PLX Technology, Inc.
390 Potrero Avenue, Sunnyvale, CA USA 94086
Tel: 408 774-9060
Fax: 408 774-2169

CONFIDENTIAL & PROPRIETARY
PRELIMINARY INFORMATION

PCI 9610 Data Book

Document #: PCI 9610 - _____

Issued To: _____
Name: _____
Company: _____
Date: _____

Authorized: _____

PLX TECHNOLOGY®
THE I/O INTERCONNECT SOLUTION

PCI 9610 Data Book

Preliminary Information

PCI 9610 Data Book

Revision 0.90

August 2000

Website: http://www.plxtech.com
Email: 9610@plxtech.com
Phone: 408 774-9060
800 759-3735
FAX: 408 774-2169

Preliminary Information

Order Number: 9610-SIL-DB-P1-0.90

Printed in the USA, August 2000

Contents

FIGURES

TABLES

REGISTERS

TIMING DIAGRAMS

PREFACE

The information contained in this document is subject to change without notice. Although an effort has been made to keep the information accurate, there may be misleading or even incorrect statements made herein.

SUPPLEMENTAL DOCUMENTATION

The following is a list of additional documentation to provide the reader with further information:

- *PCI Local Bus Specification, Revision 2.2*, December 18, 1998
  PCI Special Interest Group (PCI SIG)
  5440 SW Westgate Drive #217, Portland, OR 97221 USA
  Tel: 800 433-5177 (domestic only) or 503 693-6232, Fax: 503 693-8344, http://www.pcisig.com
- *PCI Hot-Plug Specification, Revision 1.0*
  PCI Special Interest Group (PCI SIG)
  5440 SW Westgate Drive #217, Portland, OR 97221 USA
  Tel: 800 433-5177 (domestic only) or 503 693-6232, Fax: 503 693-8344, http://www.pcisig.com
- *PCI Bus Power Management Interface Specification, Revision 1.1*, December 18, 1998
  PCI Special Interest Group (PCI SIG)
  5440 SW Westgate Drive #217, Portland, OR 97221 USA
  Tel: 800 433-5177 (domestic only) or 503 693-6232, Fax: 503 693-8344, http://www.pcisig.com
- *PICMG 2.0, CompactPCI Specification, Revision 3.0*, October 1, 1999
  PCI Industrial Computer Manufacturers Group (PICMG)
  c/o Virtual Inc., 401 Edgewater Place, Suite 500, Wakefield, MA 01880, USA
  Tel: 781 224-1100, Fax: 781 224-1239, http://www.picmg.org
- *PICMG 2.1, CompactPCI Hot Swap Specification, Revision 1.0*, August 3, 1998
  PCI Industrial Computer Manufacturers Group (PICMG)
  c/o Virtual Inc., 401 Edgewater Place, Suite 500, Wakefield, MA 01880, USA
  Tel: 781 224-1100, Fax: 781 224-1239, http://www.picmg.org
- *Intelligent I/O (I_2O) Architecture Specification, Revision 1.5*, 1997
  I_2O Special Interest Group (I_2O SIG)
  404 Balboa Street, San Francisco, CA 94118, USA
  Tel: 415 750-8352, Fax: 415 751-4829, http://www.i2osig.org
- IEEE Standard 1149.1-1990, *IEEE Standard Test Access Port and Boundary-Scan Architecture*, 1990
  The Institute of Electrical and Electronics Engineers, Inc.
  345 East 47th Street, New York, NY 10017-2394, USA
  Tel: 732 562-3800, Fax: 732 562-1571, http://www.ieee.org
- *MPC8260 PowerQUICC II User's Manual, MPC8260UM/D, Revision 1.0*, May, 1999
  Motorola Literature Distribution
  PO Box 5405, Denver, CO 80217
  Tel: 800 441-2447 (domestic only) or 303 675-2140, http://sps.motorola.com/mfax

TERMS AND DEFINITIONS

For other unfamiliar terms, refer to the index for text references.

Note: *Past PLX chips have referred to their backend interface as a "Local Bus." The MPC8260 has its own Local Bus, which is* ***not*** *the interface the PCI 9610 uses. The PCI 9610 uses the MPC8260 60x Bus interface. As a result, this document uses the term "Processor Bus," instead of "Local Bus" as a generic term for its backend interface. The term "60x Bus" is also used when the specific type of bus is referenced.*

- **60x Bus**—Protocol, as defined in the *Motorola MPC8260 PowerQUICC II User's Manual*, and used by the Motorola MPC8260, MPC603e, MPC740, MPC750, and others.
- **Direct Master**—External Processor Bus Master initiates Data write/read to/from the PCI Bus.
- **Direct Slave**—External PCI Bus Master initiates Data write/read to/from the local Processor Bus.
- **PLX Extended Burst**—Extension to the 60x Bus, which allows indefinite bursts. Requires three additional pins and an extension to the 60x Bus protocol that are transparent to the standard 60x Bus protocol.
- **Processor Bus**—Bus that includes all operations and functions the 60x Bus protocol and PLX Extended Burst protocol perform.
- **Reserved Bits versus Special Reserved Bits**—Found in register bit descriptions. For PCI 9610 purposes, ***Reserved*** bits are those which were reserved in the PCI 9054, or those that are reserved in the new PCI 9610 registers, but not used in the PCI 9054. Reserved bits cannot be written. ***Special Reserved*** bits are bits that were previously used in the PCI 9054, but are not needed in the PCI 9610. These bits can be written, but they do not carry out a function. However, they are necessary to maintain register compatibility.

Data Assignment Conventions

Data Width	PCI 9610 Convention
1 byte (8 bits)	Byte
2 bytes (16 bits)	Word
4 bytes (32 bits)	Lword
8 bytes (64 bits)	Qword

REVISION HISTORY

Date	Revision	Comments
01/2000	0.11	Initial release Red Book.
08/18/2000	0.90	Initial release Blue Book.

PCI 9610

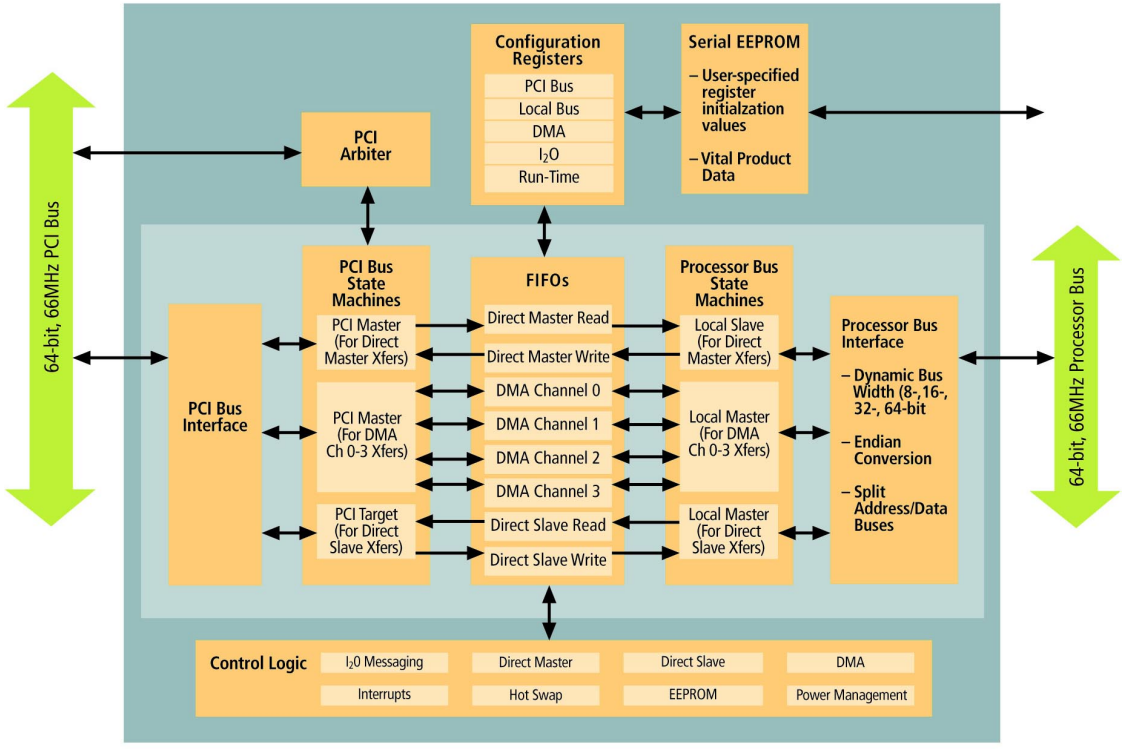
64-Bit, 66 MHz PCI Bus Mastering I/O Accelerator
for PowerQUICC II and MPC603e, MPC740, and MPC750 Processor Bus Designs

August 2000
Revision 0.90

FEATURE SUMMARY

- Bus Mastering interface between a 64-bit, 66 MHz PCI Bus and 64-bit, 66 MHz PowerPC® Processor Bus
  - PCI r2.2-compliant
    - Supports Vital Product Data (VPD)
    - Supports PCI Power Management r1.1, including D_{3cold} PME generation for PC 2001 modem and network communications adapter compliance
    - PICMG 2.1 CompactPCI® r1.0 Hot Swap *Ready*
    - PCI Hot Plug r1.0 compatible
  - Direct connection to 64-bit, 66 MHz PowerPC processors
    - Motorola MPC8260 PowerQUICC II™
    - Motorola MPC603e, MPC740, and MPC750, and other MPC603e Bus protocol-compliant devices
  - Enhanced Procesor Bus mode supports unlimited bursts
  - Asynchronous clock inputs for PCI and Processor Buses
- 416-pin (ball), 27 x 27 mm, 1.00 mm fine-pitch PBGA
  - Low-power CMOS 2.5V core, 3.3V I/O
  - 3.3 and 5.0V-tolerant PCI and Processor Bus operation
  - Industrial Temperature Range operation
  - IEEE 1149.1 JTAG boundary scan
- Three data transfer modes—Direct Master, Direct Slave, and DMA
  - **Direct Master**—Transfer data between a Master on the Processor Bus and a PCI Bus device
    - Two Processor Bus address spaces to the PCI Bus—one to PCI memory and one to PCI I/O
    - Generates all PCI Memory and I/O transaction types, including MWI and Type 0 and Type 1 configuration
    - Read Ahead, Programmable Read Prefetch Counter
    - Delayed Read, MPC8620 and IDMA support
  - **Direct Slave**—Transfer data between a Master on the PCI Bus and a 64-, 32-, 16-, or 8-bit Processor Bus device
    - Two general-purpose address spaces to the Processor Bus and one Expansion ROM address space
    - Delayed Read, Delayed Write, Read Ahead, Posted Write, Programmable Read Prefetch counter
    - Programmable READY# timeout and recovery
  - **DMA**—PCI 9610 services data transfer descriptors, mastering on both buses during transfer
    - Four independent channels
    - Block mode—Single descriptor execution
    - Scatter/Gather mode
      - Descriptors in PCI or Processor Bus memory
      - Linear descriptor list execution
      - Dynamic DMA Descriptor Ring Management with Valid bit semaphore control
      - Burst descriptor loading
    - Hardware EOT/Demand controls to stop/pause DMA in any mode
    - Programmable Processor Bus burst lengths, including infinite burst

- Eight independent, programmable FIFOs—Direct Master Read and Write, Direct Slave Read and Write, DMA Channels 0, 1, 2, and 3
- Advanced features common to Direct Master, Direct Slave, and DMA
  - Zero wait state burst operation
  - 528 MB/s bursts on PCI Bus
  - 528 MB/s bursts on Processor Bus
  - Deep FIFOs
  - Unaligned transfers on both buses
  - On-the-fly Processor Bus Endian conversion
  - Programmable Processor Bus wait states
  - Parity checking on both buses
- I_2O™ r1.5-Ready Messaging Unit
- Eight 32-bit Mailbox and two 32-bit Doorbell registers enable general-purpose messaging
- PCI arbiter supports seven external masters in addition to the PCI 9610
- Reset and interrupt signal directions configurable for host and peripheral applications
- Programmable Interrupt Generator
- Serial EEPROM interface
  - Store user-specified power-on/reset configuration register values
  - Store Vital Product Data (VPD)
- Register compatible with PCI 9060, PCI 9080, PCI 9054, and PCI 9656



PCI 9610 Block Diagram

1 INTRODUCTION

1.1 COMPANY AND PRODUCT BACKGROUND

PLX Technology, Inc., the leading supplier of high-performance PCI-to-Processor Bus chips and software, supports OEM customers in a wide variety of applications including embedded networking products, such as routers and switches, PC workstations and servers, adapter boards, and industrial implementations, such as CompactPCI and PMC.

An active participant in industry standard committees, including the PCI SIG®, PICMG®, CompactPCI, I_2O SIG®, and the RapidIO trade association, PLX maintains active development and cross-marketing partnerships with industry leaders such as Intel, IBM, Hewlett-Packard, Motorola, WindRiver, and others.

PLX provides customers with the complete PCI solution, which results in faster time to market and lower development costs. This complete solution consists of high-performance I/O chips, PCI Reference Design Kits (RDK), the PCI Hardware Development Kit (HDK) CD-ROM collection, PCI Software Development Kits (SDK), and third-party development tools through the PLX Partner Program. Our reference boards, “C” API libraries, software debug tools, RTOS and Windows device drivers enable customers to quickly bring new designs to production without worrying about the complexities of implementing PCI hardware, software, and I_2O. New tools, application notes and information updates are frequently added to the PLX website (http://www.plxtech.com).

Serving the computer industry since 1986, PLX is the leading source of high-performance, I/O silicon, software, and development tools.

1.2 DATA PIPE ARCHITECTURE TECHNOLOGY

PLX I/O accelerators feature PLX proprietary Data Pipe Architecture™ technology. This technology consists of powerful, flexible engines for high-speed data transfers, as well as intelligent messaging units for managing distributed I/O functions.

1.2.1 High-Speed Data Transfers

Data Pipe Architecture technology provides independent methods for moving data—Direct Transfers and DMA.

Regardless of the method chosen, Data Pipe Architecture technology data transfers support the following:

- PCI ↔ Processor Bus burst transfers at the maximum bus rates
- Unaligned transfers on both buses
- On-the-fly Processor Bus Endian conversion
- Programmable Processor Bus wait states
- Parity checking on both buses

1.2.1.1 Direct Transfers

Data Pipe Architecture technology Direct Transfers are used by a master on either the PCI or Processor Bus to move data through the I/O accelerator to a device on the other bus. The Master takes responsibility for moving the data either into the I/O accelerator on a write or out of the I/O accelerator on a read. The I/O accelerator is responsible for moving the data out to the target device on a write, or in from the target device on a read.

1.2.1.1.1 Direct Master

When a master on the Processor Bus uses Direct Transfer, this is a *Direct Master* transfer. The I/O accelerator is a master on the PCI Bus. Data Pipe Architecture technology provides independent FIFOs for Direct Master Read and Write transfers. It also supports multiple independent Direct Master Processor Bus Address spaces for mapping to PCI addresses, as shown in Figure 1-1.

Direct Master transfers support generation of all PCI Memory and I/O transaction types, including Type 0 and Type 1 cycles for system configuration.

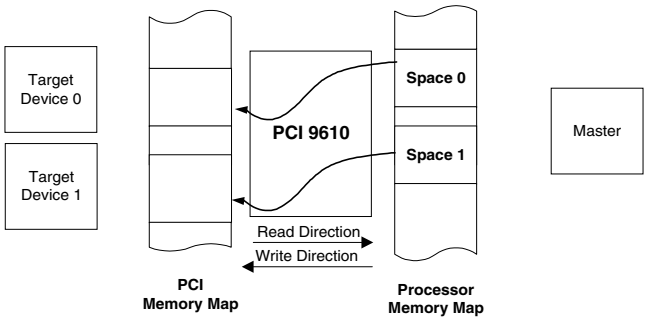


Figure 1-1. Direct Master Address Mapping

1.2.1.1.2 Direct Slave

When a master on the PCI Bus uses Direct Transfer, this is a *Direct Slave* transfer. The I/O accelerator is a slave (technically, a target) on the PCI Bus. Data Pipe Architecture technology provides independent FIFOs for Direct Slave Read and Write transfers. It also supports multiple independent Direct Slave PCI Address spaces for mapping to Processor Bus addresses, as shown in Figure 1-2.

Under software control, Direct Slave transfers support Processor Bus data transfers of various widths (*for example*, on 32-bit Processor Buses, data widths of 8, 16, and 32 bits are supported). Direct Slave Read transfers also support PCI delayed reads.

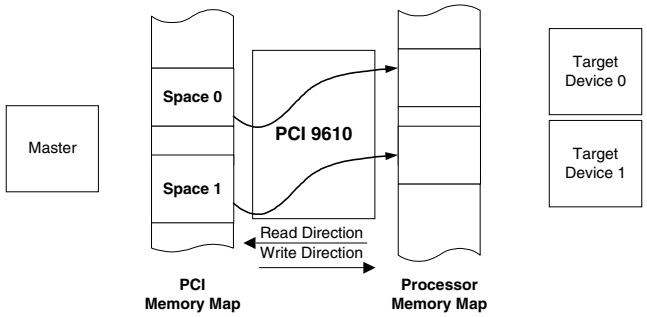


Figure 1-2. Direct Slave Address Mapping

1.2.1.2 DMA

When a Master on either bus uses Data Pipe Architecture technology DMA transfers, instead of the Master moving data, it places a description of the entire transfer in I/O accelerator registers and allows the I/O accelerator to perform the entire Data transfer with its DMA engine. This offers two main benefits:

1. Data movement responsibilities are offloaded from the Master. A transfer descriptor is short and takes little effort on the Master's part to load. Once the descriptor is loaded into the I/O accelerator, the Master is free to spend its time and resources elsewhere.

2. Because the I/O accelerator supports multiple DMA channels, each with its own FIFO, it can service multiple PCI and Processor Bus Masters simultaneously. During DMA transfers, the I/O accelerator masters each bus. Consequently, during DMA, there are no external masters to Retry. During DMA, if the I/O accelerator is Retried on either bus, it can simply change context to another transfer and continue. Furthermore, DMA can run simultaneously with Direct Master and Direct Slave transfers, providing support for several simultaneous data transfers. Direct Master and Direct Slave transfers have higher priority than DMA.

Data Pipe Architecture technology supports two DMA transfer modes—Block and Scatter/Gather.

1.2.1.2.1 Block Mode

Block mode is the simplest DMA mode. The Master simply programs the description of a single transfer in the I/O accelerator and enables the DMA channel. The I/O accelerator signals DMA completion to the Master, either by setting a bit in one of its registers that the Master polls or by asserting an interrupt.

1.2.1.2.2 Scatter/Gather Mode

In most cases, however, one descriptor is not sufficient. The Master typically generates a list of several descriptors in its memory before submitting them to the I/O accelerator. For these cases, *Scatter/Gather mode* is used to enable I/O accelerator list processing with minimal master intervention.

With Scatter/Gather mode, the Master simply tells the I/O accelerator the location of the first descriptor in its list, enables the DMA channel, then waits for the I/O accelerator to service the entire list. This offloads both data and DMA descriptor transfer responsibilities from the Master.

Data Pipe Architecture technology supports Scatter/Gather mode descriptor lists in either PCI or Processor Bus memory. It also supports linear and circular descriptor lists, the latter is called *Ring mode*.

Ring mode uses a Valid bit in each descriptor to enable dynamic list management. In this case, the Master and I/O accelerator continuously "walk" the descriptor list, the master in the lead filling invalid descriptors, setting the Valid bit when done, and the I/O accelerator following behind servicing valid descriptors, resetting the Valid bit when done. The I/O accelerator supports write back to serviced descriptors, allowing status and actual transfer counts to be posted prior to resetting the Valid bit.

1.2.1.2.3 Hardware DMA Controls—EOT and Demand Mode

To optimize DMA transfers in datacom/telecom and other applications, Data Pipe Architecture technology supports hardware controls of the data transfer.

With End of Transfer (EOT), an EOT# signal is asserted to the I/O accelerator to end the transfer. Whenever EOT# is asserted, the I/O accelerator immediately aborts the current DMA transfer and writes back to the current DMA descriptor the actual number of bytes transferred. Data Pipe Architecture technology also supports unlimited bursting. EOT and unlimited bursting are especially useful in applications such as Ethernet adapter cards where the lengths of read packets are not known until the packets are read.

With *Demand mode*, a hardware DREQ#/DACK# signal pair is used to pause and resume the DMA transfer. Data Pipe Architecture technology provides one DREQ#/DACK# signal pair for each DMA channel. Demand mode provides a means for a peripheral device with a FIFO to control DMA transfers. The peripheral device uses Demand mode both to pause the transfer when the FIFO is full on a write or empty on a read and to resume the transfer when the FIFO condition changes to allow data transfer to continue.

1.2.2 Intelligent Messaging Unit

Data Pipe Architecture technology provides two methods for managing system I/O through messaging.

The first method is provided through support for Intelligent I/O (I_2O). As the device independent, industry standard method for I/O control, I_2O is the easiest way to obtain interoperability of all PCI-based components in the system.

The second method is provided through general-purpose mailbox and doorbell registers. When all PCI-based components are under direct control of the system designer (*for example*, an embedded system, such as a set-top box), it is often desirable to implement an application-specific messaging unit through general-purpose mailbox and doorbell registers.

1.3 PCI 9610 I/O ACCELERATOR

The PCI 9610, a 64-bit 66 MHz PCI Bus Master I/O Accelerator, extends the PLX family of advanced general-purpose bus master devices to 64-bit, 66 MHz operation. The PCI 9610 Processor Bus is compatible with Motorola's 64-bit, 66 MHz PowerPC Bus introduced with the MPC603e and now adopted by the MPC8260 PowerQUICC II, the industry's first 64-bit, 66 MHz Communications Processor. (Refer to Table 1-2 for a detailed comparison of the PCI 9610 with other PLX bus mastering I/O accelerators.)

The PCI 9610 register set is backward-compatible with the previous generation PCI 9054 and PCI 9080 I/O Accelerators and offers a robust *PCI Specification r2.2* implementation, enabling data burst transfers up to 528 MB/s. It incorporates the industry-leading PLX Data Pipe Architecture technology, including programmable Direct Master and Direct Slave transfer modes, intelligent DMA engines, and PCI messaging functions.

1.3.1 Applications

The PCI 9610 continues the PLX tradition of expanding its product capabilities to meet the leading edge requirements of I/O intensive embedded-processor applications. The PCI 9610 builds upon the industry-leading PLX PCI 9080 and PCI 9054 products, providing a focused solution for Motorola 64-bit, 66 MHz PowerPC processors.

1.3.1.1 High-Performance Motorola MPC8260 and PowerQUICC II Designs

Motorola's 32-bit, 50 MHz MPC860 PowerQUICC is the industry's leading communication processor. The next generation 64-bit, 66 MHz MPC8260 PowerQUICC II is a key enabler of 64-bit, 66 MHz I/O operation for the communications industry. The

PCI 9610 is ideal for MPC8260-based PowerQUICC II designs. Targeted datacom and telecom applications include high-speed routers/switches, Frame Relay adapters, WAN/LAN controllers, and modem cards.

The PCI 9610 simplifies these designs by providing an enhanced direct-connect interface to the PowerQUICC II processor family. The flexible 3.3V, 5V tolerant I/O buffers, combined with 64-bit processor bus operation up to 66 MHz, are ideally suited for current and future PowerQUICC II processors.

The PCI 9610 supports the PowerQUICC II IDMA channels for data transfers between the integrated PowerQUICC II communication channels and PCI.

In addition, the PCI 9610 makes use of the leading edge Data Pipe Architecture technology, allowing unlimited bursts, as illustrated in Figure 1-3.

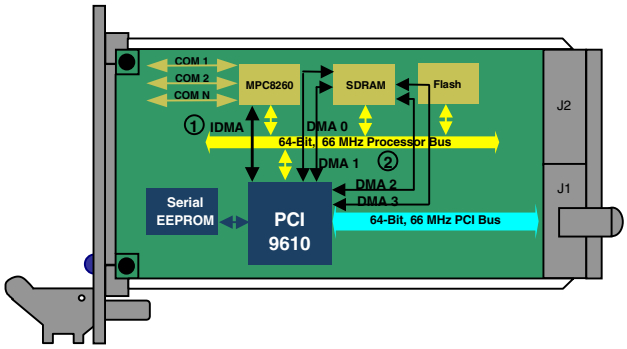


Figure 1-3. PowerQUICC II CompactPCI Adapter

1. For PowerQUICC II IDMA operation, the PCI 9610 transfers data to the PCI Bus under the control of the IDMA handshake protocol using Direct Master transfers (❶).
2. Simultaneously, the four PCI 9610 DMA channels run as masters on both buses to perform data transfers between the Processor and PCI Buses (❷).

This is a prime example of how the PCI 9610 provides greater flexibility to PowerQUICC II designers in implementing multiple simultaneous I/O transfers.

The PCI 9610 is not limited to PowerQUICC II support. Any MPC603e bus-compliant processor is supported. This includes the Motorola MPC9603e, MPC740, and MPC750. Custom devices that adhere to the MPC603e protocol are also supported.

Additionally, the PCI 9610 supports a PLX-proprietary enhancement to the MPC603e bus protocol, enabling extended burst transfers for custom devices. Without this enhancement, MPC603e bus bursts would be limited to four beats (32 bytes) each. However, with this enhancement, devices on the MPC603e bus can be designed to perform extended bursts. This feature can be exploited by application-specific devices (*such as*, Ethernet controllers) to move entire packets of data in a single burst.

1.3.1.2 High-Performance CompactPCI Adapter Cards

Applications for MPC603e-compliant processors such as the MPC8260 come in many form factors. The PCI 9610 includes specific features to support CompactPCI adapters for industrial, telecom, and networking applications. These applications include WAN/LAN controllers, modem cards, Frame Relay adapters, and telephony cards for telecom switches and remote-access systems.

The PCI 9610 has integrated key features to enable live-insertion of Hot Swap CompactPCI adapters. The PCI 9610 PICMG 2.1 CompactPCI Hot Swap *Ready* PCI interface includes all Hot Swap *Capable*, *Friendly*, and *Ready* features.

Hot Swap Capable

- Compliant with *PCI Specification r2.2*
- Tolerant of Vcc from early power, including support for pin bounce, 2.5 and 3.3V appearing in any order, I/O cell stability within 4 ms, and low current drain during insertion
- Tolerant of asynchronous reset
- Tolerant of precharge voltage
- I/O buffers meet modified V/I requirements in the *PICMG 2.1 CompactPCI Hot Swap Specification r1.0*
- Limited I/O pin leakage at precharge voltage

Hot Swap Friendly

- Incorporates the Hot Swap Control/Status register (HS_CSR)
- Incorporates an Extended Capability Pointer (ECP) to the Hot Swap Control/Status register
- Incorporates added resources for software control of the ejector switch, ENUM#, and the status LED which indicates insertion and removal to the user

Hot Swap Ready

- BIAS voltage support with integrated 10K ohm precharge resistors eliminates the need for an external resistor network
- Early power support allows transition between the operating and powered down states without external circuitry

Figure 1-4 illustrates a CompactPCI peripheral card that utilizes an MPC8260 CPU for communication I/O and the PCI 9610 for PCI-based I/O.

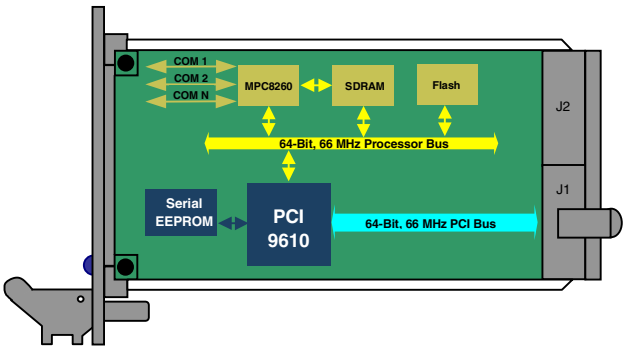


Figure 1-4. PCI 9610 CompactPCI Peripheral Card

The PCI 9610, with its internal PCI arbiter, reset signal direction control, and PCI Type 0 and Type 1 configuration support, is also an ideal choice for CompactPCI system cards.

1.3.1.3 High-Performance PC Adapter Cards

The PCI 9610 is also designed for traditional PCI adapter card applications requiring 64-bit, 66 MHz PCI operation and bandwidth. Specific applications are high-performance communications, networking, disk control, and data encryption adapters.

Today, Power Management and Green PCs are major initiatives in traditional PCI applications. The PCI 9610 supports PCI Power Management, including a generation of PME in the D_{3cold} state. This is becoming a requirement for modem and communications adapters in PC 2001-compliant systems.

Figure 1-5 illustrates the PCI 9610 in a PCI adapter card application with a CPU, using the 60x Bus Processor.

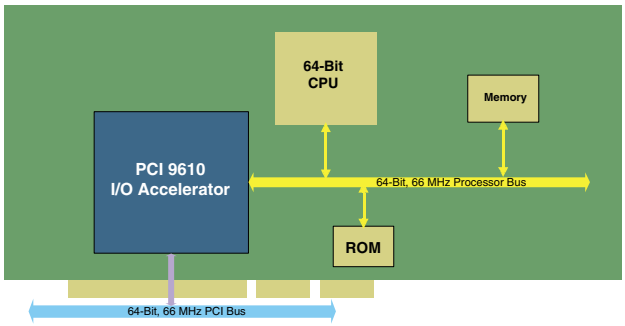


Figure 1-5. PCI 9610 PC Adapter Card with 60x Bus Processor

For applications using I/O types not supported directly by the processor (such as, SCSI for storage applications), the PCI 9610 provides a high-speed interface between the processor and PCI-based I/O chips. Furthermore, its Processor Bus interface supports processors that do not include integrated I/O.

Typically, a PCI-to-PCI bridge chip is used to isolate the add-in card's local PCI Bus and its I/O chips from the system bus. Figure 1-6 illustrates a typical PCI add-in card with local PCI I/O using the PCI 9610 and a PCI-to-PCI bridge interfacing to the system PCI Bus. The PCI 9610 internal PCI arbiter provides arbitration services to devices on the local PCI Bus.

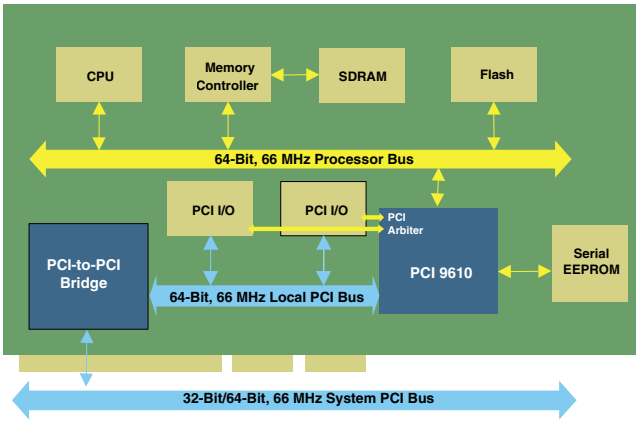


Figure 1-6. PCI 9610 PC Adapter Card with Local PCI I/O and PCI-to-PCI Bridge

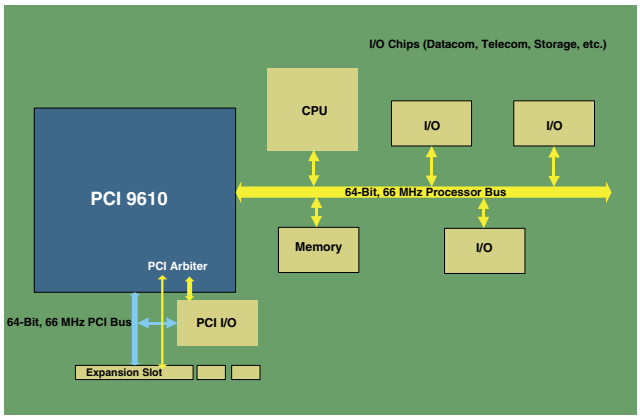


Figure 1-7. PCI 9610 Embedded Host System with Generic Host CPU

1.3.1.4 High-Performance Embedded Host Designs

I/O-intensive embedded host designs are another major application of the PCI 9610. These applications include network switches and routers, printer engines, set-top boxes, CompactPCI system cards, and industrial equipment.

While the support requirements of these embedded host designs share many similarities with peripheral card designs, there are three significant differences:

- A host must configure the PCI Bus. The PCI 9610 supports PCI Type 0 and Type 1 configuration cycles to accomplish this.
- The host must provide a PCI Bus arbiter. The PCI 9610 PCI arbiter supports seven external PCI masters in addition to the PCI 9610, sufficient for a standard 33 MHz CompactPCI backplane with seven peripheral slots and one system slot.
- For hosts, the direction of the reset and interrupt signals reverse. The PCI 9610 includes a strapping option for reversing the direction of the PCI and Processor Bus Reset and Interrupt signals. In one setting, the directions are appropriate for a peripheral. In the other, they are appropriate for a host.

Figure 1-7 illustrates the PCI 9610 in an embedded host system.

1.4 PCI 9610 MAJOR FEATURES

1.4.1 Interfaces

The PCI 9610 is a PCI Bus Master interface chip that connects a 64-bit, 66 MHz PCI Bus to a 64-bit, 66 MHz Processor Bus.

PCI r2.1 and r2.2 Compliant. Compliant with *PCI Local Bus Specification r2.1* and *r2.2*, including 64-bit, 66 MHz operation.

New Capabilities Structure. Supports New Capabilities registers to define additional capabilities of the PCI functions.

VPD Support. Supports the Vital Product Data (VPD) PCI extension through its serial EEPROM interface, providing an alternate to Expansion ROM for VPD access.

Power Management. Supports all five power states for PCI Power Management functions (D_0, D_1, D_2, D_{3hot}, and D_{3cold}) and Power Management Event interrupt (PME#) generation in all five states, including D_{3cold}.

CompactPCI Hot Swap Ready. Compliant with the Hot Swap *Ready* silicon requirements in the *PICMG 2.1 CompactPCI Hot Swap Specification, r1.0*.

PCI Hot-Plug Compliant. Compliant with *PCI Hot-Plug Specification, r1.0*.

Subsystem ID and Subsystem Vendor ID. Contains the Subsystem and Subsystem Vendor IDs in the PCI Configuration register space, in addition to the System and Vendor IDs. The PCI 9610 also contains a permanent Vendor ID (10B5h) and Device ID (9610h).

RST# Timing. Supports response to first configuration accesses after de-assertion of RST# under 2^{25} clocks.

Clocks. The PCI and Processor Bus clocks are independent and asynchronous. The Processor Bus interface runs from an external clock to provide the necessary internal clocks.

Processor Bus Direct Interface. 64-bit, 66 MHz Processor Bus interface supports direct connection to the Motorola MPC8260 PowerQUICC II, MPC603e, MPC740, MPC750, and other MPC603e bus protocol-compliant devices.

Enhanced Burst Mode. Protocol enhancements enable custom devices to support burst transfers beyond the four-beat (32-bit) limitation of the MPC8260 memory controller. During Extended Burst transfers, protocol enhancements are transparent to non-supporting devices to ensure legal bus operation.

1.4.2 Data Transfer

PCI ↔ Processor Burst Transfers up to 528 MB/s.

Eight Programmable FIFOs for Zero Wait State Burst Operation. The following table enumerates the FIFO depth.

Table 1-1. FIFO Depth

FIFO	Length
Direct Master Write	32 Qwords
Direct Master Read	16 Qwords
Direct Slave Write	32 Qwords
Direct Slave Read	16 Qwords
DMA Channel 0	32 Qwords
DMA Channel 1	32 Qwords
DMA Channel 2	32 Qwords
DMA Channel 3	32 Qwords

Unaligned Transfer Support. Capable of transferring data on any byte-boundary combination of the PCI and Processor Address spaces.

Big/Little Endian Conversion. Supports dynamic switching between Big Endian (Address Invariance) and Little Endian (Data Invariance) operations for Direct Slave, Direct Master, DMA, and internal register accesses on the Processor Bus.

Supports on-the-fly Endian conversion of Processor Bus data transfers. The Processor Bus can be Big or Little Endian by using the BIGEND# input pin or programmable internal register configuration. When BIGEND# is asserted, it overrides the internal register configuration during Direct Master, and internal register accesses on the Processor Bus.

Note: *The PCI Bus is always Little Endian.*

Keep Bus Mode. Supports program control to keep the PCI Bus by generating wait state(s) if the Direct Slave Write FIFO becomes full. Can also be programmed to keep the Processor Bus (BB# asserted) if the Direct Slave Write FIFO becomes empty or the Direct Slave Read FIFO becomes full. The Processor Bus is dropped in either case when the Processor Bus Latency Timer is enabled and expires.

Data Transfer Modes. Communicates with the Processor Bus devices, using five possible data transfer modes:

- Configuration Register Access
- Direct Master Operation
- Direct Slave Operation
- DMA Operation
- IDMA/SDMA Operation

Direct Bus Master. Supports PCI accesses from a Processor Bus master. Burst transfers are supported for memory-mapped devices and single transfers are supported for memory-mapped and I/O devices. Also supports PCI Bus interlock (LOCK#) cycles.

Direct Slave. Supports Burst Memory-Mapped and Single I/O-Mapped accesses to the Processor Bus. Supports 8-, 16-, and 32-bit Processor Bus data transfers. The Read and Write FIFOs enable high-performance bursting.

Three PCI-to-Processor Address Spaces. Supports three PCI-to-Processor Address spaces in Direct Slave mode—Space 0, Space 1, and Expansion ROM. These spaces allow any PCI Bus master to access the Processor Bus Memory spaces with programmable wait states, bus width, burst capabilities, and so forth.

Read Ahead Mode. Supports Read Ahead mode, where prefetched data can be read from the internal Read FIFO instead of the external bus. The address must be subsequent to the previous address and 64-bit aligned (next address = current address + 8). This feature allows for increased bandwidth and reduced data latency.

Programmable Prefetch Counter. Includes programmable control to prefetch data during Direct Slave and Direct Master, prefetches (known or unknown size). To perform burst reads, prefetching must be enabled. The prefetch size can be programmed to match the master burst length, or can be used as Read Ahead mode data. Reads single data (8-, 16-, 32-, or 64-bit) if the master initiates a single cycle; otherwise, prefetches the programmed size.

Posted Memory Writes. Supports the Posted Memory Writes (PMW) for maximum performance and to avoid potential deadlock situations.

Four DMA Channels with Independent FIFOs. Provides four independently programmable DMA channels with independently programmable FIFOs. Each channel supports Block and Scatter/Gather DMA modes, including ring management, as well as EOT mode. Supports Demand Mode DMA for all four channels.

PCI Dual-Address Cycle (DAC) Support (64-bit Address Space). Supports PCI Dual Address Cycle generation beyond the low 4-GB Address space. PCI DAC can be used during PCI 9610 PCI Bus Master operation (DMA and Direct Master).

1.4.3 Messaging Unit

I_2O-Ready Messaging Unit. Incorporates the I_2O-Ready Messaging Unit, which enables the adapter or embedded system to communicate with other I_2O-supported devices. The I_2O Messaging Unit is fully compatible with the PCI extension of *I_2O Specification r1.5*.

Mailbox Registers. Contains eight 32-bit Mailbox registers that may be accessed from the PCI or Processor Bus.

Doorbell Registers. Includes two 32-bit doorbell registers. One asserts interrupts from the PCI Bus to the Processor Bus. The other asserts interrupts from the Processor Bus to the PCI Bus.

1.4.4 Hosting Features

Type 0/1 Configuration. In Direct Master mode, supports Type 0 and Type 1 PCI Configuration cycles.

Internal PCI Arbiter. Includes an integrated PCI arbiter that supports seven external masters in addition to the PCI 9610.

Reset and Interrupt Signal Directions. Includes a strapping option to reverse the directions of the PCI and Processor Bus reset and interrupt signals.

1.4.5 Electrical/Mechanical

Packaging. Available in a 416-pin, 27 x 27 mm Fine Pitch PBGA package.

2.5V Core/3.3V I/O. Low power CMOS 2.5V core with 3.3V I/O.

5V Tolerant Operation. Provides 3.3V signaling with 5V I/O tolerance on both the PCI and Processor Bus.

Industrial Temperature Range Operation. The PCI 9610 works in a -40 to +85 °C temperature range.

JTAG. Supports IEEE 1149.1 JTAG boundary-scan.

1.4.6 Miscellaneous

Serial EEPROM Interface. Contains an optional serial EEPROM interface (optional only if using a local processor) that can be used to load configuration information. This is useful for loading information that is unique to a particular adapter (such as, the Network or Vendor ID).

Interrupt Generator. Can assert PCI and Processor interrupts from external and internal sources.

1.5 COMPATIBILITY WITH OTHER PLX CHIPS

1.5.1 Pin Compatibility

The PCI 9610 is ***not*** pin compatible with any other chips.

1.5.2 Register Compatibility

All registers implemented in the PCI 9060, PCI 9080, PCI 9054, and PCI 9656 are implemented in the PCI 9610. The PCI 9610 includes many new bit definitions and several new registers. (Refer to Section 9 for details.)

The PCI 9610 is ***not*** register-compatible with PLX target-only devices (*such as*, the PCI 9050, PCI 9052, and others).

1.5.3 PCI 9610 Comparison with Other PLX Chips

Table 1-2. Bus Master I/O Accelerator PLX Product Comparison

Features	PCI 9080-3	PCI 9054-AB50PI PCI 9054-AB50BI	PCI 9656-AA66BI	PCI 9610-AA66BI
Interfaces				
Host Bus Type	32-Bit, 33 MHz PCI, r2.1	32-Bit, 33 MHz PCI, r2.2	64-Bit, 66 MHz PCI, r2.2	64-Bit, 66 MHz PCI, r2.2
Processor/Local Bus Type(s): A = Address Bus D = Data Bus Mux = Multiplexed A/D Buses Non-Mux = Non-Multiplexed A/D Buses	C: Generic, 32-Bit A, 32-Bit D, non-mux J: Generic, 32-Bit A, 32-Bit D, mux S: Generic, 32-Bit A, 16-Bit D, mux	C: Generic, 32-Bit A, 32-Bit D, non-mux J: Generic, 32-Bit A, 32-Bit D, mux M: PowerPC PowerQUICC, 32-Bit A, 32-Bit D, non-mux	C: Generic, 32-Bit A, 32-Bit D, non-mux J: Generic, 32-Bit A, 32-Bit D, mux M: PowerPC PowerQUICC, 32-Bit A, 32-Bit D, non-mux	P: PowerPC PowerQUICC II, 32-Bit A, 64-bit D, non-mux
Maximum Processor Local Bus Speed	40 MHz	50 MHz	66 MHz	66 MHz
Core Voltage	5V	3.3V	2.5V	2.5V
I/O Ring Voltage	5V or 3.3V	3.3V	3.3V	3.3V
3.3V Tolerant PCI and Processor Buses	✓ (Only if 3.3V source to I/O Ring)	✓	✓	✓
5V Tolerant PCI and Processor Buses	✓ (Only if 5V source to I/O Ring)	✓	✓	✓
3.3V Tolerant Processor Bus	✓ (Only if 3.3V source to I/O Ring)	✓	✓	✓
5V Tolerant Processor Bus	✓ (Only if 5V source to I/O Ring)	✓	✓	✓
PICMG 2.1 CompactPCI Hot Swap r1.0	Capable	Friendly	Ready	Ready
Package Size/Type(s): Pin/Ball Count External Dimensions (mm) Pin/Ball Pitch (mm) Package Type	208-Pin, 30.6 x 30.6, .5 PQFP	176-Pin, 26 x 26, .5 PQFP 225-Ball, 27 x 27, 1.5 PBGA	272-Ball, 27 x 27, 1.27 PBGA	416-Ball, 27 x 27, 1.0 Fine-Pitch PBGA
Industrial Temperature Range Operation	✓	✓	✓	✓
Data Transfer				
Direct Slave Address Spaces	Two General-Purpose One Expansion ROM	Two General-Purpose One Expansion ROM	Two General-Purpose One Expansion ROM	Two General-Purpose One Expansion ROM
Direct Slave Read FIFO Depth	16 Lwords (64 bytes)	16 Lwords (64 bytes)	16 Qwords (128 bytes)	16 Qwords (128 bytes)
Direct Slave Write FIFO Depth	32 Lwords (128 bytes)	32 Lwords (128 bytes)	32 Qwords (256 bytes)	32 Qwords (256 bytes)
PCI r2.1 Delayed Read Support	✓	✓	✓	✓

Table 1-2. Bus Master I/O Accelerator PLX Product Comparison (Continued)

Features	PCI 9080-3	PCI 9054-AB50PI PCI 9054-AB50BI	PCI 9656-AA66BI	PCI 9610-AA66BI
Data Transfer (Continued)				
Programmable READY# Timeout			✓	✓
Direct Master Address Spaces	1	1	1	1
Direct Master Read FIFO Depth	16 Lwords (64 bytes)	16 Lwords (64 bytes)	16 Qwords (128 bytes)	16 Qwords (128 bytes)
Direct Master Write FIFO Depth	32 Lwords (128 bytes)	32 Lwords (128 bytes)	32 Qwords (256 bytes)	32 Qwords (256 bytes)
DMA Channels	2	2	2	4
DMA Channel 0 FIFO Depth	32 Lwords (128 bytes) Bi-directional	32 Lwords (128 bytes) Bi-directional	32 Qwords (256 bytes) Bi-directional	32 Qwords (256 bytes) Bi-directional
DMA Channel 1 FIFO Depth	16 Lwords (64 bytes) Bi-directional	16 Lwords (64 bytes) Bi-directional	32 Qwords (256 bytes) Bi-directional	32 Qwords (256 bytes) Bi-directional
DMA Channel 2 FIFO Depth				32 Qwords (256 bytes) Bi-directional
DMA Channel 3 FIFO Depth				32 Qwords (256 bytes) Bi-directional
DMA Demand Mode Hardware Control	✓	✓ (Channel 0 Only)	✓	✓
DMA EOT Mode Hardware Control	✓	✓	✓	✓
DMA Block Mode	✓	✓	✓	✓
DMA Scatter/Gather Mode	✓	✓	✓	✓
DMA Ring Management Mode			✓	✓
Programmable Prefetch Counter	✓	✓	✓	✓
Dual Address Cycle Generation		✓	✓	✓
Big Endian/Little Endian Conversion	✓	✓	✓	✓
Control				
Mailbox Registers	Eight 32-Bit	Eight 32-Bit	Eight 32-Bit	Eight 32-Bit
Doorbell Registers	Two 32-Bit	Two 32-Bit	Two 32-Bit	Two 32-Bit
I_2O Messaging Unit	✓ r1.5	✓ r1.5	✓ r1.5	✓ r1.5
PCI Arbiter			✓ Seven external masters	✓ Seven external masters
PCI Type 0/1 Configuration	✓	✓	✓	✓
PCI Power Management		✓ r1.1	✓ r1.1	✓ r1.1
D_{3cold} PME Generation			✓	✓
PCI r2.2 VPD Support		✓	✓	✓
Serial EEPROM Support	1K bit, 2K bit microwire devices with sequential read support	2K bit, 4K bit microwire devices with sequential read support	2K bit, 4K bit microwire devices with sequential read support	2K bit, 4K bit microwire devices with sequential read support
JTAG Boundary Scan			✓	✓
Register Compatibility		Backward compatible with PCI 9080	Backward compatible with PCI 9054	Backward compatible with PCI 9054

2 BUS OPERATION

2.1 PCI BUS CYCLES

The PCI 9610 is compliant with *PCI Specification r2.2*. Refer to *PCI Specification r2.2* for specific PCI Bus functions.

2.1.1 Direct Slave Command Codes

As a Target, the PCI 9610 allows external masters to access the PCI 9610 internal registers and 60x Bus, using the commands listed in Table 2-1.

All Read or Write accesses to the PCI 9610 can be Byte, Word, or Longword (Lword) accesses, defined as 32 bit. All memory commands are aliased to basic memory commands. All I/O accesses to the PCI 9610 are decoded to an Lword boundary. Byte enables are used to determine which bytes are read or written. An I/O access with illegal byte enable combinations is terminated with a Target abort.

Table 2-1. Direct Slave Command Codes

Command Type	Code (C/BE[7:0]#)
I/O Read	0010 (2h)
I/O Write	0011 (3h)
Memory Read	0110 (6h)
Memory Write	0111 (7h)
Configuration Read	1010 (Ah)
Configuration Write	1011 (Bh)
Memory Read Multiple	1100 (Ch)
Memory Read Line	1110 (Eh)
Memory Write and Invalidate	1111 (Fh)

2.1.2 PCI Master Command Codes

The PCI 9610 can access the PCI Bus to perform DMA or Direct Master 60x-to-PCI Bus transfers. During a Direct Master or DMA transfer, the command code assigned to the PCI 9610 internal register location (CNTRL[15:0]) is used as the PCI command code (except for Memory Write and Invalidate mode for DMA cycles where DMPBAM[9]=1).

Notes: *Programmable internal registers determine PCI command codes when the PCI 9610 is the Master.*

DMA cannot perform I/O or Configuration accesses.

2.1.2.1 DMA Master Command Codes

DMA controllers of the PCI 9610 can assert the Memory cycles listed in Table 2-2.

Table 2-2. DMA Master Command Codes

Command Type	Code (C/BE[7:0]#)
Memory Read	0110 (6h)
Memory Write	0111 (7h)
Memory Read Multiple	1100 (Ch)
PCI Dual Address Cycle	1101 (Dh)
Memory Read Line	1110 (Eh)
Memory Write and Invalidate	1111 (Fh)

2.1.2.2 Direct Master 60x-to-PCI Command Codes

For Direct Master 60x-to-PCI Bus accesses, the PCI 9610 asserts the cycles listed in Table 2-3 through Table 2-5.

Table 2-3. 60x-to-PCI Memory Access

Command Type	Code (C/BE[7:0]#)
Memory Read	0110 (6h)
Memory Write	0111 (7h)
Memory Read Multiple	1100 (Ch)
PCI Dual Address Cycle	1101 (Dh)
Memory Read Line	1110 (Eh)
Memory Write and Invalidate	1111 (Fh)

Table 2-4. 60x-to-PCI I/O Access

Command Type	Code (C/BE[7:0]#)
I/O Read	0010 (2h)
I/O Write	0011 (3h)

Table 2-5. 60x-to-PCI Configuration Access

Command Type	Code (C/BE[7:0]#)
Configuration Memory Read	1010 (Ah)
Configuration Memory Write	1011 (Bh)

2.1.3 PCI Arbitration

The PCI 9610 asserts REQ# to request the PCI Bus. The PCI 9610 can be programmed using the PCI Request Mode bit (MARBR[23]) to de-assert REQ# when it asserts FRAME# during a Bus Master cycle, or to keep REQ# asserted for the entire Bus Master cycle. The PCI 9610 always de-asserts REQ# for a minimum of two PCI clocks between Bus Master ownership that includes a Target disconnect.

The Direct Master Write Delay bits (DMPBAM[15:14]) can be programmed to delay the PCI 9610 from asserting PCI REQ# during a Direct Master Write cycle. DMPBAM can be programmed to wait 0, 4, 8, or 16 PCI Bus clocks after the PCI 9610 received its first Write data from the 60x Bus Master and is ready to begin the PCI Write transaction. This function is useful in applications where a 60x Bus Master is bursting and a 60x Bus clock is slower than the PCI Bus clock. This allows Write data to accumulate in the PCI 9610 Direct Master Write FIFO, which provides for better use of the PCI Bus.

2.2 60X BUS CYCLES

The 60x Bus is described in Section 3.

2.3 SERIAL EEPROM

Functional operation described can be modified through the PCI 9610 programmable internal registers.

2.3.1 Vendor and Device ID Registers

Three Vendor and Device ID registers are supported:

- **PCIIDR**—Contains normal Device and Vendor IDs. Can be loaded from the serial EEPROM or 60x Bus Master(s).
- **PCISVID**—Contains Subsystem and Subvendor IDs. Can be loaded from the serial EEPROM or 60x Bus Masters(s).
- **PCIHIDR**—Contains hardcoded PLX Vendor and Device IDs.

2.3.1.1 Serial EEPROM Initialization

During serial EEPROM initialization, the PCI 9610 responds to Direct Slave accesses with a Retry. During serial EEPROM initialization, the PCI 9610 responds to a 60x Bus Master access by delaying acknowledgement of the cycle (TA#).

2.3.1.2 60x Bus Initialization

The PCI 9610 issues a Retry to all PCI accesses until the 60x Bus Init Status bit is set (LMISC1[2=1]). This bit can be programmed three different ways:

1. By the 60x Bus Master, through the 60x Bus Configuration register.
2. By the serial EEPROM, during a serial EEPROM load, if the 60x Bus Master does not set this bit or if this bit is missing.
3. If the 60x Bus Master and/or the serial EEPROM are missing, the serial EEPROM remains blank and the PCI 9610 reverts to the default values and sets this bit. (Refer to Table 2-6 on page 2-3.)

2.3.2 Serial EEPROM Operation

After reset, the PCI 9610 attempts to read the serial EEPROM to determine its presence. An active Start bit set to 0 indicates a serial EEPROM is present. The PCI 9610 supports 93CS56L (2K bit) or 93CS66L (4K bit). (Refer to manufacturer's data sheet for the particular serial EEPROM being used.) The first Lword is then checked to verify that the serial EEPROM is programmed. If the first Lword (33 bits) is all ones, a blank serial EEPROM is present. If the first Lword (33 bits) is all zeros, no serial EEPROM is present. For both conditions, the PCI 9610 reverts to the default values. (Refer to Table 2-6.) The Programmed Serial EEPROM Present bit is set (CNTRL[28]=1) if the serial EEPROM is programmed (real or random data if a serial EEPROM is detected).

The 3.3V serial EEPROM clock (EESK) is derived from the PCI clock. The PCI 9610 generates the serial EEPROM clock by internally dividing the PCI clock by 268. For a 66.6 MHz PCI Bus, EESK is 248.7 kHz; for a 33.3 MHz PCI Bus, EESK is 124.4 kHz.

The serial EEPROM can be read or written from the PCI or 60x Buses. The Serial EEPROM Control Register bits (CNTRL[28:24]) control the PCI 9610 pins that enable reading or writing of serial EEPROM data bits. (Refer to manufacturer's data sheet for the particular serial EEPROM being used.)

The PCI 9610_AA revision provides the ability to manually access the serial EEPROM. This may be accomplished by using CNTRL[31, 27:24] (EESK, EECS, and EEDI/EEDO, controlled by software). Bit 24 is used to generate EESK (clock), bit 25 controls the chip select, and bit 31 enables the EEDO input buffer. Bit 27, when read, returns the value of EEDO.

Setting bits 24, 25, and 31 to 1 causes the output to go high. A pull-up resistor is required on EEDO to go high when bit 31 is set. When reading the serial EEPROM, bit 31 must be set to 1.

To perform the read, the basic approach is to set the EECS and EEDO bits (bits 25 and 31, respectively) to the desired level and then toggle EESK high and low until done. *For example*, reading the serial EEPROM at location 0 involves the following steps:

1. Clear EESK, EEDO and EECS bits.
2. Toggle EESK high, then low.
3. Set EECS high.
4. Toggle EESK high, then low.
5. Set EEDO bit high (start bit).
6. Toggle EESK high, then low.
7. Repeat step 6.
8. Clear EEDO.
9. Toggle EESK bit high, then low eight times (clock in 60x Bus Address 0).
10. Set EEDO to float the EEDO pin for reading.
11. Toggle EESK high, then low 16 times (clock in one word from serial EEPROM).
12. After each clock pulse, read bit 27 and save.
13. Clear EECS bit.
14. Toggle EESK high, then low.
15. Read is now complete.

The serial EEPROM can also be read or written, using the VPD function. (Refer to Section 8.)

The PCI 9610 has two serial EEPROM load options:

- **Long Load Mode**—Default. The PCI 9610 loads 17 Lwords from the Serial EEPROM and the Extra Long Load from Serial EEPROM bit (LBRD0[25])
- **Extra Long Load Mode**—The PCI 9610 loads 23 Lwords if the Serial EEPROM and the Extra Long Load from Serial EEPROM bit is set (LBRD0[25]=1) during a Long Load

Table 2-6. Serial EEPROM Guidelines

60x Bus Processor	Serial EEPROM	System Boot Condition
None	None	The PCI 9610 uses default values. The EEDI/EEDO pin must be **pulled low**—a 1K ohm resistor is required (rather than pulled high, which is typically done for this pin). If the PCI 9610 detects all zeros, it reverts to default values.
None	Programmed	Boot with serial EEPROM values. The 60x Bus Init Status bit (LMISC1[2]) must be set by the serial EEPROM.
None	Blank	The PCI 9610 detects a blank device and reverts to default values.
Present	None	The 60x Bus Master programs the PCI 9610 registers, then sets the 60x Bus Init Status bit (LMISC1[2]=done). ***Note:*** *Some systems may hang if Direct Slave reads and writes take overly long (during initialization, the PCI Master also performs Direct Slave accesses). The value of the Direct Slave Retry Delay Clocks (LBRD0[31:28]) may resolve this.*
Present	Programmed	Load serial EEPROM, but the 60x Bus Master can reprogram the PCI 9610. The 60x Bus Master or the serial EEPROM must set the 60x Bus Init Status bit (LMISC1[2]=done).
Present	Blank	The PCI 9610 detects a blank serial EEPROM and reverts to default values. ***Notes:*** *In some systems, the 60x Bus Master may be overly late to reconfigure the PCI 9610 registers before the BIOS configures them.* *The serial EEPROM can be programmed through the PCI 9610 after the system boots in this condition.*

Note: *If the serial EEPROM is missing and a 60x Bus Master is present with blank Flash, the condition None/None (as seen in Table 2-6) applies, until the Processor's Flash is programmed.*

2.3.2.1 Long Serial EEPROM Load

The registers listed in Table 2-7 are loaded from the serial EEPROM after a reset is de-asserted if the Extra Long Load from Serial EEPROM bit is not set (LBRD0[25]=0). The serial EEPROM is organized in words (16 bit). The PCI 9610 first loads the Most Significant Word bits (MSW[31:16]), starting from the Most Significant bit (MSB[31]). The PCI 9610 then loads the Least Significant Word bits (LSW[15:0]), starting again from the Most Significant bit (MSB[15]). Therefore, the PCI 9610 loads the Device ID, Vendor ID, class code, and so forth.

The serial EEPROM values can be programmed using a Data I/O programmer. The values can also be programmed using the PCI 9610 VPD function (refer to Section 8) or through the Serial EEPROM Control register (CNTRL).

The CNTRL register allows programming of the serial EEPROM, one bit at a time. To read back the value from the serial EEPROM, the CNTRL[27] bit (refer to Section 2.3.2) or the VPD function should be utilized. With full utilization of VPD, the designer can perform reads and writes from/to the serial EEPROM, 32 bits at a time. Values should be programmed in the order listed in Table 2-7. The 34, 16-bit words listed in the table are stored sequentially in the serial EEPROM.

2.3.2.2 Extra Long Serial EEPROM Load

The registers listed in the 60x Bus Address Space 0/ Expansion ROM Bus Region Descriptor register (LBRD0) are loaded from serial EEPROM after a reset is de-asserted if the Extra Long Load from Serial EEPROM bit is set (LBRD0[25]=1). The serial EEPROM is organized in words (16 bit). The PCI 9610 first loads the Most Significant Word bits (MSW[31:16]), starting from the Most Significant bit (MSB[31]). It then loads the Least Significant Word bits (LSW[15:0]), restarting from the Most Significant bit (MSB[15]). Therefore, the PCI 9610 loads Device ID, Vendor ID, class code, and so forth.

The serial EEPROM values can be programmed using a Data I/O programmer. The values can also be programmed using the PCI 9610 VPD function or through the Serial EEPROM Control register (CNTRL).

Values should be programmed in the order listed in Table 2-8 on page 2-6. The 46 16-bit words listed in Table 2-7 and Table 2-8 should be stored sequentially in the serial EEPROM.

Table 2-7. Long Serial EEPROM Load Registers

Serial EEPROM Offset	Description	Register Bits Affected
0h	Device ID	PCIIDR[31:16]
2h	Vendor ID	PCIIDR[15:0]
4h	Class Code	PCICCR[23:8]
6h	Class Code / Revision	PCICCR[7:0] / PCIREV[7:0]
8h	Maximum Latency / Minimum Grant	PCIMLR[7:0] / PCIMGR[7:0]
Ah	Interrupt Pin / Interrupt Line Routing	PCIIPR[7:0] / PCIILR[7:0]
Ch	MSW of Mailbox 0 (User Defined)	MBOX0[31:16]
Eh	LSW of Mailbox 0 (User Defined)	MBOX0[15:0]
10h	MSW of Mailbox 1 (User Defined)	MBOX1[31:16]
12h	LSW of Mailbox 1 (User Defined)	MBOX1[15:0]
14h	MSW of Range for PCI-to-60x Bus Address Space 0	LAS0RR[31:16]
16h	LSW of Range for PCI-to-60x Bus Address Space 0	LAS0RR[15:0]
18h	MSW of 60x Bus Base Address (Remap) for PCI-to-60x Bus Address Space 0	LAS0BA[31:16]
1Ah	LSW of 60x Bus Base Address (Remap) for PCI-to-60x Bus Address Space 0	LAS0BA[15:0]
1Ch	MSW of Mode/DMA Arbitration Register	MARBR[31:16]
1Eh	LSW of Mode/DMA Arbitration Register	MARBR[15:0]
20h	MSW of 60x Bus Miscellaneous Control Register 2 / MSW of Serial EEPROM Write-Protected Address	LMISC2[7:0] / PROT_AREA[7:0]
22h	LSW of 60x Bus Miscellaneous Control Register 1/ LSW of 60x Bus Big/Little Endian Descriptor Register	LMISC1[7:0] / BIGEND[7:0]
24h	MSW of Range for PCI-to-60x Bus Expansion ROM	EROMRR[31:16]
26h	LSW of Range for PCI-to-60x Bus Expansion ROM	EROMRR[15:0]
28h	MSW of 60x Bus Base Address (Remap) for PCI-to-60x Bus Expansion ROM	EROMBA[31:16]
2Ah	LSW of 60x Bus Base Address (Remap) for PCI-to-60x Bus Expansion ROM	EROMBA[15:0]
2Ch	MSW of Bus Region Descriptors for PCI-to-60x Bus Accesses	LBRD0[31:16]
2Eh	LSW of Bus Region Descriptors for PCI-to-60x Bus Accesses	LBRD0[15:0]
30h	MSW of Range for Direct Master-to-PCI	DMRR[31:16]
32h	LSW of Range for Direct Master-to-PCI	DMRR[15:0]
34h	MSW of 60x Bus Base Address for Direct Master-to-PCI Memory	DMLBAM[31:16]
36h	LSW of 60x Bus Base Address for Direct Master-to-PCI Memory	DMLBAM[15:0]
38h	MSW of 60x Bus Address for Direct Master-to-PCI I/O Configuration	DMLBAI[31:16]
3Ah	LSW of 60x Bus Address for Direct Master-to-PCI I/O Configuration	DMLBAI[15:0]
3Ch	MSW of PCI Base Address (Remap) for Direct Master-to-PCI	DMPBAM[31:16]
3Eh	LSW of PCI Base Address (Remap) for Direct Master-to-PCI	DMPBAM[15:0]
40h	MSW of PCI Configuration Address Register for Direct Master-to-PCI I/O Configuration	DMCFGA[31:16]
42h	LSW of PCI Configuration Address Register for Direct Master-to-PCI I/O Configuration	DMCFGA[15:0]

Table 2-8. Extra Long Serial EEPROM Load Registers

Serial EEPROM Offset	Description	Register Bits Affected
44h	Subsystem ID	PCISID[15:0]
46h	Subsystem Vendor ID	PCISVID[15:0]
48h	MSW of Range for PCI-to-60x Bus Address Space 1 (1 MB)	LAS1RR[31:16]
4Ah	LSW of Range for PCI-to-60x Bus Address Space 1 (1 MB)	LAS1RR[15:0]
4Ch	MSW of 60x Bus Base Address (Remap) for PCI-to-60x Bus Address Space 1	LAS1BA[31:16]
4Eh	LSW of 60x Bus Base Address (Remap) for PCI-to-60x Bus Address Space 1	LAS1BA[15:0]
50h	MSW of Bus Region Descriptors (Space 1) for PCI-to-60x Bus Accesses	LBRD1[31:16]
52h	LSW of Bus Region Descriptors (Space 1) for PCI-to-60x Bus Accesses	LBRD1[15:0]
54h	MSW of Hot Swap Control/Status	***Reserved***
56h	LSW of Hot Swap Control / Hot Swap Next Capability Pointer	HS_NEXT[7:0] / HS_CNTL[7:0]
58h	PCI Arbiter Control	PCIARB[3:0]
5Ah	***Reserved***	***Reserved***

2.3.2.3 New Capabilities Function Support

The New Capabilities Function Support includes PCI Power Management, Hot Swap, and VPD features, as listed in Table 2-9.

Table 2-9. New Capabilities Function Support Features

New Capability Function	PCI Register Offset Location
First (Power Management)	40h, if the New Capabilities Function Support bit (PCISR[4]) is enabled (PCISR[4] is enabled, by default).
Second (Hot Swap)	48h, which is pointed to from PMNEXT[7:0].
Third (VPD)	4Ch, which is pointed to from HS_NEXT[7:0]. Because PVPD_NEXT[7:0] defaults to zero, this indicates that VPD is the last New Capability Function Support feature of the PCI 9610.

2.3.2.4 Recommended Serial EEPROMs

The PCI 9610 is designed to use either a 2K bit (NM93CS56L or compatible) or 4K bit (NM93CS66L or compatible) device.

Note: *The PCI 9610 does not support serial EEPROMs that do not support sequential reads and writes (such as the NM93C56L).*

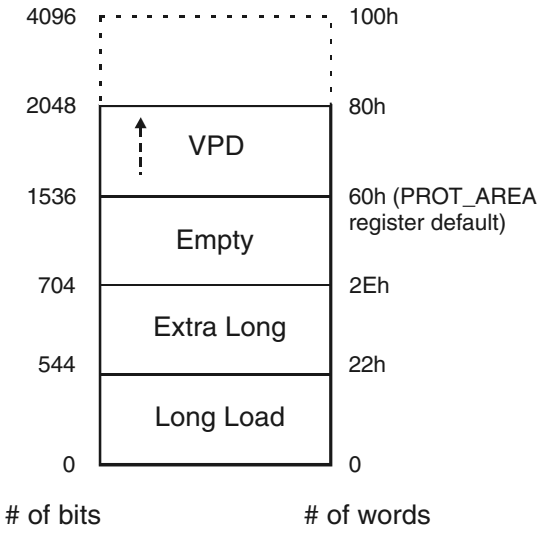


Figure 2-1. Serial EEPROM Memory Map

2.3.2.5 Serial EEPROM Initialization

During serial EEPROM initialization, the PCI 9610 responds to Direct Slave accesses with a Retry. During serial EEPROM initialization, the PCI 9610 responds to a 60x Bus Master access by delaying acknowledgement of the cycle (TA#).

2.3.3 Internal Register Access

The PCI 9610 provides several internal registers, which allow for maximum flexibility in the bus interface design and performance. These registers are accessible from the PCI and 60x Buses (refer to Figure 2-2) and include the following:

- PCI and 60x Bus Configuration registers
- DMA registers
- Mailbox registers
- PCI-to-60x Bus and 60x-to-PCI Bus Doorbell registers
- Messaging Queue registers (I_2O)
- Power Management registers
- Hot Swap registers
- VPD registers

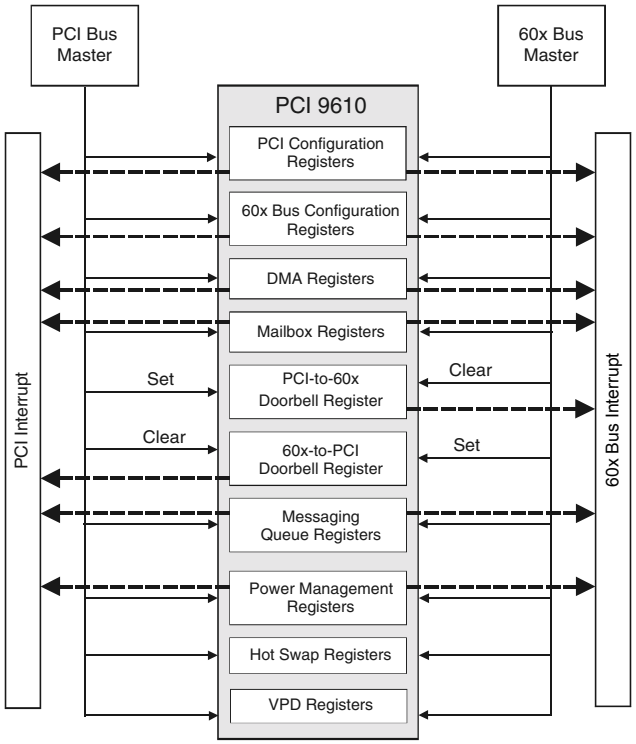


Figure 2-2. PCI 9610 Internal Register Access

2.3.3.1 PCI Bus Access to Internal Registers

The PCI 9610 PCI Configuration registers can be accessed from the PCI Bus with a Configuration Type 0 cycle.

All other PCI 9610 internal registers can be accessed by a Memory cycle, with the PCI Bus address that matches the base address specified in PCI Base Address 0 (PCIBAR0[31:8]) for the PCI 9610 Memory-Mapped Configuration register. These registers can also be accessed by an I/O cycle, with the PCI Bus address matching the base address specified in PCI Base Address 1 for the PCI 9610 I/O-Mapped Configuration register (PCIBAR1).

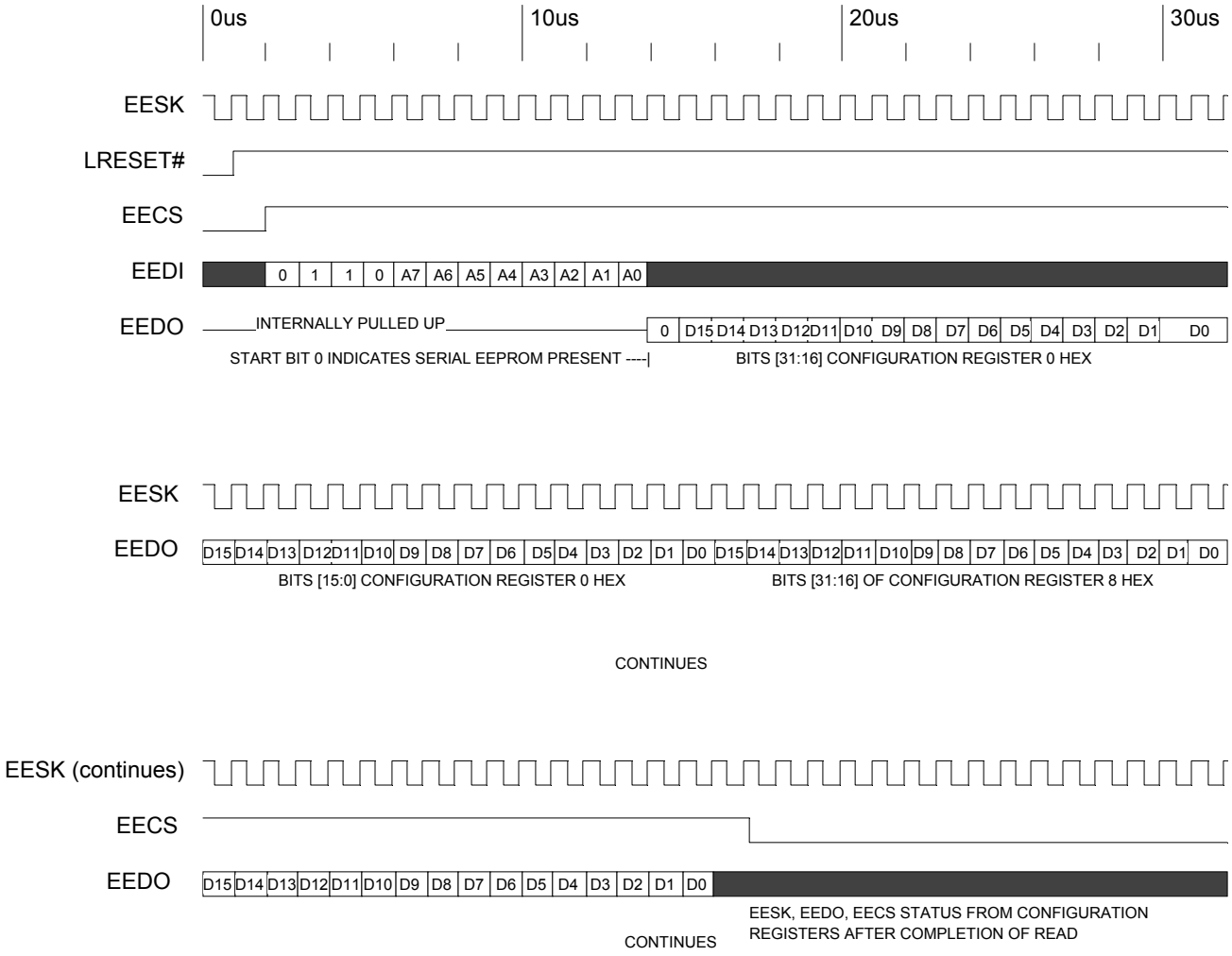
All PCI Read or Write accesses to the PCI 9610 registers can be Byte, Word, or Lword accesses. All PCI Memory accesses to the PCI 9610 registers can be Burst or Non-Burst accesses. The PCI 9610 responds with a PCI disconnect for all Burst I/O accesses (PCIBAR1[31:8]) to the PCI 9610 Internal registers.

