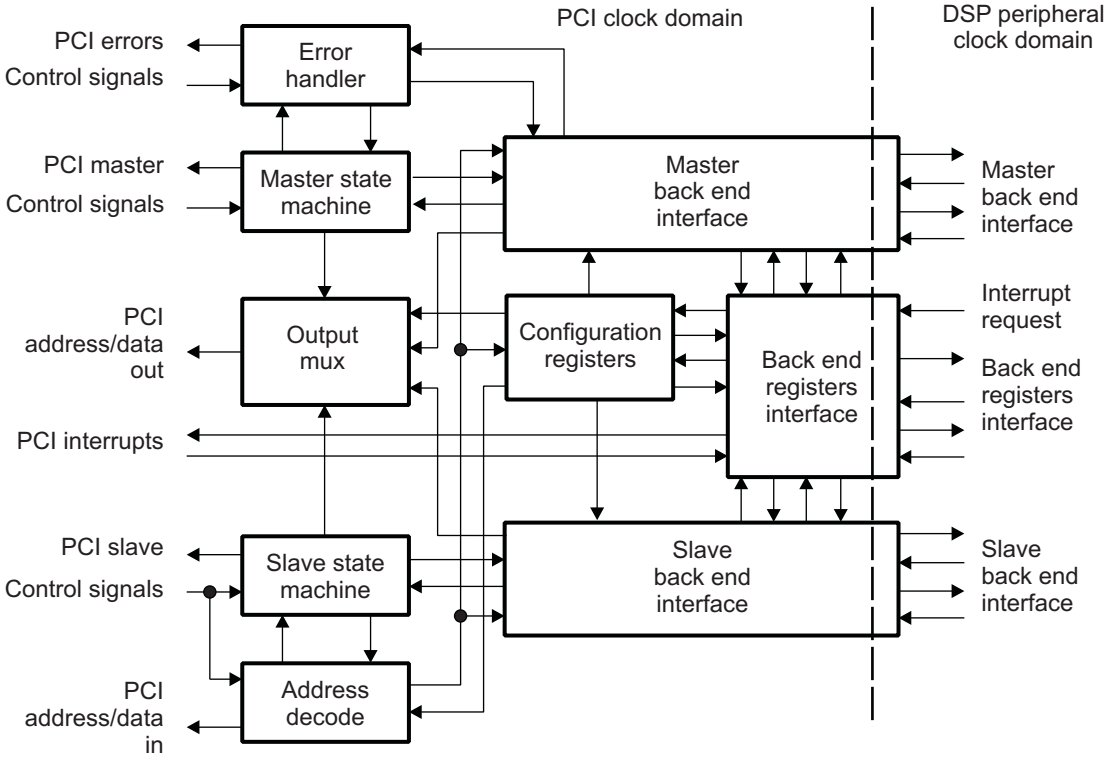
- PCI special cycles
- PCI interrupt acknowledge cycles
- PCI lock
- 64-bit bus operation
- Operation at frequencies greater than 66 MHz
- Address/data stepping
- Combining (for write posting)
- Collapsing
- Merging
- Cache line wrap accesses
- Reserved accesses
- Message signaled interrupts
- Vital product data
- Slave IO Read and IO Write Transactions

2 PCI Architecture

The PCI consists of the following blocks:

- Address decoder
- Slave state machine
- Slave back end interface
- Master back end interface
- Master state machine
- Output multiplexer
- Configuration registers
- Error handler
- Back end registers interface

Figure 1. PCI Block Diagram



2.1 Address Decoder

This block latches transaction control information from the PCI bus and decodes that information to determine if the transaction was targeted to its PCI slave. This block instructs the Slave State machine to either accept or ignore slave transactions as they are presented on the PCI bus.

2.2 Slave State Machine

This block generates and monitors all of the PCI signals necessary for accepting transactions on the bus. All of the slave PCI protocols handling functions are split between the address decoder and slave state machine blocks.

2.3 Slave Back End PCI Interface

This block accepts transactions from the slave state machine and passes those transactions to the back end interface. This block performs the asynchronous decoupling between the PCI clock domain and the peripheral clock domain for slave transactions. It also implements the slave address translation control registers and performs address translation for slave transactions.

2.4 Master Back End PCI Interface

This block accepts bus transactions from the back end interface and passes those transactions on to the master state machine. It performs the asynchronous decoupling between the peripheral clock domain and the PCI clock domain for master transactions. This block implements the master address translation control registers and also performs the address translation for master transactions.

2.5 Master State Machine

This block generates and monitors all of the PCI signals necessary for initiating transactions on the bus. The majority of the master PCI protocols handling functions are implemented in this block. This block responds to transfer requests that are presented to it from the master back end interface.

2.6 Output Multiplexer

This block multiplexes the master address, master write data, slave configuration read data, and slave memory read data on to the AD pins at the appropriate times. This block is controlled by several of the other blocks in the PCI.

2.7 Configuration Registers

This block implements the required PCI configuration registers and some of the back end registers. These registers control the modes and options in the PCI and provide vital information to the PCI host.

2.8 Error Handler

This block monitors for error conditions that may occur on the PCI bus.

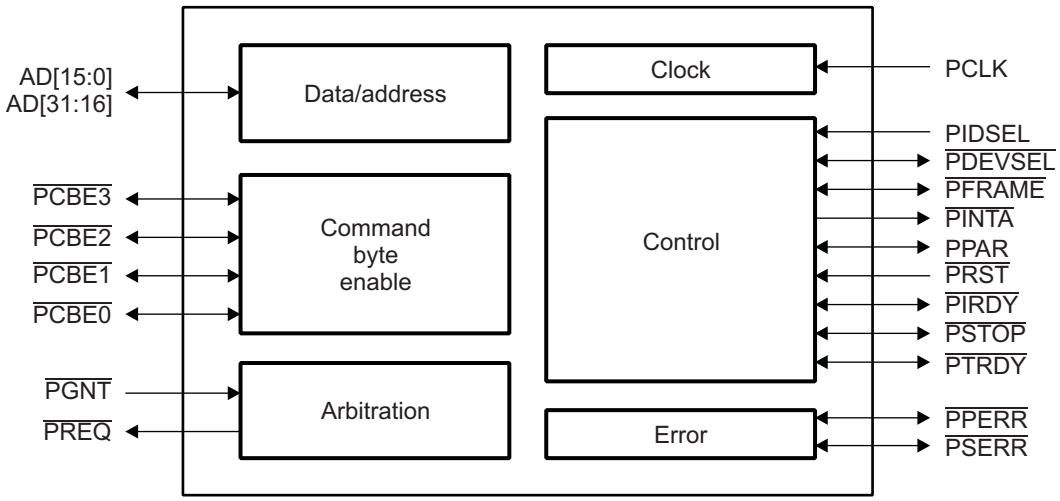
2.9 Back End Registers Interface

This block implements the asynchronous bridging function that allows DSP masters to access select PCI configuration registers, the master and slave address translation registers, and other miscellaneous PCI control/status registers. This block also implements some PCI control/status registers that reside in the back end peripheral clock domain.

3 PCI Signals

Figure 2 lists the PCI signals that are used by the PCI.

Figure 2. PCI Signals



3.1 PCI Pin Description

Table 1 shows the PCI pin name with the signal direction and description.

Table 1. PCI Pin Description

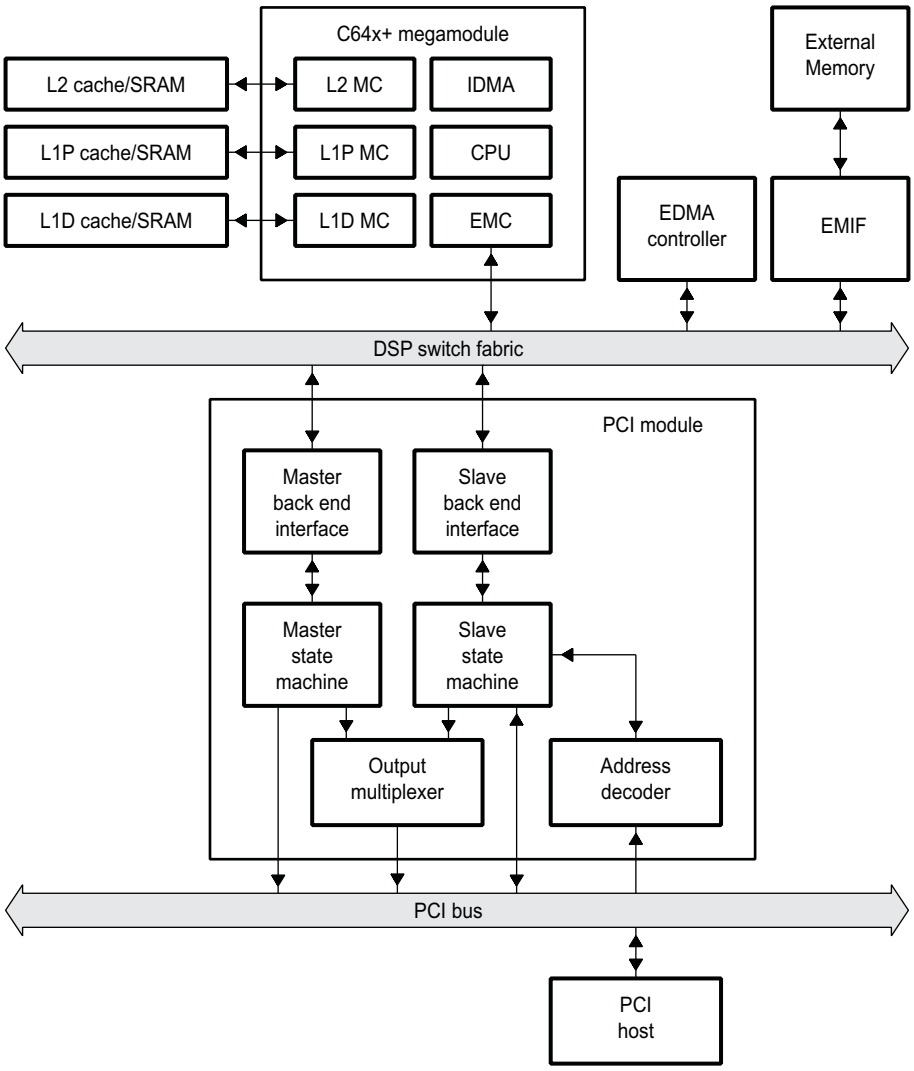
Pin Name	Signal Direction [1]	Description
$\overline{\text{PFRAME}}$	I/O/Z	PCI Frame
$\overline{\text{PDEVSEL}}$	I/O/Z	PCI Device Select
$\overline{\text{PSTOP}}$	I/O/Z	PCI Transaction Stop Indicator
PCLK	I	PCI Clock
$\overline{\text{PCBE[3:0]}}$	I/O/Z	PCI Command/Byte Enables
PPAR	I/O/Z	PCI Parity
$\overline{\text{PPERR}}$	I/O/Z	PCI Parity Error
$\overline{\text{PSERR}}$	I/O/Z	PCI System Error
$\overline{\text{PIRDY}}$	I/O/Z	PCI Initiator Ready
$\overline{\text{PREQ}}$	O/Z	PCI Bus Request
$\overline{\text{PINTA}}$	O/Z	PCI Interrupt A
$\overline{\text{PRST}}$	I	PCI Reset
$\overline{\text{PGNT}}$	I	PCI Bus Grant
PIDSEL	I	PCI Initialization Select
$\overline{\text{PTRDY}}$	I/O/Z	PCI Transmitter Ready
AD[31:16]	I/O/Z	PCI Data/Address bus [31:16]
AD[15:0]	I/O/Z	PCI DataAddress bus [16:0]

[1] For the pin type, I = Input, O = Output, Z = High impedance.

3.2 Connecting a Local PCI to an External PCI Device

Figure 3 shows a simplified block diagram of how the PCI module interfaces local DSP master modules (EDMA controller, CPU, etc.) and other DSP resources (EMIF, DSP internal memory, etc.) to external PCI memory and external PCI masters.

Figure 3. PCI to External PCI device



A EDMA: Enhanced Direct Memory Access Controller
EMIF: External Memory Interface
EMC: Extended Memory Controller
L1P MC: L1 P Memory Controller
L1D MC: L1D Memory Controller
L2 MC: L1 Memory Controller
IDMA: Internal Direct Memory Access Controller

The following steps show how the PCI module interfaces local DSP master modules (EDMA controller, CPU, etc.) to external PCI memory:

1. A DSP master initiates a transaction aimed at external PCI memory through the DSP switch fabric.
2. The address is decoded by the PCI master back end interface.
3. The PCI master back end interface claims the transaction if the DSP address falls within the master memory map (described in Section 8).
4. PCI master back end interface translates the DSP address into a PCI address and generates a request to the master state machine.
5. The master state machine initiates a transaction on the PCI bus using the PCI address.
6. The request is received by the external PCI host, which responds accordingly.

The following steps show how the PCI module interfaces external PCI masters to DSP resources (EMIF, DSP internal memory, etc.):

1. External PCI master initiates a transaction on the PCI bus.
2. The PCI address decoder decodes the PCI address of the transaction and instructs the PCI slave state machine to claim the transaction if the PCI address falls within the slave memory map (described in Section 6) assigned to the DSP.
3. The PCI slave state machine forwards the request to the PCI slave back end Interface.
4. PCI slave back end Interface translates the PCI address into a DSP address and places the DSP address on the DSP switch fabric.
5. All DSP slaves decode the address to determine if they are being accessed. If so, they respond accordingly. For example, in the case of an external memory access, the EMIF accesses external memory using the DSP address.

4 Clocks

The PCI module uses the following clocks:

- Main PCI clock from the PCLK pin
  - 33 MHz from PCLK pin when in 33-MHz mode
  - 66 MHz from PCLK pin when in 66-MHz mode
- Internal DSP peripheral clock
  - Sourced by the DSP PLL controller, see the device-specific data manual for more information.
  - Peripheral clock frequency must not be less than the main PCI clock frequency.

The PCI module must be configured at device reset for 33- or 66-MHz mode. The PCI Frequency Selection pin (PCI66) of the DSP determines the frequency of operation for the PCI module. When PCI66 = 0 at reset, the PCI module is configured for 33-MHz operation. When PCI66 = 1, the PCI module is configured for 66-MHz operation. See the device-specific data manual for more information on the PCI66 configuration pin.

5 Byte Addressing

The PCI interface is byte-addressable. It can read and write 8-bit bytes, 16-bit half words, 24-bit words, and 32-bit words. Words are aligned on an even four-byte boundary, and always start at a byte address where the two LSBs are 00. Halfwords always start at a byte address where the last LSB is 0. PCI slave transactions are fully byte-addressable, but PCI master transactions must start on a word-aligned address.

6 Slave Memory Map

The PCI module on TCI648x devices provides full visibility for a PCI host into DSP memory through six sets of PCI slave base address translation registers (PCIBAR0TRL, PCIBAR1TRL, PCIBAR2TRL, PCIBAR3TRL, PCIBAR4TRL, and PCIBAR5TRL) and PCI base address mask registers (PCIBAR0MSK, PCIBAR1MSK, PCIBAR2MSK, PCIBAR3MSK, PCIBAR4MSK, and PCIBAR5MSK). The DSP can use any of these sets of registers to map any memory region or MMRs to the PCI memory map. These registers can be configured by software at any time. The default values of these registers provide the mapping shown in Table 2.

Table 2. PCI Base Addresses

Base Address	Window Size	Prefetchable	Memory Space
0	8MB	Yes	0x008 0000
1	4MB	No	0x018 0000
2	8MB	No	0x028 0000
3	8MB	Yes	0x800 0000(1)
4	8MB	Yes	0xA00 0000
5	8MB	Yes	0xE00 0000

(1) The default value for PCI Base Address 3 Translation Register (0x800 0000) is not a supported memory address range for TCI648x devices and needs to be reprogrammed before being used.

Section 6.1 and Section 6.2 explain how to map a region in the PCI host address space to a region in the DSP memory space by setting up a slave window.

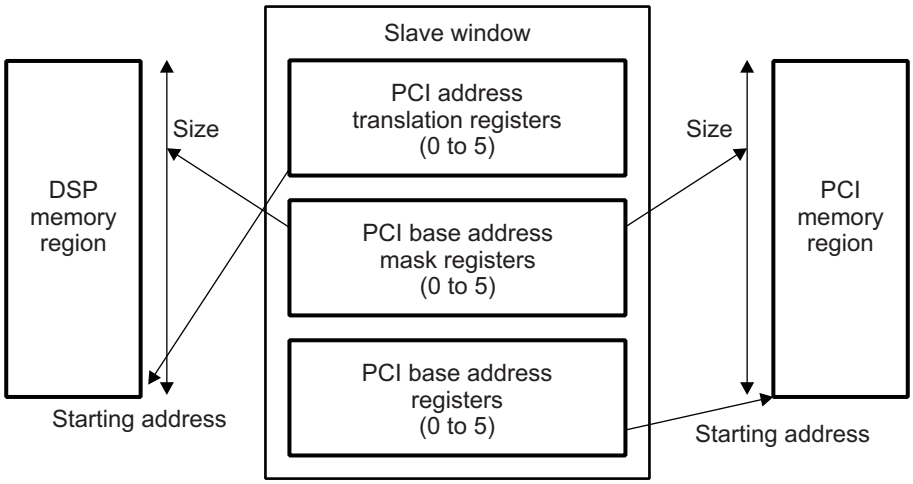
6.1 Configuring Slave Window Registers

A slave window maps a region in the DSP memory space to a region in the PCI address space. This allows a PCI host to access the DSP memory through the PCI address space. A slave window is configured with the following registers:

- PCI slave base address translation register: Configures the starting address of the window in the DSP address space
- PCI base address register: Configures the starting address of the slave window in the PCI address space
- PCI base address mask register: Configures the size of the window and prefetchability of the DSP memory region being mapped

PCI supports six slave window configurations with the support of these registers. For more information on slave access address translation, see Section 6.2. Figure 4 displays a slave window configuration.

Figure 4. Slave Window Configuration



6.1.1 Configuration of Base Address Registers 0 to 5 by PCI Host (PCIBAR*n*)

The base address registers allow the PCI host to map the DSP's address space into the host memory or I/O address space.

The base address registers reside in configuration space and a PCI host normally configures them. The PCI host can access base address registers 0 to 5 (PCIBAR0, PCIBAR1, PCIBAR2, PCIBAR3, PCIBAR4 and PCIBAR5) by performing a TYPE 0 access in the PCI bus.

The base address registers contain the following bit fields:

- ADDR (31-4) : These bits specify the base address of the slave window on the PCI address space
- PREFETCH (3) : This bit specifies the prefetchablity of the memory space controlled by the base address register
- TYPE (2:1) : These bits specify whether the base address maps to PCI I/O address space or memory address space. The TCI648x PCI supports only mapping into PCI memory space.
- IOMEM_SP_IND (0) : The size of the base address register, either 32 or 64 bits. The TCI648x PCI only supports 32-bit addressing.

Normally, a PCI host configures the ADDR bits of the base address registers during its boot time when it enumerates all the PCI devices. The write-access of the PCI host to each of the ADDR bits is determined by the corresponding bit in the address mask (ADDRMASK) bits of the PCI base address mask registers. A bit in ADDR is read-only to the PCI host when its corresponding bit in ADDRMASK is cleared. Conversely, a bit in ADDR can be both read and written by the PCI host when its corresponding bit in ADDRMASK is set. The DSP is required to complete the configuration of the address mask register before the host attempts to configure the base address registers.

6.1.2 Configuration of Slave Base Address Translation Register 0 to 5 (PCIBAR*n*TRL) by DSP

A slave base address translation register configures the DSP side parameters of a slave window. There are six slave base address translation registers (PCIBAR0TRL, PCIBAR1TRL, PCIBAR2TRL, PCIBAR3TRL, PCIBAR4TRL, and PCIBAR5TRL) that allow six slave windows to be set up. The slave base address translation registers are configured by the DSP. PCI slave base address translation registers control the translation of transaction addresses as they flow from the external PCI bus to the DSP. Section 6.2 explains the translation of PCI addresses to DSP addresses.

6.1.3 Configuration of PCI Base Address (0 to 5) Mask Registers (PCIBAR*n*MSK) by DSP

A PCI base address mask register configures the size and prefetchablity of a slave window. There are six slave base address translation registers available in PCI to support six slave windows. The PCI base address mask registers (PCIBAR0MSK, PCIBAR1MSK, PCIBAR2MSK, PCIBAR3MSK, PCIBAR4MSK, and PCIBAR5MSK) are configured by the DSP. The DSP can access the PCI base address mask registers directly, as they are mapped to the DSP memory space.

A PCI base address mask register includes the following bit fields:

- ADDRMASK(31-4): These bits control the PCI host write access of the corresponding bits in the corresponding PCI configuration base address registers (0 to 5).
- PREFETCH_EN (3): This bit specifies whether or not the memory space controlled by the corresponding PCI configuration base address register is prefetchable. This bit is reflected in bit 3 of the corresponding PCI configuration base address register.

The DSP should configure this register before the host attempts to program the base address register.

6.2 Slave Access Address Translations

Window configurations control the translation of transaction addresses as they flow from the external PCI bus to the DSP. This translation process uses the contents of the corresponding PCI base address mask register (PCIBAR0MSK, PCIBAR1MSK, PCIBAR2MSK, PCIBAR3MSK, PCIBAR4MSK, and PCIBAR5MSK) to determine which of the bits in the PCI address should be modified. Bits 31 to 4 (ADDRMASK) are replaced in the address where the corresponding bit in the base address mask register is set by the corresponding bit in the slave base address translation register.

The following steps occur during a PCI-to-DSP address translation:

1. During the address phase, the external PCI master places the PCI address on the address bus AD[31:0].
2. The PCI finds the appropriate slave window for the address by comparing the address bits given on AD[31: N] with the corresponding bits in the base address register of all the slave windows one by one. The value of N is the number of bits set in the corresponding base address mask register of the slave window. The minimum value of N is 4. The value of N indicates the number of significant bits in the PCI address that needs to be decoded. If the address on AD[31:N] matches the corresponding bits of the base address register of any one of the slave windows, the PCI claims the PCI transaction. Otherwise, it ignores the transaction.
3. If the PCI claims the transaction, it generates a DSP address by replacing bits 31:N in the PCI address with the corresponding bits in the base address translation register of the previously selected slave window.

Figure 5. PCI-to-DSP Address Translation

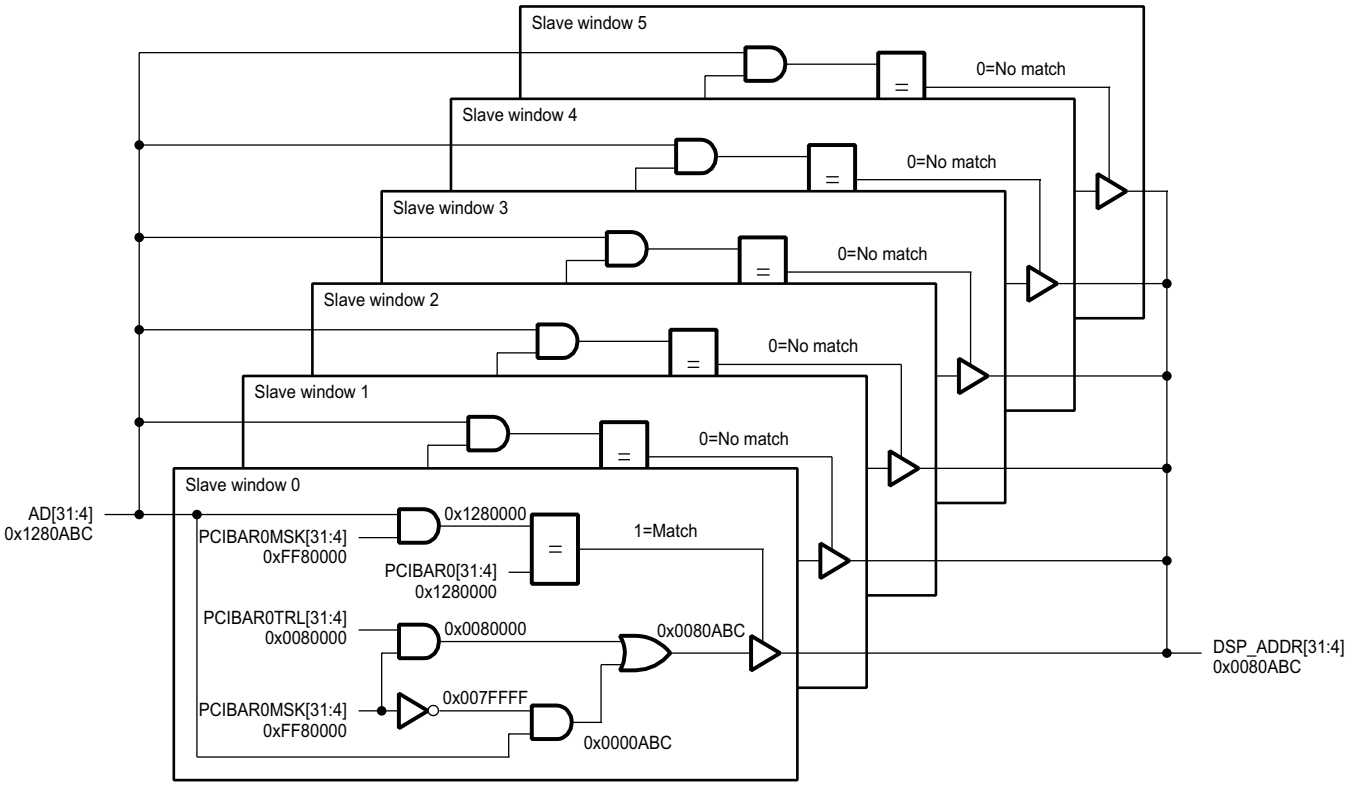


Figure 5 gives an example of a PCI-to-DSP address translation using a PCI address of 0x1280 ABC0h. In this example, slave window 0 is created using this configuration: PCIBAR0 = 1280 0000h, PCIBAR0MSK = FF80 0000h, and PCIBAR0TRL = 0800 0000h. With these settings, slave window 0 translates PCI addresses from 1280 0000h to 12FF FFFFh (8MB) to DSP addresses 0800 0000h to 08FF FFFFh. The following is the sequence of events for generating the DSP address:

1. The external PCI master places the PCI address 1280 ABC0h on the address bus AD[31:0].
2. The PCI finds the slave window corresponding to the PCI address by comparing the address given on AD[31:4] with the corresponding bits in the PCI base address registers. The PCI base address mask registers indicate bits in the PCI address that need to be compared.
3. Slave window 0 has PCIBAR0[31:4] set to 1280 000h and PCIBAR0MSK[31:4] = FF80 000h. Therefore, the PCI matches the PCI address with slave window 0 and claims transaction.
4. The value of the PCIBAR0MSK[31:4] bits is inverted and ANDed with the PCI address (AD[31:4]). The PCIBAR0TRL[31:4] bits are also ANDed with the value of the PCIBAR0MSK[31:4] register. The resulting values are ORed together to form the DSP address (0080 ABC0h).

