# PCI 6254 (HB6)
# Dual Mode
# Universal PCI-to-PCI Bridge
# Data Book

# PCI 6254 (HB6)
# Dual Mode
# Universal PCI-to-PCI Bridge
# Data Book

Version 2.1

December 2003


**Website:** http://www.plxtech.com
**Technical Support:** http://www.plxtech.com/support
**Phone:** 408 774-9060
800 759-3735
**Fax:** 408 774-2169

PLX Part Number: PCI 6254-AB66BC; Former HiNT Part Number: HB6

Order Number: 6254-SIL-DB-P1-2.0

Printed in the USA, May 2003

# PCI 6254 Dual-Mode Universal PCI-to-PCI Bridge

## Adaptive High Performance Asynchronous 66MHz 64-bit PCI-to-PCI Bridge for Servers, Storage, Telecommunication, Networking and Embedded Applications

PLX's latest PCI 6254 64-bit PCI-to-PCI bridge is designed for high performance, high availability applications in bus expansions, programmable data transfer rate control, frequency conversions from slower PCI to faster PCI or from faster PCI to slower PCI buses, address remapping, high availability Hot Swap enabling and universal system-to-system bridging. PCI 6254 has sophisticated buffer management and buffer configuration options designed to provide customizable performance optimization.

- PCI Local Bus Specification Rev 2.3 support
- High speed PCI buffer supports 3.3V signaling with 5V input signal tolerance
- CPCI Hot Swap Specification PICMG 2.1 R2.0 with PI = 1 support
- Device Hiding support eliminates mid-transaction extraction problems
- Programmable 32 bit to 64 bit access conversion.
- Programmable Address Translation to Secondary Bus
- Flow-Through 0 wait state burst up to 4K bytes for optimal large volume data transfer
- Supports up to 4 simultaneous posted write transactions and 4 simultaneous Delayed transactions in each direction
- Provides 1K Bytes of buffering
  - 256 byte upstream posted write buffer
  - 256 byte downstream posted write buffer
  - 256 byte upstream read data buffer
  - 256 byte downstream read data buffer
- Programmable prefetch amount of up to 256 bytes for maximum read performance optimization
- Supports out of order delayed transactions
- Support Secondary Port PCI Private Memory Space
- Option to eliminates possible dead lock on PCI-VME bridges before or behind PCI 6254
- Serial EEPROM loadable and programmable PCI READ ONLY Register configurations.
- External arbiter or programmable arbitration for 9 bus masters on secondary interface support
- 10 Secondary clock outputs with pin controlled enable and individual maskable control
- PCI Mobile Design Guide and Power Management D3 Cold Wakeup capable with PME# support
- 16 GPIO pins with output control and 8 are with power up status latch capabilities
- Enhanced address decoding
  - Support 32-bit I/O address range
  - 32-bit memory-mapped I/O address range
  - ISA aware mode for legacy support in the first 64KB of I/O address range
  - VGA addressing and palette snooping support
- Provides an IEEE standard 1149.1 JTAG interface for boundary scan test
- Asynchronous design supports standard 66Mhz to 33MHz and faster secondary port speed such as 33Mhz to 66MHz conversion
- PCI 6254 package ball layout is super set of PLX PCI 6154 and Intel 21154 when operating in Transparent Mode
- Industry standard 31mm x 31mm 365-ball PBGA package

## PCI 6254 Non-Transparent and Universal Mode Features

- Programmable Transparent, Non-Transparent or Universal Mode operation
- Jumper less switching between System and Peripheral Slot applications in CPCI
- Programmable Primary or Secondary Port System boot up priority.
- Semaphore backed Cross-bridge Configuration Space access
- Powerful multi-source (Direct encoded, door bell, PCI Reset, external pin) programmable interrupts
- Message Interrupt Support
- Optional power up 16M memory space claim to avoid Initially Retry or Initially Not Respond requirement
- Behave as a Memory mapped PCI device
- Primary and Secondary Port controllable GPIOs
- Power-Good input
- Available Primary and Secondary Power Status inputs for port power detection
- Independent Primary and Secondary Port Reset inputs
- Configurable Primary and Secondary Reset Outputs
- Sticky user registers immune to PCI resets
- Supporting up to 9 secondary PCI master devices