2.3.3.2 60x Bus Access to Internal Registers

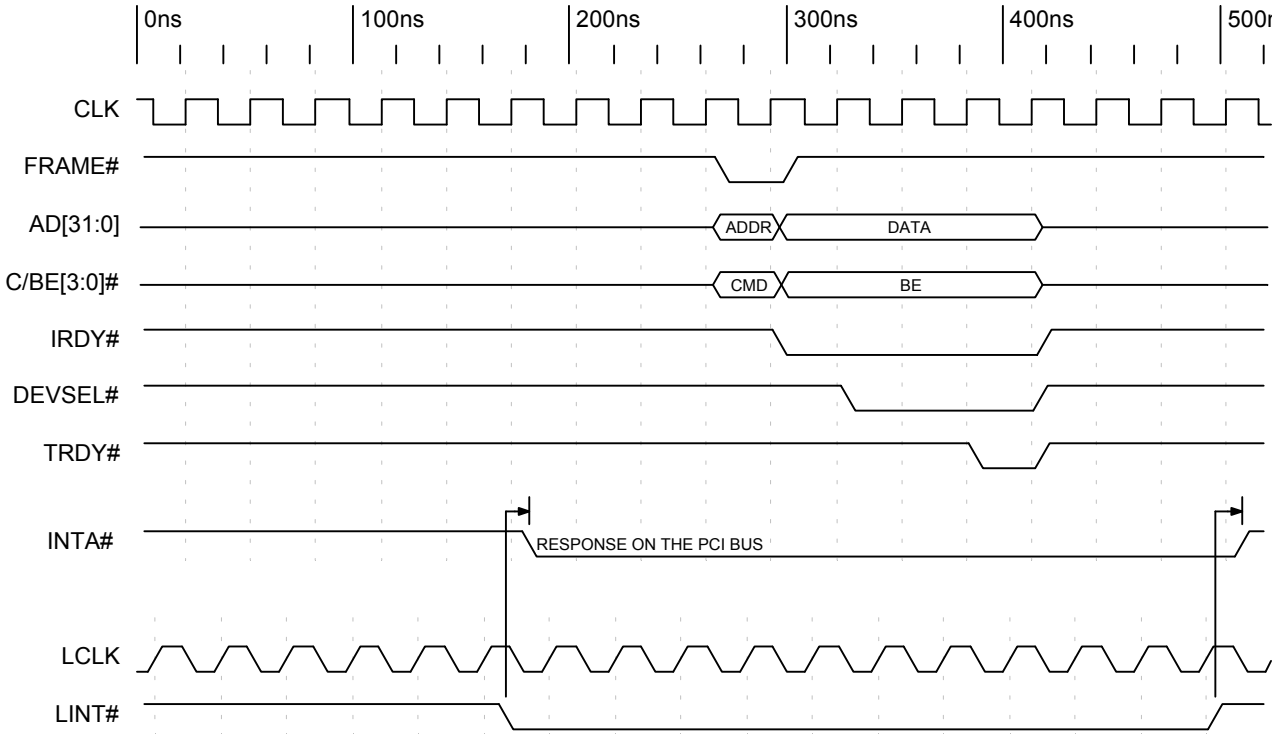
60x Bus access to internal registers is described in Section 3.6.5.3, “Accessing Internal PCI 9610 Registers Using CCS#,” on page 3-36.

2.3.4 Serial EEPROM and Configuration Initialization Timing Diagrams

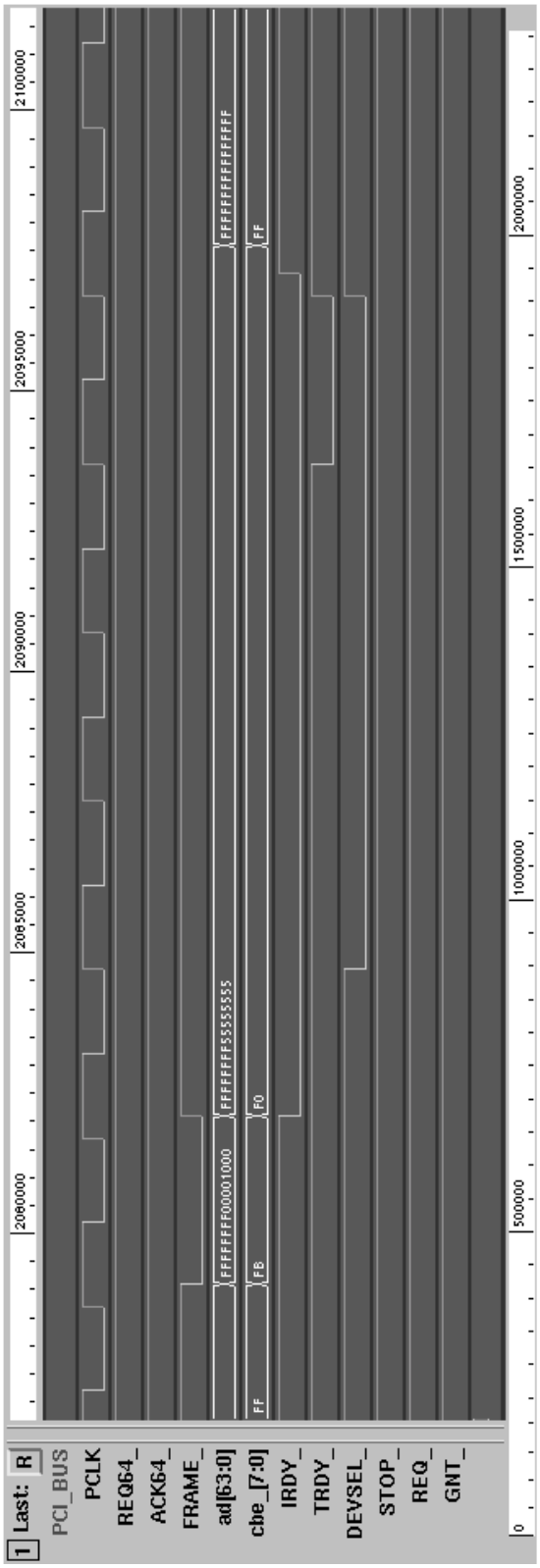
Note: *In the timing diagrams that follow, the "_" symbol at the end of the signal names represents the "#" symbol.*



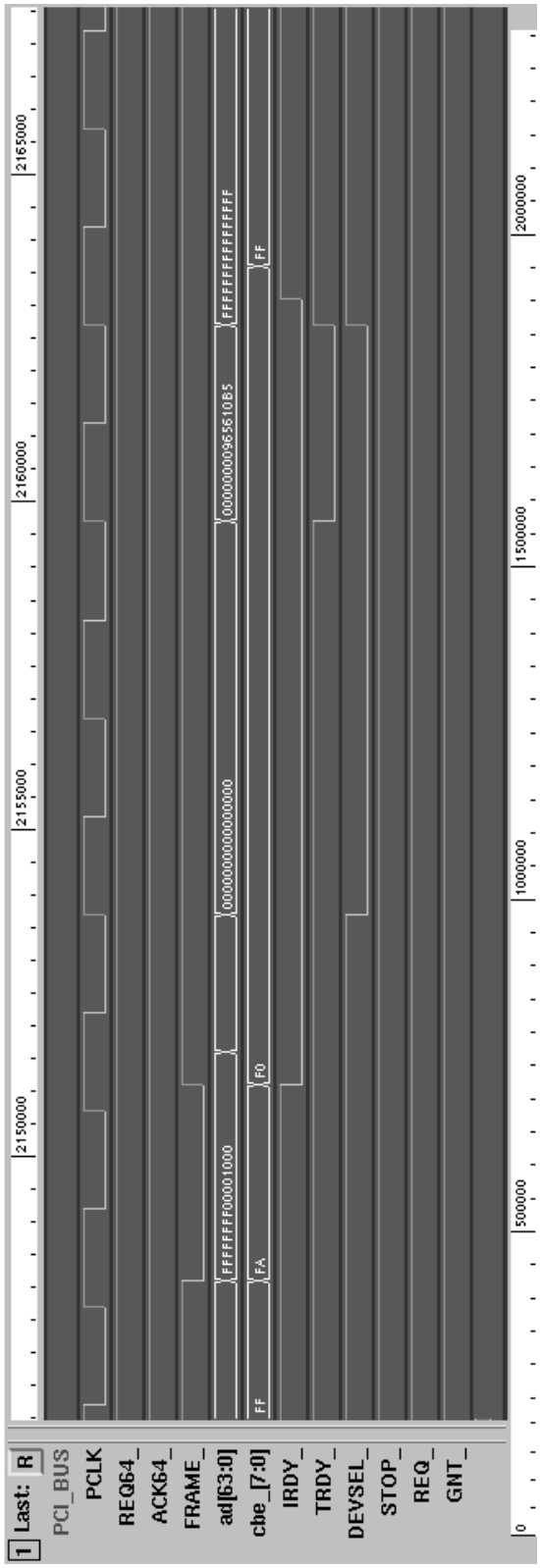
Timing Diagram 2-1. Initialization from Serial EEPROM (2K or 4K Bit)



Timing Diagram 2-2. 60x Bus Interrupt Asserting PCI Interrupt



Timing Diagram 2-3. PCI Configuration Write to PCI Configuration Register



Timing Diagram 2-4. PCI Configuration Read to PCI Configuration Register

3 FUNCTIONAL DESCRIPTION

The functional operation described in this chapter can be modified through the PCI 9610 programmable internal registers.

3.1 RESET OPERATION

3.1.1 Adapter Mode

3.1.1.1 PCI Bus Input RST#

The PCI Bus RST# input pin is a PCI Master reset. It causes all PCI Bus outputs to float, resets the entire PCI 9610 and causes the 60x Bus LRESET# signal to assert.

3.1.1.2 Software Reset

A Master on the PCI Bus can set the PCI Adapter Software Reset bit (CNTRL[30]=1) to reset the PCI 9610 and assert LRESET# output. All 60x Bus Configuration registers are reset; however, the PCI Configuration DMA and Shared Runtime registers and the 60x Bus Init Status bit (LMISC1[2]) are not reset. When the Software Reset bit (CNTRL[30]) is set, the PCI 9610 responds to PCI accesses, but not to 60x Bus accesses. The PCI 9610 remains in this reset condition until the PCI Master clears the bit. The serial EEPROM is reloaded, if the Reload Configuration Registers bit is set (CNTRL[29]=1).

Note: *The 60x Bus cannot clear this reset bit, as the 60x Bus is in a reset state, even if the 60x Bus Master does not use LRESET# to reset.*

3.1.1.3 Power Management Reset

When the power management reset is asserted (transition from D_3 to any other state), the PCI 9610 resets as if a PCI reset was asserted. (Refer to Section 6.)

3.1.2 Host Mode

3.1.2.1 PCI Reset

The PCI Bus RST# output is driven when the 60x Bus LRESET# signal is asserted, the Software Reset bit is set (CNTRL[30]=1), or the PCI 9610 initiates an external reset.

3.1.2.2 60x Bus LRESET#

When the 60x Bus LRESET# pin is asserted by an external source, the 60x Bus interface circuitry, the Configuration registers, and the PCI 9610 are reset. The PCI 9610 drives the 60x Bus LRESET# pin after it detects a reset for 62 clocks.

3.1.2.3 Software Reset

When the Software Reset bit is set (CNTRL[30]=1), the following occurs:

- PCI Master logic is held reset
- PCI 9610 PCI Configuration registers are held in reset
- FIFOs are reset
- PCI RST# pin is asserted

Only the PCI Configuration registers are in reset. A software reset can only be cleared from another Master on the 60x Bus, and the PCI 9610 remains in this reset until a 60x Bus Master clears the bit.

Note: *The PCI Bus cannot clear this reset bit because the PCI Bus is in a reset state.*

3.1.2.4 Power Management Reset

Power Management reset is not applicable for Host mode.

3.2 PCI 9610 INITIALIZATION

The PCI 9610 Configuration registers can be programmed by an optional serial EEPROM and/or by a 60x Bus Master, as listed in Table 2-6, “Serial EEPROM Guidelines,” on page 2-3. The serial EEPROM can be reloaded by setting the Reload Configuration Registers bit (CNTRL[29]=1).

The PCI 9610 Retries all PCI cycles until the 60x Bus Init Status bit is set to “done” (LMISC1[2]=1).

Note: *The PCI Master can also access Internal Configuration registers after the 60x Bus Init Status bit is set.*

If a PCI Master is present, the Master Enable, Memory Space, and I/O Space bits (PCICR[2:0], respectively) are programmed by that Master after initialization completes (LMISC1[2]=1).

3.3 RESPONSE TO FIFO FULL OR EMPTY

Table 3-1 on page 3-2 lists the PCI 9610 response to full and empty FIFOs.

3.4 DIRECT DATA TRANSFER MODES

The PCI 9610 supports three direct transfer modes:

- **Direct Master**—60x Bus Master accesses PCI memory or I/O
- **Direct Slave**—PCI Master accesses the 60x Bus device
- **DMA**—PCI 9610 DMA controller transfers data between the PCI and 60x Bus devices.

3.4.1 Direct Master Operation (60x Bus Master-to-Direct Slave)

The PCI 9610 supports a direct access of the PCI Bus by the 60x Bus Master or an intelligent controller. Master mode must be enabled in the PCI Command register. The following registers define 60x-to-PCI accesses:

- Direct Master Memory and I/O Range (DMRR)
- 60x Bus Base Address for Direct Master-to-PCI Memory (DMLBAM)
- 60x Bus Base Address for Direct Master-to-PCI I/O and Configuration (DMLBAI)
- PCI Base Address (DMPBAM)
- Direct Master Configuration (DMCFGA)
- Direct Master PCI Dual Address Cycles (DMDAC)
- Master Enable (PCICR)
- PCI Command Code (CNTRL)

Table 3-1. Response to FIFO Full or Empty

Mode	Direction	FIFO	PCI Bus	60x Bus
Direct Master Write	60x-to-PCI	Full	Normal	PCI 9610 asserts ARTRY#
		Empty	De-assert REQ# (off the PCI Bus)	Normal
Direct Master Read	PCI-to-60x	Full	De-assert REQ# or throttle IRDY#[1]	Normal
		Empty	Normal	PCI 9610 asserts ARTRY#, if delayed read is enabled
Direct Slave Write	PCI-to-60x	Full	Disconnect or throttle TRDY#[2]	Normal
		Empty	Normal	Normal
Direct Slave Read	60x-to-PCI	Full	Normal	Normal
		Empty	Throttle TRDY#[3]	Normal
DMA	60x-to-PCI	Full	Normal	Normal
		Empty	De-assert REQ#	Normal
	PCI-to-60x	Full	De-assert REQ#	Normal
		Empty	Normal	Normal

1. Throttle IRDY# depends upon the Direct Master PCI Read Mode bit (DMPBAM[4]).
2. Throttle TRDY# depends upon the Direct Slave PCI Write Mode bit (LBRD0[27]).

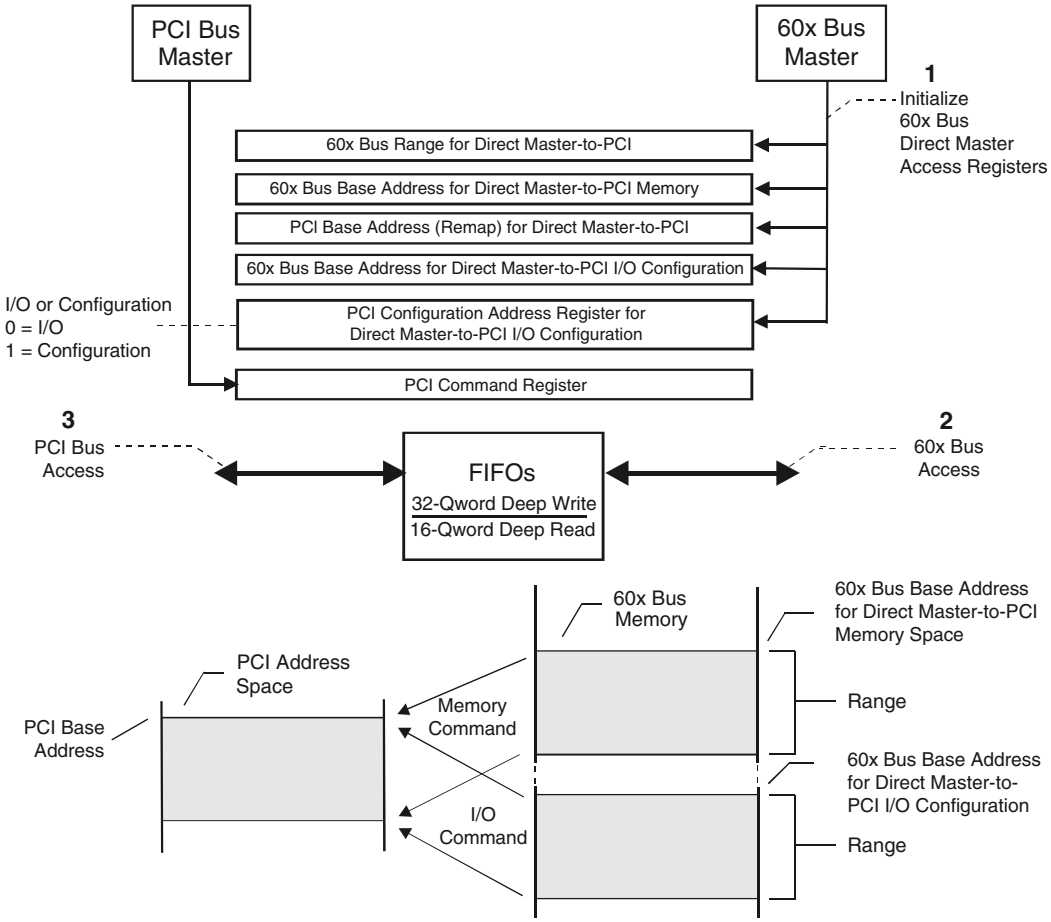


Figure 3-1. Direct Master Access to the PCI Bus

3.4.1.1 Direct Master Memory and I/O Decode

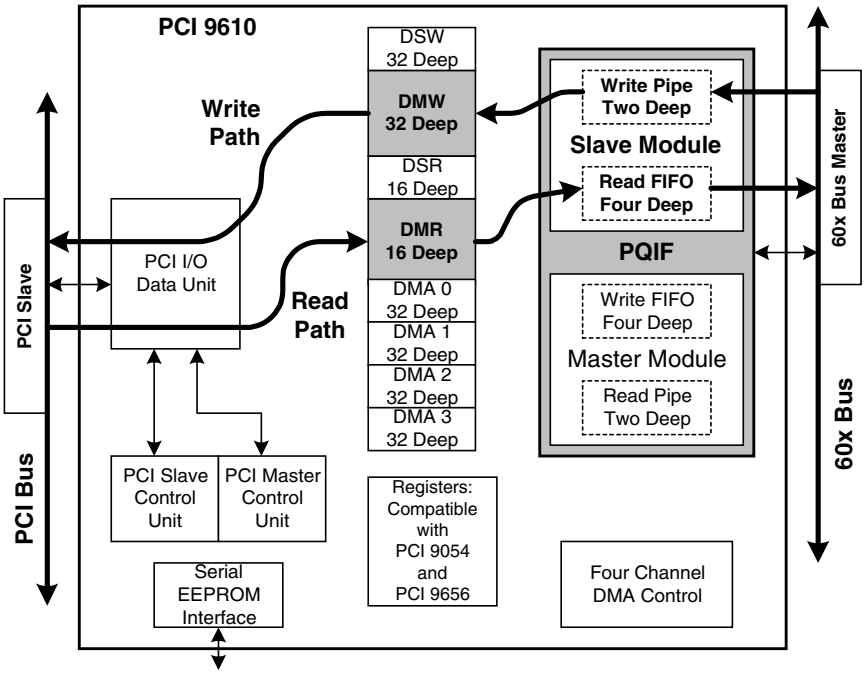
The Range register and the 60x Bus Base Address specifies the 60x Bus Address bits to use for decoding a 60x-to-PCI access (Direct Master). The range of Memory or I/O space must be a power of 2 and the Range register value must be the inverse of the range value. In addition, the 60x Bus Base Address must be a multiple of the range value.

Any 60x Bus Master Address starting from the Direct Master 60x Bus Base Address (Memory or I/O) to the range value is recognized as a Direct Master access by the PCI 9610. All Direct Master cycles are then decoded as PCI Memory, I/O, or Configuration Type 0 or 1. Moreover, a Direct Master Memory or I/O cycle is remapped according to the Remap register value. The Remap Register value must be a multiple of the Direct Master Range value (not the Range register value).

The PCI 9610 can only accept Memory cycles from a 60x Bus master. The 60x Bus Base Address and/or the range determine whether PCI Memory or PCI I/O transactions occur.

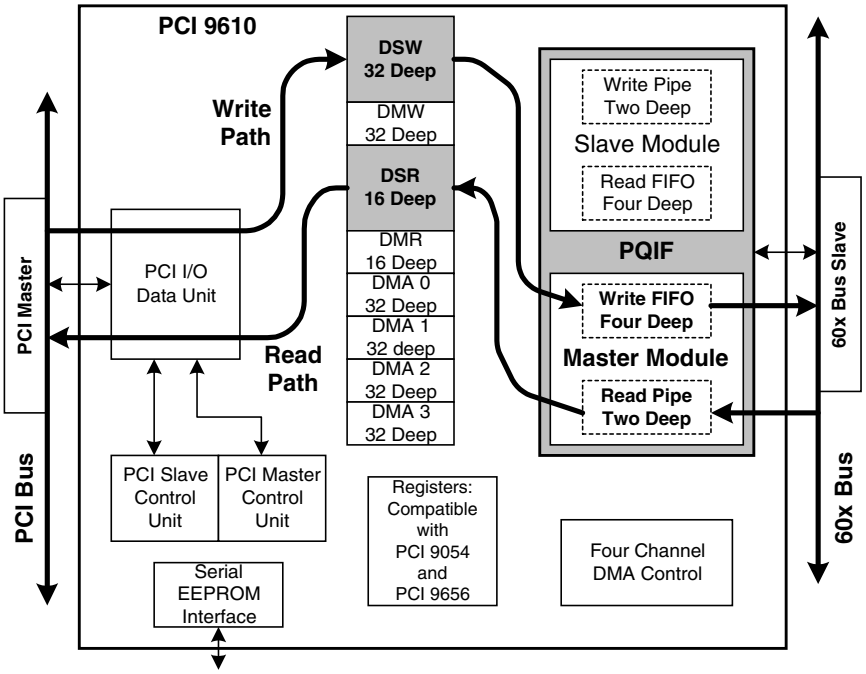
3.4.1.2 Direct Master FIFOs

For Direct Master Memory access to the PCI Bus, the PCI 9610 has a 32-Qword (256-byte) Write FIFO and a 16-Qword (128-byte) Read FIFO. The FIFOs enable the 60x Bus to operate independent of the PCI Bus and allows high-performance bursting on the PCI and 60x Buses. In a Direct Master write, the 60x Bus Master writes data to the PCI Bus (Slave). In a Direct Master read, the 60x Bus Master reads data from the PCI Bus (Slave). The FIFOs that function during a Direct Master write and read are illustrated in Figure 3-2 and Figure 3-3.



DMW = Direct Master Write; DMR = Direct Master Read;
DSW = Direct Slave Write; DSR = Direct Slave Read;
PQIF = PowerQUICC II Interface

Figure 3-2. Direct Master Write and Read



DMW = Direct Master Write; DMR = Direct Master Read;
DSW = Direct Slave Write; DSR = Direct Slave Read;
PQIF = PowerQUICC II Interface

Figure 3-3. Direct Slave Read and Write

3.4.1.3 Direct Master Memory Access

A Master on the Processor Bus accesses PCI Bus memory through the PCI 9610 using Direct Master Write and Direct Master Read data transfers.

Writes—Upon a 60x Bus write, the 60x Bus Master writes data to the Direct Master Write FIFO. When the first data reaches the FIFO, the PCI 9610 becomes the PCI Bus Master, arbitrates for the PCI Bus, and writes data to the PCI Slave device. The PCI 9610 continues to accept Address and Data tenure writes until the Write FIFO is full. It then holds off PSDVAL# until space becomes available in the Write FIFO. A programmable Direct Master "almost full" status output is provided (MDREQ#). The PCI 9610 asserts ARTRY# when the Direct Master Write FIFO is full, implying that the 60x Bus Master may finish the Write operation at a later time (LMISC1[6]).

Reads—During a Data tenure, the PCI 9610 holds off PSDVAL# while reading data from the PCI Bus. Programmable Prefetch modes are available if prefetch is enabled—prefetch, 4, 8, 16, or continuous—until the Direct Master cycle ends. Unused Read data are flushed from the FIFO.

The PCI 9610 does not prefetch Read PCI data for Single-Cycle Direct Master reads.

For Single-Cycle Direct Master reads, the PCI 9610 passes the corresponding PCI Bus byte enables from the 60x Bus address and TSIZ[0:3] signal.

For Burst-Cycle reads, the PCI 9610 reads entire Qwords (all PCI Bus byte enables are asserted), dependent upon the PCI and 60x Bus widths.

When the Direct Master Prefetch Limit bit is enabled (DMPBAM[11]=1), the PCI 9610 terminates a read prefetch at four-KB boundaries and restarts it as a new PCI Read Prefetch cycle at the start of a new boundary. If the bit is disabled, the prefetch crosses the four-KB boundaries.

When the 4-KB Prefetch Limit bit is enabled and the PCI 9610 has started a Direct Master read to a 64-bit PCI Bus, PCI Address 'hFF8 (Qword-aligned, one Qword before the 4-KB boundary) without ACK64# acknowledgment from the PCI Slave, the PCI 9610 does not perform a Burst prefetch of two Lwords. The PCI 9610 instead performs a prefetch of two Single-Cycle Lwords to prevent crossing the PCI 4-KB boundary limit. If the PCI Slave responds with ACK64#, the PCI 9610 performs a Single-Cycle read of one Qword and terminates to prevent crossing a 4-KB limit boundary. The cycle then restarts at the new boundary.

3.4.1.4 Direct Master I/O Configuration Access

When a 60x Bus Direct Master I/O access to the PCI Bus occurs, the PCI Configuration Address Register for Direct Master-to-PCI I/O Configuration Enable bit (DMCFGA[31]) determines whether an I/O or Configuration access is to be made to the PCI Bus.

60x Bus accesses to PCI I/O space and Configuration access must be equal to or less than an Lword (32 bits). If a 60x Bus Master attempts to write or read more than one Lword to PCI I/O or Configuration access, the PCI 9610 responds with a TEA# signal, aborting the access.

3.4.1.5 Direct Master I/O

When the Configuration Enable bit is cleared (DMCFGA[31]=0), a single I/O access is made to the PCI Bus. The 60x Bus Address, Remapped Decode Address bits, and 60x Bus byte enables are encoded to provide the address and are output with an I/O Read or Write command during a PCI Address cycle.

When the I/O Remap Select bit is set (DMPBAM[13]=1), the PCI Address bits [31:16] are forced to 0 for the 64-KB I/O address limit.

3.4.1.6 Direct Master Delayed Write Mode

The PCI 9610 supports Direct Master Delayed Write mode transactions, where posted Write data accumulates in the Direct Master Write FIFO before the PCI 9610 requests a PCI Bus. Direct Master Delayed Write mode is programmable to delay REQ# assertion in the amount of PCI clocks (DMPBAM[15:14]). This feature is useful for gaining higher throughput during Direct Master Write Burst transactions for conditions in which the 60x Bus clock frequency is slower than the PCI clock frequency.

The PCI 9610 only utilizes the delay counter and accumulates data in the Direct Master Write FIFO for

burst transactions on the 60x Bus. Otherwise, an immediate Single-Cycle PCI transfer occurs.

3.4.1.7 Direct Master Read Ahead Mode

The PCI 9610 also supports Direct Master Read Ahead mode (DMPBAM[2]), where prefetched data can be read from the internal FIFO of the PCI 9610 instead of from the 60x Bus. The address must be subsequent to the previous address and 32-bit aligned (next address = current address + 4) for 32-bit Direct Slave transfers and 64-bit aligned (next address = current address + 8) for 64-bit Direct Slave transfers. Read Ahead mode functions could be used with or without Delayed Read mode.

A 60x Bus Single-Cycle Direct Master transaction, with Read Ahead Mode (DMPBAM[2]) enabled results in the PCI 9610 processing continuous PCI Bus Read burst data with all bytes enabled (C/BE# = 'h0).

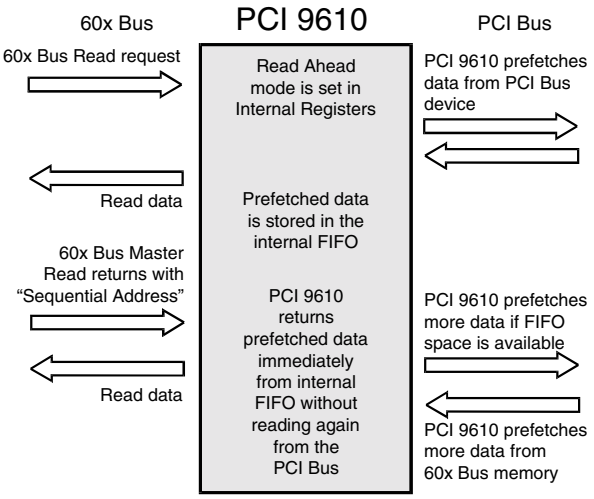


Figure 3-4. Direct Master Read Ahead Mode

Note: *Figure 3-4 represents a sequence of Bus cycles.*

3.4.1.8 ARTRY# Capability

3.4.1.8.1 Direct Master Write FIFO Full

The PCI 9610 supports the Direct Master Write FIFO full condition. When enabled (LMISC1[6]=1), the PCI 9610 asserts ARTRY# in response to the Address tenure 60x Bus Master to tell the Master to attempt the write at a later time.

In a Direct Master Write FIFO full condition, the PCI 9610 asserts the ARTRY# signal in response to the Address tenure. Otherwise, the Direct Master Write transfer goes through successfully.

3.4.1.8.2 Direct Master Delayed Read

The PCI 9610 supports Direct Master Delayed Read transactions. When the 60x Bus Direct Master Delayed Read Enable bit is set (LMISC1[4]=1), the PCI 9610 asserts ARTRY# (in response to an Address tenure from the 60x Bus Master) and prefetches Read data each time the 60x Bus Master requests a read. During a PCI data prefetch, the 60x Bus Master is capable of doing other transactions and free to return for requested data at a later time. When Delayed Direct Master Read mode is disabled, the 60x Bus Master must "keep" the 60x Bus and wait for the requested data (PSDVAL# is not asserted until data is available to the 60x Bus).

3.4.1.9 Direct Master Configuration (PCI Configuration Type 0 or Type 1 Cycles)

When the Configuration Enable bit is set (DMCFGA[31]=1), a Configuration access is made to the PCI Bus. In addition to enabling configuration of this bit, the user must provide all register information. The Register Number and Device Number bits (DMCFGA[7:2] and DMCFGA[15:11], respectively) must be modified and a new Configuration Read/Write cycle must be performed before accessing other registers or devices.

When the PCI Configuration Address register selects a Type 0 command, register bits [10:0] are copied to address bits [10:0]. Bits [15:11] (device number) are translated into a single bit being set in the PCI Address bits [31:11]. The PCI Address bits [31:11] can be used as a device select. For a Type 1 command, bits [23:0] are copied from the register to PCI address bits [23:0]. The PCI Address bits [31:24] are set to 0. A configuration Read or Write command code is output with the address during the PCI Address cycle. (Refer to the DMCFGA register.)

3.4.1.9.1 Direct Master Configuration Cycle Example

To perform a Type 0 Configuration cycle to PCI device on AD[21]:

1. The PCI 9610 must be configured to allow Direct Master access to the PCI Bus. The PCI 9610 must also be set to respond to I/O Space accesses. These bits must be set (PCICR[2:0]=111b).

   In addition, Direct Master memory and I/O access must be enabled (DMPBAM[1:0]=11).

2. The 60x Bus Memory map selects the Direct Master range. For this example, use a range of 1 MB:

   1 MB = 2^{20} = 000FFFFFh

3. The value to program into the Range register is the inverse of 000FFFFFh (FFF00000h):

   DMRR = FFF00000h

4. The 60x Bus Memory map determines the 60x Bus Base Address for the Direct Master-to-PCI I/O Configuration register. For this example, use 40000000h:

   DMLBAI = 40000000h

5. The PCI Address (Remap) for Direct Master-to-PCI Memory register must enable the Direct Master I/O access. The Direct Master I/O Access Enable bit must be set (DMPBAM[1]=1).

6. The user must know which PCI device and PCI Configuration register the PCI Configuration cycle is accessing. This example assumes the IDSEL signal of the Target PCI device is connected to AD[21] (logical device #10=0Ah).

Also access PCIBAR0 (the fourth register, counting from 0; use Table 11-2 for reference). Set DMCFGA[31, 23:0] as follows:

Bit	Description	Value
1:0	Configuration Type 0.	00b
7:2	Register Number. Fourth register. Must program a “4” into this value, beginning with bit 2.	000100b
10:8	Function Number.	000b
15:11	Device Number n-11, where n is the value in AD[n]=21-11 = 10.	01010b
23:16	Bus Number.	00000000b
31	Configuration Enable.	1

After these registers are configured, a simple 60x Bus Master Memory cycle to the I/O base address is necessary to generate a PCI Configuration Read or Write cycle. Offset to the base address is not necessary because the register offset for the read or write is specified in the Configuration register. The PCI 9610 takes the 60x Bus Master Memory cycle and checks for the Configuration Enable bit (DMCFGA[31]). If set, the PCI 9610 converts the current cycle to a PCI Configuration cycle, using the DMCFGA register and the Write/Read signal (RD/WR#).

The Register Number and Device Number bits (DMCFGA[7:2] and DMCFGA[15:11], respectively) must be modified and a new Configuration Read/Write cycle must be performed before accessing other registers or devices.

3.4.1.10 Direct Master PCI Dual Address Cycle

The PCI 9610 supports PCI Dual Address Cycle (DAC) when it is a PCI Bus Master using the DMDAC register for Direct Master transactions. The DAC command is used to transfer a 64-bit address to devices that support 64-bit addressing when the address is not in the low 4-GB address space. The PCI 9610 performs the address portion of a DAC in two PCI clock periods, where the first PCI address is a Lo-Addr with the command (C/BE[7:0]#) "D" and the second PCI address is Hi-Addr with the command (C/BE[7:0]#) "6" or "7", depending upon it being a PCI Read or Write cycle. Whenever the DMDAC register contains a value of 0x00000000, the PCI 9610 performs a Single Address Cycle (SAC) on the PCI Bus. (Refer to Figure 3-5 and Figure 3-6.)

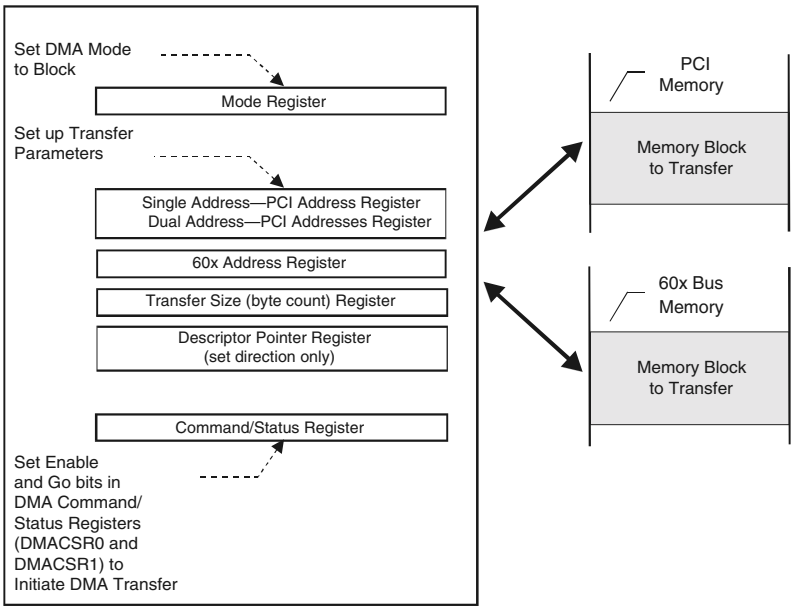


Figure 3-5. Block DMA Mode Initialization (Single or Dual Address PCI)

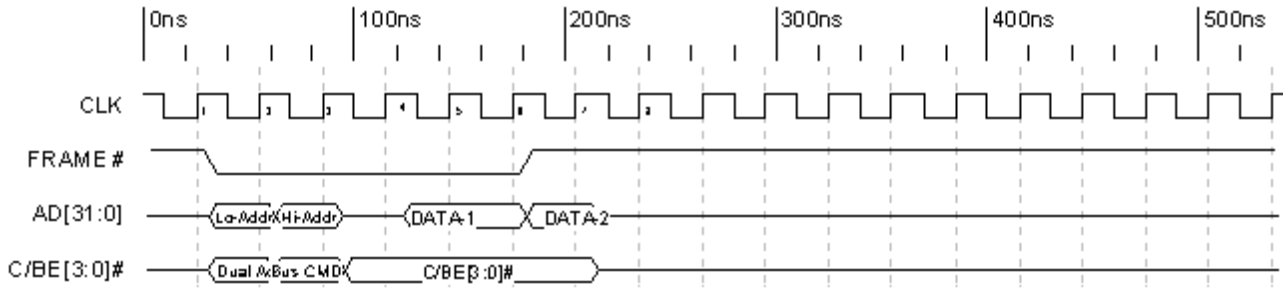


Figure 3-6. Dual Address Timing

3.4.1.11 PCI Master/Target Abort

The PCI 9610 PCI Master/Target Abort logic enables a 60x Bus Master to perform a Direct Master Bus poll of devices to determine whether devices exist (typically when the 60x Bus performs Configuration cycles to the PCI Bus). When a PCI Master device attempts to access and does not receive DEVSEL# within six PCI clocks, it results in a Master Abort. The 60x Bus Master must clear the Received Master Abort bit or Target

Abort bit (PCISR[13 or 11]=0, respectively) and continue by processing the next task.

When a PCI Master/Target Abort, or Retry Timeout is encountered during a transfer, the PCI 9610 may assert TEA# if enabled (INTCSR[0]=1). (Refer to Section 4.1.12.1, "TEA# Asserted by the PCI 9610," on page 4-5.)

When a 60x Bus Master is attempting a Burst read from a non-responsive PCI device, the PCI 9610 asserts LINTo#, if enabled, and may assert TEA#. (Refer to Section 4.1.3, "60x Bus Interrupt Output (LINTo#)," on page 4-2 and Section 4.1.12.1, "TEA# Asserted by the PCI 9610," on page 4-5.)

When a PCI Master/Target abort is encountered during a Direct Master transfer, the PCI 9610 stores the PCI Abort address into the PCI Abort Address register bits (PABTADR[31:0]).

3.4.1.12 Direct Master Memory Write and Invalidate

The PCI 9610 can be programmed to perform Memory Write and Invalidate cycles to the PCI Bus for Direct Master transfers, as well as DMA transfers. (Refer to Section 3.5.4.) The PCI 9610 supports Memory Write and Invalidate transfers for cache line sizes of 8 or 16 Lwords. Size is specified in the System Cache Line Size bits (PCICLSR[7:0]). If a size other than 8 or 16 is specified, the PCI 9610 performs Write transfers rather than Memory Write and Invalidate transfers.

Direct Master Memory Write and Invalidate transfers are enabled when the Invalidate Enable and the Memory Write and Invalidate Enable bits are set (DMPBAM[9]) and (PCICR[4], respectively).

In Memory Write and Invalidate mode, if the start address of the Direct Master transfer is on a cache line boundary, the PCI 9610 waits until the number of Lwords required for the specified cache line size are written from the 60x Bus before starting a PCI Memory Write and Invalidate access. This ensures a complete cache line write can complete in one PCI Bus ownership.

When the start address is not on a cache line boundary, the PCI 9610 starts a normal PCI Write access (PCI command code = 7h). The PCI 9610 does not terminate a normal PCI Write at an MWI cache boundary. The normal PCI Write transfer continues until the Data transfer is complete. If a Target disconnects before a cache line completes, the PCI 9610 completes the remainder of that cache line, using normal writes.

3.4.2 Direct Slave Operation (PCI Master-to-60x Bus Access)

The PCI 9610 supports Burst Memory-Mapped Transfer accesses and I/O-Mapped, Single-Transfer PCI-to-60x Bus accesses through a 32-Lword/16-Qword (128-byte) Direct Slave Read FIFO and a 64-Lword/32-Qword (256-byte) Direct Slave Write FIFO. The PCI Base Address registers are provided to set up the location of the adapter in the PCI memory and the I/O space. In addition, 60x Bus mapping registers allow address translation from the PCI Address Space to the 60x Bus Address Space. Three spaces are available:

- Space 0
- Space 1
- Expansion ROM

Expansion ROM is intended to support a bootable ROM device for the Host.

Writes—Upon a PCI Bus write, the PCI Bus Master writes data to the Direct Slave Write FIFO. When the first data is in the FIFO, the PCI 9610 asserts BR# to request mastership of the 60x Bus. The PCI 9610 continues to accept writes and returns TRDY# until the Write FIFO is full. It then holds off TRDY# until space becomes available in the Write FIFO or asserts STOP#, and Retries the PCI Bus Master, dependent upon the register bit setting (LBRD0[27]).

A 32-bit PCI Bus Master Single-Cycle Write transaction results in PCI 9610 transfers of one Lword of data onto a 60x Bus. A 64-bit PCI Bus Master Qword Data Single-Cycle write results in PCI 9610 Single-Cycle transfers of one Qword onto a 60x Bus.

Reads—The PCI 9610 holds off TRDY# while gathering data from the 60x Bus, unless the Delayed Read Mode bit is enabled (MARBR[24]=1). (Refer to Section 3.4.2.2.) Programmable Prefetch modes are available, if prefetch is enabled—prefetch, 0-16, or continuous—until the Direct Slave read ends. The Read cycles are terminated on the following clock after

FRAME# is de-asserted or the PCI 9610 issues a Retry or disconnect.

For the highest data-transfer rate, the PCI 9610 supports posted writes and can be programmed to prefetch data during a PCI Burst read. The Prefetch size, when enabled, can be from one to 16 Lwords or until the PCI Bus stops requesting. When the PCI 9610 prefetches, if enabled, it drops the 60x Bus after reaching the prefetch counter limit. In Continuous Prefetch mode, the PCI 9610 prefetches as long as FIFO space is available, and stops prefetching when the PCI Bus terminates the request. If Read prefetching is disabled, the PCI 9610 disconnects after one Read transfer.

The PCI 9610 64-bit PCI Bus Direct Slave unaligned Qword Data Prefetch Read transfers are special cases that result in prefetching one more Lword (32-bit) of 60x Bus data than specified in the prefetch counter (LBRD0[14:11] and/or LBRD1[14:11]), to sustain zero wait state 64-bit PCI Data transfers. For 64-bit Qword-aligned and 32-bit PCI Bus Direct Slave Prefetch Read transfers, the PCI 9610 prefetches the amount specified in the prefetch counter.

In addition to Prefetch mode, the PCI 9610 supports Read Ahead mode. (Refer to Section 3.4.2.3.)

A 64-bit PCI Bus Single-Cycle Direct Slave read results in a 60x Bus Single-Cycle Qword transfer, with all PCI bytes (C/BE# = 'h0) asserted.

Each 60x Bus space can be programmed to operate in an 8-, 16-, 32-, or 64-bit 60x Bus width.

With or without wait state(s), the 60x Bus, independent of the PCI Bus, can perform the following:

- Burst as long as data is available using the PLX Extended Burst option)
- Burst four Qwords at a time
- Perform a continuous Single cycle

3.4.2.1 Direct Slave Lock

The PCI 9610 supports direct PCI-to-60x exclusive accesses (locked atomic operations). A PCI-locked operation to the 60x Bus results in the entire address Space 0, Space 1, and Expansion ROM space being locked until they are released by the PCI Bus Master. Locked operations are enabled or disabled with the Direct Slave LOCK# Enable bit (MARBR[22]) for PCI-to-60x Bus accesses.

3.4.2.2 Direct Slave Delayed Read Mode

The PCI 9610 can be programmed through the PCI Specification r2.1 Mode bit (MARBR[24]=1) to perform delayed reads, as specified in *PCI Specification r2.1*.

PCI Bus Single-Cycle aligned or unaligned 32-bit Direct Slave Delayed Read transactions always result in 1-Lword Single-Cycle transfers on the 60x Bus with the corresponding 60x Bus Address and TSIZ[0:1] asserted to reflect the PCI byte enables (C/BE#), unless the PCI Read No Flush Mode bit is enabled (MARBR[28]=1). (Refer to Section 3.4.2.3 for further information.) This causes the PCI 9610 to Retry all PCI Bus Read requests that follow, until the original PCI byte enables (C/BE#) are matched.

In addition to delayed reads, the PCI 9610 supports the following *PCI Specification r2.1* functions:

- No writes while a read is pending (PCI Retry for writes)
- Write and flush pending read

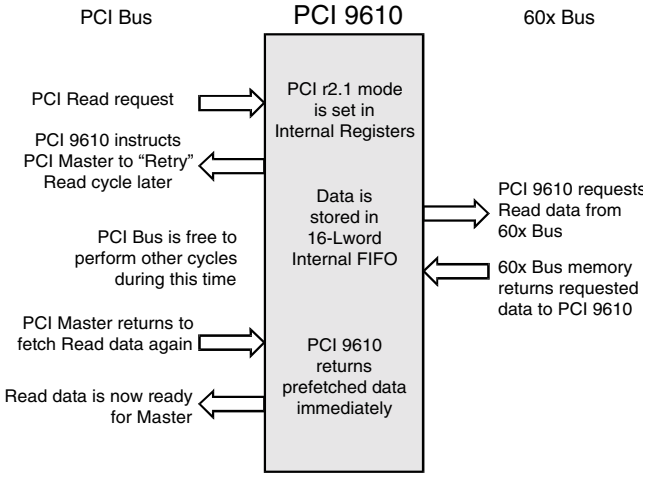


Figure 3-7. Direct Slave Delayed Reads

Note: *Figure 3-7 represents a sequence of Bus cycles.*

3.4.2.3 Direct Slave Read Ahead Mode

The PCI 9610 also supports Direct Slave Read Ahead mode (MARBR[28]), where prefetched data can be read from the internal FIFO of the PCI 9610 instead of from the 60x Bus. The address must be subsequent to the previous address and 32-bit aligned (next address = current address + 4) for 32-bit Direct Slave transfers and 64-bit aligned (next address = current address + 8) for 64-bit Direct Slave transfers. Read Ahead mode functions with or without Delayed Read mode.

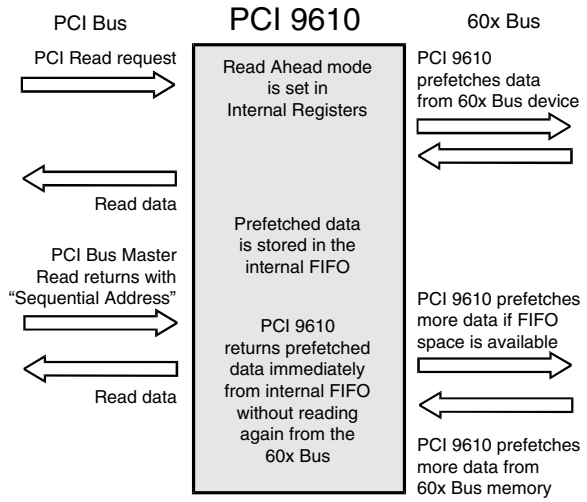


Figure 3-8. Direct Slave Read Ahead Mode

Note: *Figure 3-8 represents a sequence of Bus cycles.*

3.4.2.4 Direct Slave Delayed Write Mode

The PCI 9610 supports Direct Slave Delayed Write mode transactions, where posted Write data accumulates in the Direct Slave Write FIFO before the PCI 9610 as a 60x Bus master, initiates a Write transaction to be performed on the 60x Bus. The Direct Slave Delayed Write mode is programmable to delay the BR# assertion in the amount of LCLK clocks (LMISC2[4:2]). This feature is useful for gaining higher throughput during Direct Slave Write Burst transactions for conditions in which the PCLK frequency is slower than the LCLK frequency.

3.4.2.5 Direct Slave Transfer

A PCI Bus Master addressing the Memory space decoded for the 60x Bus initiates transactions. Upon a PCI read/write, the PCI 9610 becomes a 60x Bus master and arbitrates for the 60x Bus.

The PCI 9610 then reads data into the Direct Slave Read FIFO or writes data to the 60x Bus.

The Direct Slave or Direct Master preempts DMA; however, the Direct Slave does not preempt the Direct Master. (Refer to Section 3.4.3.1.)

The PCI 9610 can be programmed to "keep" the PCI Bus by generating a wait state(s) and de-asserting TRDY#, if the Write FIFO becomes full. In the PLX Extended Burst option, the PCI 9610 can be programmed to assert DBB# a specific number of LCLK cycles, as set in MARBR[7:0].

For Direct Slave writes, the PCI Bus writes data to the 60x Bus. Direct Slave is the "Command from the PCI Master," which has the highest priority.

For Direct Slave reads, the PCI Bus Master reads data from the 60x Bus Slave.

The PCI 9610 supports on-the-fly Endian conversion for all data transfers—Space 0, Space 1, Expansion ROM space, and all four DMA channels. The 60x Bus can be Big/Little Endian (Address/Data Invariance) by using the programmable internal register configuration.

Note: *The PCI Bus is always Little Endian.*

3.4.2.6 Direct Slave PCI-to-60x Bus Address Mapping

Note: *Not applicable in I_2O mode.*

Three 60x Bus Address spaces—Space 0, Space 1, and Expansion ROM—are accessible from the PCI Bus. Each is defined by a set of three registers:

- 60x Bus Address Range (LAS0RR, LAS1RR, and/or EROMRR)
- 60x Bus Base Address (LAS0BA, LAS1BA, and/or EROMBA)
- PCI Base Address (PCIBAR2, PCIBAR3, and/or PCIERBAR)

A fourth register, the Bus Region Descriptor register(s) for PCI-to-60x Bus Accesses (LBRD0 and/or LBRD1), defines the 60x Bus characteristics for the Direct Slave regions. (Refer to Figure 3-9.)

Each PCI-to-60x Bus Address space is defined as part of reset initialization, as described in Section 3.4.2.6.1. These 60x Bus characteristics can be modified at any time before actual data transactions.

3.4.2.6.1 Direct Slave 60x Bus Initialization

Range—Specifies which PCI Address bits to use for decoding a PCI access to 60x Bus space. Each bit corresponds to a PCI Address bit. Bit 31 corresponds to address bit 31. Write 1 to all bits that must be included in decode and 0 to all others.

Remap PCI-to-60x Bus Addresses into a 60x Bus Address Space—Bits in this register remap (replace) the PCI Address bits used in decode as the 60x Bus Address bits.

60x Bus Region Descriptor—Specifies the 60x Bus characteristics.

3.4.2.6.2 Direct Slave PCI Initialization

After a PCI reset, the software determines how much address space is required by writing all ones (1) to a PCI Base Address register and then reading back the value. The PCI 9610 returns zeroes (0) in the Don't Care Address bits, effectively specifying the address space required. The PCI software then maps the 60x Bus Address space into the PCI Address space by programming the PCI Base Address register. (Refer to Figure 3-9.)

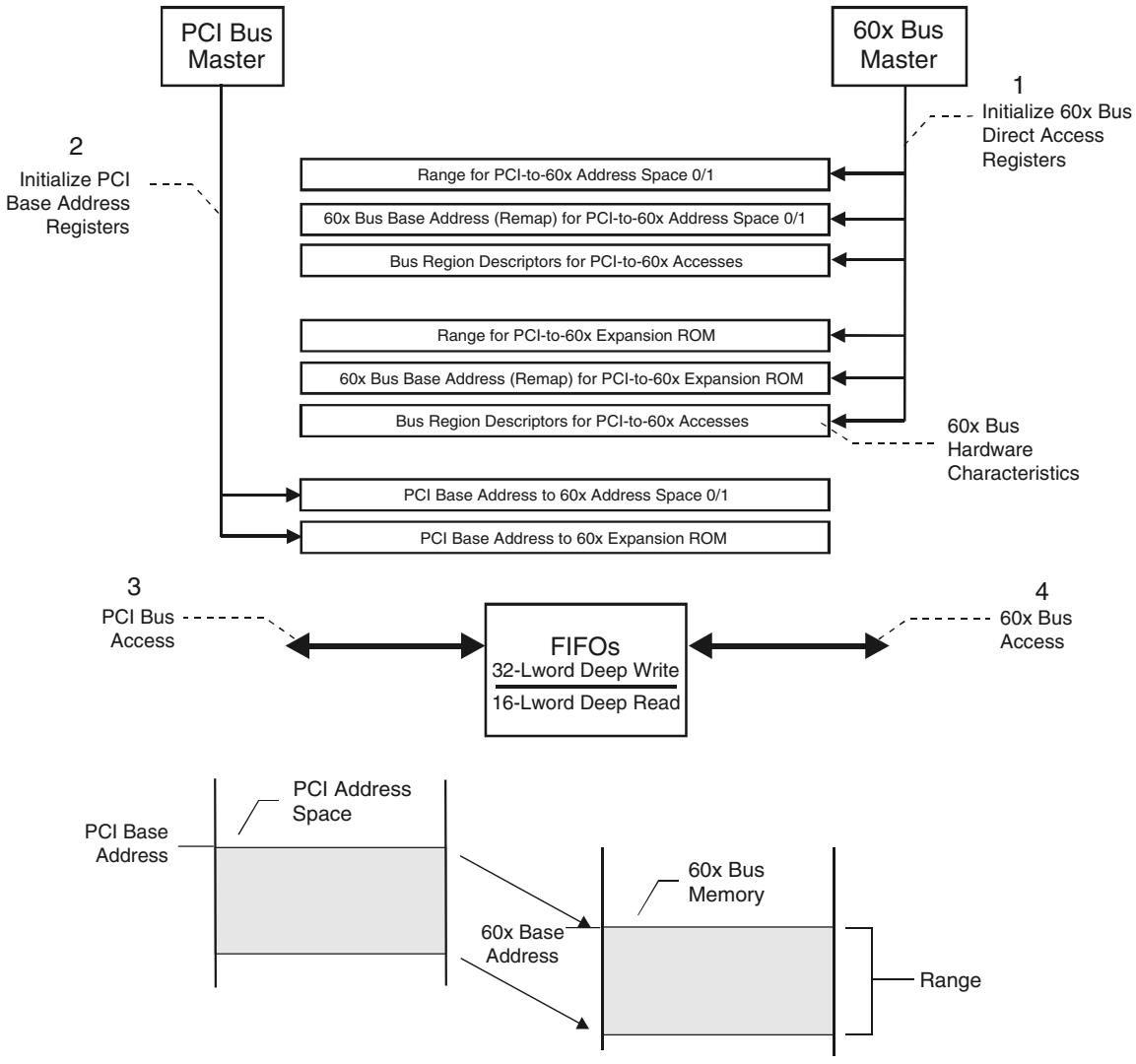


Figure 3-9. 60x Bus Direct Slave Access

3.4.3 Deadlock Conditions

Deadlock can occur when a PCI Bus master must access the PCI 9610 60x Bus at the same time a master on the PCI 9610 60x Bus must access the PCI Bus. There are two types of deadlock:

- **Partial Deadlock**—A 60x Bus master is performing a Direct Bus Master access to a PCI Bus device other than the PCI Bus device concurrently trying to access the 60x Bus
- **Full Deadlock**—A 60x Bus master is performing a Direct Bus Master access to the same PCI Bus device concurrently trying to access the 60x Bus

This applies only to Direct Master and Direct Slave accesses through the PCI 9610. Deadlock does not occur in transfers through the PCI 9610 DMA channels or the PCI 9610 internal registers (*such as*, mailboxes).

For partial deadlock, the PCI access to the 60x Bus times out [the Direct Slave Retry Delay Clock (LBRD0[31:28]), which is programmable through the 60x Bus Region Descriptor register] and the PCI 9610 responds with a PCI Retry. The PCI Specification requires that a PCI Master release its request for the PCI Bus (de-assert REQ#) for a minimum of two PCI clocks after receiving a Retry. This allows the PCI Bus arbiter to grant the PCI Bus to the PCI 9610 so that it can complete its Direct Master access and free up the 60x Bus. Possible solutions are described in the following sections for cases in which the PCI Bus arbiter does not function as described (PCI Bus architecture dependent), waiting for a time out is undesirable, or a full deadlock condition exists.

3.4.3.1 Backoff

In 60x Bus operation (not using the PLX Extended Burst), there is no Full Deadlock, because the 60x Bus protocol limits the Data tenures to a four-beat burst.

In the PLX Extended Burst option, a deadlock could occur with a Direct Master transfer where the 60x Bus Master is “hogging” the 60x Bus, and delays or fails to assert a Master terminate signal (PLX_MT#). In this case, the PCI 9610 asserts a slave termination after 64 LCLK clocks if enabled (EROMBA[0]=1).

3.4.3.1.1 Software/Hardware Solution for Systems without Backoff Capability

For adapters that do not support backoff, a possible deadlock solution is as follows.

PCI Host software, external 60x Bus hardware, general-purpose output USERo and general-purpose input USERi can be used to prevent deadlock. USERo can be asserted to request that the external arbiter not grant the bus to any 60x Bus Master except the PCI 9610. Status output from the 60x Bus arbiter can be connected to the general purpose input USERi to indicate that no 60x Bus Master owns the 60x Bus, or the PCI Host to determine that no 60x Bus Master that currently owns the 60x Bus can read input. The PCI Host can then perform Direct Slave access. When the Host finishes, it de-asserts USERo.

3.4.3.1.2 Preempt Solution

For devices that support preempt, USERo can be used to preempt the current 60x Bus Master device. When USERo is asserted, the current 60x Bus Master device completes its current cycle and releases the 60x Bus, de-asserting BB#.

3.4.3.2 Software Solutions to Deadlock

Both PCI Host and 60x Bus software can use a combination of mailbox registers, doorbell registers, interrupts, and direct 60x-to-PCI Bus and PCI-to-60x Bus accesses to avoid deadlock.

3.5 DMA OPERATION

The PCI 9610 supports four independent DMA channels capable of transferring data from:

- 60x-to-PCI Bus
- PCI-to-60x Bus

Note: *In this data book, we shall refer to DMA Channel x, where x = 0, 1, 2, or 3 for the channel number. For example, the register DMAMODEx refers to the DMAMODE register for any of the four DMA channels.*

Each channel consists of a DMA controller and a dedicated, bidirectional FIFO. Both channels support Block and Scatter/Gather transfers, with or without End of Transfer (EOTx#), with or without Demand Mode. Master mode must be enabled with the Master

Enable bit (PCICR[2]) before the PCI 9610 can become a PCI Bus Master. In addition, DMA channels can be programmed to:

- Operate in 8-, 16-, 32, or 64-bit 60x Bus width
- Enable/disable PLX Extended Burst
- Hold 60x Bus address constant (60x Bus Slave is FIFO) or increment
- Perform PCI Memory Write and Invalidate (command code = Fh) or normal PCI Memory Write (command code = 7h)
- Assert PCI interrupt (INTA#) or 60x Bus interrupt (LINTo#) when DMA transfer is complete or Terminal Count is reached during Scatter/Gather DMA mode transfers
- Operate in DMA Clear Count mode (only if the descriptor is in 60x Bus memory)

The PCI 9610 also supports PCI Dual Address with an upper 32-bit register (DMADACx).

In PLX Extended Burst mode, the 60x Bus Latency Timer (MARBR[7:0]) determines the number of LCLK clocks the PCI 9610 can burst before relinquishing the 60x Bus. The 60x Bus Pause Timer (MARBR[51:8]), used only in PLX Extended Burst mode, sets the minimum pause time between DMA Data bursts.

3.5.1 DMA PCI Dual Address Cycle

The PCI 9610 supports PCI Dual Address Cycles (DAC) when it is a PCI Bus Master using the DMADACx register for Block DMA transactions. Scatter/Gather DMA can utilize the DAC function by way of the DMADACx or DMAMODEx[18] register. The DAC command is used to transfer a 64-bit address to devices that support 64-bit addressing when the address is above the 4-GB address space. The PCI 9610 performs a DAC within two PCI clock periods, when the first PCI address is a Lo-Addr, with the command (C/BE[7:0]#) “D”, and the second PCI address is a Hi-Addr, with the command (C/BE[7:0]#) “6” or “7”, depending upon whether it is a PCI Read or PCI Write cycle.

3.5.2 Block DMA Mode

A PCI Master or the 60x Bus Master sets the 60x Bus and PCI starting addresses, transfer byte count, and transfer direction. The PCI or 60x Bus Master then sets the DMA Start bit to initiate a transfer. The PCI 9610 requests the PCI and 60x Buses and transfers data. Once the transfer completes, the PCI 9610 sets the Channel Done bit (DMACSRx[4]=1) and, if enabled, asserts an interrupt(s) (DMAMODEx [10]) to the 60x Bus Master or the PCI Master (programmable). The Channel Done bit(s) can be polled, instead of interrupt generation, to indicate the DMA transfer status.

DMA registers are accessible from the PCI and 60x Buses. (Refer to Figure 3-5 on page 3-8.)

During DMA transfers, the PCI 9610 is a Master on the PCI and 60x Buses. For simultaneous access, Direct Slave or Direct Master has a higher priority than DMA.

The PCI 9610 releases the PCI Bus, if one of the following conditions occur (refer to Figure 3-11 and Figure 3-12 on page 3-16):

- FIFO is full (PCI-to-60x Bus)
- FIFO is empty (60x-to-PCI Bus)
- Terminal count is reached
- PCI Bus Latency Timer expires (PCILTR[7:0])—normally programmed by the Host PCI BIOS—and PCI GNT# de-asserts
- PCI Host asserts STOP#

There is no 60x Bus release in the 60x Bus mode (including Strict MPC603e mode). When using PLX Extended Burst, the PCI 9610 terminates the 60x Bus Data tenure, if one of the following conditions occurs:

- FIFO is empty (PCI-to-60x Bus)
- FIFO is full (60x-to-PCI Bus)
- Terminal count is reached
- 60x Bus Latency Timer is enabled and expires (MARBR[7:0])
- Direct Slave request is pending

3.5.2.1 Block DMA PCI Dual Address Cycle

The PCI 9610 supports the DAC feature in Block DMA mode. Whenever the DMADACx register contains a value of 0x00000000, the PCI 9610 performs a Single Address Cycle (SAC) on the PCI Bus. Any other value causes a Dual Address to appear on the PCI Bus. (Refer to Figure 3-10 on page 3-15.)

3.5.3 Scatter/Gather DMA Mode

In Scatter/Gather DMA mode, the PCI or 60x Bus Master sets up descriptor blocks in 60x Bus or PCI memory composed of PCI and 60x Bus addresses, transfer count, transfer direction, and address of next descriptor block. (Refer to Figure 3-11 and Figure 3-12.) The PCI or 60x Bus Master then:

- Enables the Scatter/Gather mode bit (DMAMODEx[9]=1)
- Sets up the address of initial descriptor block in the PCI 9610 Descriptor Pointer register (DMADPRx)
- Initiates the transfer by setting a control bit (DMACSRx[1:0])

The PCI 9610 DMA channel supports multiple beat bursts, up to the limit of a four-beat burst, when the bit DMAMODEx[8]=1.

The PCI 9610 loads the first descriptor block and initiates the Data transfer. The PCI 9610 continues to load descriptor blocks and transfer data until it detects the End of Chain bit is set (DMADPRx[1]=1) (this bit is part of each descriptor). When the End of Chain bit is detected, the PCI 9610 completes the current descriptor block and sets the DMA Done bit (DMACSRx[4]). If the End of Chain bit is detected, the PCI 9610 asserts a PCI interrupt (INTA#) and/or 60x Bus interrupt (LINTo#).

The PCI 9610 can also be programmed to assert PCI or 60x Bus interrupts after each descriptor is loaded, then finish transferring.

If Scatter/Gather descriptors are in 60x Bus memory, the DMA controller can be programmed to clear the transfer size at completion of each DMA, using the DMA Clear Count Mode bit (DMAMODEx[16]).

Notes: *In Scatter/Gather DMA mode, the descriptor includes the PCI and 60x Bus Address Space, transfer size, and next descriptor pointer. It also includes a DAC value, if the DAC Chain Load bit is enabled (DMAMODEx[18]=1). Otherwise, the register (DMADACx) value is used.*

The Descriptor Pointer register (DMADPRx) contains end of chain (bit 1), direction of transfer (bit 3), next descriptor address (bits [31:4]), interrupt after terminal count (bit 2), and descriptor location (bit 0) bits.

The 60x Bus width must be the same as 60x Bus Memory Bus width.

A DMA descriptor can be on the 60x Bus memory or the PCI memory, or both (for example, one descriptor on 60x Bus memory, another descriptor on PCI memory and vice-versa).

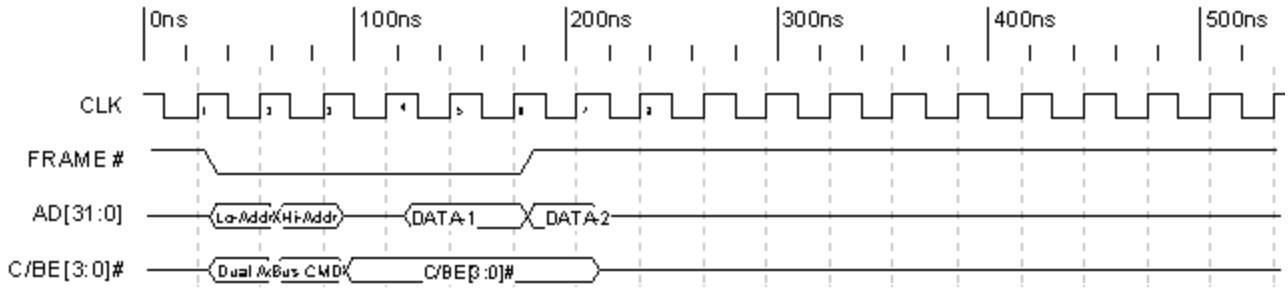


Figure 3-10. Dual Address Timing

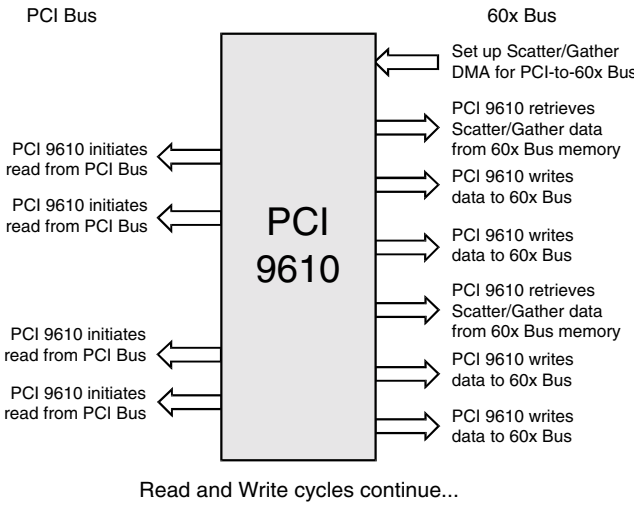


Figure 3-11. Scatter/Gather DMA Mode from PCI-to-60x Bus (Control Access from the 60x Bus)

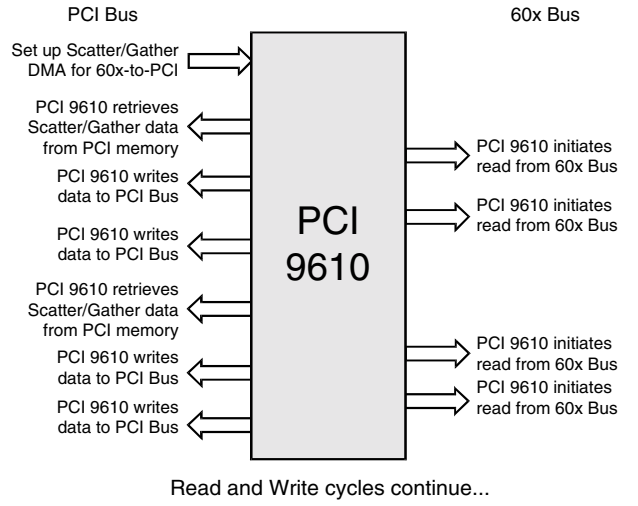


Figure 3-12. Scatter/Gather DMA Mode from 60x-to-PCI Bus (Control Access from the PCI Bus)

Note: *Figures 3-11 and 3-12 represent a sequence of Bus cycles.*

3.5.3.1 Scatter/Gather DMA PCI Dual Address Cycle

The PCI 9610 supports the DAC feature in Scatter/Gather DMA mode for Data transfers only. The descriptor blocks should reside below the 4-GB Address space.

The PCI 9610 offers three PCI DAC Scatter/Gather DMA utilization options. Assuming the descriptor blocks are located on the PCI Bus:

- DMADACx contains a non-zero value. DMAMODEx[18] is set to 0. The PCI 9610 performs a Single Address Cycle (SAC) four-Lword descriptor block load from PCI memory and DMA transfer with DAC on the PCI Bus. (Refer to Figure 3-13.)
- DMADACx contains an 0x00000000 value. DMAMODEx[18] is set to 1. The PCI 9610 performs a SAC five-Lword descriptor block load from PCI memory and DMA transfer with DAC on the PCI Bus. (Refer to Figure 3-14.)
- DMADACx contains a non-zero value. DMAMODEx[18] is set to 1. The PCI 9610 performs a SAC five-Lword descriptor block load from PCI memory and DMA transfer with DAC on the PCI Bus. The fifth descriptor overwrites the value of the DMADACx registers. (Refer to Figure 3-14.)

3.5.3.2 DMA Clear Count Mode

The PCI 9610 supports DMA Clear Count mode (Write-Back feature, DMAMODEx[16]). This feature allows users to control the Data transfer blocks during Scatter/Gather DMA operations. The PCI 9610 clears the Transfer Size descriptor to zero by writing to a descriptor-memory location at the end of each transfer chain. This feature is available for DMA descriptors located on the 60x and PCI Buses.

3.5.3.3 DMA Descriptor Ring Management (Valid Mode)

In Scatter/Gather DMA mode, when the Valid Mode Enable bit(s) is set to 0 (DMAMODEx[20]=0), the Valid bit (bit 31 of transfer count) is ignored. When the Valid Mode Enable bit is set to 1 (DMAMODEx[20]=1), the DMA descriptor proceeds only when the Valid bit is set. If the Valid bit is set, the transfer count is 0, and the descriptor is not the last descriptor, then the DMA controller moves on to the next descriptor in the chain.

When the Valid Stop Control bit is set to 0 (DMAMODEx[21]=0), the DMA Scatter/Gather controller continuously polls the descriptor with the Valid bit set to 0 (invalid descriptor) until the Valid bit is read to be a 1. When the Valid Stop Control bit is set to 1 (DMAMODEx[21]=1), the DMA Scatter/Gather controller pauses if a Valid bit with a value of 0 is detected. In this case, the PCI 9610 must restart the DMA controller by setting bit 1 of the DMA Control/Status register (DMACSRx[1]). The DMA Clear Count mode bit (DMAMODEx[16]) must be enabled for the Ring Management Valid bit to be cleared at the completion of each descriptor.

3.5.4 DMA Memory Write and Invalidate

The PCI 9610 can be programmed to perform Memory Write and Invalidate cycles to the PCI Bus for DMA transfers, as well as Direct Master transfers. (Refer to Section 3.4.1.12.) The PCI 9610 supports Memory Write and Invalidate transfers for cache line sizes of 8 or 16 Lwords. Size is specified in the System Cache Line Size bits (PCICLSR[7:0]). If a size other than 8 or 16 is specified, the PCI 9610 performs Write transfers rather than Memory Write and Invalidate transfers.

DMA Memory Write and Invalidate transfers are enabled when the DMA controller Memory Write and Invalidate Enable bit (DMAMODEx[13]) and the Memory Write and Invalidate Enable bit (PCICR[4]) are set.

In Memory Write and Invalidate mode, the PCI 9610 waits until the number of Lwords required for specified cache line size are read from the 60x Bus before starting the PCI access. This ensures a complete cache line write can complete in one PCI Bus ownership. If a target disconnects before a cache line completes, the PCI 9610 completes the remainder of that cache line, using normal writes before resuming Memory Write and Invalidate transfers. If a Memory Write and Invalidate cycle is in progress, the PCI 9610 continues to burst if another cache line is read from the 60x Bus before the cycle completes. Otherwise, the PCI 9610 terminates the burst and waits for the next cache line to be read from the 60x Bus. If the final transfer is not a complete cache line, the PCI 9610 completes the DMA transfer, using normal writes.

EOT# signal assertion, in any DMA transfer type, or DREQ0# and/or DREQ1# signal de-assertion in Demand Mode before the cache line is read from the 60x Bus, results in the PCI 9610 performing a normal PCI Memory Write to data read into a DMA FIFO.

3.5.4.1 DMA Abort

DMA transfers can be aborted, in addition to the EOT# signal, as follows:

1. Clear the DMA Channel Enable bit (DMACSRx[0]=0).
2. Abort DMA by setting the Channel Abort bit (DMACSRx[2]=1).
3. Wait until the Channel Done bit is set (DMACSRx[4]=1).

Note: *One to two Data transfers occur after the Abort bit is set. Aborting when no DMA cycles are in progress causes the next DMA to abort.*

3.5.5 DMA Priority

The DMA Channel Priority bits (LMISC2[6] and MARBR[20:19]) are used to specify the priorities listed in the following table. The priorities are assigned in descending order.