7 Slave Operations

The PCI slave operates in response to transfer requests that are presented on the PCI bus. The PCI slave was intended to enable high performance read and write performance through the use of delayed transactions combined with prefetching for reads and posting for writes. The PCI slave supports two FIFOS/buffers (a read and write) for efficient data transfer. Each buffer holds 16 32-bit words of read or write data.

7.1 Slave Configuration Operations

7.1.1 Configuration Write Transactions

The decoding of and response to a configuration write transaction by the PCI slave depends on the following:

- PCBE[3:0] must be 0xB during the address phase
- PIDSEL must be asserted during the address phase
- AD[1:0] must be 00b during the address phase

If the above conditions are met, the transaction is decoded as a hit and the PCI slave will assert $\overline{\text{PDEVSEL}}$ using medium decode timing. $\overline{\text{PTRDY}}$ will be asserted coincident with the assertion of $\overline{\text{PDEVSEL}}$. If the master on the PCI bus intends to perform more than a single data phase transaction (as determined by the state of $\overline{\text{PFRAME}}$), $\overline{\text{PSTOP}}$ will also be asserted coincident with the assertion of $\overline{\text{PDEVSEL}}$ and $\overline{\text{PTRDY}}$ to signal a disconnect. $\overline{\text{PSTOP}}$ will continue to be asserted until $\overline{\text{PFRAME}}$ is deasserted and $\overline{\text{PIRDY}}$ is asserted in accordance with the PCI specification.

Configuration write transactions will never result in a retry. Because the configuration registers are included within the PCI, no transactions will occur on any of the back end interfaces as a result of a configuration write transaction.

7.1.2 Configuration Read Transactions

Decoding of and response to a configuration read transaction by the PCI slave depends on the following:

- PCBE[3:0] must be 0xA during the address phase
- PIDSEL must be asserted during the address phase
- AD[1:0] must be 00b during the address phase

If the above conditions are met, the transaction is decoded as a hit and the PCI slave will assert $\overline{\text{PDEVSEL}}$ using medium decode timing. $\overline{\text{PTRDY}}$ will be asserted one cycle following the assertion of $\overline{\text{PDEVSEL}}$. If the master on the PCI bus intends to perform more than a single data phase transaction (as determined by the state of $\overline{\text{PFRAME}}$), $\overline{\text{PSTOP}}$ will also be asserted coincident with the assertion of $\overline{\text{PTRDY}}$ to signal disconnect. $\overline{\text{PSTOP}}$ will continue to be asserted until $\overline{\text{PFRAME}}$ is deasserted and $\overline{\text{PIRDY}}$ is asserted in accordance with the PCI specification.

Configuration read transactions will never result in a retry. Because the configuration registers are included within the PCI, no transactions will occur on any of the PCI blocks as a result of a configuration read transaction.

7.2 Slave Memory Operations

7.2.1 Memory Write or Memory Write and Invalidate Transactions

The PCI slave treats memory write and memory write and invalidate transactions identically and is therefore not intended to support cacheable memory spaces.

The decoding of and response to a memory write or a memory write and invalidate transaction by the PCI slave depends on the following conditions:

- PCBE[3:0] must be 0x7 (memory write) or 0xF (memory write and invalidate) during the address phase
- At least one of the 6 base address registers (PCIBAR0, PCIBAR1, PCIBAR2, PCIBAR3, PCIBAR4, and PCIBAR5) must have bit 0 (IOMEM_SP_IND) set to 0
- The address which is given on AD[31: N] during the address phase must match the value in the corresponding bits of one of the memory base address registers. The value of N is based on the base address mask registers (PCIBAR0MSK, PCIBAR1MSK, PCIBAR2MSK, PCIBAR3MSK, PCIBAR4MSK, and PCIBAR5MSK) and indicates the number of significant bits in the address to be decoded. The minimum value of N is 4.
- The PCI slave write buffer is empty
- The address that is given on AD[1:0] during the address phase must be 00b (linear addressing)

If only the first three of the previous conditions are met, a retry will be issued because the PCI slave write buffer is not empty. If only the first four conditions are met, a single data phase transfer will be completed on the PCI bus. This transfer will be identical in behavior to a configuration write transaction. If the byte enables have at least one byte asserted, a write transaction will initiate on the DSP as soon as any pending read burst requests have completed. If no byte enables are asserted, the transaction will be terminated internally in the PCI slave and no DSP transactions will occur. If all of the above conditions are met, a multi-data phase transfer will be completed. No wait states will be inserted by the PCI slave via the target ready indicator ($\overline{\text{PTRDY}}$), but wait states inserted by the master will be properly handled. The burst will be allowed to continue until one of the following conditions occurs:

- The PCI slave write buffer is almost full
- A data phase with no asserted byte enables is encountered
- The burst is about to extend beyond the address boundary of the PCI slave
- The master ends the transaction

7.2.2 Memory Read Transactions

The decoding of and response to a memory read transaction by the PCI slave depends on the following conditions:

- PCBE[3:0] must be 0x6 during the address phase
- At least one of the 6 base address registers (PCIBAR0, PCIBAR1, PCIBAR2, PCIBAR3, PCIBAR4, and PCIBAR5) must have bit 0 (IOMEM_SP_IND) set to 0
- The address on AD[31: N] during the address phase must match the value in the corresponding bits of one of the memory base address registers. The value of N is based on the base address mask registers (PCIBAR0MSK, PCIBAR1MSK, PCIBAR2MSK, PCIBAR3MSK, PCIBAR4MSK, and PCIBAR5MSK) and indicates the number of significant bits in the address to be decoded. The minimum value of N is 4.
- A delayed read is not outstanding, or this transaction is a re-request of a pending delayed read request and the read data is available. A delayed read is a transaction that must complete on the destination bus before completing on the originating bus.
- The PCI slave write buffer is empty

If only the first three of the previous conditions are met, a retry will be issued because the delayed transaction is not ready to complete or the PCI slave write buffer is not empty. If the first four or all of the conditions are met, a single data phase transfer will be completed on the PCI bus. This transfer will be similar in behavior to a configuration transaction.

If the transaction is not a delayed completion and the byte enables have at least one byte asserted, a single word read transaction will initiate on the DSP as soon as any pending write burst requests have completed on the interface. If no byte enables are asserted and the addressed memory region is not prefetchable, the transaction will be terminated internally in the PCI slave and no DSP transactions will occur. The byte enables which were presented for the first data phase on the PCI bus will be inverted and presented as the byte enables for the transfer. The PCI slave can wait up to 12 PCLK cycles for data to be returned from the slave back end interface. If data does not return within this time, a retry will be issued and the transaction will be tagged as a delayed read. Alternatively, for performance reasons, if the FORCE_DEL_READ bit (bit 2) in the PCI Slave Control Register (PCISLVCNTL) is asserted, a retry and delayed read can be immediately forced without waiting for the 12 PCLK cycles.

If the transaction is a delayed completion and the data is available, the transaction will complete immediately with $\overline{\text{PTRDY}}$ being asserted coincident with $\overline{\text{PDEVSEL}}$. Only a single-word prefetch is supported for the memory read command even when it is used within prefetchable memory regions.

7.2.3 Memory Read Line Transactions

The decoding of and response to a memory read line transaction by the PCI slave depends on the following conditions:

- PCBE[3:0] must be 0xE during the address phase
- At least one of the 6 base address registers (PCIBAR0, PCIBAR1, PCIBAR2, PCIBAR3, PCIBAR4, and PCIBAR5) must have bit 0 (IOMEM_SP_IND) set to 0
- The address on AD [31: N] during the address phase must match the value in the corresponding bits of one of the memory base address registers. The value of N is based on the base address mask registers (PCIBAR0MSK,PCIBAR1MSK,PCIBAR2MSK,PCIBAR3MSK,PCIBAR4MSK, and PCIBAR5MSK) and indicates the number of significant bits in the address to be decoded. The minimum value of N is 4.
- A delayed read is not outstanding, or this transaction is a re-request of a pending delayed read line request and the read data is available. A delayed read is a transaction that must complete on the destination bus before completing on the originating bus.
- The PCI slave write buffer is empty

If only the first three previous conditions are met, a retry will be issued because the delayed transaction is not ready to complete or the PCI slave buffer is not empty. If the first four or all of the conditions are met, a burst read operation will be initiated on the DSP as soon as any pending write burst requests have completed on the interface. The length of the transfer will be the number of words from the requested address to the end of the cache line, unless this value exceeds the size of the PCI slave read data FIFO (16 words). If the burst size is larger than the PCI slave read data FIFO, the transfer will be broken up into 8 word transfers in the same way as the memory read multiple transfers. Since memory read line transactions are prefetchable, all byte enables are asserted internally during the burst. The PCI slave can wait up to 12 PCLK cycles for data to be returned from the slave memory slave back end interface. If data does not return within this time, a retry will be issued and the transaction will be tagged as a delayed read. Alternatively, for performance reasons, if the FORCE_DEL_READ_LN bit (bit 3) in the PCI Slave Control Register (PCISLVCNTL) is asserted, a retry and delayed read can be immediately forced without waiting for the 12 PCLK cycles. If the transaction is a delayed completion and the data is available, the first data phase of the transaction will complete immediately with $\overline{\text{PTRDY}}$ being asserted coincident with $\overline{\text{PDEVSEL}}$. $\overline{\text{PTRDY}}$ will continue to be asserted until all of the prefetched data has been read or the burst is terminated by the master. If the master attempts to burst beyond the current cache line, the PCI will assert $\overline{\text{PSTOP}}$ on the next to the last data phase in the cache line.

7.2.4 Memory Read Multiple Transactions

The decoding of and response to a memory read multiple transaction by the PCI slave depends on the following conditions:

- PCBE[3:0] must be 0xC during the address phase
- At least one of the 6 base address registers (PCIBAR0, PCIBAR1, PCIBAR2, PCIBAR3, PCIBAR4, and PCIBAR5) must have bit 0 (IOMEM_SP_IND) set to 0
- The address on AD [31: N] during the address phase must match the value in the corresponding bits of one of the memory base address registers. The value of N is based on the base address mask registers (PCIBAR0MSK,PCIBAR1MSK,PCIBAR2MSK,PCIBAR3MSK,PCIBAR4MSK, and PCIBAR5MSK) and indicates the number of significant bits in the address to be decoded. The minimum value of N is 4
- A delayed read is not outstanding, or this transaction is a re-request of a pending delayed read line request and the read data is available. A delayed read is a transaction that must complete on the destination bus before completing on the originating bus.
- The PCI slave write buffer is empty

If only the first three previous conditions are met, a retry will be issued because the delayed transaction is not ready to complete or the PCI slave buffer is not empty. If the first four or all of the conditions are met, a burst read operation will be initiated on the DSP as soon as any pending write burst request have completed on the interface. The length of the transfer will be 16 words for the initial transfer of a burst. As memory read multiple transactions are prefetchable, all byte enables are asserted internally during the burst. The PCI slave can wait up to 12 PCLK cycles for data to be returned from the slave memory slave back end interface. If data does not return within this time, a retry will be issued and the transaction will be tagged as a delayed read. Alternatively, for performance reasons, if the FORCE_DEL_READ_MUL bit (bit 4) in the PCI Slave Control (PCISLVCNTL) is asserted, a retry and delayed read can be immediately forced without waiting for the 12 PCLK cycles.

If the transaction is a delayed completion and the data is available, the first data phase of the transaction will complete immediately with $\overline{\text{PTRDY}}$ being asserted coincident with $\overline{\text{PDEVSEL}}$. $\overline{\text{PTRDY}}$ will continue to be asserted as long as data is available in the read prefetch buffer or the burst is terminated by the master. As data is transferred from the read prefetch buffer on to the PCI bus, additional 8 word read fetches will be performed on the slave memory slave back end interface as buffer space becomes available.

8 Master Memory Map

The PCI enables the DSP to access the PCI memory through the master memory map. There is 256MB of space dedicated for PCI memory in the DSP memory map. This 256MB space is divided into 32 windows of 8MB fixed size. These windows are called master windows or PCI address windows. Each master window can be configured individually to map 8MB of PCI memory to the DSP address space.

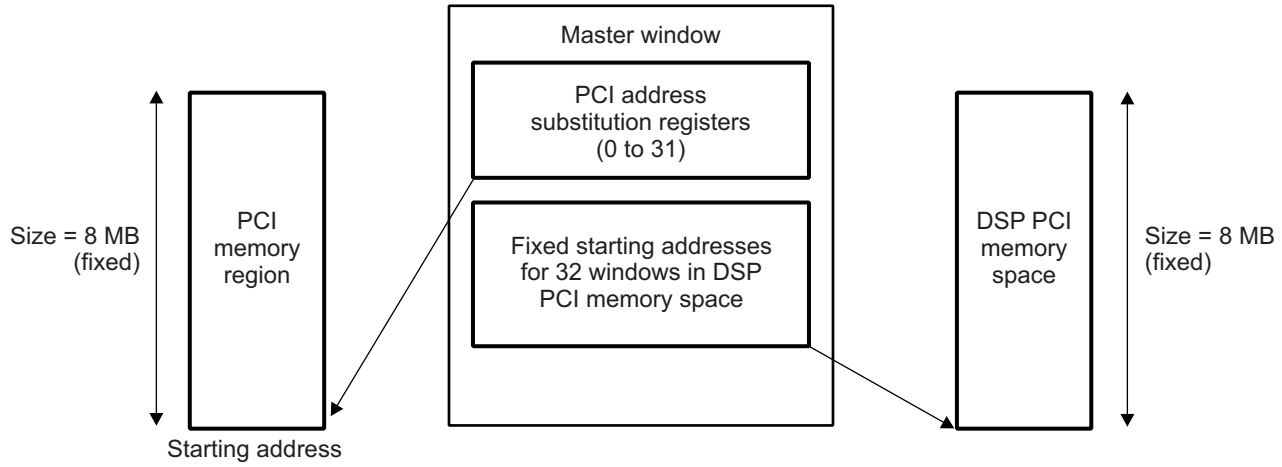
Table 3. PCI Master Windows

Master Window Number	Base Address Register	Window Size	DSP Memory Range
0	PCIADDSUB0		4000 0000h - 407F FFFFh
1	PCIADDSUB1	8MB	4080 0000h - 40FF FFFFh
2	PCIADDSUB2	8MB	4100 0000h - 417F FFFFh
3	PCIADDSUB3	8MB	4180 0000h - 41FF FFFFh
4	PCIADDSUB4	8MB	4200 0000h - 427F FFFFh
5	PCIADDSUB5	8MB	4280 0000h - 42FF FFFFh
6	PCIADDSUB6	8MB	4300 0000h - 437F FFFFh
7	PCIADDSUB7	8MB	4380 0000h - 43FF FFFFh
8	PCIADDSUB8	8MB	4400 0000h - 447F FFFFh
9	PCIADDSUB9	8MB	4480 0000h - 44FF FFFFh
10	PCIADDSUB10	8MB	4500 0000h - 457F FFFFh
11	PCIADDSUB11	8MB	4580 0000h - 45FF FFFFh
12	PCIADDSUB12	8MB	4600 0000h - 467F FFFFh
13	PCIADDSUB13	8MB	4680 0000h - 46FF FFFFh
14	PCIADDSUB14	8MB	4700 0000h - 477F FFFFh
15	PCIADDSUB15	8MB	4780 0000h - 47FF FFFFh
16	PCIADDSUB16	8MB	4800 0000h - 487F FFFFh
17	PCIADDSUB17	8MB	4880 0000h - 48FF FFFFh
18	PCIADDSUB18	8MB	4900 0000h - 497F FFFFh
19	PCIADDSUB19	8MB	4980 0000h - 49FF FFFFh
20	PCIADDSUB20	8MB	4A00 0000h - 4A7F FFFFh
21	PCIADDSUB21	8MB	4A80 0000h - 4AFF FFFFh
22	PCIADDSUB22	8MB	4B00 0000h - 4B7F FFFFh
23	PCIADDSUB23	8MB	4B80 0000h - 4BFF FFFFh
24	PCIADDSUB24	8MB	4C00 0000h - 4C7F FFFFh
25	PCIADDSUB25	8MB	4C80 0000h - 4CFF FFFFh
26	PCIADDSUB26	8MB	4D00 0000h - 4D7F FFFFh
27	PCIADDSUB27	8MB	4D80 0000h - 4DFF FFFFh
28	PCIADDSUB28	8MB	4E000 000h - 4E7F FFFFh
29	PCIADDSUB29	8MB	4E800 000h - 4EFF FFFFh
30	PCIADDSUB30	8MB	4F000 000h - 4F7F FFFFh
31	PCIADDSUB31	8MB	4F800 000h - 4FFF FFFFh

8.1 Configuring Master Windows Using Address Substitution Registers 0 to 31 (PCIADDSUB*n*)

Each master window corresponds to 8MB of the DSP's PCI memory address space. For example, window 0 corresponds to DSP addresses 4000 000h – 407F FFFFh, and window 1 corresponds to the next 8 MB, 4080 0000h – 40FF FFFFh. Each window can map an 8MB of PCI memory to its corresponding DSP's PCI memory address space through its address substitution register. Figure 6 displays a master window configuration.

Figure 6. Master Window Configuration



There are 32 address substitution registers (PCIADDSUB*n*, 0 – 31) available in the PCI. Each of these registers corresponds to a master window. These registers reside in the PCI interface and are normally programmed by the DSP. Figure 7 shows an example of an address substitution register.

Figure 7. PCI Address Substitution Register (0 to 31)

31	23	22	0
ADD_SUBS		Reserved	
R/W-0		R-0	

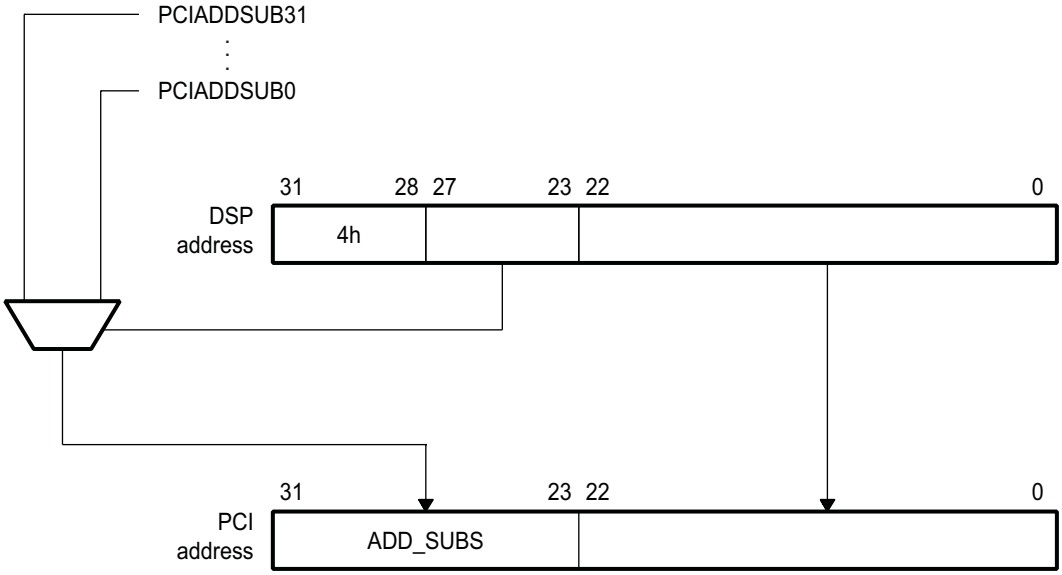
LEGEND: R/W = Read/Write, R = Read only; -*n* = value at reset

Bits 31-23 (ADD_SUBS) of the register contain the MSBs of the PCI addresses within the corresponding window. The remaining reserved 23 bits are the size of the window and these bits have no function. Reads of this field will return 0s. Section 8.2 explains the translation of addresses as a transaction flow from the DSP domain to the PCI domain.

8.2 Master Address Translation

Address translation from the DSP to the PCI domain is done using the address substitution registers 0 to 31. Figure 8 shows the address just prior to the address translation.

Figure 8. DSP-to-PCI Address Translation



During the address translation, the upper 4 bits are don't cares, as they have already been used for the address decode to reach the DSP's PCI memory space. The next 5 bits decide which PCI address window corresponds to the DSP address. Once the window is determined, the 23 LSBs of the DSP address are allowed to pass through to the PCI address and the upper 9 bits are taken from the ADD_SUBS field of the corresponding PCIADDSUB*n* register. Figure 9 illustrates this process.

Figure 9. Example of DSP-to-PCI Address Translation

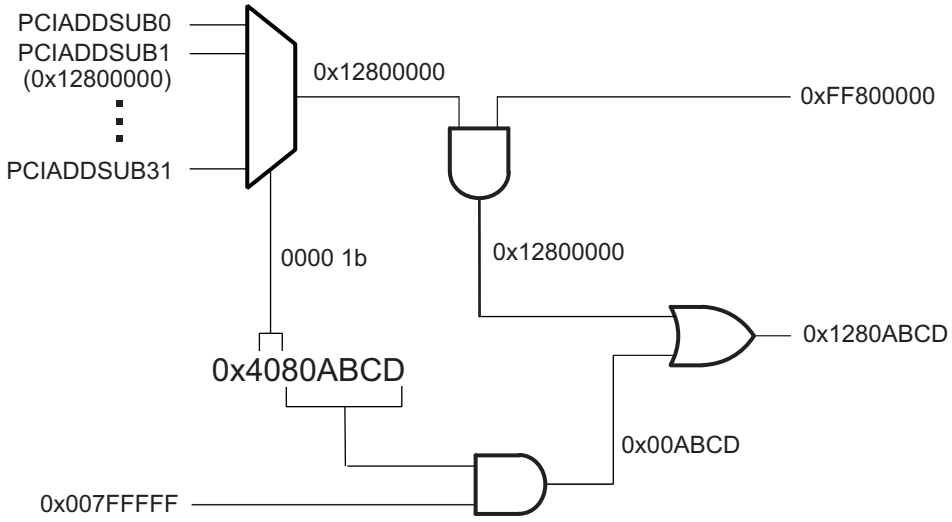


Figure 9 shows an example of a DSP-to-PCI address translation. In this example, the DSP address is 4080 ABCDh and the PCIADDSUB1 register is set to 1280 0000h.

1. The 4 MSBs of the address (0x4) indicate the DSP's PCI memory space is being accessed.
2. The next 5 bits (00001b) determine that the destination is the second 8MB PCI window, signifying that PCIADDSUB1 will be used. Note that bits 27:3 directly correspond to the PCIADDSUB*n* register used.
3. The upper 9 bits of the address substitution register 1 and the lower 23 bits of the DSP address are concatenated to form the PCI address (1280 ABCDh).
4. This address is used by the PCI module.

Example 1 details how to perform an EDMA transfer from a DSP source address (for example, L2 or EMIF) to PCI memory. For the EDMA configuration, see the *TMS320TCI648x DSP Enhanced DMA (EDMA3) Controller User's Guide* (SPRU727).

NOTE: EDMA transfer controller 0 cannot be used for PCI transfers. Only EDMA transfer controllers 1, 2, and 3 have access to the PCI memory space.

Example 1. Master Memory Write

```
/* Configure master window */
    /* Map external PCI memory (0x12800000) to its corresponding 8MB DSPs
     * PCI memory address space using following address substitution register PCIADDSUB1
     */
     PCIADDSUB1 = 0x12800000; /* PCIADDSUBn, n = 0 to 31 */
     ...
       /* Setup EDMA module and Enable the DMA Region */
       /* Setup EDMA PARAM */
       /* The OPT register contains following the bit fields
         ITCCHEN = 0x0;TCCHEN = 0x0; ITCINTEN = 0x0; TCINTEN = 0x1;
         WIMODE = 0x0; TCC = 0x0; TCCMODE = 0x0; FIFOWID = 0x2;
         STATIC = 0x0; SYNCDIM = 0x0; DAM = 0x0; SAM = 0x0; */
     PARAMSET [paramEntry]. OPT = 0x100200; /* paramEntry can be 0 to 511 */
     PARAMSET [paramEntry]. srcAddr = (Uint32)srcAddr; //srcAddr can be L2 or EMIF
     PARAMSET [paramEntry]. dstAddr = (Uint32)0x40800000;
     PARAMSET [paramEntry]. aCntbCnt = 0x00100004;
     PARAMSET [paramEntry]. srcDstBidx = 0x00000004;
     PARAMSET [paramEntry]. linkBcntrld = 0x0;
     PARAMSET [paramEntry]. srcDstCidx = 0x0;
     PARAMSET [paramEntry]. cCnt = 1;
     /* Setup EDMA Channel */
     /* Enable the Channel */
     /* Wait for a transmit completion */
     /* Example End */
```

Example 2 details how to perform an EDMA transfer from PCI memory to a DSP destination, such as L2 or EMIF. For EDMA Configuration, see the *TMS320TCI648x DSP Enhanced DMA (EDMA3) Controller User's Guide* (SPRU727).