# HISTORY

| Rev | Date | Description | Eng Chk | Mkt Chk |
|---|---|---|---|---|
| Rev 0.1 | 7/7/01 | First Marketing release of PCI 6254 data book | | |
| Rev 0.2 | 7/26/01 | Second Marketing release of PCI 6254 data book<br>- corrected some typing mistakes. | | |
| Rev 0.3 | 8/27/01 | Updated specifications<br>• CLKRUN pins can be left unconnected in Transparent Mode if CLKRUN mechanism is not enabled by software<br>• Remove Private memory descriptions from Non-Transparent Mode section. Private memory is only for Transparent Mode.<br>• Corrected typing mistakes. | | |
| Rev 0.4 | 9/27/01 | Added pin location list and corrected typing mistakes<br>• Add pin list sorted by location and name<br>• Clarify GPIO and clock control registers descriptions<br>• Clarified Reset description in the Reset Chapter<br>• Revision Code changed to 04h | | |
| Rev 0.9 | 10/11/01 | Pre-production release<br>• Added power management D3-D0 reset description<br>• More detailed EERPOM option description<br>• Add new chapter about PCI 6254 usage | | |
| Rev 0.91 | 11/5/01 | Corrected EEPROM Clock register description<br>Corrected Table references | | |
| Rev 0.92 | 12/4/01 | • Corrections in Reset Chapter: PWRGD does NOT cause P_RSTOUT# and S_RSTOUT# to go active and does NOT reset the entire chip; Software chip reset does NOT cause IO signals to go three-state and causes EEPROM load only in Non-Transparent Mode; S_RSTIN# input is NOT used in Transparent Mode; S_CLKSTB low will NOT cause Secondary port internal reset in Non-Transparent Mode; PCI 6254 needs 512 clock to initialize the bridge functions after any reset. | | |
| Rev 0.93 | 1/8/02 | • Added description about L_STATand EJECT. | | |
| Rev 1.0 | 1/18/02 | • Major modification to the RESET chapter about Non-Transparent Mode reset mechanisms. The following are major the changes:<br>- Non-Transparent Mode P_RSTIN# will also reset secondary port registers at 0-3fh.<br>- Non-Transparent Mode S_RSTIN# only resets secondary port state machine, NOT registers.<br>- PWRGD initiated EEPROM autoload in Non-Transparent Mode can only be effective if the PWRGD rising edge is aligned with or come after P_RSTIN# rising edge. EEPROM chapter is also updated to describe the requirement.<br>- Corrected Description on Power Management Initiated Reset which will NOT cause P_RSTOUT# and S_RSTOUT#.<br>• Updated specification on Register D8h, bit 4. The correct READ back data is inverted while the Write data is not. | | |

| Rev | Date | Description | Eng Chk | Mkt Chk |
|---|---|---|---|---|
| Rev 1.1 | 3/7/02 | • Added more descriptions about BAR masks in Non-Transparent mode.<br>• Updated reset pin state descriptions, added Power Up and Reset pin state table and added reset to first cycle latency description section<br>• In Transparent Mode, removed Private Device support and supports only Private Memory. Signal name is changed from PRV_DEV to PRV_MEM.<br>• EEPROM location 2Ch bit 1 and 3 definitions were swapped and are now corrected. EEPROM control registers definitions for EEPROM clock speed is corrected.<br>• Corrected minor typing mistakes | | |
| Rev 1.2 | 6/3/02 | • Remove Transparent Mode Private Memory Enable Pin Input feature.<br>• Added description about active XB_MEM window setup in Non-Transparent Mode will be decoded as additional private memory in Transparent mode if software is used to switch the PCI 6254 from Non-Transparent mode to Transparent Mode.<br>• Added Primary port access only for register E6h errata description for Hot Swap control in Universal Non-Transparent Mode.<br>• Added errata description for Universal Non-Transparent Mode EEPROM loaded Device ID.<br>• Added more detailed description for Force 32/64 bit Conversion control option.<br>• Added Force 64 bit control features<br>• Added JTAG input signals requirement for external pull-up or pull-low resistors.<br>• Updated Transparent Mode Translation Register use description. | | |
| Rev 1.3 | 9/17/02 | • Added Rev AB features descriptions and highlights Rev AA only features.<br>• Added more description for Register D9h bit 3.<br>• Correct description mistakes about Register 96h and 97h.<br>• Added new feature description for XB_MEM control<br>• Added cautions for use of Flow Through Optimizations<br>• Added Power up and reset state table | | |
| Rev 1.31 | 10/2/02 | • Corrected S_RSTIN# pin description. It is only used for Non-Transparent Mode<br>• GPIO 15:8 internal pull down claim is removed<br>• Clarified Translation address registers and address translation mechanism. | | |
| Rev 2.0 | 5/23/03 | This release reflects PLX part numbering.<br>• Updated Register DEh, bits 15-11<br>• Added three notes to table in Section 14.5, Frequency Division Options | | |
| Rev 2.1 | 12/10/03 | Remove reference to heatsink in Section 26 | | |