Table 3-2. DMA Channel Priority Bit Specifications

LMISC2[6]	MARBR[20:19]	Channel Priority
0	Must be 00	Rotating
1	00	Channel 0, 1, 2, 3
1	01	Channel 1, 2, 3, 0
1	10	Channel 2, 3, 0, 1
1	11	Channel 3, 0, 1, 2

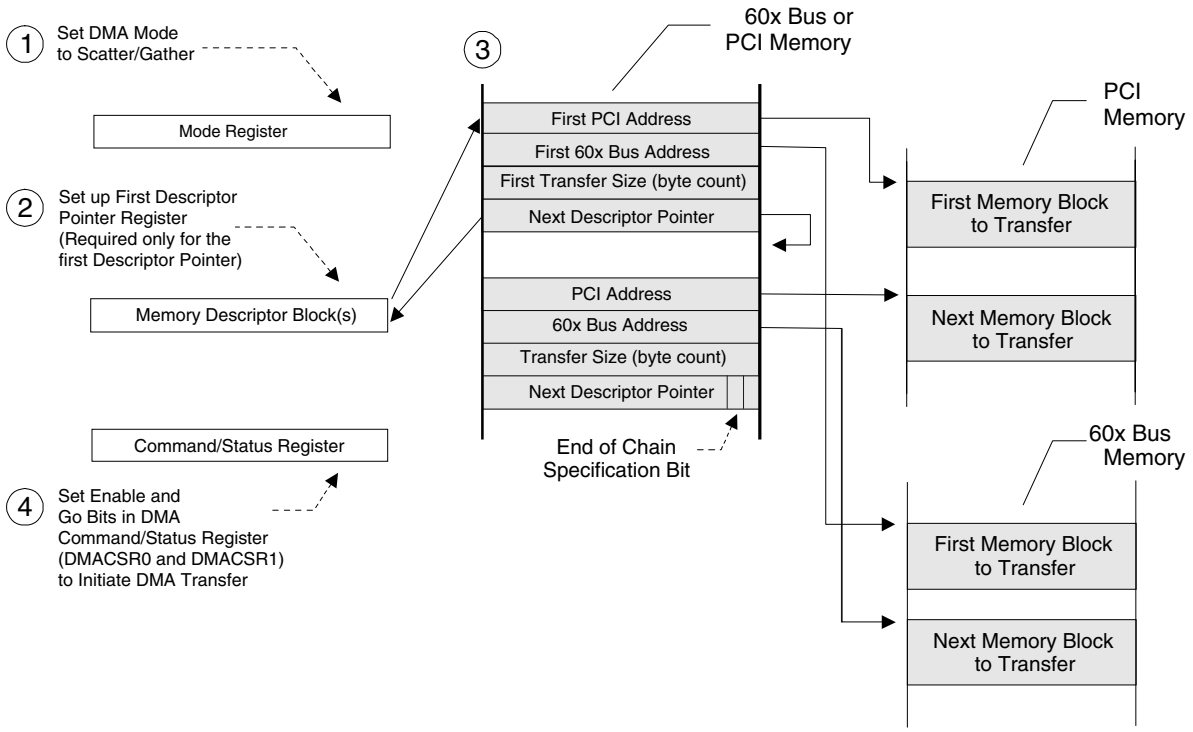


Figure 3-13. Scatter/Gather DMA Mode Descriptor Initialization [PCI SAC/DAC PCI Address (DMADACx) Register Dependent]

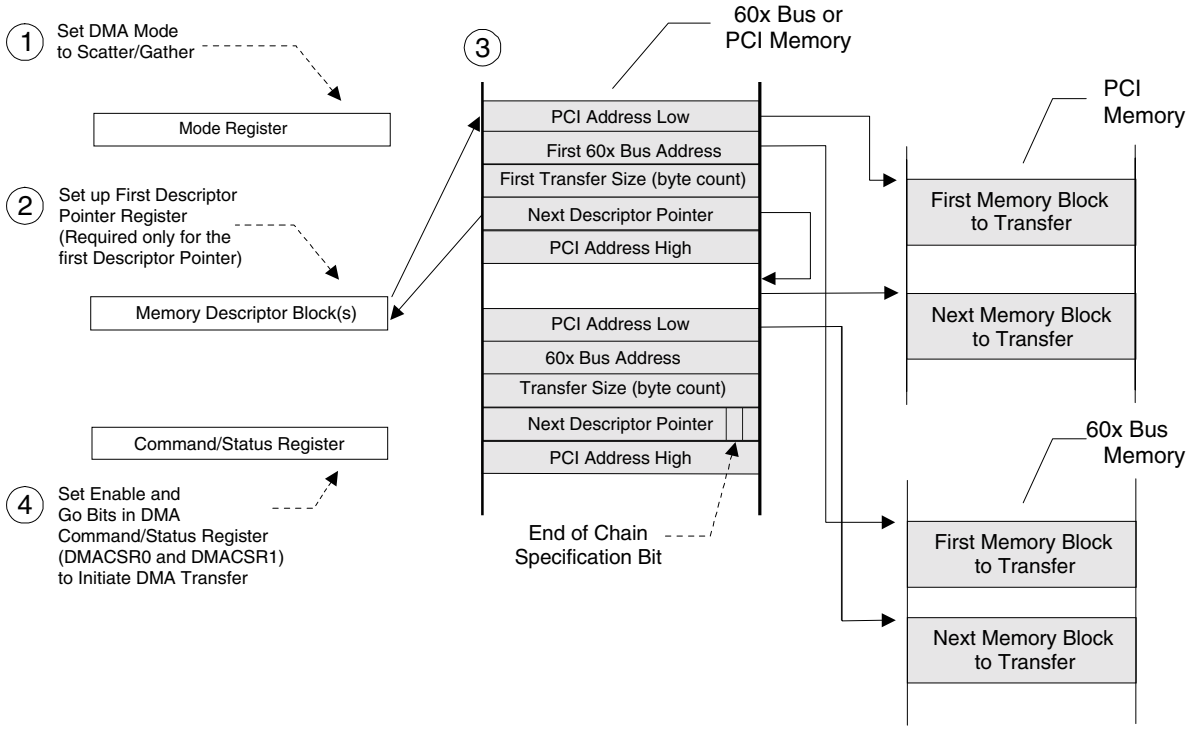


Figure 3-14. Scatter/Gather DMA Mode Descriptor Initialization [DAC PCI Address (DMAMODEx[18]) Descriptor Dependent] (PCI Address High Added)

3.5.6 DMA Channel x Interrupts

A DMA channel can assert a PCI or 60x Bus interrupt when done (transfer complete) or after a transfer is complete for the current descriptor in Scatter/Gather DMA mode. The DMA Channel Interrupt Select bit determines whether to assert a PCI Bus (DMAMODEx[17]=1) or 60x Bus (DMAMODEx[17]=0) interrupt. An interrupt is pending on DMA Channel x at the interrupt pending bits listed in the following table.

Table 3-3. Pending DMA Channel x Interrupts

Interrupt Pending Bit	Channel
INTCSR[21]	0
INTCSR[22]	1
INTCSR2[5]	2
INTCSR2[5]	3

The Channel Done bit (DMACSRx[4]) can be used to determine whether an interrupt is:

- DMA Done interrupt
- Transfer complete for current descriptor interrupt

The Done Interrupt Enable bit (DMAMODEx[10]) enable a Done interrupt. In Scatter/Gather DMA mode, a bit in the Next Descriptor Pointer register of the channel (loaded from 60x Bus memory) specifies whether to assert an interrupt at the end of the transfer for the current descriptor.

A DMA Channel interrupt is cleared by the Channel Clear Interrupt bit (DMACSRx[3]=1).

3.5.7 DMA Data Transfers

The PCI 9610 DMA controller can be programmed to transfer data from the 60x-to-PCI Bus or from the PCI-to-60x Bus.

3.5.7.1 60x-to-PCI Bus DMA Transfer

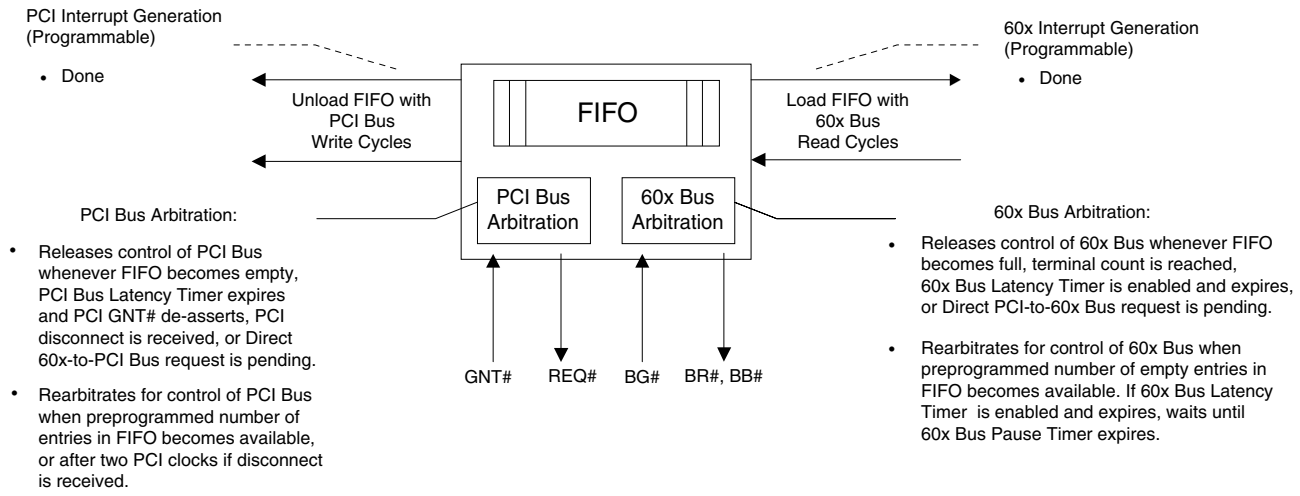


Figure 3-15. 60x-to-PCI Bus DMA Data Transfer Operation

3.5.7.2 PCI-to-60x Bus DMA Transfer

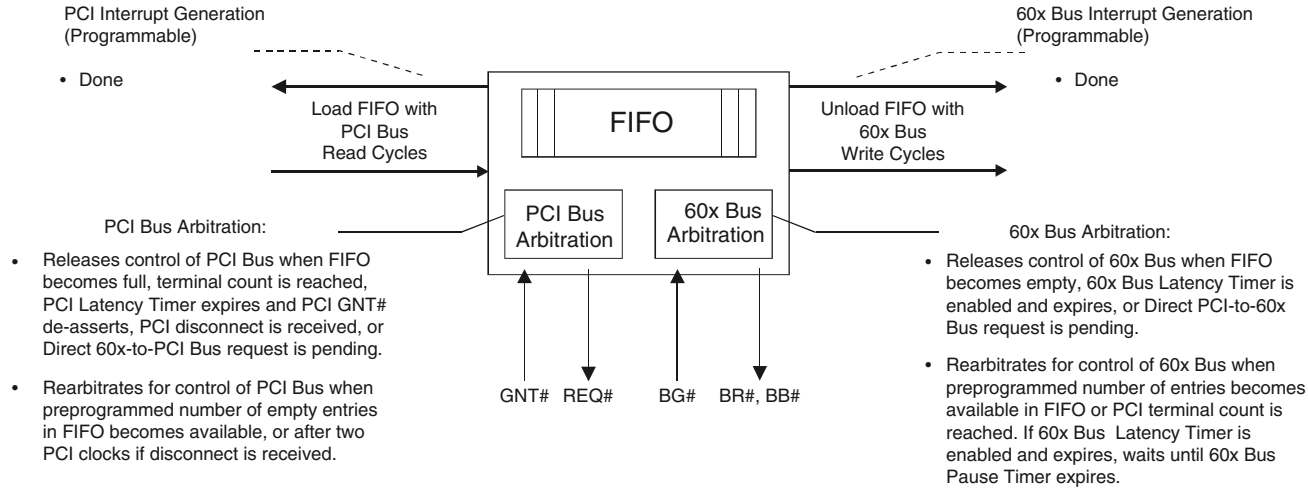


Figure 3-16. PCI-to-60x Bus DMA Data Transfer Operation

3.5.7.3 DMA Unaligned Transfers

For unaligned 60x-to-PCI transfers, the PCI 9610 reads a partial Lword from the 60x Bus. It continues to perform a Single-Cycle read (Lwords) from the 60x Bus until the nearest 16-byte boundary. If the Burst Mode bit is enabled, the PCI 9610 bursts thereafter. Lwords are assembled, aligned to the PCI Bus address, and loaded into the FIFO until the nearest 16-byte boundary.

For PCI-to-60x transfers, Lwords are read from the PCI Bus and loaded into the FIFO. On the 60x Bus, Lwords are assembled from the FIFO, aligned to the 60x Bus address and a Single cycle is written to the 60x Bus until the nearest 16-byte boundary. If burst functionality is enabled, the PCI 9610 bursts thereafter.

3.5.8 DMA Demand and End of Transfer (EOT) Operation

The PCI 9610 has four independent DMA channels. Each channel retains its own 64-bit wide, 32-deep FIFO. When transferring data using a PCI 9610 DMA channel, the PCI 9610 is the Bus Master on the PCI and 60x Buses.

External pins (not part of the 60x Bus protocol) are available to pause (Demand mode) or terminate (EOT mode) a DMA transfer.

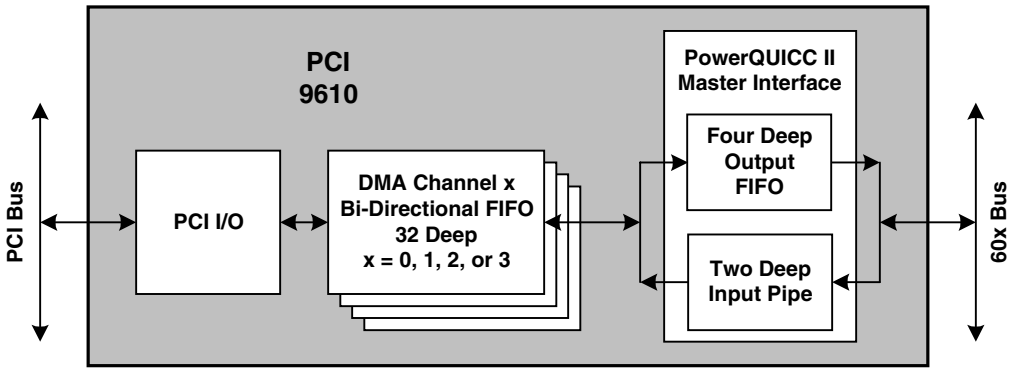


Figure 3-17. DMA General Model (Demand and EOT Modes)

3.5.8.1 Demand Operation

Demand Mode allows a 60x Bus peripheral to control the DMA data flow to/from the PCI 9610. Demand Mode is selected for each channel with register bit DMAMODEx[12].

Demand Mode uses two pins, and each channel retains its own set of Demand pins. DREQx# is an input to the PCI 9610 from the peripheral that allows data transfer when asserted low, and pauses data transfer when de-asserted high. DACKx# is the output from the PCI 9610 that tells the peripheral that the PCI 9610 is ready to transfer Channel x data when DACKx# is asserted low.

The Demand signals operate basically as follows:

- If the software issues a DMA start, and DREQx# is de-asserted, then DMA transfers wait for DREQx# assertion
- If software issues a DMA start, and DREQx# is asserted, DMA transfers begin immediately
- Single-Cycle transfers on the 60x Bus occur if the peripheral de-asserts DREQx# one clock cycle after DACKx# asserts

3.5.8.1.1 Demand Operation for Data from PCI-to-60x

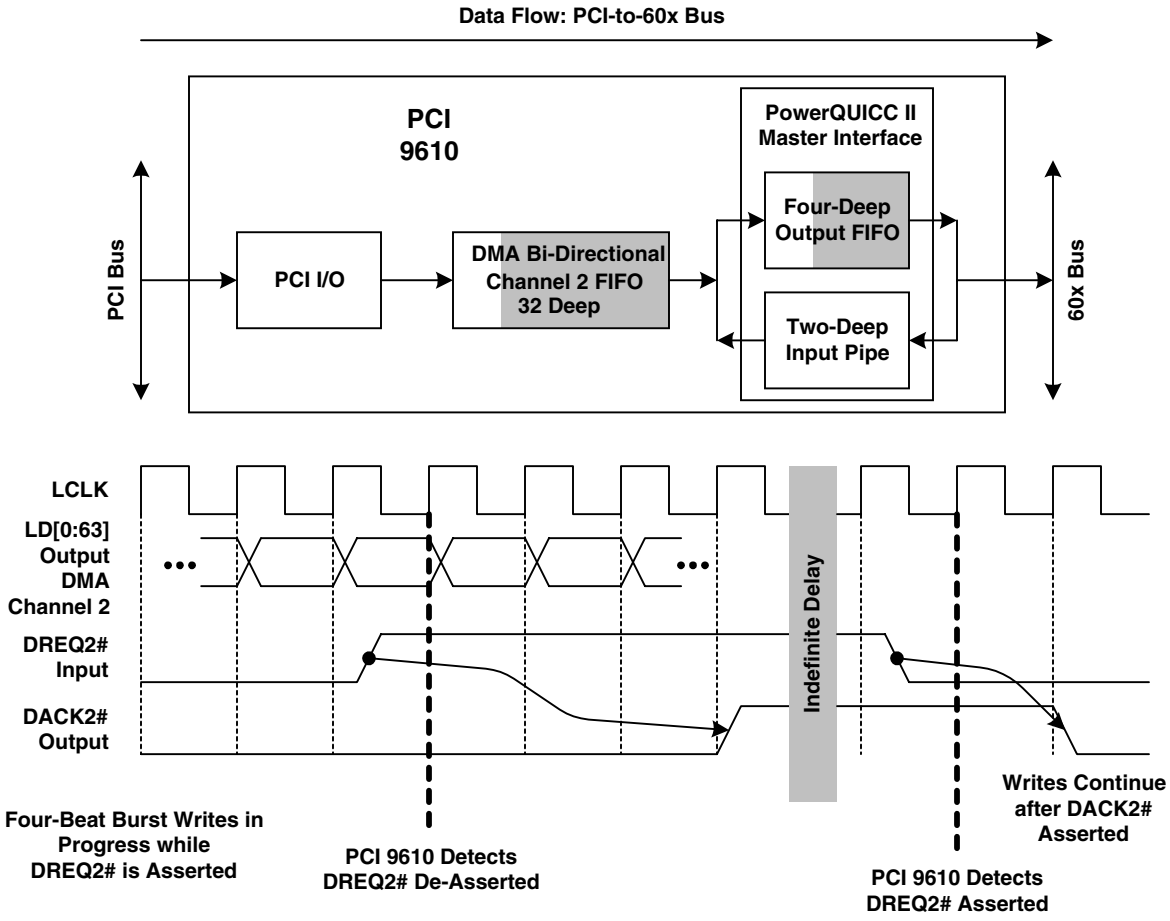


Figure 3-18. PCI-to-60x DMA Channel 2 Data Flow with FIFOs Partially Full

Note: *In Figure 3-18, shaded FIFO areas indicate how much the FIFO has filled.*

For example, DMA Demand is enabled, DREQ# asserted, and there is data in the DMA FIFO and Output FIFO, as illustrated in Figure 3-18. When the PCI PCI 9610 detects DREQx# de-asserted, the following occurs:

- If Channel x data is in the Output FIFO, but no 60x Bus Write cycle was started for Channel x, then data is held in a context switch buffer. This frees the output FIFO for other data transfers on other DMA channels or Direct Slave transfers.
- If a 60x Bus Write cycle was started (Address or Data tenure), the write completes according to protocol.
- In response to the DREQx# de-asserted, the PCI 9610 de-asserts DACKx# after the current Write cycle on Channel x completes.
- Data reads from the PCI Bus continue while Channel x is paused. PCI reads are halted if the DMA Channel x FIFO fills.
- DMA Channel x FIFO data is held until DREQx# is asserted, followed by DACKx# assertion. At this time, the PCI 9610 is ready to restart a Channel x Data transfer.

3.5.8.1.2 Demand Operation for Data from 60x-to-PCI

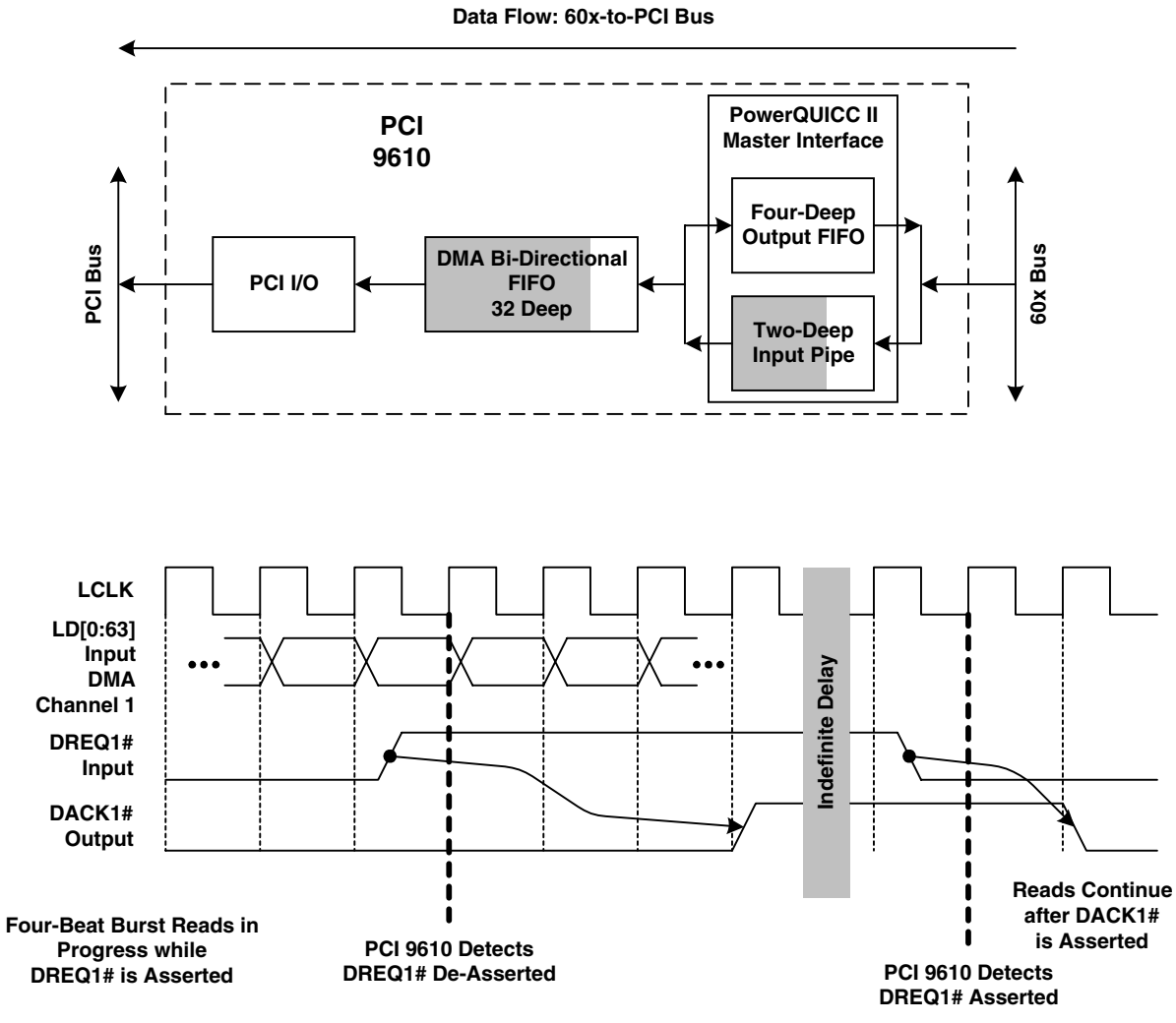


Figure 3-19. 60x-to-PCI DMA Channel 1 Data Flow with FIFOs Partially Full

Note: *In Figure 3-19, shaded FIFO areas indicate how much the FIFO has filled.*

When the PCI 9610 detects DREQ1# de-asserted, a Read cycle started on Channel 1 completes, if there was a Read cycle as illustrated in Figure 3-19. The PCI 9610 de-asserts DACK1#. Data in the input pipe for Channel 1, and data in the Channel 1 DMA FIFO continue to be transferred and written to the PCI Bus. Writing data to the PCI Bus stops when the FIFO is empty. When DREQ1# is asserted, the PCI 9610 asserts DACK1#, and resumes the 60x-to-PCI Bus DMA transfers.

3.5.8.1.3 Single-Cycle Demand

The DREQ#/DACK# pin pair allows single-cycle DMA. If a peripheral asserts DREQx#, waits for DACKx# assertion, and de-asserts DREQx# one clock cycle after DACKx# assertion, then the PCI 9610 transfers one Data Cycle read or write on the 60x Bus.

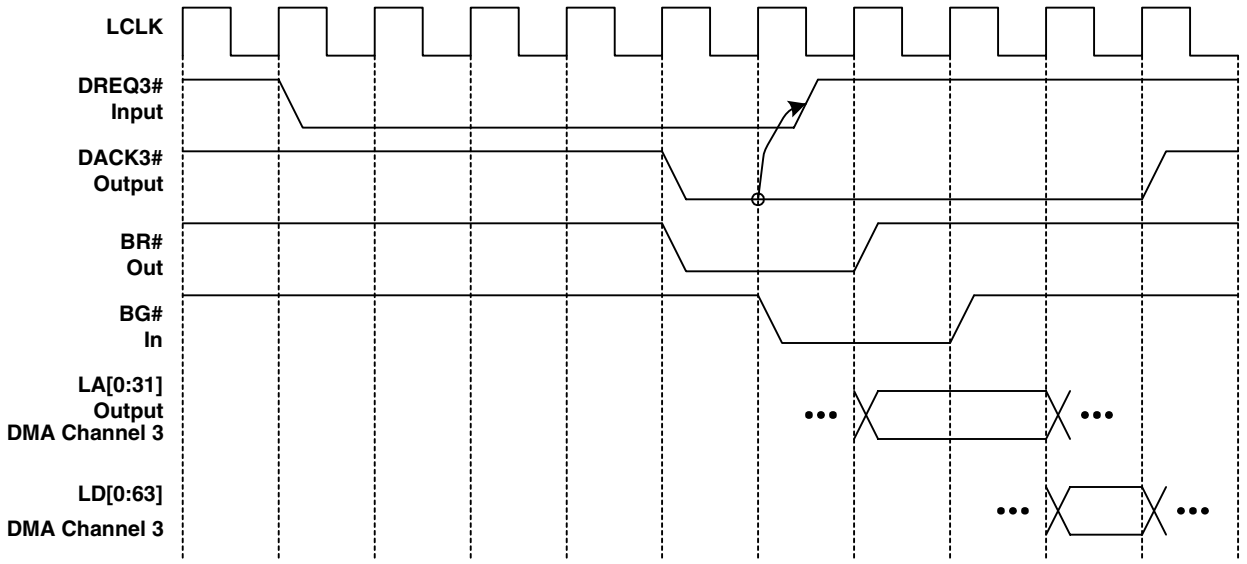


Figure 3-20. Single-Cycle Demand Example

In Figure 3-20, DACK3# assertion was delayed a few clocks after DREQ3# assertion, perhaps because the PCI 9610 was busy with other transfers. DACKx# assertion might come as soon as one clock after DREQx# assertion.

Once asserted, DACK3# continues to be asserted until the Address and Data tenures are complete.

If the external device de-asserts DREQ3# one clock after DACK3# assertion, as illustrated in Figure 3-20, there is a Single-Cycle transfer.

3.5.8.2 End of Transfer (EOT) Operation

EOT operation is enabled for each DMA channel by register bit DMAMODEx[14].

Each DMA channel retains its own EOTx# input. When EOTx# is asserted, transfer on Channel x stops. For PCI-to-60x transfers, data in the DMA FIFO is flushed at the time of the EOT. For 60x-to-PCI transfers, data in the DMA FIFO at the time the EOT is allowed to write to the PCI Bus, but no further data is taken from the 60x Bus.

3.5.8.2.1 EOT in the PCI-to-60x Direction

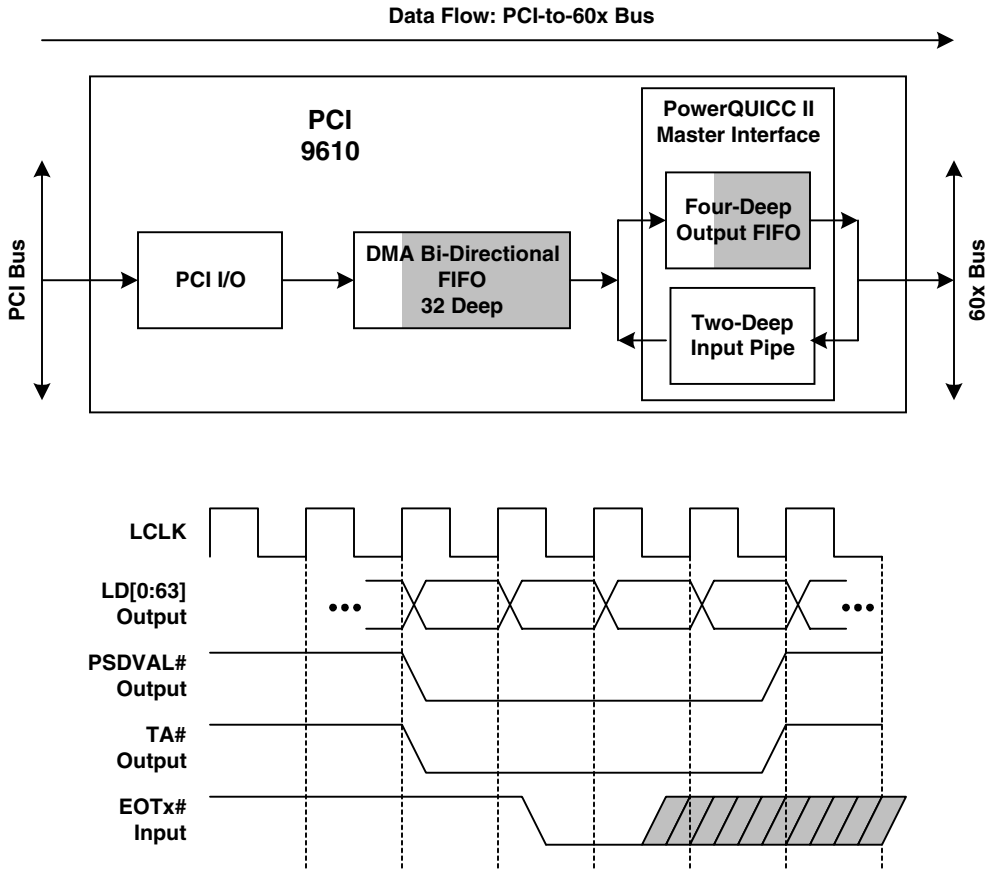


Figure 3-21. EOT Example—EOTx# Asserted as Channel x Writes to the 60x Bus

Note: *In Figure 3-21, shaded FIFO areas indicate how much the FIFO has filled.*

In Figure 3-18, "PCI-to-60x DMA Channel 2 Data Flow with FIFOs Partially Full," on page 3-22, when the PCI 9610 detects EOTx# asserted, data for the channel in progress is allowed to complete. That is, if a 60x Write cycle (Address or Data tenure) for Channel x was in progress at the time of the EOT, the Write cycle completes. All other data for Channel x in the PCI 9610 at the time of the EOT is flushed, which includes data in the Channel x DMA FIFO. This also includes data in the Output FIFO (if it is for Channel x DMA) that has not started into a 60x Bus Write cycle.

In Figure 3-21, a four-beat Burst 60x Master write has started. EOT occurs during the four-beat burst. The PCI 9610 completes the current four-beat Burst cycle, then flushes any data held in the DMA FIFO and PowerQUICC II Interface (PQIF) Master Write FIFO.

EOTx# returns de-asserted (high) any time after one full clock asserted (low). EOTx# must be a minimum of one full clock cycle in duration. EOTx# must be de-asserted before a new DMA cycle begins.

3.5.8.2.2 EOT in the 60x-to-PCI Direction

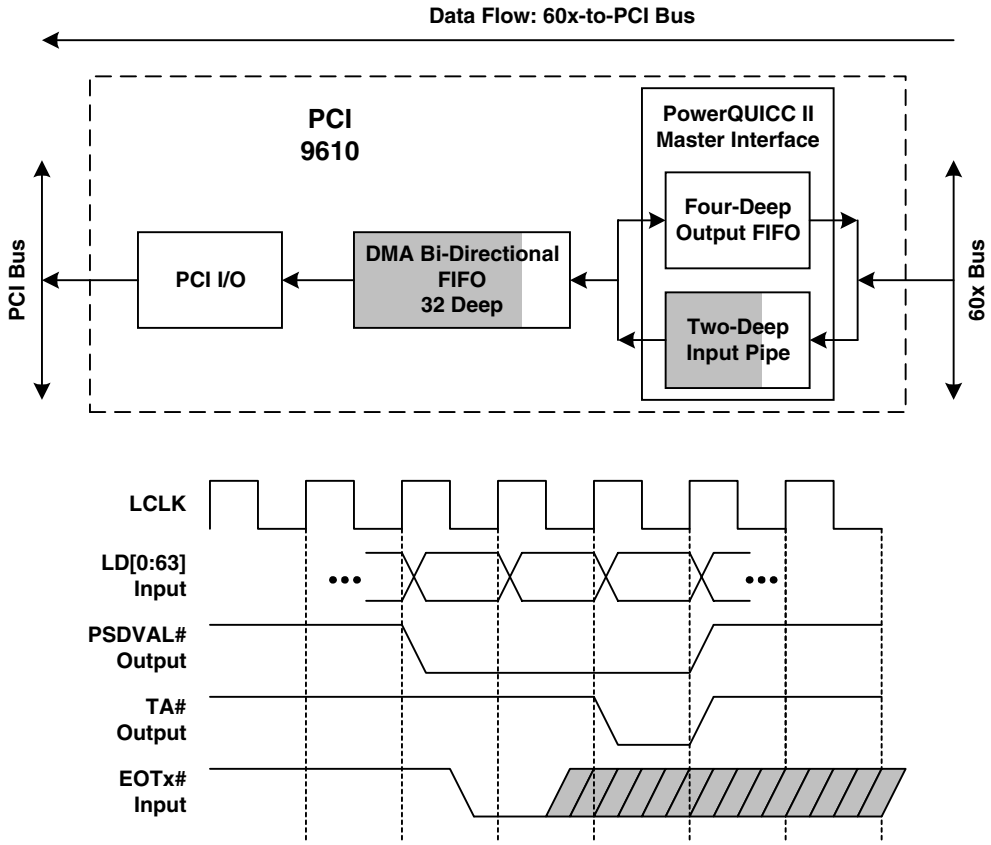


Figure 3-22. EOT Example—EOTx# Asserted as Channel x Reads to the 60x Bus

Note: *In Figure 3-22, shaded FIFO areas indicate how much the FIFO has filled.*

When the PCI 9610 detects EOTx# asserted, any data in the PowerQUICC II Input FIFO, is transferred to the DMA FIFO. Data in the DMA FIFO is transferred to the PCI Bus.

In Figure 3-22, a three-beat Burst Master read has started. EOT occurs during the three-beat burst. The PCI 9610 completes the current three-beat Burst cycle, then flushes any data held in the DMA FIFO and PQIF Master Read FIFO.

Note: *A burst less than four beats can occur at the end of a DMA transfer where the DMA length is not a multiple of 32 bytes, or for DMA transfer lengths equal to or less than 24 bytes.*

EOTx# returns de-asserted (high) any time after one full clock asserts (low). EOTx# must be de-asserted before a new DMA cycle begins.

3.5.9 60x Bus DMA Latency and Pause Timers

In the PLX Extended Burst Mode, the 60x Bus Latency Timer (MARBR[7:0]) determines the number of LCLK clocks the PCI 9610 can burst before relinquishing the 60x Bus. The 60x Bus Pause Timer (MARBR[51:8]), used only in the PLX Extended Burst, sets the minimum pause time between DMA Data bursts.

3.5.10 DMA Arbitration

The PCI 9610 asserts BR# when it needs to be the 60x Bus Master. Upon receiving BG#, the PCI 9610 waits for BB# to be de-asserted. The PCI 9610 then asserts BB# at the next rising edge of the 60x Bus clock after sensing that BB# is de-asserted (no other

device is acting as 60x Bus Master). The PCI 9610 continues to assert BB# while acting as the 60x Bus Master (*that is*, it holds the bus until instructed to release BB#) under the following conditions:

- 60x Bus Latency Timer is enabled and expires (MARBR[7:0])
- Direct Slave access is pending
- EOT# input is received (if enabled)

The DMA controller releases control of the PCI Bus when one of the following conditions occurs:

- FIFOs are full or empty
- PCI Bus Latency Timer expires (PCILTR[7:0])—and loses the PCI GNT# signal
- Target disconnect response is received

The DMA controller de-asserts PCI REQ# for a minimum of two PCI clocks.

3.5.11 60x Bus Latency and Pause Timers

The 60x Bus Latency and Pause Timers are programmable with the Mode/DMA Arbitration register (MARBR[7:0, 15:8], respectively). If the 60x Bus Latency Timer is enabled and expires, the PCI 9610 completes an Lword transfer up to the nearest 16-byte boundary and releases the 60x Bus, de-asserting BB#. After the programmable Pause Timer expires, it arbitrates for the bus by asserting BR#. When it receives BG#, it asserts BB# and continues to transfer until the FIFO is empty for a 60x-to-PCI transfer or full for a PCI-to-60x Bus transfer.

The DMA transfer can be paused by writing a 0 to the Channel Enable bit. To acknowledge the disable, the PCI 9610 gets at least one data from the bus before it stops. However, this is not recommended during a burst.

The DMA 60x Bus Timer starts after the 60x Bus is granted to the PCI 9610 and the 60x Bus Pause Timer starts after BB# de-asserts.

3.6 60X BUS PROTOCOL OPERATION

The 60x Bus consists of the Motorola 60x Bus and the PLX Extended Burst option. The PLX Extended Burst is compatible with the 60x Bus and transparent to devices connected to the 60x Bus that do not support PLX Extended Burst.

3.6.1 Motorola 60x Bus

The 60x Bus is defined in detail in the *Motorola MPC8260 PowerQUICC II User's Manual*.

The PCI 9610 supports the 60x Bus protocol as a 60x Bus slave or master.

3.6.1.1 60x Bus Overview

The 60x Bus is used by the following PowerPC processors:

- PowerQUICC II MPC8260
- MPC740 and MPC750
- PowerPC MPC603e

Major 60x Bus features are as follows:

- 32-bit address, 64-bit data, non-multiplexed
- Split Address/Data tenures provide address pipeline
- Four-beat bursts allows up to 32-byte burst
- Uses Big Endian data with support for Little Endian

3.6.1.2 Address and Data Tenures

The 60x Bus consists of two separate, split transaction buses—Address and Data. Each Bus retains its own tenure. The Address and Data buses support synchronous, one-level deep pipeline transactions.

An arbiter controls access to the Address Bus. An arbiter might be the MPC8260, or other. The PCI 9610 is not an Address Bus arbiter. Arbitration for the Address bus is as follows:

- A 60x Bus master requests the Address Bus by asserting Bus Request, BR#.
- The 60x Bus arbiter grants the Address Bus by asserting Bus Grant, BG#.
- Each prospective 60x Bus Master has its own BR#/BG# signal pair.
- When a master is granted the Address Bus, the Master initiates an Address tenure.
- The Slave decodes the address and responds.

The Data Bus tenure follows the Address Bus tenure after the current Data Bus cycle is complete. A Data Bus tenure from a previous Address tenure can coincide in time with the next Address Bus tenure.

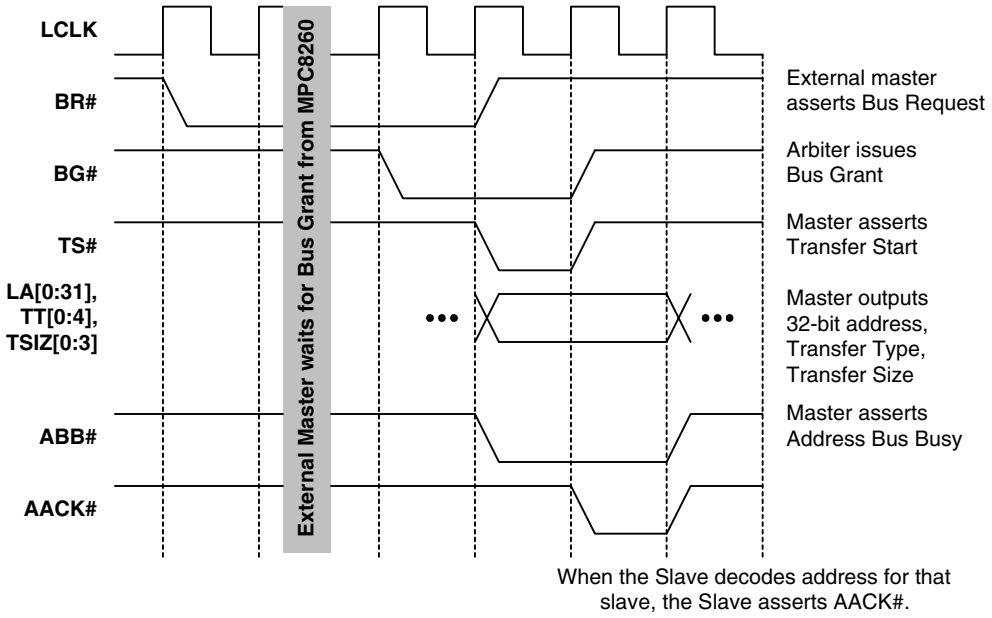


Figure 3-23. 60x Bus Address Tenure Example

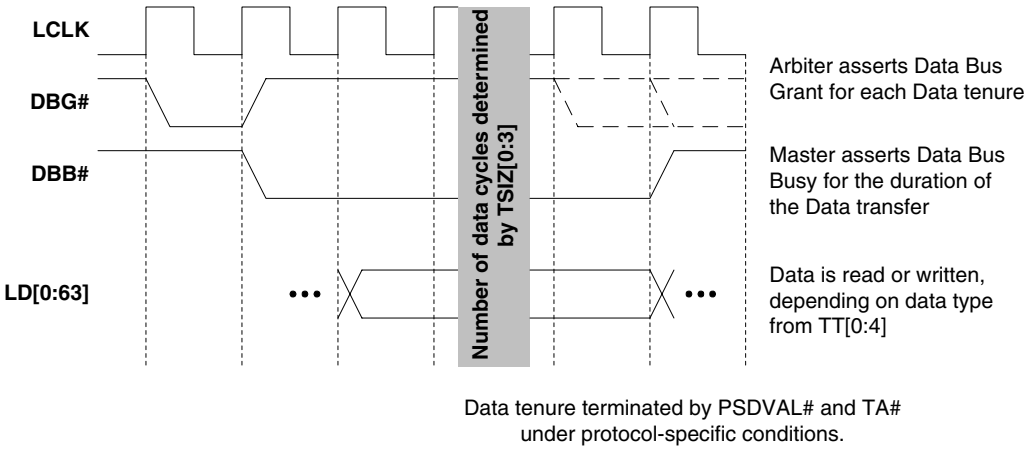


Figure 3-24. 60x Bus Data Tenure Example

3.6.1.3 Pipelining

Address and Data tenures may be non-pipelined, meaning that the tenures do not overlap. If the two tenures overlap, they are pipelined. The 60x Bus allows a one-deep pipeline.

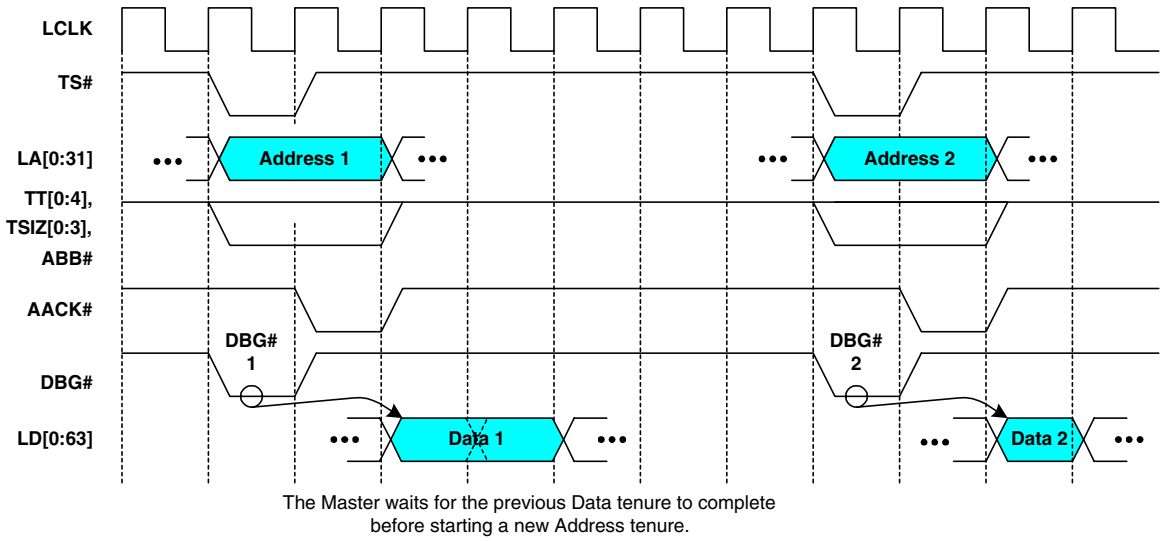


Figure 3-25. Non-Pipelined Example

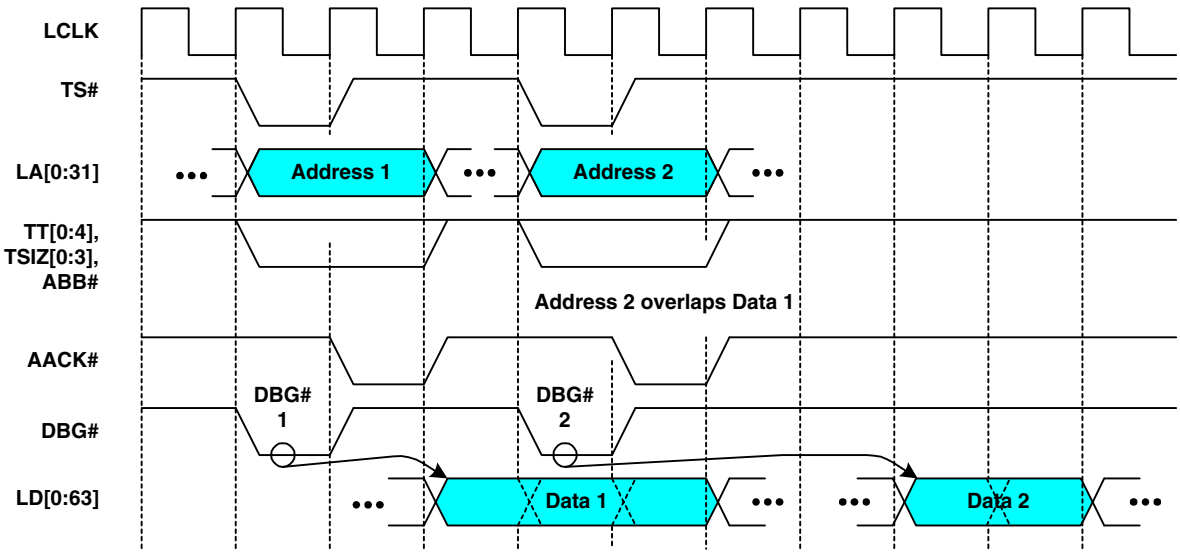


Figure 3-26. Pipelined (Overlapped) Address/Data Tenure Example

If a 60x Bus Master cannot pipeline, then the Master delays the start of the second Address tenure until the completion of the first Data tenure. If a 60x Bus Master can pipeline, then the Master starts the second Address tenure as soon as possible. The PCI 9610 works with either case.

As a slave, the PCI 9610 Slave tracks a master doing pipelining. The PCI 9610 is a slave for Direct Master Read and Write transfers.

If a targeted 60x Bus Slave cannot pipeline, that Slave delays AACK# on the second Address tenure. If the Slave can pipeline, AACK# assertion by the Slave is not delayed. The PCI 9610 works with either case.

As a master, the PCI 9610 is ready for pipelined or non-pipelined operation from the addressed Slave. The PCI 9610 is a master for Direct Slave Read and Write transfers, and for DMA transfers.

3.6.1.4 Transfer Types Supported by the PCI 9610

Each Address tenure retains a transfer type, generated by the 60x Bus Master, to determine the type of transfer. When the PCI 9610 is a 60x Bus Master, it supports the transfer types listed in the following table.

Table 3-4. Transfer Types when the PCI 9610 is a 60x Bus Master (Direct Slave and DMA)

TT[0:4]	Transfer Types
01010	Read.
11010	Read Atomic if Lock = 1.
00010	Write with Flush.
10010	Write with Flush Atomic.

The “Lock bit” refers to a tag bit in the Direct Slave Write FIFO. When the PCI Bus LOCK# input is asserted for a Direct Slave write, the PCI 9610 initiates a write with Flush Atomic in the corresponding 60x Bus transfer. If the PCI Bus LOCK# input was asserted for a Direct Slave read, then the PCI 9610 initiates a Read Atomic in the corresponding 60x Bus transfer. (The Read command goes through the Direct Slave Write FIFO before the read is initiated.)

As a 60x Bus slave, the PCI 9610 supports the transfer types listed in the following table.

Table 3-5. Transfer Types when the PCI 9610 is a 60x Bus Slave (Direct Master Only)

TT[0:4]	Transfer Types
01010	Read
11110	Read with intent to modify atomic. The PCI 9610 sets the Lock bit.
11010	Read atomic. The PCI 9610 sets the Lock bit.
01011	Read with no intent to cache.
01110	Read with no intent to modify.
11100	Graphics read.
00010	Write with Flush.
10010	Write with Flush Atomic. The PCI 9610 sets the Lock bit.
10100	Graphics write.
00110	Write with terminate.

The PCI 9610, as a 60x Bus slave, supports address-only transfer types, as listed in the following table. When the PCI 9610 detects an address-only transfer type, within an address space recognized by the PCI 9610, then the PCI 9610 asserts AACK#. There is no associated read or write with an address-only transfer type.

Table 3-6. Address-Only Transfer Types Supported when the PCI 9610 is a 60x Bus Slave (Direct Master Only)

TT[0:4]	Address-Only Transfer Types
00000	Clean Block.
00100	Flush Block.
01000	Sync.
01100	Kill Block.
10000	EIEIO.
11000	TLB Invalidate.
00001	LWARX.
01001	TLB Sync.
01101	ICBI.

All transfer types not included in the previous three tables are ***reserved***. The PCI 9610 does not respond to a reserved TT[0:4] as a slave, nor does the PCI 9610 issue a reserved TT[0:4] as a master.

3.6.1.5 Transfer Size

During an Address tenure, the 60x Bus Master sets the transfer size. For the PLX Extended Burst, TSIZ# is set to the bus width. (Refer to Section 3.6.7.2.) The transfer sizes are listed in Table 3-7.

3.6.1.6 Strict MPC603e Mode

Strict MPC603e mode allows the MPC740 and MPC750, as well as other PowerPC chips, to interface with the PCI 9610. It is selected independently for Space 0, Space 1, Expansion ROM, and each DMA channel. (Refer to the register control bits, PPCTL2[7:3] and PPCTL3[3:0].)

The PCI 9610 affects only the Strict MPC603e-compatible processor by not using TSIZ# values of 5, 6, 7, 9, and 10 bytes when writing to the 60x Bus.

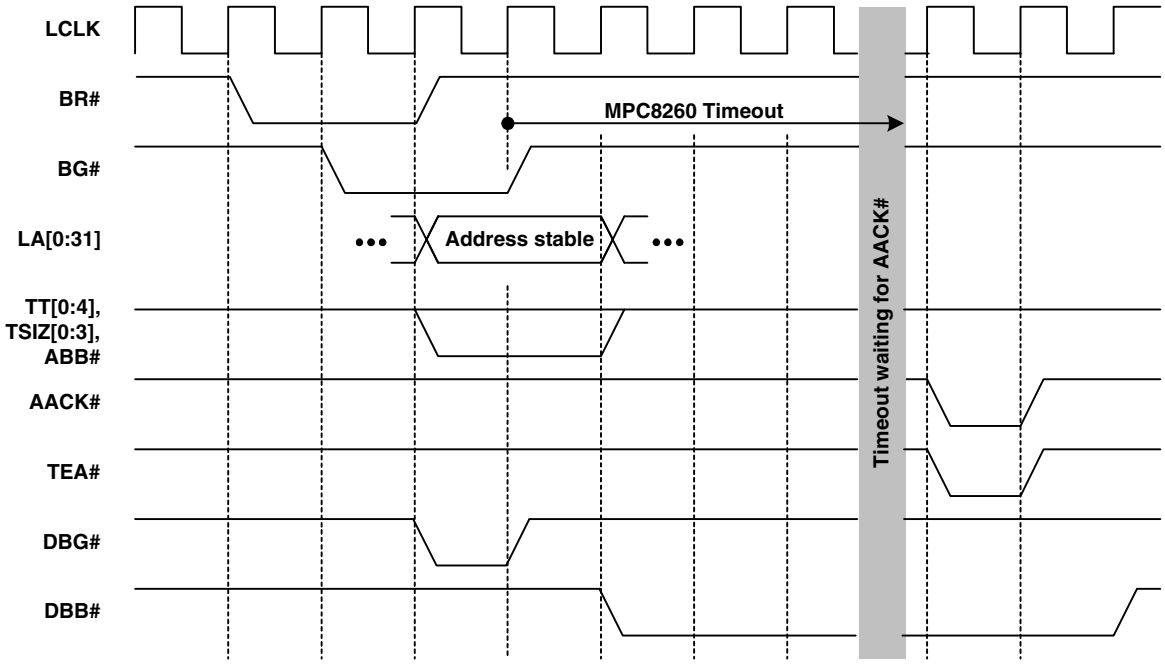
The MPC603e-compatible processor must connect its TSIZ0, TSIZ1, and TSIZ2 pins to the 60x Bus TSIZ1, TSIZ2, and TSIZ3, respectively.

3.6.1.7 AACK# Timeout

When the PCI 9610 is a master on the 60x Bus, and there is no AACK# response from the Slave, it is the responsibility of the MPC8260, or other 60x Bus arbiter, to recover. The recovery is shown in Figure 3-27.

Table 3-7. Address Tenure Transfer Sizes

TBST#	TSIZ[0:3]	Size in Bytes	Size in Qwords	Comments
De-Asserted	0001	1	—	
	0010	2		
	0011	3		
	0100	4		
	0101	5		Extended five bytes
	0110	6		Extended six bytes
	0111	7		Extended seven bytes
	0000	8	1	On a 64-bit bus, this is a single-beat burst
	1001	16	2	On a 64-bit bus, this is a two-beat burst
	1010	24	3	On a 64-bit bus, this is a three-beat burst
Asserted	0010	32	4	On a 64-bit bus, this is a four-beat burst



The MPC8260 times out when there is no AACK# assertion. The MPC8260 simultaneously asserts AACK# and TEA# to terminate both Address and Data tenures.

Figure 3-27. MPC8260 AACK# Timeout and Address/Data Tenure Termination

3.6.2 Cache Wrap

The following describes the 60x Bus Cache Wrap:

- Critical Qword (64 bits) First
- Address bits LA[27:28] select the Qword boundary
- The PCI 9610 has a separate cache wrap enable/disable bit, PPCTL1[0]
  - When PPCTL1[0]=0, cache wrap is enabled (default)
  - When PPCTL1[0]=1, cache wrap is disabled
- Cache wrap applies only when the PCI 9610 is a slave on the 60x Bus. Therefore, this applies only in Direct Master Write and Read transfers. Cache wrap does not apply to DMA, nor Direct Slave read or write.

Table 3-8. Cache Wrap Qword Starting Address

Data Transfer	LA[27:28]=00	LA[27:28]=01	LA[27:28]=10	LA[27:28]=11
1st Data Beat	Qword 0	Qword 1	Qword 2	Qword 3
2nd Data Beat	Qword 1	Qword 2	Qword 3	Qword 0
3rd Data Beat	Qword 2	Qword 3	Qword 0	Qword 1
4th Data Beat	Qword 3	Qword 0	Qword 1	Qword 2

3.6.3 Big/Little Endian Control

The PCI Bus is always Little Endian. The 60x Bus is Big Endian, and supports Big Endian and Little Endian devices.

Individual configuration bits select 60x Bus Big/Little Endian for Register space, Direct Master, Direct Slave Space 0 and 1, Slave Expansion ROM space, and each DMA channel. Each configuration bit selects Little Endian when 0, and Bit Endian when 1, and each bit defaults to 0 after reset. Table 3-9 provides a summary of each bit.

Register space and Direct Master configuration bit selections may be overridden by the BIGEND# input pin. If the BIGEND# input is low, Register space and Direct Master access are Big Endian after reset. If the BIGEND# input is high, Register space and Direct Master access are Little Endian after reset.

If BIGEND# is tied to ground, then Register space and Direct Master are Big Endian after reset. Then, the software may set all Endian configuration bits to 1.

Byte swapping is performed inside the PCI 9610 and adds no latency, no matter which Endian is selected.

3.6.4 Byte Lanes

The PCI 9610 may connect to a 64- or 32-bit PCI Bus.

The 60x Bus is a 64-bit Bus. The 60x Bus may have 64-, 32-, 16-, and 8-bit peripherals connected.

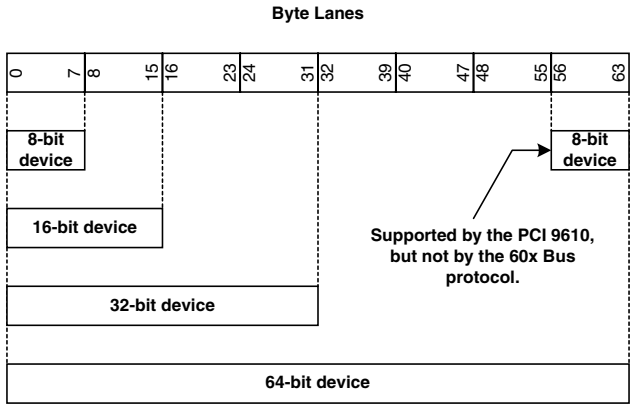


Figure 3-28. PCI 9610 Byte Lane Possibilities

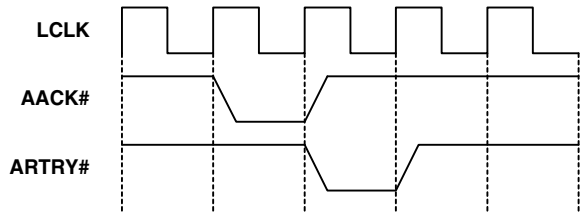
Single byte lane assignment as bits [0:7], default, when BIGEND[4] is 0. Bits [56:63] are selected when BIGEND[4] is 1.

Table 3-9. Endian Configuration Bit Summary

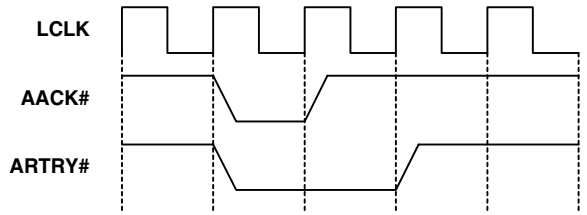
Bit	Endian Description	Comments
BIGEND[0]	Register Space	May be overridden by BIGEND# is this bit is set to 0.
BIGEND[1]	Direct Master Access	May be overridden by BIGEND# is this bit is set to 0.
BIGEND[2]	Direct Slave Space 0	
BIGEND[3]	Direct Slave Expansion ROM	
BIGEND[5]	Direct Slave Space 1	
BIGEND[6]	DMA Channel 0	
BIGEND[7]	DMA Channel 1	
BIGEND2[0]	DMA Channel 2	
BIGEND2[1]	DMA Channel 3	

3.6.5 Address Retry (ARTRY#)

A valid ARTRY# is one that is asserted in the clock immediately following the assertion of AACK#.



ARTRY# immediately follows AACK#



ARTRY# starts with AACK#, and continues for one more clock

Figure 3-29. Valid ARTRY# Examples

When outputting an ARTRY# as a 60x Bus slave, the PCI 9610 uses the top waveform for Delayed Read Direct Master read, and the lower waveform for Direct Master write.

3.6.5.1 ARTRY# Output when the PCI 9610 is a 60x Bus Slave

The PCI 9610 generates an ARTRY# output under the following conditions:

- The PCI 9610 operates as a 60x slave in Direct Master Write and Direct Master Read transfers. If enabled, the PCI 9610 may assert ARTRY# in Direct Master write or Direct Master read, as described in this section.
- As a 60x Bus Master, the PCI 9610 asserts ARTRY# if DBG# input is not received at the correct TS# or on the clock immediately following TS#. (Refer to Section 3.6.5.2.)

3.6.5.1.1 ARTRY# Output Enables for the PCI 9610 as a 60x Bus Slave

The 60x Bus Direct Master Delayed Read Enable bit (LMISC1[4]) enables ARTRY# for Direct Master reads. The Direct Master Write FIFO Almost Full ARTRY# Enable bit (LMISC1[6]) enables ARTRY# for Direct Master writes. If both LMISC1[4] and LMISC1[6] are 0, the PCI 9610 never asserts ARTRY# as a slave.

Writing a 1 to LMISC1[4] enables the PCI 9610 to operate in Delayed Transaction mode for Direct Master reads. The PCI 9610 issues an ARTRY# to the 60x Bus Master and prefetches Read data from the PCI Bus.

When LMISC1[6] is set to 1, the PCI 9610 issues an ARTRY# if the Direct Master Write FIFO is almost full.

3.6.5.1.2 ARTRY# Asserted by the PCI 9610 in Delayed Direct Master Read

Given that a 60x Bus master has addressed the PCI 9610 for a read from the PCI Bus, then the following applies:

- If LMISC1[4]=0, the PCI 9610 issues AACK# in the Address tenure, and starts the cycle to obtain the data from the PCI Bus. The PCI 9610 does not issue ARTRY#. The PCI 9610 delays the assertion of PSDVAL# until the Read data is ready to be outputted onto the 60x Bus. If PCLK and LCLK are nearly the same frequency, then the number of LCLKs from AACK# to PSDVAL# assertion is approximately 17 clocks.
- If LMISC1[4]=1, the PCI 9610 asserts ARTRY# (one clock after asserting AACK# as shown in the previous drawing). The PCI 9610 starts receiving data from the PCI Bus. When the 60x Master addresses the PCI 9610, and the Read data is not ready, then another ARTRY# is issued. When the data is ready, the PCI 9610 responds with a normal Direct Master read, and the Data read continues.
- A timer, RTY_TMR, and an address buffer, RD_RTY_ADDR, control the states of address Retry. The RTY_TMR is a counter of 2^{15} LCLKs.

In Delayed Direct Master Read, the following applies:

- When the first read command is received by the PCI 9610, it issues ARTRY#, and starts filling the Direct Master Read FIFO from the PCI Bus. The RTY_TMR is started. The address is saved in the RD_RTY_ADDR buffer.
- If a read command is received with address that matches RD_RTY_ADDR, and data is ready in the Direct Master Read FIFO, then the PCI 9610 outputs the data for the read command. The RTY_TMR timer is reset.
- If a Direct Master Read command is received with an address that matches RD_RTY_ADDR, and data is not ready from the Direct Master Read FIFO, the PCI 9610 issues an ARTRY#, and re-starts the RTY_TMR.
- If a Direct Master Read command is received that does not match RD_RTY_ADDR, or if a Direct Master Write command is received, then the PCI 9610 issues an ARTRY#. Under this condition, the address is not saved. The contents of RD_RTY_ADDR are not changed. The RTY_TMR timer continues.
- If a read or write command is received for access to an internal PCI 9610 register, then the PCI 9610 issues an ARTRY#. Under this condition, the address is not saved. The contents of RD_RTY_ADDR are not changed. The RTY_TMR timer continues.
- If the RTY_TMR expires, the PCI 9610 clears the RD_RTY_ADDR buffer, and returns to the normal state. Register access, Direct Master read, and Direct Master write may operate as normal.

3.6.5.1.3 ARTRY Asserted by the PCI 9610 in Direct Master Write when the FIFO is Almost Full

When the PCI 9610 Slave receives a series of Direct Master Write transfers from a 60x Bus master, the Direct Master Write FIFO passes the almost full threshold, and LMISC1[6] is set to 1, then the PCI 9610 issues an ARTRY#. The ARTRY# in this case starts with AACK# and continues for one full clock after AACK# de-asserts.

The PCI 9610 internal PQIF 60x Slave module contains no address buffer for Direct Master write, and is not expecting the master to return. If the Direct Master Write FIFO is stuck (cannot empty the data), then there is an error on the PCI Bus that is handled in the PCI Bus error status/recovery.

3.6.5.2 PCI 9610 as a 60x Master, Response to Missing DBG#

If the PCI 9610, as a 60x Bus Master, issued BR#, and received BG#, then the PCI 9610 expects a Data Bus Grant (DBG#). The DBG# must come with TS# (on the same clock cycle), or in the clock immediately following DBG#. As shown in Figure 3-30, the PCI 9610 issues ARTRY# one clock after AACK# if there was no DBG#.

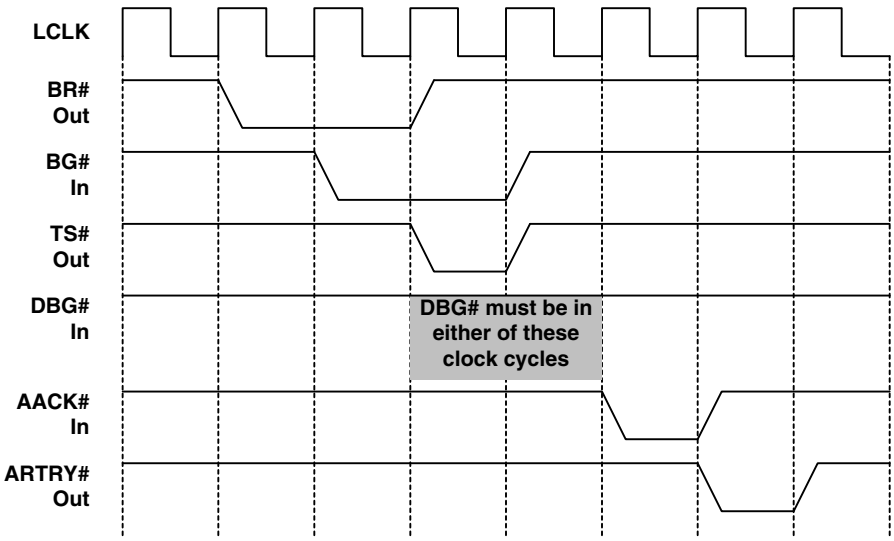


Figure 3-30. Missing DBG# Response

3.6.5.3 Accessing Internal PCI 9610 Registers Using CCS#

To access a PCI 9610 internal register from the 60x Bus, CCS# input must be used. The 60x Bus Master asserts CCS# with the ABB# signal. When accessing PCI or PCI I/O using Direct Slave, the 60x Bus Master (or PLX Extended Burst Master) must de-assert CCS#.

Internal access to PCI 9610 registers cannot be accomplished using PLX Extended Burst.

3.6.5.4 Using MDREQ#

MDREQ# is an output from the PCI 9610, which indicates whether the Direct Master Write FIFO is almost full.

One method of using MDREQ# would be for a 60x Master to address the PCI 9610 only when MDREQ# is asserted low. If the 60x Master sees that MDREQ# is de-asserted high, the Master waits for MDREQ# to assert.

The MDREQ# signal may be used with or without the ARTRY# option described in Section 3.6.5.1.3.

3.6.5.5 How the PCI 9610 Reacts to ARTRY# Input

When the PCI 9610 is a 60x Bus master, if the slave asserts ARTRY# on the Clock cycle immediately following AACK#, the PCI 9610 de-asserts DBB# and any other asserted bus signal. In other words, the PCI 9610 “gets off the bus.” The PCI 9610 then starts a new Master cycle by asserting BR#. The PCI 9610 Master Retries the same Master access as before, and continues until a successful Retry completes.

3.6.6 60x Bus Accesses to PCI I/O and PCI 9610 Registers

Read and write access to PCI I/O from the 60x Bus is limited Lword (32-bit) access. Read and write access to internal PCI 9610 registers from the 60x Bus is limited Lword (32-bit) access.

When a 60x Bus Master attempts to access PCI I/O or PCI 9610 internal registers with more than one Lword, the PCI 9610 asserts TEA#, and the access aborts.

3.6.7 PLX Extended Burst

3.6.7.1 PLX Extended Burst Features

While the 60x Bus protocol limits data bursts to four beats, the PLX Extended Burst operation allows indefinite bursts on the 60x Bus. This mode is an extension of the 60x Bus protocol, and is backward compatible to the 60x Bus and Strict MPC603e mode.

The PLX Extended Burst mode is transparent to the MPC8260 and any other device on the 60x Bus that does not use the PLX Extended Burst.

3.6.7.2 PLX Extended Burst Operation

In PLX Extended Burst, the Transfer Types (TT signals) are identical to the 60x Bus protocol. The TSIZ field is set (by the 60x Master using PLX Extended) to be equal to the bus width (in bytes).

The following three signals facilitate PLX Extended bursts:

- **PLX_TBST#**—Transfer start during PLX Burst mode. Indicates the master wants to transfer data using PLX extended burst mode (driven by the Master).
- **PLX_DVAL#**—Data valid during PLX Burst mode. Indicates extended burst data is available for read or needed for write (driven by the Slave).
- **PLX_MT#**—Master Terminate Burst. Indicates the PowerQUICC II Master wants to terminate the extended burst on the following Data transfer (driven by the Master).

A PLX Extended Burst transfer is started with the same protocol as the 60x Bus, except that the TSIZ# is set equal to the bus width. This tells the PLX Extended Burst capable slave that the Data tenure is PLX Extended. Address and Data tenure operation are the same as the 60x Bus, except that the data burst may perform longer than four beats.

Because transfers are of indefinite length, there is a mechanism to terminate the transfer. Either the Master or the Slave may terminate the transaction.

A Master termination occurs when the Master asserts PLX_MT# followed, on the next clock, by the Slave asserting PSDVAL# and TA# at the same time.

In a Slave termination, the slave asserts PSDVAL# and TA# at the same time without a prompt from the master assertion of PLX_MT#.

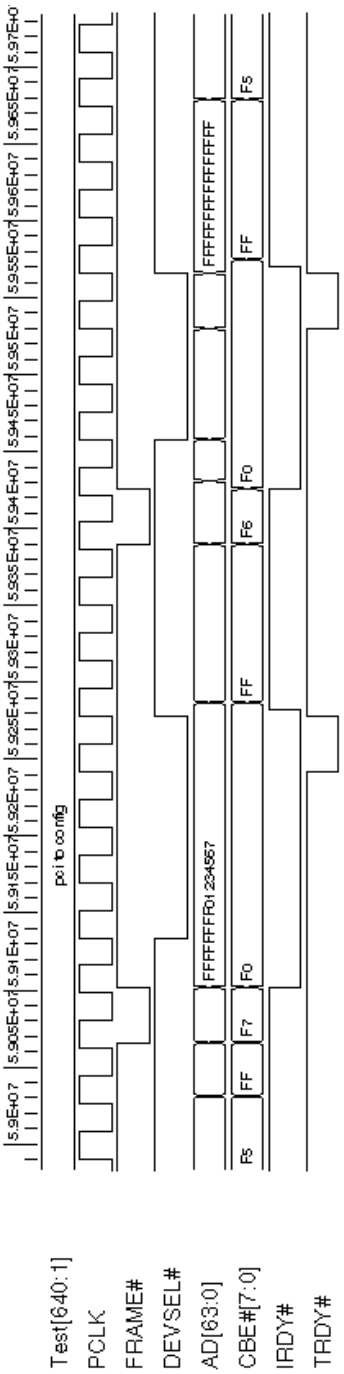
Therefore, an arbiter on the 60x Bus need not know about the PLX Extended Burst, because the arbiter watches PSDVAL# and TA# to determine when the transfer is complete.

The following table lists the PLX Extended Burst TSIZ# fields.