Example 2. Master Memory Read

```
/* Configure master window */
    /* Map external PCI memory (0x12800000) to its corresponding 8MB DSPs
     * PCI memory address space using following address substitution register PCIADDSUB1
     */
     PCIADDSUB1 = 0x12800000; /* PCIADDSUBn, n = 0 to 31 */
     ...
       /* Setup EDMA module and Enable the DMA Region */
       /* Setup EDMA PARAM */
       /* The OPT register contains following the bit fields
         ITCCHEN = 0x0;TCCHEN = 0x0; ITCINTEN = 0x0; TCINTEN = 0x1;
         WIMODE = 0x0; TCC = 0x0; TCCMODE = 0x0; FIFOWID = 0x2;
         STATIC = 0x0; SYNCDIM = 0x0; DAM = 0x0; SAM = 0x0; */
     PARAMSET [paramEntry]. OPT = 0x00100200; /* paramEntry can be 0 to 511 */
     PARAMSET [paramEntry]. srcAddr = (Uint32) 0x40800000;
     PARAMSET [paramEntry]. dstAddr = (Uint32) dstAddr; //dstAddr can be L2 or EMIF
     PARAMSET [paramEntry]. aCntbCnt = 0x00100004;
     PARAMSET [paramEntry]. srcDstBidx = 0x00040000;
     PARAMSET [paramEntry]. linkBcntrld = 0x0;
     PARAMSET [paramEntry]. srcDstCidx = 0x0;
     PARAMSET [paramEntry]. cCnt = 1;
     /* Setup EDMA Channel */
     /* Enable the Channel */
     /* Wait for a transmit completion */
     /* Example End */
```

9 Master Operations

The PCI master operates in response to memory transfer requests that are presented on the master back end interface, and to indirect IO and configuration requests that are presented via the master configuration/IO transaction proxy registers. For memory transactions, the master can be programmed to only use the basic memory read or write transactions, or can also perform burst transfers as efficiently as possible by automatically selecting the proper command for memory transactions based on the transaction length and the cache line size. The PCI master supports two FIFOS/buffers (a read and write) for efficient data transfer. Each buffer holds 16 32-bit words of read or write data.

9.1 Master Configuration Operations

During a configuration space access, AD[31:02] is used to address the PCI device, the function within the device and a DWORD in the function's configuration space. AD[1:0] is ignored. However, you can set AD[1:0] to 00 for TYPE 0 access, or set it to 01 for TYPE 1 access.

The byte enables select the bytes within the addressed DWORD. The byte enables allow access to a byte, word, DWORD, or non-contiguous bytes in the addressed DWORD. In the addressed DWORD, BE0 enables byte 0, BE1 enables the byte 1, and so on.

9.1.1 Configuration Write Transactions

The back end application causes the PCI master to perform a configuration write operation by executing the following steps:

1. Reading the PCI master configuration/IO access command register (PCIMCFGCMD) and ensuring that the READY bit (bit 31) is asserted.
2. Writing the data for the configuration write to the PCI master configuration/IO access data register (PCIMCFGDAT) through the DSP PCI interface
3. Writing the address for the configuration write to the PCI master configuration/IO access address register (PCIMCFGADR).
4. Writing to the PCI master configuration/IO access command Register (PCIMCFGCMD) with the TYPE field set to 0 (Configuration Transaction), the RD_WR field set to 0 (Write), and the BYTE_EN fields set to the desired value.

On the next cycle, $\overline{\text{PREQ}}$ is asserted. Once $\overline{\text{PGNT}}$ is sampled asserted, the master asserts $\overline{\text{PFRAME}}$ and outputs the write address onto the AD pins and the Configuration Write command (0xB) onto the PCBE pins. On the next cycle, the master asserts $\overline{\text{PIRDY}}$, deasserts $\overline{\text{PFRAME}}$, and outputs the data to be written. The master will never insert wait states during a transfer but will respond to wait states as controlled by $\overline{\text{PTRDY}}$. As is required, the master continually checks the bus for exceptions (master abort, target abort, retry, disconnect, latency timeout, parity error, system error) while the transfer is ongoing. A ready signal is returned to the DSP PCI Master interface through the READY bit of the PCI Master Configuration/IO Access Command register (PCIMCFGCMD) when the transfer is complete.

9.1.2 Configuration Read Transactions

The back end application can request that the PCI master to perform a configuration read operation by doing the following:

1. Reading the PCI master configuration/IO access command register (PCIMCFGCMD) and ensuring that the READY bit (bit 31) is asserted.
2. Writing the address for the configuration write to the PCI master configuration/IO access address register (PCIMCFGADR).
3. Writing to the PCI master configuration/IO access command register (PCIMCFGCMD) with the TYPE field set to 0 (Configuration Transaction), the RD_WR field set to 1 (Read), and the BYTE_EN fields set to the desired value.

On the next cycle, PREQ is asserted. Once PGNT is sampled asserted, the master asserts PFRAME and outputs the read address onto the AD pins and the Configuration Read command (0xA) onto the PCBE pins. On the next cycle, the master asserts PIRDY, and deasserts PFRAME. The master will never insert wait states during a transfer but will respond to wait states as controlled by PTRDY. As is required, the master continually checks the bus for exceptions (master abort, target abort, retry, disconnect, latency timeout, parity error, system error) while the transfer is ongoing. When the transfer is complete, the data is returned to the PCI Master Configuration/IO Access Data Register (PCIMCFGDAT) and the READY bit of the PCI Master Configuration/IO Access Command register (PCIMCFGCMD) is set to 1.

9.2 Master I/O Operations

In the I/O Address Space, all 32 AD lines are used to provide a full byte address. The master that initiates an I/O transaction is required to ensure that AD[1:0] indicates the least significant valid byte for the transaction.

The byte enables indicate the size of the transfer and the affected bytes within the DWORD and must be consistent with AD[1:0]. Table 4 lists the valid combinations for AD[1:0] and the byte enables for the initial data phase. Byte enables are asserted when 0.

Table 4. Byte Enables and AD[1:0] Encodings

AD[1:0]	Starting Byte	Valid BE#[3:0] Combinations
00	Byte 0	xxx0 or 1111
01	Byte 1	xx01 or 1111
10	Byte 2	x011 or 1111
11	Byte 3	0111 or 1111

9.2.1 I/O Write Transactions

The back end application can request for the PCI master to perform an I/O write operation by doing the following:

1. Reading the PCI master configuration/IO access command register (PCIMCFGCMD) and ensuring that the READY bit (bit 31) is asserted.
2. Writing the data for the configuration write to the PCI master configuration/IO access data register (PCIMCFGDAT), through the PCI Back End Registers interface.
3. Writing the address for the configuration write to the PCI master configuration/IO access address register (PCIMCFGADR).
4. Writing to the PCI master configuration/IO access command register (PCIMCFGCMD) with the TYPE field set to 0 (I/O Transaction), the RD_WR field set to 0 (Write), and the BYTE_EN fields set to the desired value.

When a request is made for I/O write operation, the DSP PCI master interface asserts PREQ on the PCI bus. Once PGNT is sampled asserted, the master asserts PFRAME and outputs the write address onto the AD pins and the I/O Write command (0x3) onto the PCBE pins. On the next cycle, the master asserts PIRDY, deasserts PFRAME, and outputs the data to be written. The master will never insert wait states during a transfer but will respond to wait states as controlled by PTRDY. As is required, the master continually checks the bus for exceptions (master abort, target abort, retry, disconnect, latency timeout, parity error, system error) while the transfer is ongoing. A ready signal is returned to the DSP PCI Master interface through the READY bit of the PCI Master Configuration/IO Access Command register (PCIMCFGCMD) when the transfer is complete.

9.2.2 I/O Read Transactions

The back end application can request for the PCI master to perform an I/O read operation by doing the following:

1. Reading the PCI master configuration/IO access command register (PCIMCFGCMD) and ensuring that the READY bit (bit 31) is asserted.
2. Writing the address for the configuration write to the PCI master configuration/IO access address register (PCIMCFGADR).
3. Writing to the PCI master configuration/IO access command register (PCIMCFGCMD) with the TYPE field set to 0 (Configuration Transaction), the RD_WR field set to 1 (Read), and the BYTE_EN fields set to the desired value.

When a request is made for I/O read operation, the DSP PCI master interface asserts $\overline{\text{PREQ}}$ on the PCI bus, $\overline{\text{PREQ}}$ is asserted. Once $\overline{\text{PGNT}}$ is sampled asserted, the master asserts $\overline{\text{PFRAME}}$ and outputs the read address onto the AD pins and the I/O Read command (0x2) onto the PCBE pins. On the next cycle, the master asserts $\overline{\text{PIRDY}}$, and deasserts $\overline{\text{PFRAME}}$. The master will never insert wait states during a transfer but will respond to wait states as controlled by $\overline{\text{PTRDY}}$. As is required, the master continually checks the bus for exceptions (master abort, target abort, retry, disconnect, latency timeout, parity error, system error) while the transfer is ongoing. When the transfer is complete, the data is returned to the PCI Master Configuration/IO Access Data Register (PCIMCFGDAT) and the READY bit of the PCI Master Configuration/IO Access Command register (PCIMCFGCMD) is set.

9.3 Master Memory Operations

In the memory access, bits AD[31:02] are used to address the PCI device, the function within the device and a DWORD in the function's memory space. Bits AD[1:0] are ignored. However, bits AD[1:0] indicate the order in which the master is requesting the data to be transferred.

9.3.1 Memory Write Transactions

The back end application can request for the PCI master to perform a memory write operation by making a memory write access to PCI master memory map. When a request is made for a memory write operation, the PCI master asserts $\overline{\text{PREQ}}$ on the PCI bus. Once $\overline{\text{PGNT}}$ is sampled asserted; the master asserts $\overline{\text{PFRAME}}$ and outputs the write address onto the AD pins and the Memory Write command (0x7) onto the PCBE pins. On the next cycle, the master asserts $\overline{\text{PIRDY}}$ and outputs the first word of data to be written. If the transfer only consists of a single data phase, $\overline{\text{PFRAME}}$ is deasserted as required by the PCI specification coincident with the assertion of $\overline{\text{PIRDY}}$. Otherwise, $\overline{\text{PFRAME}}$ continues to be asserted until the next to the last data phase completes. The master will never insert wait states during a burst but will respond to wait states as controlled by $\overline{\text{PTRDY}}$. As is required, the master continually checks the bus for exceptions (master abort, target abort, retry, disconnect, latency timeout, parity error, system error) while the transfer is ongoing. As the burst progresses, an internal ready signal is returned from the master back end interface for each successful data phase completion until the entire burst is finished.

Section 8.2 provides an example that describes how to program the EDMA to perform a master memory write operation.

9.3.2 Memory Read Transactions

The back end application can request for the PCI master to perform a memory read operation by making a memory read access to PCI master memory map.

When a request is made for a memory read operation, $\overline{\text{PREQ}}$ is asserted and the read burst is performed following the same behavior as for the write burst (no wait states, etc.). During the address phase, the Memory Read (0x6), Memory Read Line (0xE), or Memory Read Multiple command (0xC) is output on the PCBE pins depending on the length of the transfer and the cache line size. The byte enables that were registered from the master back end interface are inverted and output during the data phases. Unlike the write burst, the byte enables on the PCI bus will not change as the transaction progresses. This will not cause problems since bursts are only performed to prefetchable regions of memory. As the burst progresses, an internal ready signal is returned from the master back end interface along with the read data for each successful data phase completion until the entire burst is finished.

Section 8.2 provides an example that describes how to program the EDMA to perform a master memory read operation.

10 Exceptions, Status Reporting, and Interrupts

10.1 PCI Exceptions

The PCI supports the detection of the error conditions listed in Table 5. The PCI can generate an interrupt to the Host and DSP for a particular set of error conditions. The PCISTATSET, PCIHINTSET and PCIBINTSET registers enable interrupts to the Host and DSP. The PCI also provides the status of these PCI errors, as described in section Section 10.2.

Table 5. PCI Exceptions

Exception Name	Description
PERR_DET	Data parity error. Detected during a read transaction of the PCI bus master and write transaction of a PCI bus target.
SERR_DET	System error. Detected when the PCI has received a target abort while mastering the bus or when an address parity error is detected on the PCI bus.
MS_ABRT_DET	Master Abort. Generated by the PCI master unit in the PCI to indicate that it terminated a transaction with a master abort.
TGT_ABRT_DET	Target Abort. Generated by the PCI slave unit in the PCI to indicate that it has initiated a target abort.

10.1.1 Parity Error

If the PCI master is mastering the bus, the Data Parity Reported bit (MS_DPAR_REP) in the PCI configuration space command/status register (PCICSR) will be set under either of the following conditions:

- If it detects a parity error during the data phase of a read transaction
- If it detects that $\overline{\text{PPERR}}$ has been asserted by the target during the data phase of a write transaction

The Data Parity Detected bit (DET_PAR_ERR) in the PCI configuration space command/status register (PCICSR) will be set under any of the following conditions:

- If it is acting as the PCI bus master and it detects a data parity error during a read transaction
- If it is acting as a PCI bus target and it detects a data parity error during a write transaction
- If it detects an address parity error

The PCI will assert $\overline{\text{PPERR}}$ if the Parity Error Response bit (PAR_ERR_RES) in the PCI configuration space command/status register (PCICSR) is set and the DET_PAR_ERR bit is set. The assertion of $\overline{\text{PPERR}}$ will remain valid until the second clock after the cycle in which the error occurred.

If a parity error is detected during a transfer involving the PCI, the transaction will be allowed to complete unless the PCI is the master and a target disconnect is detected (i.e., the PCI will not master abort due to a parity error).

10.1.2 System Error

The PCI will set an internal system error flag under any of the following conditions:

- If an address parity error is detected on the PCI bus (even if the PCI is not the target of the transaction) and the Parity Error Response bit (PAR_ERR_RES) is set in the PCI configuration space command/status register (PCICSR)
- If the PCI detected $\overline{\text{PPERR}}$ asserted while mastering the bus
- If the PCI received a target abort (disconnect without retry) while mastering the bus

The PCI will assert $\overline{\text{PSERR}}$ if the SERR_N Enable bit (SERR_N_EN) in the PCI configuration space Command/Status register (PCICSR) is set and the internal system error flag is set. The PCI will halt and wait for software or hardware reset after $\overline{\text{PSERR}}$ has been asserted. The PCI will set the Signaled System Error bit (SIG_SYS_ERR) in the PCI configuration space Command/Status register (PCICSR) whenever $\overline{\text{PSERR}}$ is asserted.

10.1.3 Master Abort Protocol

If a master abort occurs while the PCI is the master, the current transfer will be gracefully terminated on both the PCI (by de-asserting $\overline{\text{PFRAME}}$ and asserting $\overline{\text{PIRDY}}$) and back end buses (by supplying ready signals through the back end interface until the burst is completed). Both the Received Master Abort signal in the PCI command/status register (PCICSR) and the Master Abort bit (RCV_MS_ABRT) in the PCI Back End Command/Status Mirror Register (PCICSRMIR) will be set.

10.1.4 Target Abort Protocol

If a target abort occurs while the PCI is the master, the current transfer will be gracefully terminated on both the PCI and back end buses (in the same way as for the master abort). Both the Received Target Abort signal in the PCI command/status register (PCICSR) and the Target Abort bit (RCV_TGT_ABRT) in the PCI Back End Command/Status Mirror Register (PCICSRMIR) will be set.

10.1.5 Retry /Disconnect Protocol

If a transaction is disconnected or retried, the master will unconditionally repeat the transaction starting at the location of the first remaining uncompleted word. The back end has no knowledge of retry or disconnections on the bus.

10.2 Status Reporting

The PCI provides the status of various PCI errors and interrupts generated or detected by it in the Command/Status register and an internal Status Register. The status of the bits in the internal Status Register can also be changed manually. Setting or clearing a bit in the internal Status Register does not affect the corresponding bit in the Command/Status register. Similarly, clearing the Command/Status register does not affect the corresponding bit in the internal Status Register.

The Command/Status Register (PCICSR) is present in configuration space. The host can access this register through the TYPE0 configuration space access. The DSP can access this register through Command/Status Mirror register (PCICSRMIR). This register provides the status of the following error conditions:

- Detected Parity Error (DET_PAR_ERR)
- Signaled System Error (SIG_SYS_ERR)
- Received Master Abort Error (RCV_MS_ABRT)
- Received Target Abort Error (RCV_TGT_ABRT)
- Signaled Target Abort (SIG_TGT_ABRT)
- Master Data Parity Reported (MS_DPAR_REP)

Status bits in PCICSR cannot be set manually. They are set only by the PCI. A status bit in this register can be cleared by writing a 1 to that bit.

The internal Status Register is internal to PCI and not directly accessible to the host. It is also not directly accessible to the back end application. The PCI provides the Status Set Register (PCISTATSET) and the Status Clear Register (PCISTATCLR) to set and clear the bits in the internal Status Register. These two registers are available in the back end interface. Reading of both these registers returns the value of the internal PCI Status Register.

The Status Register provides the status of the following PCI interrupt and errors:

- DSP interrupt (DSPINT)
- Software interrupts (SOFT_INT)
- Parity Error Detected (PERR_DET)
- System Error Detected (SERR_DET)
- Master Abort Error Detected (MS_ABRT_DET)
- Target Abort Error Detected (TGT_ABRT_DET)

The Host/Backend interrupt is asserted for an error condition, provided the corresponding bit is enabled in the internal Host/Backend Interrupt Enable Set register, and only when the bit corresponding to that error is set in the Status register. To clear an interrupt or error condition, the corresponding status bit in the Command/Status register needs to be cleared first and then, if available, the corresponding bit in the internal status register also needs to be cleared.

10.3 PCI Interrupts

The PCI can generate an interrupt for the status conditions listed in Table 6. When a status condition is set, PCI can raise an interrupt to the DSP or PCI host or both based on what interrupts have been enabled for host and back end interface. See Section 10.3.1 and Section 10.3.2 for enabling interrupts to host and DSP for a particular set of status conditions.

Table 6. PCI Interrupts

Interrupt Name	Description
INTA	A level-sensitive active-low interrupt is generated to the host on the $\overline{\text{PINTA}}$ pin if a bit in the internal PCI Status Register is asserted and the corresponding bit in the internal Host Interrupt Enable Register is also asserted, as long as the PCI is in the D0 power state.
PERR_DET	A parity error is detected during a read transaction of PCI bus master and write transaction of a PCI bus target.
SERR_DET	A system error is detected when the PCI has received a target abort while mastering the bus, or when an address parity error is detected on the PCI bus.
MS_ABRT_DET	A Master Abort is generated by the PCI master unit in the PCI to indicate that it terminated a transaction with a master abort.
TGT_ABRT_DET	A Target Abort is generated by the PCI slave unit in the PCI to indicate that it has initiated a target abort.

10.3.1 DSP-to-Host Interrupts

PCI can raise an interrupt to the host for various status conditions, as described in Table 6. The PCI includes an internal Host Interrupt Enable Register that specifies which status conditions will generate interrupts to the host. The PCI Host Interrupt Enable Register is not directly accessible by the host or the back end application. Two registers are provided to set or clear bits in the internal PCI Host Interrupt Enable Register: PCI Host Interrupt Enable Set Register (PCIHINTSET) and PCI Host Interrupt Enable Clear Register (PCIHINTCLR).

An interrupt for a particular status condition can be enabled by setting the corresponding bit in PCIHINTSET. The interrupt for INTA is enabled by default when PCI is in D0 power state. Reading the PCIHINTSET register returns the contents of the internal Host Interrupt Enable Register.

An interrupt for a particular status condition can be disabled by setting the corresponding bit in the PCIHINTCLR register. Reading PCIHINTCLR returns the bitwise ANDing of the internal PCI Status Register and the internal PCI Host Interrupt Enable Register. The PCIHINTCLR register is typically read by the host to determine the source of an interrupt when the $\overline{\text{PINTA}}$ pin is asserted.

A level-sensitive active-low interrupt is generated to the host on the $\overline{\text{PINTA}}$ pin if a bit in the internal Status Register is asserted and the corresponding bit in the internal Host Interrupt Enable Register is also asserted. When an interrupt is raised on $\overline{\text{PINTA}}$ pin, the host can read the PCIHINTCLR register to determine the status condition that caused the interrupt.

Software can also use the SOFT_INT bits to interrupt the host via the $\overline{\text{PINTA}}$ pin. Software interrupts are enabled and disabled by writing to the PCI Host Interrupt Enable Register. Setting the corresponding bit in the internal Status Register will assert a level sensitive active low interrupt on the $\overline{\text{PINTA}}$ pin. The DSP or host can clear this interrupt condition by clearing applicable bit in the internal Status Register.

Interrupt generation on the $\overline{\text{PINTA}}$ pin is enabled only when the PCI is in D0 power state. It is disabled in the D1, D2, or D3 power states.

10.3.2 Host-to-DSP Interrupts

PCI can raise an interrupt to back end (DSP) for various status conditions described in Table 6. The PCI includes an internal PCI Back End Interrupt Enable Register that specifies which status conditions will generate interrupts to the back end. The PCI Back End Interrupt Enable Register is not directly accessible by the host or the back end application. Two registers are provided to set or clear bits in the internal PCI Back End Interrupt Enable Register: the Back End Application Interrupt Enable Set Register (PCIBINTSET), and the Back End Application Interrupt Enable Clear Register (PCIBINTCLR).

An interrupt for a particular status condition can be enabled by setting the corresponding bit in the Back End Interrupt Enable Set Register (PCIBINTSET). Reading the PCIBENTSET register returns the contents of the internal Back End Interrupt Enable Register.

An interrupt for a particular status condition can be disabled by setting the corresponding bit in the PCIBINTCLR register. Reading PCIBINTCLR returns the bitwise ANDing of the internal PCI Status Register and the internal PCI Back End Interrupt Enable Register. The PCIBINTCLR register is typically read by the back end application to determine the source of an interrupt.

An interrupt request is generated to the DSP if a bit in the internal Status Register (PCISTATSET) is asserted and the corresponding bit in the internal Back End Application Interrupt Enable Register (PCIBINTSET) is also asserted. When an interrupt is raised to the DSP (back end), the DSP can read PCISTATCLR to know the status condition that caused the interrupt.

The interrupt request to the DSP is generated through the PCI-to-DSP interrupt (DSPINT) line. This interrupt can also be forced by setting the DSPINT bit or any of the SOFT_INT bits of the PCI Status Set Register (PCISTATSET). Note that the DSPINT and SOFT_INT interrupts are independently enabled through the internal PCI Back End Application Interrupt Enable register.

11 PCI Reset Information

11.1 PCI Pin Reset

The PCI reset pin ($\overline{\text{PRST}}$) is the main PCI hardware reset. This resets most of the PCI logic within the main PCI clock domain. This reset brings PCI-specific registers, sequencers, and signals to a consistent state.

11.2 C64x+ Megamodule Local Reset

An external PCI host can control the local reset to the C64x+ Megamodule by setting or clearing the LRESET bit of the PCI LRESET Register (PCILRSTREG). For a description of the local reset, see the *TMS320C64x+ Megamodule Reference Guide* (SPRU871).

11.3 PCI Register Reset Values

The reset values for some PCI registers are specified in the PCI Configuration Hook Registers. The values in the PCI Configuration Hook Registers are latched to their corresponding PCI registers following a PCI hardware reset (through the $\overline{\text{PRST}}$ pin). This functionality is implemented mainly to support PCI autoinitialization, which is described in more detail in Section 12.4. Section 13 lists the PCI Configuration Hook Registers.

12 PCI Configuration

The operation of PCI is configured through the configuration space registers and back end registers.

12.1 Programming the PCI Configuration Space Registers

Configuration space registers can be programmed both from the external PCI host and DSP. Normally, the PCI host programs a part of configuration space registers and DSP programs the remaining part.

A PCI host can control modes and options in PCI by programming the configuration space registers. For example, the PCI host can program the Command/Status register (PCICSR) to enable PCI bus master capability for the DSP and to enable the memory access to DSP. It can program the Base Address Registers (PCIBAR0 to PCIBAR5) to map the DSP memory regions into PCI address space.

The PCI host can access the configuration space registers by performing a TYPE 0 access in the PCI bus.

The DSP needs to program a set of registers in the configuration space before the PCI host system software scans the PCI, as part of enumerating the PCI devices. For example, it needs to program Vendor ID/Device ID (PCIVENDEV) and Class Code/Revision ID registers (PCICLREV) so that the PCI host system software can identify the device and load the respective host driver if required. This can be done automatically using the I2C EEPROM initialization method, as described in Section 12.4.

The DSP cannot access the configuration space registers directly. To facilitate access to the configuration space registers, mirror registers are supported. Updating the mirror registers updates the corresponding configuration space registers.

12.2 Programming the PCI Back End Registers

The back end registers include configuration space mirror registers, master and slave address translation registers, and other miscellaneous PCI control registers.

Back end registers can be programmed both from the PCI host and the DSP. Normally, the PCI host programs a part of the back end registers and the DSP programs the remaining part of the registers.

For example, the PCI host may want to program the Host Interrupt Enable Set register (PCIHINTSET) and the Host Interrupt Enable Clear register (PCIHINTCLR) to selectively enable host interrupts.

The PCI host can access the back end registers through the slave interface supported by PCI, provided the DSP has mapped those registers to PCI through the slave window base address registers.

The DSP may program the configuration space mirror registers, slave and master address translation registers, and other miscellaneous configuration registers.

The DSP can access all the back end registers directly, as these registers are either mapped to DSP memory space or implemented as MMRs.

The back end configuration registers have a default value as described in Table 7. These values can be overwritten by the application software, and, in some cases, by the software routine located in the internal ROM of the DSP, as explained in Section 12.4.

Table 7. Back End PCI Configuration Registers

Register	Default Value
PCI Vendor ID / Device ID Mirror Register	Device ID – 0xB000 Vendor ID – 0x104C
PCI Command / Status Mirror Register	0200 0000h
PCI Class Code / Revision ID Mirror Register	0000 0001h
Subsystem Vendor ID, Subsystem ID	0000 0000h
Max Latency / Min Grant	0000 0000h
PCI Base Address Mask Register0	FF80 000h, see address register for prefetchable value
PCI Base Address Mask Register 1	FFC0 000h, see address register for prefetchable value
PCI Base Address Mask Register 2	FF80 000h, see address register for prefetchable value
PCI Base Address Mask Register 3	FF80 000h, see address register for prefetchable value
PCI Base Address Mask Register 4	FF80 000h, see address register for prefetchable value
PCI Base Address Mask Register 5	FF80 000h, see address register for prefetchable value
PCI Base Address Register 0	1b
PCI Base Address Register 1	1b
PCI Base Address Register 2	1b
PCI Base Address Register 3	1b
PCI Base Address Register 4	1b
PCI Base Address Register 5	1b
PCI Slave Control Register	111111b
PCI Base Address 0 Translation Register	0080 000h
PCI Base Address 1 Translation Register	0180 000h
PCI Base Address 2 Translation Register	0280 000h
PCI Base Address 3 Translation Register	8000 000h(1)
PCI Base Address 4 Translation Register	A000 000h
PCI Base Address 5 Translation Register	E000 000h

(1) This is not a supported memory address range in TCI648x devices and needs to be reprogrammed before being used.