# CONTENTS

# 1 Register Index

When looking up registers, please also check registers below preceded with "Primary" or "Secondary".

# 2 Ordering Information

| Part Number | Rev. | Description |
|---|---|---|
| PCI 6254 | AA | Dual Mode Universal PCI-to-PCI bridge |
| PCI 6254 | AB | Dual Mode Universal PCI-to-PCI bridge |

# 3 Using the PCI 6254

## *3.1 Transparent Mode Application*

Since the PCI 6254 Primary and Secondary ports are asynchronous to each other, the two independent systems can run at different frequencies. It is possible to run the Secondary bus faster than the Primary bus.

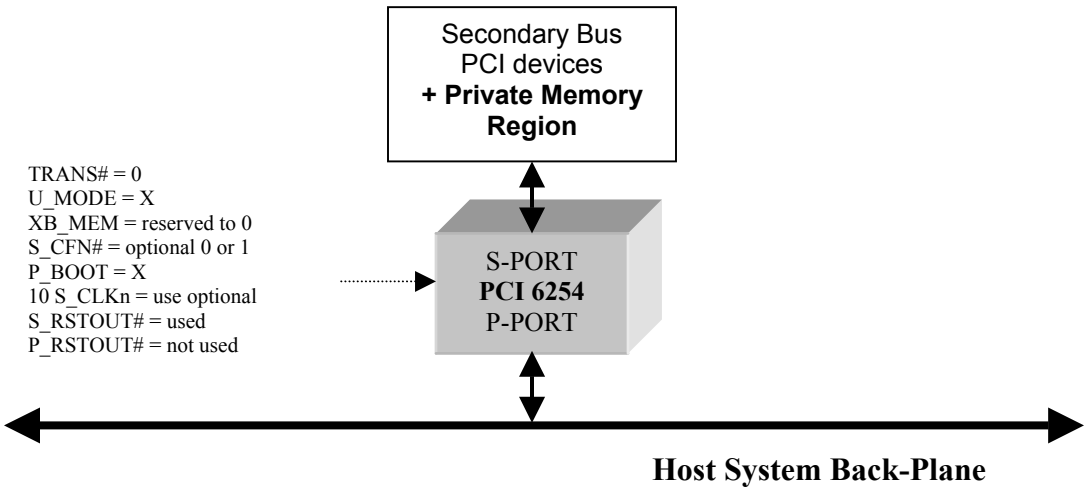
PCI 6254 has powerful programmable buffer control, which can be used to regulate data throughput for multiple PCI masters on the secondary port. The data prefetch size can be programmed up to 256 bytes.

In Transparent Mode, the host system PCI bus is connected to the PCI 6254 Primary port. The Secondary PCI port can use either a custom designed external arbiter or the PCI 6254 internal arbiter. Designers can either use custom designed clock generations, or the PCI 6254 S_CLK[9:0] outputs derived out of the Primary port PCI clock input or an external oscillator, to provide clocks to secondary PCI devices and the PCI 6254 S_CLKIN input.

Primary Port and Secondary port have independent PCI reset inputs. S_RSTIN# is feed-back from the S_RSTOUT#.

In Transparent application, XB_MEM must be set to “0”. Only software can program special memory range registers to reserve a private memory region for Secondary port devices use only. PCI 6254 will not respond to any access to this private memory region by any Secondary PCI masters or Primary PCI masters.