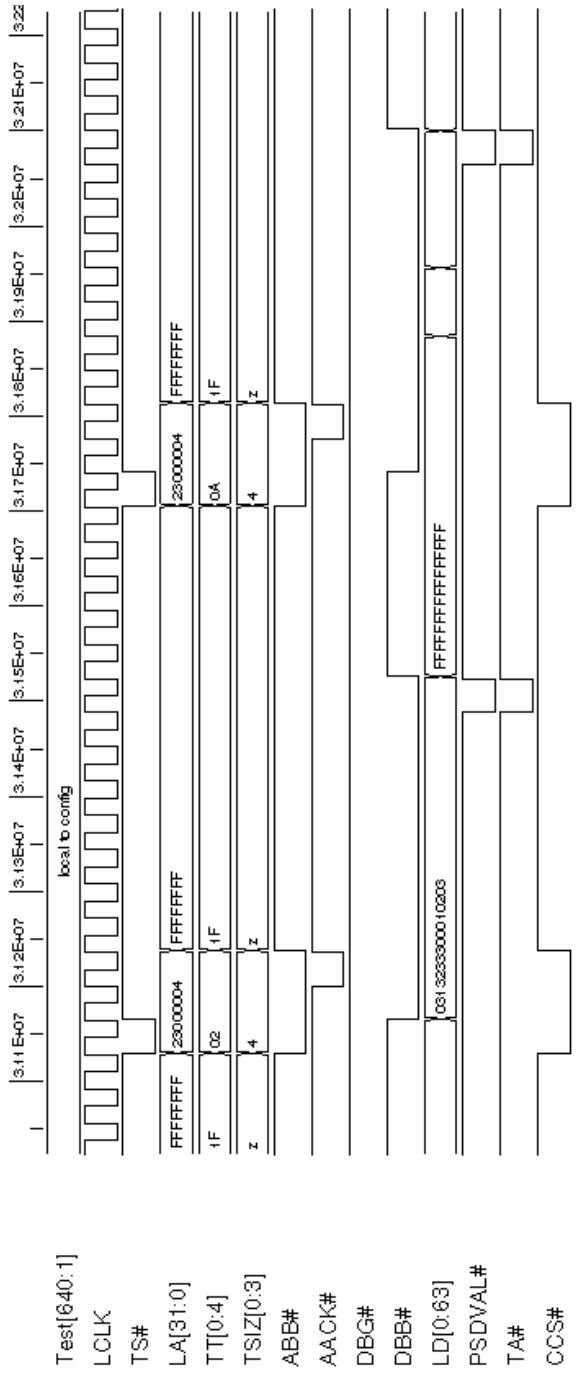
Table 3-10. PLX Extended Burst TSIZ Fields

TBST#	PLX_TBST#	Bus Size	TSIZ[0:3]
De-Asserted	Asserted	64 bits	0000
		32 bits	0100
		16 bits	0010
		8 bits	0001

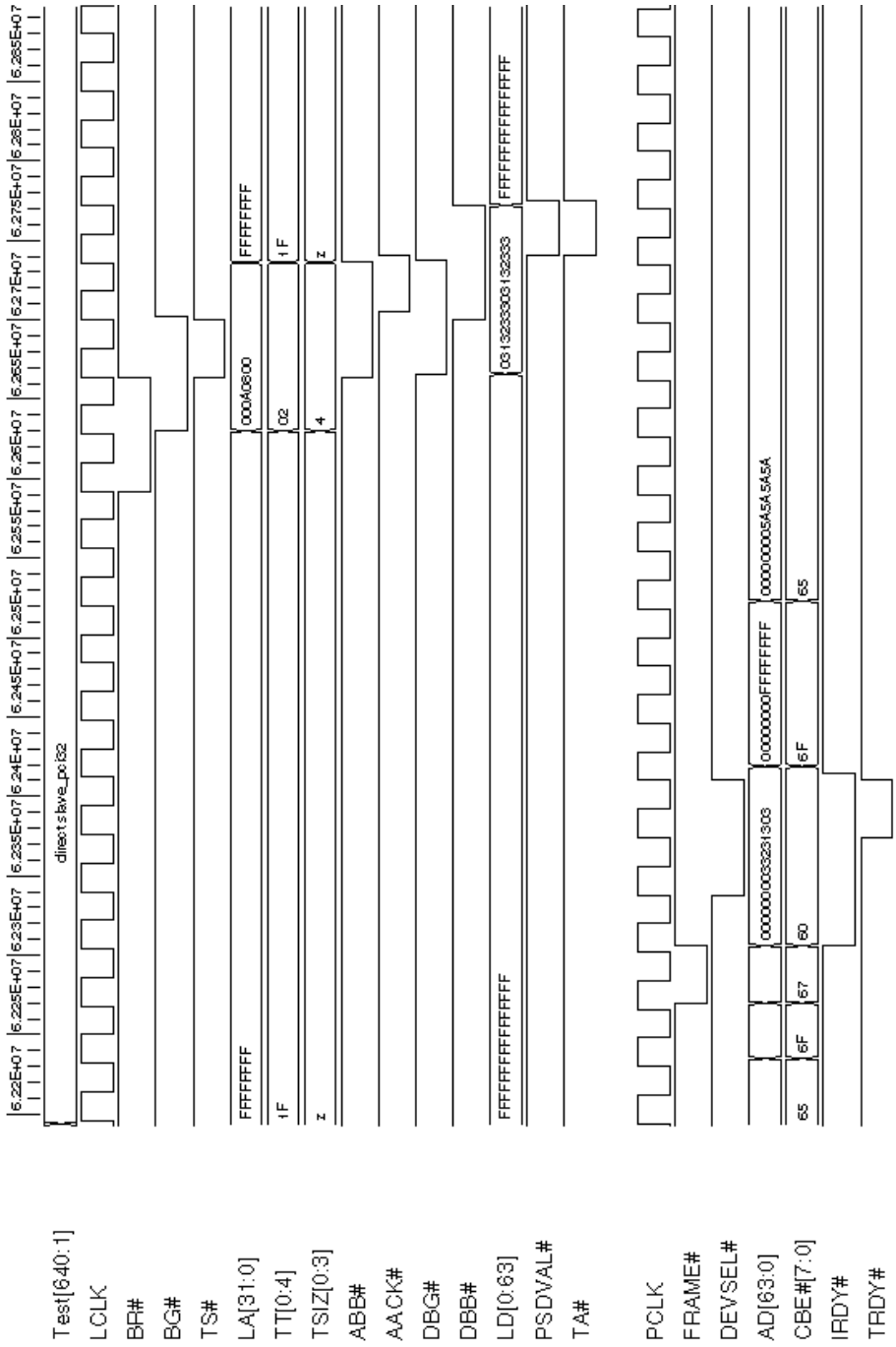
3.7 60X BUS TIMING DIAGRAMS



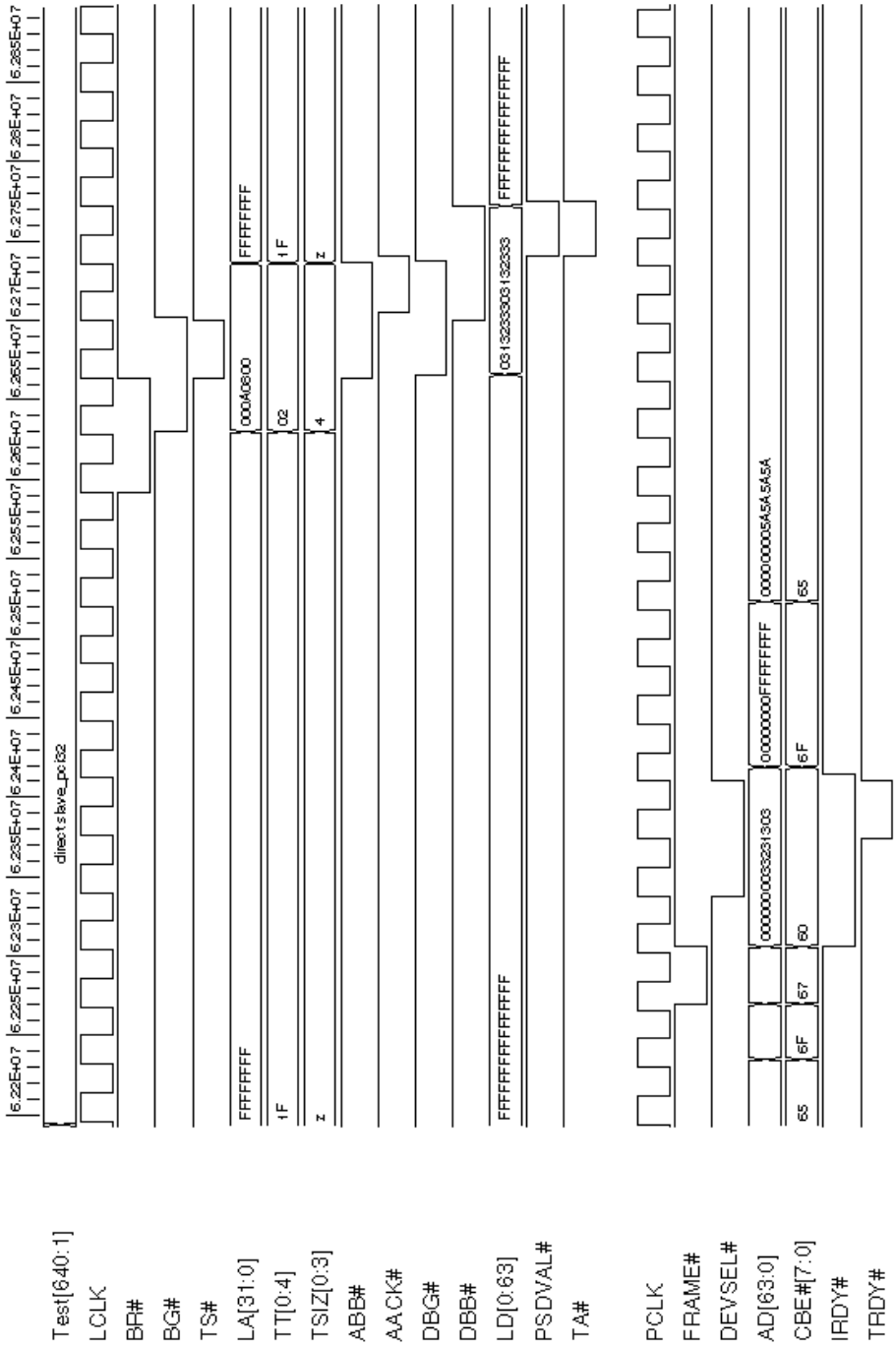
Timing Diagram 3-1. PCI Access to Internal Register, Write followed by Read



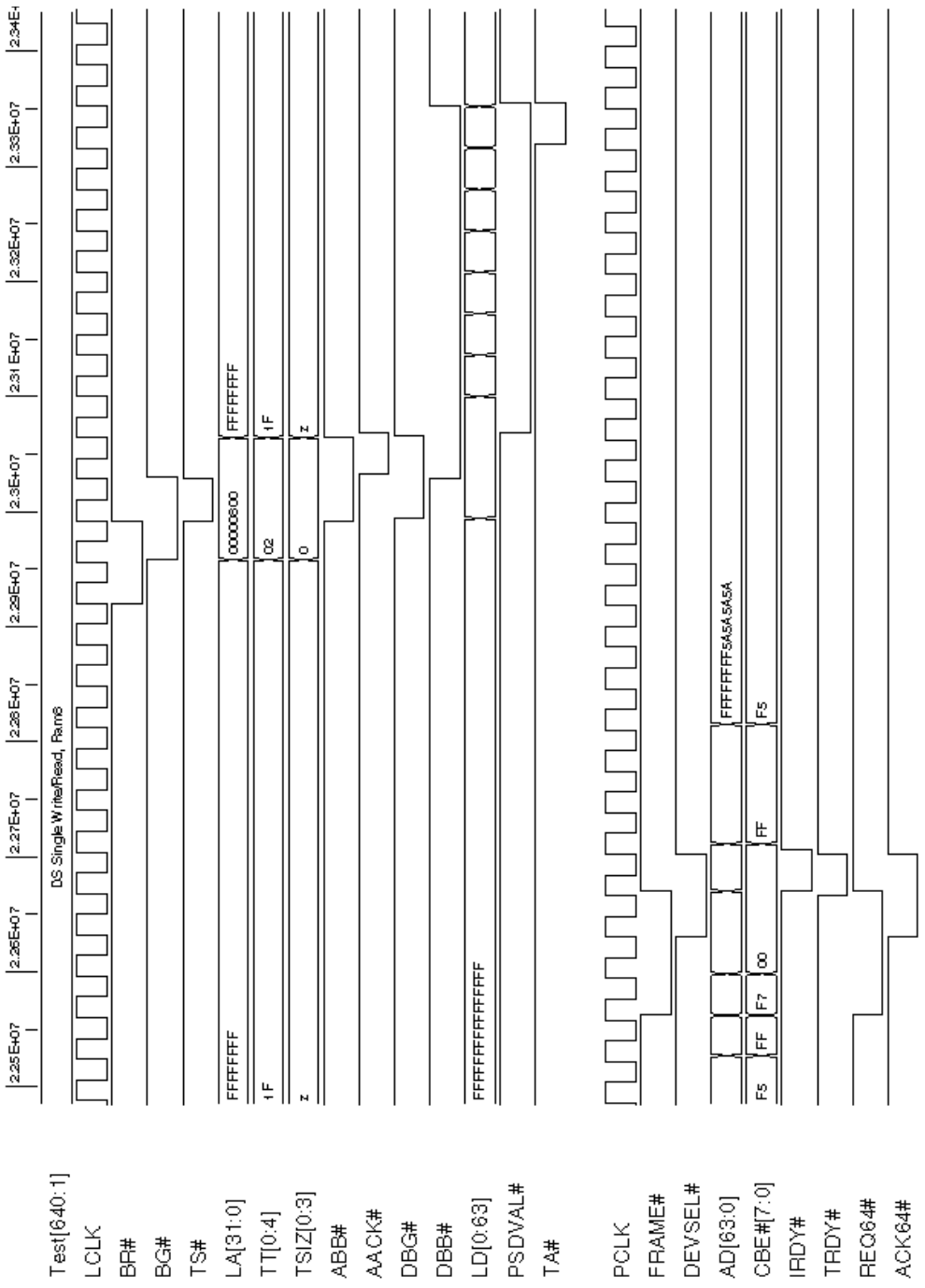
Timing Diagram 3-2. 60x Bus Access to Internal Register, Write followed by Read



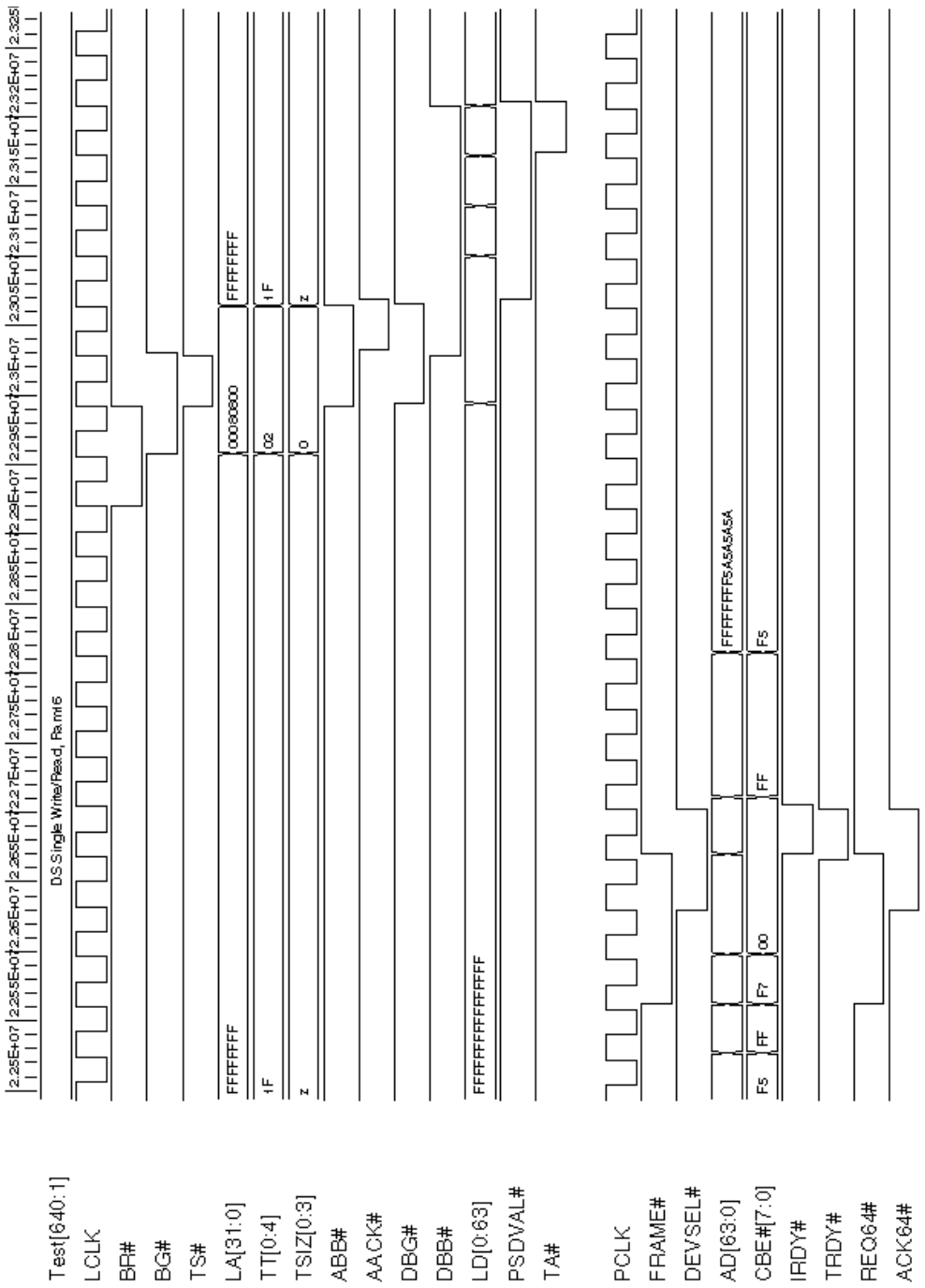
Timing Diagram 3-3. Direct Slave 32-Bit PCI Single-Cycle Write (64-Bit 60x Bus)



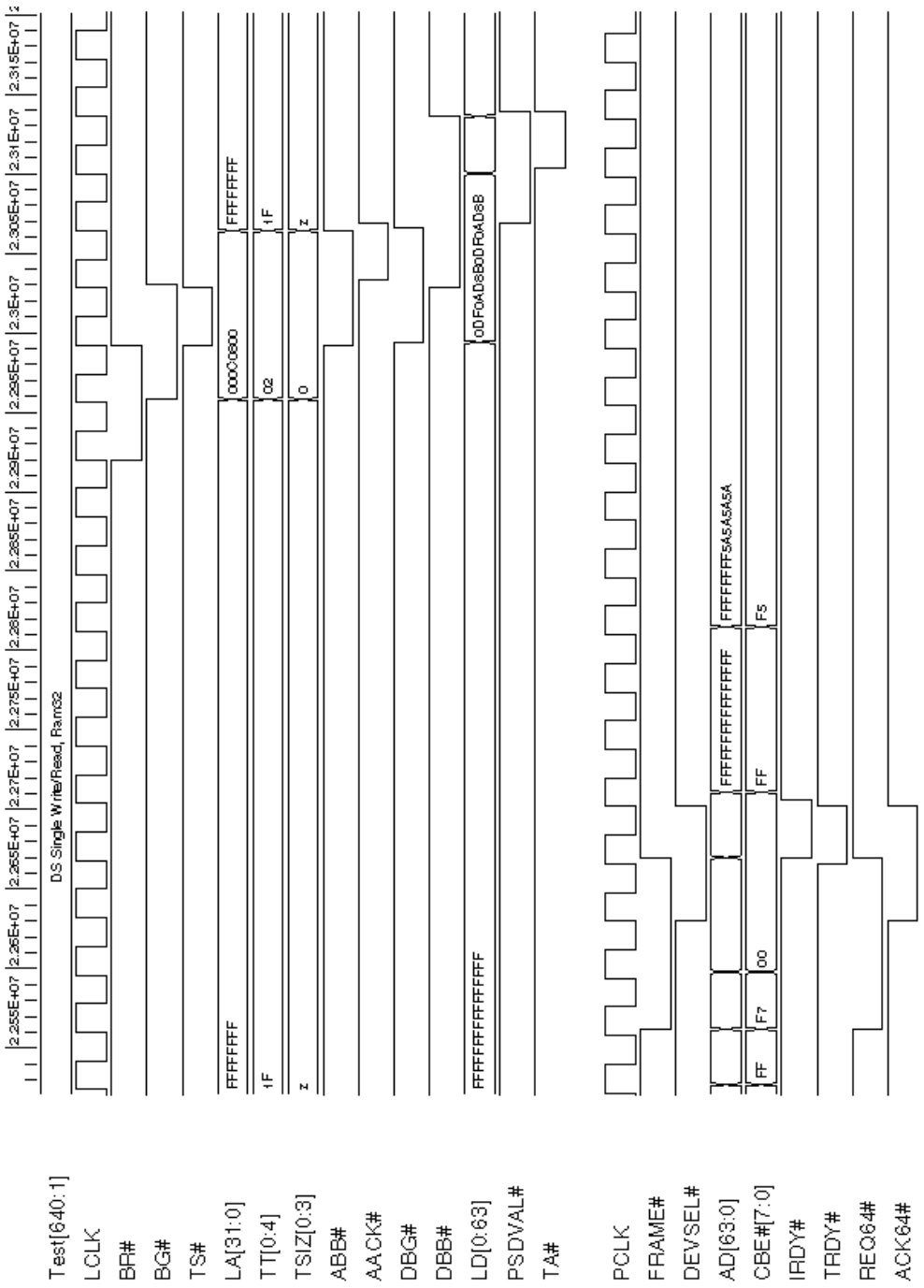
Timing Diagram 3-4. Direct Slave 32-Bit PCI Single-Cycle Read (64-Bit 60x Bus)



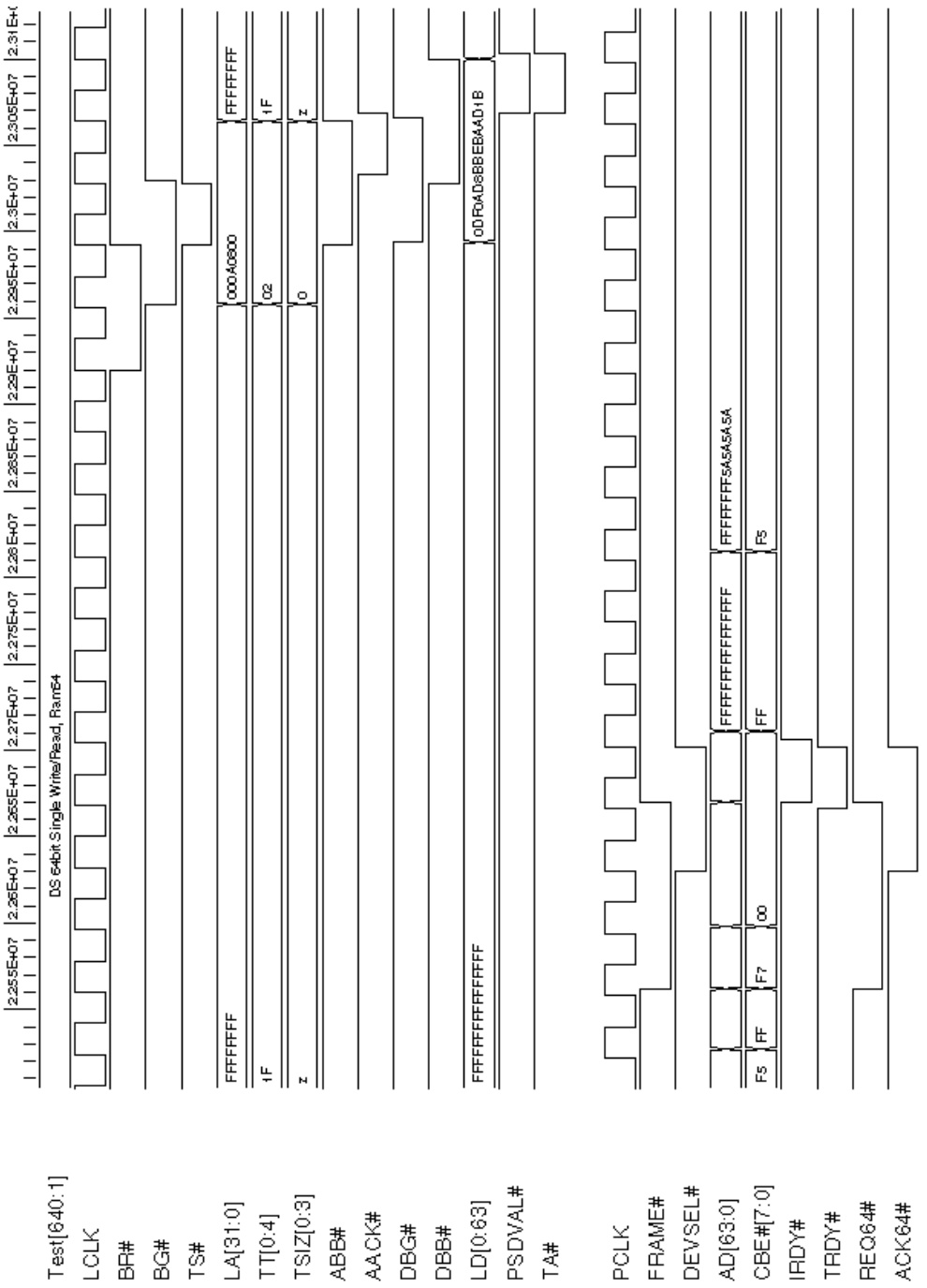
Timing Diagram 3-5. Direct Slave 64-Bit PCI Single-Cycle Write (8-Bit 60x Bus)



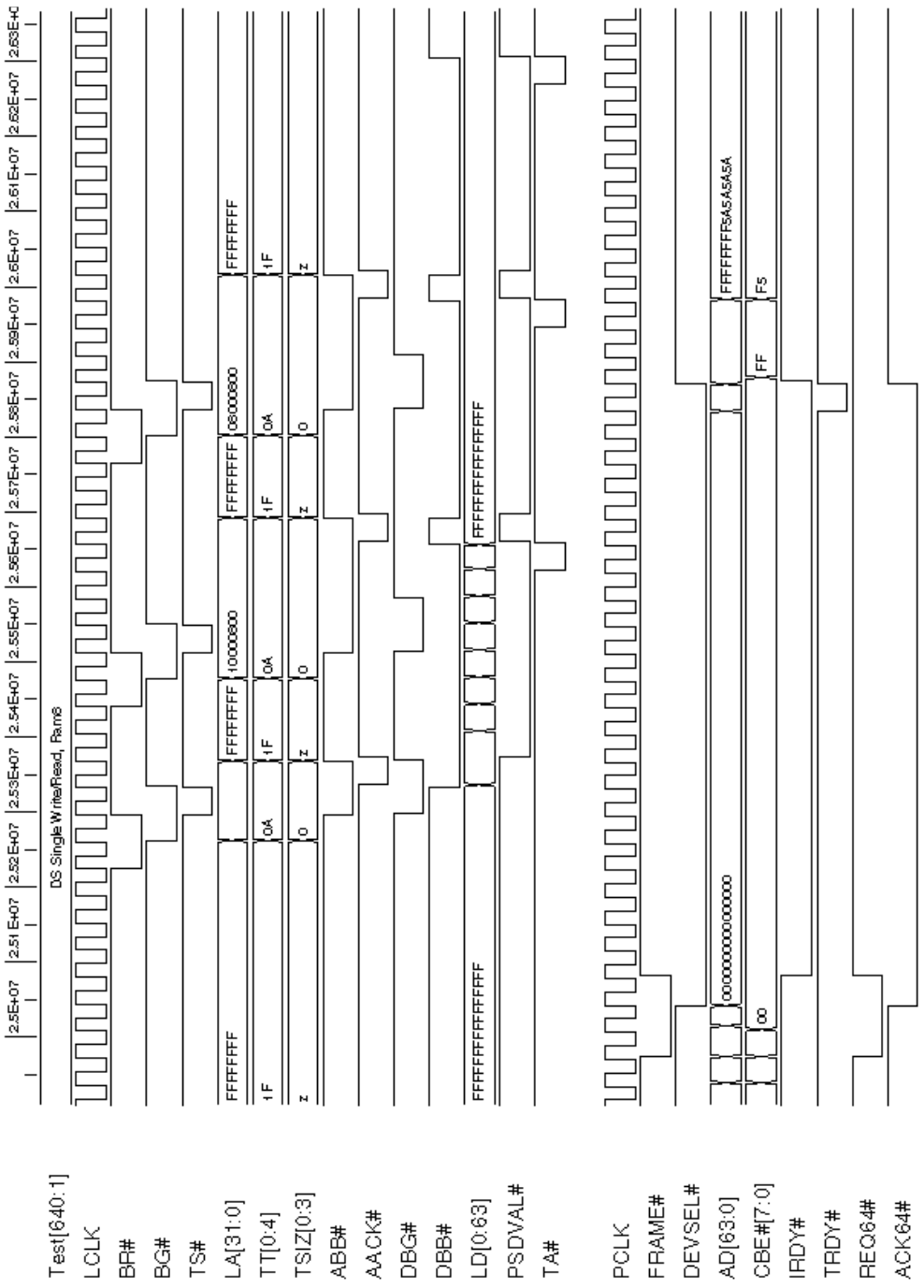
Timing Diagram 3-6. Direct Slave 64-Bit PCI Single-Cycle Write (16-Bit 60x Bus)



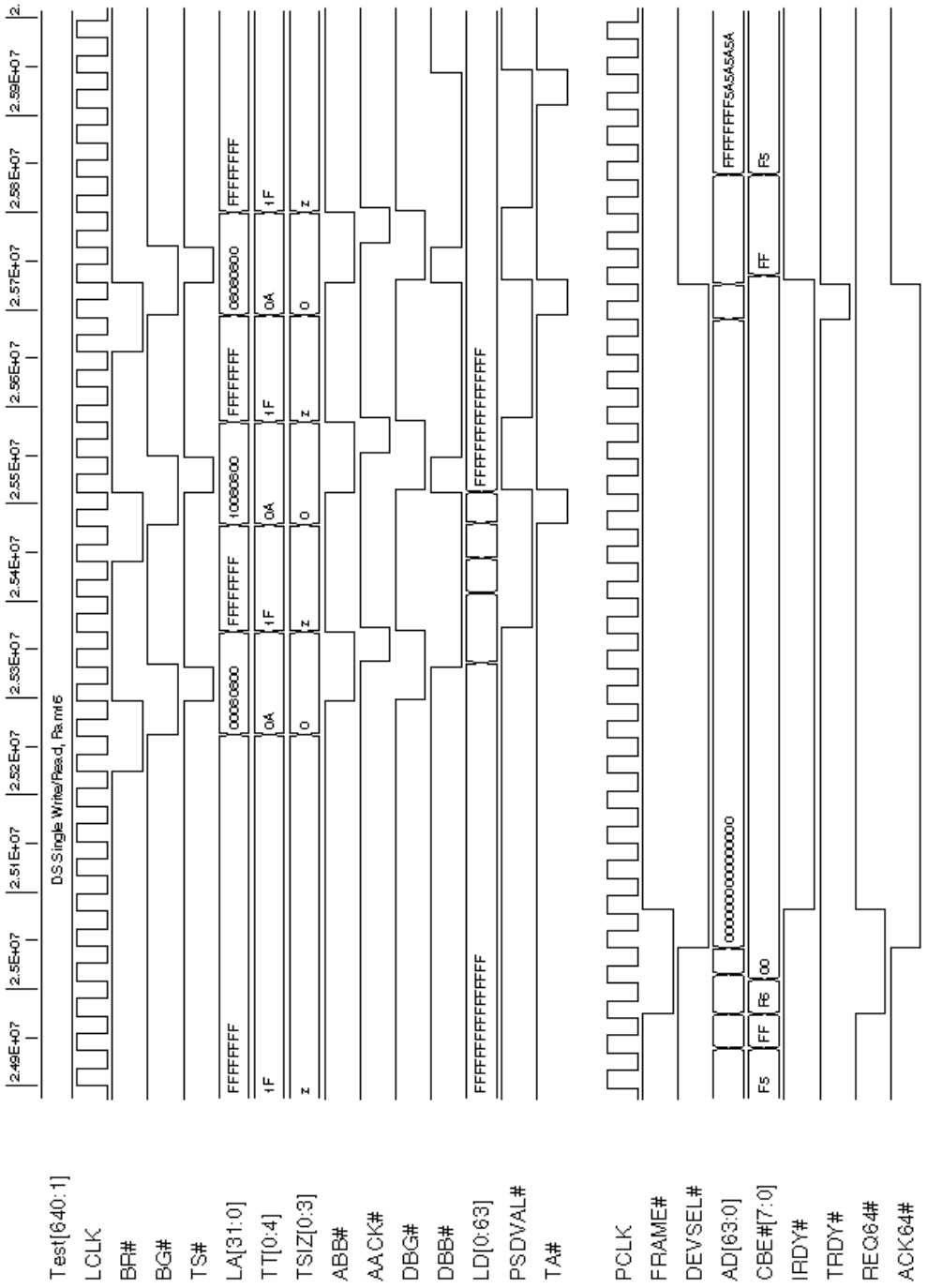
Timing Diagram 3-7. Direct Slave 64-Bit PCI Single-Cycle Write (32-Bit 60x Bus)



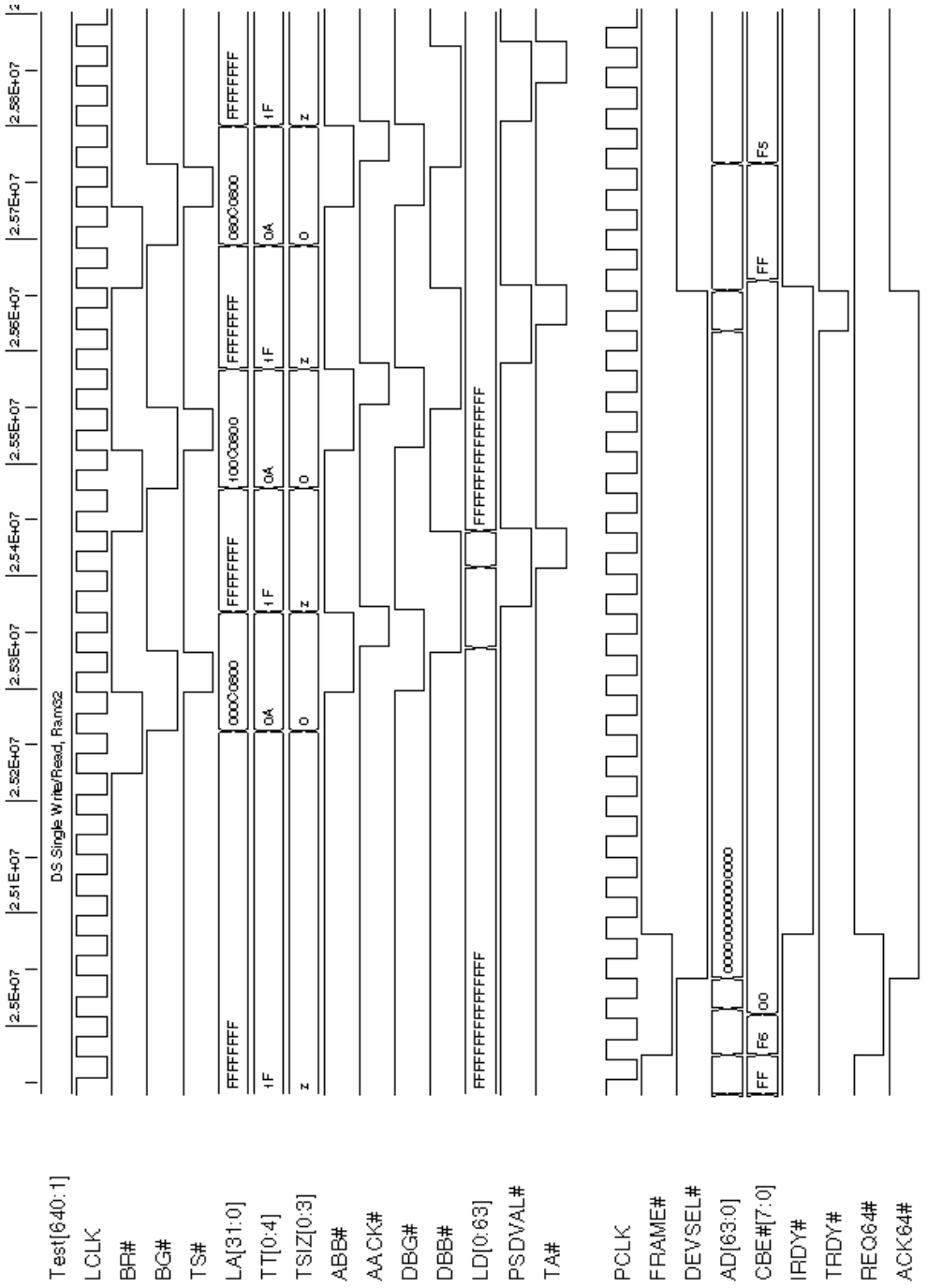
Timing Diagram 3-8. Direct Slave 64-Bit PCI Single-Cycle Write (64-Bit 60x Bus)



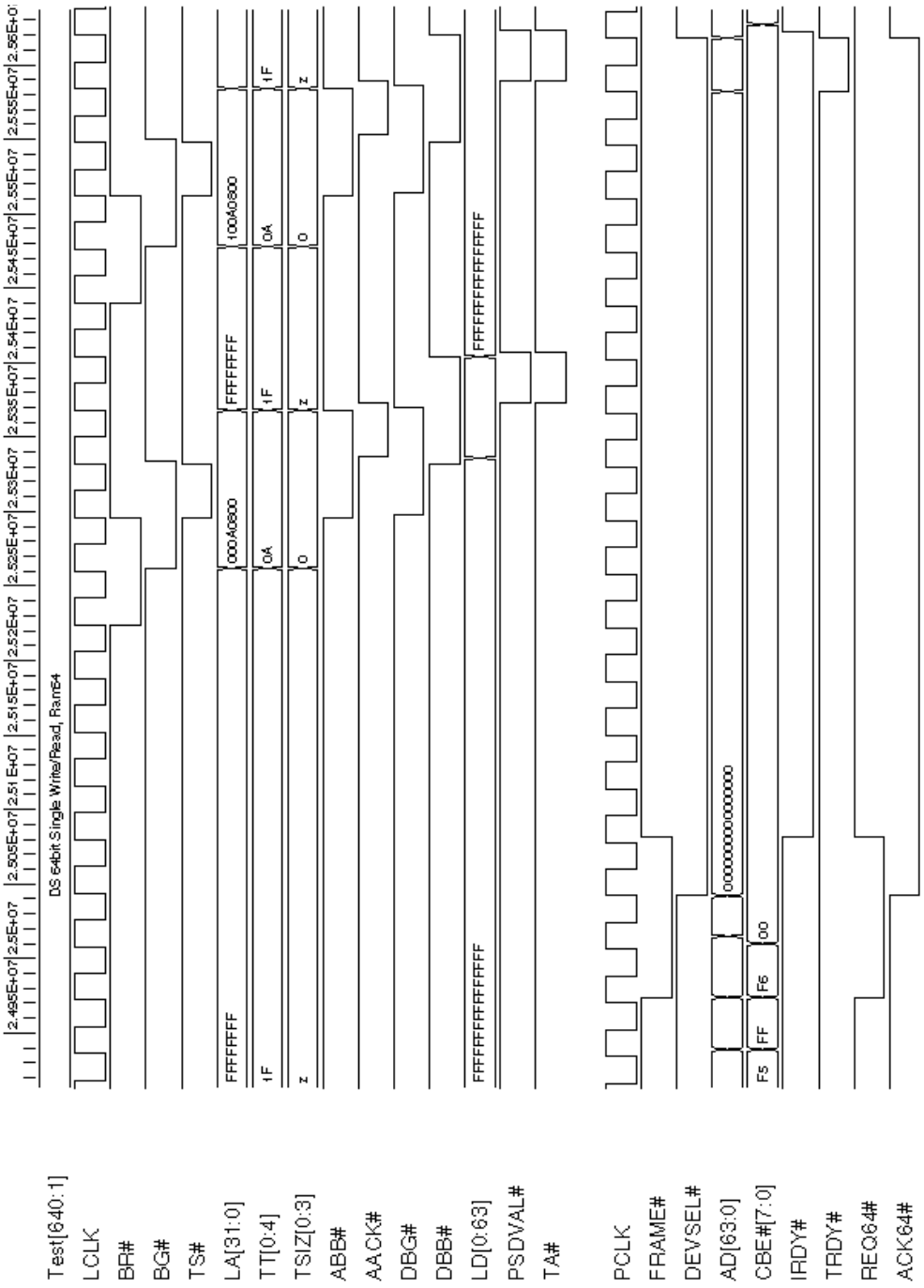
Timing Diagram 3-9. Direct Slave 64-Bit PCI Single-Cycle Read (8-Bit 60x Bus)



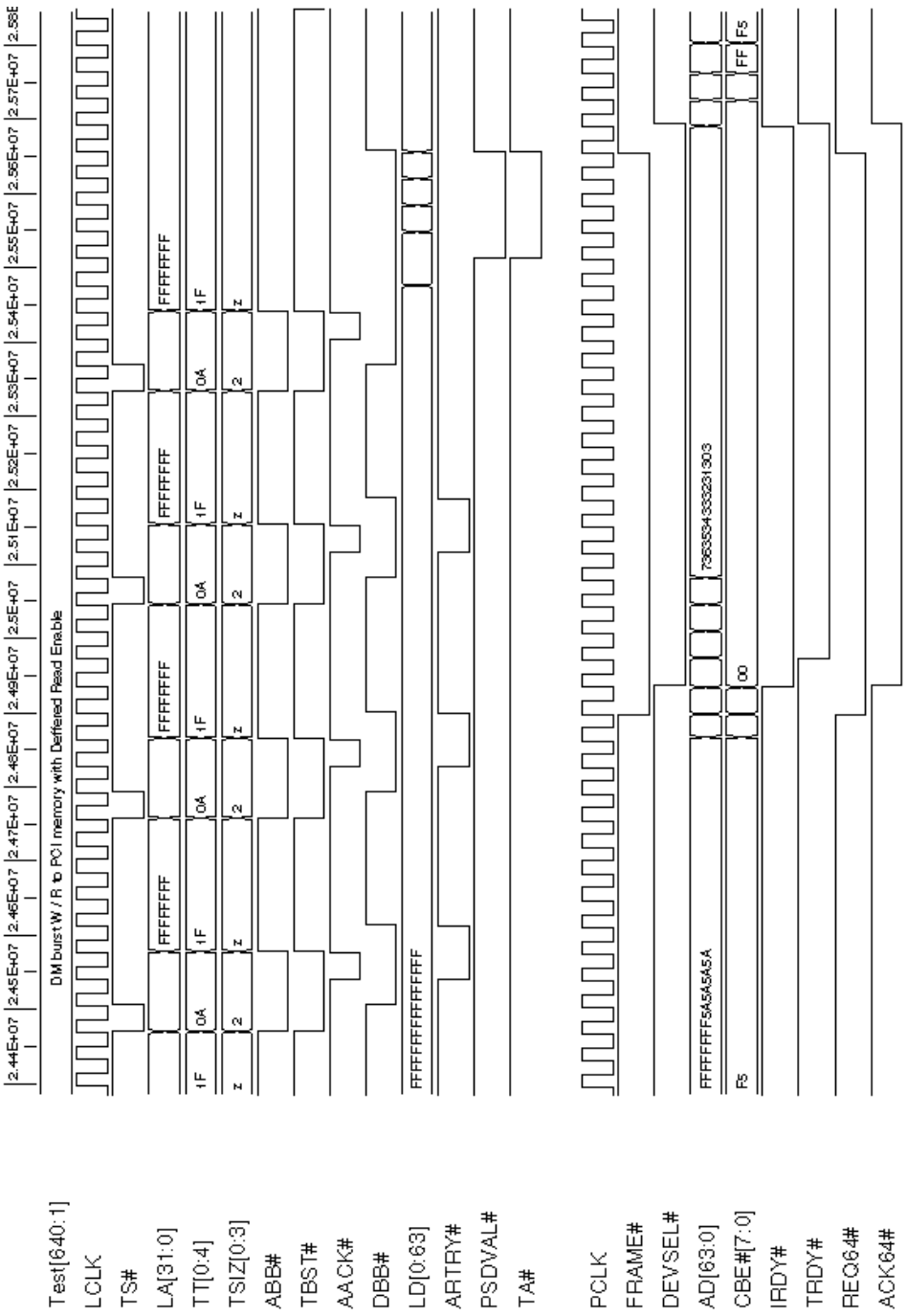
Timing Diagram 3-10. Direct Slave 64-Bit PCI Single-Cycle Read (16-Bit 60x Bus)



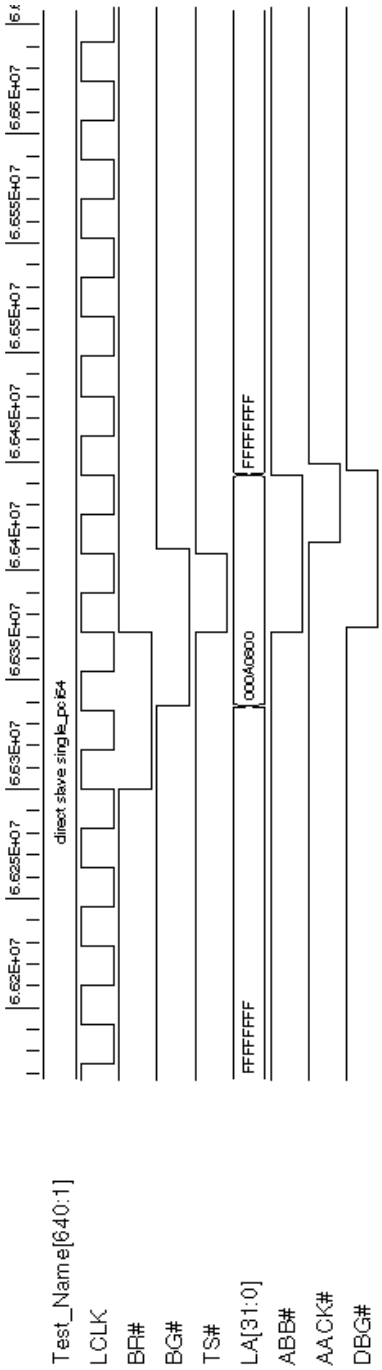
Timing Diagram 3-11. Direct Slave 64-Bit PCI Single-Cycle Read (32-Bit 60x Bus)



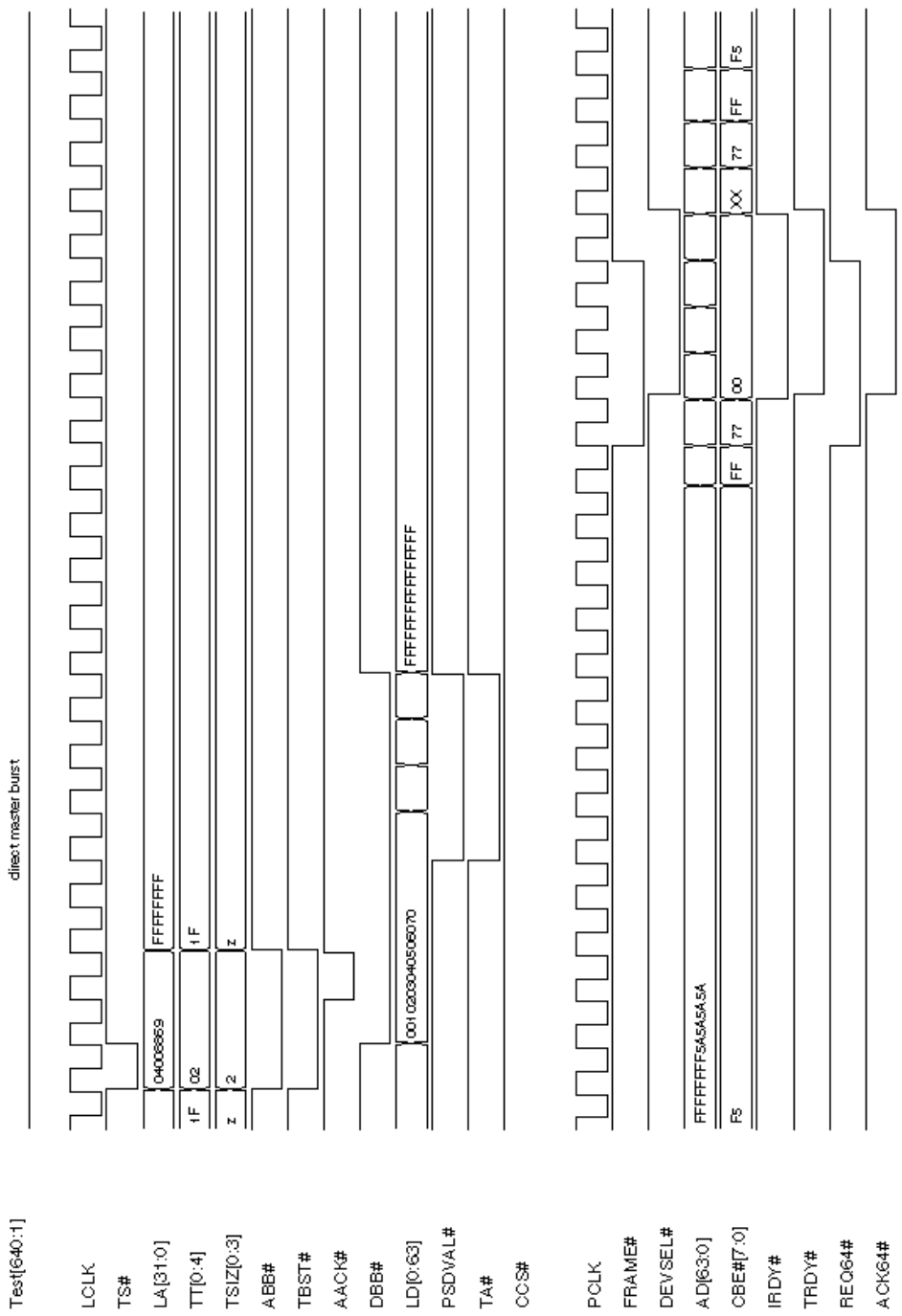
Timing Diagram 3-12. Direct Slave 64-Bit PCI Single-Cycle Read (64-Bit 60x Bus)



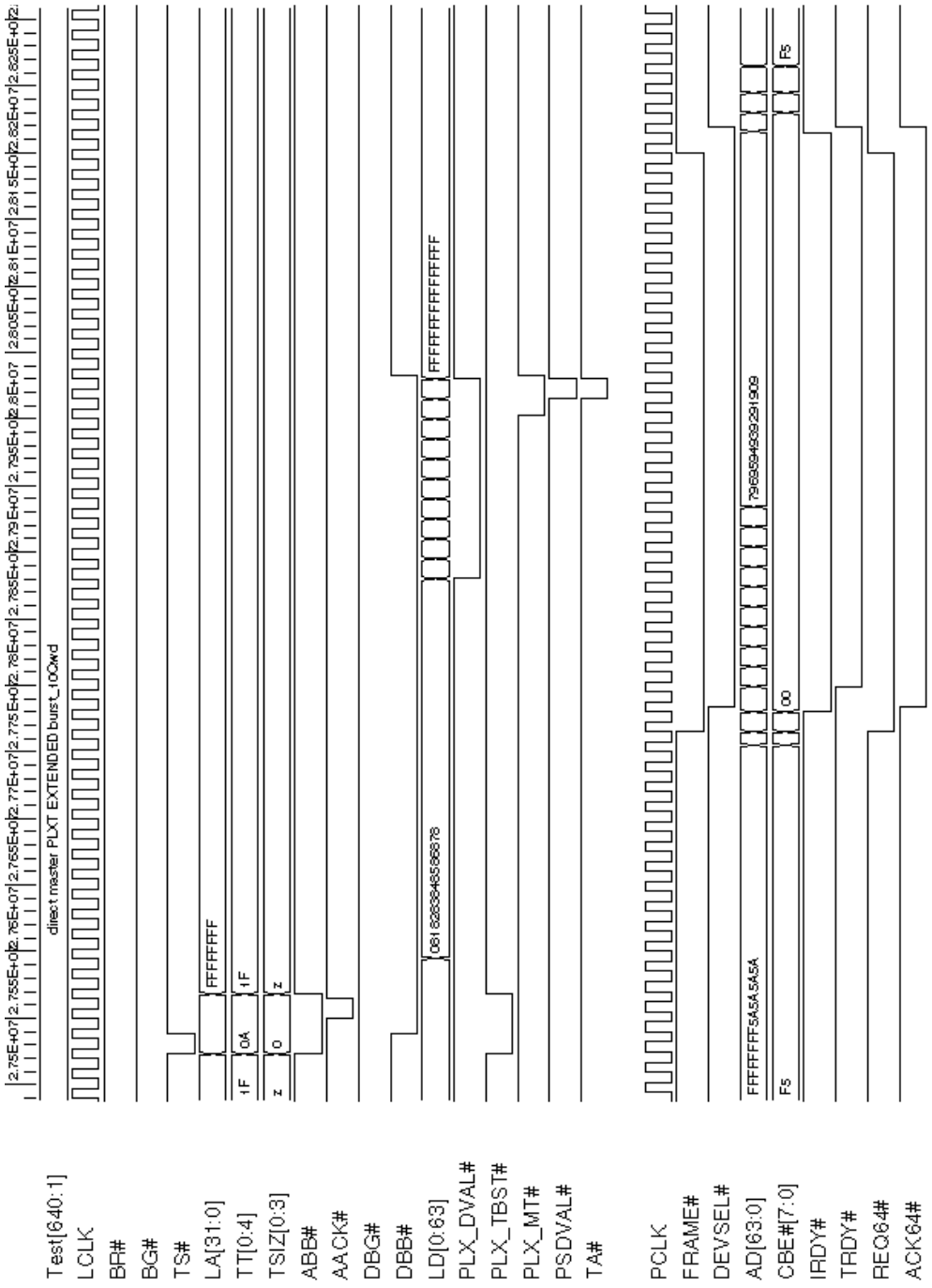
Timing Diagram 3-13. Direct Master Delayed Read Showing ARTRY#



Timing Diagram 3-14. 60x Bus Arbitration

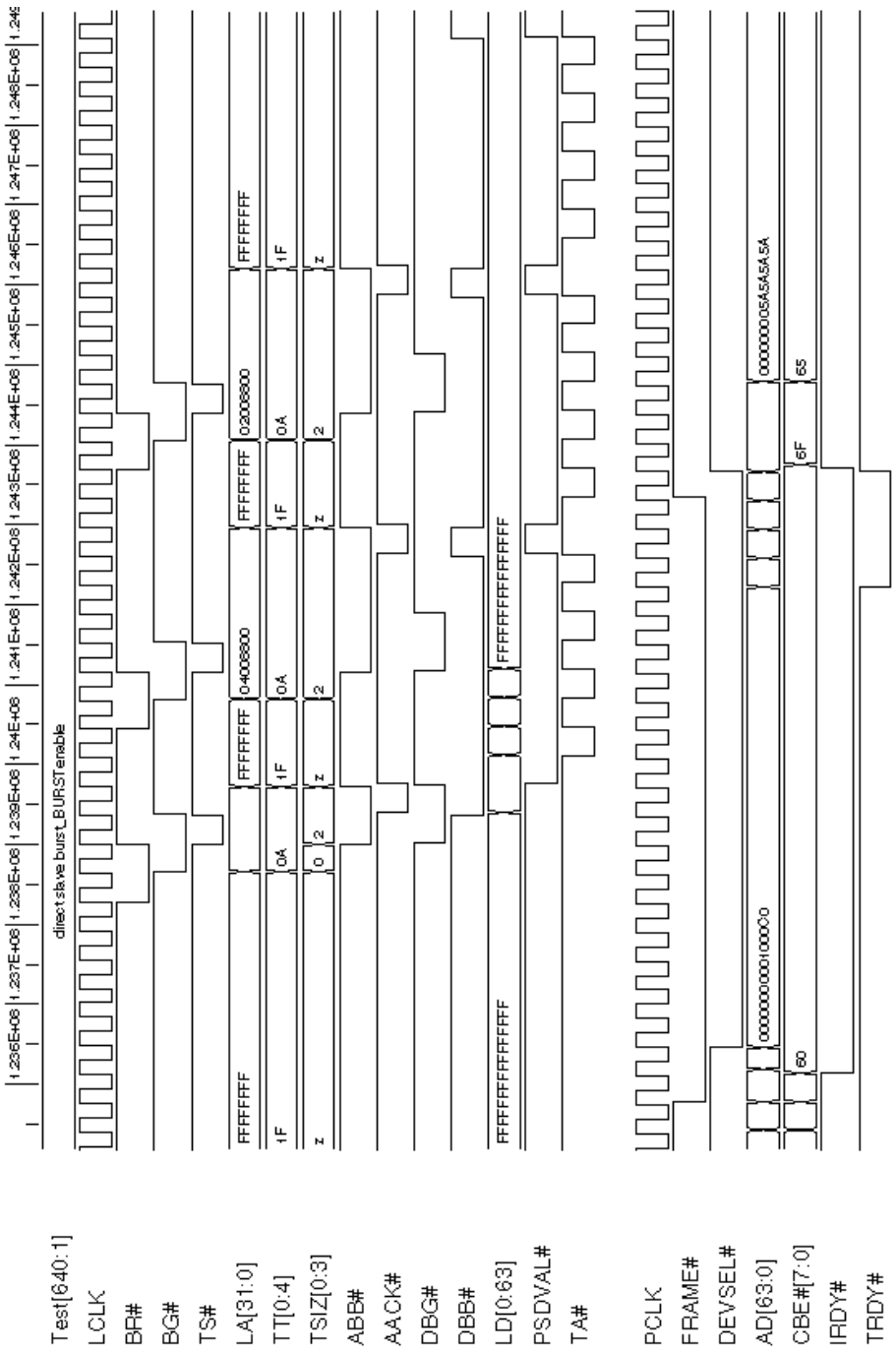


Timing Diagram 3-15. Direct Master Write Burst



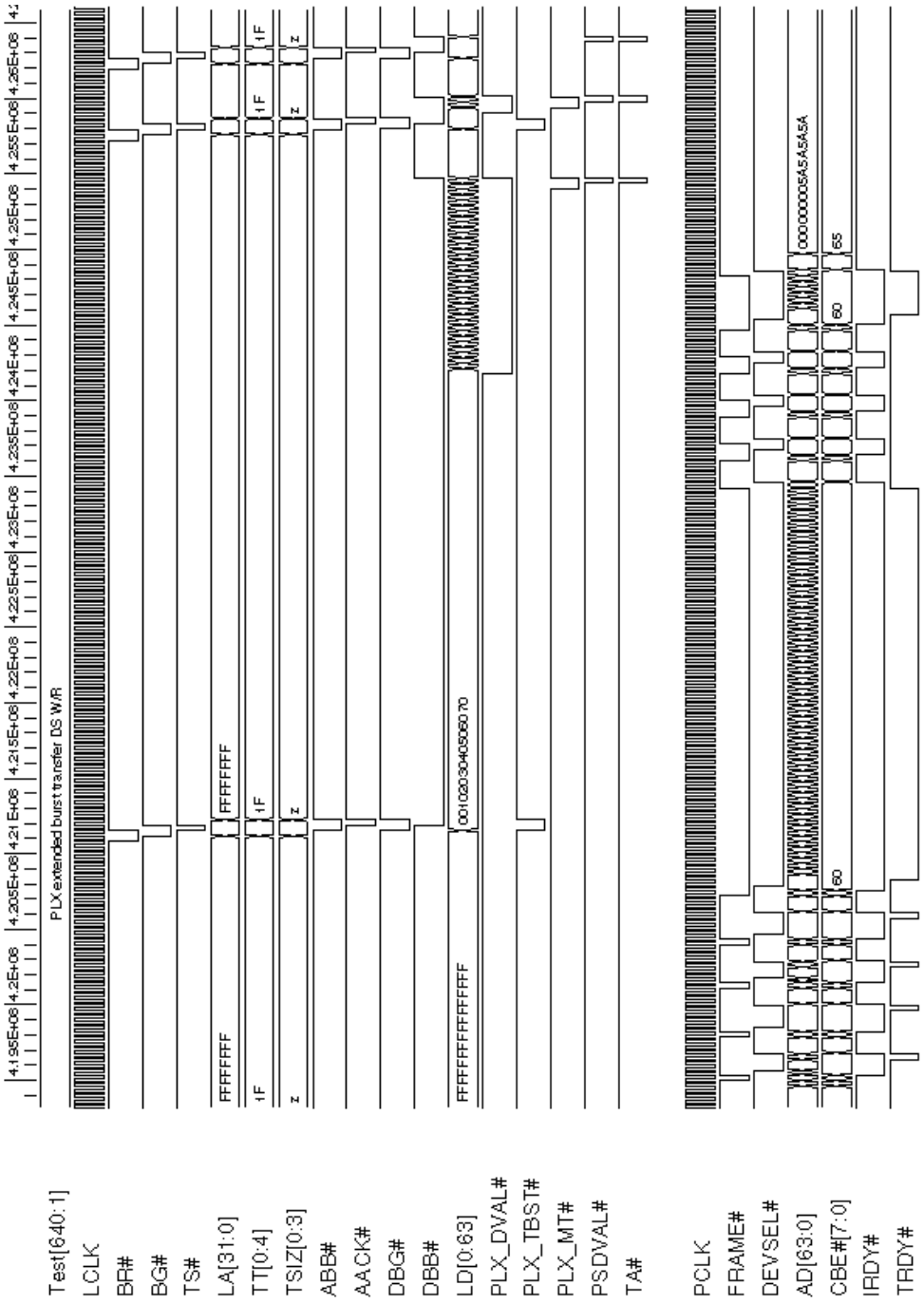
Timing Diagram 3-16. Direct Master Read PLX Extended Burst

Note: *Notice the Mster termination with PLX_MT# asserted,followed one clock later by the assertion of PSDVAL# and TA#.*

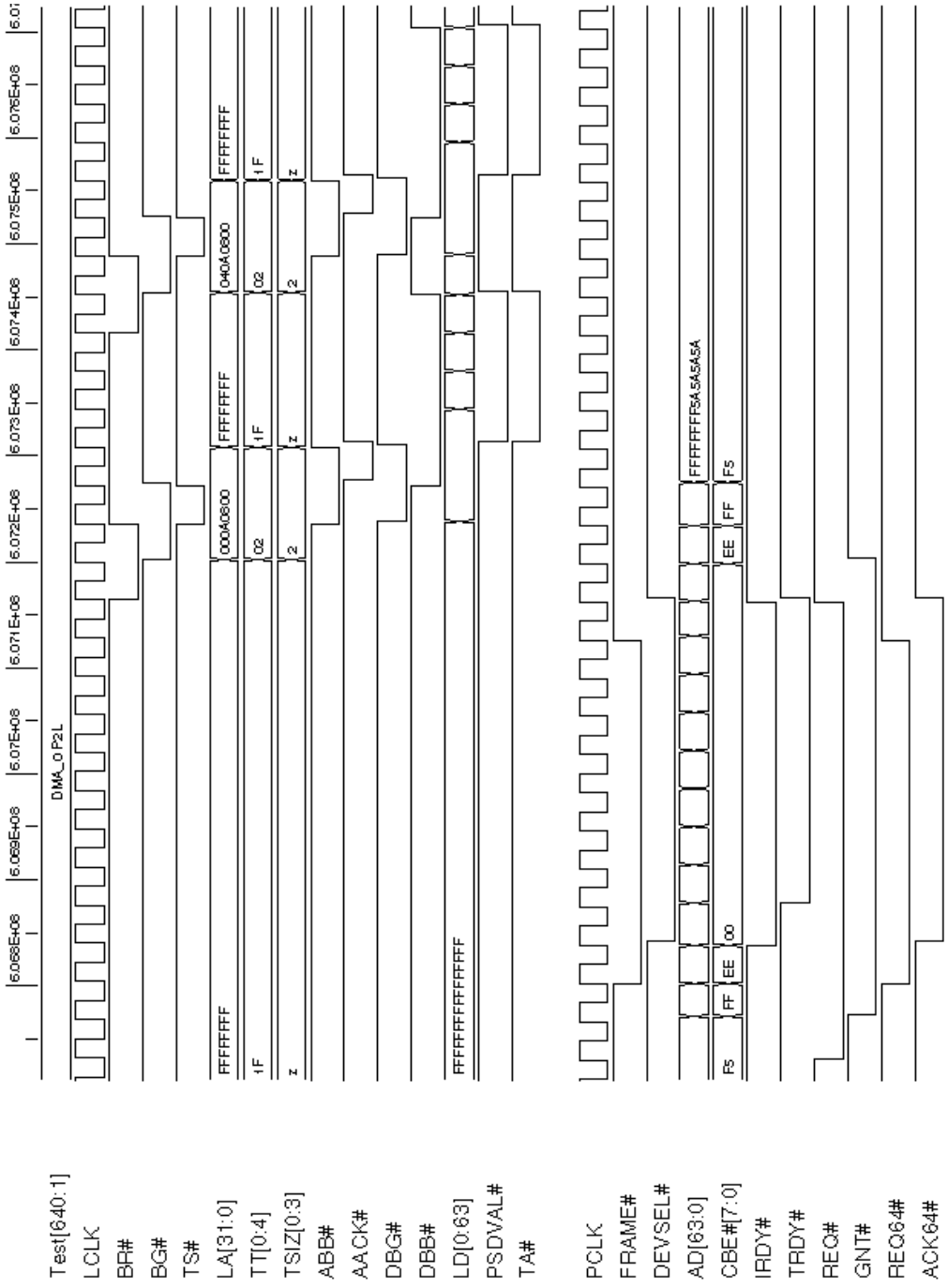


Timing Diagram 3-17. Direct Slave Read Burst

Note: *32-bit 60x Bus, 32-bit PCI Bus (REQ64# and ACK64# de-asserted), prefetch enabled.*

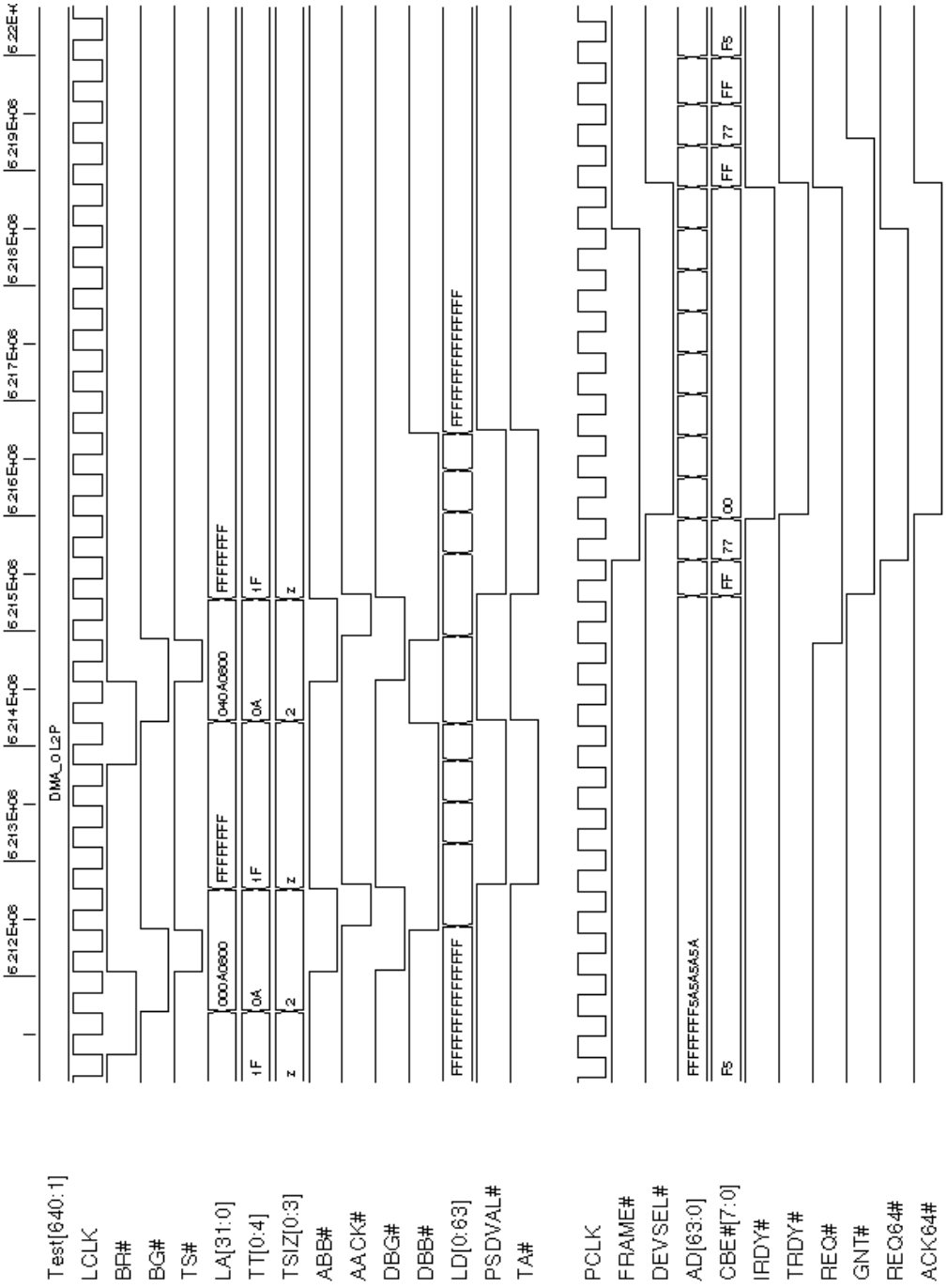


Timing Diagram 3-18. PLX Extended Burst Transfer Direct Slave Write, 32-Bit PCI Bus, Master Terminated

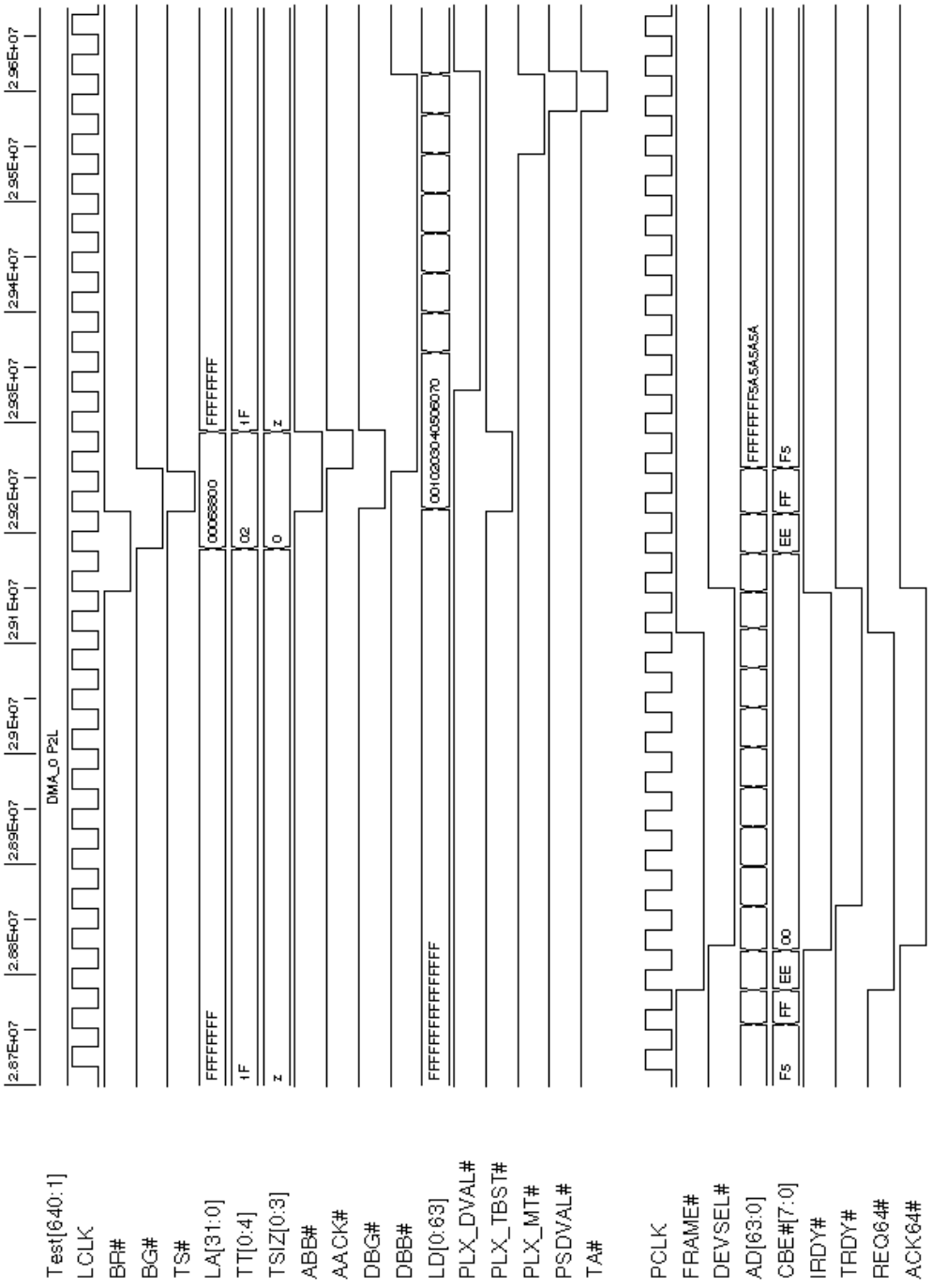


Timing Diagram 3-19. PCI-to-60x, DMA Channel 0

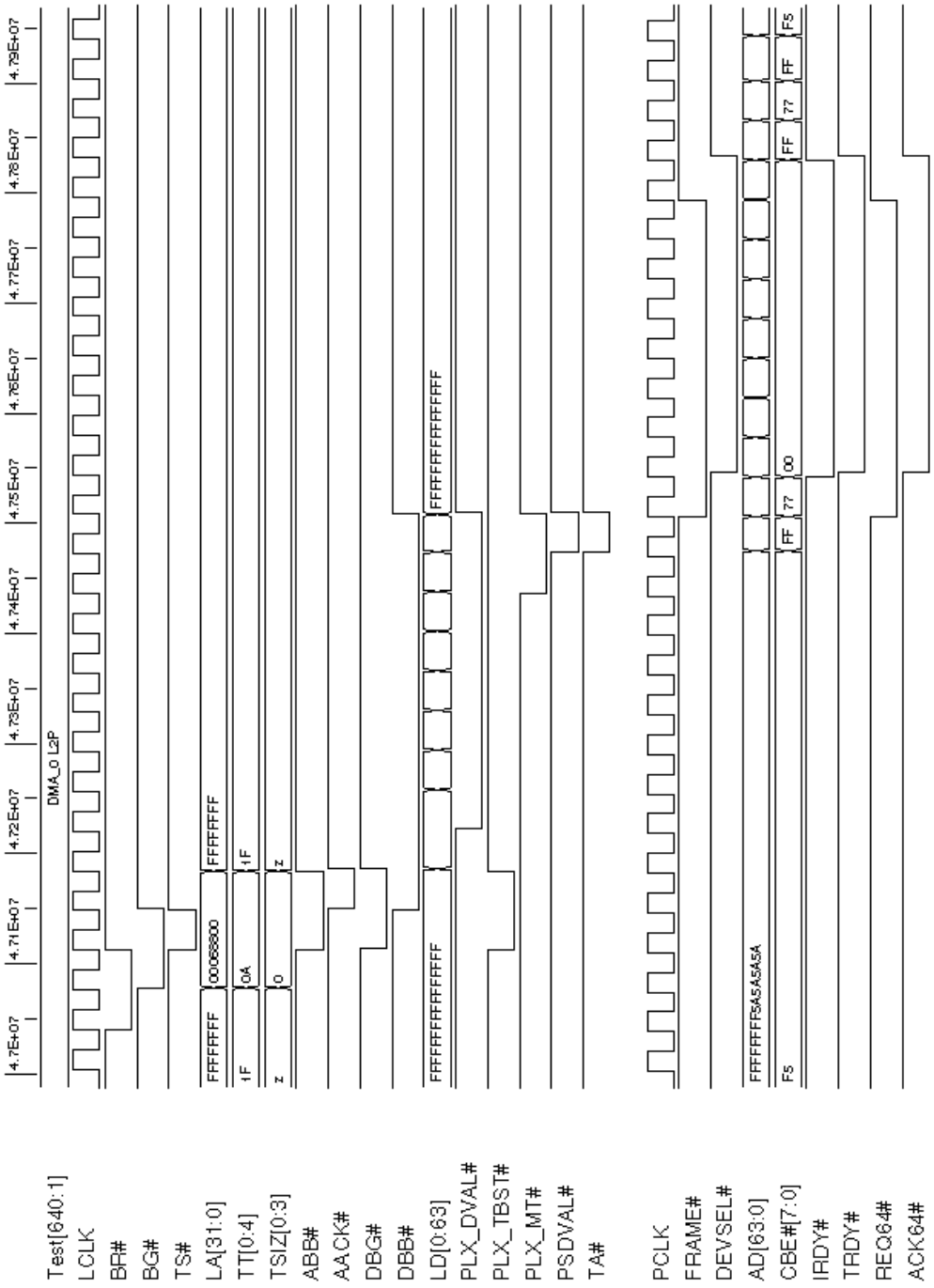
Note: *It is not possible to see which DMA channel is operating from the signals.This information is provided by the DMA descriptors.*