12.3 PCI Configuration Hook Registers

The reset values of all the PCI back end configuration registers listed in Table 7 can be specified through a set of MMRs called the Configuration Hook Registers. See Table 9 for a list of hook registers supported. The values in these hook registers are latched to the actual PCI module registers on a PCI reset (through $\overline{\text{PRST}}$). The default values in these hook registers can be overwritten by software. These registers are implemented mainly to support PCI I2C EEPROM Autoinitialization, as explained in Section 12.4.

12.4 PCI I2C EEPROM Autoinitialization

Normally, at boot time, the PCI host provides a PCI reset to all the PCI devices. The host can start accessing a PCI device once the device completes the PCI reset. So, if any PCI configuration registers need to be programmed before the host starts accessing the PCI, it should be done before PCI reset is completed. The PCI I2C EEPROM autoinitialization method performs this function.

12.4.1 PCI Autoinitialization from I2C EEPROM

When autoinitialization is used, the PCI back end configuration registers are programmed by software located in internal DSP ROM with the values stored in an I2C EEPROM.

PCI I2C EEPROM autoinitialization is selected through the PCI_EEAI configuration input. The PCI_EEAI pin selects autoinitialization at reset as follows:

- PCI_EEAI pin value 0: PCI registers use default values
- PCI_EEAI pin value 1: DSP ROM code initializes PCI registers with values read from EEPROM

If autoinitialization is not enabled, the PCI configuration registers are left with their default values and the I2C EEPROM is not accessed for PCI configuration purposes.

If the PCI is to be autoinitialized, the PCI does not respond to accesses until the DSP ROM software has completed initializing the PCI configuration registers as follows. When autoinitialization is enabled, the CONFIG_DONE bit in the PCICFGDONE register takes a default value of 0. This prevents the PCI from responding to any requests. When autoinitialization is completed, the software sets the CONFIG_DONE bit to 1 to allow the PCI to respond to requests. Autoinitialization is carried out differently when the last reset was a power-on reset. After a power-on reset, the PCI Back End Configuration registers and their corresponding PCI Configuration Hook registers are written with the values shown in Table 8. For any other type of reset, only the PCI Configuration Hook registers are initialized.

The values in the PCI Configuration Hook registers specify the default values of corresponding PCI Back End Configuration registers. The values in the PCI Configuration Hook registers are latched to the PCI Back End Configuration registers only during a PCI reset.

12.4.2 I2C EEPROM Memory Map

The DSP requires big-endian format for the data stored in the I2C EEPROM. Byte addresses 0x400 through 0x41B of the I2C EEPROM are reserved for autoinitialization of PCI configuration registers. The remaining locations are not used for autoinitialization and can be used for storing other data. Table 8 summarizes the I2C EEPROM memory layout, as required for PCI autoinitialization.

Table 8. I2C EEPROM Memory Layout

Byte Address	Contents
0x400	Vendor ID [15:8]
0x401	Vendor ID [7:0]
0x402	Device ID [15:8]
0x403	Device ID [7:0]
0x404	Class code [7:0]
0x405	Revision ID [7:0]
0x406	Class code [23:16]
0x407	Class code [15:8]
0x408	Subsystem vendor ID [15:8]
0x409	Subsystem vendor ID [7:0]
0x40a	Subsystem ID [15:8]
0x40b	Subsystem ID [7:0]
0x40c	Max_Latency
0x40d	Min_Grant
0x40e	Reserved (use 0x00)
0x40f	Reserved (use 0x00)
0x410	Reserved (use 0x00)
0x411	Reserved (use 0x00)
0x412	Reserved (use 0x00)
0x413	Reserved (use 0x00)
0x414	Reserved (use 0x00)
0x415	Reserved (use 0x00)
0x416	Reserved (use 0x00)
0x417	Reserved (use 0x00)
0x418	Reserved (use 0x00)
0x419	Reserved (use 0x00)
0x41a	Checksum [15:8]
0x41b	Checksum [7:0]

12.4.3 I2C EEPROM Checksum

The PCI configuration data contained in the I2C EEPROM is checked against a checksum. The configuration data bytes are treated as an array of 16-bit words (little-endian format). The checksum is a 16-bit cumulative exclusive-OR (XOR) of the configuration data words, starting with an initial value of AAAAh. You must ensure that the proper 16-bit checksum value is written to address 41Ah and 41Bh when programming the I2C EEPROM.
Checksum = AAAAh XOR Word0(401h:400h) XOR Word1(403h:402)… XOR Word12(419h:418h)

If the I2C EEPROM is not accessed for PCI configuration purposes (that is, PCI_EEAI = 0 at reset), then the checksum is not performed. If the checksum fails, the DSP ROM code will re-read the I2C EEPROM. After fifty tries, if the checksum continues to fail, the DSP ROM code will simply set the CONFIG_DONE bit to 1 in the PCI Configuration Done Register (PCICFGDONE) and exit; the PCI configuration registers will be left with their default values. See the specific PCI configuration registers to determine their default values.

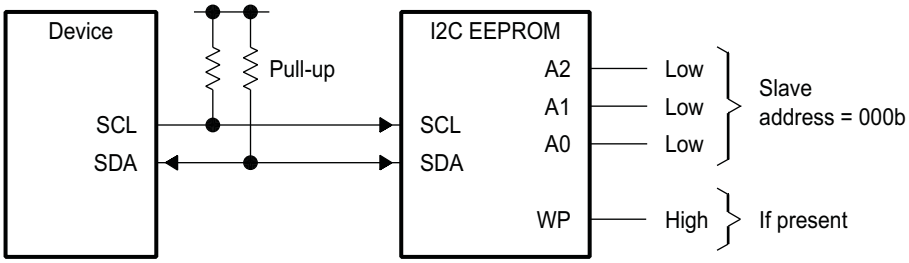
12.4.4 DSP I2C EEPROM Interface

For PCI autoinitialization, the DSP ROM code supports I2C EEPROMs or devices operating as I2C slaves with the following features:

- The memory device complies with Philips I2C Bus Specification v 2.1
- The memory device uses two bytes for internal addressing; that is, the read/write bit followed by two bytes for addressing
- The memory device has the capability to autoincrement its internal address counter such that the contents of the memory device can be read sequentially

During PCI autoinitialization, the DSP acts as the master and the I2C EEPROM acts as the slave. Figure 10 shows the minimum connection required between the DSP and one I2C EEPROM. The required pull-ups must be placed on SDA and SCL to ensure that the I2C EEPROM interface works correctly. The slave address of the I2C EEPROM slave address must be set to 0x50.

Figure 10. Signal Connections for I2C EEPROM Boot Mode



Some I2C EEPROMs have a write-protect (WP) feature that prevents unauthorized writes to memory. This feature is not needed for autoinitialization because the DSP will only read data from the I2C EEPROM. The write protect feature can be enabled or disabled without impacting operation of the DSP ROM code.

The I2C EEPROM may be used by the DSP ROM for purposes other than PCI autoinitialization. For more details, see the *TMS320TCI648x Bootloader User's Guide* (SPRUEA7).

For detailed information on the I2C, see the *TMS320TCI648x DSP Inter-Integrated Circuit (I2C) Module User's Guide* (SPRUE11).

13 PCI Registers

There are four types of PCI registers:

- PCI Configuration Registers - Configuration space registers can be programmed both from the PCI host and DSP.
- PCI Back End Configuration Registers - Back end registers can be programmed both from the PCI host and the DSP. Normally, the PCI host programs a part of back end registers and the DSP programs the remaining part of the registers.
- DSP-To-PCI Address Translation Registers – These registers will be located in the same address space as the PCI Back End Configuration registers, so no new memory space is needed. These registers are programmed from the DSP.
- PCI Configuration Hook Registers - These registers will be located in the same address space as the PCI Back End Configuration registers. These registers can be used to specify the reset value of other PCI registers and are implemented mainly to support PCI I2C EEPROM Autoinitialization.

Table 9 lists the memory-mapped registers for PCI.

Table 9. PCI Registers

Offset	Acronym	Register Description	Section
		PCI Configuration Registers	
0x000	PCIVENDEV	Vendor ID/Device ID Register	Section 13.1.1
0x004	PCICSR	Command/Status Register	Section 13.1.2
0x008	PCICLREV	Class Code/Revision ID Register	Section 13.1.3
0x00C	PCICLINE	BIST/Header Type/Latency Timer/Cacheline Size Register	Section 13.1.4
0x010	PCIBAR0	Base Address Register 0	Section 13.1.5
0x014	PCIBAR1	Base Address Register 1	Section 13.1.6
0x018	PCIBAR2	Base Address Register 2	Section 13.1.7
0x01C	PCIBAR3	Base Address Register 3	Section 13.1.8
0x020	PCIBAR4	Base Address Register 4	Section 13.1.9
0x024	PCIBAR5	Base Address Register 5	Section 13.1.10
0x02C	PCISUBID	Subsystem Vendor ID/Subsystem ID Register	Section 13.1.11
0x034	PCICPBPTR	Capabilities Pointer Register	Section 13.1.12
0x03C	PCILGINT	Max Latency/Min Grant/Interrupt Pin/Interrupt Line Register	Section 13.1.13
		PCI Back End Configuration Registers	
0x010	PCISTATSET[1]	Status Set Register	Section 13.2.1
0x014	PCISTATCLR[1]	Status Clear Register	Section 13.2.2
0x020	PCIHINTSET[1]	Host Interrupt Enable Set Register	Section 13.2.3
0x024	PCIHINTCLR[1]	Host Interrupt Enable Clear Register	Section 13.2.4
0x030	PCIBINTSET[1]	Back End Application Interrupt Enable Set Register	Section 13.2.5
0x034	PCIBINTCLR[1]	Back End Application Interrupt Enable Clear Register	Section 13.2.6
0x100	PCIVENDEVMIR	Vendor ID/Device ID Mirror Register	Section 13.2.7
0x104	PCICSRMIR	Command/Status Mirror Register	Section 13.2.8
0x108	PCICLREVMIR	Class Code/Revision ID Mirror Register	Section 13.2.9
0x10C	PCICLINEMIR	BIST/Header Type/Latency Timer/Cacheline Size Mirror Register	Section 13.2.10
0x110	PCIBAR0MSK	Base Address Mask Register 0	Section 13.2.11
0x114	PCIBAR1MSK	Base Address Mask Register 1	Section 13.2.12
0x118	PCIBAR2MSK	Base Address Mask Register 2	Section 13.2.13
0x11C	PCIBAR3MSK	Base Address Mask Register 3	Section 13.2.14
0x120	PCIBAR4MSK	Base Address Mask Register 4	Section 13.2.15
0x124	PCIBAR5MSK	Base Address Mask Register 5	Section 13.2.16

(1) These PCI Back End Configuration Registers are located in the peripheral clock domain. Registers in the peripheral clock domain can be accessed even if the PCI clock ($\overline{\text{PCLK}}$) is not running. Other registers in the PCI Back End Configuration Registers are located in the PCI clock domain, and can only be accessed if both the PCI and the peripheral clocks are running.

Table 9. PCI Registers (continued)

Offset	Acronym	Register Description	Section
0x12C	PCISUBIDMIR	Subsystem Vendor ID/Subsystem ID Mirror Register	Section 13.2.17
0x134	PCICPBPTRMIR	Capabilities Pointer Mirror Register	Section 13.2.18
0x13B	PCILGINTMIR	Max Latency/Min Grant/Interrupt Pin/Interrupt Line Mirror Register	Section 13.2.19
0x180	PCISLVCNTL	Slave Control Register	Section 13.2.20
0x1C0	PCIBAR0TRL	Slave Base Address 0 Translation Register	Section 13.2.21.1
0x1C4	PCIBAR1TRL	Slave Base Address 1 Translation Register	Section 13.2.21.2
0x1C8	PCIBAR2TRL	Slave Base Address 2 Translation Register	Section 13.2.21.3
0x1CC	PCIBAR3TRL	Slave Base Address 3 Translation Register	Section 13.2.21.4
0x1D0	PCIBAR4TRL	Slave Base Address 4 Translation Register	Section 13.2.21.5
0x1D4	PCIBAR5TRL	Slave Base Address 5 Translation Register	Section 13.2.21.6
0x1E0	PCIBAR0MIR	Base Address 0 Mirror Register	Section 13.2.22
0x1E4	PCIBAR1MIR	Base Address 1 Mirror Register	Section 13.2.22
0x1E8	PCIBAR2MIR	Base Address 2 Mirror Register	Section 13.2.22
0x1EC	PCIBAR3MIR	Base Address 3 Mirror Register	Section 13.2.22
0x1F0	PCIBAR4MIR	Base Address 4 Mirror Register	Section 13.2.22
0x1F4	PCIBAR5MIR	Base Address 5 Mirror Register	Section 13.2.22
0x300	PCIMCFGDAT	Master Configuration/IO Access Data Register	Section 13.2.23.1
0x304	PCIMCFGADR	Master Configuration/IO Access Address Register	Section 13.2.23.2
0x308	PCIMCFGCMD	Master Configuration/IO Access Command Register	Section 13.2.23.3
0x310	PCIMSTCFG	Master Configuration Register	Section 13.2.24
		DSP-to-PCI Address Translation Registers	
0x314-0x390	PCIADDSUB[0-31]	Address Substitution [0-31] Registers	Section 13.3.1
		PCI Configuration Hook Registers	
0x394	PCIVENDEVPRG	Vendor ID and Device ID Program Register	Section 13.4.1
0x398	PCICMDSTATPRG	Command and Status Program Register	Section 13.4.2
0x39C	PCICLREVPRG	Class Code and Revision ID Program Register	Section 13.4.3
0x3A0	PCISUBIDPRG	Subsystem Vendor ID and Subsystem ID Program Register	Section 13.4.4
0x3A4	PCIMAXLGPRG	Max Latency and Min Grant Program Register	Section 13.4.5
0x3A8	PCILRSTREG	LRESET Register	Section 13.4.6
0x3AC	PCICFGDONE	Configuration Done Register	Section 13.4.7
0x3B0	PCIBAR0MPRG	Base Address Mask Register 0 Program Register	Section 13.4.8
0x3B4	PCIBAR1MPRG	Base Address Mask Register 1 Program Register	Section 13.4.9
0x3B8	PCIBAR2MPRG	Base Address Mask Register 2 Program Register	Section 13.4.10
0x3BC	PCIBAR3MPRG	Base Address Mask Register 3 Program Register	Section 13.4.11
0x3C0	PCIBAR4MPRG	Base Address Mask Register 4 Program Register	Section 13.4.12
0x3C4	PCIBAR5MPRG	Base Address Mask Register 5 Program Register	Section 13.4.13
0x3C8	PCIBAR0PRG	Base Address Register 0 Program Register	Section 13.4.14
0x3CC	PCIBAR1PRG	Base Address Register 1 Program Register	Section 13.4.15
0x3D0	PCIBAR2PRG	Base Address Register 2 Program Register	Section 13.4.16
0x3D4	PCIBAR3PRG	Base Address Register 3 Program Register	Section 13.4.17
0x3D8	PCIBAR4PRG	Base Address Register 4 Program Register	Section 13.4.18
0x3DC	PCIBAR5PRG	Base Address Register 5 Program Register	Section 13.4.19
0x3E0	PCIBAR0TRLPRG	Base Address Translation Register 0 Program Register	Section 13.4.20
0x3E4	PCIBAR1TRLPRG	Base Address Translation Register 1 Program Register	Section 13.4.21
0x3E8	PCIBAR2TRLPRG	Base Address Translation Register 2 Program Register	Section 13.4.22
0x3EC	PCIBAR3TRLPRG	Base Address Translation Register 3 Program Register	Section 13.4.23
0x3F0	PCIBAR4TRLPRG	Base Address Translation Register 4 Program Register	Section 13.4.24

Table 9. PCI Registers (continued)

Offset	Acronym	Register Description	Section
0x3F4	PCIBAR5TRLPRG	Base Address Translation Register 5 Program Register	Section 13.4.25
0x3F8	PCIBASENPRG	Base Enable Program Register	Section 13.4.26

The following sections describe the various software accessible registers that are contained within the PCI. In the register figures, the bit field type/reset value row refers to accessibility from the software and is defined according to the mnemonics given in Table 10.

Table 10. Bit Field Type Encodings

Type	Meaning
R	Read Only
W	Write Only
RW	Read/Write
RW1C	Read/Write 1 to Clear
RW1S	Read/Write 1 to Set

13.1 PCI Configuration Registers

The PCI provides a standard 256-byte configuration space. This register space is provided for access by the host system software for configuration, initialization, and error handling. These registers are accessed through the PCI pins interface using Type 0 Configuration Read and Write transactions.

13.1.1 Vendor ID/Device ID Register (PCIVENDEV)

Vendor ID and/or Device ID register in the PCI Configuration Space.

Figure 11. Vendor ID/Device ID Register (PCIVENDEV)

31	16
DEV_ID	
RW-0xB000	
15	0
VEN_ID	
RW-0x104C	

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

Table 11. Vendor ID/Device ID Register (PCIVENDEV) Field Descriptions

Bit	Field	Value	Description
31-16	DEV_ID		Device ID bits. Identifies a specific device from the manufacturer. The Device ID is specified by the device manufacturer.
15-0	VEN_ID		Vendor ID bits. Uniquely identifies the manufacturer of the device. The Vendor ID is specified by the PCI Special Interest Group to insure uniqueness.

13.1.2 Command/Status Register (PCICSR)

This register is used to initialize certain bits in the PCI Configuration Space Command/Status register.

Figure 12. Command/Status Register (PCICSR)

31	30	29	28	27	26	25	24
DET_PAR_ERR	SIG_SYS_ERR	RCV_MS_ABRT	RCV_TGT_ABRT	SIG_TGT_ABRT	DEVSEL_TIM		MS_DPAR_REP
RW1C-0x0	RW1C-0x0	RW1C-0x0	RW1C-0x0	RW1C-0x0	R-0x1		RW1C-0x0

23	22	21	20	19	18	16
FAST_BTOB_CAP	Reserved	66MHZ_CAP	CAP_LIST_IMPL	INT_STAT	Reserved	
R-0x0	R-0x0	R-Undefined	R-0x0	R-0x0	R-0x0	

15	11	10	9	8
Reserved		INT_DIS	FAST_BTOB_EN	SERR_N_EN
R-0x0		RW-0x0	R-0x0	RW-0x0

7	6	5	4	3	2	1	0
WAITCYCLE CNTL	PAR_ERR_RES	VGA_PAL_SNP	MEM_WRINV_EN	SP_CYCL	BUS_MS	MEM_SP	IO_SP
R-0x0	RW-0x0	R-0x0	RW-0x0	R-0x0	RW-0x0	RW-0x0	R-0x0

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

Table 12. Command/Status Register (PCICSR) Field Descriptions

Bit	Field	Value	Description
31	DET_PAR_ERR		Detected Parity Error bit. This bit is set by the PCI to indicate that it detected a parity error, which was not necessarily reported on $\overline{\text{PPERR}}$ if parity reporting is disabled.
30	SIG_SYS_ERR		Signaled System Error bit. This bit is set by the PCI to indicate that it signaled a system error on the $\overline{\text{PSERR}}$ pin.
29	RCV_MS_ABRT		Received Master Abort bit. This bit is set by the PCI master unit in the PCI to indicate that it terminated a transaction with a master abort.
28	RCV_TGT_ABRT		Received Target Abort bit. This bit is set by the PCI master unit in the PCI to indicate that it has received a target abort when acting as a bus master.
27	SIG_TGT_ABRT		Signaled Target Abort bit. This bit is set by the PCI slave unit in the PCI to indicate that it has initiated a target abort.
26-25	DEVSEL_TIM		DEVSEL timing bits. This bit indicates the decode response time capability of the device. The PCI decode logic supports medium DEVSEL_N timing; therefore, these bits are hardwired to 01.
24	MS_DPAR_REP		Master Data Parity Reported bit. This bit is set by the PCI master unit in the PCI when all of the following conditions are met. 1. The PCI asserted $\overline{\text{PPERR}}$ or observed $\overline{\text{PPERR}}$ asserted. 2. The PCI master unit was the bus master during the observed $\overline{\text{PPERR}}$ assertion. 3. The PAR_ERR_RES bit is set.
23	FAST_BTOB_CAP		Fast Back to Back Capable bit. This bit indicates that the device is capable of performing fast back to back transactions. The PCI does not support fast back to back transactions; therefore, this bit is hardwired to 0.
22	Reserved		Reserved
21	66MHZ_CAP		66MHz Capable bit. This bit indicates whether or not the interface is capable of meeting the 66MHz PCI timing requirements.
20	CAP_LIST_IMPL		Capabilities List Implemented bit. This bit indicates whether or not the interface provides at least one capabilities list.
19	INT_STAT		Interrupt Status bit. This bit indicates the current interrupt status for the function as generated by the interrupt registers in the back end interface. If this bit is set and INT_DIS is cleared, the $\overline{\text{PINTA}}$ pin will be asserted low. INT_DIS has no effect on the value of this bit.
18-11	Reserved		Reserved, always returns 0.

Table 12. Command/Status Register (PCICSR) Field Descriptions (continued)

Bit	Field	Value	Description
10	INT_DIS		Interrupt Disable bit. This bit controls whether or not the device can assert the $\overline{\text{PINTA}}$ pin. This bit is a disable for the output driver on the $\overline{\text{PINTA}}$ pin. If this bit is set, the interrupt condition will not appear on the external $\overline{\text{PINTA}}$ pin.
9	FAST_BTOB_EN		Fast Back to Back Enable bit. This bit controls whether or not the device is allowed to perform back to back writes to different targets. The PCI will not perform fast back to back transactions; therefore, this bit is hardwired to a 0.
8	SERR_N_EN		SERR_N Enable bit. This bit is an enable for the output driver on the $\overline{\text{PSERR}}$ pin. If this bit is cleared and a system error condition is set inside the PCI, the error signal will not appear on the external $\overline{\text{PSERR}}$ pin.
7	WAITCYCLECNTL		Wait Cycle Control bit. This bit indicates whether or not the device performs address stepping. The PCI does not require address stepping; therefore, this bit is hardwired to 0.
6	PAR_ERR_RES		Parity Error response bit. This bit controls whether or not the device responds to detected parity errors. If this bit is set, the PCI will respond normally to parity errors. If this bit is cleared, the PCI will set its Detected Parity Error status bit (DET_PAR_ERR) when an error is detected, but does not assert $\overline{\text{PPERR}}$ and continues normal operation.
5	VGA_PAL_SNP		VGA Palette Snoop bit. This bit is generally not applicable for the PCI and is hardwired to 0.
4	MEM_WRINV_EN		Memory Write and Invalidate Enable bit. This bit enables the device to use the Memory Write and Invalidate command. If this bit is cleared, the PCI will not attempt to use the Memory Write and Invalidate command.
3	SP_CYCL		Special Cycles bit. This bit controls the device's response to special cycle commands. If this bit is cleared, the device will ignore all special cycle commands. If this bit is set to 1, the device can monitor special cycle commands.
2	BUS_MS		Bus Master bit. This bit enables the device to act as a PCI bus master. If this bit is cleared, the device will not act as a master on the PCI bus.
1	MEM_SP		Memory Space bit. This bit enables the device to respond to memory accesses within its address space. If this bit is cleared, the PCI will not respond to memory mapped accesses.
0	IO_SP		I/O Space bit. This bit enables the device to respond to I/O accesses within its address space. The PCI does not support IO accesses as a slave therefore this bit is hardwired to 0.

13.1.3 Class Code/Revision ID Register (PCICLREV)

Class Code and/or Revision ID register.

Figure 13. Class Code/Revision ID Register (PCICLREV)

31	16
CL_CODE	
R-0x000000	

15	8	7	0
CL_CODE		REV_ID	
R-0x000000		R-0x01	

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

Table 13. Class Code/Revision ID Register (PCICLREV) Field Descriptions

Bit	Field	Value	Description
31-8	CL_CODE		Class code bits. Identifies the generic and specific function and programming interface of the device. The Class Code is specified by the device manufacturer.
7-0	REV_ID		Revision ID bits. Identifies a revision of the specific device. The Revision ID is specified by the device manufacturer.

13.1.4 BIST/Header Type/Latency Timer/Cacheline Size Register (PCICLINE)

This register is used to set latency timer and cacheline size.

Figure 14. BIST/Header Type/Latency Timer/Cacheline Size Register (PCICLINE)

31	24	23	16
BIST		HDR_TYPE	
R-0x0		R-0x0	

15	8	7	0
LAT_TMR		CACHELN_SIZ	
RW-0x0		RW-0x0	

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

Table 14. BIST/Header Type/Latency Timer/Cacheline Size Register (PCICLINE) Field Descriptions

Bit	Field	Value	Description
31-24	BIST		Built-in self test bits. Hardwired to 0.
23-16	HDR_TYPE		Header type bits. Identifies the layout of bytes 0x10 through 0x3F and if the device is single or multi-function.
15-8	LAT_TMR		Latency timer bits. The latency timer register is provided so that the host can restrict the continued usage of the PCI bus by a master involved in a multiple data cycle transaction after its $\overline{\text{PGNT}}$ has been removed. The host is required to write a value into this register indicating the maximum number of PCI cycles for which the master can hold the bus (beginning from the assertion of $\overline{\text{PFRAME}}$). If $\overline{\text{PGNT}}$ is never removed during the transaction, the value in the latency timer value will not be used. Since the PCI will support transactions with multiple data cycles, the latency timer register is implemented. The latency timer register is initialized with all zeroes at reset. This register is not cleared on software reset.
7-0	CACHELN_SIZ		Cache line size bits. This register is provided so that the host can inform the device of the cache line size in units of 32 bit words. The value of this register is used by the PCI as a master device to determine whether to use Memory Write, Memory Write and Invalidate, Read, Read Line, or Read Multiple commands for accessing memory. This register is also used by the slave state machine to determine the size of prefetches that are performed on the Slave Back End Interface. Supported values for this register are as follows.
		00h	Disabled
		04h	Cache Line is 16 bytes
		08h	Cache Line is 32 bytes
		10h	Cache Line is 64 bytes
		20h	Cache Line is 128 bytes
			Writing an unsupported value to this register will result in the value being set to 0 as specified in the PCI Local Bus Specification.