The basic design idea is optimized for the following:



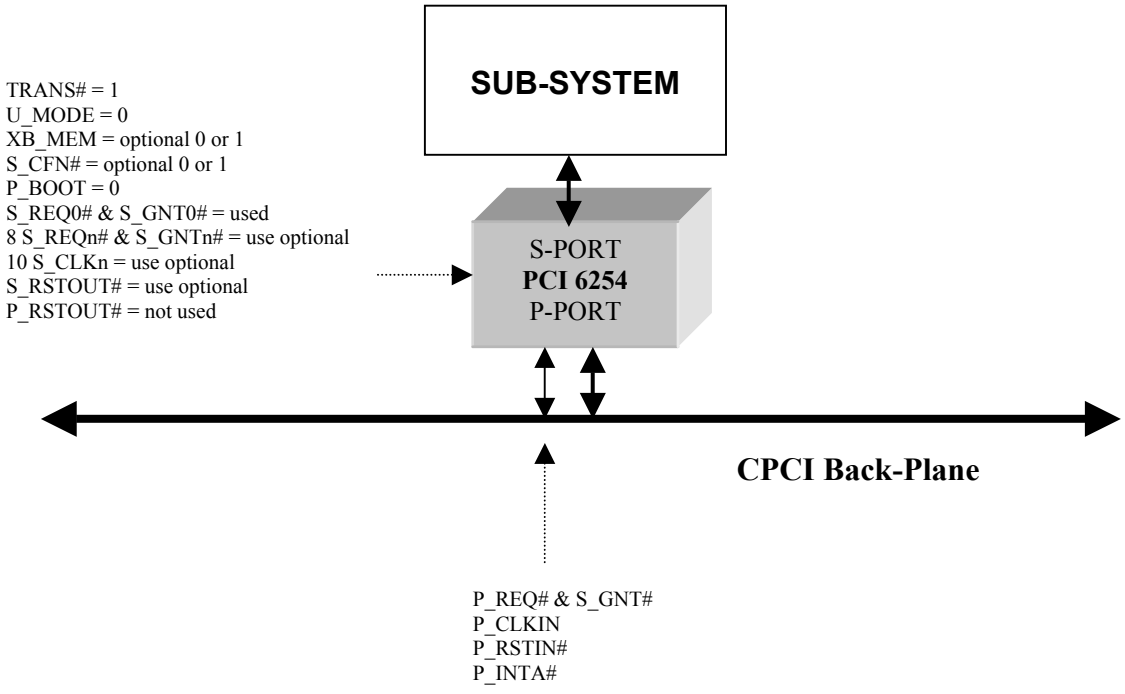
## 3.2 Standard Non–Transparent Application

The PCI 6254 Non-Transparent Mode acts as a memory mapped PCI device on a PCI back-plane. The PCI 6254 Primary side is used to connect to the PCI back-plane, like that of the Transparent mode applications. The intelligent subsystem is connected to the Secondary port  The subsystem can use either an external arbiter or the PCI 6254 internal arbiter. Subsystem clock generation is generally achieved using a clock synthesizer to provide CPU clocks, subsystem PCI clocks to subsystem PCI devices and the PCI 6254 S_CLKIN input. However it can also the PCI 6254 S_CLK[9:0] outputs derived out of the Primary port PCI clock input or an external oscillator

The P_BOOT pin should be programmable to “0” such that the Subsystem at Secondary port has higher boot priority. The subsystem must either setup the BAR registers first or the XB_MEM option must be active for the Primary port host to be able to complete its system initialization sequence. The use of XB_MEM option forces PCI 6254 to declare a fixed 16M memory window for cross-bridge communication at power up. If necessary, this window size can be changed by EEPROM or software after power up.

Primary Port and Secondary port have independent PCI reset inputs. Designers can either use custom designed Reset or the PCI 6254 S_RSTOUT# for the Secondary port and subsystem reset.

The basic design idea is optimized for the following:



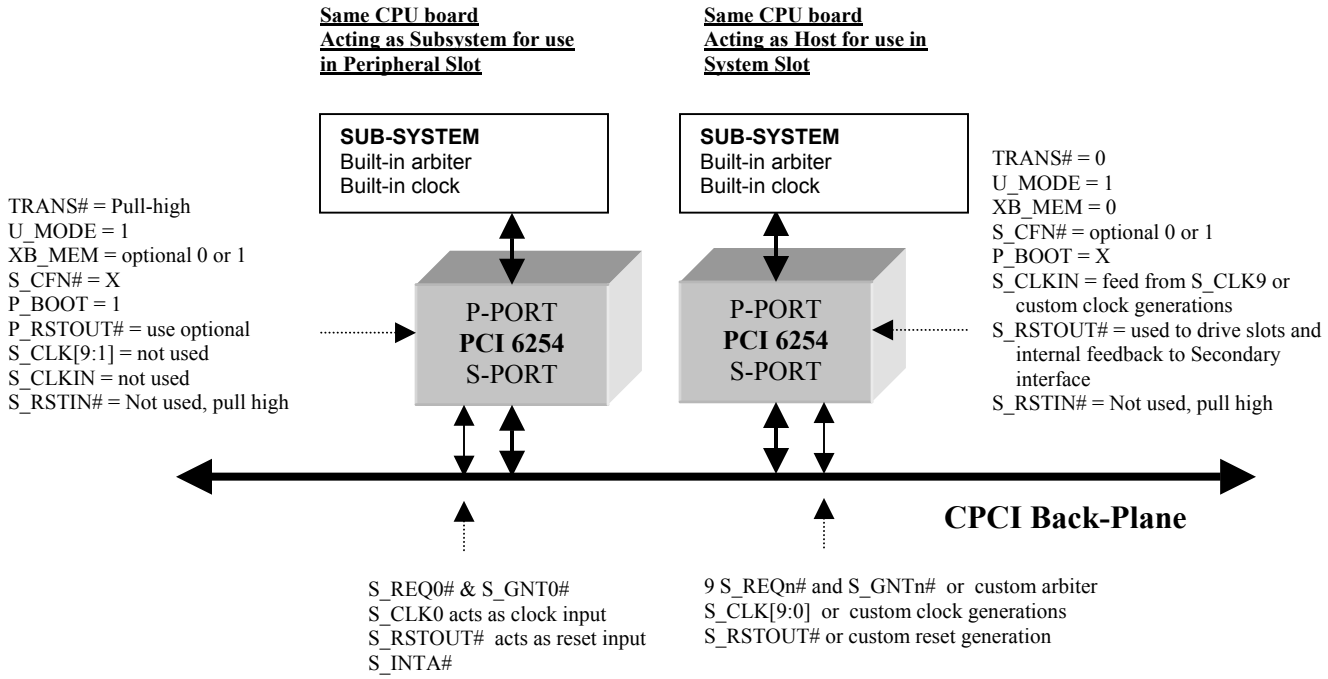
## 3.3 Universal Bridging Application

PCI 6254 is designed such that it allows the design of an intelligent subsystem to act as a host or as a memory mapped device by setting the U_MODE (Universal Mode) pin to “1”.

When acting as a host, the subsystem uses the PCI 6254 Universal Transparent Mode. The subsystem PCI bus is connected to the PCI 6254 Primary port. The subsystem uses either an external arbiter or an arbiter that is built-in to the North Bridge for subsystem PCI bus support. As for the back-plane PCI arbiter, either a custom designed arbiter or the PCI 6254 internal arbiter can be used. Subsystem clock generation is generally achieved using a clock synthesizer to provide CPU clocks, Subsystem PCI clocks to subsystem PCI devices and the PCI 6254 P_CLKIN input. Designers can either use custom designed clock outputs, or the PCI 6254 S_CLK[9:0] outputs derived out of the Primary port PCI clock input or an external oscillator, to drive the PCI back-plane.

When acting as an intelligent subsystem behaving as a memory mapped PCI device on the back-plane PCI bus, the subsystem uses the PCI 6254 Universal Non-Transparent Mode. The PCI 6254 External Arbiter Mode is selected so that the S_REQ0# and S_GNT0# act as the PCI_REQ# and PCI_GNT# respectively for direct connection to back-plane or customer arbiter interface. The P_BOOT pin should be connected to 1 indicating that the Primary port has boot priority. The subsystem at the Primary port must either setup the BAR registers first or the XB_MEM option must be active for the host from the Secondary port to be able to complete its system initialization sequence. The use of XB_MEM option forces PCI 6254 to declare a fixed 16 M memory window for cross-bridge communication at power up. If necessary, this window size can be changed by EEPROM or software after power up.