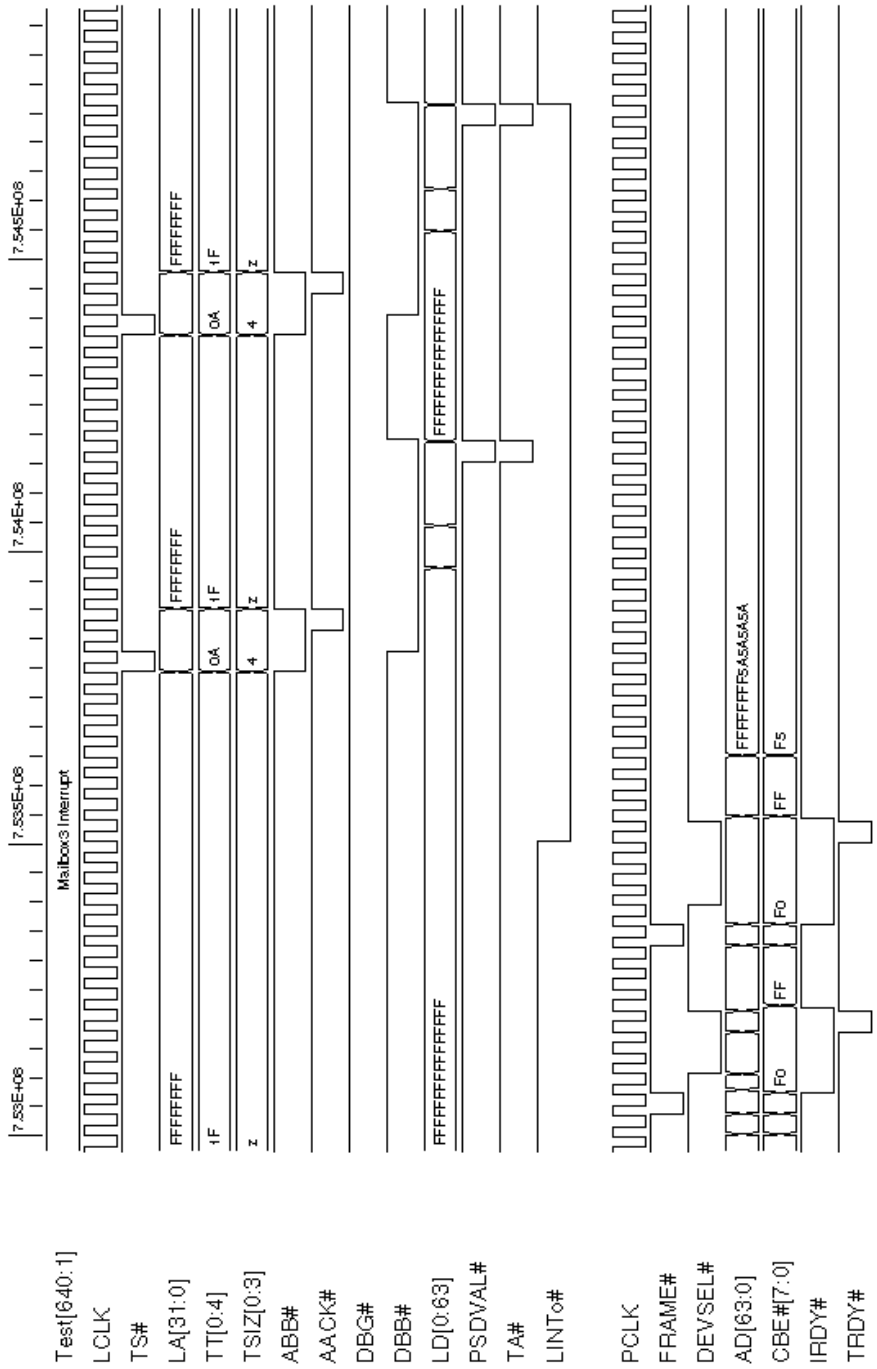
Timing Diagram 3-20. 60x-to-PCI, DMA Channel 0



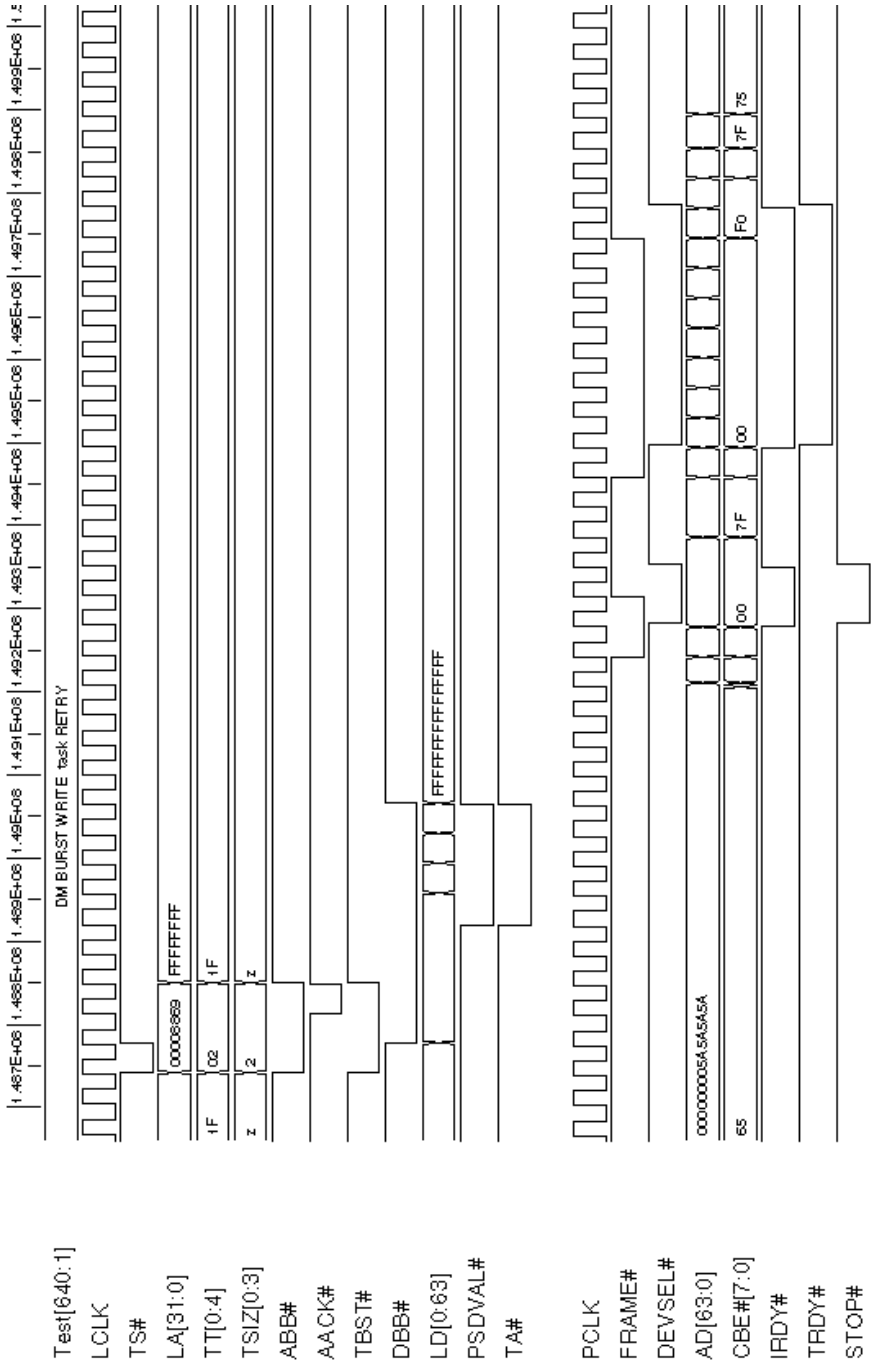
Timing Diagram 3-21. PCI-to-PLX Extended DMA



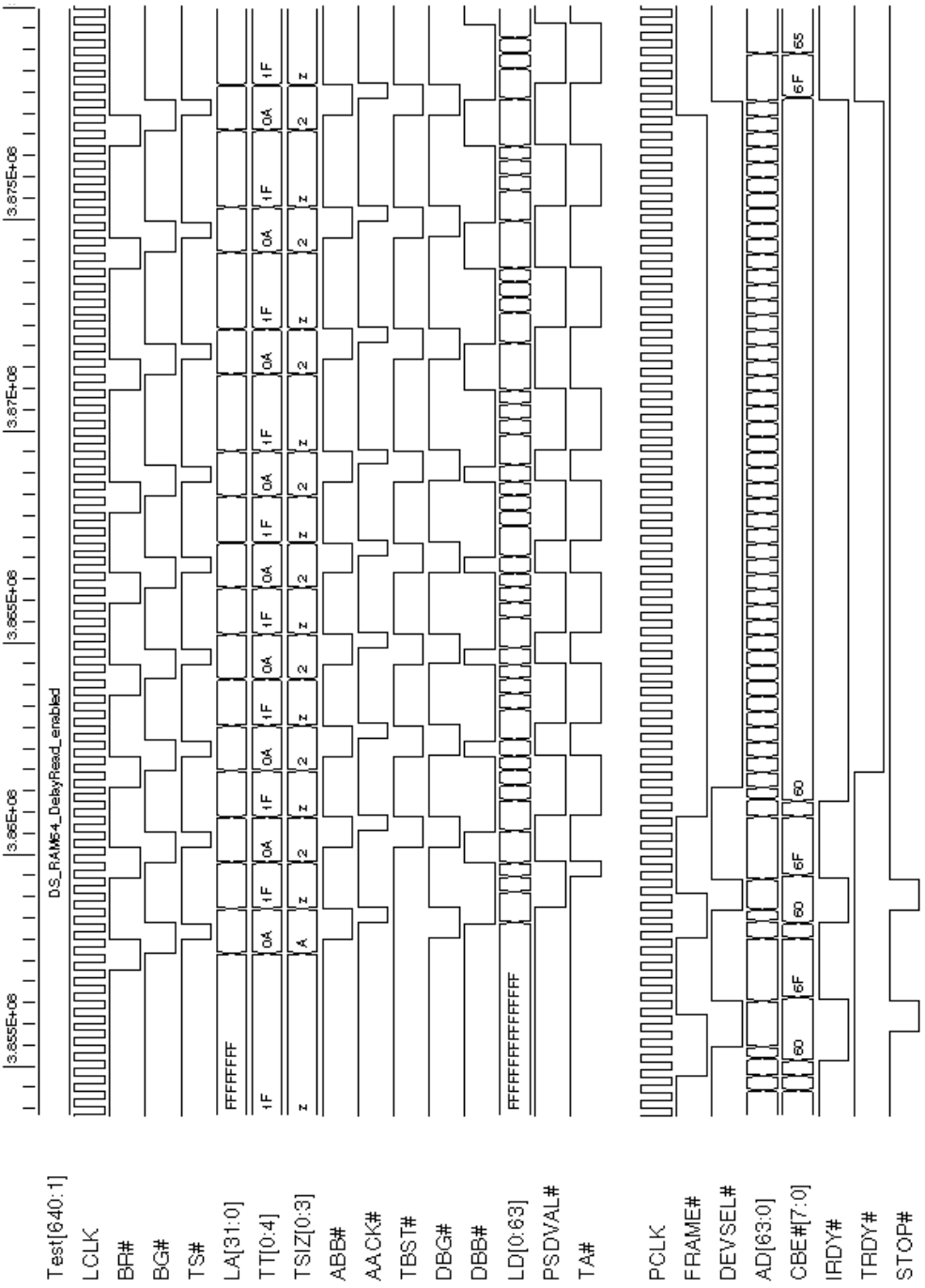
Timing Diagram 3-22. PLX Extended-to-PCI DMA



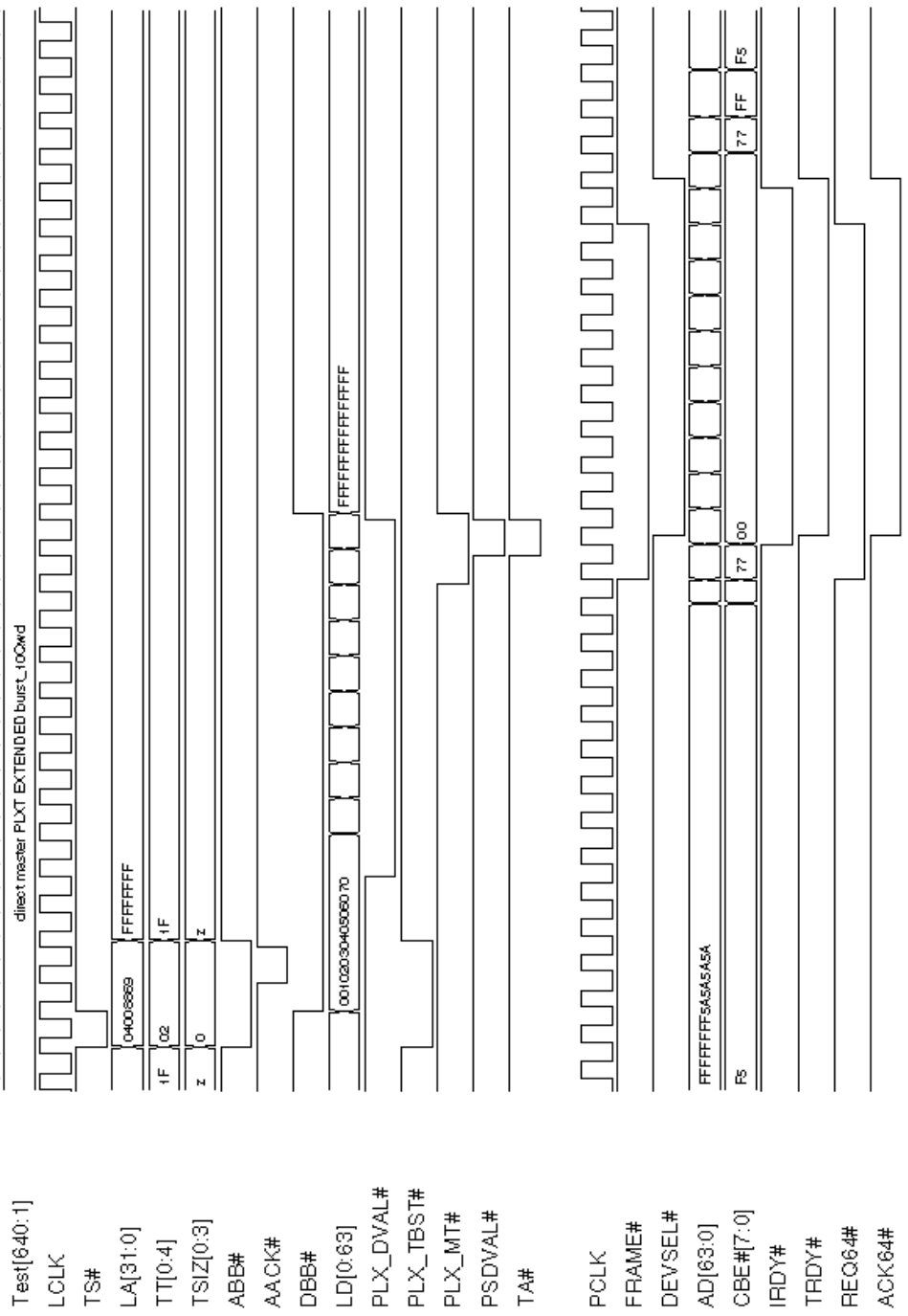
Timing Diagram 3-23. Mailbox 3 LINTo# Interrupt



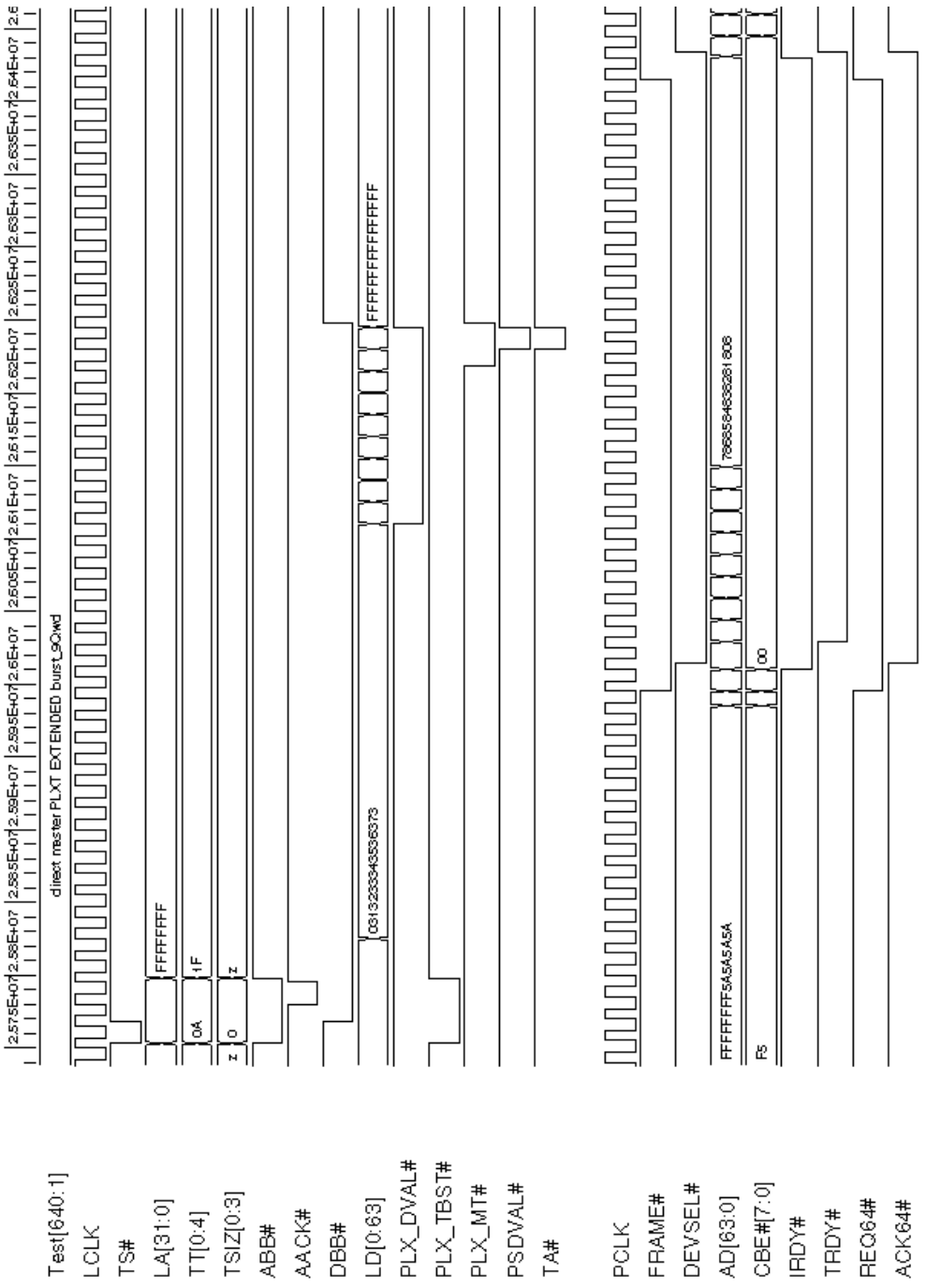
Timing Diagram 3-24. Direct Master Burst with PCI Retry



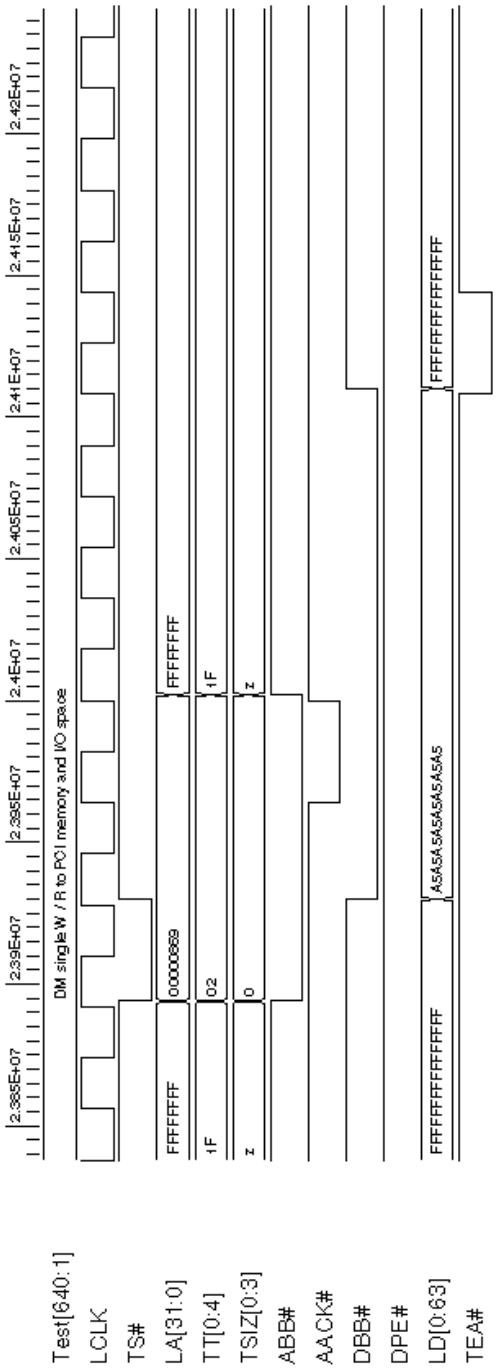
Timing Diagram 3-25. Direct Slave Delayed Read



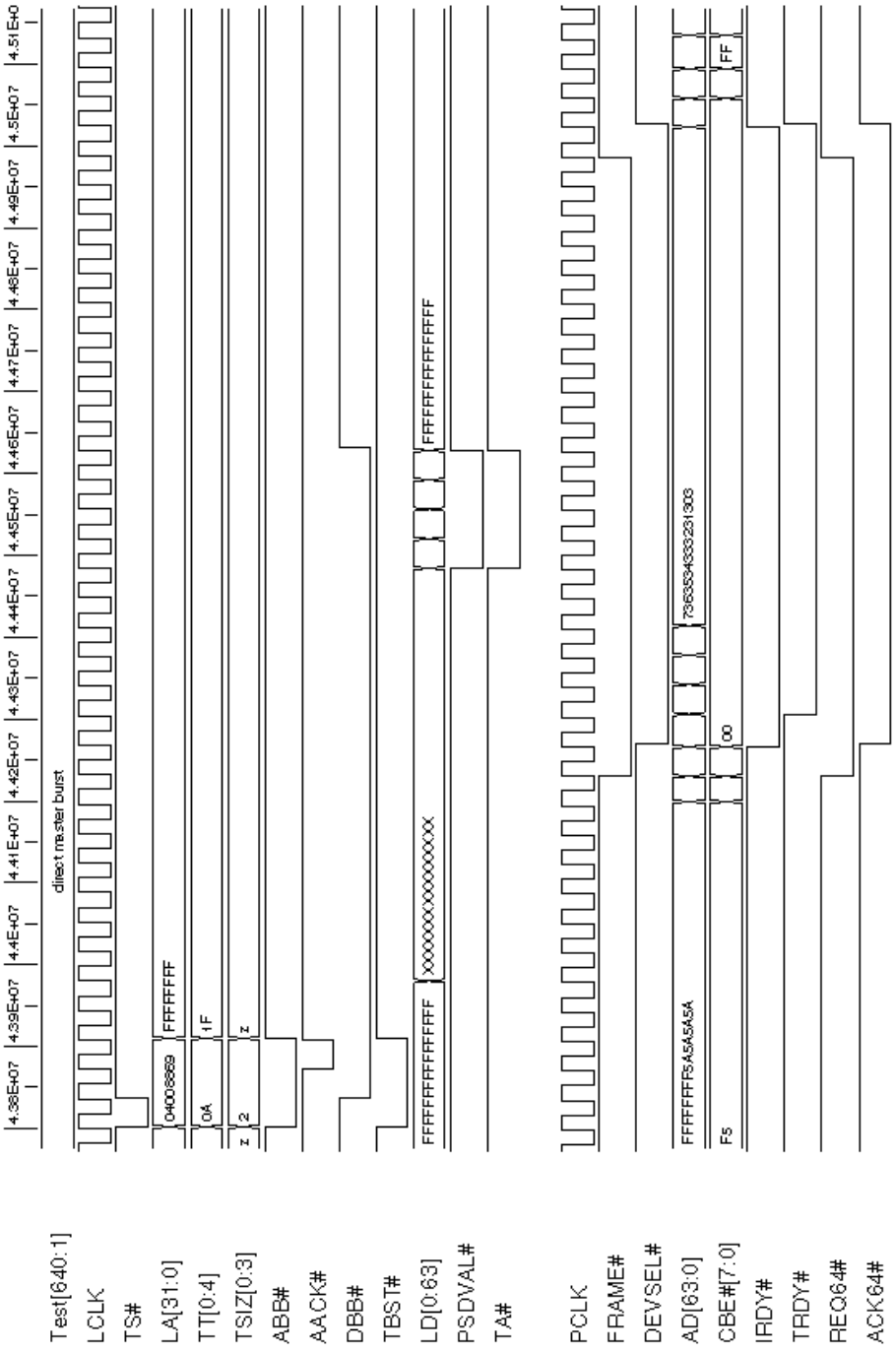
Timing Diagram 3-26. PLX Extended Burst Direct Master Write, 64-Bit PCI Bus, 10 Qwords, Master Terminated



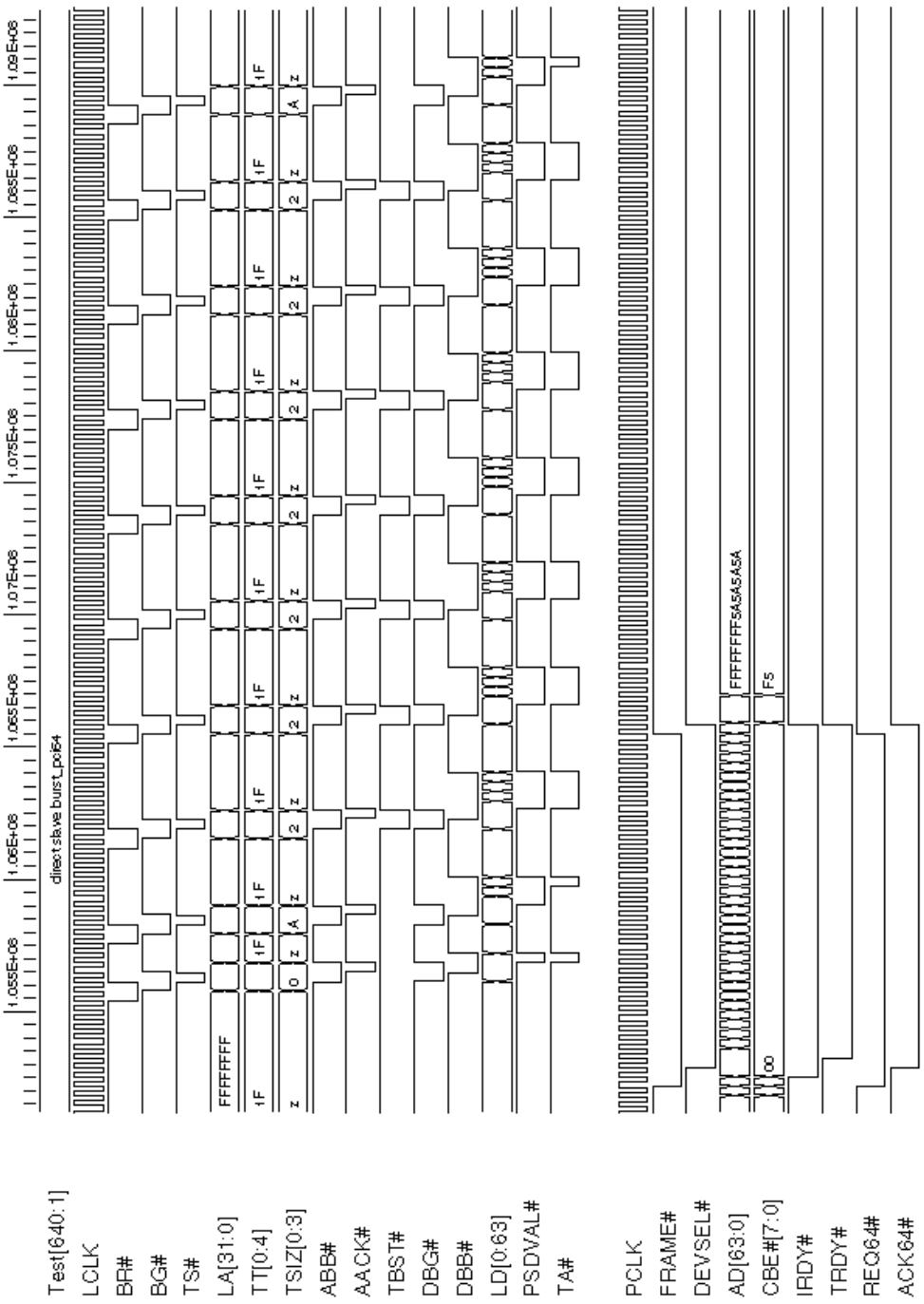
Timing Diagram 3-27. PLX Extended Burst Direct Master Read, 64-Bit PCI Bus, 9 Qwords, Master Terminated



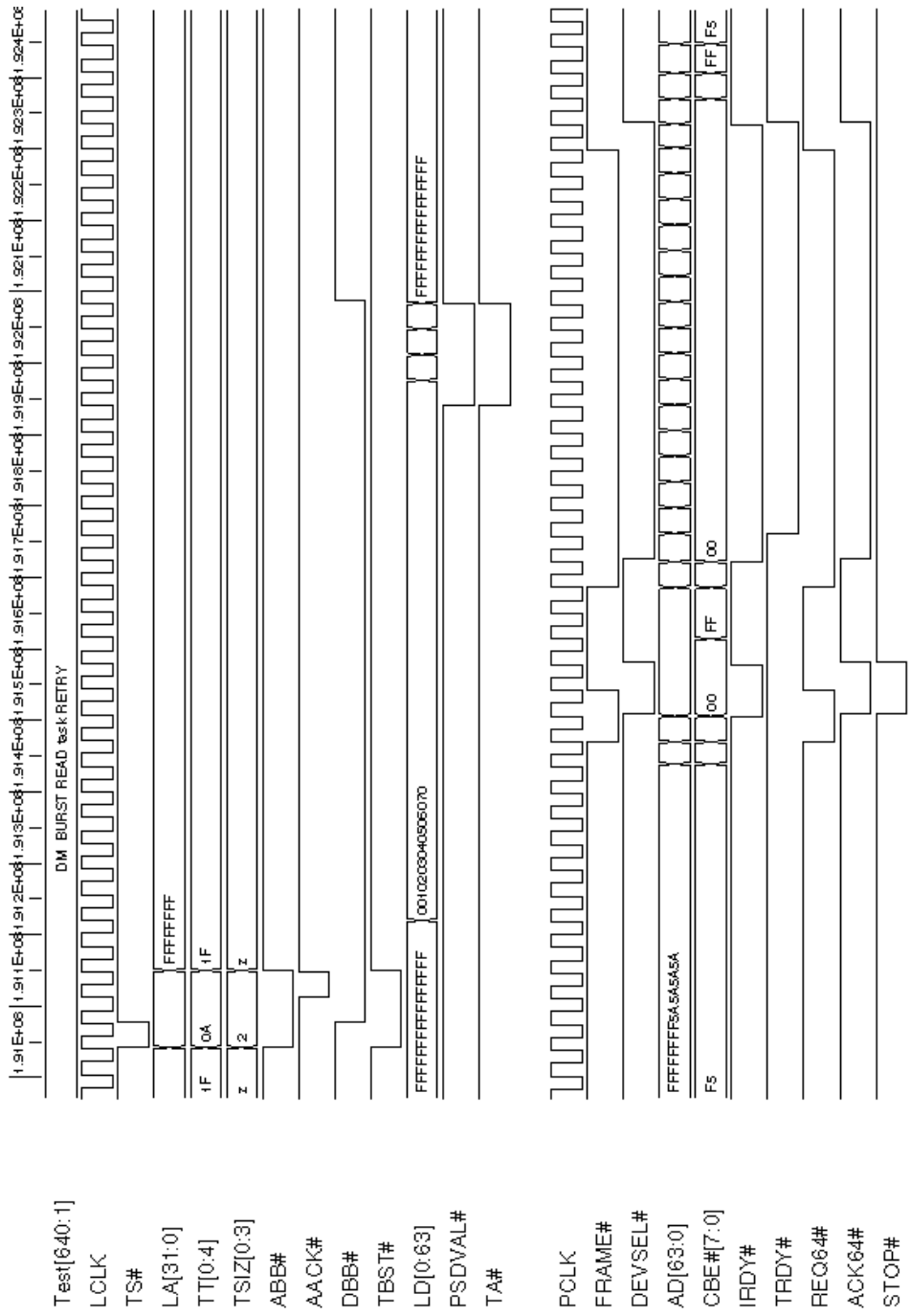
Timing Diagram 3-28. Attempted 60x Bus Write of 64 Bits to PCI I/O Showing TEA#



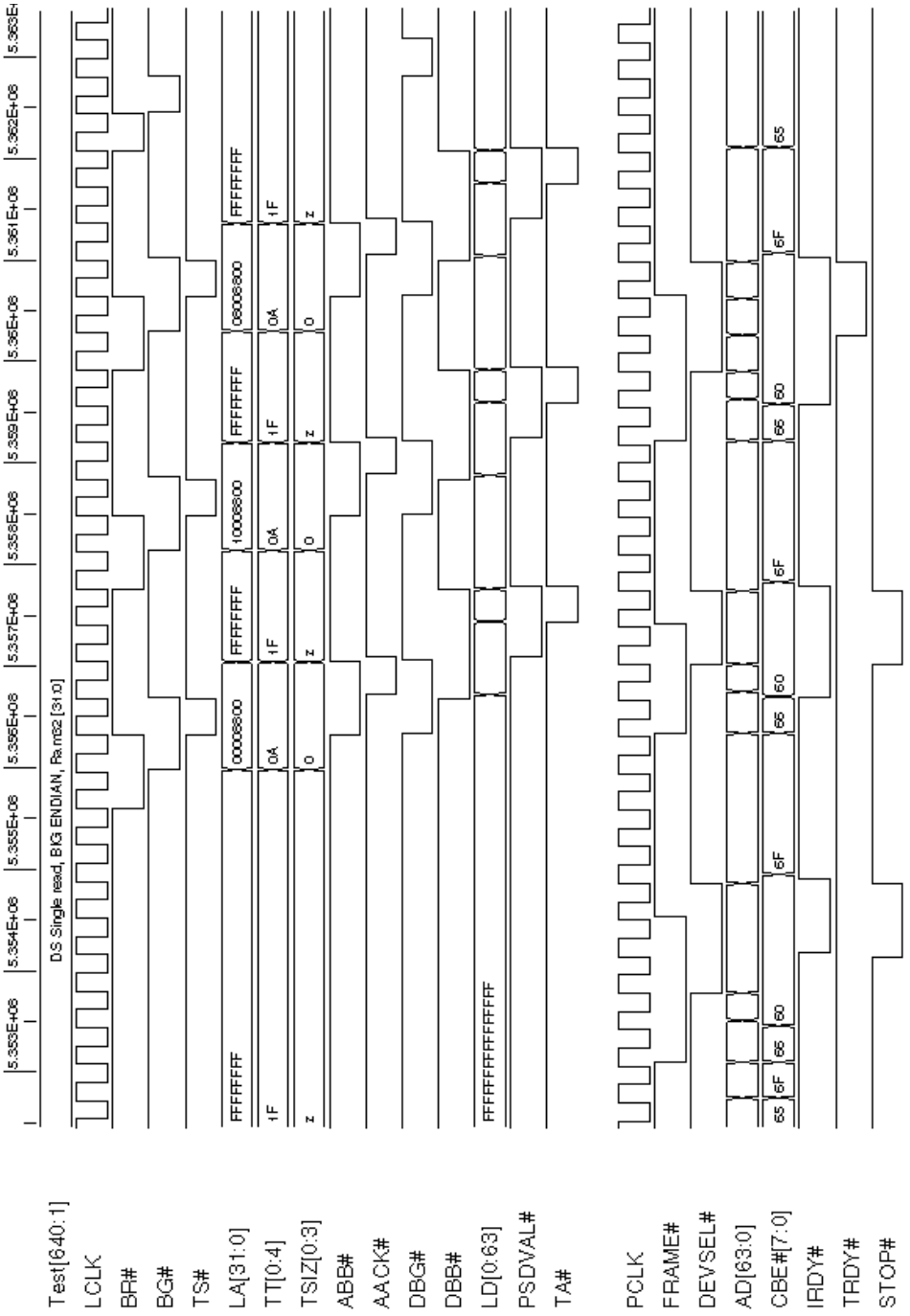
Timing Diagram 3-29. Direct Master Read Burst, Non-Delayed



Timing Diagram 3-30. Direct Slave Write Burst



Timing Diagram 3-31. Direct Master Read Burst with PCI Retry



Timing Diagram 3-32. Single-Qword Direct Slave Read, 32 Bits Each Bus, Prefetch Enabled

4 PCI/60X BUS INTERRUPTS AND USER I/O

4.1 INTERRUPTS

The PCI 9610 has two interrupt outputs and one interrupt input:

- INTA# is an output on the PCI Bus
- LINTo# is an output on the 60x Bus
- LINTi# is an input on the 60x Bus

4.1.1 PCI Interrupts (INTA#)

A PCI 9610 PCI Interrupt (INTA#) can be asserted by one of the following (refer to Figure 4-1):

- 60x-to-PCI Doorbell register (L2PDBELL[31:0])
- 60x Bus Interrupt input (LINTi#)
- Master/Target Abort Status condition (PCISR[13, 12])
- DMA Channel x Done (DMACSRx[4])
- DMA Channel x Terminal Count is reached (DMADPRx[2])
- Messaging Outbound Post Queue not empty (OPQIS[3])
- 256 consecutive PCI Retrys

INTA#, or individual sources of an interrupt, can be enabled or disabled with the PCI 9610 Interrupt Control/Status register (INTCSR). This register also provides the interrupt status of each interrupt source.

The PCI 9610 PCI Bus interrupt is a level output. Disabling an interrupt enable bit or clearing the cause of the interrupt can clear an interrupt.

4.1.2 60x Bus Interrupt Input (LINTi#)

The 60x Interrupt Input Enable bit must be set (INTCSR[11]=1) for interrupts to be acknowledged by the PCI 9610.

Asserting the 60x Bus input LINTi# can assert a PCI Bus interrupt. The PCI Host processor can read the 60x Bus Input Interrupt Active bit to determine whether an interrupt is pending as a result of LINTi# being asserted (INTCSR[15]=1).

The interrupt remains asserted as long as LINTi# is asserted and the 60x Interrupt Input Enable bit is set. The PCI Host processor can take adapter-specific action to cause the 60x Bus to release LINTi#.

If the PCI Interrupt Enable bit is cleared (INTCSR[8]=0), the PCI interrupt, INTA#, is de-asserted; however, the 60x Bus interrupts (LINTi#) and the status bit remain active.

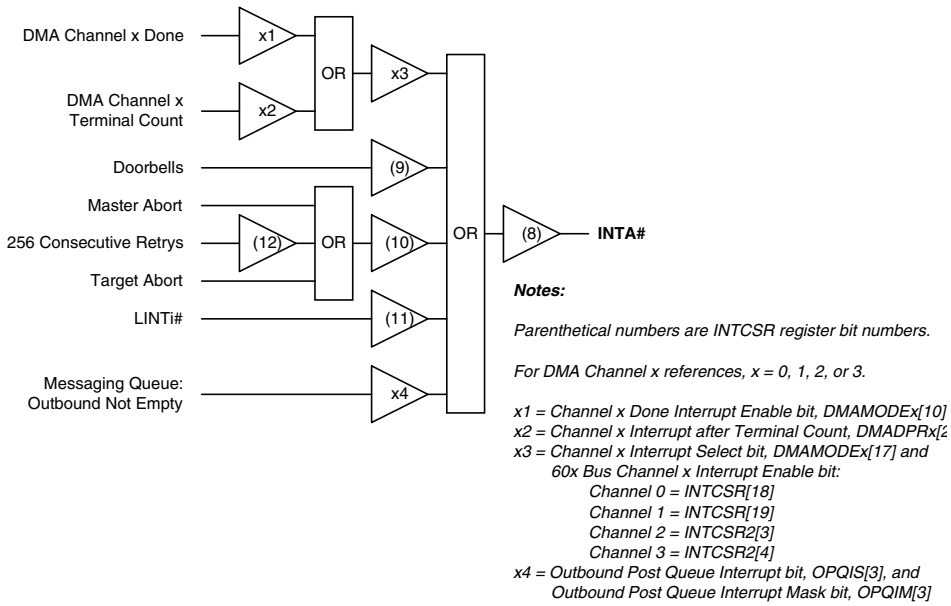


Figure 4-1. Interrupt and Error Sources

4.1.3 60x Bus Interrupt Output (LINTo#)

Table 4-1. 60x Bus LINTo# Sources

Source	Enable	Active Indicator	How Cleared	Possible Asserted TEA#	Comments
INTA#	HOSTEN# low	CNTRL[20]	Write 1 to active indicator bit	No	—
Master PCI Data Parity Error	PCICR[6]	PCISR[8]	Write 1 to active indicator bit	Possible	This interrupt source is activated when three conditions are met. (Refer to PCISR[8]).
PCI 256 Master Retrys	INTCSR[1] INTCSR[12]	INTCSR[27] read as 0	Write 1 to PCISR[11]	Possible	The PCI 9610, as a PCI Master, has issued 256 consecutive Retrys.
PCI Target Abort	INTCSR[1]	PCISR[12]	Write 1 to active indicator bit	Possible	The PCI 9610 has received a Target abort.
PCI Master Abort	INTCSR[1]	PCISR[13]	Write 1 to active indicator bit	Possible	The PCI 9610 has received a Master abort.
PCI Parity	PCICR[6]	PCISR[15]	Write 1 to active indicator bit	Possible	Any of three types of PCI parity errors detected by the PCI 9610 can cause a parity error. (Refer to PCISR[15]).
DMA Done or Transfer Complete	See DMA Notes	DMACSRx[4] and see DMA Notes	Write 1 to DMACSRx[3]	No	—
Outbound Free Queue Full	QSR[6]	QSR[7]	Write 1 to active indicator bit	Possible	—
Inbound Post Queue Not Empty	QSR[4]	QSR[5]	Processor reads queue	No	—
Mailboxes	INTCSR[3] and INTCSR[16]	No active indicator bit	Processor reads mailboxes	No	—
Doorbell	None	P2LDBELL	Clear L2PDBELL	No	A PCI Host writes to L2PDBELL to cause an interrupt.
Power Management	INTCSR[4]	INTCSR[5]	Write 1 to active indicator bit	No	A change in PMCR[1:0] causes an interrupt.
BIST	PCIBISTR[6]	INTCSR[23]	PCIBISTR[3:0] are all zeroes	No	—
TEA# Detected	INTCSR2[1]	INTCSR2[0]	Write 1 to active indicator bit	No	—
TEA# Due to Multibeat Attempt into PCI I/O	INTCSR2[11]	INTCSR2[10]	Write 1 to active indicator bit	Yes	Not only possible, but TEA# is asserted.
PCI SERR# Host	INTCSR[1]	CNTRL[21]	Write 1 to active indicator bit	No	The PCI 9610 has received an SERR# assertion.

DMA Notes: *DMA Done or DMA Transfer Complete is enabled when DMAMODEx[10]=1 AND DMAMODEx[17]=0, where DMAMODEx[10] is the interrupt enable, and DMAMODEx[17] routes the interrupt output to either INTA# or LINTo#.*

In addition to DMACSRx[4], each DMA interrupt has an active status indicator.

Table 4-2. DMA Channel Active Indicator Bits

DMA Channel	Active Status Bit
DMA Channel 0	INTCSR[21]
DMA Channel 1	INTCSR[22]
DMA Channel 2	INTCSR2[5]
DMA Channel 3	INTCSR2[6]

4.1.4 Master/Target Abort Interrupt

The PCI 9610 sets the Received Master Abort or the Target Abort bit (PCISR[13, 11]=1, respectively) when it detects a Master or Target Abort. These status bits cause the PCI INTA# to be asserted if interrupts are enabled.

Interrupt remains set as long as the Receive Master Abort or Target Abort bits remain set and the Master/ Target Abort interrupt is enabled. Use PCI Type 0 Configuration or 60x Bus accesses to clear the Received Master Abort and the Target Abort bits (PCISR[13, 11]=0, respectively).

The Interrupt Control/Status register bits (INTCSR [26:24] and INTCSR2[8:7]) are latched at the time of a Master or Target Abort interrupt. These bits provide information when an abort occurs, such as which device was the Master when the abort occurred.

The PCI Abort Address is stored in PABTADR[31:0].

4.1.5 Mailbox Registers

The PCI 9610 has eight 32-bit Mailbox registers that can be written to and read from both the PCI and 60x Buses. These registers can be used to pass command and status information directly between the PCI and 60x Bus devices.

A 60x Bus interrupt can be asserted, if enabled (INTCSR[3, 16], respectively), when the PCI Host writes to one of the first four Mailbox registers (MBOX0, MBOX1, MBOX2, or MBOX3).

To clear the Mailbox registry, the destination bus should read the values currently in the Mailbox registers.

4.1.6 Doorbell Registers

The PCI 9610 has two 32-bit Doorbell Interrupt/Status registers. One is assigned to the PCI Bus interface. The other is assigned to the 60x Bus interface.

A PCI Host can assert a 60x Bus interrupt by writing any number other than all zeroes (0) to the PCI-to-60x Doorbell register bits (P2LDBELL[31:0]). The PCI and 60x Bus interrupts remain asserted until all bits are cleared to zero.

A 60x Bus Master can assert a PCI Bus interrupt by writing any number other than that of all zeroes (0) to the 60x-to-PCI Doorbell register bits (L2PDBELL [31:0]).

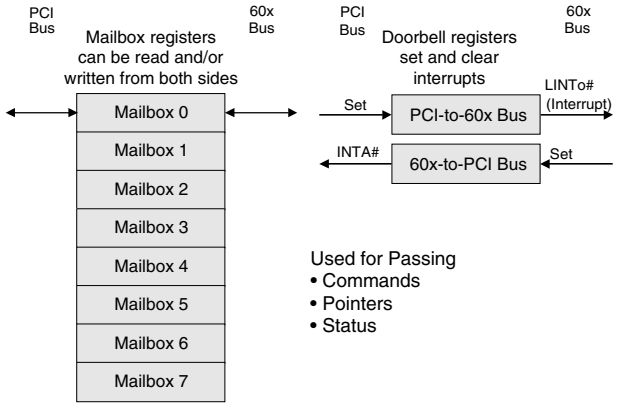


Figure 4-2. Mailbox and Doorbell Message Passing

4.1.6.1 60x-to-PCI Doorbell Interrupt

A 60x Bus master can assert a PCI Bus interrupt by writing to the 60x-to-PCI Doorbell Register bit(s) (L2PDBELL[31:0]). The PCI Host processor can read the PCI Doorbell Interrupt Active bit to determine whether a PCI Doorbell interrupt is pending (INTCSR[13]), and if so, read the PCI 9610 60x-to-PCI Doorbell register.

Each bit in the 60x-to-PCI Doorbell register is individually controlled. The 60x Bus can only set bits in the 60x-to-PCI Doorbell register. From the 60x Bus, writing 1 to any bit position sets that bit and writing 0 has no effect. Bits in the 60x-to-PCI Doorbell register can only be cleared from the PCI Bus. From the PCI Bus, writing 1 to any bit position clears that bit and writing 0 has no effect.

Interrupts remain set as long as any 60x-to-PCI Doorbell register bits are set and the PCI Doorbell Interrupt Enable bit (INTCSR[9]) is set.

4.1.6.2 PCI Internal Register Race Conditions Prevented

When accessing PCI 9610 internal registers, there are no race conditions between a PCI Bus master and a master on the 60x Bus.

If a master on the 60x Bus attempts to read or write to a PCI 9610 register while a PCI Bus master is accessing a PCI 9610 register, the PCI 9610 holds off asserting the PSDVAL# signal until the PCI Bus Master access is complete. The PCI 9610, as a 60x Bus Slave, completes the Address tenure, and allows Data tenure to begin. The PCI 9610 waits until the PCI Bus access to the internal register is complete before asserting PSDVAL# to continue the 60x Bus access.

4.1.6.3 PCI-to-60x Doorbell Interrupt

A PCI Bus Master can assert a 60x Bus interrupt by writing to the PCI-to-60x Doorbell register bits (P2LDBELL[31:0]). The 60x Bus Master can read the 60x Bus Doorbell Interrupt Active bit to determine whether a 60x Bus Doorbell interrupt is pending (P2LDBELL[20]), and if so, read the PCI 9610 PCI-to-60x Doorbell register.

Each bit in the PCI-to-60x Doorbell register is individually controlled. The PCI Bus only sets bits in the PCI-to-60x Doorbell register. From the PCI Bus, writing 1 to any bit position sets that bit and writing 0 to a bit position has no effect. Bits in the PCI-to-60x Doorbell register can be cleared only from the 60x Bus. From the 60x Bus, writing 1 to any bit position clears that bit and writing 0 has no effect.

Note: *If the 60x Bus cannot clear a Doorbell Interrupt, do not use the PCI-to-60x Doorbell register.*

Interrupts remain set as long as any PCI-to-60x Doorbell register bits are set and the 60x Bus Doorbell Interrupt Enable bit is set (INTCSR[17]=1).

To prevent race conditions when the 60x Bus is accessing the PCI-to-60x Doorbell register (or any Configuration register), the PCI 9610 automatically issues a Retry to the PCI Bus.

4.1.7 Built-In Self Test Interrupt (BIST)

A PCI Bus master can assert a 60x Bus interrupt by performing a PCI Type 0 Configuration write to a bit in the PCIBISTR register. A 60x Bus master can read the BIST Interrupt Active bit (INTCSR[23]) to determine whether a BIST interrupt is pending.

Interrupts remain set as long as the bit is set and the PCI BIST Interrupt Enable bit is set (PCIBISTR[6]=1). The 60x Bus then resets the bit when BIST completes. The PCI Host software may fail the device if the bit is not reset after two seconds.

Note: *The PCI 9610 does not have an internal BIST.*

4.1.8 DMA Channel x Interrupts

A DMA channel can assert a PCI Bus or 60x Bus interrupt when done (transfer complete) or after a transfer is complete for the current descriptor in Scatter/Gather DMA mode. The DMA Channel Interrupt Select bit(s) determine whether to assert a PCI Bus interrupt (DMAMODEx[17]=1) or 60x Bus LINTo# interrupt (DMAMODEx [17]=0). The PCI or 60x Bus Master can read the DMA Channel x Interrupt Active bit(s) to determine whether a DMA Channel x interrupt is pending (INTSCR2[6:5]=1 and/or INTCSR[22:21]=1).

The Channel x Done bit(s) (DMACSRx[4]) can be used to determine whether an interrupt is one of the following:

- DMA Done interrupt
- Transfer complete for current descriptor interrupt

The Channel x Done Interrupt Enable bit(s) (DMAMODEx[10]) enable a Done interrupt. In Scatter/Gather DMA mode, a bit in the Next Descriptor Pointer register of the channel (loaded from 60x Bus memory) specifies whether to assert an interrupt at the end of the transfer for the current descriptor.

A DMA Channel interrupt is cleared by the Channel x Clear Interrupt bit(s) (DMACSRx[3]=1).

4.1.9 PCI SERR# (PCI NMI)

The PCI 9610 asserts an SERR# pulse when the following occurs:

- Parity checking is enabled in the Parity Error Response bit (PCICR[6]=1), and
- An address is detected or 1 is written to the Generate PCI Bus SERR# Interrupt bit (INTCSR[2]) with a current value of 0

SERR# output can be enabled or disabled with the SERR# Enable bit (PCICR[8]).

4.1.10 60x Bus PCI SERR#

When the PCI 9610 is a master on the 60x Bus, if the Slave responds with a TEA# assertion, then the PCI 9610 asserts PCI SERR# output if the following bits are set:

- TEA# Input Interrupt Mask (LMISC1[5]=1)
- SERR# Enable (PCICR[8]=1)

Whenever the PCI 9610 asserts SERR#, the Signaled System Error bit is set (PCISR[14]=1).

4.1.11 60x Bus NMI

If the Parity Error Response bit is set (PCICR[6]=1), the PCI 9610 sets the Master Data Parity Error Detected bit (PCISR[8]=1) when the following three conditions are met:

- The PCI 9610 asserted PERR# or acknowledged PERR# was asserted
- The PCI 9610 was the Bus Master for the operation in which the error occurred
- The Parity Error Response bit is set (PCICR[6]=1)

The PCI 9610 sets the Detected Parity Error bit (PCISR[15]=1) if it detects one of the following conditions:

- The PCI 9610 detected a parity error during a PCI Address phase
- The PCI 9610 detected a data parity error when it is the target of a write
- The PCI 9610 detected a data parity error when performing Master Read operation

4.1.12 TEA# and Interrupts

The interrupt associated with the 60x Bus is the PCI 9610 output signal LINTo#. Some of the causes of a LINTo# interrupt may also cause the assertion of TEA# by the PCI 9610.

4.1.12.1 TEA# Asserted by the PCI 9610

A Transfer Error Acknowledge (TEA#) is asserted by the PCI 9610 only during the Data Tenure phase when the PCI 9610 is a 60x Bus slave. TEA# assertion indicates to the 60x Bus Master that a bus error has occurred.

A given condition to cause a TEA# assertion may, or may not, actually cause a TEA# assertion. This is because the timing for the TEA# signal may not align with the detected error. In each case, there is a status bit associated with the error condition. Each status bit may be enabled to cause a LINTo# interrupt output from the PCI 9610.

There are nine errors that *might* cause a TEA#:

- Master PCI Parity Error
- PCI Parity Error
- PCI Target Abort received by the PCI 9610
- PCI Master 256 Consecutive Retrys
- PCI Master Abort
- PCI Parity (any of the three types defined in the Detected Parity Error bit, PCISR[15])
- Outbound Messaging Free Queue Full
- Direct Master burst (greater than 32 bits) to/from PCI I/O
- 60x Bus processor attempted burst (greater than 32 bits) to/from Configuration space

If a master on the 60x Bus attempts a Direct Master Read or Write access to PCI I/O space with larger than a 4-byte access, or if the 60x Bus Master attempts a PCI 9610 internal register access of larger than four bytes, the PCI 9610 asserts TEA#.

4.1.12.2 TEA# as an Input to the PCI 9610

When the PCI 9610 is a 60x Bus master, if TEA# is asserted while DBB# is asserted, the PCI 9610 terminates the transfer and "gets off the bus." Any data in a PCI 9610 FIFO for that transaction, if a PCI 9610 output, are flushed. Any data in a PCI 9610 FIFO for that transaction that are destined to the PCI Bus continue.

The cycle is not Retried.

The TEA# Assertion Detected bit is set (INTCSR2[0]=1). If the Enable TEA# Assertion Detected as LINTo# Output bit is set (INTCSR2[1]=1), LINTo# is asserted.

4.2 USER INPUT AND OUTPUT

The PCI 9610 supports user input and output pins, USERi and USERo (grid locations T3 and U1, respectively). Both are multiplexed with other functional pins. The default PCI 9610 conditions are the USERi and USERo functions. USERi is selected when CNTRL[18]=1, and USERo is selected when CNTRL[19]=1. User output data can be logged by writing to the General Purpose Output bit (CNTRL[16]). User input data can be read from the General Purpose Input bit (CNTRL[17]).

5 INTELLIGENT I/O (I_2O)

5.1 I_2O-COMPATIBLE MESSAGE UNIT

The I_2O-compatible Messaging Unit supplies two paths for messages, two inbound FIFOs to receive messages from the primary PCI Bus, and two outbound FIFOs to pass messages to the primary PCI Bus. Refer to *I_2O Architecture Specification r1.5* for details.

Figure 5-1 and Figure 5-2 illustrate I_2O architecture.

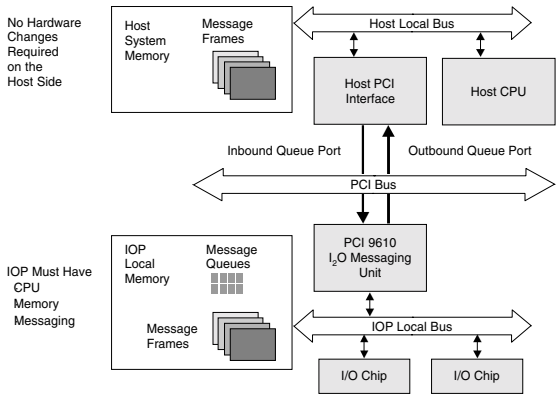


Figure 5-1. Typical I_2O Server/Adapter Card Design

Note: *IOP = I/O Processor.*

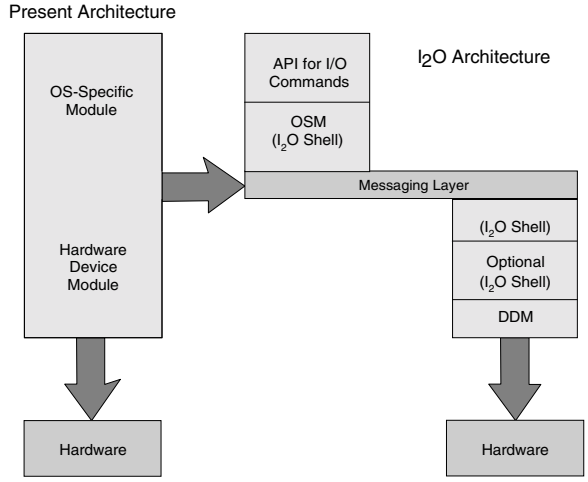


Figure 5-2. Driver Architecture Compared

5.1.1 Inbound Messages

Inbound messages reside in a pool of message frames (minimum 64-byte frames) allocated in the shared 60x Bus memory. The inbound message queue is comprised of a pair of rotating FIFOs implemented in 60x Bus memory. The Inbound Free List FIFO holds the message frame addresses (MFA) of available message frames in 60x Bus Master memory. The Inbound Post Queue FIFO holds the MFA of all currently posted messages.

External PCI agents, through the Inbound Queue Port location in PCI Address space, access inbound circular FIFOs. (Refer to Table 5-2 on page 5-6.) The Inbound Queue Port, when read by an external PCI agent, returns the Inbound Free List FIFO MFA. The external PCI agent places a message frame into the Inbound Post Queue FIFO by writing its MFA to the Inbound Queue Port location.

5.1.2 Outbound Messages

Outbound messages reside in a pool of message frames (minimum 64-byte frames) allocated in the shared PCI Bus (Host System) memory. The Outbound message queue is comprised of a pair of rotating FIFOs implemented in 60x Bus memory. The Outbound Free List FIFO holds the message frame addresses (MFA) of available message frames in system memory. The Outbound Post Queue FIFO holds the MFA of all currently posted messages.

External PCI agents, through the Outbound Queue Port location in PCI Address space access outbound circular FIFOs. (Refer to Table 5-2 on page 5-6.) The Outbound Queue Port, when read by an external PCI agent, returns the Outbound Post Queue FIFO MFA. The External PCI agent places free message frames into the Outbound Free List FIFO by writing the free MFA into the Outbound Queue Port location.

Memory for the circular FIFOs must be allocated in 60x Bus memory. The base address of the queue is contained in the Queue Base Address bits (QBAR[31:20]). Each FIFO entry is a 32-bit data value. Each read and write of the queue must be a single 32-bit access.

Circular FIFOs range in size from 4- to 64-KB entries. All four FIFOs must be the same size and contiguous. Therefore, the total amount of 60x Bus memory needed for circular FIFOs ranges from 64 KB to 1 MB. A FIFO size is specified in the Circular Queue Size bits (MQCR[5:1]).

The starting address of each FIFO is based on the Queue Base Address and the FIFO Size, as listed in the following table.

Table 5-1. Queue Starting Address

FIFO	Starting Address
Inbound Free List	QBAR
Inbound Post List	QBAR + (1 * FIFO Size)
Outbound Post List	QBAR + (2 * FIFO Size)
Outbound Free List	QBAR + (3 * FIFO Size)

5.1.3 I_2O Pointer Management

The FIFOs always reside in shared 60x Bus memory and are allocated and initialized by the I/O Processor (IOP). Before setting the Queue Enable bit (MQCR[0]=1), the 60x Bus Master must initialize the following registers, with the initial offset according to the configured FIFO size:

- Inbound Post and Free Head Pointer registers (IPHPR)
- Inbound Post and Free Tail Pointer registers (IPTPR)
- Outbound Post and Free Head Pointer registers (OFHPR)
- Outbound Post and Free Tail Pointer registers (OFTPR)

The Messaging Unit automatically adds the Queue Base Address to offset in each head and tail pointer register. The software can then enable I_2O. After initialization, the 60x Bus Master software should not write to the pointers managed by the MU hardware.

Empty flags are set if the queues are disabled (MQCR[0]=0) or head and tail pointers are equal. This occurs independent of how the head and tail pointers are set.

An empty flag is cleared, signifying not empty, only if the queues are enabled and pointers become not equal.

If an empty flag is cleared and the queues are enabled, the empty flag is set only if the tail pointer is incremented and the head and tail pointers become equal.

Full flags are always cleared when the queues are disabled or the head and tail pointers are not equal.

A full flag is set when the queues are enabled, the head pointer is incremented, and the head and tail pointers become equal.

Each circular FIFO has a head pointer and a tail pointer, which are offsets from the Queue Base Address. (Refer to Table 5-2 on page 5-6.) Writes to a FIFO occur at the head of the FIFO and reads occur from the tail. Head and tail pointers are incremented by a 60x Bus Master. The unit that writes to the FIFO also maintains the pointer. Pointers are incremented after a FIFO access. Both pointers wrap around to the first address of the circular FIFO when they reach the FIFO size, so that the head and tail pointers continuously “chase” each other around in the circular FIFO. The MU wraps the pointers automatically for the pointers that it maintains. IOP software must wrap the pointers that it maintains. Whenever they are equal, the FIFO is empty. To prevent overflow conditions, I_2O specifies that the number of message frames allocated should be less than or equal to the number of entries in a FIFO. (Refer to Figure 5-3.)

Each inbound MFA is specified by I_2O as the offset from the start of shared 60x Bus memory region 0 to the start of the message frame. Each outbound MFA is specified as the offset from Host memory location 0x00000000h to the start of the message frame in shared Host memory. Because the MFA is an actual address, the message frames need not be contiguous. IOP allocates and initializes inbound message frames in shared IOP memory using any suitable memory allocation technique. Host allocates and initializes outbound message frames in shared Host memory using any suitable memory allocation technique. Message frames are a minimum of 64 bytes in length.

I_2O uses a “push” (write-preferred) memory model. That means the IOP writes messages and data to the shared Host memory, and the Host writes messages and data to shared IOP memory. Software should make use of Burst and DMA transfers whenever possible to ensure efficient use of the PCI Bus for message passing.

Additional information on message passing implementation may be found in *I_2O Architecture Specification r1.5*.

5.1.4 Inbound Free List FIFO

The 60x Bus Master allocates inbound message frames in its shared memory and can place the address of a free (available) message frame into the Inbound Free List FIFO by writing its MFA into the FIFO location pointed to by the Queue Base register + Inbound Free Head Pointer register. The 60x Bus Master must then increment the Inbound Free Head Pointer register.

A PCI Master (Host or other IOP) can obtain the MFA of a free message frame by reading the Inbound Queue Port Address (40h of the first PCI Memory Base Address register). If the FIFO is empty (no free inbound message frames are currently available, head and tail pointers are equal), the MU returns -1 (FFFFFFFFh). If the FIFO is not empty (head and tail pointers are not equal), the MU reads the MFA pointed to by the Queue Base register + Inbound Free Tail Pointer register, returns its value and increments the Inbound Free Tail Pointer register. If the Inbound Free Queue is not empty, and the Inbound Free Queue Prefetch Enable bit is set (QSR[3]=1), the next entry in the FIFO is read from the Processor Bus into a prefetch register. The prefetch register then provides the data for the next PCI read from this queue, thus reducing the number of PCI wait states. (Refer to Figure 5-3.)

5.1.5 Inbound Post Queue FIFO

A PCI Master (Host or other IOP) can write a message into an available message frame in the shared 60x Bus Master memory. It can then post that message by writing the Message Frame Address (MFA) to the Inbound Queue Port Address, IQP (40h of the first PCI Memory Base Address register). When the port is written, the MU writes the MFA to the Inbound Post Queue FIFO location pointed to by the Queue Base register + FIFO Size + Inbound Post Head Pointer register. After the MU writes the MFA to the Inbound Post Queue FIFO, it increments the Inbound Post Head Pointer register.

The Inbound Post Tail Pointer register points to the Inbound Post Queue FIFO location, which holds the MFA of the oldest posted message. The 60x Bus Master maintains the tail pointer. After a 60x Bus Master reads the oldest MFA, it can remove the MFA from the Inbound Post Queue FIFO by incrementing the Inbound Post Tail Pointer register.

The PCI 9610 asserts a 60x Bus Interrupt when the Inbound Post Queue FIFO is not empty. The Inbound Post Queue FIFO Interrupt bit in the Queue Status/Control register (QSR[5]) indicates the interrupt status. The interrupt clears when the Inbound Post Queue FIFO is empty. The Inbound Post Queue FIFO Interrupt Mask bit (QSR[4]) can mask the interrupt.

To prevent racing between the time the PCI Write transaction is received until the data is written in 60x Bus memory and the Inbound Post Head Pointer register is incremented, any Direct Slave access to the PCI 9610 is issued a Retry.

5.1.6 Outbound Post Queue FIFO

A 60x Bus Master can write a message into an available message frame in shared Host memory. It can then post that message by writing the Message Frame Address (MFA) to the Outbound Post Queue FIFO location pointed to by the Queue Base register + Outbound Post Head Pointer register + (2 * FIFO Size). The 60x Bus Master should then increment the Outbound Post Head Pointer register.

A PCI Master can obtain the MFA of the oldest posted message by reading the Outbound Queue Port Address (44h of the first PCI Memory Base Address register). If the FIFO is empty (no more outbound messages are posted, head and tail pointers are equal), the MU returns -1 (FFFFFFFFh). If the Outbound Post Queue FIFO is not empty (head and tail pointers are not equal), the MU reads the MFA pointed to by the Queue Base register + (2 * FIFO Size) + outbound Post Tail Pointer register, returns its value and increments the Outbound Post Tail Pointer register.

The PCI 9610 asserts a PCI Interrupt when the Outbound Post Head Pointer register is not equal to the Outbound Post Tail Pointer register. The Outbound Post Queue FIFO Interrupt bit of the Outbound Post Queue Interrupt Status register (OPQIS) indicates the interrupt status. When the pointers become equal, both the interrupt and the Outbound Post Queue FIFO interrupt bit are automatically cleared. Pointers become equal when a PCI Master (Host or other IOP) reads sufficient FIFO entries to empty the FIFO. The Outbound Post Queue FIFO Interrupt Mask register (OPLFIM) can mask the Interrupt.

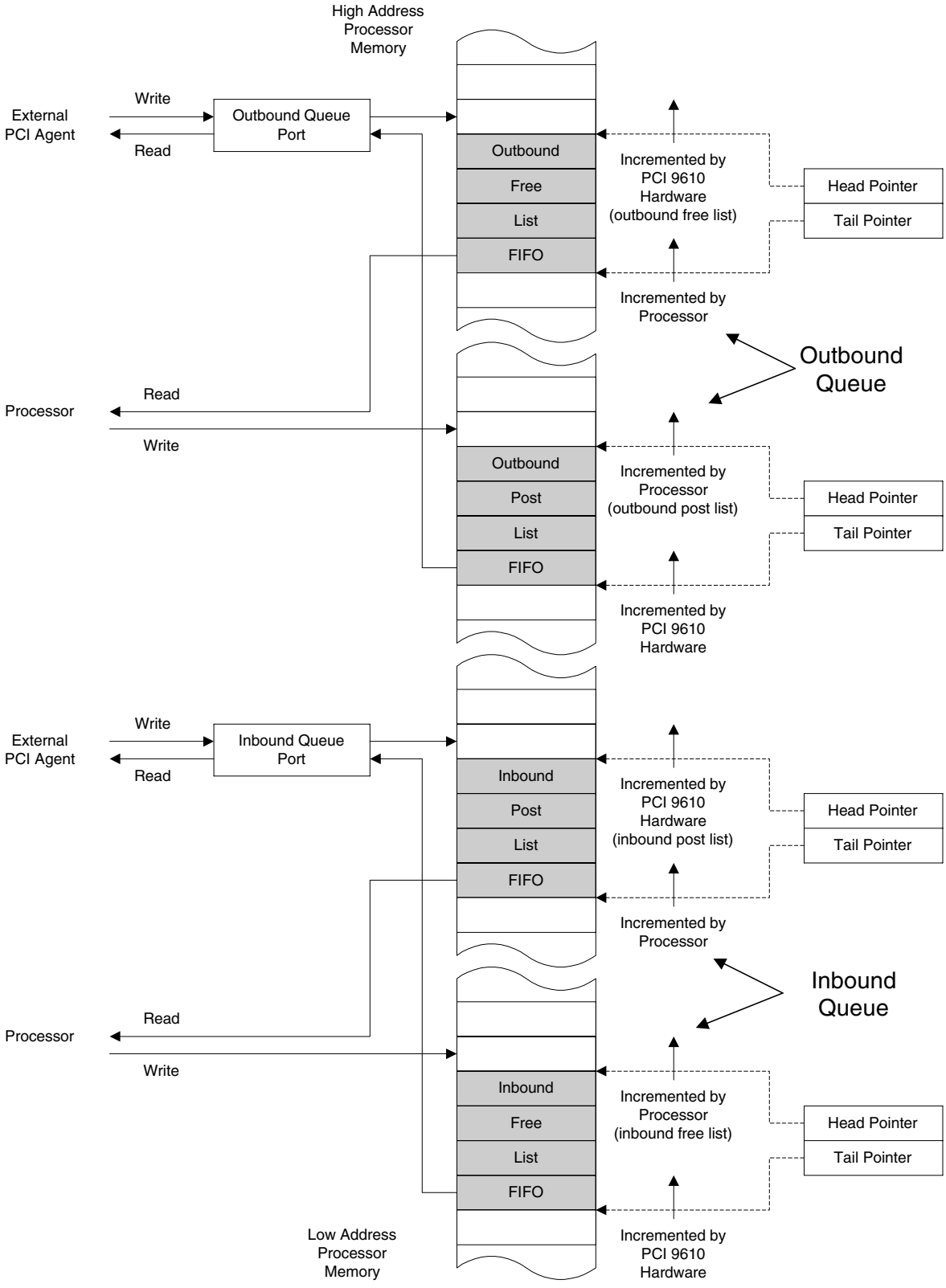


Figure 5-3. Circular FIFO Operation

5.1.7 Outbound Post Queue

To reduce read latency, prefetching from the tail of the queue occurs whenever the queue is not empty and the tail pointer is incremented (queue has been read from), or when the queue is empty and the head pointer is incremented (queue has been written to). When the Host CPU reads the Outbound Post Queue, the data is immediately available.

5.1.8 Inbound Free Queue

To reduce read latency, prefetching from the tail of the queue occurs whenever the queue is not empty and the tail pointer is incremented (queue has been read from), or when the queue is empty and the head pointer is incremented (queue has been written to). When the Host CPU reads the Inbound Free Queue, the data is immediately available.

5.1.9 Outbound Free List FIFO

The PCI Bus Master (Host or other IOP) allocates outbound message frames in its shared memory. The PCI Bus Master can place the address of a free (available) message frame into the Outbound Free List FIFO by writing a Message Frame Address (MFA) to the Outbound Queue Port Address (44h of the first PCI Memory Base Address register). When the port is written, the MU writes the MFA to the Outbound Free List FIFO location pointed to by the Queue Base register + (3 * FIFO Size) + Outbound Free Head Pointer register. After the MU writes the MFA to the Outbound Free List FIFO, it increments the Outbound Free Head Pointer register.

When the IOP needs a free outbound message frame, it must first check whether any free frames are available. If the Outbound Free List FIFO is empty (outbound free head and tail pointers are equal), the IOP must wait for the Host to place additional outbound free message frames in the Outbound Free List FIFO. If the Outbound Free List FIFO is not empty (head and tail pointers are not equal), the IOP can obtain the MFA of the oldest free outbound message frame by reading the location pointed to by the Queue Base register + (3 * FIFO Size) + Outbound Free Tail Pointer register. After the IOP reads the MFA, it must increment the Outbound Free Tail Pointer register. To prevent overflow conditions, I_2O specifies the number of message frames allocated should be less than or equal to the number of entries in a FIFO. The MU also checks for overflows of the Outbound Free List FIFO. When the head pointer is incremented and becomes equal to the tail pointer, the Outbound Free List FIFO is full, and the MU asserts a LINTo# interrupt. The interrupt is recorded in the Queue Status/Control register (QSR).