13.1.5 Base Address Register 0 (PCIBAR0)

PCI Configuration Base Address Register 0.

Figure 15. Base Address Register 0 (PCIBAR0)

31	8
ADDR	
RW-0x0	

7	4	3	2	1	0
ADDR		PREFETCH	TYPE		IOMEM_SP_IND
RW-0x0		R-0x1b	R-0x00		R-0x0

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

(#IMPLIED) Depending on mask.

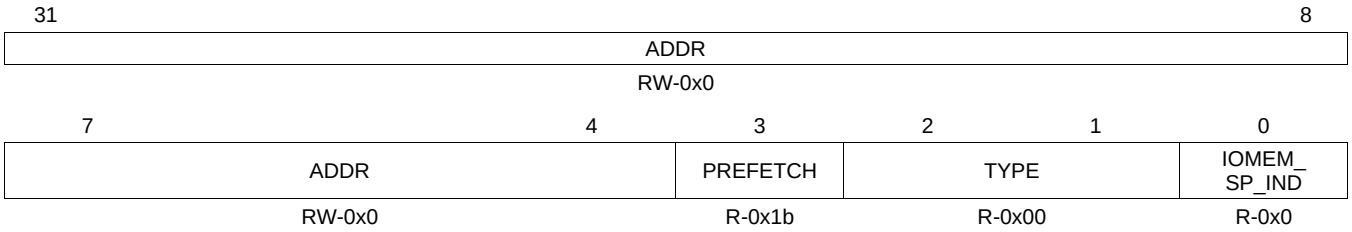
Table 15. Base Address Register 0 (PCIBAR0) Field Descriptions

Bit	Field	Value	Description
31-4	ADDR		Address bits. These bits can be written by the host to allow initialization of the base address at startup. The writeability of individual bits is determined by the corresponding bit in the Base Address 0 Mask Register in the Back End Configuration registers.
3	PREFETCH		Prefetchable bit. This bit specifies whether or not the memory space controlled by this base address register is prefetchable. This bit reflects the value that is input from the PREFETCH_EN bit of the Base Address 0 Mask Register in the Back End Configuration Registers.
2-1	TYPE		Type bits. These bits indicate the size of the base address register/decoder. This version of the PCI only supports 32-bit addressing, so these bits are hardwired to 00.
0	IOMEM_SP_IND		IO/ Memory Space Indicator bit. This bit indicates whether the base address maps into the host's memory or I/O space. This version of the PCI only supports memory mapped base address registers, so this bit is hardwired to a 0.

13.1.6 Base Address Register 1 (PCIBAR1)

PCI Configuration Base Address Register 1.

Figure 16. Base Address Register 1 (PCIBAR1)



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

(#IMPLIED) Depending on mask.

Table 16. Base Address Register 1 (PCIBAR1) Field Descriptions

Bit	Field	Value	Description
31-4	ADDR		Address bits. These bits can be written by the host to allow initialization of the base address at startup. The writeability of individual bits is determined by the corresponding bit in the Base Address 1 Mask Register in the Back End Configuration registers.
3	PREFETCH		Prefetchable bit. This bit specifies whether or not the memory space controlled by this base address register is prefetchable. This bit reflects the value that is input from the PREFETCH_EN bit of the Base Address 1 Mask Register in the Back End Configuration Registers.
2-1	TYPE		Type bits. These bits indicate the size of the base address register/decoder. This version of the PCI only supports 32-bit addressing, so these bits are hardwired to 00.
0	IOMEM_SP_IND		IO/ Memory Space Indicator bit. This bit indicates whether the base address maps into the host's memory or I/O space. This version of the PCI only supports memory mapped base address registers, so this bit is hardwired to a 0.

13.1.7 Base Address Register 2 (PCIBAR2)

PCI Configuration Base Address Register 2.

Figure 17. Base Address Register 2 (PCIBAR2)

31	8
ADDR	
RW-0x0	

7	4	3	2	1	0
ADDR		PREFETCH	TYPE		IOMEM_SP_IND
RW-0x0		R-0x1b	R-0x00		R-0x0

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

(#IMPLIED) Depending on mask.

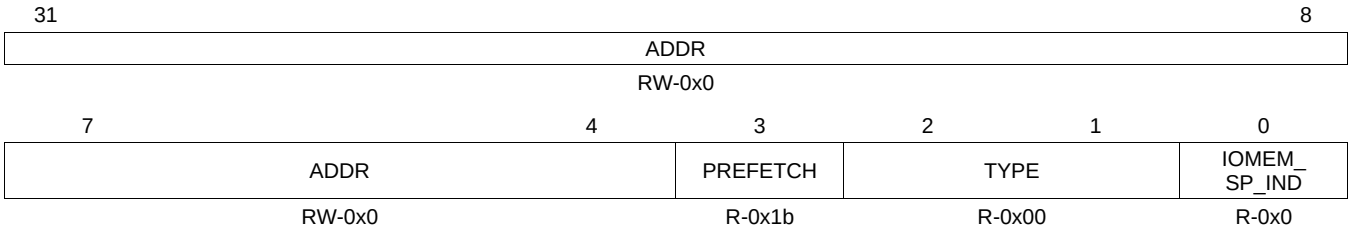
Table 17. Base Address Register 2 (PCIBAR2) Field Descriptions

Bit	Field	Value	Description
31-4	ADDR		Address bits. These bits can be written by the host to allow initialization of the base address at startup. The writeability of individual bits is determined by the corresponding bit in the Base Address 2 Mask Register in the Back End Configuration registers.
3	PREFETCH		Prefetchable bit. This bit specifies whether or not the memory space controlled by this base address register is prefetchable. This bit reflects the value that is input from the PREFETCH_EN bit of the Base Address 2 Mask Register in the Back End Configuration Registers.
2-1	TYPE		Type bits. These bits indicate the size of the base address register/decoder. This version of the PCI only supports 32-bit addressing, so these bits are hardwired to 00.
0	IOMEM_SP_IND		IO/ Memory Space Indicator bit. This bit indicates whether the base address maps into the host's memory or I/O space. This version of the PCI only supports memory mapped base address registers, so this bit is hardwired to a 0.

13.1.8 Base Address Register 3 (PCIBAR3)

PCI Configuration Base Address Register 3.

Figure 18. Base Address Register 3 (PCIBAR3)



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

(#IMPLIED) Depending on mask.

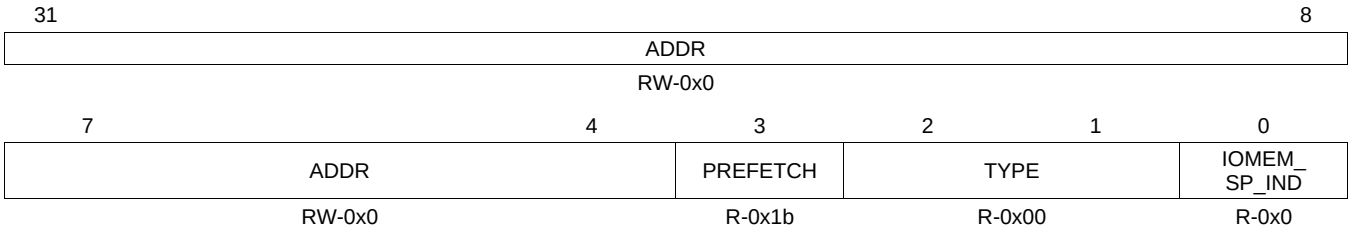
Table 18. Base Address Register 3 (PCIBAR3) Field Descriptions

Bit	Field	Value	Description
31-4	ADDR		Address bits. These bits can be written by the host to allow initialization of the base address at startup. The writeability of individual bits is determined by the corresponding bit in the Base Address 3 Mask Register in the Back End Configuration registers.
3	PREFETCH		Prefetchable bit. This bit specifies whether or not the memory space controlled by this base address register is prefetchable. This bit reflects the value that is input from the PREFETCH_EN bit of the Base Address 3 Mask Register in the Back End Configuration Registers.
2-1	TYPE		Type bits. These bits indicate the size of the base address register/decoder. This version of the PCI only supports 32-bit addressing, so these bits are hardwired to 00.
0	IOMEM_SP_IND		IO/ Memory Space Indicator bit. This bit indicates whether the base address maps into the host's memory or I/O space. This version of the PCI only supports memory mapped base address registers, so this bit is hardwired to a 0.

13.1.9 Base Address Register 4 (PCIBAR4)

PCI Configuration Base Address Register 4.

Figure 19. Base Address Register 4 (PCIBAR4)



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

(#IMPLIED) Depending on mask.

Table 19. Base Address Register 4 (PCIBAR4) Field Descriptions

Bit	Field	Value	Description
31-4	ADDR		Address bits. These bits can be written by the host to allow initialization of the base address at startup. The writeability of individual bits is determined by the corresponding bit in the Base Address 4 Mask Register in the Back End Configuration registers.
3	PREFETCH		Prefetchable bit. This bit specifies whether or not the memory space controlled by this base address register is prefetchable. This bit reflects the value that is input from the PREFETCH_EN bit of the Base Address 4 Mask Register in the Back End Configuration Registers.
2-1	TYPE		Type bits. These bits indicate the size of the base address register/decoder. This version of the PCI only supports 32-bit addressing, so these bits are hardwired to 00.
0	IOMEM_SP_IND		IO/ Memory Space Indicator bit. This bit indicates whether the base address maps into the host's memory or I/O space. This version of the PCI only supports memory mapped base address registers, so this bit is hardwired to a 0.

13.1.10 Base Address Register 5 (PCIBAR5)

PCI Configuration Base Address Register 5.

Figure 20. Base Address Register 5 (PCIBAR5)

31	8
ADDR	
RW-0x0	

7	4	3	2	1	0
ADDR		PREFETCH	TYPE		IOMEM_SP_IND
RW-0x0		R-0x1b	R-0x00		R-0x0

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

(#IMPLIED) Depending on mask.

Table 20. Base Address Register 5 (PCIBAR5) Field Descriptions

Bit	Field	Value	Description
31-4	ADDR		Address Bits. These bits can be written by the host to allow initialization of the base address at startup. The writeability of individual bits is determined by the corresponding bit in the Base Address 5 Mask Register in the Back End Configuration registers.
3	PREFETCH		Prefetchable bit. This bit specifies whether or not the memory space controlled by this base address register is prefetchable. This bit reflects the value that is input from the PREFETCH_EN bit of the Base Address 5 Mask Register in the Back End Configuration Registers.
2-1	TYPE		Type bits. These bits indicate the size of the base address register/decoder. This version of the PCI only supports 32-bit addressing, so these bits are hardwired to 00.
0	IOMEM_SP_IND		IO/memory space indicator bit. This bit indicates whether the base address maps into the host's memory or I/O space. This version of the PCI only supports memory mapped base address registers, so this bit is hardwired to a 0.

13.1.11 Subsystem Vendor ID/Subsystem ID Register (PCISUBID)

Subsystem Vendor ID/Subsystem ID

Figure 21. Subsystem Vendor ID/Subsystem ID Register (PCISUBID)

31	16
SUBSYS_ID	
R-0x0	

15	0
SUBSYS_VEN_ID	
R-0x0	

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

Table 21. Subsystem Vendor ID/Subsystem ID Register (PCISUBID) Field Descriptions

Bit	Field	Value	Description
31-16	SUBSYS_ID		Subsystem ID bits. Identifies the board level device. The Subsystem ID is specified by the board level manufacturer. The actual value is usually loaded through a serial EEPROM interface.
15-0	SUBSYS_VEN_ID		Subsystem vendor ID bits. Identifies the board level manufacturer. The Subsystem Vendor ID is specified by the PCI Special Interest Group. The actual value is usually loaded through a serial EEPROM interface.

13.1.12 Capabilities Pointer Register (PCICPBPTR)

Capabilities Pointer

Figure 22. Capabilities Pointer Register (PCICPBPTR)

31	16
Reserved	
R-0x0	

15	8	7	0
Reserved		CAP	
R-0x0		R-0x40	

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

Table 22. Capabilities Pointer Register (PCICPBPTR) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved		Reserved
7-0	CAP		Capabilities pointer bits. This constant specifies the address in configuration space where the first entry in the capabilities list is located.

13.1.13 Max Latency/Min Grant/Interrupt Pin/Interrupt Line Register (PCILGINT)

Set Max Latency/Min Grant

Figure 23. Max Latency/Min Grant/Interrupt Pin/Interrupt Line Register (PCILGINT)

31	24	23	16
MAX_LAT		MIN_GRNT	
R-0x0		R-0x0	

15	8	7	0
INT_PIN		INT_LINE	
R-0x1		RW-0x0	

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

Table 23. Max Latency/Min Grant/Interrupt Pin/Interrupt Line Register (PCILGINT) Field Descriptions

Bit	Field	Value	Description
31-24	MAX_LAT		Max latency bits. This specifies how often the device needs to gain access to the PCI bus in 0.25 μsec units. This field reflects the value which is input on the max latency port on the PCI module.
23-16	MIN_GRNT		Min grant bits. This specifies the length of the burst period for the device needs in 0.25 μsec units. This field reflects the value which is input from the min grant port on the PCI module.
15-8	INT_PIN		Interrupt pin bits. This register tells which interrupt pin the device uses. This register is hardwired to a 01h in the PCI to indicate that interrupt A will be used.
7-0	INT_LINE		Interrupt line bits. The value in this 8 bit register is written by the host and indicates to which input of the system interrupt controller the PCI interrupt pin is connected. This register is not cleared on software reset.

13.2 PCI Back End Configuration Registers

The back end configuration registers provide the back end application access to control and status information within the PCI. Portions of the back end configuration registers are located inside the peripheral clock domain and the remaining registers are located inside the PCI clock ($\overline{\text{PCLK}}$) domain. Table 9 notes which registers are located in the peripheral clock domain. Registers in the PCI clock domain can only be accessed if both the PCI and peripheral clocks are running. Registers in the peripheral clock domain can be accessed even if the PCI clock is not running.

13.2.1 Status Set Register (PCISTATSET)

The PCI includes an internal Status Register that is not directly writable by the DSP for software simplification. As a result, two registers, the Status Set Register and the Status Clear Register, are provided to set or clear bits in the internal Status Register. Writing a 1 to any of the bits in the Status Set Register will set the corresponding bit in the internal Status Register. The Status Set and Status Clear registers both return the contents of the internal PCI Status Register on reads.

Figure 24. Status Set Register (PCISTATSET)

31	30 28	27	26	25	24
DSPINT	Reserved	SOFT_INT3	SOFT_INT2	SOFT_INT1	SOFT_INT0
RW1S-0x0	R-0x0	RW1S-0x0	RW1S-0x0	RW1S-0x0	RW1S-0x0

23 8
Reserved
R-0x0

7	6	5	4 3	2	1	0
Reserved	PERR_DET	SERR_DET	Reserved	MS_ABRT_DET	TGT_ABRT_DET	Reserved
R-0x0	RW1S-0x0	RW1S-0x0	R-0x0	RW1S-0x0	RW1S-0x0	R-0x0

LEGEND: RW1S = Read/Write 1 to Set; R = Read only; *-n* = value after reset

Table 24. Status Set Register (PCISTATSET) Field Descriptions

Bit	Field	Value	Description
31	DSPINT		DSP interrupt bit. When the DSPINT bit of the internal Back End Application Interrupt Enable Register is set to 1 (PCIBINTSET), writing a 1 to this bit forces the host-to-DSP interrupt (DSPINT).
		0	A write of 0 is ignored
		1	Generate DSP interrupt
30-28	Reserved		Reserved
27-24	SOFT_INT[3:0]		Software interrupt set bits. Writing a 1 to these bits sets the corresponding software interrupt. If the corresponding bit in the PCI Host Interrupt Enable Register is also set to 1, an interrupt is generated to the host via the $\overline{\text{PINTA}}$ pin. The host-to-DSP interrupt (DSPINT) is also generated if the corresponding bit in the PCI Back End Application Interrupt Enable Register is set to 1.
		0	A write of 0 is ignored
		1	Generate software interrupt
23-7	Reserved		Reserved
6	PERR_DET		Parity error detect bit. Writing a 1 to this bit sets a parity error.
		0	A write of 0 is ignored
		1	Set parity error
5	SERR_DET		System error detect bit. Writing a 1 to this bit sets a system error.
		0	A write of 0 is ignored
		1	Set the system error
4-3	Reserved		Reserved
2	MS_ABRT_DET		Master abort detect bit. Writing a 1 to this bit sets master abort.
		0	A write of 0 is ignored
		1	Set the master abort error

Table 24. Status Set Register (PCISTATSET) Field Descriptions (continued)

Bit	Field	Value	Description
1	TGT_ABRT_DET		Target abort detect bit. Writing a 1 to this bit sets target abort.
		0	A write of 0 is ignored
		1	Set the target abort error
0	Reserved		Reserved

13.2.2 Status Clear Register (PCISTATCLR)

Writing a 1 to any of the bits in the Status Clear Register will clear the corresponding bit in the internal Status Register as long as the corresponding condition that sets that bit is not also asserted. Bits in the Status register cannot be cleared unless the underlying condition that caused the bit to be set has also been cleared. This is important for the $\overline{\text{PINTA}}$, $\overline{\text{PPERR}}$, and $\overline{\text{PSERR}}$ interrupts that are generated and accessed on a level sensitive external pin. Additionally, there is a synchronization delay of 3 peripheral clocks present between the time that a level sensitive status condition is cleared in the PCI clock domain and when that condition will be cleared in the peripheral clock domain. Interrupt service routines need to be designed to include this delay.

Figure 25. Status Clear Register (PCISTATCLR)

31	30 28	27	26	25	24
DSPINT	Reserved	SOFT_INT3	SOFT_INT2	SOFT_INT1	SOFT_INT0
RW1C-0x0	R-0x0	RW1C-0x0	RW1C-0x0	RW1C-0x0	RW1C-0x0

23 8
Reserved
R-0x0

7	6	5	4 3	2	1	0
Reserved	PERR_DET	SERR_DET	Reserved	MS_ABRT_DET	TGT_ABRT_DET	Reserved
R-0x0	RW1C-0x0	RW1C-0x0	R-0x0	RW1C-0x0	RW1C-0x0	R-0x0

LEGEND: RW1C = Read/Write 1 to Clear; R = Read only; -*n* = value after reset

Table 25. Status Clear Register (PCISTATCLR) Field Descriptions

Bit	Field	Value	Description
31	DSPINT		DSP interrupt bit. Writing a 1 to this bit clears the host-to-DSP interrupt (DSPINT).
		0	A write of 0 is ignored
		1	Clear DSP interrupt
30-28	Reserved		Reserved
27-24	SOFT_INT[3:0]		Software interrupt clear bits. Writing a 1 to these bits clears the corresponding software interrupt.
		0	A write of 0 is ignored
		1	Clear software interrupt
23-7	Reserved		Reserved
6	PERR_DET		Parity error detect bit. Writing 1 to this bit clears the parity error.
		0	A write of 0 is ignored
		1	Clears the parity error
5	SERR_DET		System error detect bit. Writing 1 to this bit clears the system error.
		0	A write of 0 is ignored
		1	Clears the system error
4-3	Reserved		Reserved
2	MS_ABRT_DET		Master abort detect bit. Writing 1 to this bit clears the master abort error.
		0	A write of 0 is ignored
		1	Clears the master abort error
1	TGT_ABRT_DET		Target abort detect bit. Writing 1 to this bit clears the target abort error.
		0	A write of 0 is ignored
		1	Clears the target abort error
0	Reserved		Reserved

13.2.3 Host Interrupt Enable Set Register (PCIHINTSET)

The PCI includes an internal Host Interrupt Enable Register which for software simplification is not directly writable by the host. As a result, two registers, the Host Interrupt Enable Set and Host Interrupt Enable Clear Registers are provided to set or clear bits in the internal Host Interrupt Enable Register.

The host uses the internal PCI Host Interrupt Enable Register to configure on which status conditions an interrupt will be generated to the host. A level-sensitive active-low interrupt is generated to the host on the $\overline{\text{PINTA}}$ pin if a bit in the internal PCI Status Register is asserted and the corresponding bit in the internal Host Interrupt Enable Register is also asserted, as long as the PCI is in the D0 power state. Interrupt generation on the $\overline{\text{PINTA}}$ pin is disabled whenever the PCI is in the D1, D2, or D3 power states. Reading this register will return the internal PCI Host Interrupt Enable register contents.

Writing a 1 to any of the bits in the Host Interrupt Enable Set Register will set the corresponding bit in the internal Host Interrupt Enable Register.

Figure 26. Host Interrupt Enable Set Register (PCIHINTSET)

31	28	27	26	25	24
Reserved		SOFT_INT3	SOFT_INT2	SOFT_INT1	SOFT_INT0
R-0x0		RW1S-0x0	RW1S-0x0	RW1S-0x0	RW1S-0x0

23	8
Reserved	
R-0x0	

7	6	5	4	3	2	1	0
Reserved	PERR_DET	SERR_DET	Reserved		MS_ABRT_DET	TGT_ABRT_DET	Reserved
R-0x0	RW1S-0x0	RW1S-0x0	R-0x0		RW1S-0x0	RW1S-0x0	R-0x0

LEGEND: RW1S = Read/Write 1 to Set; R = Read only; *-n* = value after reset

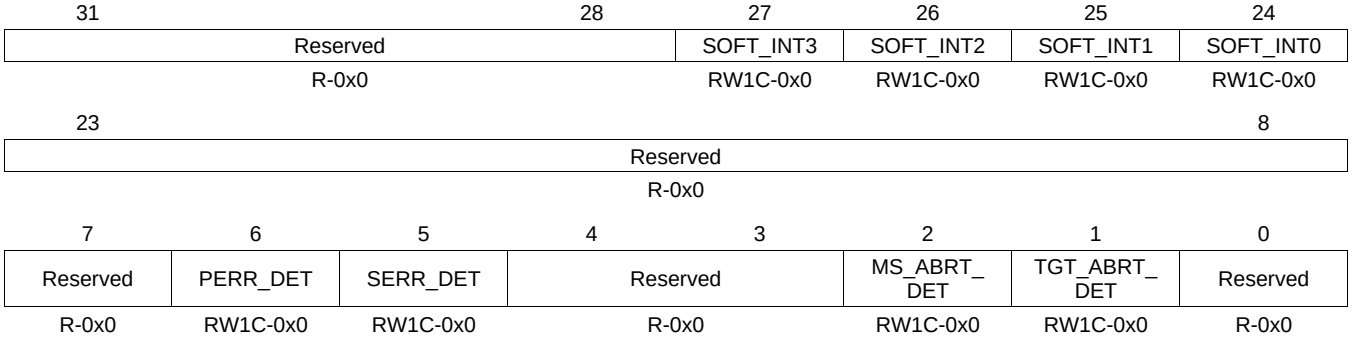
Table 26. Host Interrupt Enable Set Register (PCIHINTSET) Field Descriptions

Bit	Field	Value	Description
31-28	Reserved		Reserved
27-24	SOFT_INT[3:0]		Software interrupt enable bits. Writing a 1 to these bits enables the corresponding software interrupt. When the corresponding bit in the internal Status Register is set to 1, an interrupt is generated to the host via the $\overline{\text{PINTA}}$ pin.
		0	A write of 0 is ignored
		1	Enables software interrupt
23-7	Reserved		Reserved
6	PERR_DET		Parity error detect enable bit. Writing 1 to this bit enables the parity error detection.
		0	A write of 0 is ignored.
		1	Enables the parity error
5	SERR_DET		System error detect enable bit. Writing 1 to this bit enables the system error detection.
		0	A write of 0 is ignored.
		1	Enables the system error
4-3	Reserved		Reserved
2	MS_ABRT_DET		Master abort detect enable bit. Writing 1 to this bit enables the master abort detection.
		0	A write of 0 is ignored.
		1	Enables the master abort detection
1	TGT_ABRT_DET		Target abort detect enable bit. Writing 1 to this bit enables the target abort detection.
		0	A write of 0 is ignored.
		1	Enables the target abort detection
0	Reserved		Reserved

13.2.4 Host Interrupt Enable Clear Register (PCIHINTCLR)

Writing a 1 to any of the bits in the PCI Host Interrupt Enable Clear Register will clear the corresponding bit in the internal Host Interrupt Enable Register. Reading from this register will return the masked Host status that is the bitwise ANDing of the internal Status register and the internal Host Interrupt Enable register. This register is typically read by the host to determine the interrupt source when the $\overline{\text{PINTA}}$ pin is asserted.

Figure 27. Host Interrupt Enable Clear Register (PCIHINTCLR)



LEGEND: RW1C = Read/Write 1 to Clear; R = Read only; -*n* = value after reset

Table 27. Host Interrupt Enable Clear Register (PCIHINTCLR) Field Descriptions

Bit	Field	Value	Description
31-28	Reserved		Reserved
27-24	SOFT_INT[3:0]		Software interrupt disable bits. Writing a 1 to these bits disables the corresponding software interrupt.
		0	A write of 0 is ignored
		1	Disables software interrupt
23-7	Reserved		Reserved
6	PERR_DET		Parity error detect disable bit. Writing 1 to this bit disables the parity error detection.
		0	A write of 0 is ignored.
		1	Disables the parity error
5	SERR_DET		System error detect disable bit. Writing 1 to this bit disables the system error detection.
		0	A write of 0 is ignored.
		1	Disables the system error
4-3	Reserved		Reserved
2	MS_ABRT_DET		Master abort detect disable bit. Writing 1 to this bit disables the master abort detection.
		0	A write of 0 is ignored.
		1	Disables the master abort detection
1	TGT_ABRT_DET		Target abort detect disable bit. Writing 1 to this bit disables the target abort detection.
		0	A write of 0 is ignored.
		1	Disables the target abort detection
0	Reserved		Reserved

13.2.5 Back End Application Interrupt Enable Set Register (PCIBINTSET)

The PCI includes an internal Back End Application Interrupt Enable Register that is not directly writable by the back end application for software simplification. As a result, two registers, the Back End Application Interrupt Enable Set and the Back End Application Interrupt Enable Clear registers are provided to set or clear bits in the internal Back End Application Interrupt Enable Register.