From the time the PCI Write transaction is received until the data is written into 60x Bus memory and the Outbound Free Head Pointer register is incremented, any Direct Slave access to the PCI 9610 is issued a Retry.

5.1.10 I_2O Enable Sequence

To enable I_2O, the 60x Bus Master performs the following:

- Initialize Space 1 address and range
- Initialize all FIFOs and Message Frame memory
- Set the PCI Base Class Code bits (PCICCR[23:16]) to be an I_2O device with programming interface 01h
- Set the I_2O Decode Enable bit (QSR[0])
- Set 60x Bus Init Status bit to “done” (LMISC1[2]=1)

Note: *The serial EEPROM must not set the 60x Bus Init Status bit so that the PCI 9610 issues Retrys to all PCI accesses until the 60x Bus Init Status bit is set to “done” by the 60x Bus.*

The I_2O Decode Enable bit (QSR[0]) causes remapping of resources for use in I_2O mode. When set, all Memory-Mapped Configuration registers (*for example*, queue ports 40h and 44h) and Space 1 share the PCIBAR0 register. PCI accesses to offset 00h-FFh of PCIBAR0 result in accesses to the PCI 9610 Internal Configuration registers.

Accesses above offset FFh of PCIBAR0 result in 60x Bus Space accesses, beginning at offset 100h from the Remap PCI Address to 60x Bus Address Space 1 into the 60x Bus Address Space bits (LAS1BA[31:4]). Therefore, space located at offset 00h-FFh from LAS1BA is not addressable from the PCI Bus using PCIBAR0.

Note: *Because PCI accesses to offset 00h-FFh of PCIBAR0 result in internal configuration accesses, the Inbound Free MFAs must be greater than FFh.*

Table 5-2. Circular FIFO Summary

FIFO Name	PCI Port	Generate PCI Interrupt	Generate 60x Bus Interrupt	Head Pointer Maintained By	Tail Pointer Maintained By
Inbound Free List FIFO	Inbound Queue Port (Host read)	No	No	60x Bus Master	MU hardware
Inbound Post List FIFO	Inbound Queue Port (Host write)	No	Yes, when Port is written	MU hardware	60x Bus Master
Outbound Post List FIFO	Outbound Queue Port (Host read)	Yes, when FIFO is not empty	No	60x Bus Master	MU hardware
Outbound Free List FIFO	Outbound Queue Port (Host write)	No	Yes, (Linto#) when FIFO is full	MU hardware	60x Bus Master

6 PCI POWER MANAGEMENT

6.1 OVERVIEW

The *PCI Bus Power Management Interface Specification, r1.1*, provides a standard mechanism for operating systems to control add-in cards for power management. The Specification defines four PCI functional power states—D_0, D_1, D_2, and D_3. States D_0 and D_3 are required, while states D_1 and D_2 are optional. State D_0 represents the highest power consumption and state D_3 the least.

- **D_0 (Uninitialized)**—Enters this state from Power-On Reset or from state D_{3hot}. Supports Direct Slave PCI transactions only.
- **D_0 (Active)**—All functions active.
- **D_1**—Uses less power than State D_0, and more than state D_2. Light sleep state.
- **D_2**—Uses very little power.

  The functional states are defined by the allowed activities of the add-in card with the PCI 9610.

  The function supports PCI Configuration cycles to function if clock is running (memory, I/O, bus mastering, and interrupts are disabled). It also supports the Wakeup Event from function, but not standard PCI interrupts.
- **D_{3hot}**—Uses lower power than any other state. Supports PCI Configuration cycles to function if clock is running. Supports Wakeup Event from function, but not standard PCI interrupts. When programmed for state D_0, an internal soft reset occurs. The PCI Bus drivers must be disabled. PME# context must be retained during this soft reset.
- **D_{3cold}**—No power. Supports Bus reset only. All context is lost in this state.

From a power management perspective, the PCI Bus can be characterized at any point in time by one of four power management states—B_0, B_1, B_2, and B_3:

- **B_0 (Fully On)**—Bus is fully usable with full power and clock frequency, *PCI r2.1*-compliant. Fully operational bus activity. This is the only Power Management state in which data transactions can occur.
- **B_1**—Intermediate power management state. Full power with clock frequency, *PCI r2.1*-compliant. PME Event driven bus activity. Vcc is applied to all devices on the bus, and no transactions are allowed to occur on the bus.
- **B_2**—Intermediate power management state. Full power clock frequency stopped, *PCI r2.1*-compliant (in the low state). PME Event-driven bus activity. Vcc is applied to all devices on the bus; however, the clock is stopped and held in the Low state.
- **B_3 (Off)**—Power to the bus is switched off. PME Event-driven bus activity. Vcc is removed from all devices on the PCI Bus.

All system PCI Buses have an originating device, which can support one or more power states. In most cases, this creates a bridge (such as a Host-to-PCI Bus or a PCI-to-PCI bridge).

Function States must be at the same or lower energy state than the bus on which they reside.

6.1.1 PCI Power Management Functional Description

The PCI 9610 passes power management information and has no inherent power-saving feature.

The PCI Status register (PCISR) and the New Capability Pointer register (CAP_PTR) indicate whether a new capability (the Power Management function) is available. The New Capability Functions Support bit (PCISR[4]) enables a PCI BIOS to identify a New Capability function support. This bit is executable for writes from the Processor Bus, and reads from the Processor and PCI Buses. CAP_PTR provides an offset into PCI Configuration Space, the start location of the first item in a New Capabilities Linked List.

The Power Management Capability ID register (PMCAPID) specifies the Power Management Capability ID, 01h, assigned by the PCI SIG. The Power Management Next Capability Pointer register (PMNEXT) points to the first location of the next item in the capabilities linked list. If Power Management is the last item in the list, then this register should be set to 0. The default value for the PCI 9610 is 48h (Hot Swap).

For the PCI 9610 to change the power state and assert PME#, a 60x or PCI Bus Master should set the PME_En bit (PMCSR[8]=1). The Processor Host then determines to which power state the backplane should change by reading the Power State bits (PMCSR[1:0]).

The Processor Host sets up the following:

- D_2_Support and D_1_Support bits (PMC[10:9]) are used by the Processor Host to identify power state support
- PME_Support bits (PMC[14:11]) are used by the PCI 9610 to identify the PME# Support correspondent to a specific power state (PMCSR[1:0])

The Processor Host then sets the PME_Status bit (PMCSR[15]=1) and the PCI 9610 asserts PME#. To clear the PME_Status bit, the PCI Host must write a 1 to the PME# Status bit (PMCSR[15]=1). To disable the PME# Interrupt signal, either Host can write a 0 to the PME_En bit (PMCSR[8]=0).

LINTo# is asserted every time the power state in the PMCSR register changes. Transmission from state 11 (D_{3hot}) to state 00 (D_0) causes a soft reset. A soft reset should only be initiated from the PCI Bus because the Processor Bus interface is reset during a soft reset. The PCI 9610 issues LRESET# and resets all its internal registers to their default values. In state D_{3hot}, PCI Memory and I/O accesses are disabled, as well as PCI interrupts, and only configuration is allowed. Before making LINTo# work, set the Power Management Interrupt Enable bit (INTCSR[4]=1), and clear the interrupt by setting the Power Management Interrupt bit (INTCSR[5]=1).

The Data_Scale bits (PMCSR[14:13]) indicate the scaling factor to use when interpreting the value of the Power Management Data bits (PMDATA[7:0]). The value and meaning of the bits depend upon the data value specified in the Data_Select bits (PMCSR[12:9]). The Data_Scale bit value is unique for each Data_Select bit. For Data_Select values from 8 to 15, the Data_Scale bits always return a zero (PMCSR[14:13]=0).

PMDATA provides operating data, such as power consumed or heat dissipation.

6.1.2 66 MHz PCI Clock Power Management D_2 Support

The PCI 9610 provides full support for the D_2 Power Management state at a 33 MHz PCI clock frequency. The *PCI r2.2*-compliant 66 MHz PCI clock frequency prohibits any change to the clock without the system reset (RST#) being asserted. (Refer to the *PCI Local Bus Specification, r2.2* and *PCI Bus Power Management Interface Specification, r1.1*.) Therefore, the PCI 9610 cannot support D_2 Power Management state at 66 MHz. To do that, the PCI 9610 requires an external control to avoid enabling the D_2 Power Management feature at a 66 MHz clock frequency. Default booting of the PCI 9610 sets D_2 support to a disabled state. All 66 MHz add-in cards, capable of running at a 66 MHz PCI clock frequency, must monitor the M66EN# PCI connector pin. When this pin is present on a card, the 60x Bus Master can monitor the pin, and enable D_2 Power Management support (PMC[10]) by way of the register access whenever the M66EN# PCI connector pin is sampled false.

6.1.3 Power Management D_{3cold} Support

The PCI 9610 provides full support for the D_{3cold} Power Management state with PME# assertion and register contents storage. The PCI 9610 has all pins required by the *PCI Bus Power Management Interface Specification, r1.1*. Special attention is necessary for the following pins:

- **2.5V$_{AUX}$**—Power input pin routed to the D_{3cold} support core logic. Due to unavailable power from the PCI slot, 2.5V power must be supplied by an external power source, voltage regulator.
- **Card_V$_{AUX}$**—3.3V$_{AUX}$ power input pin driven by the PCI backplane through add-in card Auxiliary Power Routing. (Refer to *PCI Bus Power Management Interface Specification, r1.1*, Figure 12.)
- **PRESENT_DET**—Present Detect pin provided by add-in card Auxiliary Power Routing (refer to *PCI Bus Power Management Interface Specification, r1.1,* Figure 12) to enable the D_{3cold} PME# assertion feature within the PCI 9610 silicon.

- **PME#**—Optional open drain, active low signal intended to be driven low by the PCI 9610 to request a change in its current power management state and/or to indicate that a PME# has occurred. The PCI 9610 requires external logic to avoid unexpected Wake-Up events to occur whenever an add-in card is plugged into the *PCI r2.2*-compliant PCI backplane. (Refer to *PCI Bus Power Management Interface Specification, r1.1,* Chapter 7.)
- **PMEREQ#**—Input signal used to request a wake-up event only when the add-in card is in the D_{3cold} Power Management state.
- **IDDQEN#**—Input signal providing main power status to the PCI 9610 D_{3cold} Power Management logic.

Note: *All signal I/Os used for D_{3cold} Power Management support are powered by Card_V_{AUX} power.*

6.1.4 System Changes Power Mode Example

1. The Host writes to the PCI 9610 PMCSR register to change the power states.
2. The PCI 9610 sends a 60x Bus interrupt (LINTo#) to a Processor CPU (LCPU).
3. The LCPU has 200 µs to read the power management information from the PCI 9610 PMCSR register to implement the power-saving function.
4. After the LCPU implements the power saving function, the PCI 9610 disables all Direct Slave accesses and PCI Interrupt output (INTA#). In addition, the BIOS disables the PCI 9610 Master Access Enable bit (PCICR[2]).

Notes: *In Power-Saving mode, all PCI and 60x Bus Configuration cycles are granted.*

The PCI 9610 automatically performs a soft reset to the 60x Bus on D_3-to-D_0 transitions.

6.1.5 Non-D_{3cold} Wake-Up Request Example

1. The add-in card (with a PCI 9610 chip installed) is in a powered-down state.
2. The Processor CPU performs a write to the PCI 9610 PMCSR register to request a wake-up procedure.
3. As soon as the request is detected, the PCI 9610 drives PME# out to the PCI Bus.
4. The PCI Host accesses the PCI 9610 PMCSR register to disable the PME# output signal and restores the PCI 9610 to the D_0 power state.
5. The PCI 9610 completes the power management task by issuing the 60x Bus interrupt (LINTo#) to the Processor CPU, indicating that the power mode has changed.

7 COMPACTPCI HOT SWAP

The PCI 9610 is a CompactPCI Hot Swap *Ready*-compliant device.

7.1 OVERVIEW

Hot Swap is used for many CompactPCI applications. Hot Swap functionality allows the orderly insertion and removal of boards without adversely affecting system operation. This is done for repair of faulty boards or system reconfiguration. Additionally, Hot Swap provides access to Hot Swap services, allowing system reconfiguration and fault recovery to occur with no system down time and minimum operator interaction. Adapter insertion/removal logic control resides on the individual adapters. The PCI 9610 uses five pins—64EN#, BD_SEL#, CPCISW, ENUM#, and LEDon#—to implement the hardware aspects of Hot Swap functionality. The PCI 9610 uses the Hot Swap Capabilities register to implement the software aspects of Hot Swap.

To avoid confusion in the industry, Hot Swap defines three levels of compatibility:

- Hot Swap-*Capable* devices contain the minimum requirements to operate in a Hot Swap environment
- Hot Swap-*Friendly* devices contain additional functions to ease the designer's job
- Hot Swap-*Ready* devices contain all necessary functions for Hot Swap

Hot Swap-*Capable* requirements are mandatory for a device to be used in a Hot Swap environment. These requirements are attributes for which a system user must compensate using external circuitry, as follows:

- *PCI Specification r2.1* compliance
- Tolerate Vcc from early power
- Tolerate asynchronous reset
- Tolerate precharge voltage
- I/O Buffers must meet modified V/I requirements
- Limited I/O pin leakage at precharge voltage

Hot Swap-*Ready* silicon includes all required *Capable* functions and adds others from the following list. The PCI 9610 integrates these functions into the PCI silicon, thereby reducing the amount and cost of required external circuitry.

- **Incorporates Hot Swap Control/Status register (HS_CSR)**—Contained within the configuration space.
- **Incorporates an Extended Capability Pointer (ECP) mechanism**—It is required that Software retain a standard method of determining whether a specific function is designed in accordance with the specification. The Capabilities Pointer is located within standard CSR space, using a bit in the PCI Status register (offset 04h).
- Incorporates remaining software connection control resources. Provides ENUM#, Hot Swap switch, and the blue LED.

Hot Swap-*Ready* silicon includes all required *Friendly* functions and adds others from the following list. The PCI 9610 integrates these functions into the PCI silicon, thereby reducing the amount and cost of external circuitry required.

- Early Power Support.
- **Incorporates a 1V BIAS precharge voltage to the PCI I/O pins**—All PCI Bus signals are required to be precharged to a 1V BIAS through a 10K ohm resistor during the Hot Swap process. The PCI 9610 provides an internal voltage regulator to supply 1V, with a built-in 10K ohm resistor, to all required PCI I/O buffers. Other PCI signals can be precharged to V_{IO}.

The PCI 9610 is a Hot Swap-*Ready* PCI silicon device. The PCI 9610 incorporates all compliant functions defined by the *PICMG 2.1 CompactPCI Hot Swap Specification, r1.0*. The PCI 9610 incorporates LEDon#, CPCISW, BD_SEL#, and ENUM#, as well as Hot Swap Capabilities registers—HS_CNTL, HS_NEXT, and HS_CSR.

7.2 CONTROLLING CONNECTION PROCESSES

The following sections are excerpted from the *PICMG 2.1 CompactPCI Hot Swap Specification, r1.0*, and modified, as appropriate, for the PCI 9610. (Refer to the Specification for more details.)

7.2.1 Connection Control

Hardware Control provides a means for the platform to control the hardware connection process. The signals listed in the following sections must be supported on all Hot Swap boards for interoperability. Implementations on different platforms may vary.

7.2.1.1 Board Slot Control

BD_SEL#, one of the shortest pins from the CompactPCI backplane, is driven low to enable power-on. For systems not implementing hardware control, it is grounded on the backplane.

Systems implementing hardware control radially connect BD_SEL# to a Hot Swap Controller (HSC). The controller terminates the signal with a weak pulldown, and can detect board present when the board pull-up overrides the pull-down. HSC can then control the power-on process by driving BD_SEL# low.

The PCI 9610 uses the BD_SEL# signal to high-impedance all 60x Bus output buffers during the insertion and extraction process. In addition, the PCI 9610 uses BD_SEL# as a qualifier to dynamically connect 1V and VIO BIAS precharge resistors to all required PCI I/O buffers. A pull-up resistor must be provided to the BD_SEL# pin or add-in card, where the pull-up resistor is connected to an early power Power Supply, which provides for proper PCI 9610 operation. (Refer to Section 10 for precharge connections.)

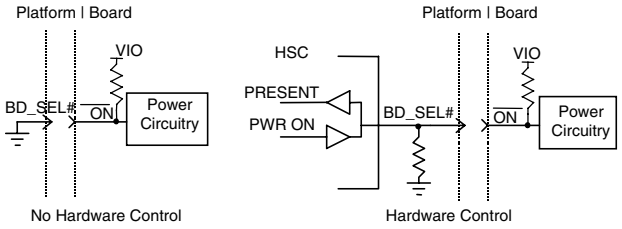


Figure 7-1. Redirection of BD_SEL#

7.2.1.2 Board Healthy

A second radial signal is used to acknowledge board health. It signals that a board is suitable to be released from reset and allowed onto the PCI Bus.

Minimally, this signal must be connected to the board's power controller "power good" status line. Use of HEALTHY# can be expanded for applications requiring additional conditions to be met for the board to be considered healthy.

On platforms that do not use Hardware Connection Control, this line is not monitored. Platforms implementing this signaling, route these signals radially to a Hot Swap Controller.

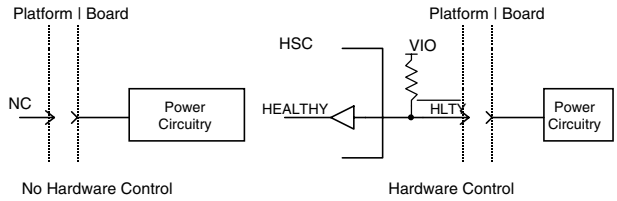


Figure 7-2. Board Healthy

7.2.1.3 Platform Reset

Reset (PCI RST#), as defined by *CompactPCI Specification*, is a bus signal on the backplane, driven by the Host. Platforms may implement this signal as a radial signal from the Hot Swap Controller to further control the electrical connection process. Platforms that maintain function of the bus signal, must *OR* the Host reset signal with the slot-specific signal.

Locally, boards must not exit reset until the H1 State is reached (healthy), and they must honor the backplane reset. The 60x Bus board reset (LRESET#) must be the logical *OR* of these two conditions. LRESET# is connected to the PCI 9610 RST# input pin.

During a BIAS voltage precharge and platform reset, in insertion and extraction procedures, all PCI I/O buffers must be in a high-impedance state. The PCI 9610 supports this condition when the Host RST# is asserted (*PCI r2.1*). To protect the 60x Bus Processor board components from early power, the PCI 9610 floats the 60x Bus I/Os. The BD_SEL# pin is used to perform the high-impedance condition on the 60x Bus. With full contact of the add-in card to the backplane, BD_SEL# is asserted which ensures that the PCI 9610 asserts the LRESET# signal to complete a 60x Bus board reset task.

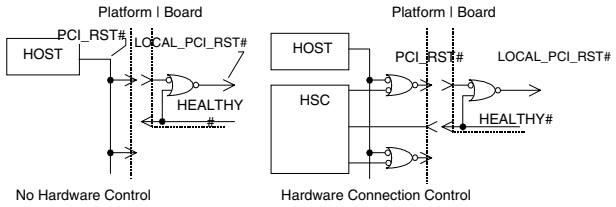


Figure 7-3. PCI Reset

7.2.2 Software Connection Control

Software Connection Control provides a means to control the Software Connection Process. Hot Swap board resources facilitate software Connection Control. Access to these resources occurs by way of the bus, using PCI protocol transfers (in-band).

These resources consist of four elements:

- ENUM# driven active indicates the need to change the Hot Swap Board state
- A switch, tied to the ejector, indicates the intent to remove a board
- LED indicates the software connection process status
- Control/Status register allows the software to interact with these resources

7.2.2.1 Ejector Switch and Blue LED

A microswitch (switch), located in the Hot Swap CompactPCI board card-ejector mechanism, is used to signal impending board removal. This signal asserts ENUM#. When the switch is activated, it is necessary to wait for the LED to light, indicating it is now okay to remove the board. The PCI 9610 implements separate control logic for the microswitch and Blue LED in two different pins (CPCISW and LEDon#, respectively).

When the ejector is opened or closed, the switch bounces for a time. The PCI 9610 uses internal debounce circuitry to clean the signal before the remainder of Hot Swap logic acknowledges it. The switch state is sampled six times, at 1 ms intervals, before it is determined to be closed or open.

The Blue “Status” LED, located on the front of the Hot Swap CompactPCI board, is illuminated when it is permissible to remove a board. The hardware connection layer provides protection for the system during all insertions and extractions. This LED indicates the system software is in a state that tolerates board extraction.

Upon insertion, the LED is automatically illuminated by the hardware until the hardware connection process completes. The LED remains *OFF* until the software uses it to indicate extraction is once again permitted.

The PCI 9610 uses an open-drain output pin to sink the external LED. The LED state is driven from the LED Software On/Off Switch bit (HS_CSR[3]).

The CPCISW input signal acknowledges the state ejector handle change to identify when a board is inserted or removed. The appropriate status bits are set (HS_CSR[7:6]).

7.2.2.2 ENUM#

ENUM# is provided to notify the Host CPU that a board was recently inserted or is about to be removed. This signal informs the CPU that system configuration changed, at which time the CPU performs necessary maintenance such as installing a device driver upon board insertion, or quiescing a device driver prior to board extraction.

ENUM# is an open collector bused signal with a pull-up on the Host bus. It may drive an interrupt (preferred) or be polled by the system software at regular intervals. The CompactPCI Hot-Plug System Driver on the system Host manages the ENUM# sensing. Full Hot Swap Boards assert ENUM# until serviced by the Hot-Plug system driver.

When a board is inserted into the system and comes out of reset, the PCI 9610 acknowledges the ejector switch state. If this switch is open (ejector handle closed), the PCI 9610 asserts the ENUM# interrupt and sets the ENUM# Status Indicator for Board Insertion bit (HS_CSR[7]). Once the Host CPU installs the proper drivers, it can logically include this board by clearing the interrupt.

When a board is about to be removed, the PCI 9610 acknowledges the ejector handle is open, asserts the ENUM# interrupt, and sets the ENUM# Status Indicator for the Board Removal bit (HS_CSR[6]). The Host then logically removes the board and turns on the LED, at which time the board can be removed from the system.

7.2.2.3 Hot Swap Control/Status Register (HS_CSR)

The PCI 9610 supports Hot Swap directly, as a control/status register is provided in Configuration space. This register is accessed through the PCI Extended Capabilities Pointer (ECP) mechanism.

The Hot Swap Control/Status register (HS_CSR) provides status read-back for the Hot-Plug System Driver to determine which board is driving ENUM#. This register is also used to control the Hot Swap Status LED on the board front panel, and to de-assert ENUM#.

7.2.2.4 Hot Swap Capabilities Register

Register 7-1. Hot Swap Capabilities Register

31 24	23 16	15 8	7 0
Reserved	Control	Next_Cap Pointer	Hot Swap ID (06h)

Hot Swap ID. Bits [7:0] (HS_CNTL[7:0]; PCI:48h, LOC:188h). These bits are set to a default value of 0x06.

Next_Cap Pointer. Bits [15:8] (HS_NEXT[7:0]; PCI:49h, LOC:189h). These bits either point to the next New Capability structure, or are set to 0 if this is the last capability in the structure.

Control. Bits [23:16] (HS_CSR[7:0]; PCI:4Ah, LOC:18Ah). This 8-bit control register is defined in the following table.

Table 7-1. Hot Swap Control

Bit	Description
23	ENUM# status—Insertion (1 = board is inserted).
22	ENUM# status—Removal (1 = board is being removed).
21	Not used.
20	Not used.
19	LED state (1 = LED on, 0 = LED off).
18	Not used.
17	ENUM# interrupt enable (1 = de-assert, 0 = enable interrupt).
16	Not used.

8 PCI VITAL PRODUCT DATA (VPD)

8.1 OVERVIEW

The Vital Product Data (VPD) function in the *PCI Specification r2.2* defines a new location and access method. It also defines the Read Only and Read/Write bits. Currently Device ID, Vendor ID, Revision ID, Class Code, Subsystem ID, and Subsystem Vendor ID are required in the Configuration Space Header and are required for basic identification of the device and device configuration. Though this information allows a device to be configured, it is not sufficient to allow a device to be uniquely identified. With the addition of VPD, optional information is provided that allows a device to be uniquely identified and tracked. These additional bits enable current and/or future support tools and reduces the total cost of ownership of PCs and systems.

This provides an alternate access method other than Expansion ROM for VPD. VPD is stored in an external serial EEPROM, which is accessed using the Configuration Space New Capabilities function.

The VPD registers—PVPDCNTL, PVPD_NEXT, PVPDAD, and PVPDATA—are not accessible for reads from the Processor Bus. It is recommended that the VPD function be exercised only from the PCI Bus.

8.1.1 VPD Capabilities Register

VPD ID. Bits [7:0] (PVPDCNTL[7:0]; PCI:4Ch, LOC:18Ch). The PCI SIG assigns these bits a value of 03h by. The VPD ID is hardcoded.

Next_Cap Pointer. Bits [15:8] (PVPD_NEXT[7:0]; PCI:4Dh, LOC:18Dh). These bits either point to the next New Capability structure, or are set to 0 if this is the last capability in the structure. The PCI 9610 defaults to 0x00. This value can be overwritten from the Processor Bus.

VPD Address. Bits [24:16] (PVPDAD[14:0]; PCI:4Eh, LOC:18Eh). These bits specify the byte address of the VPD to be accessed. All accesses are 32-bits wide; bits [17:16] must be 0, with the maximum serial EEPROM size being 4K bits. Bits [30:25] are ignored.

F. Bit 31 (PVPDAD[15]; PCI:4Eh, LOC:18Eh). This bit sets a flag to indicate when a serial EEPROM data operation is complete. For Write cycles, the four bytes of data are first written into the VPD Data bits, after which the VPD Address is written at the same time the F flag is set to 1. The F flag clears when the serial EEPROM Data transfer completes. For Read cycles, the VPD Address is written at the same time the F flag is cleared to 0. The F flag is set when four bytes of data are read from the serial EEPROM.

VPD Data. Bits [31:0] (PVPDATA[31:0]; PCI:50h, LOC:190h). The PVPDATA register is not a pure read/write register. Data read into the register depends upon the last Read operation performed in PVPDAD[15]. VPD data is written or read through this register. The least-significant byte corresponds to the VPD byte at the address specified by the VPD Address register. Four bytes are always transferred between the register and the serial EEPROM.

Register 8-1. VPD Capabilities Register

31	30 16	15 8	7 0
F	VPD Address	Next_Cap Pointer (0h)	VPD ID (03h)
VPD Data			

8.1.2 VPD Serial EEPROM Partitioning

To support VPD, the serial EEPROM is partitioned into read only and read/write sections.

8.1.3 Sequential Read Only

The first 1456 bits, 182 bytes of the serial EEPROM contain read-only information. The read-only portion of the serial EEPROM is loaded into the PCI 9610, using a sequential Read command to serial EEPROM and occurs once after power-on.

8.1.4 Random Read and Write

The PCI 9610 can read and write the read/write portion of serial EEPROM. The Serial EEPROM Starting at Lword Boundary for VPD Accesses bits (PROT_AREA[6:0]) designates this portion. This register is loaded upon power-on and can be written with a desired value starting at location 0. This provides the capability of writing the entire serial EEPROM. Writes to serial EEPROM are comprised of the following three commands:

- Write Enable
- Write Enable, followed by Write data
- Write Disable

This is done to ensure against accidental writes to the serial EEPROM. Random cycles allow VPD information to be written and read at any time.

To perform a simple VPD write to the serial EEPROM, the following steps are necessary:

1. Change the write-protected serial EEPROM address in PROT_AREA[6:0] to the desired address. 0x0000000 makes the serial EEPROM removable from the beginning.
2. Write desired data into the PVPDATA register.
3. Write destination serial EEPROM address and flag of operation, value of 1.
4. Probe a flag of operation until it changes to a 0 to ensure the write completes.

To perform a simple VPD read from serial EEPROM, the following steps are necessary:

1. Write a destination serial EEPROM address and flag of operation, value of 0.
2. Probe a flag of operation until it changes to a 1 to ensure the Read data is available.
3. Read back the PVPDATA register to see the requested data.

9 REGISTERS

Note: *Shaded letters (such as 60x Bus Miscellaneous Control 2) indicate an added or modified bit/function from the PCI 9054.*

9.1 REGISTER ADDRESS MAPPING

9.1.1 PCI Configuration Registers

Table 9-1. PCI Configuration Registers

PCI Configuration Register Address	60x Bus Access (Offset from Chip Select Address)	To ensure software compatibility with other versions of the PCI 9610 family and to ensure compatibility with future enhancements, write 0 to all unused bits. 31 30 24 23 16 15 8 7 0	PCI/60x Bus Writable	Serial EEPROM Writable
00h	00h	Device ID | Vendor ID	60x	Y
04h	04h	Status | Command	Y	N
08h	08h	Class Code | Revision ID	60x	Y [31:8]
0Ch	0Ch	BIST | Header Type | PCI Bus Latency Timer | Cache Line Size	Y [7:0] 60x	N
10h	10h	PCI Base Address 0; used for Memory-Mapped Configuration Registers (PCIBAR0)	Y	N
14h	14h	PCI Base Address 1; used for I/O-Mapped Configuration Registers (PCIBAR1)	Y	N
18h	18h	PCI Base Address 2; used for 60x Address Space 0 (PCIBAR2)	Y	N
1Ch	1Ch	PCI Base Address 3; used for 60x Address Space 1 (PCIBAR3)	Y	N
20h	20h	PCI Base Address 4; used for 60x Address Space 2 (PCIBAR4)	N	N
24h	24h	PCI Base Address 5; used for 60x Address Space 3 (PCIBAR5)	N	N
28h	28h	Cardbus CIS Pointer ***(Not supported)***	N	N
2Ch	2Ch	Subsystem ID | Subsystem Vendor ID	60x	Y
30h	30h	PCI Base Address for 60x Bus Expansion ROM	Y	N
34h	34h	***Reserved*** | Next_Cap Pointer	Y [7:0] 60x [7:0]	N
38h	38h	***Reserved***	N	N
3Ch	3Ch	Max_Lat | Min_Gnt | Interrupt Pin | Interrupt Line	Y [31:8] 60x	Y
40h	180h	Power Management Capabilities | Next_Cap Pointer | Capability ID	60x	N
44h	184h	Data | PMCSR Bridge Support Extensions | Power Management Control/Status Register	Y [15, 12:8, 1:0]	N
48h	188h	***Reserved*** | Control/Status Register | Next_Cap Pointer | Capability ID	PCI [23:16] 60x [15:0]	Y [15:0]
4Ch	18Ch	F | VPD Address | Next_Cap Pointer | VPD ID	Y [31:16], 60x [15:8]	N
50h	190h	VPD Data	Y	N

9.1.2 60x Bus Configuration Registers

Table 9-2. 60x Bus Configuration Registers

PCI (Offset from Base Address)	60x Bus Access (Offset from Chip Select Address)	To ensure software compatibility with other versions of the PCI 9610 family and to ensure compatibility with future enhancements, write 0 to all unused bits. 31 24 23 16 15 8 7 0				PCI/60x Bus Writable	Serial EEPROM Writable
00h	80h	Range for PCI-to-60x Bus Address Space 0				Y	Y
04h	84h	60x Bus Base Address (Remap) for PCI-to-60x Bus Address Space 0				Y	Y
08h	88h	Mode/DMA Arbitration				Y	Y
0Ch	8Ch	60x Bus Miscellaneous Control 2	Serial EEPROM Write-Protected Address Boundary	60x Bus Miscellaneous Control	Big/Little Endian Descriptor	Y	Y
10h	90h	Range for PCI-to-60x Bus Expansion ROM				Y	Y
14h	94h	60x Bus Base Address (Remap) for PCI-to-60x Bus Expansion ROM				Y	Y
18h	98h	60x Bus Region Descriptors (Space 0 and Expansion ROM) for PCI-to-60x Bus Accesses				Y	Y
1Ch	9Ch	Range for Direct Master-to-PCI				Y	Y
20h	A0h	60x Bus Base Address for Direct Master-to-PCI Memory				Y	Y
24h	A4h	60x Bus Base Address for Direct Master-to-PCI I/O Configuration				Y	Y
28h	A8h	PCI Base Address (Remap) for Direct Master-to-PCI				Y	Y
2Ch	ACh	PCI Configuration Address Register for Direct Master-to-PCI I/O Configuration				Y	Y
F0h	170h	Range for PCI-to-60x Bus Address Space 1				Y	Y
F4h	174h	60x Bus Base Address (Remap) for PCI-to-60x Bus Address Space 1				Y	Y
F8h	178h	60x Bus Region Descriptor (Space 1) for PCI-to-60x Bus Accesses				Y	Y
FCh	17Ch	PCI Base Dual Address Cycle (Remap) for Direct Master-to-PCI (Upper 32 bits)				Y	N
100h	1A0h	***Reserved***		PCI Arbiter Control		Y	Y
104h	1A4h	PCI Arbiter Address				N	N
108h	1A8h	Big/Little Endian Descriptor 2	60x Bus Control Register 3	60x Bus Control Register 2	60x Bus Control Register 1	Y	Y

9.1.3 Runtime Registers

Table 9-3. Runtime Registers

PCI (Offset from Base Address)	60x Bus Access (Offset from Chip Select Address)	To ensure software compatibility with other versions of the PCI 9610 family and to ensure compatibility with future enhancements, write 0 to all unused bits. 31 16 15 0		PCI/60x Bus Writable	Serial EEPROM Writable
40h	C0h	Mailbox Register 0 (refer to Note)		Y	Y
44h	C4h	Mailbox Register 1 (refer to Note)		Y	Y
48h	C8h	Mailbox Register 2		Y	N
4Ch	CCh	Mailbox Register 3		Y	N
50h	D0h	Mailbox Register 4		Y	N
54h	D4h	Mailbox Register 5		Y	N
58h	D8h	Mailbox Register 6		Y	N
5Ch	DCh	Mailbox Register 7		Y	N
60h	E0h	PCI-to-60x Doorbell Register		Y	N
64h	E4h	60x-to-PCI Doorbell Register		Y	N
68h	E8h	Interrupt Control/Status		Y	N
6Ch	ECh	Serial EEPROM Control, PCI Command Codes, User I/O Control, and Init Control		Y	N
70h	F0h	Device ID	Vendor ID	N	N
74h	F4h	***Unused***	Revision ID	N	N
78h	C0h	Mailbox Register (refer to Note)		Y	N
7Ch	C4h	Mailbox Register (refer to Note)		Y	N
110h	1B0h	Interrupt Control/Status Register 2		Y	N

Note: *Mailbox registers 0 and 1 are always accessible at addresses 78h/C0h and 7Ch/Ch4. When the I_2O function is disabled (QSR[0]=0), Mailbox registers 0 and 1 are also accessible at PCI addresses 40h and 44h for PCI 9060 compatibility. When the I_2O function is enabled (QSR[0]=1), the Inbound and Outbound Queue pointers are accessed at addresses 40h and 44h, replacing mailbox registers in PCI Address space.*

9.1.4 DMA Registers

Refer to Note following Table 9.4.

Table 9-4. DMA Registers

PCI (Offset from Base Address)	60x Bus Access (Offset from Chip Select Address)	To ensure software compatibility with other versions of the PCI 9610 family and to ensure compatibility with future enhancements, write 0 to all unused bits. 31 24 23 16 15 8 7 0	PCI/60x Bus Writable	Serial EEPROM Writable
DMA Channel 0				
80h	100h	DMA Channel 0 Mode (DMAMODE0)	Y	N
84h	104h	DMA Channel 0 PCI Address (DMAPADR0)	Y	N
88h	108h	DMA Channel 0 60x Address (DMALADR0)	Y	N
8Ch	10Ch	DMA Channel 0 Transfer Byte Count (DMASIZ0)	Y	N
90h	110h	DMA Channel 0 Descriptor Pointer (DMADPR0)	Y	N
DMA Channel 1				
94h	114h	DMA Channel 1 Mode (DMAMODE1)	Y	N
98h	118h	DMA Channel 1 PCI Address (DMAPADR1)	Y	N
9Ch	11Ch	DMA Channel 1 60x Address (DMALADR1)	Y	N
A0h	120h	DMA Channel 1 Transfer Byte Count (DMASIZ1)	Y	N
A4h	124h	DMA Channel 1 Descriptor Pointer (DMADPR1)	Y	N
A8h	128h	***Reserved*** / DMA Channel 1 Command/ Status / DMA Channel 0 Command/ Status	Y	N
ACh	12Ch	Mode/DMA Arbitration	Y	N
B0h	130h	DMA Threshold Channel 0 and 1	Y	N
B4h	134h	DMA Channel 0 PCI Dual Address Cycle (Upper 32 bits)	Y	N
B8h	138h	DMA Channel 1 PCI Dual Address Cycle (Upper 32 bits)	Y	N
DMA Channel 2				
1C0h	1C0h	DMA Channel 2 Mode (DMAMODE2)	Y	N
1C4h	1C4h	DMA Channel 2 PCI Address (DMAPADR2)	Y	N
1C8h	1C8h	DMA Channel 2 60x Address (DMALADR2)	Y	N
1CCh	1CCh	DMA Channel 2 Transfer Byte Count (DMASIZ2)	Y	N
1D0h	1D0h	DMA Channel 2 Descriptor Pointer (DMADPR2)	Y	N
DMA Channel 3				
1D4h	1D4h	DMA Channel 3 Mode (DMAMODE3)	Y	N
1D8h	1D8h	DMA Channel 3 PCI Address (DMAPADR3)	Y	N
1DCh	1DCh	DMA Channel 3 60x Address (DMALADR3)	Y	N
1E0h	1E0h	DMA Channel 3 Transfer Byte Count (DMASIZ3)	Y	N
1E4h	1E4h	DMA Channel 3 Descriptor Pointer (DMADPR3)	Y	N
1E8h	1E8h	***Reserved*** / DMA Channel 3 Command/ Status / DMA Channel 2 Command/ Status	Y	N
1ECh	1ECh	***Reserved***	N	N
1F0h	1F0h	DMA Threshold Channel 2 and 3	Y	N
1F4h	1F4h	DMA Channel 2 PCI Dual Address Cycle (Upper 32 bits)	Y	N
1F8h	1F8h	DMA Channel 3 PCI Dual Address Cycle (Upper 32 bits)	Y	N

Note: *The DMA Descriptor order depends upon the Valid Mode Enable bit setting in DMAMODEx[20], where x is the channel number—0, 1, 2, or 3.*

When Valid mode is disabled (DMAMODEx[20]=0), the register order is as follows:

DMAPADRx
DMALADRx
DMASIZx
DMADPRx

When Valid mode is enabled (DMAMODEx[20]=1), the register order is as follows:

DMASIZx
DMAPADRx
DMALADRx
DMADPRx

No other register locations are affected by DMAMODEx[20].

9.1.5 Messaging Queue Registers

Table 9-5. Messaging Queue Registers

PCI (Offset from Base Address)	60x Bus Access (Offset from Chip Select Address)	To ensure software compatibility with other versions of the PCI 9610 family and to ensure compatibility with future enhancements, write 0 to all unused bits. 31 0	PCI/60x Bus Writable	Serial EEPROM Writable
30h	B0h	Outbound Post Queue Interrupt Status	N	N
34h	B4h	Outbound Post Queue Interrupt Mask	Y	N
40h	—	Inbound Queue Port	PCI	N
44h	—	Outbound Queue Port	PCI	N
C0h	140h	Messaging Unit Configuration	Y	N
C4h	144h	Queue Base Address	Y	N
C8h	148h	Inbound Free Head Pointer	Y	N
CCh	14Ch	Inbound Free Tail Pointer	Y	N
D0h	150h	Inbound Post Head Pointer	Y	N
D4h	154h	Inbound Post Tail Pointer	Y	N
D8h	158h	Outbound Free Head Pointer	Y	N
DCh	15Ch	Outbound Free Tail Pointer	Y	N
E0h	160h	Outbound Post Head Pointer	Y	N
E4h	164h	Outbound Post Tail Pointer	Y	N
E8h	168h	Queue Status/Control	Y	N

Notes: *When I_2O messaging is enabled (QSR[0]= 1), the PCI Master (Host or another IOP) uses the Inbound Queue Port to read Message Frame Addresses (MFAs) from the Inbound Free List FIFO and to write MFAs to the Inbound Post Queue FIFO. The Outbound Queue Port reads MFAs from the Outbound Post Queue FIFO and writes MFAs to the Outbound Free List FIFO.*

Each Inbound MFA is specified by I_2O as an offset from the PCI Base Address 0 (programmed in PCIBAR0) to the start of the message frame. This means that all inbound message frames should reside in PCI Base Address 0 Memory space.

Each Outbound MFA is specified by I_2O as an offset from system address 0x00000000h. An Outbound MFA is a physical 32-bit address of the frame in shared PCI system memory.

The Inbound and Outbound queues may reside in 60x Bus Address Space 0 or 1 by programming QSR. The queues need not be in shared memory.

9.2 PCI CONFIGURATION REGISTERS

All registers may be written to or read from in Byte, Word, Lword, or Qword accesses.

Note: Reserved versus Special Reserved Bits. *For PCI 9610 purposes,* ***Reserved*** *bits are those which were reserved in the PCI 9054, or those that are reserved in the new PCI 9610 registers, but not used in the PCI 9054. Reserved bits cannot be written.* ***Special Reserved*** *bits are bits that were previously used in the PCI 9054, but are not needed in the PCI 9610. These bits can be written, but they do not carry out a function. However, they are necessary to maintain register compatibility.*

Register 9-1. (PCIIDR; PCI:00h, LOC:00h) PCI Configuration ID

Bit	Description	Read	Write	Value after Reset
15:0	**Vendor ID.** Identifies device manufacturer. Defaults to the PLX (10B5h) PCI SIG-issued Vendor ID, when no serial EEPROM is present or the serial EEPROM is blank.	Yes	60x/ Serial EEPROM	10B5h or 0h
31:16	**Device ID.** Identifies particular device. Defaults to PLX part number 9610h, when no serial EEPROM is present or the serial EEPROM is blank.	Yes	60x/ Serial EEPROM	9610h or 0h

Register 9-2. (PCICR; PCI:04h, LOC:04h) PCI Command

Bit	Description	Read	Write	Value after Reset
0	**I/O Space.** Writing 1 allows the device to respond to I/O Space accesses. Writing 0 disables the device from responding to I/O Space accesses.	Yes	Yes	0
1	**Memory Space.** Writing 1 allows the device to respond to Memory Space accesses. Writing 0 disables the device from responding to Memory Space accesses.	Yes	Yes	0
2	**Master Enable.** Writing 1 allows device to behave as a Bus Master. Writing 0 disables device from generating Bus Master accesses.	Yes	Yes	0
3	**Special Cycle.** ***Not supported.***	Yes	No	0
4	**Memory Write and Invalidate Enable.** Writing 1 enables Memory Write and Invalidate mode for Direct Master and DMA. (Refer to the DMA Mode register(s), DMAMODEx[13].)	Yes	Yes	0
5	**VGA Palette Snoop.** ***Not supported.***	Yes	No	0
6	**Parity Error Response.** Writing 0 indicates parity error is ignored and the operation continues. Writing 1 indicates parity checking is enabled.	Yes	Yes	0
7	**Wait Cycle Control.** Controls whether a device does address/data stepping. Writing 0 indicates the device never does stepping. Writing 1 indicates the device always does stepping. ***Note:*** *Hardcoded to 0.*	Yes	No	0
8	**SERR# Enable.** Writing 1 enables the SERR# driver. Writing 0 disables the SERR# driver.	Yes	Yes	0
9	**Fast Back-to-Back Enable.** Indicates what type of fast back-to-back transfers a Master can perform on the bus. Writing 1 indicates fast back-to-back transfers can occur to any agent on the bus. Writing 0 indicates fast back-to-back transfers can only occur to the same agent as in the previous cycle. ***Note:*** *Hardcoded to 0.*	Yes	No	0
15:10	***Reserved.***	Yes	No	0h

Register 9-3. (PCISR; PCI:06h, LOC:06h) PCI Status

Bit	Description	Read	Write	Value after Reset
3:0	***Reserved.***	Yes	No	0h
4	**New Capability Functions Support.** Writing 1 supports New Capabilities Functions. When enabled, the first New Capability Function ID is located at PCI Configuration offset 40h. Can be written only from the 60x Bus. Read-only from the PCI Bus.	Yes	60x	1
5	**66 MHz-Capable.** When set to 1, this device supports a 66 MHz PCI clock environment.	Yes	60x	1
6	**User Definable Functions.** When set to 1, this device supports user-definable functions. Can be written only from the 60x Bus. Read-only from the PCI Bus.	Yes	60x	0
7	**Fast Back-to-Back Capable.** Writing 1 indicates an adapter can accept fast back-to-back transactions. ***Note:*** *Hardcoded to 1.*	Yes	No	1
8	**Master Data Parity Error Detected.** Set to 1 when three conditions are met: 1) PCI 9610 asserted PERR# or acknowledged PERR# asserted; 2) PCI 9610 was the Bus Master for the operation in which the error occurred; 3) Parity Error Response bit is set (PCICR[6]=1). Writing 1 clears this bit to 0.	Yes	Yes/Clr	0
10:9	**DEVSEL# Timing.** Indicates timing for DEVSEL# assertion. Writing 01 sets this bit to medium. ***Note:*** *Hardcoded to 01.*	Yes	No	01
11	**Target Abort.** When set to 1, indicates the PCI 9610 has signaled a Target abort. Writing 1 clears this bit to 0.	Yes	Yes/Clr	0
12	**Received Target Abort.** When set to 1, indicates the PCI 9610 has received a Target Abort signal. Writing 1 clears this bit to 0.	Yes	Yes/Clr	0
13	**Received Master Abort.** When set to 1, indicates the PCI 9610 has received a Master Abort signal. Writing 1 clears this bit to 0.	Yes	Yes/Clr	0
14	**Signaled System Error.** When set to 1, indicates the PCI 9610 has reported a system error on SERR#. Writing 1 clears this bit to 0.	Yes	Yes/Clr	0
15	**Detected Parity Error.** When set to 1, indicates the PCI 9610 has detected a PCI Bus parity error, even when parity error handling is disabled (the Parity Error Response bit in the Command register is clear). One of three conditions can cause this bit to be set: 1) PCI 9610 detected a parity error during PCI Address phase; 2) PCI 9610 detected a data parity error when it is the target of a write; 3) PCI 9610 detected a data parity error when performing Master Read operation. Writing 1 clears this bit to 0.	Yes	Yes/Clr	0

Register 9-4. (PCIREV; PCI:08h, LOC:08h) PCI Revision ID

Bit	Description	Read	Write	Value after Reset
7:0	**Revision ID.** Silicon revision of the PCI 9610.	Yes	60x/ Serial EEPROM	Current Rev # (AA)

Register 9-5. (PCICCR; PCI:09-0Bh, LOC:09-0Bh) PCI Class Code

Bit	Description	Read	Write	Value after Reset
7:0	**Register Level Programming Interface.** None defined.	Yes	60x/ Serial EEPROM	0h
15:8	**Subclass Code (Other Bridge Device).**	Yes	60x/ Serial EEPROM	80h
23:16	**Base Class Code (Bridge Device).**	Yes	60x/ Serial EEPROM	06h

Register 9-6. (PCICLSR; PCI:0Ch, LOC:0Ch) PCI Cache Line Size

Bit	Description	Read	Write	Value after Reset
7:0	**System Cache Line Size.** Specified in units of 32-bit words (8 or 16 Lwords). When a size other than 8 or 16 is specified, the PCI 9610 performs Write transfers rather than Memory Write and Invalidate transfers.	Yes	Yes	0h

Register 9-7. (PCILTR; PCI:0Dh, LOC:0Dh) PCI Bus Latency Timer

Bit	Description	Read	Write	Value after Reset
7:0	**PCI Bus Latency Timer.** Specifies amount of time (in units of PCI Bus clocks) the PCI 9610, as a Bus Master, can burst data on the PCI Bus.	Yes	Yes	0h

Register 9-8. (PCIHTR; PCI:0Eh, LOC:0Eh) PCI Header Type

Bit	Description	Read	Write	Value after Reset
6:0	**Configuration Layout Type.** Specifies layout of bits 10h through 3Fh in configuration space. Only one encoding, 0h, is defined. All other encodings are ***reserved***.	Yes	60x	0h
7	**Header Type.** Writing 1 indicates multiple functions. Writing 0 indicates single function.	Yes	60x	0

Register 9-9. (PCIBISTR; PCI:0Fh, LOC:0Fh) PCI Built-In Self Test (BIST)

Bit	Description	Read	Write	Value after Reset
3:0	**BIST Pass/Fail.** Writing 0h indicates a device passed its test. Non-0h values indicate a device failed its test. Device-specific failure codes can be encoded in a non-0h value.	Yes	60x	0h
5:4	***Reserved.***	Yes	No	00
6	**PCI BIST Interrupt Enable.** The PCI Bus writes 1 to enable BIST. Generates an interrupt to the 60x Bus. The 60x Bus resets this bit when BIST is complete. The software fails the device when BIST is not complete after two seconds. Refer to the Runtime registers for Interrupt Control/Status.	Yes	Yes	0
7	**BIST Support.** Returns 1 when the device supports BIST. Returns 0 when the device is not BIST-compatible.	Yes	60x	0

Register 9-10. (PCIBAR0; PCI:10h, LOC:10h) PCI Base Address Register for Memory Accesses to 60x Bus, Runtime, and DMA Registers

Bit	Description	Read	Write	Value after Reset
0	**Memory Space Indicator.** Writing 0 indicates the register maps into Memory space. Writing 1 indicates the register maps into I/O space. ***Note:*** *Hardcoded to 0.*	Yes	No	0
2:1	**Register Location.** Values: 00—Locate anywhere in 32-bit Memory Address space 01—Locate below 1-MB Memory Address space 10—Locate anywhere in 64-bit Memory Address space 11—***Reserved*** ***Note:*** *Hardcoded to 00.*	Yes	No	00
3	**Prefetchable.** Writing 1 indicates there are no side effects on reads. Does not affect PCI 9610 operation. ***Note:*** *Hardcoded to 0.*	Yes	No	0
7:4	**Memory Base Address.** Memory base address for access to 60x Bus, Runtime, and DMA registers (requires 256 bytes). ***Note:*** *Hardcoded to 0h.*	Yes	No	0h
31:8	**Memory Base Address.** Memory base address for access to 60x Bus, Runtime, and DMA registers.	Yes	Yes	0h

Note: *For I_2O, the inbound message frame pool must reside in the address space pointed to by PCIBAR0. Message Frame Address (MFA) is defined by I_2O as offset from this base address to the start of the message frame.*

Register 9-11. (PCIBAR1; PCI:14h, LOC:14h) PCI Base Address Register for I/O Accesses to 60x Bus, Runtime, and DMA

Bit	Description	Read	Write	Value after Reset
0	**Memory Space Indicator.** Writing 0 indicates the register maps into Memory space. Writing 1 indicates the register maps into I/O space. ***Note:*** *Hardcoded to 1.*	Yes	No	1
1	***Reserved.***	Yes	No	0
7:2	**I/O Base Address.** Base Address for I/O access to 60x Bus, Runtime, and DMA registers (requires 256 bytes). ***Note:*** *Hardcoded to 0h.*	Yes	No	0h
31:8	**I/O Base Address.** Base Address for I/O access to 60x Bus, Runtime, and DMA registers. PCIBAR1 can be enabled or disabled by setting or clearing the Base Address Register 1 Enable bit (LMISC1[0]).	Yes	Yes	0h

Register 9-12. (PCIBAR2; PCI:18h, LOC:18h) PCI Base Address Register for Memory Accesses to 60x Address Space 0

Bit	Description	Read	Write	Value after Reset
0	**Memory Space Indicator.** Writing 0 indicates the register maps into Memory space. Writing 1 indicates the register maps into I/O space. (Specified in LAS0RR register.)	Yes	No	0
2:1	**Register Location (When Memory Space).** Values: 00—Locate anywhere in 32-bit Memory Address space 01—Locate below 1-MB Memory Address space 10—Locate anywhere in 64-bit Memory Address space 11—***Reserved*** (Specified in LAS0RR register.) When I/O Space, bit 1 is always 0 and bit 2 is included in the base address.	Yes	Mem: No I/O: Bit 1 No, Bit 2 Yes	00
3	**Prefetchable (When Memory Space).** Writing 1 indicates there are no side effects on reads. Reflects the value of LAS0RR[3] and provides only status to the system. Does not affect PCI 9610 operation. The associated Bus Region Descriptor register controls prefetching functions of this address space. (Specified in LAS0RR register.) When I/O Space, bit 3 is included in the base address.	Yes	Mem: No I/O: Yes	0
31:4	**Memory Base Address.** Memory base address for access to 60x Bus Address Space 0. PCIBAR2 can be enabled or disabled by setting or clearing the Space 0 Enable bit (LAS0BA[0]).	Yes	Yes	0h

Register 9-13. (PCIBAR3; PCI:1Ch, LOC:1Ch) PCI Base Address Register for Memory Accesses to 60x Address Space 1

Bit	Description	Read	Write	Value after Reset
0	**Memory Space Indicator.** Writing 0 indicates the register maps into Memory space. Writing 1 indicates the register maps into I/O space. (Specified in LAS1RR register.)	Yes	No	0
2:1	**Register Location.** Values: 00—Locate anywhere in 32-bit Memory Address space 01—Locate below 1-MB Memory Address space 10—Locate anywhere in 64-bit Memory Address space 11—***Reserved*** (Specified in LAS1RR register.) When I/O Space, bit 1 is always 0 and bit 2 is included in the base address.	Yes	Mem: No I/O: Bit 1 No, Bit 2 Yes	00
3	**Prefetchable (When Memory Space).** Writing 1 indicates there are no side effects on reads. Reflects value of LAS1RR[3] and only provides status to the system. Does not affect PCI 9610 operation. The associated Bus Region Descriptor register controls prefetching functions of this address space. (Specified in LAS1RR register.) When I/O Space, bit 3 is included in base address.	Yes	Mem: No I/O: Yes	0
31:4	**Memory Base Address.** Memory base address for access to 60x Bus Address Space 1. PCIBAR3 can be enabled or disabled by setting or clearing the Space 1 Enable bit (LAS1BA[0]). When QSR[0]=1, PCIBAR3 returns 0h.	Yes	Yes	0h

Register 9-14. (PCIBAR4; PCI:20h, LOC:20h) PCI Base Address

Bit	Description	Read	Write	Value after Reset
31:0	***Reserved.***	Yes	No	0h

Register 9-15. (PCIBAR5; PCI:24h, LOC:24h) PCI Base Address

Bit	Description	Read	Write	Value after Reset
31:0	***Reserved.***	Yes	No	0h

Register 9-16. (PCICIS; PCI:28h, LOC:28h) PCI Cardbus CIS Pointer

Bit	Description	Read	Write	Value after Reset
31:0	**Cardbus Information Structure Pointer for PC Cards.** ***Not supported.***	Yes	No	0h

Register 9-17. (PCISVID; PCI:2Ch, LOC:2Ch) PCI Subsystem Vendor ID

Bit	Description	Read	Write	Value after Reset
15:0	**Subsystem Vendor ID.** Unique add-in board Vendor ID.	Yes	60x/ Serial EEPROM	10B5h

Register 9-18. (PCISID; PCI:2Eh, LOC:2Eh) PCI Subsystem ID

Bit	Description	Read	Write	Value after Reset
15:0	**Subsystem ID.** Unique add-in board Device ID.	Yes	60x/ Serial EEPROM	9610h

Register 9-19. (PCIERBAR; PCI:30h, LOC:30h) PCI Expansion ROM Base

Bit	Description	Read	Write	Value after Reset
0	**Address Decode Enable.** Writing 1 indicates a device accepts accesses to the Expansion ROM space. Writing 0 indicates a device does not accept accesses to Expansion ROM space. Should be set to 0 when there is no Expansion ROM. Works in conjunction with EROMRR[0].	Yes	Yes	0
10:1	***Reserved.***	Yes	No	0h
31:11	**Expansion ROM Base Address (upper 21 bits).**	Yes	Yes	0h

Register 9-20. (CAP_PTR; PCI:34h, LOC:34h) New Capability Pointer

Bit	Description	Read	Write	Value after Reset
7:0	**New Capability Pointer.** Offset into PCI Configuration Space for the location of the first item in the New Capabilities Linked List.	Yes	60x	40h
31:8	***Reserved.***	Yes	No	0h

Register 9-21. (PCIILR; PCI:3Ch, LOC:3Ch) PCI Interrupt Line

Bit	Description	Read	Write	Value after Reset
7:0	**Interrupt Line Routing Value.** Value indicates which input of the system interrupt controller(s) is connected to each device interrupt line.	Yes	Yes	0h

Register 9-22. (PCIIPR; PCI:3Dh, LOC:3Dh) PCI Interrupt Pin

Bit	Description	Read	Write	Value after Reset
7:0	**Interrupt Pin Register.** Indicates which interrupt pin the device uses. The following values are decoded (the PCI 9610 supports only INTA#): 0 = No Interrupt pin 1 = INTA# 2 = INTB# 3 = INTC# 4 = INTD#	Yes	60x/ Serial EEPROM	1h

Register 9-23. (PCIMGR; PCI:3Eh, LOC:3Eh) PCI Min_Gnt

Bit	Description	Read	Write	Value after Reset
7:0	**Min_Gnt.** Specifies how long a burst period device needs, assuming a clock rate of 33 MHz. Value is a multiple of 1/4 µs increments.	Yes	60x/ Serial EEPROM	0h

Register 9-24. (PCIMLR; PCI:3Fh, LOC:3Fh) PCI Max_Lat

Bit	Description	Read	Write	Value after Reset
7:0	**Max_Lat.** Specifies how often the device must gain access to the PCI Bus. Value is a multiple of 1/4 µs increments.	Yes	60x/ Serial EEPROM	0h

Register 9-25. (PMCAPID; PCI:40h, LOC:180h) Power Management Capability ID

Bit	Description	Read	Write	Value after Reset
7:0	**Power Management Capability ID.**	Yes	No	1h

Register 9-26. (PMNEXT; PCI:41h, LOC:181h) Power Management Next Capability Pointer

Bit	Description	Read	Write	Value after Reset
7:0	**Next_Cap Pointer.** Points to the first location of the next item in the capabilities linked list. When Power Management is the last item in the list, then this register should be set to 0.	Yes	60x	48h

Register 9-27. (PMC; PCI:42h, LOC:182h) Power Management Capabilities

Bit	Description	Read	Write	Value after Reset
2:0	**Version.** Writing 1 indicates this function complies with *PCI Power Management Interface Specification r1.1*.	Yes	60x	001
3	**PCI Clock Required for PME# Signal.** When set to 1, indicates a function relies on the presence of the PCI clock for PME# operation. The PCI 9610 does not require the PCI clock for PME#, so this bit should be set to 0.	Yes	60x	0
4	**Auxiliary Power Source.** Because the PCI 9610 does not support PME# while in a D_{3cold} state, this bit is always set to 0.	Yes	No	0
5	**DSI.** When set to 1, the PCI 9610 requires special initialization following a transition to a D_0 uninitialized state before a generic class device driver is able to use it.	Yes	60x	0
8:6	***Reserved.***	Yes	No	000
9	**D_1_Support.** When set to 1, the PCI 9610 supports the D_1 power state.	Yes	60x	0
10	**D_2_Support.** When set to 1, the PCI 9610 supports the D_2 power state.	Yes	60x	0
15:11	**PME_Support.** Indicates power states in which the PCI 9610 asserts PME#. Value — Description XXXX1 — PME# can be asserted from D_0 XXX1X — PME# can be asserted from D_1 XX1XX — PME# can be asserted from D_2 X1XXX — PME# can be asserted from D_{3hot} 1XXXX — PME# can be asserted from D_{3cold}	Yes	60x	0h

Register 9-28. (PMCSR; PCI:44h, LOC:184h) Power Management Control/Status

Bit	Description	Read	Write	Value after Reset
1:0	**Power State.** Determines or changes the current power state. Value State 00 D_0 01 D_1 10 D_2 11 D_{3hot} Transition from a D_{3hot} state to a D_0 state causes a soft reset. Should only be initiated from the PCI Bus because the 60x Bus interface is reset during a soft reset. In a D_{3hot} state, PCI Memory and I/O accesses, as well as PCI interrupts are disabled, and only configuration is allowed. The same is true for the D_2 state when the corresponding D_2_Support pin is set.	Yes	Yes	00
7:2	***Reserved.***	Yes	No	0h
8	**PME_En.** Writing 1 enables PME# to be asserted.	Yes	Yes	0
12:9	**Data_Select.** Selects which data to report through the Data register and Data_Scale bits.	Yes	Yes	0h
14:13	**Data_Scale.** Indicates the scaling factor to use when interpreting the Data register value. Value and meaning of this bit depends on the data value selected by the Data_Select bit. When the Processor Bus CPU initializes the Data_Scale values, the Data_Select bit must be used to determine which Data_Scale value it is writing. For Power Consumed and Power Dissipated data, the following scale factors are used. Unit values are in watts. Value Scale 0 Unknown 1 0.1x 2 0.01x 3 0.001x	Yes	60x	00
15	**PME_Status.** Indicates PME# is being driven when the PME_En bit is set (PMCSR[8]=1). Writing 1 from the 60x Bus sets this bit; writing 1 from the PCI Bus clears this bit to 0. Depending on the current power state, set only when the appropriate PME_Support bit(s) is set (PMC[15:11]=1).	Yes	60x/Set, PCI/Clr	0

Register 9-29. (PMCSR_BSE; PCI:46h, LOC:186h) PMCSR Bridge Support Extensions

Bit	Description	Read	Write	Value after Reset
7:0	***Reserved.***	Yes	No	0h

Register 9-30. (PMDATA; PCI:47h, LOC:187h) Power Management Data

Bit	Description	Read	Write	Value after Reset
7:0	**Power Management Data.** Provides operating data, such as power consumed or heat dissipation. Data returned is selected by the Data_Select bit(s) (PMCSR[12:9]) and scaled by the Data_Scale bit(s) (PMCSR[14:13]). Data_Select — Description 0 — D_0 Power Consumed 1 — D_1 Power Consumed 2 — D_2 Power Consumed 3 — D_3 Power Consumed 4 — D_0 Power Dissipated 5 — D_1 Power Dissipated 6 — D_2 Power Dissipated 7 — D_{3hot} Power Dissipated	Yes	60x	0h

Register 9-31. (HS_CNTL; PCI:48h, LOC:188h) Hot Swap Control

Bit	Description	Read	Write	Value after Reset
7:0	**Hot Swap ID.**	Yes	60x/ Serial EEPROM	06h

Register 9-32. (HS_NEXT; PCI:49h, LOC:189h) Hot Swap Next Capability Pointer

Bit	Description	Read	Write	Value after Reset
7:0	**Next_Cap Pointer.** Points to the first location of the next item in the capabilities linked list. When Hot Swap is the last item in the list, then this register should be set to 0.	Yes	60x/ Serial EEPROM	4Ch

Register 9-33. (HS_CSR; PCI:4Ah, LOC:18Ah) Hot Swap Control/Status

Bit	Description	Read	Write	Value after Reset
0	***Reserved.***	Yes	No	0
1	**ENUM# Interrupt Clear.** Writing 0 enables the interrupt. Writing 1 clears the interrupt.	Yes	Yes/Clr	0
2	***Reserved.***	Yes	No	0
3	**LED Software On/Off Switch.** Writing 1 turns on the LED. Writing 0 turns off the LED.	Yes	PCI	0
4	***Reserved.***	Yes	No	0
5	***Reserved.***	Yes	No	0
6	**Board Removal ENUM# Status Indicator.** Writing 1 reports the ENUM# assertion for removal process.	Yes	Yes	0
7	**Board Insertion ENUM# Status Indicator.** Writing 1 reports the ENUM# assertion for insertion process.	Yes	Yes	0
15:8	***Reserved.***	Yes	No	0h

Register 9-34. (PVPDCNTL; PCI:4Ch, LOC:18Ch) PCI Vital Product Data Control

Bit	Description	Read	Write	Value after Reset
7:0	**VPD ID.** Capability ID = 03h for VPD.	PCI	No	03h

Register 9-35. (PVPD_NEXT; PCI:4Dh, LOC:18Dh) PCI Vital Product Data Next Capability Pointer

Bit	Description	Read	Write	Value after Reset
7:0	**Next_Cap Pointer.** Points to first location of next item in the capabilities linked list. VPD is the last item in the capabilities linked list. This register is set to 0h.	PCI	60x	0h

Register 9-36. (PVPDAD; PCI:4Eh, LOC:18Eh) PCI Vital Product Data Address

Bit	Description	Read	Write	Value after Reset
14:0	**VPD Address.** Byte address of the VPD address to be accessed. Supports a 2K- or 4K-bit serial EEPROM.	PCI	Yes	0h
15	**F.** Flag used to indicate when the data transfer between PVPDATA and the storage component is complete. Writing 0 along with the VPD address causes a read of VPD information into PVPDATA. The hardware sets this bit to 1 when the VPD Data transfer is complete. Writing 1 along with the VPD address causes a write of VPD information from PVPDATA into a storage component. The hardware sets this bit to 0 after the Write operation is complete.	PCI	Yes	0

Register 9-37. (PVPDATA; PCI:50h, LOC:190h) PCI VPD Data

Bit	Description	Read	Write	Value after Reset
31:0	**VPD Data.**	PCI	Yes	0h

9.3 60X BUS CONFIGURATION REGISTERS

Register 9-38. (LAS0RR; PCI:00h, LOC:80h) 60x Bus Address Space 0 Range Register for PCI-to-60x Bus

Bit	Description	Read	Write	Value after Reset
0	**Memory Space Indicator.** Writing 0 indicates 60x Bus Address Space 0 maps into PCI Memory space. Writing 1 indicates 60x Bus Address Space 0 maps into PCI I/O space.	Yes	Yes	0
2:1	When mapped into Memory space, encoding is as follows: 2/1 Meaning 0 0 Locate anywhere in 32-bit PCI Address space 0 1 Locate below 1 MB in PCI Address space 1 0 Locate anywhere in 64-bit PCI Address space 1 1 ***Reserved*** When mapped into I/O space, bit 1 must be set to 0. Bit 2 is included with bits [31:3] to indicate the decoding range.	Yes	Yes	00
3	When mapped into Memory space, writing 1 indicates reads are prefetchable (does not affect PCI 9610 operation, but is used for system status). When mapped into I/O space, it is included with bits [31:2] to indicate the decoding range.	Yes	Yes	0
31:4	Specifies which PCI Address bits to use for decoding a PCI access to 60x Bus Space 0. Each bit corresponds to a PCI Address bit. Bit 31 corresponds to address bit 31. Write 1 to all bits that must be included in decode and 0 to all others (used in conjunction with PCIBAR2). Default is 1 MB. ***Notes:*** *Range (**not** Range register) must be power of 2. "Range register value" is inverse of range.* *All I/O spaces should be limited to 256 bytes per PCI r2.1 spec.*	Yes	Yes	FFF0000h

Register 9-39. (LAS0BA; PCI:04h, LOC:84h) 60x Bus Address Space 0 60x Bus Base Address (Remap)

Bit	Description	Read	Write	Value after Reset
0	**Space 0 Enable.** Writing 1 enables decoding of PCI addresses for Direct Slave access to 60x Bus Space 0. Writing 0 disables decoding.	Yes	Yes	0
1	***Reserved.***	Yes	No	0
3:2	When 60x Bus Space 0 is mapped into Memory space, these bits are not used. When mapped into I/O space, these bits are included with bits [31:4] for remapping.	Yes	Yes	00
31:4	**Remap PCI Address to 60x Bus Address Space 0 into 60x Bus Address Space.** Bits in this register remap (replace) the PCI Address bits used in decode as 60x Bus Address bits. ***Note:*** *Remap Address value must be a multiple of the Range (**not** the Range register).*	Yes	Yes	0h

Register 9-40. (MARBR; PCI:08h or ACh, LOC:88h or 12Ch) Mode/DMA Arbitration

Bit	Description	Read	Write	Value after Reset
7:0	**60x Bus Latency Timer.** Used only in PLX Extended mode, and when the PCI 9610 is a 60x Bus Master. The maximum number of LCLK clocks that DBB# asserted by the PCI 9610. Enabled by MARBR[16].	Yes	Yes	0h
15:8	**60x Bus Pause Timer.** Used only in PLX Extended mode. Number of LCLK clock cycles to occur before re-asserting DBB#, after the completion of the previous Address and Data tenure. The Pause Timer is valid only during DMA. Enabled by MARBR[17].	Yes	Yes	0h
16	**60x Bus Latency Timer Enable.** Writing 1 enables the latency timer. Writing 0 disables the latency timer.	Yes	Yes	0
17	**60x Bus Pause Timer Enable.** Writing 1 enables the pause timer. Writing 0 disables the pause timer.	Yes	Yes	0
18	***Special Reserved.***	Yes	Yes	0
20:19	**DMA Highest Priority Channel.** When LMISC2[6]=0, these bits have no meaning, and should remain as 0. When LMISC2[6]=1, then these two bits select which channel has the highest to lowest priority among the channels. **Value** — **Priorities Assigned in Descending Order** 00 — Channel 0, 1, 2, 3 01 — Channel 1, 2, 3, 0 10 — Channel 2, 3, 0, 1 11 — Channel 3, 0, 1, 2	Yes	Yes	00
21	***Special Reserved.***	Yes	Yes	1
22	**Direct Slave LOCK# Enable.** Writing 1 enables Direct Slave locked sequences. Writing 0 disables Direct Slave locked sequences.	Yes	Yes	0
23	**PCI Request Mode.** Writing 1 causes the PCI 9610 to de-assert REQ# when it asserts FRAME# during a Bus Master cycle. Writing 0 causes the PCI 9610 to leave REQ# asserted for the entire Bus Master cycle.	Yes	Yes	0
24	**Delayed Read Mode.** When set to 1, the PCI 9610 operates in Delayed Transaction mode for Direct Slave reads. The PCI 9610 issues a Retry to the PCI Host and prefetches Read data.	Yes	Yes	0
25	**PCI Read No Write Mode.** Writing 1 forces a Retry on writes when a read is pending. Writing 0 allows writes to occur while a read is pending.	Yes	Yes	0
26	**PCI Read with Write Flush Mode.** Writing 1 submits a request to flush a pending Read cycle when a Write cycle is detected. Writing 0 submits a request to not effect pending reads when a Write cycle occurs (*PCI Specification r2.1* compatible).	Yes	Yes	0
27	***Special Reserved.***	Yes	Yes	0
28	**PCI Read No Flush Mode.** Writing 1 submits a request to not flush the Read FIFO when the PCI Read cycle completes (Read Ahead mode). Writing 0 submits a request to flush the Read FIFO when a PCI Read cycle completes.	Yes	Yes	0
29	When set to 0, reads from the PCI Configuration register address 00h return the Device ID and Vendor ID. When set to 1, reads from the PCI Configuration register address 00h return the Subsystem ID and Subsystem Vendor ID.	Yes	Yes	0
30	**FIFO Full Status Flag.** When set to 1, the Direct Master Write FIFO is almost full. Reflects the MDREQ# pin value.	Yes	No	0
31	***Special Reserved.***	Yes	Yes	0

Register 9-41. (BIGEND; PCI:0Ch, LOC:8Ch) Big/Little Endian Descriptor

Bit	Description	Read	Write	Value after Reset
0	**Configuration Register Big Endian Mode (Address Invariance).** Writing 1 specifies use of Big Endian data ordering for 60x Bus accesses to the Configuration registers. Writing 0 specifies Little Endian ordering. Big Endian mode can be specified for Configuration register accesses by asserting BIGEND# during the Address phase of the access.	Yes	Yes	0
1	**Direct Master Big Endian Mode (Address Invariance).** Writing 1 specifies use of Big Endian data ordering for Direct Master accesses. Writing 0 specifies Little Endian ordering. Big Endian mode can be specified for Direct Master accesses by asserting BIGEND# during the Address phase of the access.	Yes	Yes	0
2	**Direct Slave Address Space 0 Big Endian Mode (Address Invariance).** Writing 1 specifies use of Big Endian data ordering for Direct Slave accesses to 60x Bus Address Space 0. Writing 0 specifies Little Endian ordering.	Yes	Yes	0
3	**Direct Slave Address Expansion ROM 0 Big Endian Mode (Address Invariance).** Writing 1 specifies use of Big Endian data ordering for Direct Slave accesses to Expansion ROM. Writing 0 specifies Little Endian ordering.	Yes	Yes	0
4	**Big Endian Byte Lane Mode.** Writing 0 indicates byte lane [0:7] is used for an 8-bit bus, and writing 1 indicates byte lane [56:63] is used for an 8-bit bus. For the 16-, 32- and 64-bit buses, the byte lanes [0:15], [0:31] and [0:63] are used respectively, and there are no other options.	Yes	Yes	0
5	**Direct Slave Address Space 1 Big Endian Mode (Address Invariance).** Writing 1 specifies use of Big Endian data ordering for Direct Slave accesses to 60x Bus Address Space 1. Writing 0 specifies Little Endian ordering.	Yes	Yes	0
6	**DMA Channel 1 Big Endian Mode (Address Invariance).** Writing 1 specifies use of Big Endian data ordering for DMA Channel 1 accesses to the 60x Bus Address space. Writing 0 specifies Little Endian ordering.	Yes	Yes	0
7	**DMA Channel 0 Big Endian Mode (Address Invariance).** Writing 1 specifies use of Big Endian data ordering for DMA Channel 0 accesses to the 60x Bus Address space. Writing 0 specifies Little Endian ordering.	Yes	Yes	0

Register 9-42. (LMISC1; PCI:0Dh, LOC:8Dh) 60x Bus Miscellaneous Control 1

Bit	Description	Read	Write	Value after Reset
0	**Base Address Register 1 Enable.** When set to 1, the Base Address 1 Register for I/O accesses to Configuration registers is enabled. When set to 0, the Base Address 1 Register for I/O accesses to Configuration registers is disabled.	Yes	Yes	1
1	**Base Address Register 1 Shift.** When the Base Address Register 1 Enable bit is set to low, and this bit is set to 0, then PCIBAR2 and PCIBAR3 remain at PCI Configuration addresses 18h and 1Ch. When Base Address Register 1 Enable bit is set to low, and this bit is set to 1, then PCIBAR2 (60x Bus Address Space 0) and PCIBAR3 (60x Bus Address Space 1) are shifted to become PCIBAR1 and PCIBAR2 at PCI Configuration addresses 14h and 18h. Set when a blank region in Base Address Register Space could not be accepted by the system BIOS.	Yes	Yes	0
2	**60x Bus Init Status.** Writing 1 indicates 60x Bus Init done. Responses to PCI accesses are Retrys until this bit is set. When the PCI 9610 has a blank serial EEPROM attached, the CPU on the Processor Bus must set the 60x Bus Init Status bit to 1.	Yes	60x/ Serial EEPROM	0
3	**Flush Direct Master (PCI Initiator) Write FIFO.** When set to 1, the PCI 9610 flushes Write data in the Direct Master Write FIFO after encountering a PCI Master abort. Writing 0 in this bit indicates no flushing of the Write FIFO.	Yes	Yes	0
4	**60x Bus Direct Master Delayed Read Enable.** Writing 1 enables the PCI 9610 to operate in Delayed Transaction mode for Direct Master reads. The PCI 9610 issues ARTRY# to the 60x Bus Master and prefetches Read data from the PCI Bus.	Yes	Yes	0
5	**TEA# Input Interrupt Mask.** When set to 1, TEA# input causes SERR# output on the PCI Bus when enabled (PCICR[8]=1) and the Signaled System Error bit is set (PCISR[14]=1). Writing 0 masks the TEA# input to create SERR#. The SERR# Status bit is set in both cases.	Yes	Yes	0
6	**Direct Master Write FIFO Almost Full ARTRY# Enable.** When set to 0, the PCI 9610 disables issuing of ARTRY#, regardless of whether the Direct Master Write FIFO is almost full. When set to 1, the PCI 9610 issues ARTRY#, when the Direct Master Write FIFO is almost full.	Yes	Yes	0
7	**Disconnect with Flush Read FIFO.** Value of 1 causes a disconnect with flushing of the Read FIFO in Delayed Read mode (MARBR[24]). Value of 0 causes a disconnect without flushing the Read FIFO (as a Retry).	Yes	Yes	0

Register 9-43. (PROT_AREA; PCI:0Eh, LOC:8Eh) Serial EEPROM Write-Protected Address Boundary

Bit	Description	Read	Write	Value after Reset
6:0	**Serial EEPROM Starting at Lword Boundary (48 Lwords = 192 bytes) for VPD Accesses.** Any serial EEPROM address below this boundary is read-only. ***Note:*** *Anything below the programmed address may contain the PCI 9610 Configuration data.*	Yes	Yes	0110000
7	***Reserved.***	Yes	No	0

Register 9-44. (LMISC2; PCI:0Fh, LOC:8Fh) 60x Bus Miscellaneous Control 2

Bit	Description	Read	Write	Value after Reset
1:0	***Special Reserved.***	Yes	Yes	10
4:2	**Direct Slave Write Delay.** Delay in LCLK of BR#. Values: 0 = 0 LCLK 2 = 8 LCLK 4 =20 LCLK 6 = 28 LCLK 1 = 4 LCLK 3 = 16 LCLK 5 = 24 LCLK 7 = 32 LCLK	Yes	Yes	000
5	**Direct Slave Write FIFO Full Condition.** Value of 1 guarantees that when the Direct Slave Write FIFO is full with Direct Slave Write data, there is always one location remaining empty for the Direct Slave Read address to be accepted by the PCI 9610. Value of 0 Retries all Direct Slave Read accesses when the Direct Slave Write FIFO is full with Direct Slave Write data.	Yes	Yes	0
6	**DMA Channel Priority.** The PCI 9610 contains an internal arbiter to arbitrate PCI and 60x Bus access among the four DMA channels. Sets the type of DMA channel arbitration as follows: **Value** — **DMA Channel Arbitration** 0 — Round-robin priority scheme 1 — One of the four DMA channels is assigned the highest priority over the other three. The channel is selected in MARBR[20:19].	Yes	Yes	0
7	***Reserved.***	Yes	No	0

Register 9-45. (EROMRR; PCI:10h, LOC:90h) Expansion ROM Range

Bit	Description	Read	Write	Value after Reset
0	**Address Decode Enable.** Bit 0 can only be enabled from the serial EEPROM. To disable, set the PCI Expansion ROM Address Decode Enable bit to 0 (PCIERBAR[0]=0).	Yes	Serial EEPROM Only	0
10:1	***Reserved.***	Yes	No	0h
31:11	Specifies which PCI Address bits to use for decoding a PCI-to-60x Bus Expansion ROM. Each bit corresponds to a PCI Address bit. Bit 31 corresponds to address bit 31. Write 1 to all bits that must be included in decode and 0 to all others (used in conjunction with PCIERBAR). Default is 64 kilobits. ***Note:*** *Range (**not** Range register) must be power of 2. "Range register value" is inverse of range.*	Yes	Yes	FFFF00h

Register 9-46. (EROMBA; PCI:14h, LOC:94h) Expansion ROM 60x Bus Base Address (Remap)

Bit	Description	Read	Write	Value after Reset
0	**Backoff Timer.** Applies only to the PLX Extended Burst Mode. When this bit is set to 0, the backoff timer is disabled. When this bit is set to 1, the backoff timer is started when a: • Direct Slave access is pending • PLX Extended Burst Direct Master is in progress, addressing the PCI 9610 as a slave The backoff timer is a fixed value of 64 LCLKs. When the Direct Master transfer has not completed before the timer times out, the PCI 9610 asserts a Slave termination by asserting PSDVAL# and TA# in the same clock cycle.	Yes	Yes	0
5:1	***Special Reserved.***	Yes	No	0h
10:6	***Reserved.***	Yes	No	0h
31:11	**Remap PCI Expansion ROM Space into 60x Bus Address Space.** Bits in this register remap (replace) the PCI Address bits used in decode as 60x Bus Address bits. ***Note:*** *Remap Address value must be a multiple of the Range (**not** the Range register).*	Yes	Yes	0h

Register 9-47. (LBRD0; PCI:18h, LOC:98h) 60x Bus Address Space 0/Expansion ROM Bus Region Descriptor

Bit	Description	Read	Write	Value after Reset
1:0	**Memory Space 0 60x Bus Width.** Writing 00 indicates an 8-bit bus width. Writing 01 indicates a 16-bit bus width. Writing 10 or 11 indicates a 32-bit bus width.	Yes	Yes	10
5:2	***Special Reserved.***	Yes	Yes	0h
6	***Special Reserved.***	Yes	Yes	1
7	***Special Reserved.***	Yes	Yes	0
8	**Memory Space 0 Prefetch Disable.** When mapped into Memory space, writing 0 enables Read prefetching. Writing 1 disables prefetching. When prefetching is disabled, the PCI 9610 disconnects after each Memory read.	Yes	Yes	0
9	**Expansion ROM Space Prefetch Disable.** Writing 0 enables Read prefetching. Writing 1 disables prefetching. When prefetching is disabled, the PCI 9610 disconnects after each Memory read.	Yes	Yes	0
10	**Prefetch Counter Enable.** When set to 1 and Memory prefetching is enabled, the PCI 9610 prefetches up to the number of Lwords specified in the prefetch counter. When set to 0, the PCI 9610 ignores the count and continues prefetching until it is terminated by the PCI Bus.	Yes	Yes	0
14:11	**Prefetch Counter.** Number of Qwords to prefetch during Memory Read cycles (0-15). A count of zero selects a prefetch of 16 Qwords.	Yes	Yes	0h
15	**60x Bus Direct Slave Space 0 Assert GBL#.** Writing 1 allows the PCI 9610 to assert GBL# when Address Space 0 is accessed. Writing 0 prevents the PCI 9610 from asserting GBL# when Address Space 0 is accessed.	Yes	No	0
17:16	**Expansion ROM Space Mode Bus Width.** Writing 00 indicates an 8-bit bus width. Writing 01 indicates a 16-bit bus width. Writing 10 indicates a 64-bus width. Writing 11 indicates a 32-bit bus width.	Yes	Yes	11
21:18	***Special Reserved.***	Yes	Yes	0h
22	***Special Reserved.***	Yes	Yes	1
23	***Special Reserved.***	Yes	Yes	0
24	**Memory Space 0 Mode Burst Enable.** Writing 1 enables a PLX Extended Burst. Writing 0 disables a PLX Extended Burst.	Yes	Yes	0
25	**Extra Long Load from Serial EEPROM.** Writing 1 loads the Subsystem ID and 60x Bus Address Space 1 registers. Writing 0 indicates not to load them.	Yes	Serial EEPROM Only	0
26	**Expansion ROM Space PLX Extended Burst Enable.** Writing 1 enables a PLX Extended Burst. Writing 0 disables a PLX Extended Burst.	Yes	Yes	0
27	**Direct Slave PCI Write Mode.** Writing 0 indicates the PCI 9610 should disconnect when the Direct Slave Write FIFO is full. Writing 1 indicates the PCI 9610 should de-assert TRDY# when the Direct Slave Write FIFO is full.	Yes	Yes	0
31:28	**Direct Slave Retry Delay Clocks.** Contains the value (multiplied by 8) of the number of PCI Bus clocks after receiving a PCI-to-60x Bus Read or Write access and not successfully completing a transfer. Pertains only to Direct Slave writes when the Direct Slave PCI Write Mode bit is set (bit [27]=1).	Yes	Yes	4h (32 clocks)

Register 9-48. (DMRR; PCI:1Ch, LOC:9Ch) 60x Bus Range Register for Direct Master-to-PCI

Bit	Description	Read	Write	Value after Reset
15:0	***Reserved*** (64-KB increments).	Yes	No	0h
31:16	Specifies which 60x Bus Address bits to use for decoding a 60x-to-PCI Bus access. Each bit corresponds to a PCI Address bit. Bit 31 corresponds to address bit 31. Write 1 to all bits that must be included in decode and 0h to all others. ***Note:*** *Range (**not** Range register) must be power of 2. "Range register value" is inverse of range.*	Yes	Yes	0h

Register 9-49. (DMLBAM; PCI:20h, LOC:A0h) 60x Bus Base Address Register for Direct Master-to-PCI Memory

Bit	Description	Read	Write	Value after Reset
15:0	***Reserved.***	Yes	No	0h
31:16	Assigns a value to bits to use for decoding 60x-to-PCI Memory accesses. ***Note:*** *60x Bus Base Address value must be a multiple of the Range (**not** the Range register).*	Yes	Yes	0h

Register 9-50. (DMLBAI; PCI:24h, LOC:A4h) 60x Bus Base Address Register for Direct Master-to-PCI I/O Configuration

Bit	Description	Read	Write	Value after Reset
15:0	***Reserved.***	Yes	No	0h
31:16	Assigns a value to bits to use for decoding 60x-to-PCI I/O or Configuration accesses. ***Notes:*** *60x Bus Base Address value must be a multiple of the Range (**not** the Range register).* *Refer to DMPBAM[13] for the I/O Remap Address option.*	Yes	Yes	0h

Register 9-51. (DMPBAM; PCI:28h, LOC:A8h) PCI Base Address (Remap) Register for Direct Master-to-PCI Memory

Bit	Description	Read	Write	Value after Reset
0	**Direct Master Memory Access Enable.** Writing 1 enables decode of Direct Master Memory accesses. Writing 0 disables decode of Direct Master Memory accesses.	Yes	Yes	0
1	**Direct Master I/O Access Enable.** Writing 1 enables decode of Direct Master I/O accesses. Writing 0 disables decode of Direct Master I/O accesses.	Yes	Yes	0
2	**Direct Master Cache Enable.** Writing 1 causes prefetch to occur infinitely.	Yes	Yes	0
12, 3	**Direct Master Read Prefetch Size Control.** Values: 00 = The PCI 9610 continues to prefetch Read data from the PCI Bus until the Direct Master access is finished. This may result in an additional four unneeded Lwords being prefetched from the 32-bit PCI Bus, or four additional unneeded Qwords being prefetched from the 64-bit PCI Bus. 01 = Prefetch up to four Lwords from the 32-bit PCI Bus. Prefetch up to four Qwords from the 64-bit PCI Bus. 10 = Prefetch up to eight Lwords from the 32-bit PCI Bus. Prefetch up to eight Qwords from the 64-bit PCI Bus. 11 = Prefetch up to 16 Lwords from the 32-bit PCI Bus. Prefetch up to 16 Qwords from the 64-bit PCI Bus. Direct Master Burst reads should not exceed programmed limit.	Yes	Yes	00
4	**Direct Master PCI Read Mode.** Writing 0 indicates the PCI 9610 should release the PCI Bus when the Read FIFO becomes full. Writing 1 indicates the PCI 9610 retains the PCI Bus and de-assert IRDY# when the Read FIFO becomes full.	Yes	Yes	0
10, 8:5	**Programmable Almost Full Flag.** When the number of entries in the 32-word Direct Master Write FIFO exceeds a (programmed value +1), MDREQ# is asserted high.	Yes	Yes	00000
9	**Memory Write and Invalidate Mode.** When set to 1, the PCI 9610 waits for 8 or 16 Lwords to be written from the 60x Bus before starting a PCI access. In addition, all Memory Write and Invalidate cycles to the PCI Bus must be 8 or 16 Lword bursts.	Yes	Yes	0
11	**Direct Master Prefetch Limit.** Writing 1 causes the PCI 9610 to not prefetch past 4-KB boundaries.	Yes	Yes	0
13	**I/O Remap Select.** Writing 1 forces PCI Address bits [31:16] to all zeros. Writing 0 uses bits [31:16] of this register as PCI Address bits [31:16].	Yes	Yes	0
15:14	**Direct Master Write Delay.** Delays PCI Bus request after a Direct Master Burst Write cycle has started. Values: 00 = No delay; start cycle immediately 01 = Delay 4 PCI clocks 10 = Delay 8 PCI clocks 11 = Delay 16 PCI clocks	Yes	Yes	00
31:16	**Remap 60x-to-PCI Space into PCI Address Space.** Bits in this register remap (replace) 60x Bus Address bits used in decode as the PCI Address bits. ***Note:*** *Remap Address value must be a multiple of the Range (**not** the Range register).*	Yes	Yes	0h

Register 9-52. (DMCFGA; PCI:2Ch, LOC:ACh) PCI Configuration Address Register for Direct Master-to-PCI I/O Configuration

Bit	Description	Read	Write	Value after Reset
1:0	**Configuration Type.** Values: 00 = Type 0 01 = Type 1	Yes	Yes	00
7:2	**Register Number.**	Yes	Yes	0h
10:8	**Function Number.**	Yes	Yes	0h
15:11	**Device Number.**	Yes	Yes	0h
23:16	**Bus Number.**	Yes	Yes	0h
30:24	***Reserved.***	Yes	No	0h
31	**Configuration Enable.** Writing 1 allows 60x-to-PCI I/O accesses to be converted to a PCI Configuration cycle. Parameters in this table are used to assert the PCI Configuration address.	Yes	Yes	0

Register 9-53. (LAS1RR; PCI:F0h, LOC:170h) 60x Bus Address Space 1 Range Register for PCI-to-60x Bus

Bit	Description	Read	Write	Value after Reset
0	**Memory Space Indicator.** Writing 0 indicates 60x Bus Address Space 1 maps into PCI Memory space. Writing 1 indicates Address Space 1 maps into PCI I/O space.	Yes	Yes	0
2:1	When mapped into Memory space, encoding is as follows: 2/1 Meaning 0 0 Locate anywhere in 32-bit PCI Address space 0 1 Locate below 1 MB in PCI Address space 1 0 Locate anywhere in 64-bit PCI Address space 1 1 ***Reserved*** When mapped into I/O space, bit 1 must be set to 0. Bit 2 is included with bits [31:3] to indicate the decoding range.	Yes	Yes	00
3	When mapped into Memory space, writing 1 indicates reads are prefetchable (does not affect PCI 9610 operation, but is used for system status). When mapped into I/O space, included with bits [31:2] to indicate the decoding range.	Yes	Yes	0
31:4	Specifies which PCI Address bits to use for decoding a PCI access to 60x Bus Space 1. Each bit corresponds to a PCI Address bit. Bit 31 corresponds to address bit 31. Write 1 to all bits that must be included in decode and 0 to all others. (Used in conjunction with PCIBAR3.) Default is 1 MB. When QSR[0]=1, defines PCI Base Address 0. ***Notes:*** *Range (**not** Range register) must be power of 2. "Range register value" is inverse of range.* *All I/O spaces should be limited to 256 bytes per PCI Specification r2.1.*	Yes	Yes	FFF0000h

Register 9-54. (LAS1BA; PCI:F4h, LOC:174h) 60x Bus Address Space 1 60x Bus Base Address (Remap)

Bit	Description	Read	Write	Value after Reset
0	**Space 1 Enable.** Writing 1 enables decoding of PCI addresses for Direct Slave accesses to 60x Bus Space 1. Writing 0 disables decoding.	Yes	Yes	0
1	***Reserved.***	Yes	No	0
3:2	Not used when 60x Bus Space 1 is mapped into Memory space. Included with bits [31:4] for remapping when mapped into I/O space.	Yes	Yes	00
31:4	**Remap PCI Address to 60x Bus Address Space 1 into 60x Bus Address Space.** Bits in this register remap (replace) the PCI Address bits used in decode as 60x Bus Address bits. ***Note:*** *Remap Address value must be a multiple of the Range (**not** the Range register).*	Yes	Yes	0h

Register 9-55. (LBRD1; PCI:F8h, LOC:178h) 60x Bus Address Space 1 Bus Region Descriptor

Bit	Description	Read	Write	Value after Reset
1:0	**Memory Space 1 60x Bus Width.** Writing 00 indicates an 8-bit bus width. Writing 01 indicates a 16-bit bus width. Writing 10 indicates a 64-bus width. Writing 11 indicates a 32-bit bus width.	Yes	Yes	10
5:2	***Special Reserved.***	Yes	Yes	0h
6	***Special Reserved.***	Yes	Yes	1
7	***Special Reserved.***	Yes	Yes	0
8	**Memory Space 1 60x Bus Burst Enable.** Writing 1 enables a PLX Extended Burst. Writing 0 disables a PLX Extended Burst.	Yes	Yes	0
9	**Memory Space 1 Prefetch Disable.** When mapped into Memory space, writing 0 enables Read prefetching. Writing 1 disables prefetching. When prefetching is disabled, the PCI 9610 disconnects after each Memory read.	Yes	Yes	0
10	**Read Prefetch Count Enable.** When set to 1 and Memory prefetching is enabled, the PCI 9610 prefetches up to the number of Qwords specified in the prefetch counter. When set to 0, the PCI 9610 ignores the count and continues prefetching until it is terminated by the PCI Bus.	Yes	Yes	0
14:11	**Read Prefetch Counter.** Number of Qwords to prefetch during Memory Read cycles (0-15).	Yes	Yes	0h
15	**60x Bus Direct Slave Space 1 Assert GBL#.** Writing 1 allows the PCI 9610 to assert GBL# when Address Space 1 is accessed. Writing 0 prevents the PCI 9610 from asserting GBL# when Address Space 1 is accessed.	Yes	Yes	0
31:16	***Reserved.***	Yes	No	0h

Register 9-56. (DMDAC; PCI:FCh, LOC:17Ch) Direct Master PCI Dual Address Cycle Upper Address

Bit	Description	Read	Write	Value after Reset
31:0	**Upper 32 Bits of PCI Dual Address Cycle PCI Address during Direct Master Cycles.** When set to 0, the PCI 9610 performs 32-bit Direct Master Address accesses.	Yes	Yes	0h

Register 9-57. (PCIARB; PCI:100h, LOC:1A0h) PCI Arbiter Control

Bit	Description	Read	Write	Value after Reset
0	**PCI Arbiter Enable.** Value of 0 indicates the PCI arbiter is disabled and REQ0# and GNT0# are used by the PCI 9610 to acquire PCI Bus use. Value of 1 indicates the PCI arbiter is enabled.	PCI/ 60x	Yes	0
1	**PCI 9610 High Priority.** Value of 0 indicates the PCI 9610 participates in round-robin arbitration with the other PCI Masters. Value of 1 indicates a two-level, round-robin arbitration scheme is enabled. The other PCI Bus Masters participate in their own round-robin arbitration. The winner of this arbitration then arbitrates for the PCI Bus with the PCI 9610 (when using the internal PCI arbiter).	PCI/ 60x	Yes	0
2	**Early Grant Release.** Value of 0 indicates the PCI 9610 keeps GNT# asserted until another master requests PCI Bus use. Value of 1 indicates the PCI 9610 always de-asserts the current GNT# when FRAME# is asserted (when using the internal PCI arbiter).	PCI/ 60x	Yes	0
3	**PCI Arbiter Parking on PCI 9610.** Value of 1 indicates the PCI arbiter parks the grant on the PCI 9610. Value of 0 indicates the PCI arbiter parks the grant on the current PCI Master (when using the internal PCI arbiter).	PCI/ 60x	Yes	0
31:4	***Reserved.***	Yes	No	0h

Register 9-58. (PABTADR; PCI:104h, LOC:1A4h) PCI Abort Address

Bit	Description	Read	Write	Value after Reset
31:0	**PCI Abort Address.** When a PCI Master/Target abort occurs, the starting address of the current access is returned to this register.	Yes	No	0h

Register 9-59. (PPCTL1; PCI:108h, LOC:1A8h) 60x Bus Control 1

Bit	Description	Read	Write	Value after Reset
0	**60x Bus Direct Master Cache Wrap Disable.** Writing 1 disables standard bursts from non-cache line-aligned addresses. The default value of 0 enables cache wrap.	Yes	Yes	0
1	**60x Bus Direct Slave Space 0 PLX Extended Burst Enable.** Writing 1 allows the PCI 9610 60x Bus Master to initiate the PLX/60x Bus Extended Burst protocol for devices located in Direct Slave Address Space 0. Writing 0 prevents the PCI 9610 from initiating the Extended Burst protocol.	Yes	Yes	0
2	**60x Bus Direct Slave Space 1 PLX Extended Burst Enable.** Writing 1 allows the PCI 9610 60x Bus Master to initiate the PLX/60x Bus Extended Burst protocol for devices located in Direct Slave Address Space 1. Writing 0 prevents the PCI 9610 from initiating the Extended Burst protocol.	Yes	Yes	0
3	**60x Bus DMA Channel 0 Assert GBL#.** Writing 1 causes the PCI 9610 to assert GBL# when accessing slave devices using DMA Channel 0. Writing 0 prevents the PCI 9610 from asserting GBL# when accessing slave devices using DMA Channel 0.	Yes	Yes	0
4	**60x Bus DMA Channel 1 Assert GBL#.** Writing 1 causes the PCI 9610 to assert GBL# when accessing slave devices using DMA Channel 1. Writing 0 prevents the PCI 9610 from asserting GBL# when accessing slave devices using DMA Channel 1.	Yes	Yes	0
5	**60x Bus DMA Channel 2 Assert GBL#.** Writing 1 causes the PCI 9610 to assert GBL# when accessing slave devices using DMA Channel 2. Writing 0 prevents the PCI 9610 from asserting GBL# when accessing slave devices using DMA Channel 2.	Yes	Yes	0
6	**60x Bus DMA Channel 3 Assert GBL#.** Writing 1 causes the PCI 9610 to assert GBL# when accessing slave devices using DMA Channel 3. Writing 0 prevents the PCI 9610 from asserting GBL# when accessing slave devices using DMA Channel 3.	Yes	Yes	0
7	**60x Bus DMA Channel 0 PLX Extended Burst Enable.** Writing 1 allows the PCI 9610 60x Bus Master to initiate the PLX Extended Burst protocol when accessing slave devices using DMA Channel 0. Writing 0 prevents the PCI 9610 from initiating the Extended Burst protocol when accessing slave devices using DMA Channel 0.	Yes	Yes	0

Note: *There are two Burst Enables:*

- ***60x Burst Enable****—Allows multiple beat accesses to the 60x Bus within the 60x Bus protocol. The limit is a four-beat burst. There are four 60x Burst Enable bits—DMAMODEx[8], LBRD0[26, 24], and/or LBRD1[8].*
- ***PLX Extended Burst Enable****—For the PLX Extended Burst mode, an extension of the 60x Bus protocol, Allows more than a four-beat burst. There are eight PLX Extended Burst Enable bits—PPCTL1[7, 2:1], PPCTL2[2:0], and/or PPCTL3[2:1].*

For a given space or DMA channel, when both 60x Burst and PLX Extended Burst are enabled, the PLX Extended Burst takes precedence.

Register 9-60. (PPCTL2; PCI:109h, LOC:1A9h) 60x Bus Control 2

Bit	Description	Read	Write	Value after Reset
0	**60x Bus DMA Channel 1 PLX Extended Burst Enable.** Writing 1 allows the PCI 9610 60x Bus Master to initiate the PLX Extended Burst protocol when accessing slave devices using DMA Channel 1. Writing 0 prevents the PCI 9610 from initiating the Extended Burst protocol when accessing slave devices using DMA Channel 1.	Yes	Yes	0
1	**60x Bus DMA Channel 2 PLX Extended Burst Enable.** Writing 1 allows the PCI 9610 60x Bus Master to initiate the PLX Extended Burst protocol when accessing slave devices using DMA Channel 2. Writing 0 prevents the PCI 9610 from initiating the Extended Burst protocol when accessing slave devices using DMA Channel 2.	Yes	Yes	0
2	**60x Bus DMA Channel 3 PLX Extended Burst Enable.** Writing 1 allows the PCI 9610 60x Bus Master to initiate the PLX Extended Burst protocol when accessing slave devices using DMA Channel 3. Writing 0 prevents the PCI 9610 from initiating the Extended Burst protocol when accessing slave devices using DMA Channel 3.	Yes	Yes	0
3	**60x Bus Direct Slave Space 0 Strict MPC603e Mode.** Writing 1 prevents the PCI 9610 from sending transfer sizes 5, 6, 7, 9, and 10 to devices located in Address Space 0. Writing 0 allows the PCI 9610 to send all PowerQUICC II legal transfer sizes to devices located in Direct Slave Address Space 0.	Yes	Yes	0
4	**60x Bus Direct Slave Space 1 Strict MPC603e Mode.** Writing 1 prevents the PCI 9610 from sending transfer sizes 5, 6, 7, 9, and 10 to devices located in Address Space 1. Writing 0 allows the PCI 9610 to send all PowerQUICC II legal transfer sizes to devices located in Direct Slave Address Space 1.	Yes	Yes	0
5	**60x Bus DMA Channel 0 Strict MPC603e Mode.** Writing 1 prevents the PCI 9610 from sending transfer sizes 5, 6, 7, 9, and 10 when accessing devices using DMA Channel 0. Writing 0 allows the PCI 9610 to send all PowerQUICC II legal transfer sizes when accessing devices using DMA Channel 0.	Yes	Yes	0
6	**60x Bus DMA Channel 1 Strict MPC603e Mode.** Writing 1 prevents the PCI 9610 from sending transfer sizes 5, 6, 7, 9, and 10 when accessing devices using DMA Channel 1. Writing 0 allows the PCI 9610 to send all PowerQUICC II legal transfer sizes when accessing devices using DMA Channel 1.	Yes	Yes	0
7	**60x Bus DMA Channel 2 Strict MPC603e Mode.** Writing 1 prevents the PCI 9610 from sending transfer sizes 5, 6, 7, 9, and 10 when accessing devices using DMA Channel 2. Writing 0 allows the PCI 9610 to send all PowerQUICC II legal transfer sizes when accessing devices using DMA Channel 2.	Yes	Yes	0

Register 9-61. (PPCTL3; PCI:10Ah, LOC:1AAh) 60x Bus Control 3

Bit	Description	Read	Write	Value after Reset
0	**60x Bus DMA Channel 3 Strict MPC603e Mode.** Writing 1 prevents the PCI 9610 from sending transfer sizes 5, 6, 7, 9, and 10 when accessing devices using DMA Channel 3. Writing 0 allows the PCI 9610 to send all PowerQUICC II legal transfer sizes when accessing devices using DMA Channel 3.	Yes	Yes	0
1	***Reserved.***	Yes	No	0
2	**60x Bus Direct Slave Space Expansion ROM PLX Extended Burst Enable.** Writing 1 allows the PCI 9610 60x Bus Master to initiate the PLX Extended Burst protocol for devices located in Direct Slave Address Space Expansion ROM. Writing 0 prevents the PCI 9610 from initiating the Extended Burst protocol to Expansion ROM.	Yes	Yes	0
3	**60x Bus Direct Slave Space Expansion ROM Strict MPC603e Mode.** Writing 1 prevents the PCI 9610 from sending transfer sizes 5, 6, 7, 9, and 10 to devices located in Address Space Expansion ROM. Writing 0 allows the PCI 9610 to send all PowerQUICC II legal transfer sizes to devices located in Direct Slave Address Space Expansion ROM.	Yes	Yes	0
4	**60x Bus Direct Slave Space Expansion ROM Assert GBL#.** Writing 1 allows the PCI 9610 to assert GBL# when Address Space Expansion ROM is accessed. Writing 0 prevents the PCI 9610 from asserting GBL# when Address Space Expansion ROM is accessed.	Yes	Yes	0
5	**60x Bus Direct Master Address Retry Enable.** Writing 1 enables address Retry, when the Direct Master Write FIFO is full, or Direct Master Read FIFO is empty. Writing 0 disables address Retry. Wait states are inserted until the Direct Master Write FIFO is not full or the Direct Master Read FIFO is not empty.	Yes	Yes	0
7:6	***Reserved.***	Yes	No	00

Register 9-62. (BIGEND2; PCI:10Bh, LOC:1ABh) Big/Little Endian Descriptor 2

Bit	Description	Read	Write	Value after Reset
0	**DMA Channel 2 Big Endian Mode (Address Invariance).** Writing 1 specifies use of Big Endian data ordering for DMA Channel 2 accesses to the 60x Bus Address space. Writing 0 specifies Little Endian ordering.	Yes	Yes	0
1	**DMA Channel 3 Big Endian Mode (Address Invariance).** Writing 1 specifies use of Big Endian data ordering for DMA Channel 3 accesses to the 60x Bus Address space. Writing 0 specifies Little Endian ordering.	Yes	Yes	0
7:2	***Reserved.***	Yes	No	0h

9.4 RUNTIME REGISTERS

Register 9-63. (MBOX0; PCI:40h or 78h, LOC:C0h) Mailbox 0

Bit	Description	Read	Write	Value after Reset
31:0	**32-Bit Mailbox Register.** ***Note:*** *Inbound Queue Port replaces Mailbox Register 0 when the I_2O function is enabled (QSR[0]=1). Mailbox Register 0 is always accessible at PCI address 78h and 60x Bus address C0h.*	Yes	Yes	0h

Register 9-64. (MBOX1; PCI:44h or 7Ch, LOC:C4h) Mailbox 1

Bit	Description	Read	Write	Value after Reset
31:0	**32-Bit Mailbox Register.** ***Note:*** *Mailbox Register 1 is replaced by Outbound Queue Port when the I_2O function is enabled (QSR[0]=1). The Mailbox 1 register is always accessible at PCI address 7Ch and 60x Bus address C4h.*	Yes	Yes	0h

Register 9-65. (MBOX2; PCI:48h, LOC:C8h) Mailbox 2

Bit	Description	Read	Write	Value after Reset
31:0	**32-Bit Mailbox Register.**	Yes	Yes	0h

Register 9-66. (MBOX3; PCI:4Ch, LOC:CCh) Mailbox 3

Bit	Description	Read	Write	Value after Reset
31:0	**32-Bit Mailbox Register.**	Yes	Yes	0h

Register 9-67. (MBOX4; PCI:50h, LOC:D0h) Mailbox 4

Bit	Description	Read	Write	Value after Reset
31:0	**32-Bit Mailbox Register.**	Yes	Yes	0h

Register 9-68. (MBOX5; PCI:54h, LOC:D4h) Mailbox 5

Bit	Description	Read	Write	Value after Reset
31:0	**32-Bit Mailbox Register.**	Yes	Yes	0h

Register 9-69. (MBOX6; PCI:58h, LOC:D8h) Mailbox 6

Bit	Description	Read	Write	Value after Reset
31:0	**32-Bit Mailbox Register.**	Yes	Yes	0h

Register 9-70. (MBOX7; PCI:5Ch, LOC:DCh) Mailbox 7

Bit	Description	Read	Write	Value after Reset
31:0	**32-Bit Mailbox Register.**	Yes	Yes	0h

Register 9-71. (P2LDBELL; PCI:60h, LOC:E0h) PCI-to-60x Doorbell

Bit	Description	Read	Write	Value after Reset
31:0	**Doorbell Register.** The PCI Bus Master can write to this register and assert a LINTo# interrupt for the 60x Bus Master. That processor can then read this register to determine which doorbell bit was set. The PCI Bus Master sets the doorbell by writing 1 to a particular bit. The 60x Bus Master can clear a doorbell bit by writing 1 to that bit position.	Yes	Yes/Clr	0h

Register 9-72. (L2PDBELL; PCI:64h, LOC:E4h) 60x-to-PCI Doorbell

Bit	Description	Read	Write	Value after Reset
31:0	**Doorbell Register.** The 60x Bus Master can write to this register and assert a PCI interrupt. The PCI Bus Master can then read this register to determine which doorbell bit was set. The 60x Bus Master sets the doorbell by writing 1 to a particular bit. The PCI Bus Master can clear a doorbell bit by writing 1 to that bit position.	Yes	Yes/Clr	0h

Register 9-73. (INTCSR; PCI:68h, LOC:E8h) Interrupt Control/Status

Bit	Description	Read	Write	Value after Reset
0	**Enable 60x Bus TEA# Output.** Writing 1 enables the PCI 9610 to assert a TEA# interrupt when the PCI 9610, as a 60x Bus slave, cannot complete a data transfer. Writing 0 disables TEA# as an output from the PCI 9610.	Yes	Yes	0
1	**Enable LINTo# Output for Interrupt Status Bits.** These bits are PCISR[13:12] and CNTRL[21].	Yes	Yes	0
2	**Generate PCI Bus SERR# Interrupt.** When set to 0, writing 1 asserts the PCI Bus SERR# interrupt.	Yes	Yes	0
3	**Mailbox Interrupt Enable.** Writing 1 enables a LINTo# Interrupt to be asserted when the PCI Bus writes to MBOX0 through MBOX3. To clear the Interrupt, the 60x Bus Master must read the Mailbox. Used in conjunction with the 60x Bus Interrupt Output Enable (bit 16).	Yes	Yes	0
4	**Power Management Interrupt Enable.** Writing 1 enables a LINTo# Interrupt to be asserted when the Power Management power state changes.	Yes	Yes	0
5	**Power Management Interrupt.** When set to 1, indicates a Power Management interrupt is pending. A Power Management interrupt is caused by a change in the Power State register (PMCSR). Writing 1 clears the interrupt.	Yes	Yes/Clr	0
6	***Special Reserved.***	Yes	Yes	0
7	**Direct Master Write/Direct Slave Read 60x Bus Data Parity Check Error Status.** When set to 1, indicates the PCI 9610 has detected a 60x Bus Data Parity check error, although the Check Parity Error bit is disabled. Writing 1 clears this bit to 0.	Yes	Yes/Clr	0
8	**PCI Interrupt Enable.** Writing 1 enables PCI interrupts. Writing 0 disables PCI interrupts.	Yes	Yes	1
9	**PCI Doorbell Interrupt Enable.** Writing 1 enables Doorbell interrupts. Used in conjunction with the PCI Interrupt Enable (bit 8). Clearing the Doorbell Interrupt bits that caused the interrupt also clears the interrupt.	Yes	Yes	0
10	**PCI Abort Interrupt Enable.** Value of 1 enables Master abort or Master detect of a Target abort to assert a PCI interrupt. Used in conjunction with the PCI Interrupt Enable (bit 8). Clearing the Abort Status bits also clears the PCI interrupt.	Yes	Yes	0
11	**60x Interrupt Input Enable.** Writing 1 enables a LINTi# interrupt input to assert a PCI interrupt. Used in conjunction with the PCI Interrupt Enable (bit 8). Clearing the 60x Bus cause of the interrupt also clears the interrupt.	Yes	Yes	0
12	**Retry Abort Enable.** Writing 1 enables the PCI 9610 to treat 256 consecutive Master Retrys to a Target as a Target abort. Writing 0 enables the PCI 9610 to attempt Master Retrys indefinitely.	Yes	Yes	0
13	**PCI Doorbell Interrupt Active.** When set to 1, indicates the PCI Doorbell interrupt is active.	Yes	No	0
14	**PCI Abort Interrupt Active.** When set to 1, indicates the PCI Abort interrupt is active.	Yes	No	0
15	**60x Bus Input Interrupt Active.** When set to 1, indicates the LINTi# Input interrupt is active.	Yes	No	0
16	**60x Bus Interrupt Output Enable.** Writing 1 enables LINTo# interrupt output. Used in conjunction with the Mailbox Interrupt Enable (bit 3).	Yes	Yes	1
17	**60x Bus Doorbell Interrupt Enable.** Writing 1 enables Doorbell interrupts. Used in conjunction with the Interrupt Enable bit. Clearing the 60x Bus Doorbell Interrupt bits that caused the interrupt also clears the interrupt.	Yes	Yes	0
18	**60x Bus DMA Channel 0 Interrupt Enable.** Writing 1 enables DMA Channel 0 interrupts. Used in conjunction with the Interrupt Enable bit. Clearing the DMA Status bits also clears the interrupt.	Yes	Yes	0

Register 9-73. (INTCSR; PCI:68h, LOC:E8h) Interrupt Control/Status (Continued)

Bit	Description	Read	Write	Value after Reset
19	**60x Bus DMA Channel 1 Interrupt Enable.** Writing 1 enables DMA Channel 1 interrupts. Used in conjunction with the Interrupt Enable bit. Clearing the DMA Status bits also clears the interrupt.	Yes	Yes	0
20	**60x Bus Doorbell Interrupt Active.** Reading 1 indicates the 60x Bus Doorbell interrupt is active.	Yes	No	0
21	**DMA Channel 0 Interrupt Active.** Reading 1 indicates the DMA Channel 0 interrupt is active.	Yes	No	0
22	**DMA Channel 1 Interrupt Active.** Reading 1 indicates the DMA Channel 1 interrupt is active.	Yes	No	0
23	**BIST Interrupt Active.** Reading 1 indicates the built-in self test (BIST) interrupt is active. The BIST interrupt is asserted by writing 1 to the PCI BIST Interrupt Enable bit (PCIBISTR[6]=1). Clearing PCIBISTR[6] clears the interrupt. Refer to the PCIBISTR register for a description of the self test.	Yes	No	0
24	Reading 0 indicates the Direct Master was the Bus Master during a Master or Target abort.	Yes	No	1
25	Reading 0 indicates DMA Channel 0 was the Bus Master during a Master or Target abort.	Yes	No	1
26	Reading 0 indicates DMA Channel 1 was the Bus Master during a Master or Target abort.	Yes	No	1
27	Reading 0 indicates a Target abort was asserted by the PCI 9610 after 256 consecutive Master Retrys to a Target.	Yes	No	1
28	Reading 1 indicates the PCI Bus wrote data to MBOX0. Enabled only when the Mailbox Interrupt Enable bit is set (bit 3 =1).	Yes	No	0
29	Reading 1 indicates the PCI Bus wrote data to MBOX1. Enabled only when the Mailbox Interrupt Enable bit is set (bit 3 =1).	Yes	No	0
30	Reading 1 indicates the PCI Bus wrote data to MBOX2. Enabled only when the Mailbox Interrupt Enable bit is set (bit 3 =1).	Yes	No	0
31	Reading 1 indicates the PCI Bus wrote data to MBOX3. Enabled only when the Mailbox Interrupt Enable bit is set (bit 3 =1).	Yes	No	0

Register 9-74. (CNTRL; PCI:6Ch, LOC:ECh) Serial EEPROM Control, PCI Command Codes, User I/O Control, and Init Control

Bit	Description	Read	Write	Value after Reset
3:0	**PCI Read Command Code for DMA.**	Yes	Yes	1110
7:4	**PCI Write Command Code for DMA.**	Yes	Yes	0111
11:8	**PCI Memory Read Command Code for Direct Master.**	Yes	Yes	0110
15:12	**PCI Memory Write Command Code for Direct Master.**	Yes	Yes	0111
16	**General Purpose Output.** Writing 1 causes USERo output to go high. Writing 0 causes USERo output to go low.	Yes	Yes	1
17	**General Purpose Input.** Reading 1 indicates the USERi input pin is high. Reading 0 indicates the USERi pin is low.	Yes	No	—
18	**USERi/DACK0# Select.** Writing 1 selects USERi to be an input to the chip. Writing 0 selects DACK0# as an output. Enables the user to select between the USERi and DACK0# functions when USERi is chosen to be an input. DMAMODE0[12] is the select bit for the pin to function as DACK0#.	Yes	Yes	1
19	**USERo/DREQ0# Select.** Writing 1 selects USERo to be an output from the chip. Writing 0 selects DREQ0# as an input. Enables the user to select between the USERo and DREQ0# functions when USERo is chosen to be an output. DMAMODE0[12] is the select bit for the pin to function as DREQ0#.	Yes	Yes	1
20	**LINTo# Interrupt Status.** When HOSTEN# is enabled, reading 1 indicates the LINTo# interrupt is active by way of the INTA# PCI interrupt. Writing 1 clears this bit.	Yes	Yes/Clr	0
21	**SERR# Interrupt.** When the PCI 9610 is in Peripheral mode (HOSTEN# input is high), this bit has no meaning. When the Host mode (HOSTEN#) is low, SERR# Interrupt is an input status bit that indicates SERR# was asserted. Writing 1 clears this bit.	Yes	Yes/Clr	0
23:22	***Reserved.***	Yes	No	00
24	**Serial EEPROM Clock for 60x or PCI Bus Reads or Writes to Serial EEPROM.** Toggling this bit asserts the serial EEPROM clock. (Refer to manufacturer's data sheet for particular serial EEPROM being used.)	Yes	Yes	0
25	**Serial EEPROM Chip Select.** For 60x or PCI Bus reads or writes to the serial EEPROM, setting this bit to 1 provides the serial EEPROM chip select.	Yes	Yes	0
26	**Write Bit to Serial EEPROM.** For writes, this output bit is input to the serial EEPROM. Clocked into the serial EEPROM by the serial EEPROM clock.	Yes	Yes	0
27	**Read Bit from Serial EEPROM.** (Refer to Section 2.3.2 and 2.3.2.1.)	Yes	No	—
28	**Programmed Serial EEPROM Present.** When set to 1, indicates that a blank or programmed serial EEPROM is present.	Yes	No	0
29	**Reload Configuration Registers.** When set to 0, writing 1 causes the PCI 9610 to reload the 60x Configuration registers from the serial EEPROM.	Yes	Yes	0
30	**PCI Adapter Software Reset.** Writing 1 holds the PCI 9610 60x Bus logic in a reset state, and asserts LRESET# output. Contents of the PCI Configuration registers and shared Runtime registers are not reset. A software reset can only be cleared from the PCI Bus.	Yes	Yes	0
31	**EEDI/EEDO Three-State Enable.** When set to 1, the EEDI/EEDO signal is placed in a high-impedance state, and could be used as a Data Read input signal from the serial EEPROM.	Yes	No	0

Register 9-75. (PCIHIDR; PCI:70h, LOC:F0h) PCI Hardcoded Configuration ID

Bit	Description	Read	Write	Value after Reset
15:0	**Vendor ID.** Identifies manufacturer of device. Hardcoded to the PCI SIG-issued Vendor ID of PLX (10B5h).	Yes	No	10B5h
31:16	**Device ID.** Identifies particular device. Hardcoded to the PLX part number for PCI interface chip PCI 9610.	Yes	No	9610h

Register 9-76. (PCIHREV; PCI:74h, LOC:F4h) PCI Hardcoded Revision ID

Bit	Description	Read	Write	Value after Reset
7:0	**Revision ID.** Hardcoded silicon revision of the PCI 9610.	Yes	No	Current Rev # (AA)

Register 9-77. (INTCSR2; PCI:110h, LOC:1B0h) Interrupt Control/Status 2

Bit	Description	Read	Write	Value after Reset
0	**TEA# Assertion Detected.** Writing 1 clears this bit. When the PCI 9610 was a master on the Mode Bus, the Data tenure was terminated by the assertion of TEA# input to the PCI 9610.	Yes	Yes/Clr	0
1	**Enable TEA# Assertion Detected as LINTo# Output.** Writing 1 enables this bit. Writing 0 disables this bit.	Yes	Yes	0
2	***Reserved***.	Yes	No	0
3	**60x Bus DMA Channel 2 Interrupt Enable.** Writing 1 enables DMA Channel 2 interrupts. Used in conjunction with the 60x Bus Interrupt Enable bit. Clearing the DMA Status bits also clears the interrupt.	Yes	Yes	0
4	**60x Bus DMA Channel 3 Interrupt Enable.** Writing 1 enables DMA Channel 3 interrupts. Used in conjunction with the 60x Bus Interrupt Enable bit. Clearing the DMA Status bits also clears the interrupt.	Yes	Yes	0
5	**DMA Channel 2 Interrupt Active.** Reading 1 indicates the DMA Channel 2 interrupt is active.	Yes	No	0
6	**DMA Channel 3 Interrupt Active.** Reading 1 indicates the DMA Channel 3 interrupt is active.	Yes	No	0
7	Reading 0 indicates DMA Channel 2 was the Bus Master during a Master or Target abort.	Yes	No	1
8	Reading 0 indicates DMA Channel 3 was the Bus Master during a Master or Target abort.	Yes	No	1
9	Because the 60x Bus Master attempted an illegal access (read or write) to a configuration space, TEA# was asserted. Writing 1 clears this bit. There is no interrupt associated with this status bit.	Yes	Yes/Clr	0
10	Because the 60x Bus Master attempted a Multibeat burst to PCI I/O space, TEA# was asserted. Writing 1 clears this bit to 0.	Yes	Yes/Clr	0
11	**Enables LINTo# for Status Bit INTCSR2[10].** Writing 1 enables LINTo# when the 60x Bus Master attempts a Multibeat burst to the PCI I/O space.	Yes	Yes	0
31:12	***Reserved***.	Yes	No	0h

9.5 DMA REGISTERS

Register 9-78. (DMAMODE0; PCI:80h, LOC:100h) DMA Channel 0 Mode

Bit	Description	Read	Write	Value after Reset
1:0	**60x Bus Width.** Writing 00 indicates an 8-bit bus width. Writing 01 indicates a 16-bit bus width. Writing 10 indicates a 64-bit bus width. Writing 11 indicates a 32-bit bus width.	Yes	Yes	10
5:2	***Special Reserved.***	Yes	Yes	0h
6	***Special Reserved.***	Yes	Yes	1
7	***Special Reserved.***	Yes	Yes	0
8	**PLX Extended Burst Enable.** Writing 1 enables a PLX Extended Burst. Writing 0 disables a PLX Extended Burst.	Yes	Yes	0
9	**Scatter/Gather Mode.** Writing 1 indicates Scatter/Gather mode is enabled. For Scatter/Gather mode, DMA source address, destination address, and byte count are loaded from memory in PCI or 60x Bus Address spaces. Writing 0 indicates Block mode is enabled.	Yes	Yes	0
10	**Done Interrupt Enable.** Writing 1 enables an interrupt when done. Writing 0 disables an interrupt when done. When DMA Clear Count mode is enabled (bit[16] = 1), the interrupt does not occur until the byte count is cleared.	Yes	Yes	0
11	**60x Bus Addressing Mode.** Writing 1 holds the 60x Bus Address constant. Writing 0 indicates the 60x Bus Address is incremented.	Yes	Yes	0
12	**Demand Mode.** Writing 1 causes the DMA controller to operate in Demand mode. In Demand mode, the DMA controller transfers data when its DREQ0# input is asserted. DACK0# is asserted to indicate the current 60x Bus transfer is in response to DREQ0# input.	Yes	Yes	0
13	**Memory Write and Invalidate Mode for DMA Transfers.** When set to 1, the PCI 9610 performs Memory Write and Invalidate cycles to the PCI Bus. The PCI 9610 supports Memory Write and Invalidate sizes of 8 or 16 Lwords. Size is specified in the System Cache Line Size bits (PCICLSR[7:0]). When a size other than 8 or 16 is specified, the PCI 9610 performs Write transfers rather than Memory Write and Invalidate transfers. Transfers must start and end at cache line boundaries.	Yes	Yes	0
14	**DMA EOT0# Enable.** Writing 1 enables the EOT0# input pin. Writing 0 disables the EOT0# input pin.	Yes	Yes	0
15	***Special Reserved.***	Yes	Yes	0
16	**DMA Clear Count Mode.** Writing 1 clears the byte count in each Scatter/Gather descriptor when the corresponding DMA transfer is complete. Applies only in Non-Valid mode.	Yes	Yes	0
17	**DMA Channel 0 Interrupt Select.** Writing 1 routes the DMA Channel 0 interrupt to the PCI Bus interrupt. Writing 0 routes the DMA Channel 0 interrupt to the 60x Bus LINTo# interrupt.	Yes	Yes	0
18	**DAC Chain Load.** When set to 1, enables the descriptor to load the PCI Dual Address Cycle value. Otherwise, it uses the register contents.	Yes	Yes	0
19	**EOT0# End Link.** Used only for Scatter/Gather DMA transfers. When EOT0# is asserted, a value of 1 indicates the DMA transfer completes the current Scatter/Gather link but continues with the remaining Scatter/Gather transfers. When EOT0# is asserted, a value of 0 indicates the DMA transfer completes the current Scatter/Gather transfer, but does not continue with the remaining Scatter/Gather transfers.	Yes	Yes	0
20	**Valid Mode Enable.** Value of 0 indicates the Valid bit (DMASIZ0[31]) is ignored. Value of 1 indicates the DMA descriptors are processed only when the Valid bit is set (DMASIZ0[31=1]). When the Valid bit is set, the transfer count is 0, and the descriptor is not the last descriptor in the chain. The DMA controller then moves to the next descriptor in the chain. Value of 0 also indicates the Non-Valid mode descriptor-loading sequence, whereas a value of 1 indicates the Valid Mode Ring Management descriptor-loading sequence.	Yes	Yes	0

Register 9-78. (DMAMODE0; PCI:80h, LOC:100h) DMA Channel 0 Mode (Continued)

Bit	Description	Read	Write	Value after Reset
21	**Valid Stop Control.** Value of 0 indicates the DMA Scatter/Gather controller continuously polls a descriptor with the Valid bit set to 0 (invalid descriptor) when the Valid Mode Enable bit is set (bit [20]=1). Value of 1 indicates the Scatter/Gather controller stops polling when the Valid bit with a value of 0 is detected (DMASIZ0[31]=0). In this case, the CPU must restart the DMA controller by setting the Start bit (DMACSR0[1]=1). A pause sets the DMA Done register.	Yes	Yes	0
31:22	***Reserved.***	Yes	No	0h

Register 9-79. (DMAPADR0; (PCI:84h, LOC:104h when DMAMODE0[20]=0 or PCI:88h, LOC:108h when DMAMODE0[20]=1) DMA Channel 0 PCI Address

Bit	Description	Read	Write	Value after Reset
31:0	**PCI Address Register.** Indicates from where in PCI Memory space DMA transfers (reads or writes) start.	Yes	Yes	0h

Register 9-80. (DMALADR0; PCI:88h, LOC:108h when DMAMODE0[20]=0 or PCI:8Ch, LOC:10Ch when DMAMODE0[20]=1) DMA Channel 0 60x Address

Bit	Description	Read	Write	Value after Reset
31:0	**60x Address Register.** Indicates from where in 60x Bus Memory space DMA transfers (reads or writes) start.	Yes	Yes	0h

Register 9-81. (DMASIZ0; PCI:8Ch, LOC:10Ch when DMAMODE0[20]=0 or PCI:84h, LOC:104h when DMAMODE0[20]=1) DMA Channel 0 Transfer Size (Bytes)

Bit	Description	Read	Write	Value after Reset
22:0	**DMA Transfer Size (Bytes).** Indicates the number of bytes to transfer during a DMA operation.	Yes	Yes	0h
30:23	***Reserved.***	Yes	No	0h
31	**Valid.** When the Valid Mode Enable bit is set (DMAMODE0[20]=1), indicates the validity of this DMA descriptor.	Yes	Yes	0

Register 9-82. (DMADPR0; PCI:90h, LOC:110h) DMA Channel 0 Descriptor Pointer

Bit	Description	Read	Write	Value after Reset
0	**Descriptor Location.** Writing 1 indicates PCI Address space. Writing 0 indicates 60x Bus Address space.	Yes	Yes	0
1	**End of Chain.** Writing 1 indicates end of chain. Writing 0 indicates not end of chain descriptor. (Same as Block mode.)	Yes	Yes	0
2	**Interrupt after Terminal Count.** Writing 1 causes an interrupt to be asserted after the terminal count for this descriptor is reached. Writing 0 disables interrupts from being asserted.	Yes	Yes	0
3	**Transfer Direction.** Writing 1 indicates transfers from the 60x Bus to the PCI Bus. Writing 0 indicates transfers from the PCI Bus to the 60x Bus.	Yes	Yes	0
31:4	**Next Descriptor Address.** Qword-aligned (bits [3:0]=0000).	Yes	Yes	0h

Register 9-83. (DMAMODE1; PCI:94h, LOC:114h) DMA Channel 1 Mode

Bit	Description	Read	Write	Value after Reset
1:0	**60x Bus Width.** Writing 00 indicates an 8-bit bus width. Writing 01 indicates a 16-bit bus width. Writing 10 indicates a 64-bit bus width. Writing 11 indicates a 32-bit bus width.	Yes	Yes	10
5:2	***Special Reserved.***	Yes	Yes	0h
6	***Special Reserved.***	Yes	Yes	1
7	***Special Reserved.***	Yes	Yes	0
8	**PLX Extended Burst Enable.** Writing 1 enables a PLX Extended Burst. Writing 0 disables a PLX Extended Burst.	Yes	Yes	0
9	**Scatter/Gather Mode.** Writing 1 indicates Scatter/Gather mode is enabled. For Scatter/Gather mode, the DMA source address, destination address, and byte count are loaded from memory in PCI or 60x Bus Address spaces. Writing 0 indicates Block mode is enabled.	Yes	Yes	0
10	**Done Interrupt Enable.** Writing 1 enables an interrupt when done. Writing 0 disables the interrupt when done. When DMA Clear Count mode is enabled, the interrupt does not occur until the byte count is cleared.	Yes	Yes	0
11	**60x Bus Addressing Mode.** Writing 1 holds the 60x Bus Address constant. Writing 0 indicates the 60x Bus Address is incremented.	Yes	Yes	0
12	**Demand Mode.** Writing 1 causes the DMA controller to operate in Demand mode. In Demand mode, the DMA controller transfers data when its DREQ1# input is asserted. Asserts DACK1# to indicate the current 60x Bus transfer is in response to DREQ1# input.	Yes	Yes	0
13	**Memory Write and Invalidate Mode for DMA Transfers.** When set to 1, the PCI 9610 performs Memory Write and Invalidate cycles to the PCI Bus. The PCI 9610 supports Memory Write and Invalidate sizes of 8 or 16 Lwords. Size is specified in the System Cache Line Size bits (PCICLSR[7:0]). When a size other than 8 or 16 is specified, the PCI 9610 performs Write transfers rather than Memory Write and Invalidate transfers. Transfers must start and end at cache line boundaries.	Yes	Yes	0
14	**DMA EOT1# Enable.** Writing 1 enables the EOT1# input pin. Writing 0 disables the EOT1# input pin.	Yes	Yes	0
15	***Special Reserved.***	Yes	Yes	0
16	**DMA Clear Count Mode.** Writing 1 clears the byte count in each Scatter/Gather descriptor when the corresponding DMA transfer is complete. Applies only in Non-Valid mode.	Yes	Yes	0

Register 9-83. (DMAMODE1; PCI:94h, LOC:114h) DMA Channel 1 Mode (Continued)

Bit	Description	Read	Write	Value after Reset
17	**DMA Channel 1 Interrupt Select.** Writing 1 routes the DMA Channel 1 interrupt to the PCI Bus interrupt. Writing 0 routes the DMA Channel 1 interrupt to the 60x Bus LINTo# interrupt.	Yes	Yes	0
18	**DAC Chain Load.** When set to 1, enables the descriptor to load the PCI Dual Address Cycle value. Otherwise, it uses the register contents.	Yes	Yes	0
19	**EOT1# End Link.** Used only for DMA Scatter/Gather transfers. When EOT1# is asserted, value of 1 indicates the DMA transfer completes the current Scatter/Gather link but continues with the remaining Scatter/Gather transfers. When EOT1# is asserted, a value of 0 indicates the DMA transfer completes the current Scatter/Gather transfer, but does not continue with the remaining Scatter/Gather transfers.	Yes	Yes	0
20	**Valid Mode Enable.** Value of 0 indicates the Valid bit (DMASIZ1[31]) is ignored. Value of 1 indicates the DMA descriptors are processed only when the Valid bit is set (DMASIZ1[31=1]). When the Valid bit is set, the transfer count is 0, and the descriptor is not the last descriptor in the chain. The DMA controller then moves to the next descriptor in the chain. Value of 0 also indicates the Non-Valid mode descriptor-loading sequence, whereas a value of 1 indicates Valid Mode Ring Management descriptor-loading sequence.	Yes	Yes	0
21	**Valid Stop Control.** Value of 0 indicates the DMA Scatter/Gather controller continuously polls a descriptor with the Valid bit set to 0 (invalid descriptor) When the Valid Mode Enable bit is set (bit [20]=1). Value of 1 indicates the Scatter/Gather controller stops polling when the Valid bit with a value of 0 is detected (DMASIZ1[31]=0). In this case, the CPU must restart the DMA controller by setting the Start bit (DMACSR1[1]=1). A pause sets the DMA Done register.	Yes	Yes	0
31:22	***Reserved.***	Yes	No	0h

Register 9-84. (DMAPADR1;PCI:98h, LOC:118h when DMAMODE1[20]=0 or PCI:9Ch, LOC:11Ch when DMAMODE1[20]=1) DMA Channel 1 PCI Address

Bit	Description	Read	Write	Value after Reset
31:0	**PCI Address Register.** Indicates from where in PCI Memory space DMA transfers (reads or writes) start.	Yes	Yes	0h

Register 9-85. (DMALADR1;PCI:9Ch, LOC:11Ch when DMAMODE1[20]=0 or PCI:A0h, LOC:120h when DMAMODE1[20]=1) DMA Channel 1 60x Address

Bit	Description	Read	Write	Value after Reset
31:0	**60x Address Register.** Indicates from where in 60x Bus Memory space DMA transfers (reads or writes) start.	Yes	Yes	0h

Register 9-86. (DMASIZ1; PCI:A0h, LOC:120h when DMAMODE1[20]=0 or PCI:98h, LOC:118h when DMAMODE1[20]=1) DMA Channel 1 Transfer Size (Bytes)

Bit	Description	Read	Write	Value after Reset
22:0	**DMA Transfer Size (Bytes).** Indicates the number of bytes to transfer during a DMA operation.	Yes	Yes	0h
30:23	***Reserved.***	Yes	No	0h
31	**Valid.** When the Valid Mode Enable bit is set (DMAMODE1[20]=1), indicates the validity of this DMA descriptor.	Yes	Yes	0

Register 9-87. (DMADPR1; PCI:A4h, LOC:124h) DMA Channel 1 Descriptor Pointer

Bit	Description	Read	Write	Value after Reset
0	**Descriptor Location.** Writing 1 indicates PCI Address space. Writing 0 indicates 60x Address space.	Yes	Yes	0
1	**End of Chain.** Writing 1 indicates end of chain. Writing 0 indicates not end of chain descriptor. (Same as Block mode.)	Yes	Yes	0
2	**Interrupt after Terminal Count.** Writing 1 causes an interrupt to be asserted after the terminal count for this descriptor is reached. Writing 0 disables interrupts from being asserted.	Yes	Yes	0
3	**Transfer Direction.** Writing 1 indicates transfers from the 60x Bus to the PCI Bus. Writing 0 indicates transfers from the PCI Bus to the 60x Bus.	Yes	Yes	0
31:4	**Next Descriptor Address.** Qword-aligned (bits [3:0]=0000).	Yes	Yes	0h

Register 9-88. (DMACSR0; PCI:A8h, LOC:128h) DMA Channel 0 Command/Status

Bit	Description	Read	Write	Value after Reset
0	**Channel 0 Enable.** Writing 1 enables Channel 0 to transfer data. Writing 0 disables Channel 0 from starting a DMA transfer, and when in the process of transferring data, suspends the transfer (pause).	Yes	Yes	0
1	**Channel 0 Start.** Writing 1 causes Channel 0 to start transferring data when Channel 0 is enabled.	No	Yes/Set	0
2	**Channel 0 Abort.** Writing 1 causes Channel 0 to abort current transfer. Channel 0 Enable bit must be cleared (bit [0]=0). Sets Channel 0 Done (bit [4]=1) when abort is complete.	No	Yes/Set	0
3	**Channel 0 Clear Interrupt.** Writing 1 clears Channel 0 interrupts.	No	Yes/Clr	0
4	**Channel 0 Done.** Reading 1 indicates a channel transfer is complete. Reading 0 indicates a channel transfer is not complete.	Yes	No	1
7:5	***Reserved.***	Yes	No	000

Register 9-89. (DMACSR1; PCI:A9h, LOC:129h) DMA Channel 1 Command/Status

Bit	Description	Read	Write	Value after Reset
0	**Channel 1 Enable.** Writing 1 enables Channel 1 to transfer data. Writing 0 disables Channel 1 from starting a DMA transfer, and when in the process of transferring data, suspends the transfer (pause).	Yes	Yes	0
1	**Channel 1 Start.** Writing 1 causes Channel 1 to start transferring data when Channel 1 is enabled.	No	Yes/Set	0
2	**Channel 1 Abort.** Writing 1 causes Channel 1 to abort current transfer. Channel 1 Enable bit must be cleared (bit [0]=0). Sets Channel 1 Done (bit [4]=1) when abort is complete.	No	Yes/Set	0
3	**Channel 1 Clear Interrupt.** Writing 1 clears Channel 1 interrupts.	No	Yes/Clr	0
4	**Channel 1 Done.** Reading 1 indicates a channel transfer is complete. Reading 0 indicates a channel transfer is not complete.	Yes	No	1
7:5	***Reserved.***	Yes	No	000

Register 9-90. (DMAARB; PCI:ACh, LOC:12Ch) DMA Arbitration

Same as Register 9-40 “(MARBR; PCI:08h or ACh, LOC:88h or 12Ch) Mode/DMA Arbitration,” on page 9-20.