The back end application uses the internal PCI Back End Application Interrupt Enable Register to configure on which status conditions an interrupt will be generated to the DSP. An interrupt request is generated to the DSP via the DSPINT interrupt line if a bit in the internal PCI Status Register is asserted and the corresponding bit in the internal Back End Application Interrupt Enable Register is also asserted. Reading this register will return the internal Back End Application Interrupt Enable register contents.

NOTE: Clearing the DSPINT bit of the internal Back End Application Interrupt Enable Register will not gate all interrupts to the DSP. An interrupt will be generated to the DSP via the DSPINT interrupt line if any bit in the internal Back End Application Interrupt Enable is set to 1 and the corresponding bit in the internal Status register is also set to 1.

Writing a 1 to any of the bits in the PCI Back End Application Interrupt Enable Set Register will set the corresponding bit in the PCI Back End Application Interrupt Enable Register.

Figure 28. Back End Application Interrupt Enable Set Register (PCIBINTSET)

31	30 28	27	26	25	24
DSPINT	Reserved	SOFT_INT3	SOFT_INT2	SOFT_INT1	SOFT_INT0
RW1S-0x0	R-0x0	RW1S-0x0	RW1S-0x0	RW1S-0x0	RW1S-0x0

23 8
Reserved
R-0x0

7	6	5	4 3	2	1	0
Reserved	PERR_DET	SERR_DET	Reserved	MS_ABRT_DET	TGT_ABRT_DET	Reserved
R-0x0	RW1S-0x0	RW1S-0x0	R-0x0	RW1S-0x0	RW1S-0x0	R-0x0

LEGEND: RW1S = Read/Write 1 to Set; R = Read only; *-n* = value after reset

Table 28. Back End Application Interrupt Enable Set Register (PCIBINTSET) Field Descriptions

Bit	Field	Value	Description
31	DSPINT		DSP interrupt enable bit. Writing a 1 to this bit enables the host-to-DSP interrupt (DSPINT).
		0	A write of 0 is ignored
		1	Enables the host-to-DSP interrupt (DSPINT)
30-28	Reserved		Reserved
27-24	SOFT_INT[3:0]		Software interrupt enable bits. Writing a 1 to these bits enables the corresponding software interrupt. When the corresponding bit in the internal Status Register is set to 1, an interrupt is generated to the DSP via the DSPINT interrupt line.
		0	A write of 0 is ignored
		1	Enables software interrupt
23-7	Reserved		Reserved
6	PERR_DET		Parity error detect enable bit. Writing 1 to this bit enables the parity error detection.
		0	A write of 0 is ignored
		1	Enables the parity error
5	SERR_DET		System error detect enable bit. Writing 1 to this bit enables the system error detection
		0	A write of 0 is ignored
		1	Enables the system error
4-3	Reserved		Reserved

Table 28. Back End Application Interrupt Enable Set Register (PCIBINTSET) Field Descriptions (continued)

Bit	Field	Value	Description
2	MS_ABRT_DET		Master abort detect enable bit. Writing 1 to this bit enables the master abort detection
		0	A write of 0 is ignored
		1	Enables the master abort detection
1	TGT_ABRT_DET		Target abort detect enable bit. Writing 1 to this bit enables the target abort detection.
		0	A write of 0 is ignored
		1	Enables the target abort detection
0	Reserved		Reserved

13.2.6 Back End Application Interrupt Enable Clear Register (PCIBINTCLR)

Writing a 1 to any of the bits in the PCI Back End Application Interrupt Enable Clear Register will clear the corresponding bit in the internal PCI Back End Application Interrupt Enable Register. Reading from this register will return the masked Back End Application status that is the bitwise ANDing of the internal PCI Status Register and the internal PCI Back End Application Interrupt Enable Register. This register is typically read by the Back End Application to determine the source of a back end interrupt.

Figure 29. Back End Application Interrupt Enable Clear Register (PCIBINTCLR)

31	30 28	27	26	25	24
DSPINT	Reserved	SOFT_INT3	SOFT_INT2	SOFT_INT1	SOFT_INT0
RW1C-0x0	R-0x0	RW1C-0x0	RW1C-0x0	RW1C-0x0	RW1C-0x0

23 8
Reserved
R-0x0

7	6	5	4 3	2	1	0
Reserved	PERR_DET	SERR_DET	Reserved	MS_ABRT_DET	TGT_ABRT_DET	Reserved
R-0x0	RW1C-0x0	RW1C-0x0	R-0x0	RW1C-0x0	RW1C-0x0	R-0x0

LEGEND: RW1C = Read/Write 1 to Clear; R = Read only; -*n* = value after reset

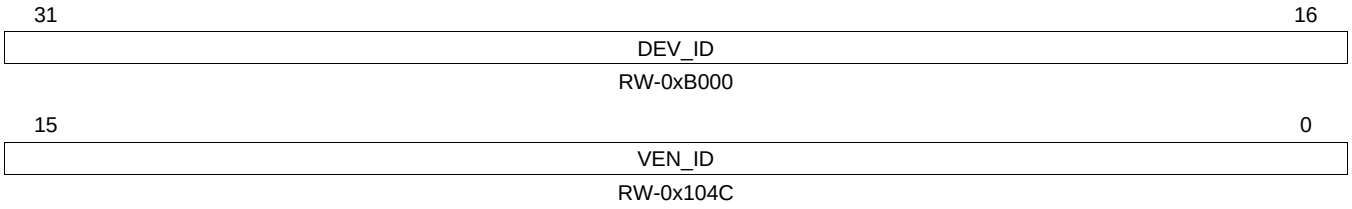
Table 29. Back End Application Interrupt Enable Clear Register (PCIBINTCLR) Field Descriptions

Bit	Field	Value	Description
31	DSPINT		DSP interrupt disable bit. Writing a 1 to this bit disables the host-to-DSP interrupt (DSPINT).
			NOTE: Clearing the DSPINT bit of the internal Back End Application Interrupt Enable Register will not gate all interrupts to the DSP. An interrupt will be generated to the DSP via the DSPINT interrupt line if any bit in the internal Back End Application Interrupt Enable is set to 1 and the corresponding bit in the internal Status register is also set to 1.
		0	A write of 0 is ignored
		1	Disables the host-to-DSP interrupt (DSPINT)
30-28	Reserved		Reserved
27-24	SOFT_INT[3:0]		Software interrupt disable bits. Writing a 1 to these bits disables the corresponding software interrupt.
		0	A write of 0 is ignored
		1	Disables software interrupt
23-7	Reserved		Reserved
6	PERR_DET		Parity error detect disable bit. Writing 1 to this bit disables the parity error detection.
		0	A write of 0 is ignored
		1	Disables the parity error
5	SERR_DET		System error detect disable bit. Writing 1 to this bit disables the system error detection
		0	A write of 0 is ignored
		1	Disables the system error
4-3	Reserved		Reserved
2	MS_ABRT_DET		Master abort detect disable bit. Writing 1 to this bit disables the master abort detection
		0	A write of 0 is ignored
		1	Disables the master abort detection
1	TGT_ABRT_DET		Target abort detect disable bit. Writing 1 to this bit disables the target abort detection.
		0	A write of 0 is ignored
		1	Disables the target abort detection
0	Reserved		Reserved

13.2.7 Vendor ID/Device ID Mirror Register (PCIVENDEVMIR)

This register is used to initialize the Vendor ID and/or Device ID in the Configuration Space Vendor ID/Device ID register (PCIVENDEV) prior to enabling configuration accesses. This register is typically written by either an EEPROM controller or an on-chip CPU.

Figure 30. Vendor ID/Device ID Mirror Register (PCIVENDEVMIR)



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

Table 30. Vendor ID/Device ID Mirror Register (PCIVENDEVMIR) Field Descriptions

Bit	Field	Value	Description
31-16	DEV_ID		Device ID bits. Identifies a specific device from the manufacturer. The Device ID is specified by the device manufacturer.
15-0	VEN_ID		Vendor ID bits. Uniquely identifies the manufacturer of the device. The Vendor ID is specified by the PCI Special Interest Group to insure uniqueness.

13.2.8 Command/Status Mirror Register (PCICSRMIR)

This register initializes certain bits in the Configuration Space Command/Status register (PCICSR) prior to enabling configuration accesses. These bits are typically written by either an EEPROM controller or an on-chip CPU. This register can also be used by the back end application to directly query the status of the PCI.

Figure 31. Command/Status Mirror Register (PCICSRMIR)

31	30	29	28	27	26 25	24
DET_PAR_ERR	SIG_SYS_ERR	RCV_MS_ABRT	RCV_TGT_ABRT	SIG_TGT_ABRT	DEVSEL_TIM	MS_DPAR_REP
RW1C-0x0	RW1C-0x0	RW1C-0x0	RW1C-0x0	R-0x0	R-0x1b	RW1C-0x0

23	22	21	20	19	18 16
FAST_BTOB_CAP	Reserved	66MHZ_CAP	CAP_LIST_IMPL	INT_STAT	Reserved
R-0x0	R-0x0	RW-Undefined	RW-0x0	R-0x0	R-0x0

15 11	10	9	8
Reserved	INT_DIS	FAST_BTOB_EN	SERR_N_EN
R-0x0	RW-0x0	R-0x0	RW-0x0

7	6	5	4	3	2	1	0
Reserved	PAR_ERR_RES	VGA_PAL_SNP	MEM_WRINV_EN	SP_CYCL	BUS_MS	MEM_SP	IO_SP
R-0x0	RW-0x0	R-0x0	RW-0x0	R-0x0	RW-0x0	RW-0x0	R-0x0

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

Table 31. Command/Status Mirror Register (PCICSRMIR) Field Descriptions

Bit	Field	Value	Description
31	DET_PAR_ERR		Detected parity error bit. This bit is set by the PCI to indicate that it detected a parity error, which was not necessarily reported on $\overline{\text{PPERR}}$ if parity reporting is disabled.
30	SIG_SYS_ERR		Signaled system error bit. This bit is set by the PCI to indicate that it signaled a system error on the $\overline{\text{PSERR}}$ pin.
29	RCV_MS_ABRT		Received master abort bit. This bit is set by the PCI master unit in the PCI to indicate that it terminated a transaction with a master abort.
28	RCV_TGT_ABRT		Received target abort bit. This bit is set by the PCI master unit in the PCI to indicate that it has received a target abort when acting as a bus master.
27	SIG_TGT_ABRT		Signaled target abort bit. This bit is always 0 because the PCI cannot issue a target abort.
26-25	DEVSEL_TIM		DEVSEL timing bits. This bit indicates the decode response time capability of the device. The PCI decode logic supports medium DEVSEL_N timing; therefore, these bits are hardwired to 01.
24	MS_DPAR_REP		Master data parity reported bit. This bit is set by the PCI master unit in the PCI when all of the following conditions are met: 1. The PCI asserted $\overline{\text{PPERR}}$ or observed $\overline{\text{PPERR}}$ asserted 2. The PCI master unit was the bus master during the observed $\overline{\text{PPERR}}$ assertion 3. The Parity Error Response bit (PAR_ERR_RES) is set
23	FAST_BTOB_CAP		Fast back to back capable bit. This bit indicates that the device is capable of performing fast back to back transactions. The PCI does not support fast back to back transactions; therefore, this bit is hardwired to 0.
22	Reserved		Reserved
21	66MHZ_CAP		66MHz capable bit. This bit indicates whether or not the interface is capable of meeting the 66MHz PCI timing requirements.
20	CAP_LIST_IMPL		Capabilities list implemented bit. This bit indicates whether or not the interface provides at least one capabilities list.
19	INT_STAT		Interrupt status bit. This bit indicates the current interrupt status for the function as generated by the interrupt registers in the back end interface. If this bit is set and INT_DIS is cleared, the $\overline{\text{PINTA}}$ pin will be asserted low. INT_DIS has no effect on the value of this bit.
18-11	Reserved		Reserved. Always return 0.
10	INT_DIS		Interrupt disable bit. This bit controls whether or not the device can assert the $\overline{\text{PINTA}}$ pin. This bit is a disable for the output driver on the $\overline{\text{PINTA}}$ pin. If this bit is set, the interrupt condition will not appear on the external $\overline{\text{PINTA}}$ pin.

Table 31. Command/Status Mirror Register (PCICSRMIR) Field Descriptions (continued)

Bit	Field	Value	Description
9	FAST_BTOB_EN		Fast back to back enable bit. This bit controls whether or not the device is allowed to perform back to back writes to different targets. The PCI will not perform fast back to back transactions; therefore, this bit is hardwired to a 0.
8	SERR_N_EN		SERR_N enable bit. This bit is an enable for the output driver on the $\overline{\text{PSERR}}$ pin. If this bit is cleared, and a system error condition is set inside the PCI, the error signal will not appear on the external $\overline{\text{PSERR}}$ pin.
7	Reserved		Reserved
6	PAR_ERR_RES		Parity error response bit. This bit controls whether or not the device responds to detected parity errors. If this bit is set, the PCI will respond normally to parity errors. If this bit is cleared, the PCI will set its Detected Parity Error status bit (DET_PAR_ERR) when an error is detected, but does not assert $\overline{\text{PPERR}}$ and continues normal operation.
5	VGA_PAL_SNP		VGA Palette Snoop bit. This bit is not generally applicable for the PCI and is hardwired to a 0.
4	MEM_WRINV_EN		Memory write and invalidate enable bit. This bit enables the device to use the Memory Write and Invalidate command. If this bit is cleared, the PCI will not attempt to use the Memory Write and Invalidate command.
3	SP_CYCL		Special cycle bit. This bit controls the device's response to special cycle commands. If this bit is cleared, the device will ignore all special cycle commands. If this bit is set to 1, the device can monitor special cycle commands.
2	BUS_MS		Bus master bit. This bit enables the device to act as a PCI bus master. If this bit is cleared, the device will not act as a master on the PCI bus.
1	MEM_SP		Memory access bit. This bit enables the device to respond to memory accesses within its address space. If this bit is cleared, the PCI will not respond to memory mapped accesses.
0	IO_SP		IO access bit. This bit enables the device to respond to I/O accesses within its address space. The PCI does not support IO accesses as a slave; therefore, this bit is hardwired to 0.

13.2.9 Class Code/Revision ID Mirror Register (PCICLREVMIR)

This register is used to initialize the Class Code and / or Revision ID in the PCI Configuration Space Class Code / Revision ID register (PCICLREV) prior to enabling configuration accesses. This register is typically written by either an EEPROM controller or an on-chip CPU.

Figure 32. Class Code/Revision ID Mirror Register (PCICLREVMIR)

31	16
CL_CODE	
RW-Undefined	

15	8	7	0
CL_CODE		REV_ID	
RW-Undefined		RW-Undefined	

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

Table 32. Class Code/Revision ID Mirror Register (PCICLREVMIR) Field Descriptions

Bit	Field	Value	Description
31-8	CL_CODE		Class Code bits. Identifies the generic and specific function and programming interface of the device. The Class Code is specified by the device manufacturer.
7-0	REV_ID		Revision ID bits. Identifies a revision of the specific device. The Revision ID is specified by the device manufacturer.

13.2.10 BIST/Header Type/Latency Timer/Cacheline Size Mirror Register (PCICLINEMIR)

This register is used to set latency timer and cacheline size.

Figure 33. BIST/Header Type/Latency Timer/Cacheline Size Mirror Register (PCICLINEMIR)

31	24	23	16
BIST		HDR_TYPE	
R-0x0		R-0x0	

15	8	7	0
LAT_TMR		CACHELN_SIZ	
RW-0x0		RW-0x0	

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

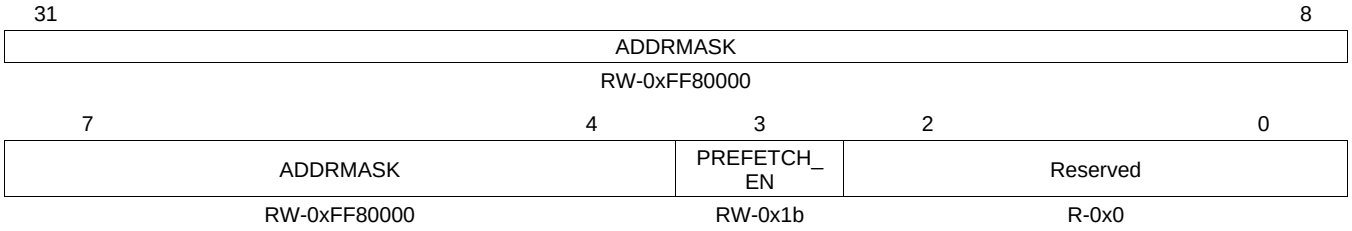
Table 33. BIST/Header Type/Latency Timer/Cacheline Size Mirror Register (PCICLINEMIR) Field Descriptions

Bit	Field	Value	Description
31-24	BIST		Built-in self test bits. Hardwired to 0.
23-16	HDR_TYPE		Header type bits. Identifies the layout of bytes 10h through 3Fh, and if the device is single or multi-function.
15-8	LAT_TMR		Latency timer bits. The latency timer register is provided so that the host can restrict the continued usage of the PCI bus by a master involved in a multiple data cycle transaction after its $\overline{\text{PGNT}}$ has been removed. The host is required to write a value into this register indicating the maximum number of PCI cycles for which the master can hold the bus (beginning from the assertion of $\overline{\text{PFRAME}}$). If $\overline{\text{PGNT}}$ is never removed during the transaction, the value in the latency timer value will not be used. Since the PCI will support transactions with multiple data cycles, the latency timer register is implemented. The latency timer register is initialized with all zeroes at reset. This register is not cleared on software reset.
7-0	CACHELN_SIZ		Cache line size bits. This register is provided so that the host can inform the device of the cache line size in units of 32 bit words. The value of this register is used by the PCI as a master device to determine whether to use Memory Write, Memory Write and Invalidate, Read, Read Line, or Read Multiple commands for accessing memory. This register is also used by the slave state machine to determine the size of prefetches that are performed on the Slave Back End Interface. Supported values for this register are as follows:
		00h	Disabled
		04h	Cache Line is 16 bytes
		08h	Cache Line is 32 bytes
		10h	Cache Line is 64 bytes
		20h	Cache Line is 128 bytes
			Writing an unsupported value to this register will result in the value being set to 0, as specified in the PCI Local Bus Specification.

13.2.11 Base Address Mask Register 0 (PCIBAR0MSK)

The Base Address Mask Registers control the size and prefetchability of the PCI Configuration Base Address 0 Register.

Figure 34. Base Address Mask Register 0 (PCIBAR0MSK)



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

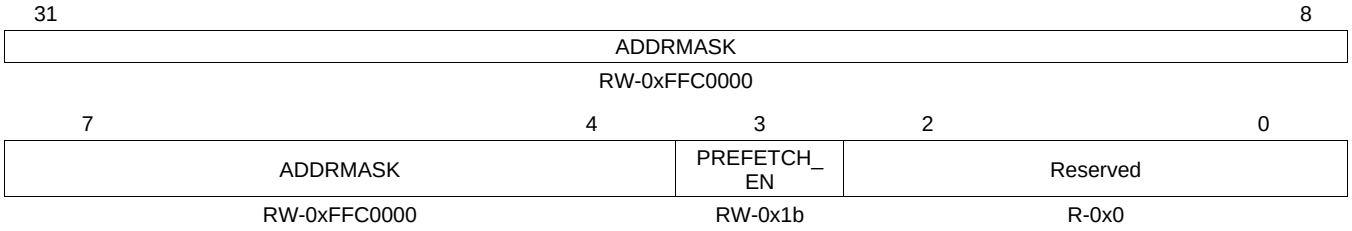
Table 34. Base Address Mask Register 0 (PCIBAR0MSK) Field Descriptions

Bit	Field	Value	Description
31-4	ADDRMASK		Address mask bits. These bits control the writeability of the corresponding bits in the corresponding PCI Configuration Base Address Register.
3	PREFETCH_EN		Prefetchable enable bit. This bit specifies whether or not the memory space controlled by the corresponding PCI Configuration Base Address Register is prefetchable. This bit is reflected in the PREFETCH bit of the corresponding PCI Configuration Base Address Register.
2-0	Reserved		Reserved

13.2.12 Base Address Mask Register 1 (PCIBAR1MSK)

The Base Address Mask Registers control the size and prefetchability of the PCI Configuration Base Address 1 Register.

Figure 35. Base Address Mask Register 1 (PCIBAR1MSK)



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

Table 35. Base Address Mask Register 1 (PCIBAR1MSK) Field Descriptions

Bit	Field	Value	Description
31-4	ADDRMASK		Address mask bits. These bits control the writeability of the corresponding bits in the corresponding PCI Configuration Base Address Register.
3	PREFETCH_EN		Prefetchable enable bit. This bit specifies whether or not the memory space controlled by the corresponding PCI Configuration Base Address Register is prefetchable. This bit is reflected in the PREFETCH bit of the corresponding PCI Configuration Base Address Register.
2-0	Reserved		Reserved

13.2.13 Base Address Mask Register 2 (PCIBAR2MSK)

The Base Address Mask Registers control the size and prefetchability of the PCI Configuration Base Address Registers

Figure 36. Base Address Mask Register 2 (PCIBAR2MSK)

31	8
ADDRMASK	
RW-0xFF80000	

7	4	3	2	0
ADDRMASK		PREFETCH_EN	Reserved	
RW-0xFF80000		RW-0x1b	R-0x0	

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

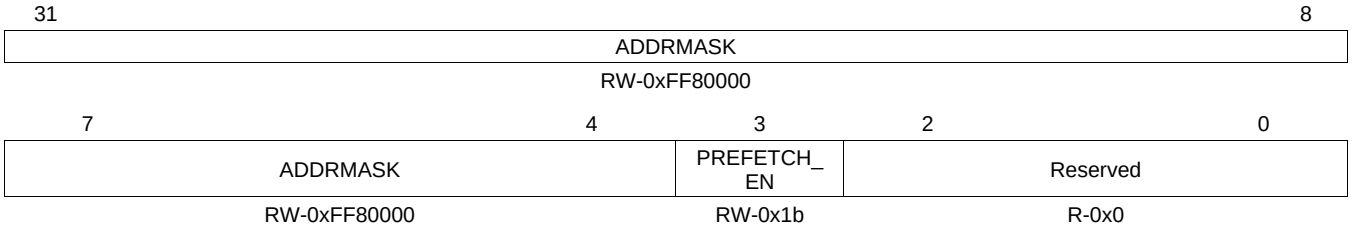
Table 36. Base Address Mask Register 2 (PCIBAR2MSK) Field Descriptions

Bit	Field	Value	Description
31-4	ADDRMASK		Address mask bits. These bits control the writeability of the corresponding bits in the corresponding PCI Configuration Base Address Register.
3	PREFETCH_EN		Prefetchable enable bit. This bit specifies whether or not the memory space controlled by the corresponding PCI Configuration Base Address Register is prefetchable. This bit is reflected in the PREFETCH bit of the corresponding PCI Configuration Base Address Register.
2-0	Reserved		Reserved

13.2.14 Base Address Mask Register 3 (PCIBAR3MSK)

The Base Address Mask Registers control the size and prefetchability of the PCI Configuration Base Address 3 Register.

Figure 37. Base Address Mask Register 3 (PCIBAR3MSK)



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

Table 37. Base Address Mask Register 3 (PCIBAR3MSK) Field Descriptions

Bit	Field	Value	Description
31-4	ADDRMASK		Address mask bits. These bits control the writeability of the corresponding bits in the corresponding PCI Configuration Base Address Register.
3	PREFETCH_EN		Prefetchable enable bit. This bit specifies whether or not the memory space controlled by the corresponding PCI Configuration Base Address Register is prefetchable. This bit is reflected in the PREFETCH bit of the corresponding PCI Configuration Base Address Register.
2-0	Reserved		Reserved

13.2.15 Base Address Mask Register 4 (PCIBAR4MSK)

The Base Address Mask Registers control the size and prefetchability of the PCI Configuration Base Address 4 Register.

Figure 38. Base Address Mask Register 4 (PCIBAR4MSK)

31	8
ADDRMASK	
RW-0xFF80000	

7	4	3	2	0
ADDRMASK		PREFETCH_EN	Reserved	
RW-0xFF80000		RW-0x1b	R-0x0	

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

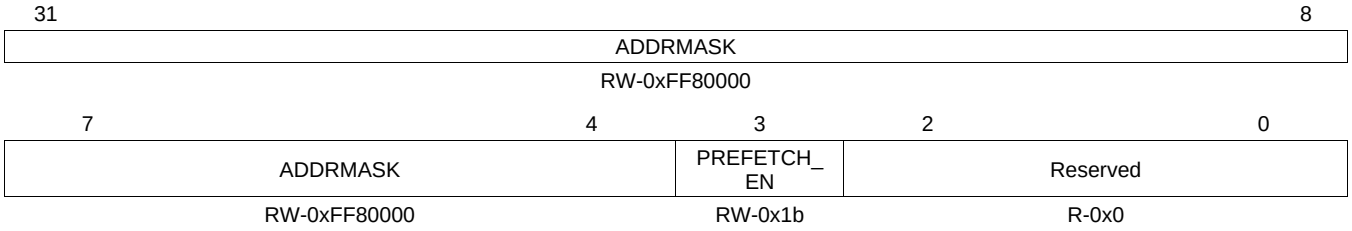
Table 38. Base Address Mask Register 4 (PCIBAR4MSK) Field Descriptions

Bit	Field	Value	Description
31-4	ADDRMASK		Address mask bits. These bits control the writeability of the corresponding bits in the corresponding PCI Configuration Base Address Register.
3	PREFETCH_EN		Prefetchable enable bit. This bit specifies whether or not the memory space controlled by the corresponding PCI Configuration Base Address Register is prefetchable. This bit is reflected in the PREFETCH bit of the corresponding PCI Configuration Base Address Register.
2-0	Reserved		Reserved

13.2.16 Base Address Mask Register 5 (PCIBAR5MSK)

The Base Address Mask Registers control the size and prefetchability of the PCI Configuration Base Address 5 Register.