Register 9-91. (DMATHR01; PCI:B0h, LOC:130h) DMA Channels 0 and 1 Threshold

Bit	Description	Read	Write	Value after Reset
3:0	**DMA Channel 0 PCI-to-60x Almost Full (C0PLAF).** Number of full entries (divided by two, minus one) in the FIFO before requesting the 60x Bus for writes. (C0PLAF+1) + (C0PLAE+1) should be ≤ a FIFO Depth of 32.	Yes	Yes	0h
7:4	**DMA Channel 0 60x-to-PCI Almost Empty (C0LPAE).** Number of empty entries (divided by two, minus one) in the FIFO before requesting the 60x Bus for reads. (C0LPAF+1) + (C0LPAE+1) should be ≤ a FIFO depth of 32.	Yes	Yes	0h
11:8	**DMA Channel 0 60x-to-PCI Almost Full (C0LPAF).** Number of full entries (divided by two, minus one) in the FIFO before requesting the PCI Bus for writes.	Yes	Yes	0h
15:12	**DMA Channel 0 PCI-to-60x Almost Empty (C0PLAE).** Number of empty entries (divided by two, minus one) in the FIFO before requesting the PCI Bus for reads.	Yes	Yes	0h
19:16	**DMA Channel 1 PCI-to-60x Almost Full (C1PLAF).** Number of full entries (divided by two, minus one) in the FIFO before requesting the 60x Bus for writes. (C1PLAF+1) + (C1PLAE+1) should be ≤ a FIFO Depth of 32.	Yes	Yes	0h
23:20	**DMA Channel 1 60x-to-PCI Almost Empty (C1LPAE).** Number of empty entries (divided by two, minus one) in the FIFO before requesting the 60x Bus for reads. (C1LPAF+1) + (C1LPAE+1) should be ≤ a FIFO depth of 32.	Yes	Yes	0h
27:24	**DMA Channel 1 60x-to-PCI Almost Full (C1LPAF).** Number of full entries (divided by two, minus one) in the FIFO before requesting the PCI Bus for writes.	Yes	Yes	0h
31:28	**DMA Channel 1 PCI-to-60x Almost Empty (C1PLAE).** Number of empty entries (divided by two, minus one) in the FIFO before requesting the PCI Bus for reads.	Yes	Yes	0h

Note: For DMA Channel 0 only, *when the number of entries needed is x, then the value is one less than half the number of entries (that is, x/2 - 1).*

Register 9-92. (DMADAC0; PCI:B4h, LOC:134h) DMA Channel 0 PCI Dual Address Cycle Upper Address

Bit	Description	Read	Write	Value after Reset
31:0	**Upper 32 Bits of the PCI Dual Address Cycle PCI Address during DMA Channel 0 Cycles.** When set to 0h, the PCI 9610 performs a 32-bit DMA Channel 0 Address access.	Yes	Yes	0h

Register 9-93. (DMADAC1; PCI:B8h, LOC:138h) DMA Channel 1 PCI Dual Address Cycle Upper Address

Bit	Description	Read	Write	Value after Reset
31:0	**Upper 32 Bits of the PCI Dual Address Cycle PCI Address during DMA Channel 1 Cycles.** When set to 0h, the PCI 9610 performs a 32-bit DMA Channel 1 Address access.	Yes	Yes	0h

Register 9-94. (DMAMODE2; PCI:1C0h, LOC:1C0h) DMA Channel 2 Mode

Bit	Description	Read	Write	Value after Reset
1:0	**60x Bus Width.** Writing 00 indicates an 8-bit bus width. Writing 01 indicates a 16-bit bus width. Writing 10 indicates a 64-bit bus width. Writing 11 indicates a 32-bit bus width.	Yes	Yes	10
5:2	***Special Reserved.***	Yes	Yes	0h
6	***Special Reserved.***	Yes	Yes	1
7	***Special Reserved.***	Yes	Yes	0
8	**PLX Extended Burst Enable.** Writing 1 enables a PLX Extended Burst. Writing 0 disables a PLX Extended Burst.	Yes	Yes	0
9	**Scatter/Gather Mode.** Writing 1 indicates Scatter/Gather mode is enabled. For Scatter/Gather mode, the DMA source address, destination address, and byte count are loaded from memory in PCI or 60x Address spaces. Writing 0 indicates Block mode is enabled.	Yes	Yes	0
10	**Done Interrupt Enable.** Writing 1 enables an interrupt when done. Writing 0 disables the interrupt when done. When DMA Clear Count mode is enabled (bit [16]=1), the interrupt does not occur until the byte count is cleared.	Yes	Yes	0
11	**60x Bus Addressing Mode.** Writing 1 holds the 60x Bus Address constant. Writing 0 indicates the 60x Bus Address is incremented.	Yes	Yes	0
12	**Demand Mode.** Writing 1 causes the DMA controller to operate in Demand mode. In Demand mode, the DMA controller transfers data when its DREQ2# input is asserted. Asserts DACK2# to indicate the current 60x Bus transfer is in response to DREQ2# input.	Yes	Yes	0
13	**Memory Write and Invalidate Mode for DMA Transfers.** When set to 1, the PCI 9610 performs Memory Write and Invalidate cycles to the PCI Bus. The PCI 9610 supports Memory Write and Invalidate sizes of 8 or 16 Lwords. Size is specified in the System Cache Line Size bits (PCICLSR[7:0]). When a size other than 8 or 16 is specified, the PCI 9610 performs Write transfers rather than Memory Write and Invalidate transfers. Transfers must start and end at cache line boundaries.	Yes	Yes	0
14	**DMA EOT2# Enable.** Writing 1 enables the EOT2# input pin. Writing 0 disables the EOT2# input pin.	Yes	Yes	0
15	***Special Reserved.***	Yes	Yes	0
16	**DMA Clear Count Mode.** Writing 1 clears the byte count in each Scatter/Gather descriptor when the corresponding DMA transfer is complete. Applies only in Non-Valid mode.	Yes	Yes	0

Register 9-94. (DMAMODE2; PCI:1C0h, LOC:1C0h) DMA Channel 2 Mode (Continued)

Bit	Description	Read	Write	Value after Reset
17	**DMA Channel 2 Interrupt Select.** Writing 1 routes the DMA Channel 2 interrupt to the PCI Bus interrupt. Writing 0 routes the DMA Channel 2 interrupt to the 60x Bus LINTo# interrupt.	Yes	Yes	0
18	**DAC Chain Load.** When set to 1, enables the descriptor to load the PCI Dual Address Cycle value. Otherwise, it uses the register contents.	Yes	Yes	0
19	**EOT2# End Link.** Used only for DMA Scatter/Gather transfers. When EOT2# is asserted, value of 1 indicates the DMA transfer completes the current Scatter/Gather link but continues with the remaining Scatter/Gather transfers. When EOT2# is asserted, a value of 0 indicates the DMA transfer completes the current Scatter/Gather transfer, but does not continue with the remaining Scatter/Gather transfers.	Yes	Yes	0
20	**Valid Mode Enable.** Value of 0 indicates the Valid bit (DMASIZ2[31]) is ignored. Value of 1 indicates the DMA descriptors are processed only when the Valid bit is set (DMASIZ2[31]). When the Valid bit is set, the transfer count is 0, and the descriptor is not the last descriptor in the chain. The DMA controller then moves to the next descriptor in the chain. Value of 0 also indicates the Non-Valid mode descriptor-loading sequence, whereas a value of 1 indicates Valid Mode Ring Management descriptor-loading sequence.	Yes	Yes	0
21	**Valid Stop Control.** Value of 0 indicates the DMA Scatter/Gather controller continuously polls a descriptor with the Valid bit set to 0 (invalid descriptor) when the Valid Mode Enable bit is set (bit [20]=1). Value of 1 indicates the Scatter/Gather controller stops polling when the Valid bit with a value of 0 is detected (DMASIZ2[31]=0). In this case, the CPU must restart the DMA controller by setting the Start bit (DMACSR2[1]=1). A pause sets the DMA Done register.	Yes	Yes	0
31:22	***Reserved***.	Yes	No	0h

Register 9-95. (DMAPADR2; PCI:1C4H, LOC:1C4H when DMAMODE2[20]=0 or PCI:1C8H, LOC:1C8H when DMAMODE2[20]=1) DMA Channel 2 PCI Address

Bit	Description	Read	Write	Value after Reset
31:0	**PCI Address Register.** Indicates from where in PCI Memory space DMA transfers (reads or writes) start.	Yes	Yes	0h

Register 9-96. (DMALADR2; PCI:1C8Ch, LOC:1C8h when DMAMODE2[20]=0 or PCI:1CCH, LOC:1CCH when DMAMODE2[20]=1) DMA Channel 2 60x Address

Bit	Description	Read	Write	Value after Reset
31:0	**60x Address Register.** Indicates from where in 60x Bus Memory space DMA transfers (reads or writes) start.	Yes	Yes	0h

Register 9-97. (DMASIZ2; PCI:1CCh, LOC:1CCh when DMAMODE2[20]=0 or PCI:1C4h, LOC:1C4h when DMAMODE2[20]=1) DMA Channel 2 Transfer Size (Bytes)

Bit	Description	Read	Write	Value after Reset
22:0	**DMA Transfer Size (Bytes).** Indicates the number of bytes to transfer during a DMA operation.	Yes	Yes	0h
30:23	***Reserved***.	Yes	No	0h
31	**Valid.** When the Valid Mode Enable bit is enabled (DMAMODE2[20]=1), indicates the validity of this DMA descriptor.	Yes	Yes	0

Register 9-98. (DMADPR2; PCI:1D0h, LOC:1D0h) DMA Channel 2 Descriptor Pointer

Bit	Description	Read	Write	Value after Reset
0	**Descriptor Location.** Writing 1 indicates PCI Address space. Writing 0 indicates 60x Address space.	Yes	Yes	0
1	**End of Chain.** Writing 1 indicates end of chain. Writing 0 indicates not end of chain descriptor. (Same as Block mode.)	Yes	Yes	0
2	**Interrupt after Terminal Count.** Writing 1 causes an interrupt to be asserted after the terminal count for this descriptor is reached. Writing 0 disables interrupts from being asserted.	Yes	Yes	0
3	**Transfer Direction.** Writing 1 indicates transfers from the 60x Bus to the PCI Bus. Writing 0 indicates transfers from the PCI Bus to the 60x Bus.	Yes	Yes	0
31:4	**Next Descriptor Address.** Qword-aligned (bits [3:0]=0000).	Yes	Yes	0h

Register 9-99. (DMAMODE3; PCI:1D4h, LOC:1D4h) DMA Channel 3 Mode

Bit	Description	Read	Write	Value after Reset
1:0	**60x Bus Width.** Writing 00 indicates an 8-bit bus width. Writing 01 indicates a 16-bit bus width. Writing 10 indicates a 64-bit bus width. Writing 11 indicates a 32-bit bus width.	Yes	Yes	10
5:2	***Special Reserved.***	Yes	Yes	0h
6	***Special Reserved.***	Yes	Yes	1
7	***Special Reserved.***	Yes	Yes	0
8	**PLX Extended Burst Enable.** Writing 1 enables a PLX Extended Burst. Writing 0 disables a PLX Extended Burst.	Yes	Yes	0
9	**Scatter/Gather Mode.** Writing 1 indicates Scatter/Gather mode is enabled. For Scatter/Gather mode, the DMA source address, destination address, and byte count are loaded from memory in PCI or 60x Bus Address spaces. Writing 0 indicates Block mode is enabled.	Yes	Yes	0
10	**Done Interrupt Enable.** Writing 1 enables an interrupt when done. Writing 0 disables the interrupt when done. When DMA Clear Count mode is enabled (bit [16]=1), the interrupt does not occur until the byte count is cleared.	Yes	Yes	0
11	**60x Bus Addressing Mode.** Writing 1 holds the 60x Bus Address constant. Writing 0 indicates the 60x Bus Address is incremented.	Yes	Yes	0
12	**Demand Mode.** Writing 1 causes the DMA controller to operate in Demand mode. In Demand mode, the DMA controller transfers data when its DREQ3# input is asserted. Asserts DACK3# to indicate the current 60x Bus transfer is in response to DREQ3# input.	Yes	Yes	0
13	**Memory Write and Invalidate Mode for DMA Transfers.** When set to 1, the PCI 9610 performs Memory Write and Invalidate cycles to the PCI Bus. The PCI 9610 supports Memory Write and Invalidate sizes of 8 or 16 Lwords. Size is specified in the System Cache Line Size bits (PCICLSR[7:0]). When a size other than 8 or 16 is specified, the PCI 9610 performs Write transfers rather than Memory Write and Invalidate transfers. Transfers must start and end at cache line boundaries.	Yes	Yes	0
14	**DMA EOT3# Enable.** Writing 1 enables the EOT3# input pin. Writing 0 disables the EOT3# input pin.	Yes	Yes	0
15	***Special Reserved.***	Yes	Yes	0
16	**DMA Clear Count Mode.** Writing 1 clears the byte count in each Scatter/Gather descriptor when the corresponding DMA transfer is complete. Applies only in Non-Valid mode.	Yes	Yes	0
17	**DMA Channel 3 Interrupt Select.** Writing 1 routes the DMA Channel 3 interrupt to the PCI Bus interrupt. Writing 0 routes the DMA Channel 3 interrupt to the 60x Bus LINTo# interrupt.	Yes	Yes	0
18	**DAC Chain Load.** When set to 1, enables the descriptor to load the PCI Dual Address Cycle value. Otherwise, it uses the register contents.	Yes	Yes	0
19	**EOT3# End Link.** Used only for DMA Scatter/Gather transfers. When EOT3# is asserted, value of 1 indicates the DMA transfer completes the current Scatter/Gather link but continues with the remaining Scatter/Gather transfers. When EOT3# is asserted, a value of 0 indicates the DMA transfer completes the current Scatter/Gather transfer, but does not continue with the remaining Scatter/Gather transfers.	Yes	Yes	0
20	**Valid Mode Enable.** Value of 0 indicates the Valid bit (DMASIZ3[31]) is ignored. Value of 1 indicates the DMA descriptors are processed only when the Valid bit is set (DMASIZ3[31=1]). When the Valid bit is set, the transfer count is 0, and the descriptor is not the last descriptor in the chain. The DMA controller then moves to the next descriptor in the chain. Value of 0 also indicates Non-Valid mode descriptor-loading sequence, whereas a value of 1 indicates Valid mode Ring Management descriptor-loading sequence.	Yes	Yes	0

Register 9-99. (DMAMODE3; PCI:1D4h, LOC:1D4h) DMA Channel 3 Mode (Continued)

Bit	Description	Read	Write	Value after Reset
21	**Valid Stop Control.** Value of 0 indicates the DMA Scatter/Gather controller continuously polls a descriptor with the Valid bit set to 0 (invalid descriptor) when the Valid Mode Enable bit is set (bit [20]=1). Value of 1 indicates the Scatter/Gather controller stops polling when the Valid bit with a value of 0 is detected (DMASIZ3[31]=0). In this case, the CPU must restart the DMA controller by setting the Start bit (DMACSR3[1]=1). A pause sets the DMA Done register.	Yes	Yes	0
31:22	***Reserved***.	Yes	No	0h

Register 9-100. (DMAPADR3; PCI:1D8H, LOC:1D8H when DMAMODE3[20]=0 or PCI:1DCH, LOC:1DCH when DMAMODE3[20]=1) DMA Channel 3 PCI Address

Bit	Description	Read	Write	Value after Reset
31:0	**PCI Address Register.** Indicates from where in PCI Memory space DMA transfers (reads or writes) start.	Yes	Yes	0h

Register 9-101. (DMALADR3; PCI:1DCH, LOC:1DCH when DMAMODE3[20]=0 or PCI:1E0H, LOC:1E0H when DMAMODE3[20]=1) DMA Channel 3 60x Address

Bit	Description	Read	Write	Value after Reset
31:0	**60x Address Register.** Indicates from where in 60x Bus Memory space DMA transfers (reads or writes) start.	Yes	Yes	0h

Register 9-102. (DMASIZ3; PCI:1E0h, LOC:1E0h when DMAMODE3[20]=0 or PCI:1D8h, LOC:1D8h when DMAMODE3[20]=1) DMA Channel 3 Transfer Size (Bytes)

Bit	Description	Read	Write	Value after Reset
22:0	**DMA Transfer Size (Bytes).** Indicates the number of bytes to transfer during a DMA operation.	Yes	Yes	0h
30:23	***Reserved***.	Yes	No	0h
31	**Valid.** When the Valid Mode Enable bit is enabled (DMAMODE3[20]=1), indicates the validity of this DMA descriptor.	Yes	Yes	0

Register 9-103. (DMADPR3; PCI:1E4h, LOC:1E4h) DMA Channel 3 Descriptor Pointer

Bit	Description	Read	Write	Value after Reset
0	**Descriptor Location.** Writing 1 indicates PCI Address space. Writing 0 indicates 60x Address space.	Yes	Yes	0
1	**End of Chain.** Writing 1 indicates end of chain. Writing 0 indicates not end of chain descriptor. (Same as Block mode.)	Yes	Yes	0
2	**Interrupt after Terminal Count.** Writing 1 causes an interrupt to be asserted after the terminal count for this descriptor is reached. Writing 0 disables interrupts from being asserted.	Yes	Yes	0
3	**Transfer Direction.** Writing 1 indicates transfers from the 60x Bus to the PCI Bus. Writing 0 indicates transfers from the PCI Bus to the 60x Bus.	Yes	Yes	0
31:4	**Next Descriptor Address.** Qword aligned (bits [3:0]=0000).	Yes	Yes	0h

Register 9-104. (DMACSR2; PCI:1E8h, LOC:1E8h) DMA Channel 2 Command/Status

Bit	Description	Read	Write	Value after Reset
0	**Channel 2 Enable.** Writing 1 enables Channel 2 to transfer data. Writing 0 disables Channel 2 from starting a DMA transfer, and when in the process of transferring data, suspends the transfer (pause).	Yes	Yes	0
1	**Channel 2 Start.** Writing 1 causes Channel 2 to start transferring data when Channel 2 is enabled.	No	Yes/Set	0
2	**Channel 2 Abort.** Writing 1 causes Channel 2 to abort current transfer. Channel 2 Enable bit must be cleared (bit [0]=0). Sets Channel 2 Done (bit [4]=1) when abort is complete.	No	Yes/Set	0
3	**Channel 2 Clear Interrupt.** Writing 1 clears Channel 2 interrupts.	No	Yes/Clr	0
4	**Channel 2 Done.** Reading 1 indicates a channel transfer is complete. Reading 0 indicates a channel transfer is not complete.	Yes	No	1
7:5	***Reserved***.	Yes	No	000

Register 9-105. (DMACSR3; PCI:1E9h, LOC:1E9h) DMA Channel 3 Command/Status

Bit	Description	Read	Write	Value after Reset
0	**Channel 3 Enable.** Writing 1 enables Channel 3 to transfer data. Writing 0 disables Channel 3 from starting a DMA transfer, and when in the process of transferring data, suspends the transfer (pause).	Yes	Yes	0
1	**Channel 3 Start.** Writing 1 causes Channel 3 to start transferring data when Channel 3 is enabled.	No	Yes/Set	0
2	**Channel 3 Abort.** Writing 1 causes the channel to abort current transfer. Channel 1 Enable bit must be cleared (bit [0]=0). Sets Channel 3 Done (bit [4]=1) when abort is complete.	No	Yes/Set	0
3	**Channel 3 Clear Interrupt.** Writing 1 clears Channel 3 interrupts.	No	Yes/Clr	0
4	**Channel 3 Done.** Reading 1 indicates a channel transfer is complete. Reading 0 indicates a channel transfer is not complete.	Yes	No	1
7:5	***Reserved***.	Yes	No	000

Register 9-106. (DMATHR23; PCI:1F0h, LOC:1F0h) DMA Channels 2 and 3 Threshold

Bit	Description	Read	Write	Value after Reset
3:0	**DMA Channel 2 PCI-to-60x Almost Full (C2PLAF).** Number of full entries (divided by two, minus one) in the FIFO before requesting the 60x Bus for writes. (C2PLAF+1) + (C2PLAE+1) should be ≤ a FIFO Depth of 32.	Yes	Yes	0h
7:4	**DMA Channel 2 60x-to-PCI Almost Empty (C2LPAE).** Number of empty entries (divided by two, minus one) in the FIFO before requesting the 60x Bus for reads. (C2LPAF+1) + (C2LPAE+1) should be ≤ a FIFO depth of 32.	Yes	Yes	0h
11:8	**DMA Channel 2 60x-to-PCI Almost Full (C2LPAF).** Number of full entries (divided by two, minus one) in the FIFO before requesting the PCI Bus for writes.	Yes	Yes	0h
15:12	**DMA Channel 2 PCI-to-60x Almost Empty (C2PLAE).** Number of empty entries (divided by two, minus one) in the FIFO before requesting the PCI Bus for reads.	Yes	Yes	0h
19:16	**DMA Channel 3 PCI-to-60x Almost Full (C3PLAF).** Number of full entries (divided by two, minus one) in the FIFO before requesting the 60x Bus for writes. (C3PLAF+1) + (C3PLAE+1) should be ≤ a FIFO Depth of 32.	Yes	Yes	0h
23:20	**DMA Channel 3 60x-to-PCI Almost Empty (C3LPAE).** Number of empty entries (divided by two, minus one) in the FIFO before requesting the 60x Bus for reads. (C3LPAF+1) + (C3LPAE+1) should be ≤ a FIFO depth of 32.	Yes	Yes	0h
27:24	**DMA Channel 3 60x-to-PCI Almost Full (C3LPAF).** Number of full entries (divided by two, minus one) in the FIFO before requesting the PCI Bus for writes.	Yes	Yes	0h
31:28	**DMA Channel 3 PCI-to-60x Almost Empty (C3PLAE).** Number of empty entries (divided by two, minus one) in the FIFO before requesting the PCI Bus for reads.	Yes	Yes	0h

Register 9-107. (DMADAC2; PCI:1F4h, LOC:1F4h) DMA Channel 2 PCI Dual Address Cycle Upper Address

Bit	Description	Read	Write	Value after Reset
31:0	**Upper 32 Bits of the PCI Dual Address Cycle PCI Address during DMA Channel 2 Cycles.** When set to 0h, the PCI 9610 performs a 32-bit DMA Channel 2 Address access.	Yes	Yes	0h

Register 9-108. (DMADAC3; PCI:1F8h, LOC:1F8h) DMA Channel 3 PCI Dual Address Cycle Upper Address

Bit	Description	Read	Write	Value after Reset
31:0	**Upper 32 Bits of the PCI Dual Address Cycle PCI Address during DMA Channel 3 Cycles.** When set to 0h, the PCI 9610 performs a 32-bit DMA Channel 3 Address access.	Yes	Yes	0h

9.6 MESSAGING QUEUE REGISTERS

Register 9-109. (OPQIS; PCI:30h, LOC:B0h) Outbound Post Queue Interrupt Status

Bit	Description	Read	Write	Value after Reset
2:0	***Reserved.***	Yes	No	000
3	**Outbound Post Queue Interrupt.** Set when the Outbound Post Queue is not empty. Not affected by the Interrupt Mask bit.	Yes	No	0
31:4	***Reserved.***	Yes	No	0h

Register 9-110. (OPQIM; PCI:34h, LOC:B4h) Outbound Post Queue Interrupt Mask

Bit	Description	Read	Write	Value after Reset
2:0	***Reserved.***	Yes	No	000
3	**Outbound Post Queue Interrupt Mask.** Writing 1 masks the interrupt.	Yes	Yes	1
31:4	***Reserved.***	Yes	No	0h

Register 9-111. (IQP; PCI:40h) Inbound Queue Port

Bit	Description	Read	Write	Value after Reset
31:0	Value written by the PCI Master is stored into the Inbound Post Queue, which is located in 60x Bus Memory at the address pointed to by the Queue Base Address + Queue Size + Inbound Post Head Pointer. From the time of the PCI write until the 60x Bus Memory write and update of the Inbound Post Queue Head Pointer, further accesses to this register result in a Retry. A 60x Bus interrupt is asserted when the Inbound Post Queue is not empty. When the port is read by the PCI Master, the value is read from the Inbound Free Queue, which is located in 60x Bus Memory at the address pointed to by the Queue Base Address + Inbound Free Tail Pointer. When the queue is empty, FFFFFFFh is returned.	PCI	PCI	0h

Register 9-112. (OQP; PCI:44h) Outbound Queue Port

Bit	Description	Read	Write	Value after Reset
31:0	Value written by the PCI Master is stored into the Outbound Free Queue, which is located in 60x Bus Memory at the address pointed to by the Queue Base Address + 3*Queue Size + Outbound Free Head Pointer. From the time of the PCI write until the 60x Memory write and update of the Outbound Free Queue Head Pointer, further accesses to this register result in a Retry. When the queue fills up, a 60x NMI interrupt is asserted. When the port is read by the PCI Master, the value is read from the Outbound Post Queue, which is located in 60x Bus Memory at the address pointed to by the Queue Base Address + 2*Queue Size + Outbound Post Tail Pointer. When the queue is empty, FFFFFFFh is returned. A PCI interrupt is asserted when the Outbound Post Queue is not empty.	PCI	PCI	0h

Register 9-113. (MQCR; PCI:C0h, LOC:140h) Messaging Queue Configuration

Bit	Description	Read	Write	Value after Reset
0	**Queue Enable.** Writing 1 allows accesses to the Inbound and Outbound Queue ports. When cleared to 0, writes are accepted but ignored and reads return FFFFFFFFh.	Yes	Yes	0
5:1	**Circular Queue Size.** Contains the size of one of the circular FIFO queues. Each of the four queues are the same size. Queue Size Encoding values: Bits [5:1] — Number of entries — Total size 00001 — 4-KB entries — 64 KB 00010 — 8-KB entries — 128 KB 00100 — 16-KB entries — 256 KB 01000 — 32-KB entries — 512 KB 10000 — 64-KB entries — 1 MB	Yes	Yes	00001
31:6	***Reserved.***	Yes	No	0h

Register 9-114. (QBAR; PCI:C4h, LOC:144h) Queue Base Address

Bit	Description	Read	Write	Value after Reset
19:0	***Reserved.***	Yes	No	0h
31:20	**Queue Base Address.** 60x Bus Memory base address of circular queues. Queues must be aligned on a 1 MB boundary.	Yes	Yes	0h

Register 9-115. (IFHPR; PCI:C8h, LOC:148h) Inbound Free Head Pointer

Bit	Description	Read	Write	Value after Reset
1:0	***Reserved.***	Yes	No	00
19:2	**Inbound Free Head Pointer.** 60x Bus Memory Offset for the Inbound Free Queue. Maintained by the 60x CPU software.	Yes	Yes	0h
31:20	**Queue Base Address.**	Yes	No	0h

Register 9-116. (IFTPR; PCI:CCh, LOC:14Ch) Inbound Free Tail Pointer

Bit	Description	Read	Write	Value after Reset
1:0	***Reserved.***	Yes	No	00
19:2	**Inbound Free Tail Pointer.** 60x Bus Memory offset for the Inbound Free Queue. Maintained by the hardware and incremented modulo the queue size.	Yes	Yes	0h
31:20	**Queue Base Address.**	Yes	No	0h

Register 9-117. (IPHPR; PCI:D0h, LOC:150h) Inbound Post Head Pointer

Bit	Description	Read	Write	Value after Reset
1:0	***Reserved.***	Yes	No	00
19:2	**Inbound Post Head Pointer.** 60x Bus Memory offset for the Inbound Post Queue. Maintained by the hardware and incremented modulo the queue size.	Yes	Yes	0h
31:20	**Queue Base Address.**	Yes	No	0h

Register 9-118. (IPTPR; PCI:D4h, LOC:154h) Inbound Post Tail Pointer

Bit	Description	Read	Write	Value after Reset
1:0	***Reserved.***	Yes	No	00
19:2	**Inbound Post Tail Pointer.** 60x Bus Memory offset for the Inbound Post Queue. Maintained by the 60x CPU software.	Yes	Yes	0h
31:20	**Queue Base Address.**	Yes	No	0h

Register 9-119. (OFHPR; PCI:D8h, LOC:158h) Outbound Free Head Pointer

Bit	Description	Read	Write	Value after Reset
1:0	***Reserved.***	Yes	No	00
19:2	**Outbound Free Head Pointer.** 60x Bus Memory offset for the Outbound Free Queue. Maintained by the hardware and incremented modulo the queue size.	Yes	Yes	0h
31:20	**Queue Base Address.**	Yes	No	0h

Register 9-120. (OFTPR; PCI:DCh, LOC:15Ch) Outbound Free Tail Pointer

Bit	Description	Read	Write	Value after Reset
1:0	***Reserved.***	Yes	No	00
19:2	**Outbound Free Tail Pointer.** 60x Bus Memory offset for the Outbound Free Queue. Maintained by the 60x CPU software.	Yes	Yes	0h
31:20	**Queue Base Address.**	Yes	No	0h

Register 9-121. (OPHPR; PCI:E0h, LOC:160h) Outbound Post Head Pointer

Bit	Description	Read	Write	Value after Reset
1:0	***Reserved.***	Yes	No	00
19:2	**Outbound Post Head Pointer.** 60x Bus Memory offset for the Outbound Post Queue. Maintained by the 60x CPU software.	Yes	Yes	0h
31:20	**Queue Base Address.**	Yes	No	0h

Register 9-122. (OPTPR; PCI:E4h, LOC:164h) Outbound Post Tail Pointer

Bit	Description	Read	Write	Value after Reset
1:0	***Reserved.***	Yes	No	00
19:2	**Outbound Post Tail Pointer.** 60x Bus Memory offset for the Outbound Post Queue. Maintained by the hardware and incremented modulo the queue size.	Yes	Yes	0h
31:20	**Queue Base Address.**	Yes	No	0h

Register 9-123. (QSR; PCI:E8h, LOC:168h) Queue Status/Control

Bit	Description	Read	Write	Value after Reset
0	**I_2O Decode Enable.** When set, replaces the MBOX0 and MBOX1 registers with the Inbound and Outbound Queue Port registers and redefines Space 1 as PCI Base Address 0 to be accessed by PCIBAR0. Former Space 1 registers LAS1RR, LAS1BA, and LBRD1 should be programmed to configure their shared I_2O Memory space, defined as PCI Base Address 0.	Yes	Yes	0
1	**Queue 60x Bus Space Select.** When set to 0, use the 60x Address Space 0 Bus Region descriptor for Queue accesses. When set to 1, use the 60x Address Space 1 Bus Region descriptor for Queue accesses.	Yes	Yes	0
2	**Outbound Post Queue Prefetch Enable.** Writing 1 causes prefetching to occur from the Outbound Post Queue when it is not empty.	Yes	Yes	0
3	**Inbound Free Queue Prefetch Enable.** Writing 1 causes prefetching to occur from the Inbound Free Queue when it is not empty.	Yes	Yes	0
4	**Inbound Post Queue Interrupt Mask.** Writing 1 masks the interrupt.	Yes	Yes	1
5	**Inbound Post Queue Interrupt Not Empty.** Set when the Inbound Post Queue is not empty. Not affected by the Interrupt Mask bit.	Yes	No	0
6	**Outbound Free Queue Overflow Interrupt Mask.** Masks the interrupt when set.	Yes	Yes	1
7	**Outbound Free Queue Overflow Interrupt Full.** Set when the Outbound Free Queue becomes full. When bit 6 is set to 1, a LINTo# interrupt is asserted. Writing 1 clears the bit to 0.	Yes	Yes/Clr	0
31:8	***Unused.***	Yes	No	0h

10 PIN DESCRIPTION

10.1 PIN SUMMARY

Tables in this section describe each PCI 9610 pin types:

- PCI System Bus Interface
- JTAG
- CompactPCI Hot Swap
- System
- Serial EEPROM Interface
- Power and Ground
- 60x Bus
- PLX Extended Burst Mode
- Miscellaneous

For a visual view of the pinout, refer to Section 11.

The IDDQEN# pin has an internal pull-down resistor.

The pins in the following table have internal pull-up resistors.

Table 10-1. Pins with Internal Pull-Up Resistors

AACK#	CCS#	DREQ[3:1]#	EEDI/EEDO
EOT[3:0]#	HOSTEN#	LA[0:31]	LAP[0:3]
LD[0:63]	LDP[0:7]	LINTi#	LINTo#
LRESET#	MCP#/DPE#	MDREQ#	PLX_DVAL#
PLX_MT#	PLX_TBST#	PMEREQ#	PSDVAL#
TA#	TBST#	TEA#	TS#
TSIZ[0:3]	TT[0:4]	USERo/DREQ0#	

The pins in the following table have no internal resistors. A pull-up or pull-down is recommended, based upon the pin functionality.

Table 10-2. Pins with No Internal Resistors

ABB#	APE#	ARTRY#	BG#
BIGEND#	BR#	DACK[3:1]#	DBB#
DBG#	EECS	EESK	GBL#
USERi/DACK0#			

The following table lists abbreviations used in this section to represent various pin types.

Table 10-3. Pin Type Abbreviations

Abbreviation	Pin Type
I/O	Input and output
I	Input only
O	Output only
TS	Three-state
OD	Open drain
TP	Totem pole
STS	Sustained three-state—driven high for one CLK before float
DTS	Driven three-state—driven high for one-half CLK before float (ABB# and DBB# only)

All 60x Bus internal pull-ups and pull-downs go through a 50k-ohm resistor.

All 60x Bus I/O pins should have external pull-ups or pull-downs, which depend upon the application and pin polarity. (Use approximately 3k to 10k ohms.) This is recommended due to the weak value of the internal pull-ups and pull-downs.

Unspecified pins are not connected (NC).

Note for PCI Pins: ***DO NOT*** *pull any pins up or down unless the PCI 9610 is being used in an embedded design. Refer to PCI Local Bus Specification, r2.1, page 123.*

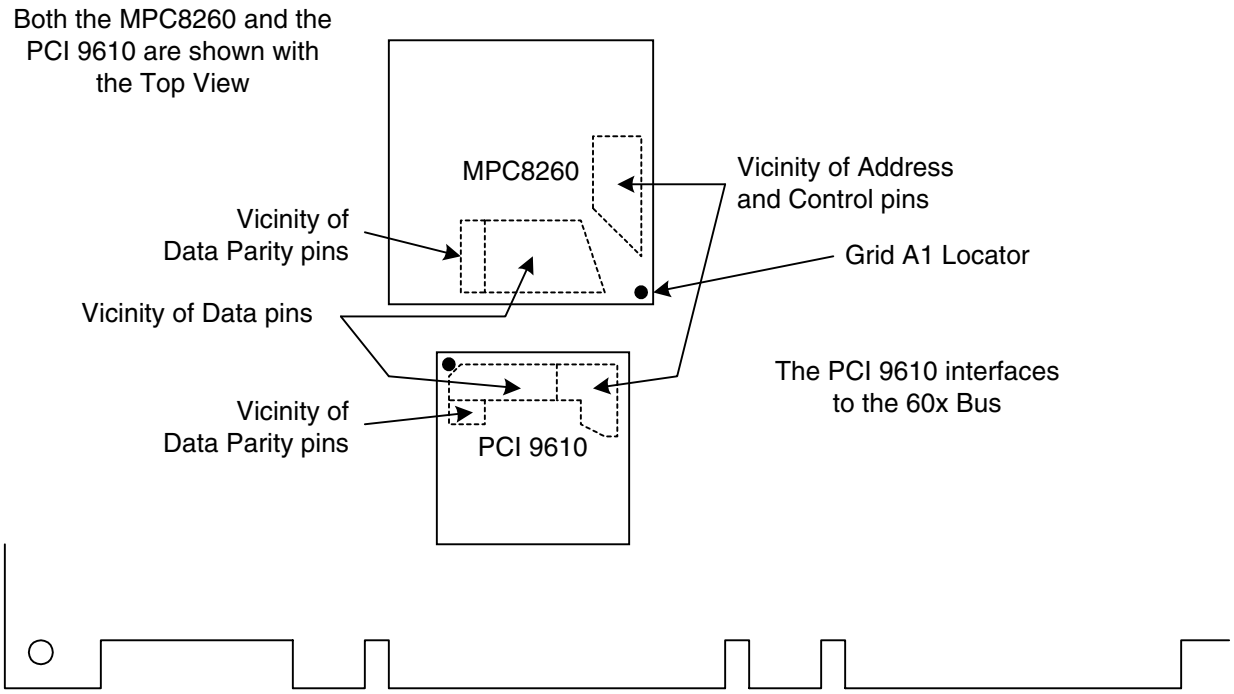


Figure 10-1. MPC8260 and PCI 9610 Orientation on a 64-Bit PCI Card

10.2 PINOUT

Note: *Symbol names inside { } are the short names, when different from the full symbol name, used in the ball grid drawing (Figure 12-2 on page 12-2).*

Table 10-4. PCI System Bus Interface Pins

Symbol	Signal Name	Total Pins	Pin Type	Pin Number	Function
ACK64#	64-Bit Transfer Acknowledge	1	I/O STS PCI	AF18	Asserted by the Direct Slave in response to REQ64# to acknowledge a 64-bit data transfer.
AD[63:0]	Address and Data	64	I/O TS PCI	AF23, AD22, AE26, AD25, AD26, AC25, AC24, AC26, AB25, AB24, AB26, AA25, AA23, AA24, AA26, Y25, Y26, Y24, W25, W23, W26, W24, V25, U25, V24, U26, T25, U24, T26, R25, R26, T24, AF3, AE4, AD4, AF4, AE5, AC5, AD5, AF5, AD6, AF6, AE7, AF7, AE8, AF8, AD8, AE9, AC12, AF13, AD12, AE14, AF14, AD13, AE15, AD14, AD15, AF16, AE17, AD16, AF17, AC17, AE18, AD17	PCI Multiplexed Address/Data Bus.
C/BE[7:0]# {CBE[7:0]#}	Bus Command and Byte Enables	8	I/O TS PCI	AD20, AE22, AD21, AE23, AE6, AF9, AE13, AF15	All multiplexed on the same PCI pins. During the Address phase of a transaction, defines the bus command. During the Data phase, used as byte enables. Refer to the PCI spec for additional information.
DEVSEL#	Device Select	1	I/O STS PCI	AC10	When actively driven, indicates the driving device has decoded its address as the Target of the current access. As input, indicates whether any device on the bus is selected.
FRAME# {FRM#}	Cycle Frame	1	I/O STS PCI	AE10	Driven by the current Master to indicate the beginning and duration of an access. FRAME# is asserted to indicate a Bus transaction is beginning. While FRAME# is asserted, Data transfers continue. When FRAME# is de-asserted, the transaction is in the final Data phase.

Table 10-4. PCI System Bus Interface Pins (Continued)

Symbol	Signal Name	Total Pins	Pin Type	Pin Number	Function
GNT0# REQ# {R_G0#}	Internal Arbiter Grant 0 External Arbiter Request	1	O O STS PCI	AE3	GNT0#: When the internal PCI arbiter is enabled, the PCI GNT0# signal is an output to an arbitrating master. The PCI 9610 arbiter asserts GNT0# to grant the PCI Bus to the Master. REQ#: When the internal PCI arbiter is disabled, GNT0# becomes the REQ# output from the PCI 9610 to an external arbiter. The PCI 9610 asserts REQ# to request the PCI Bus.
GNT[6:1]#	Internal Arbiter Grant 6–1	6	O TP	P26, P25, AE21, AF20, AE20, AF19	When the internal PCI arbiter is enabled, the PCI GNT[6:1]# signals are outputs, one each to an arbitrating master. The PCI 9610 arbiter asserts one of the GNT# signals to grant the PCI Bus to the corresponding master. ***Note:*** *PCI Arbiter pins are type "TP" when the PCI arbiter is enabled. Otherwise, they are left floating.*
IDSEL	Initialization Device Select	1	I	AC6	Used as a chip select during Configuration Read and Write transactions.
INTA#	Interrupt A	1	I/O OD PCI	AC3	As an input, available only when HOSTEN# is asserted (drives LINTo# onto the Processor Bus). The PCI 9610 is a PCI Host. As an output, the PCI 9610 drives INTA# to perform a PCI Interrupt request.
IRDY#	Initiator Ready	1	I/O STS PCI	AD9	Indicates ability of the initiating agent (Bus Master) to complete the current Data phase of the transaction.
LOCK# {LCK#}	Lock	1	I/O STS PCI	AD10	Indicates an atomic operation that may require multiple transactions to complete.
PAR	Parity	1	I/O TS PCI	AD11	Even parity across AD[31:0] and C/BE[3:0]#. All PCI agents require parity generation. PAR is stable and valid one clock after the Address phase. For Data phases, PAR is stable and valid one clock after either IRDY# is asserted on a Write transaction or TRDY# is asserted on a Read transaction. Once PAR is valid, it remains valid until one clock after the current Data phase completes.
PAR64	Upper 32 Bits Parity	1	I/O TS PCI	AC22	Even parity across AD[63:32] and C/BE[7:4]#. All PCI agents require parity generation. PAR is stable and valid one clock after the Address phase. For Data phases, PAR is stable and valid one clock after either IRDY# is asserted on a Write transaction or TRDY# is asserted on a Read transaction. Once PAR is valid, it remains valid until one clock after the current Data phase completes.
PCLK	Clock	1	I	AE16	Provides timing for all transactions on PCI and is an input to every PCI device. The PCI 9610 PCI Bus operates up to 66 MHz.
PERR#	Parity Error	1	I/O STS PCI	AF11	Reports data parity errors during all PCI transactions, except during a special cycle.

Table 10-4. PCI System Bus Interface Pins (Continued)

Symbol	Signal Name	Total Pins	Pin Type	Pin Number	Function
PME#	Power Management Event	1	O OD PCI	AB2	Asserted to alert system to a Power Management Event.
REQ0# GNT# {G_R0#}	Internal Arbiter Request 0 External Arbiter Grant	1	I	AF2	REQ0#: When the internal PCI arbiter is enabled, the PCI REQ0# signal is an input from an arbitrating master. REQ0# is asserted to the PCI 9610 arbiter by the Master to request the PCI Bus. GNT#: When the internal PCI arbiter is disabled, REQ0# becomes the GNT# input to the PCI 9610 from an external arbiter. The arbiter asserts GNT# to grant the PCI Bus to the PCI 9610. REQ[6:1]# are not used.
REQ[6:1]#	Internal Arbiter Request 6–1	6	I	R24, R23, AF21, AD19, AC19, AD18	When the internal PCI arbiter is enabled, the PCI REQ[6:1]# signals are inputs, one each from an arbitrating master. REQ[6:1]# is asserted to the PCI 9610 arbiter by the corresponding master to request the PCI Bus.
REQ64#	64-Bit Transfer Request	1	I/O STS PCI	AE19	Asserted with FRAME# by the PCI Master to request a 64-bit data transfer.
RST#	Reset	1	I/O	AD1	As an input, used to bring PCI-specific registers, sequencers, and signals to a consistent state. As an output, available only when HOSTEN# is asserted, causing the entire PCI Bus to reset by way of LRESET# assertion. The PCI 9610 is a PCI Host.
SERR#	Systems Error	1	I/O OD PCI	AF12	As an input, available only when HOSTEN# is asserted, causing LINTo# to be asserted any time the PCI error occurs. The PCI 9610 is a PCI Host. As an output, reports address parity errors, data parity errors on the Special Cycle command, or any other system error where the result is catastrophic.
STOP# {STP#}	Stop	1	I/O STS PCI	AE11	Indicates the current Target is requesting that the Master stop the current transaction.
TRDY#	Target Ready	1	I/O STS PCI	AF10	Indicates ability of the Target agent (selected device) to complete the current Data phase of the transaction.
Total		103			

Table 10-5. JTAG Pins

Symbol	Signal Name	Total Pins	Pin Type	Pin Number	Function
TCK	Test Clock Input	1	I	AB4	Clock source for the PCI 9610 test access port (TAP). Legal rates for this pin are equal to the LCLK rate or less than one-half the LCLK rate.
TDI	Test Data Input	1	I	AD2	Used to input data into the TAP. When the TAP enables this pin, it is sampled on the rising edge of TCK and the sampled value is input to the selected TAP shift register.
TDO	Test Data Output	1	O TS PCI	AB3	Used to transmit serial data from the PCI 9610 TAP. Data from the selected shift register is shifted out of TDO.
TMS	Test Mode Select	1	I	AC1	Sampled by the TAP on the rising edge of TCK. The TAP state machine uses this pin to determine the TAP mode.
TRST#	Test Reset	1	I	AC2	Resets JTAG.
Total		5			

Table 10-6. CompactPCI Hot Swap Pins

Symbol	Signal Name	Total Pins	Pin Type	Pin Number	Function
64EN#	CompactPCI 64-Bit Enable	1	I	N25	Input that enables 64-bit CompactPCI operation.
BD_SEL#	CompactPCI Board Select	1	I	AA3	CompactPCI board select for Hot Swap system. For non-CompactPCI systems, this pin should be grounded.
CPCISW	CompactPCI Switch Sense	1	I	N24	Input that monitors CompactPCI board latch status. For non-CompactPCI systems, this pin should be pulled high.
ENUM# {ENM#}	Enumeration	1	O OD PCI	N26	Interrupt output asserted when an adapter using the PCI 9610 has been inserted or is ready to be removed from a PCI slot.
LEDon#	CompactPCI LED On	1	O TP 24 mA	M25	Activates the CompactPCI Hot Swap board indicator LED.
Total		5			

Table 10-7. System Pins

Symbol	Signal Name	Total Pins	Pin Type	Pin Number	Function
HOSTEN# {HSTEN#}	Host Enable	1	I	W1	When asserted, configures the PCI 9610 as a host bridge, setting Reset and Interrupt signal directions for system board applications. When de-asserted, configures the PCI 9610 as a peripheral bridge, setting Reset and Interrupt signal directions for peripheral board applications.
IDDQEN#	Buffered PCI Frame	1	I	AA2	Provides main power status to the PCI 9610 D_{3cold} Power Management logic. For all normal operations, this pin should be connected directly to the 3.3V power line. For IDDQ' tests, the pin should be grounded.
LRESET# {LRST#}	60x Bus Reset	1	I/O TP 24 mA	V1	As an input, available only when HOSTEN# is asserted, causing RST# to be asserted on the PCI Bus. The PCI 9610 is a PCI Host. As an output, asserted when the PCI 9610 chip is in reset. Can be used to reset the board backend logic.
PMEREQ#	Power Management Event Request	1	I	AA4	Requests a Power Management Event during a D_{3cold} power state. Other Power Management events should be completed through the PCI 9610 Power Management registers.
Total		4			

Table 10-8. Serial EEPROM Interface Pins

Symbol	Signal Name	Total Pins	Pin Type	Pin Number	Function
EECS	Serial EEPROM Chip Select	1	O TP 12 mA	Y2	Serial EEPROM chip select.
EEDI/EEDO	Serial EEPROM Data In/ Serial EEPROM Data Out	1	I/O TP 12 mA	Y1	Multiplexed Write and Read data to the serial EEPROM pin.
EESK	Serial Data Clock	1	O TP 12 mA	W4	Serial EEPROM clock pin.
Total		3			

Table 10-9. Power and Ground Pins

Symbol	Signal Name	Total Pins	Pin Type	Pin Number	Function
$2.5V_{AUX}$ {Vaux2v}	PME 2.5V D_{3cold} Power	1	I	Y3	2.5V to PME logic during D_{3cold} state. For Power Management systems, connect directly to 2.5V regulated power line from Card_V_{AUX}. Otherwise, connect directly to 2.5V power line.
Card_V_{AUX} {Vaux3v}	PME 3.3V D_{3cold} Power	1	I	AA1	3.3V to PME logic during D_{3cold} state. Refer to the *PCI Power Management Interface Specification, r1.1*, Figure 12. For non-Power Management systems, connect directly to 3.3V.

Table 10-9. Power and Ground Pins (Continued)

Symbol	Signal Name	Total Pins	Pin Type	Pin Number	Function
PRESENT_DET {PRESDET}	3.3V V_{AUX} Present Detect Power	1	I	W3	When sampled as 1, 3.3V_{AUX} power is present and PME# assertion in D_{3cold} is supported. When sampled as 0, 3.3V_{AUX} power is not present and PME# assertion in D_{3cold} is not supported by the PME_Support D_{3cold} bit (PMC[15]). Refer to the *PCI Power Management Interface Specification, r1.1*, Figure 12. For non-Power Management systems, connect directly to ground.
V_{BB} {VBB}	Analog BIAS Voltage	1	I	AD23	Connect directly to ground.
V_{CORE} {Vcor}	Core Power	8	I	B4, B26, D17, P2, V26, AD7, AE1, AF22	2.5V to core.
V_{DDA} {VDDA}	Analog Power	1	I	AF25	Connect directly to 3.3V power line.
V_{IO} {VIO}	PCI System Voltage	4	I	P24, AB1, AE12, AE24	System voltage select, 3.3 or 5V, from PCI Bus.
V_{RING} {Vdd}	I/O Ring Power	16	I	D7, D11, D16, D20, G4, G23, L4, L23, T4, T23, Y4, Y23, AC7, AC11, AC16, AC20	3.3V to I/O ring.
V_{SS} {GND}	Ground	109	I	A1, A26, B2, B25, C3, C24, D4–D6, D9, D12–D14, D18, D21–D23, E23, J4, J23, K10–K17, L10–L17, M10–M17, M23, N4, N10–N17, N23, P4, P10–P17, P23, R10–R17, T10–T17, U10–U17, U23, V3, V4, V23, AB23, AC4, AC8, AC9, AC13–AC15, AC18, AC21, AC23, AD3, AD24, AE2, AE25, AF1, AF26	Ground.
V_{SSA} {VSSA}	Analog Ground	1	I	AF24	Connect directly to ground.
Total		143			

Table 10-10. 60x Bus Pins

Symbol	Signal Name	Total Pins	Pin Type	Pin Number	Function
AACK#	Address Acknowledge	1	I/O TS 24 mA	H25	As an input, when asserted, indicates that a 60x Bus slave is terminating the Address tenure. On the cycle following the AACK# assertion, the PCI 9610 releases the Address Tenure-Related signals to the high-impedance state and samples ARTRY#. As an output, when asserted, indicates that a Transaction Address tenure is terminated. On the cycle following the assertion of AACK#, the Bus Master releases the Address, Tenure-Related signals to the high-impedance state and samples ARTRY#.
ABB#	Address Bus Busy	1	I/O DTS 24 mA	K26	As an input, when asserted, indicates that an external device is the Address Bus Master. When de-asserted, indicates that the Address Bus may be available for PCI 9610 use (refer to BG#). As an output, indicates that the PCI 9610 is the current Address Bus Master. The PCI 9610 may not assume Address Bus ownership in case a Bus request is internally cancelled by the cycle a qualified BG# could recognize.
APE#	Address Parity Error	1	OD 12 mA	L25	Indicates detection of the wrong Address Bus parity.
ARTRY#	Address Retry	1	I/O TS 24 mA	L24	As an input, when asserted and the PCI 9610 is the Address Bus Master, indicates that the PCI 9610 must Retry the preceding Address tenure and immediately de-assert BR# (when asserted). When the associated Data tenure has begun, the PCI 9610 also immediately aborts the Data tenure although the Burst data was received. When the PCI 9610 is not the Address Bus Master, this input indicates that the PCI 9610 should de-assert BR# for one Bus Clock cycle immediately after an external device asserts ARTRY# to permit a copy-back operation to the main memory. ***Note:*** *The subsequent address presented to the Address Bus may not be the one that generated the ARTRY# assertion.* As an output, when asserted, indicates that the PCI 9610 detects a condition in which an Address tenure must Retry. When the PCI 9610 60x Bus Master must update memory as a result of a snoop that caused the Retry, the PCI 9610 asserts BR# the second cycle after AACK# when ARTRY# is asserted.

Table 10-10. 60x Bus Pins (Continued)

Symbol	Signal Name	Total Pins	Pin Type	Pin Number	Function
BG#	Bus Grant	1	I	H26	When asserted, indicates that the PCI 9610 may, with proper qualification, begin a Bus transaction and assume Address Bus ownership. A Qualified Bus Grant (QBG) is generally determined from the bus state as follows: QBG = BG# and ˜ABB# and ˜ARTRY# where ARTRY# is only during the cycle after AACK#. ***Note:*** *BR# assertion is not required for a qualified Bus grant (for Bus parking).*
BR#	Bus Request	1	O 24 mA	B15	When asserted, indicates that the PCI 9610 has a Bus transaction to perform and is waiting for a qualified BG# to begin the Address tenure.
DBB#	Data Bus Busy	1	I/O DTS 24 mA	B17	As an input, when asserted, indicates that an external device is the Data Bus Master. When de-asserted, indicates that the Data Bus is free (with proper qualification, refer to DBG#) for use by the PCI 9610. As an output, when asserted, indicates the PCI 9610 is the Data Bus Master. The PCI 9610 always assumes Data Bus Mastership when it needs the Data Bus and determines a qualified Data Bus grant (refer to DBG#).
DBG#	Data Bus Grant	1	I	A18	Indicates that the PCI 9610 may, with proper qualification, assume Data Bus ownership.
GBL#	Global	1	I/O TS 24 mA	A17	When asserted, indicates that the transaction is global and should be snooped by other devices. Also indicates an attempt to read or write cacheable space.
LA[0:31]	60x Bus Address	32	I/O TS 24 mA	H23, G24, F26, F23, D26, D25, F24, E26, E25, C26, D24, C25, E24, C23, B23, A24, B24, A25, A23, C21, B21, A22, C22, B22, C20, A20, B20, A21, B19, D19, C19, A19	Specifies the physical address of the Bus transaction. For Burst or Extended operations, the address is a double-word.
LAP[0:3]	60x Bus Address Parity	4	I/O TS 24 mA	A15, B16, C17, A16	Odd parity for each address byte.
LCLK	60x Bus Clock	1	I	C8	60x Bus Clock. Maximum frequency = 66 MHz.

Table 10-10. 60x Bus Pins (Continued)

Symbol	Signal Name	Total Pins	Pin Type	Pin Number	Function
LD[0:63]	60x Bus Data	64	I/O TS 24 mA	K4, H4, F2, A2, A3, B7, B11, B12, K1, H2, E4, B1, C7, D8, D10, B13, J3, G3, E1, C2, B5, C10, C14, C13, K2, G1, E3, C1, A5, A9, C15, C11, J1, G2, E2, A4, C6, B8, B9, A14, J2, F1, D2, C5, A7, A8, C12, A13, H3, F3, D3, B3, B6, A10–A12, H1, F4, D1, C4, A6, C9, B10, D15	Data Bus.
LDP[0:7]	60x Bus Data Parity	8	I/O TS 24 mA	M4, N2, L3, M1, M2, L1, K3, L2	Represents odd parity for each of eight bytes of Data Write transactions, which means that an odd number of bits, including the Parity bit, are driven high.
MCP#/ DPE# {MCPDPE#}	Data Parity Error	1	I/O TS 24 mA	B18	MCP#: PowerQUICC II Bus Data Parity indicator. DPE#: indicates a Data Parity error was detected on the 60x Bus.
PSDVAL#	Data Valid	1	I/O TS 24 mA	C16	Indicates that a Data transfer (full- or partial-beat) successfully completed. ***Note:*** *PSDVAL# must be asserted for each data beat in a Single-Beat, Port Size, or Burst transaction.*
TA#	Transfer Acknowledge	1	I/O TS 24 mA	N1	As an input, when asserted, indicates that a Single-Beat Data transfer successfully completed or that a data-beat in a Burst transfer successfully completed. ***Note:*** *Must be asserted for each data beat in a Burst transaction.* As an output, when asserted, indicates that the data was latched for a Write operation or is valid for a Read operation, therefore terminating the current data beat. When it is the last or only data beat, this also terminates the Data tenure.
TBST#	Transfer Burst	1	I/O TS 24 mA	L26	Indicates that a Burst transfer is in progress.

Table 10-10. 60x Bus Pins (Continued)

Symbol	Signal Name	Total Pins	Pin Type	Pin Number	Function
TEA#	Transfer Error Acknowledge	1	I/O TS 24 mA	C18	When asserted, indicates that a Bus error occurred. TEA# assertion causes DBB# de-assertion/high impedance in the next Clock cycle. Also, terminates the transaction in progress (asserting TA# is unnecessary, as the target device ignores it). TEA# is asserted by a slave on the 60x Bus during Data tenure. The PCI 9610 may assert TEA# as a slave, when enabled, when it cannot complete a Data transfer due to an error condition on the PCI Bus or a Message Queue Output full.
TS#	Transfer Start	1	I/O TS 24 mA	K23	As an input, when asserted, indicates that another device has begun a Bus transaction and that the Address Bus and Transfer Attribute signals are valid for snooping. As an output, when asserted, indicates the PCI 9610 has begun a Bus transaction and the Address Bus and Transfer Attribute signals are valid. It is also an implied Data Bus request when the transfer attributes TT[0:4] indicate that a Data tenure is required for the transaction.
TSIZ[0:3]	60x Bus Transfer Size	4	I/O TS 24 mA	M26, J25, M24, J24	Indicates the transfer byte size in progress. The Motorola MPC603e does not use TSIZ3.
TT[0:4]	60x Bus Transfer Type	5	I/O TS 24 mA	K24, F25, H24, G26, J26	Specifies the type of transfer in progress.
Total		133			

Table 10-11. PLX Extended Burst Mode Pins

Symbol	Signal Name	Total Pins	Pin Type	Pin Number	Function
PLX_DVAL# {PLXDVL#}	PLX Extended Data Valid	1	I/O TS 24 mA	B14	Data is valid during PLX Burst mode. Indicates Extended Burst data is available for reads or as needed for writes. Driven by the Slave.
PLX_MT#	PLX Extended Master Terminate	1	I/O TS 24 mA	M3	Master Terminate Burst. Indicates the PowerQUICC II Master wants to terminate the Extended Burst on the following Data transfer. Driven by the Master.
PLX_TBST# {PLXTBS#}	PLX Extended Transfer Burst	1	I/O TS 24 mA	K25	Transfer start during PLX Burst mode. Indicates the Master wants to transfer data using the PLX Extended Burst mode. Driven by the Master.
Total		3			

Table 10-12. Miscellaneous Pins

Symbol	Signal Name	Total Pins	Pin Type	Pin Number	Function
BIGEND#	Big/Little Endian Override	1	I	G25	When BIGEND# is asserted low, Direct Master and Register accesses are Big Endian, which overrides the configuration bits BIGEND[1:0]. When BIGEND# is de-asserted high, then the BIGEND[1:0] bits determine the Endian for Direct Master and Register accesses.
CCS#	Configuration Chip Select	1	I	U3	Register selection. When a master on the 60x Bus reads or writes internal PCI 9610 registers, CCS# is asserted low during the Address Data Tenure phase. When a 60x Bus Master reads or writes PCI Space Direct Slave write, Direct Slave read, Type 0, Type 1, or PCI I/O, the CCS# input is de-asserted high.
DACK[3:1]# {DCK[3:1]#}	DMA Transfer Acknowledge	3	O 24 mA	N3, R1, T1	DMA Channels 3, 2, and 1 Demand Mode Output acknowledgement.
DREQ[3:1]# {DRQ[3:1]#}	DMA Transfer Request	3	I	R2, T2, R4	DMA Channels 3, 2, and 1 Demand Mode Input acknowledgement.
EOT[3:0]#	End Of Transfer	4	I	P1, P3, R3, U2	DMA Channels 3, 2, 1, and 0 EOT input.
LINTi#	Interrupt Input	1	I	V2	60x Bus Interrupt input.
LINTo#	Interrupt Output	1	O 24 mA	U4	60x Bus Interrupt output.
MDREQ#	Data Request Output to 60x Bus Master	1	O 24 mA	W2	Data Request for a 60x Bus Master. Used when the PCI 9610 is a slave on the 60x Bus in a Direct Master Write operation. When the Direct Master Write FIFO Almost Full status output is set (DMPBAM[10]=1), MDREQ# de-asserts high. Otherwise, MDREQ# is asserted low.
USERi/ DACK0# {USRiDK0#}	User Input or DACK0# Output	1	I/O 24 mA	T3	Multiplexed pin. USERi: General-purpose input that can be read by way of the PCI 9610 Configuration registers. Default PCI 9610 condition for this pin. (Refer to Section 4.2.) DACK0#: DMA Ack Output for DMA Channel 0.
USERo/ DREQ0# {USRoDQ0#}	User Output or DREQ0# Input	1	I/O 24 mA	U1	Multiplexed pin. USERo: General-purpose output controlled from the PCI 9610 Configuration registers. Default PCI 9610 condition for this pin. (Refer to Section 4.2.) DREQ0#: DMA data request input for DMA Channel 0.
Total		17			

10.3 DEBUG INTERFACE

The PCI 9610 provides a JTAG Boundary Scan interface which can be utilized to debug a pin's connectivity to the board.

10.3.1 IEEE 1149.1 Test Access Port (JTAG Debug Port)

The IEEE 1149.1 Test Access Port (TAP), commonly called the JTAG (Joint Test Action Group) debug port, is an architectural standard described in IEEE Standard 1149.1–1990, *IEEE Standard Test Access Port and Boundary-Scan Architecture.* The standard describes a method for accessing internal chip facilities using a four- or five-signal interface.

The JTAG debug port, originally designed to support scan-based board testing, is enhanced to support the attachment of debug tools. The enhancements, which comply with the IEEE 1149.1 specifications for vendor-specific extensions, are compatible with standard JTAG hardware for boundary-scan system testing.

- **JTAG Signals**—JTAG debug port implements the four required JTAG signals, TCK, TMS, TDI, TDO, and the optional TRST# signal.
- **JTAG Clock Requirements**—The TCK signal frequency can range from DC to one-half of the internal chip clock frequency.
- **JTAG Reset Requirements**—JTAG debug port logic is reset at the same time as a system reset. Upon receiving TRST#, the JTAG TAP controller returns to the Test-Logic Reset state.

10.3.2 JTAG Instructions

The JTAG debug port provides the standard **extest, sample/preload,** and **bypass** instructions. Invalid instructions behave as the **bypass** instruction. There are three private instructions.

The following tables list the JTAG instructions and infrared (IR) outputs.

Table 10-13. JTAG Instructions

Instruction	Input Code	Comments
Extest	0000	IEEE 1149.1 standard
Sample/Preload	0100	IEEE 1149.1 standard
Bypass	1111	IEEE 1149.1 standard
JTAG3	0011	Private

Table 10-14. JTAG Infrared Outputs

Instruction	IR Output	Comments
Extest	0001	IEEE 1149.1 standard
Sample/Preload	0101	IEEE 1149.1 standard
Bypass	1101	IEEE 1149.1 standard
JTAG3	0011	Private

10.3.3 JTAG Boundary Scan

Boundary Scan Description Language (BSDL), IEEE 1149.1b-1994, is a supplement to IEEE 1149.1-1990 and IEEE 1149.1a-1993, *IEEE Standard Test Access Port and Boundary-Scan Architecture*. BSDL, a subset of the IEEE 1076-1993 Standard VHSIC Hardware Description Language (VHDL), allows a rigorous description of testability features in components which comply with the standard. It is used by automated test pattern generation tools for package interconnect tests and electronic design automation (EDA) tools for synthesized test logic and verification. BSDL supports robust extensions that can be used for internal test generation and to write software for hardware debug and diagnostics.

The primary components of BSDL include the logical port description, physical pin map, instruction set, and boundary register description.

The logical port description assigns symbolic names to the pins of a chip. Each pin has a logical type of in, out, inout, buffer, or linkage that defines the logical direction of signal flow.

The physical pin map correlates the logical ports of the chip to the physical pins of a specific package. A BSDL description can have several physical pin maps; each map is given a unique name.

Instruction set statements describe the bit patterns that must be shifted into the Instruction Register to place the chip in the various test modes defined by the standard. Instruction set statements also support descriptions of instructions that are unique to the chip.

The boundary register description lists each cell or shift stage of the Boundary Register. Each cell has a unique number; the cell numbered 0 is the closest to the TDO pin and the cell with the highest number is closest to the TDI pin. Each cell contains additional information, including:

- Cell type
- Logical port associated with the cell
- Logical function of the cell
- Safe value
- Control cell number
- Disable value
- Result value

11 ELECTRICAL SPECIFICATIONS

11.1 GENERAL ELECTRICAL SPECIFICATIONS

Table 11-1. Absolute Maximum Ratings

Specification	Maximum Rating
Storage Temperature	-55 to +125 °C
Ambient Temperature with Power Applied	-40 to +85 °C
Supply Voltage to Ground (I/O VDD)	-0.5 to +4.6V
Supply Voltage to Ground (Core VDD)	-0.5 to +3.6V
Supply Voltage to Ground (V_{IO})	-0.5 to +6.5V
Input Voltage (VIN)	V_{SS} -0.5 to 6.5V
Output Voltage (VOUT)	V_{SS} -0.5V to VDD +0.5
Maximum Package Power Dissipation	2.0W

Table 11-2. Operating Ranges

Ambient Temperature	Supply Voltage (I/O VDD)	Supply Voltage (Core VDD)	Input Voltage (VIN)	
			Min	Max
-40 to +85 °C	3.0 to 3.6V	2.3 to 2.7V	V_{SS}	V_{IO}

Table 11-3. Capacitance (Sample Tested Only)

Parameter	Test Conditions	Pin Type	Value		Units
			Typical	Maximum	
CIN	VIN = 0V	Input	4	6	pF
COUT	VOUT = 0V	Output	6	10	pF

The following table lists the package thermal resistance in °C/W (Θ_{j-a}).

Table 11-4. Package Thermal Resistance

Linear Air Flow			
0m/s	1m/s	2m/s	3m/s
30	22	19	17

Table 11-5. Electrical Characteristics over Operating Range

Parameter	Description	Test Conditions		Min	Max	Units
VOH	Output High Voltage	VDD = Min VIN = VIH or VIL	IOH = -12.0 mA	2.4	—	V
VOL	Output Low Voltage		IOL = 12 mA	—	0.4	V
VOH	Output High Voltage	VDD = Min VIN = VIH or VIL	IOH = -24.0 mA	2.4	—	V
VOL	Output Low Voltage		IOL = 24 mA	—	0.4	V
VIH	Input High Level	—	—	2.0	5.5	V
VIL	Input Low Level	—	—	-0.5	0.8	V
VOH3	PCI 3.3V Output High Voltage	VDD = Min VIN = VIH or VIL	IOH = -500 µA	0.9 VDD	—	V
VOL3	PCI 3.3V Output Low Voltage		IOL = 1500 µA	—	0.1 VDD	V
VIH3	PCI 3.3V Input High Level	—	—	0.5 VDD	VDD +0.5	V
VIL3	PCI 3.3V Input Low Level	—	—	-0.5	0.3 VDD	V
IIL	Input Leakage Current	VSS ≤ VIN ≤ VDD, VDD = Max		-10	+10	µA
ILPC[1]	DC Current Per Pin during Precharge	VP = 0.8 to 1.2V		—	1.0	mA

Table 11-5. Electrical Characteristics over Operating Range (Continued)

Parameter	Description	Test Conditions	Min	Max	Units
IOZ	Three-State Output Leakage Current	VDD = Max	-10	+10	µA
ICC	Power Supply Current[1]	I/O VDD = 3.6V Core VDD = 2.7V PCLK = 66 MHz, LCLK = 66 MHz 160 outputs switching simultaneously	—	200	mA
ICCL ICCH ICCZ	Quiescent Power Supply Current	VCC = Max VIN = GND or VCC	—	50	µA

1. ILPC is the DC current flowing from VDD to Ground during precharge, as both PMOS and NMOS devices remain on during precharge. It is not the leakage current flowing into or out of the pin under precharge. (Maximum value based upon 160 simultaneously switching outputs.)

11.2 60X BUS AC TIMING REQUIREMENTS

- Maximum LCLK = 66 MHz
- Output load = 50 pf, except as noted

Definitions:

- **Tsu**—Setup time. The time that an input signal is stable before the rising edge of LCLK.
- **Tval**—Output valid (clock-to-out). The time after the rising edge of LCLK until the output is stable.

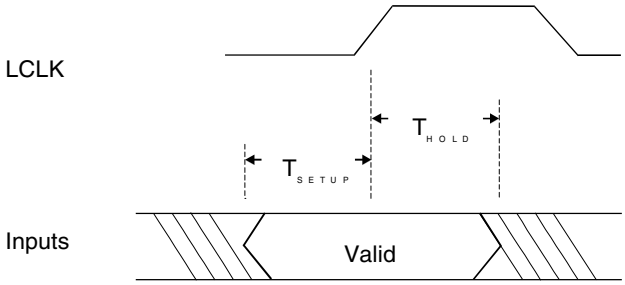


Figure 11-1. PCI 9610 Processor Bus Input Setup and Hold Waveform

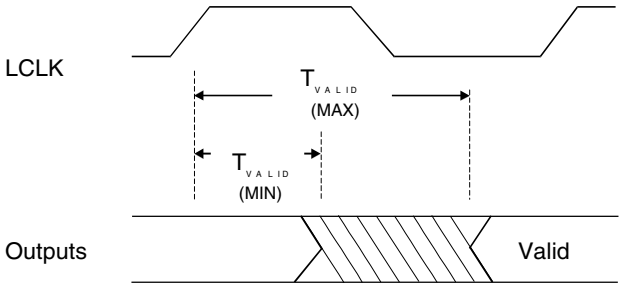


Figure 11-2. PCI 9610 Processor Bus Output Delay

Table 11-6. 60x Bus AC Timing Requirements

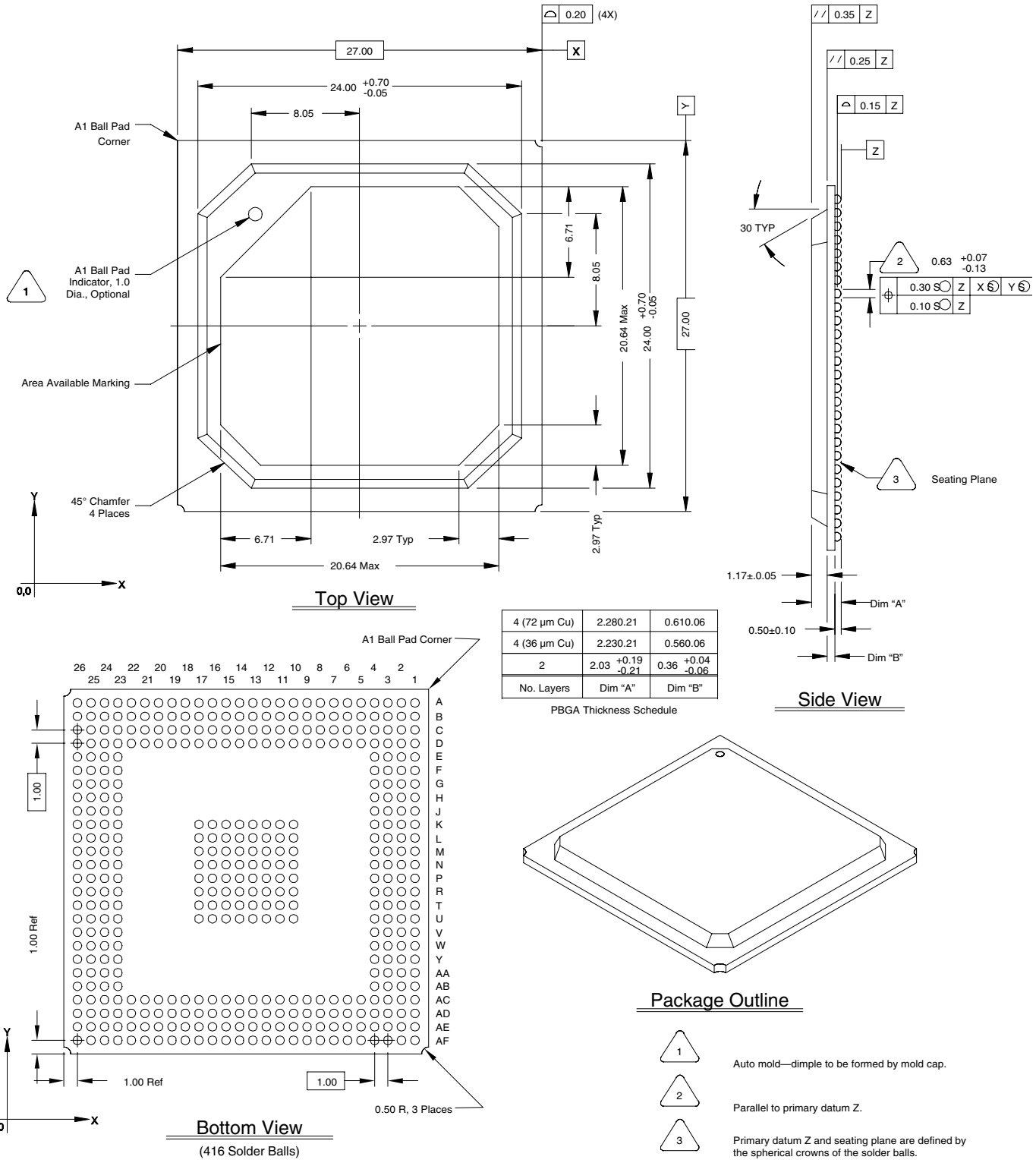
Signal Name	Type	Tval (ns) Minimum	Tsu (ns) Minimum
LD[0:63], LDP[0:7], PLX_DVAL#, PSDVAL#, TA#	Bi-Directional	6	3
AACK#, ARTRY#, PLX_MT#, PLX_TBST#, TEA#	Bi-Directional	8	3
BR#	Out	8	N/A
BG#, DBG#	In	N/A	3
ABB#, DBB#, GBL#, LA[0:31], LAP[0:3], TBST#, TS#, TSIZ[0:3], TT[0:4]	Bi-Directional	8	3
APE#, MCP#/DPE#	Open Drain	8	N/A
64EN#, BD_SEL#, BIGEND#, CCS#, CPCISW, DREQ[3:1]#, EOT[3:0]#, HOSTEN#, LINTi#, PMEREQ#	In	N/A	4.5
EEDI/EEDO, USERi/DACK0#, USERo/DREQ0#	Bi-Directional	9	4.5
DACK[3:1]#, EECS, EESK, LINTo#, LRESET#, MDREQ#	Out	9	N/A
ENUM#, LEDon#, PME#	Open Drain	9	N/A

Note: *All outputs listed in Table 11-6 are rated based on a 50 pf load, except for the following:*

- *10 pf load—ENUM#, PME#*
- *20 pf load—DACK[3:1]#, EECS, EEDI/EEDO, EESK, EOT[3:0]#, LEDon#, USERi/DACK0#*

12 PHYSICAL SPECS

12.1 MECHANICAL LAYOUT

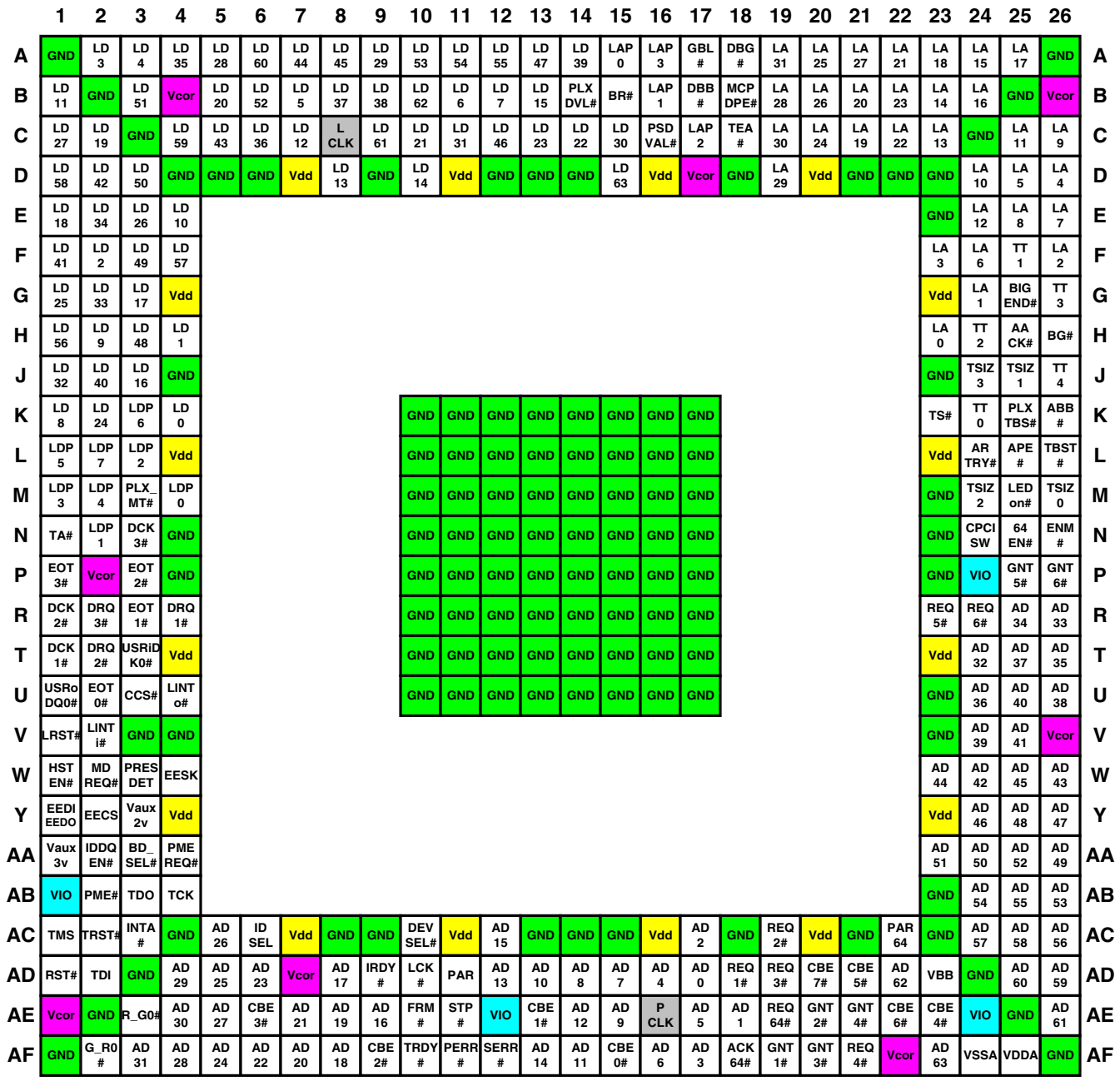


No. Layers	Dim "A"	Dim "B"
4 (72 µm Cu)	2.280.21	0.610.06
4 (36 µm Cu)	2.230.21	0.560.06
2	2.03 +0.19 -0.21	0.36 +0.04 -0.06

PBGA Thickness Schedule

Figure 12-1. Mechanical Dimensions and Package Outline

12.2 BALL GRID ASSIGNMENTS



	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	
A	GND	LD 3	LD 4	LD 35	LD 28	LD 60	LD 44	LD 45	LD 29	LD 53	LD 54	LD 55	LD 47	LD 39	LAP 0	LAP 3	GBL #	DBG #	LA 31	LA 25	LA 27	LA 21	LA 18	LA 15	LA 17	GND	A
B	LD 11	GND	LD 51	Vcor	LD 20	LD 52	LD 5	LD 37	LD 38	LD 62	LD 6	LD 7	LD 15	PLX DVL#	BR#	LAP 1	DBB #	MCP DPE#	LA 28	LA 26	LA 20	LA 23	LA 14	LA 16	GND	Vcor	B
C	LD 27	LD 19	GND	LD 59	LD 43	LD 36	LD 12	L CLK	LD 61	LD 21	LD 31	LD 46	LD 23	LD 22	LD 30	PSD VAL#	LAP 2	TEA #	LA 30	LA 24	LA 19	LA 22	LA 13	GND	LA 11	LA 9	C
D	LD 58	LD 42	LD 50	GND	GND	GND	Vdd	LD 13	GND	LD 14	Vdd	GND	GND	GND	LD 63	Vdd	Vcor	GND	LA 29	Vdd	GND	GND	GND	LA 10	LA 5	LA 4	D
E	LD 18	LD 34	LD 26	LD 10																			GND	LA 12	LA 8	LA 7	E
F	LD 41	LD 2	LD 49	LD 57																			LA 3	LA 6	TT 1	LA 2	F
G	LD 25	LD 33	LD 17	Vdd																			Vdd	LA 1	BIG END#	TT 3	G
H	LD 56	LD 9	LD 48	LD 1																			LA 0	TT 2	AA CK#	BG#	H
J	LD 32	LD 40	LD 16	GND																			GND	TSIZ 3	TSIZ 1	TT 4	J
K	LD 8	LD 24	LDP 6	LD 0						GND	GND	GND	GND	GND	GND	GND	GND						TS#	TT 0	PLX TBS#	ABB #	K
L	LDP 5	LDP 7	LDP 2	Vdd						GND	GND	GND	GND	GND	GND	GND	GND						Vdd	AR TRY#	APE #	TBST #	L
M	LDP 3	LDP 4	PLX_ MT#	LDP 0						GND	GND	GND	GND	GND	GND	GND	GND						GND	TSIZ 2	LED on#	TSIZ 0	M
N	TA#	LDP 1	DCK 3#	GND						GND	GND	GND	GND	GND	GND	GND	GND						GND	CPCI SW	64 EN#	ENM #	N
P	EOT 3#	Vcor	EOT 2#	GND						GND	GND	GND	GND	GND	GND	GND	GND						GND	VIO	GNT 5#	GNT 6#	P
R	DCK 2#	DRQ 3#	EOT 1#	DRQ 1#						GND	GND	GND	GND	GND	GND	GND	GND						REQ 5#	REQ 6#	AD 34	AD 33	R
T	DCK 1#	DRQ 2#	USRiD K0#	Vdd						GND	GND	GND	GND	GND	GND	GND	GND						Vdd	AD 32	AD 37	AD 35	T
U	USRo DQ0#	EOT 0#	CCS#	LINT o#						GND	GND	GND	GND	GND	GND	GND	GND						GND	AD 36	AD 40	AD 38	U
V	LRST#	LINT i#	GND	GND																			GND	AD 39	AD 41	Vcor	V
W	HST EN#	MD REQ#	PRES DET	EESK																			AD 44	AD 42	AD 45	AD 43	W
Y	EEDI EEDO	EECS	Vaux 2v	Vdd																			Vdd	AD 46	AD 48	AD 47	Y
AA	Vaux 3v	IDDQ EN#	BD_ SEL#	PME REQ#																			AD 51	AD 50	AD 52	AD 49	AA
AB	VIO	PME#	TDO	TCK																			GND	AD 54	AD 55	AD 53	AB
AC	TMS	TRST#	INTA #	GND	AD 26	ID SEL	Vdd	GND	GND	DEV SEL#	Vdd	AD 15	GND	GND	GND	Vdd	AD 2	GND	REQ 2#	Vdd	GND	PAR 64	GND	AD 57	AD 58	AD 56	AC
AD	RST#	TDI	GND	AD 29	AD 25	AD 23	Vcor	AD 17	IRDY #	LCK #	PAR	AD 13	AD 10	AD 8	AD 7	AD 4	AD 0	REQ 1#	REQ 3#	CBE 7#	CBE 5#	AD 62	VBB	GND	AD 60	AD 59	AD
AE	Vcor	GND	R_G0#	AD 30	AD 27	CBE 3#	AD 21	AD 19	AD 16	FRM #	STP #	VIO	CBE 1#	AD 12	AD 9	P CLK	AD 5	AD 1	REQ 64#	GNT 2#	GNT 4#	CBE 6#	CBE 4#	VIO	GND	AD 61	AE
AF	GND	G_R0 #	AD 31	AD 28	AD 24	AD 22	AD 20	AD 18	CBE 2#	TRDY #	PERR #	SERR #	AD 14	AD 11	CBE 0#	AD 6	AD 3	ACK 64#	GNT 1#	GNT 3#	REQ 4#	Vcor	AD 63	VSSA	VDDA	GND	AF
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	

Figure 12-2. Ball Grid Assignments

Notes: *Vdd = 3.3 volts, nominal, Ring Voltage (16 places)*
Vcor = 2.5 volts, nominal, Core Voltage (8 places)

A GENERAL INFORMATION

A.1 ORDERING INSTRUCTIONS

The PCI 9610 is a 64-bit, 66 MHz PCI I/O Accelerator featuring advanced PLX proprietary Data Pipe Architecture technology, which includes four DMA engines, programmable Direct Slave and Direct Master Data Transfer modes, and PCI messaging functions. The PCI 9610 offers 3.3V, 5V tolerant PCI and 60x Bus signaling, and supports Universal PCI adapter designs, 3.3V core, low-powered CMOS offered in a 416-pin (ball) fine-pitch PBGA. The device is designed to operate at Industrial Temperature range.

Table A-1. Available Package

Package	Ordering Part Number
416-pin fine-pitch PBGA	PCI 9610-AA66BI

A.2 UNITED STATES AND INTERNATIONAL REPRESENTATIVES, AND DISTRIBUTORS

A list of PLX Technology, Inc., representatives and distributors can be found at http://www.plxtech.com.

A.3 TECHNICAL SUPPORT

PLX Technology, Inc., technical support information is listed at http://www.plxtech.com; or call 408 774-9060 or 800 759-3735.

Index

A

D

E

F

G

H

I

Index

N

O

P

Q

R

S

T