Figure 39. Base Address Mask Register 5 (PCIBAR5MSK)



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

Table 39. Base Address Mask Register 5 (PCIBAR5MSK) Field Descriptions

Bit	Field	Value	Description
31-4	ADDRMASK		Address mask bits. These bits control the writeability of the corresponding bits in the corresponding PCI Configuration Base Address Register.
3	PREFETCH_EN		Prefetchable enable bit. This bit specifies whether or not the memory space controlled by the corresponding PCI Configuration Base Address Register is prefetchable. This bit is reflected in the PREFETCH bit of the corresponding PCI Configuration Base Address Register.
2-0	Reserved		Reserved

13.2.17 Subsystem Vendor ID/Subsystem ID Mirror Register (PCISUBIDMIR)

Subsystem Vendor ID/Subsystem ID

Figure 40. Subsystem Vendor ID/Subsystem ID Mirror Register (PCISUBIDMIR)

31	16
SUBSYS_ID	
RW-0x0	

15	0
SUBSYS_VEN_ID	
RW-0x0	

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

Table 40. Subsystem Vendor ID/Subsystem ID Mirror Register (PCISUBIDMIR) Field Descriptions

Bit	Field	Value	Description
31-16	SUBSYS_ID		Subsystem ID bits. Identifies the board level device. The Subsystem ID is specified by the board level manufacturer.
15-0	SUBSYS_VEN_ID		Subsystem Vendor ID bits. Identifies the board level manufacturer. The Subsystem Vendor ID is specified by the PCI Special Interest Group.

13.2.18 Capabilities Pointer Mirror Register (PCICPBPTRMIR)

Capabilities Pointer mirror register.

Figure 41. Capabilities Pointer Mirror Register (PCICPBPTRMIR)

31	16
Reserved	
R-0x0	

15	8	7	0
Reserved		CAP	
R-0x0		R-0x40	

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

Table 41. Capabilities Pointer Mirror Register (PCICPBPTRMIR) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved		Reserved
7-0	CAP		Capabilities pointer bits. This constant specifies the address in configuration space where the first entry in the capabilities list is located.

13.2.19 Max Latency/Min Grant/Interrupt Pin/Interrupt Line Mirror Register (PCILGINTMIR)

Set Max Latency/Min Grant

Figure 42. Max Latency/Min Grant/Interrupt Pin/Interrupt Line Mirror Register (PCILGINTMIR)

31	24	23	16
MAX_LAT		MIN_GRNT	
RW-0x0		RW-0x0	

15	8	7	0
INT_PIN		INT_LINE	
R-0x1		RW-0x0	

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

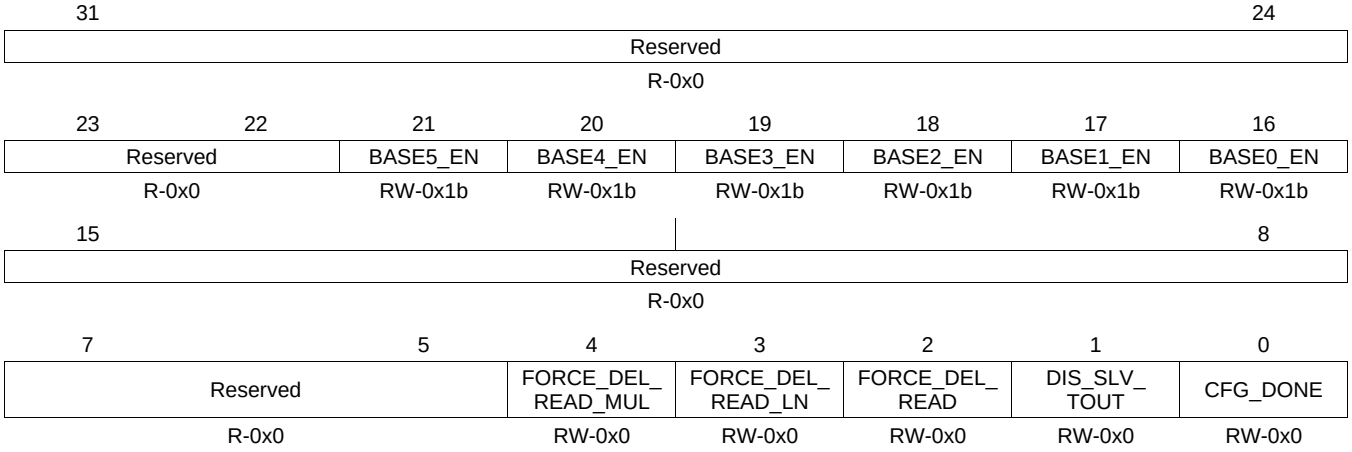
Table 42. Max Latency/Min Grant/Interrupt Pin/Interrupt Line Mirror Register (PCILGINTMIR) Field Descriptions

Bit	Field	Value	Description
31-24	MAX_LAT		Maximum latency bits. This specifies how often the device needs to gain access to the PCI bus in 0.25 μsec units.
23-16	MIN_GRNT		Minimum grant bits. This specifies the length of the burst period for the device in 0.25 μsec units.
15-8	INT_PIN		Interrupt pin bits. This bit tells which interrupt pin the device uses. This register is hardwired to 01h in the PCI to indicate that interrupt A will be used.
7-0	INT_LINE		Interrupt line bits. The value in these 8 bits is written by the host and indicates to which input of the system interrupt controller the PCI interrupt pin is connected.

13.2.20 Slave Control Register (PCISLVCNTL)

Slave Control.

Figure 43. Slave Control Register (PCISLVCNTL)



31–24	23–22	21	20	19	18	17	16
Reserved	Reserved	BASE5_EN	BASE4_EN	BASE3_EN	BASE2_EN	BASE1_EN	BASE0_EN
R-0x0	R-0x0	RW-0x1b	RW-0x1b	RW-0x1b	RW-0x1b	RW-0x1b	RW-0x1b

15–8	7–5	4	3	2	1	0
Reserved	Reserved	FORCE_DEL_READ_MUL	FORCE_DEL_READ_LN	FORCE_DEL_READ	DIS_SLV_TOUT	CFG_DONE
R-0x0	R-0x0	RW-0x0	RW-0x0	RW-0x0	RW-0x0	RW-0x0

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

Table 43. Slave Control Register (PCISLVCNTL) Field Descriptions

Bit	Field	Value	Description
31-22	Reserved		Reserved
21	BASE5_EN		Base5 enable bit. This bit enables/disables the BASE 5 Configuration.
		0	Disable BASE 5 Configuration
		1	Enable BASE 5 Configuration
20	BASE4_EN		Base4 enable bit. This bit enables/disables the BASE 4 Configuration.
		0	Disable BASE 4 Configuration
		1	Enable BASE 4 Configuration
19	BASE3_EN		Base3 enable bit. This bit enables/disables the BASE 3 Configuration.
		0	Disable BASE 3 Configuration
		1	Enable BASE 3 Configuration
18	BASE2_EN		Base2 enable bit. This bit enables/disables the BASE 2 Configuration.
		0	Disable BASE 2 Configuration
		1	Enable BASE 2 Configuration
17	BASE1_EN		Base1 enable bit. This bit enables/disables the BASE 1 Configuration.
		0	Disable BASE 1 Configuration
		1	Enable BASE 1 Configuration
16	BASE0_EN		Base0 enable bit. This bit enables/disables the BASE 0 Configuration.
		0	Disable BASE 0 Configuration
		1	Enable BASE 0 Configuration
15-5	Reserved		Reserved
4	FORCE_DEL_READ_MUL		Force Delayed Read Multiple bit.
		0	Slave should respond with normal 16 clock cycle timeout and retry mechanism for Memory Read Multiple transactions
		1	Slave should immediately respond with a retry whenever a Memory Read Multiple transaction is decoded for this slave. This bit overrides the DIS_SLV_TOUT bit for Memory Read Multiple transactions.

Table 43. Slave Control Register (PCISLVCNTL) Field Descriptions (continued)

Bit	Field	Value	Description
3	FORCE_DEL_READ_LN		Force Delayed Read Line bit.
		0	Slave should respond with normal 16 clock cycle timeout and retry mechanism for Memory Read Line transactions
		1	Slave should immediately respond with a retry whenever a Memory Read Line transaction is decoded for this slave. This bit overrides the DIS_SLV_TOUT bit for Memory Read Line transactions.
2	FORCE_DEL_READ		Force Delayed Read bit.
		0	Slave should respond with normal 16 clock cycle timeout and retry mechanism for Memory Read transactions
		1	Slave should immediately respond with a retry whenever a Memory Read transaction is decoded for this slave. This bit overrides DIS_SLV_TOUT for Memory Read transactions.
1	DIS_SLV_TOUT		Disable Slave timeout bit.
		0	Slave responds with normal 16 clock cycle timeout mechanism
		1	Slave will insert wait states on the PCI bus indefinitely until the access is ready to complete
0	CFG_DONE		Configuration done bit. Indicates if the Configuration Registers have been loaded with their proper reset values.
		0	Configuration Registers are being loaded. No access allowed into PCI interface.
		1	Configuration Registers loading is complete. PCI interface will accept accesses.

13.2.21 Slave Base Address Translation Registers (0x1C0 – 0x1D4)

These registers control the translation of transaction addresses as they flow from the PCI bus to the back end interface. The translation registers are programmed with a value that replaces the most significant portion of the PCI address as it is converted to a DSP address. The Slave Base Address Translation Registers are formatted as follows.

13.2.21.1 Slave Base Address 0 Translation Register (PCIBAR0TRL)

Figure 44. Slave Base Address 0 Translation Register (PCIBAR0TRL)

31	16
TRANS_ADDR	
RW-0x0080000	

15	4	3	0
TRANS_ADDR		Reserved	
RW-0x0080000		R-0x0	

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

Table 44. Slave Base Address 0 Translation Register (PCIBAR0TRL) Field Descriptions

Bit	Field	Value	Description
31-4	TRANS_ADDR		Translation address bits. This is the address that is to bitwise replace the slave transaction address. This is multiplexed on a bitwise basis with the original PCI address using the corresponding PCI Base Address Mask register bits 31:4 (ADDRMASK) as the multiplexer select.
3-0	Reserved		Reserved

13.2.21.2 Slave Base Address 1 Translation Register (PCIBAR1TRL)

Figure 45. Slave Base Address 1 Translation Register (PCIBAR1TRL)

31	16
TRANS_ADDR	
RW-0x0180000	

15	4	3	0
TRANS_ADDR		Reserved	
RW-0x0180000		R-0x0	

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

Table 45. Slave Base Address 1 Translation Register (PCIBAR1TRL) Field Descriptions

Bit	Field	Value	Description
31-4	TRANS_ADDR		Translation address bits. This is the address that is to bitwise replace the slave transaction address. This is multiplexed on a bitwise basis with the original PCI address using the corresponding Base Address Mask register bits 31:4 (ADDRMASK) as the multiplexer select.
3-0	Reserved		Reserved

13.2.21.3 Slave Base Address 2 Translation Register (PCIBAR2TRL)

Figure 46. Slave Base Address 2 Translation Register (PCIBAR2TRL)

31	16
TRANS_ADDR	
RW-0x0280000	

15	4	3	0
TRANS_ADDR		Reserved	
RW-0x0280000		R-0x0	

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

Table 46. Slave Base Address 2 Translation Register (PCIBAR2TRL) Field Descriptions

Bit	Field	Value	Description
31-4	TRANS_ADDR		Translation address bits. This is the address that is to bitwise replace the slave transaction address. This is multiplexed on a bitwise basis with the original PCI address using the corresponding Base Address Mask register bits 31:4 (ADDRMASK) as the multiplexer select.
3-0	Reserved		Reserved

13.2.21.4 Slave Base Address 3 Translation Register (PCIBAR3TRL)

Figure 47. Slave Base Address 3 Translation Register (PCIBAR3TRL)

31	16
TRANS_ADDR	
RW-0x8000000	

15	4	3	0
TRANS_ADDR		Reserved	
RW-0x8000000		R-0x0	

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

(#IMPLIED) This is not a supported memory address range in TCI648x devices and needs to be reprogrammed before being used.

Table 47. Slave Base Address 3 Translation Register (PCIBAR3TRL) Field Descriptions

Bit	Field	Value	Description
31-4	TRANS_ADDR		Translation address bit. This is the address that is to bitwise replace the slave transaction address. This is multiplexed on a bitwise basis with the original PCI address using the corresponding Base Address Mask register bits 31:4 (ADDRMASK) as the multiplexer select.
3-0	Reserved		Reserved

13.2.21.5 Slave Base Address 4 Translation Register (PCIBAR4TRL)

Figure 48. Slave Base Address 4 Translation Register (PCIBAR4TRL)

31	16
TRANS_ADDR	
RW-0xA000000	

15	4	3	0
TRANS_ADDR		Reserved	
RW-0xA000000		R-0x0	

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

Table 48. Slave Base Address 4 Translation Register (PCIBAR4TRL) Field Descriptions

Bit	Field	Value	Description
31-4	TRANS_ADDR		Translation address bit. This is the address that is to bitwise replace the slave transaction address. This is multiplexed on a bitwise basis with the original PCI address using the corresponding Base Address Mask register bits 31:4 (ADDRMASK) as the multiplexer select.
3-0	Reserved		Reserved

13.2.21.6 Slave Base Address 5 Translation Register (PCIBAR5TRL)

Figure 49. Slave Base Address 5 Translation Register (PCIBAR5TRL)

31	16
TRANS_ADDR	
RW-0xE000000	

15	4	3	0
TRANS_ADDR		Reserved	
RW-0xE000000		R-0x0	

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

Table 49. Slave Base Address 5 Translation Register (PCIBAR5TRL) Field Descriptions

Bit	Field	Value	Description
31-4	TRANS_ADDR		Translation address bit. This is the address that is to bitwise replace the slave transaction address. This is multiplexed on a bitwise basis with the original PCI address using the corresponding Base Address Mask register bits 31:4 (ADDRMASK) as the multiplexer select.
3-0	Reserved		Reserved

13.2.22 Base Address *n* Mirror Register (PCIBAR*n*MIR)

Base Address Mirror Register 0-5.

Figure 50. Base Address *n* Mirror Register (PCIBAR*n*MIR)

31	8
ADDR	
RW-0x0	

7	4	3	2	1	0
ADDR		PREFETCH	TYPE		IOMEM_SP_IND
RW-0x0		R-0x1b	R-0x00		R-0x0

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

Table 50. Base Address *n* Mirror Register (PCIBAR*n*MIR) Field Descriptions

Bit	Field	Value	Description
31-4	ADDR		Address bits. These bits can be written by the host to allow initialization of the base address at startup. The writeability of individual bits is determined by the corresponding bit in the Base Address *n* Mask Register in the Back End Configuration registers.
3	PREFETCH		Prefetchable bit. This bit specifies whether or not the memory space controlled by this base address register is prefetchable. This bit reflects the value that is input from the PREFETCH_EN bit of the Base Address *n* Mask Register in the Back End Configuration Registers.
2-1	TYPE		Type bits. These bits indicate the size of the base address register/decoder. This version of the PCI only supports 32-bit addressing, so these bits are hardwired to 00.
0	IOMEM_SP_IND		IO/Memory Space Indicator bit. This bit indicates whether the base address maps into the host's memory or I/O space. This version of the PCI only supports memory mapped base address registers, so this bit is hardwired to a 0.

13.2.23 Master Configuration/IO Transaction Proxy Registers

The Master Configuration/IO Access registers cause the PCI Master to generate Configuration or IO transactions. By using an indirect access method (or proxy) the entire Configuration and IO Space can be addressed, which otherwise would be impossible. For write transactions, the software should write to the Data Register, then the Address Register, and then the Command Register. For read transactions, the software should write to the Address Register and then the Command Register. The transaction will start when the Command Register is written.

13.2.23.1 Master Configuration/IO Access Data Register (PCIMCFGDAT)

When requesting a Configuration or IO Access, software either writes the data into this register for a write, or reads the data from this register for a read.

Figure 51. Master Configuration/IO Access Data Register (PCIMCFGDAT)

31	16
DATA	
RW-0x0	

15	0
DATA	
RW-0x0	

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

Table 51. Master Configuration/IO Access Data Register (PCIMCFGDAT) Field Descriptions

Bit	Field	Value	Description
31-0	DATA		Data bits. Software writes the data into this register bit for a configuration/IO write, or reads the data from this register for a configuration/IO read.

13.2.23.2 Master Configuration/IO Access Address Register (PCIMCFGADR)

When requesting a Configuration or IO Access, software writes the desired address for the transaction into this register.

Figure 52. Master Configuration/IO Access Address Register (PCIMCFGADR)

31	16
ADDR	
RW-0x0	

15	0
ADDR	
RW-0x0	

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

Table 52. Master Configuration/IO Access Address Register (PCIMCFGADR) Field Descriptions

Bit	Field	Value	Description
31-0	ADDR		Address bits. The address bits provide the address for Configuration / IO transactions. Bits 1:0 are ignored for a Configuration Transaction. Software must ensure that Bits 1:0 of the Address correspond to the BYTE_EN bits that are written in the Configuration/IO Access Command register for IO transactions, thus ensuring that the access is legal on the PCI bus.

13.2.23.3 Master Configuration/IO Access Command Register (PCIMCFGCMD)

The back end application will program the Configuration Access Command Register to start a configuration read or write. The READY bit should be checked to determine if the previous transaction is complete.

Figure 53. Master Configuration/IO Access Command Register (PCIMCFGCMD)

31	30	16
READY	Reserved	
R-0x1	R-0x0	

15	8
Reserved	
R-0x0	

7	4	3	2	1	0
BYTE_EN		Reserved	TYPE	Reserved	RD_WR
RW-0x0		R-0x0	RW-0x0	R-0x0	RW-0x0

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

Table 53. Master Configuration/IO Access Command Register (PCIMCFGCMD) Field Descriptions

Bit	Field	Value	Description
31	READY		Ready bit. The READY bit should be checked to determine if the previous transaction is complete.
		0	Register is not ready to accept a new command
		1	Register is ready to accept a new command
30-8	Reserved		Reserved
7-4	BYTE_EN		Byte enable bits. This bit determines which bytes within the addressed DWORD are being accessed. Byte enables indicate the size of the transfer and must be consistent with the Bits 1:0 of the Address that is specified using the PCI Master Configuration/IO Access Address Register.
3	Reserved		Reserved
2	TYPE		Type bit. Set Configuration or IO Transaction.
		0	Configuration Transaction
		1	IO Transaction
1	Reserved		Reserved
0	RD_WR		Read/write bit. Set read or write operation.
		0	Write
		1	Read

13.2.24 Master Configuration Register (PCIMSTCFG)

Master Configuration Register

Figure 54. Master Configuration Register (PCIMSTCFG)

31	16
Reserved	
R-0x0	

15	11	10	9	8
Reserved		CFG_FLUSH_IF_NOT_ENABLED	IO_FLUSH_IF_NOT_ENABLED	MEM_FLUSH_IF_NOT_ENABLED
R-0x0		RW-0x0	RW-0x0	RW-0x0

7	3	2	1	0
Reserved		SW_MEM_RD_MULT_EN	SW_MEM_RD_LINE_EN	SW_MEM_RD_WRINV_EN
R-0		RW-0x1	RW-0x1	RW-0x1

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

Table 54. Master Configuration Register (PCIMSTCFG) Field Descriptions

Bit	Field	Value	Description
31-11	Reserved		Reserved
10	CFG_FLUSH_IF_NOT_ENABLED		Configuration flush bit. Controls whether or not the Master will flush Configuration transactions from the proxy registers if the Master is not enabled.
9	IO_FLUSH_IF_NOT_ENABLED		IO flush bit. Controls whether or not the Master will flush I/O transactions from the proxy registers if the Master is not enabled.
8	MEM_FLUSH_IF_NOT_ENABLED		Memory flush bit. Controls whether or not the Master will flush transactions on the PCIM interface if the Master is not enabled.
7-3	Reserved		Reserved
2	SW_MEM_RD_MULT_EN		Memory read multiple enable bit. Controls whether or not the Master command generation logic is permitted to use the Memory Read Multiple command.
		0	Master will not generate Memory Read Multiple transactions
		1	Master is enabled to generate Memory Read Multiple transactions for appropriate length bursts
1	SW_MEM_RD_LINE_EN		Memory read line enable bit. Controls whether or not the Master command generation logic is permitted to use the Memory Read Line command.
		0	Master will not generate Memory Read Line transactions
		1	Master is enabled to generate Memory Read Line transactions for appropriate length bursts
0	SW_MEM_WRINV_EN		Memory write invalid enable bit. Controls whether or not the Master command generation logic is permitted to use the Memory Write and Invalidate command.
		0	Master will not generate Memory Write and Invalidate transactions
		1	Master is enabled to generate Memory Write and Invalidate transactions for appropriate length bursts.

13.3 DSP-To-PCI Address Translation Registers

The DSP-to-PCI Address Translation registers are used for address translation from the DSP to PCI domain. Using these 32 registers, the master windows can be configured. Each of these 32 registers correspond to 1/32 of the 256 MB addressable PCI Space.

13.3.1 PCI Address Substitution *n* Register (PCIADDSUB*n*)

PCI Address Substitution [0-31] Registers

Figure 55. PCI Address Substitution *n* Register (PCIADDSUB*n*)

31	23	22	16
ADD_SUBS		Reserved	
RW-0x0		R-0x0	

15	0
Reserved	
R-0x0	

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

Table 55. PCI Address Substitution *n* Register (PCIADDSUB*n*) Field Descriptions

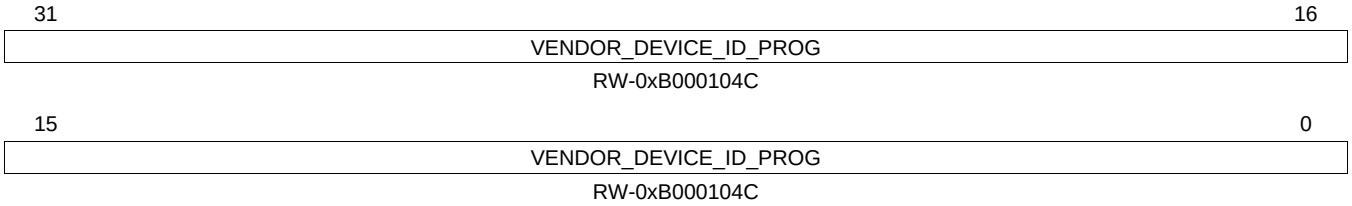
Bit	Field	Value	Description
31-23	ADD_SUBS		Address substitution bits. Substitutes the 9 MSBs of DSP address.
22-0	Reserved		Reserved

13.4 PCI Hook Configuration Registers

All the PCI back end configuration registers (Section 13.2) are hooked up to a set of MMRs called the Configuration Hook Registers. See Table 9 for the list of hook registers supported. The values in these hook registers are latched to the actual PCI registers on PCI reset. The default values in these hook registers can be overwritten by software. These registers are implemented mainly to support PCI I2C EEPROM Autoinitialization, as discussed in Section 12.4.

13.4.1 PCI Vendor ID and Device ID Program Register (PCIVENDEVPRG)

Figure 56. PCI Vendor ID and Device ID Program Register (PCIVENDEVPRG)



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

Table 56. PCI Vendor ID and Device ID Program Register (PCIVENDEVPRG) Field Descriptions

Bit	Field	Value	Description
31-0	VENDOR_DEVICE_ID_PROG		Vendor device ID program bits. This bit provides the default values for the VEN_ID and DEV_ID.

13.4.2 PCI Command and Status Program Register (PCICMDSTATPRG)

Provides default values for the command and status mirror register.

Figure 57. PCI Command and Status Program Register (PCICMDSTATPRG)

31	8
Reserved	
R-0x0	

7	2	1	0
Reserved		66MHZ_CAP	CAP_LIST
R-0x0		RW-Undefined	RW-0x0

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

Table 57. PCI Command and Status Program Register (PCICMDSTATPRG) Field Descriptions

Bit	Field	Value	Description
31-2	Reserved		Reserved
1	66MHZ_CAP		66MHz capabilities bit. Default value for 66MHZ_CAP bit of command status register (PCICSRMIR).
0	CAP_LIST		Capabilities list implemented bit. Default value for CAP_LIST_IMPL of command status register (PCICSRMIR).

13.4.3 PCI Class Code and Revision ID Program Register (PCICLREVPRG)

Provides default values to the class code/revision ID mirror register

Figure 58. PCI Class Code and Revision ID Program Register (PCICLREVPRG)

31	16
CLASS_CODE_REV_ID_PROG	
RW-0x1	

15	0
CLASS_CODE_REV_ID_PROG	
RW-0x1	

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

Table 58. PCI Class Code and Revision ID Program Register (PCICLREVPRG) Field Descriptions

Bit	Field	Value	Description
31-0	CLASS_CODE_REV_ID_PROG		Class code revision ID program bits. This bit provides the default value for CL_CODE and REV_ID.

13.4.4 PCI Subsystem Vendor ID and Subsystem ID Program Register (PCISUBIDPRG)

Provides default values to subsystem vendor ID and subsystem ID.

Figure 59. PCI Subsystem Vendor ID and Subsystem ID Program Register (PCISUBIDPRG)

31	16
SUBSYS_VENDOR_ID_SUBSYS_ID_PROG	
RW-0x0	

15	0
SUBSYS_VENDOR_ID_SUBSYS_ID_PROG	
RW-0x0	

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

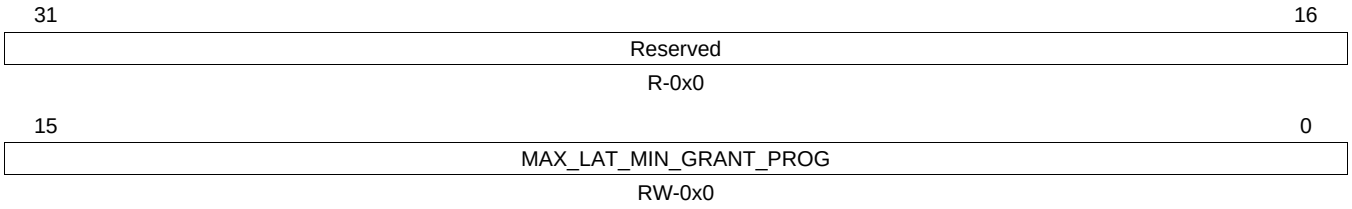
Table 59. PCI Subsystem Vendor ID and Subsystem ID Program Register (PCISUBIDPRG) Field Descriptions

Bit	Field	Value	Description
31-0	SUBSYS_VENDOR_ID_SUBSYS_ID_PROG		Subsystem vendor ID and subsystem ID program bits. This bit provides the default value for SUBSYS_VEN_ID and SUBSYS_ID.

13.4.5 Max Latency and Min Grant Program Register (PCIMAXLGPRG)

Provides default values for max latency and min grant

Figure 60. Max Latency and Min Grant Program Register (PCIMAXLGPRG)



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

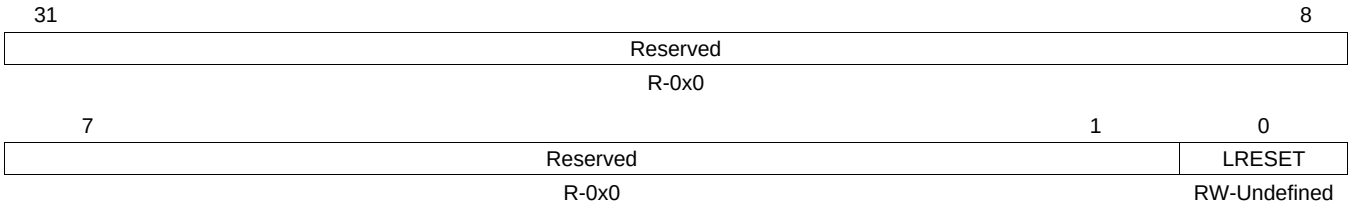
Table 60. Max Latency and Min Grant Program Register (PCIMAXLGPRG) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved		Reserved
15-0	MAX_LAT_MIN_GRANT_PROG		Maximum latency minimum grant program bits. This bit provides the default value for MAX_LAT and MIN_GRNT.

13.4.6 LRESET Register (PCILRSTREG)

Supplies LRESET to device.

Figure 61. LRESET Register (PCILRSTREG)



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

Table 61. LRESET Register (PCILRSTREG) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved		Reserved
0	LRESET		Local reset bit. Provides local reset to the C64x+ Megamodule. The CPU is in reset whenever the C64x+ Megamodule is in reset.
		0	The C64x+ Megamodule is not in reset (local reset deasserted).
		1	The C64x+ Megamodule is in reset (local reset asserted).

13.4.7 Configuration Done Register (PCICFGDONE)

PCI can be autoinitialized by the DSP ROM code after reset. This is indicated by PCI_EEAI being 1 at boot. When autoinitialization is selected, accesses to the PCI are held off until the CONFIG_DONE bit of the PCI Configuration Done Register (PCICFGDONE) is 1. After initialization is done, software will set this bit.

Figure 62. Configuration Done Register (PCICFGDONE)

31	8
Reserved	
R-0x0	

7	1	0
Reserved		CONFIG_DONE
R-0x0		RW-(1)

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

(1) 0 when PCI_EEAI = 1, else 1.

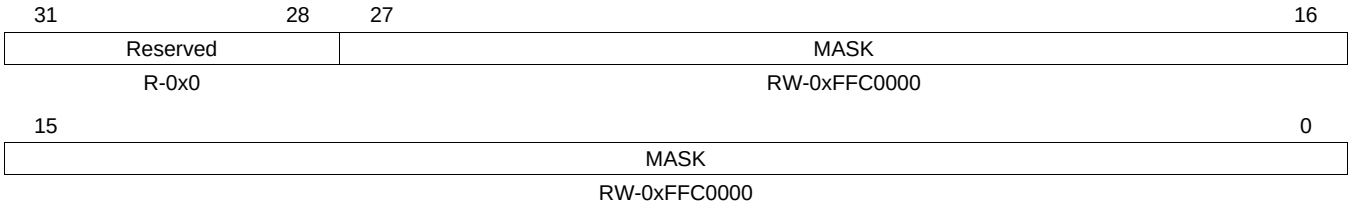
Table 62. Configuration Done Register (PCICFGDONE) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved		Reserved
0	CONFIG_DONE		Configuration done bit. Holds off accesses to the PCI until configuration by the DSP ROM code is complete.
		0	Configuration in progress, accesses to the PCI are not permitted.
		1	Configuration complete, accesses to the PCI are allowed.

13.4.8 Base Address Mask Register 0 Program Register (PCIBAR0MPRG)

Default value for PCI base address mask register 0.

Figure 63. Base Address Mask Register 0 Program Register (PCIBAR0MPRG)



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

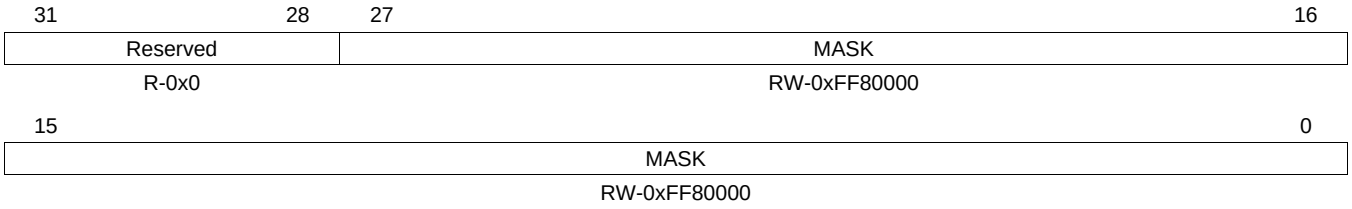
Table 63. Base Address Mask Register 0 Program Register (PCIBAR0MPRG) Field Descriptions

Bit	Field	Value	Description
31-28	Reserved		Reserved
27-0	MASK		Mask bits. This bit provides the default value for Base address mask register 0.

13.4.9 Base Address Mask Register 1 Program Register (PCIBAR1MPRG)

Default value for PCI base address mask register 1.

Figure 64. Base Address Mask Register 1 Program Register (PCIBAR1MPRG)



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

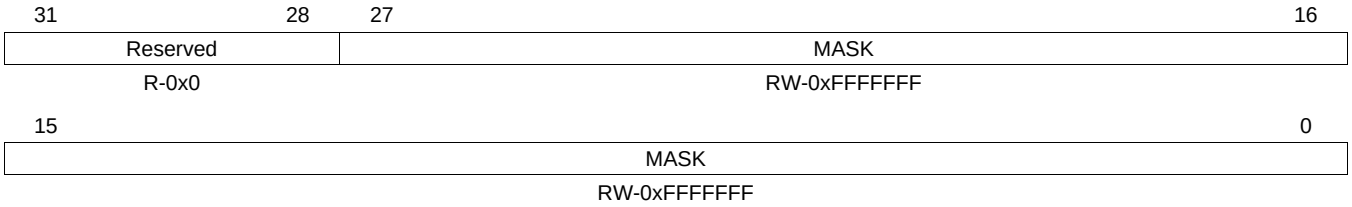
Table 64. Base Address Mask Register 1 Program Register (PCIBAR1MPRG) Field Descriptions

Bit	Field	Value	Description
31-28	Reserved		Reserved
27-0	MASK		Mask bits. This bit provides the default value for Base address mask register 1.

13.4.10 Base Address Mask Register 2 Program Register (PCIBAR2MPRG)

Default value for PCI base address mask register 2.

Figure 65. Base Address Mask Register 2 Program Register (PCIBAR2MPRG)



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

Table 65. Base Address Mask Register 2 Program Register (PCIBAR2MPRG) Field Descriptions

Bit	Field	Value	Description
31-28	Reserved		Reserved
27-0	MASK		Mask bits. This bit provides the default value for Base address mask register 2.

13.4.11 Base Address Mask Register 3 Program Register (PCIBAR3MPRG)

Default value for PCI base address mask register 3.

Figure 66. Base Address Mask Register 3 Program Register (PCIBAR3MPRG)

31	28	27	16
Reserved		MASK	
R-0x0		RW-0x0	

15	0
MASK	
RW-0x0	

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

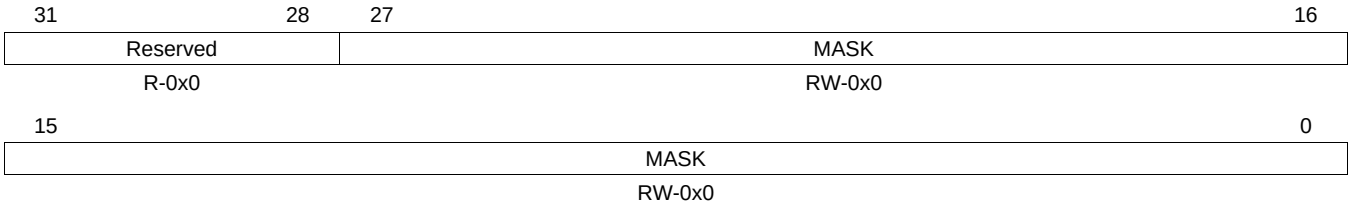
Table 66. Base Address Mask Register 3 Program Register (PCIBAR3MPRG) Field Descriptions

Bit	Field	Value	Description
31-28	Reserved		Reserved
27-0	MASK		Mask bits. This bit provides the default value for Base address mask register 3.

13.4.12 Base Address Mask Register 4 Program Register (PCIBAR4MPRG)

Default value for PCI base address mask register 4.

Figure 67. Base Address Mask Register 4 Program Register (PCIBAR4MPRG)



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

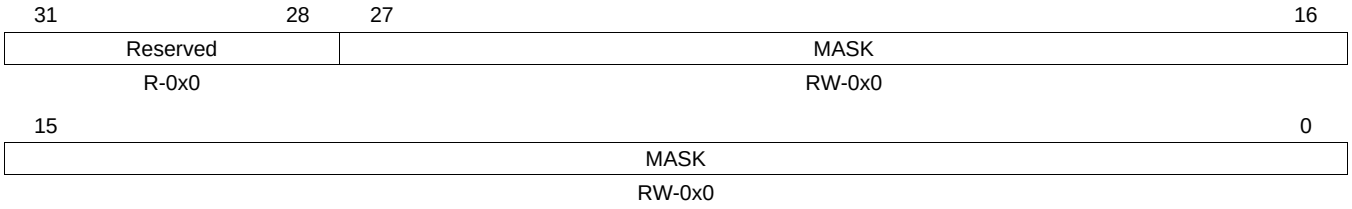
Table 67. Base Address Mask Register 4 Program Register (PCIBAR4MPRG) Field Descriptions

Bit	Field	Value	Description
31-28	Reserved		Reserved
27-0	MASK		Mask bits. This bit provides the default value for Base address mask register 4.

13.4.13 Base Address Mask Register 5 Program Register (PCIBAR5MPRG)

Default value for PCI base address mask register 5.

Figure 68. Base Address Mask Register 5 Program Register (PCIBAR5MPRG)



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

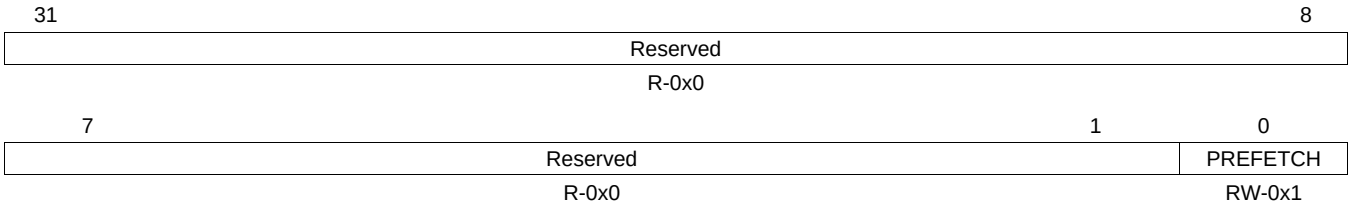
Table 68. Base Address Mask Register 5 Program Register (PCIBAR5MPRG) Field Descriptions

Bit	Field	Value	Description
31-28	Reserved		Reserved
27-0	MASK		Mask bits. This bit provides the default value for Base address mask register 5.

13.4.14 Base Address Register 0 Program Register (PCIBAR0PRG)

Default value for PCI base address register 0 .

Figure 69. Base Address Register 0 Program Register (PCIBAR0PRG)



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

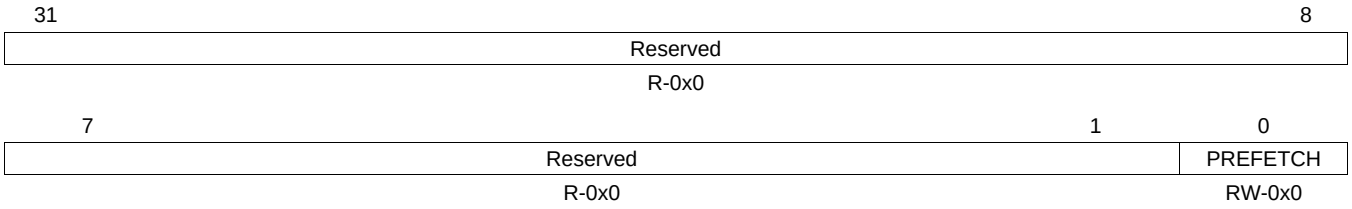
Table 69. Base Address Register 0 Program Register (PCIBAR0PRG) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved		Reserved
0	PREFETCH		Prefetch bit. Provides default value for prefetchable bit of Base address register 0.

13.4.15 Base Address Register 1 Program Register (PCIBAR1PRG)

Default value for PCI base address register 1.

Figure 70. Base Address Register 1 Program Register (PCIBAR1PRG)



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

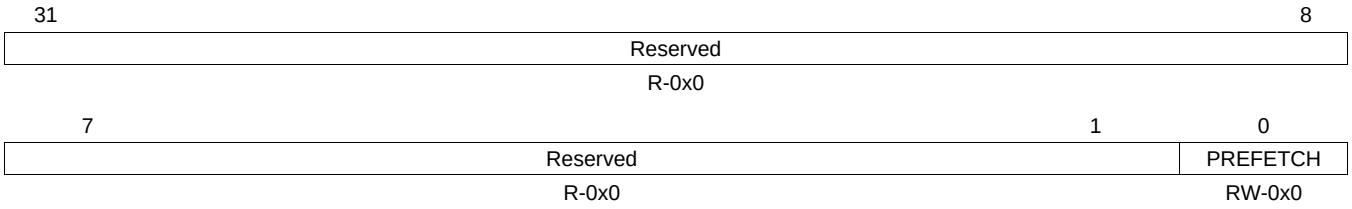
Table 70. Base Address Register 1 Program Register (PCIBAR1PRG) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved		Reserved
0	PREFETCH		Prefetch bit. Provides default value for prefetchable bit of Base address register 1.

13.4.16 Base Address Register 2 Program Register (PCIBAR2PRG)

Default value for PCI base address register 2.

Figure 71. Base Address Register 2 Program Register (PCIBAR2PRG)



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

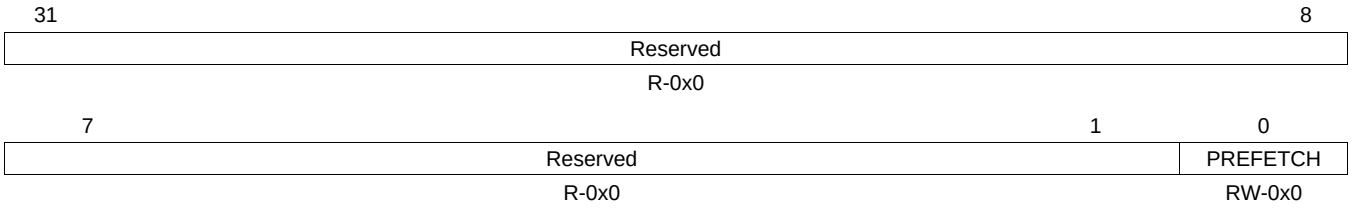
Table 71. Base Address Register 2 Program Register (PCIBAR2PRG) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved		Reserved
0	PREFETCH		Prefetch bit. Provides default value for prefetchable bit of Base address register 2.

13.4.17 Base Address Register 3 Program Register (PCIBAR3PRG)

Default value for PCI base address register 3.

Figure 72. Base Address Register 3 Program Register (PCIBAR3PRG)



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

Table 72. Base Address Register 3 Program Register (PCIBAR3PRG) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved		Reserved
0	PREFETCH		Prefetch bit. Provides default value for prefetchable bit of Base address register 3.

13.4.18 Base Address Register 4 Program Register (PCIBAR4PRG)

Default value for PCI base address register 4.

Figure 73. Base Address Register 4 Program Register (PCIBAR4PRG)

31	8
Reserved	
R-0x0	

7	1	0
Reserved		PREFETCH
R-0x0		RW-0x0

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

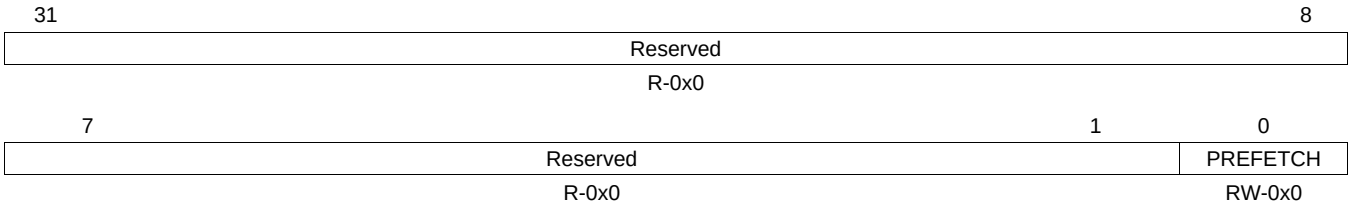
Table 73. Base Address Register 4 Program Register (PCIBAR4PRG) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved		Reserved
0	PREFETCH		Prefetch bit. Provides default value for prefetchable bit of Base address register 4.

13.4.19 Base Address Register 5 Program Register (PCIBAR5PRG)

Default value for PCI base address register 5.

Figure 74. Base Address Register 5 Program Register (PCIBAR5PRG)



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

Table 74. Base Address Register 5 Program Register (PCIBAR5PRG) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved		Reserved
0	PREFETCH		Prefetch bit. Provides default value for prefetchable bit of Base address register 5.

13.4.20 Base Address Translation Register 0 Program Register (PCIBAR0TRLPRG)

Default value for Base address translation register 0.

Figure 75. Base Address Translation Register 0 Program Register (PCIBAR0TRLPRG)

31	28	27	16
Reserved		TRANS_ADDR	
R-0x0		RW-0x0080000	

15	0
TRANS_ADDR	
RW-0x0080000	

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

Table 75. Base Address Translation Register 0 Program Register (PCIBAR0TRLPRG) Field Descriptions

Bit	Field	Value	Description
31-28	Reserved		Reserved
27-0	TRANS_ADDR		Translation address bits. Provides default value for Base address translation register 0.

13.4.21 Base Address Translation Register 1 Program Register (PCIBAR1TRLPRG)

Default value for Base address translation register 1.

Figure 76. Base Address Translation Register 1 Program Register (PCIBAR1TRLPRG)

31	28	27	16
Reserved		TRANS_ADDR	
R-0x0		RW-0x0180000	

15	0
TRANS_ADDR	
RW-0x0180000	

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

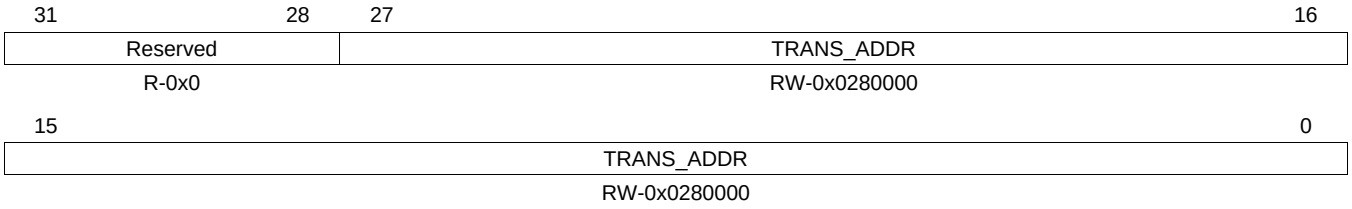
Table 76. Base Address Translation Register 1 Program Register (PCIBAR1TRLPRG) Field Descriptions

Bit	Field	Value	Description
31-28	Reserved		Reserved
27-0	TRANS_ADDR		Translation address bits. Provides default value for Base address translation register 1.

13.4.22 Base Address Translation Register 2 Program Register (PCIBAR2TRLPRG)

Default value for Base address translation register 2.

Figure 77. Base Address Translation Register 2 Program Register (PCIBAR2TRLPRG)



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

Table 77. Base Address Translation Register 2 Program Register (PCIBAR2TRLPRG) Field Descriptions

Bit	Field	Value	Description
31-28	Reserved		Reserved
27-0	TRANS_ADDR		Translation address bits. Provides default value for Base address translation register 2.

13.4.23 Base Address Translation Register 3 Program Register (PCIBAR3TRLPRG)

Default value for Base address translation register 3.

Figure 78. Base Address Translation Register 3 Program Register (PCIBAR3TRLPRG)

31	28	27	16
Reserved		TRANS_ADDR	
R-0x0		RW-0x8000000	

15	0
TRANS_ADDR	
RW-0x8000000	

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

(#IMPLIED) This is not a supported memory address range in TCI648x devices and needs to be reprogrammed before being used.

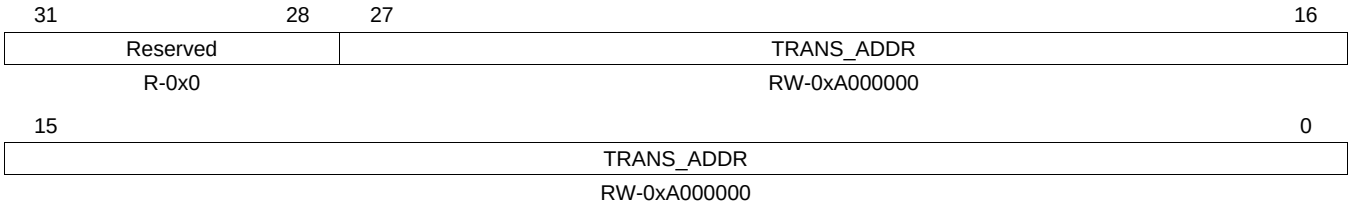
Table 78. Base Address Translation Register 3 Program Register (PCIBAR3TRLPRG) Field Descriptions

Bit	Field	Value	Description
31-28	Reserved		Reserved
27-0	TRANS_ADDR		Translation address bits. Provides default value for Base address translation register 3.

13.4.24 Base Address Translation Register 4 Program Register (PCIBAR4TRLPRG)

Default value for Base address translation register 4.

Figure 79. Base Address Translation Register 4 Program Register (PCIBAR4TRLPRG)



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

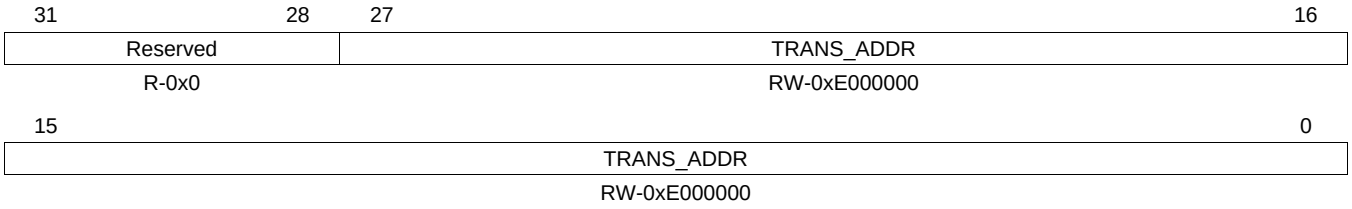
Table 79. Base Address Translation Register 4 Program Register (PCIBAR4TRLPRG) Field Descriptions

Bit	Field	Value	Description
31-28	Reserved		Reserved
27-0	TRANS_ADDR		Translation address bits. Provides default value for Base address translation register 4.

13.4.25 Base Address Translation Register 5 Program Register (PCIBAR5TRLPRG)

Default value for Base address translation register 5.

Figure 80. Base Address Translation Register 5 Program Register (PCIBAR5TRLPRG)



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

Table 80. Base Address Translation Register 5 Program Register (PCIBAR5TRLPRG) Field Descriptions

Bit	Field	Value	Description
31-28	Reserved		Reserved
27-0	TRANS_ADDR		Translation address bits. Provides default value for Base address translation register 5.

13.4.26 Base Enable Program Register (PCIBASENPRG)

Default value for enables for the base address.

Figure 81. Base Enable Program Register (PCIBASENPRG)

31	16
Reserved	
R-0x0	

15	6	5	0
Reserved		BASE_EN	
R-0x0		RW-0x3F	

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

Table 81. Base Enable Program Register (PCIBASENPRG) Field Descriptions

Bit	Field	Value	Description
31-6	Reserved		Reserved
5-0	BASE_EN		Base enable program bits. These bits control bits 21:16 (BASE*n*_EN) of the PCI Slave control register (PCISLVCNTL). These bits provide default values for base address enable for base address0 (bit 0), base address1 (bit 1), base address2 (bit 2), base address3 (bit 3), base address4 (bit 4), and base address5 (bit 5).

Appendix A Revision History

This revision history highlights the technical changes made to the document in this revision.

See	Additions/Modifications/Deletions
Table 2	Modified Footnote (1)
Figure 5	Modified figure
Section 8.2	Added Note
Example 1	Modified code example
Example 2	Modified code example
Section 12.4.1	Modified third paragraph
Table 8	Modified table
Section 12.4.3	Modified paragraphs

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**	
Amplifiers	amplifier.ti.com	Audio	www.ti.com/audio
Data Converters	dataconverter.ti.com	Automotive	www.ti.com/automotive
DLP® Products	www.dlp.com	Communications and Telecom	www.ti.com/communications
DSP	dsp.ti.com	Computers and Peripherals	www.ti.com/computers
Clocks and Timers	www.ti.com/clocks	Consumer Electronics	www.ti.com/consumer-apps
Interface	interface.ti.com	Energy	www.ti.com/energy
Logic	logic.ti.com	Industrial	www.ti.com/industrial
Power Mgmt	power.ti.com	Medical	www.ti.com/medical
Microcontrollers	microcontroller.ti.com	Security	www.ti.com/security
RFID	www.ti-rfid.com	Space, Avionics & Defense	www.ti.com/space-avionics-defense
RF/IF and ZigBee® Solutions	www.ti.com/lprf	Video and Imaging	www.ti.com/video
		Wireless	www.ti.com/wireless-apps

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