The basic design idea is optimized for the following assuming the same CPU board behaving differently:



### 3.3.1 Universal Mode CLK, RST#, REQ#, GNT# and SYSEN# Signal connections

The PCI 6254 allows a jumper less automatic switch between CPCI System Slot or Peripheral Slot applications. In Universal Mode, the PCI 6254 switches between System and Peripheral Mode using the TRANS# input pin. The TRANS# pin is designed for direct connection to the CPCI SYSEN# pin.

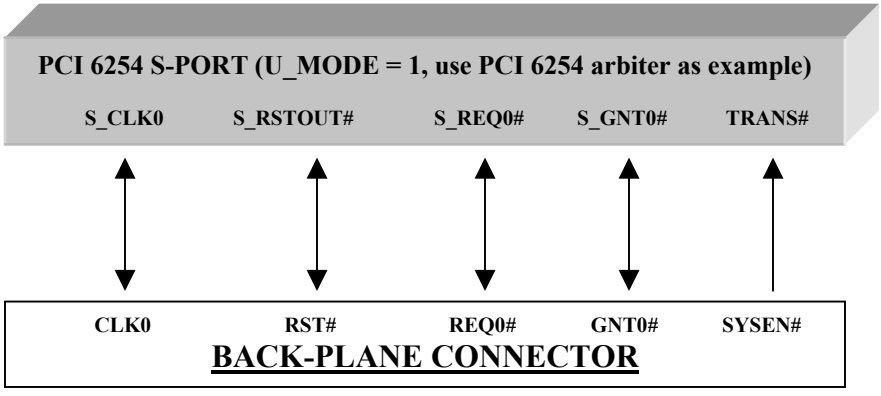
When TRANS# is "0", the PCI 6254 switches to the Universal Transparent Mode in which the PCI 6254 drives out S_RSTOUT# to the back-plane as well as enables the use of the S_RSTOUT# internal feedback for Secondary reset. S_RSTIN# input is internally "AND" with the S_RSTOUT# feedback and should be tied to "1" or can be fed with custom reset input. Secondary port logic uses the S_CLKIN. Designer should use, for example, S_CLK9 to feed the S_CLKIN with a clock trace length that matches the back-plane clock traces length. If custom arbiter is not used, S_REQ0# and S_GNT0# can be directly connected to the back-plane.

When TRANS# is "1", the PCI 6254 switches to the Universal Non-Transparent Mode in which the PCI 6254 three-states the S_RSTOUT# to allow back-plane RST# to drive the S_RSTOUT# pin which acts as Secondary Reset input. S_RSTIN# input is internally "AND" with the S_RSTOUT# feedback and should be tied to "1" or can be fed with custom reset input. The S_CLKIN pin input is ignored inside the PCI 6254 and the Secondary port logic uses the S_CLK0 internal feedback as Secondary Clock Input. S_CFN# is "Don't Care" and if custom arbiter is not used, S_REQ0# and S_GNT0# can be directly connected to the back-plane.

**Hot swap pin ENUM#** must be handled with custom logic for universal applications.

The following is an example of Universal Mode connections (U_MODE is set to "1"):

| | PIN NAME | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Description** | **TRANS#** connect to CPCI **SYSEN#** pin | **U_MODE** | **S_CFN#** | **S_CLK0** | **S_CLK IN** | **S_CLK9** | **S_RST IN#** | **S_RST OUT#** | **S_REQ0#** | **S_GNT0#** |
| **System Slot application using PCI 6254 arbiter** | **0** use US_CLKIN as Secondary port input clock | **1** | **0** | **OUTPUT** | **INPUT e.g. from S_CLK9** | **OUTPUT** | **Not used** | **OUTPUT** | **REQ0# INPUT** | **GNT0# OUTPUT** |
| **System Slot application using external arbiter** | **0** use US_CLKIN as Secondary port input clock | **1** | **1** | **OUTPUT** | **INPUT e.g. from S_CLK9** | **OUTPUT** | **Not used** | **OUTPUT** | **Custom Arbiter GNT# INPUT** | **Custom Arbiter REQ# OUTPUT** |
| **Peripheral Slot application using PCI 6254 Universal Mode** | **1** use S_CLKIN as Secondary port input clock | **1** | **X** | **Used as Secondary Clock INPUT** | Ignored in PCI 6254 NOT USED | OUTPUT NOT USED | **Not used** | **Used as Secondary Reset INPUT** | **REQ# OUTPUT** | **GNT# INPUT** |



## 3.4 Symmetrical Non–Transparent Application

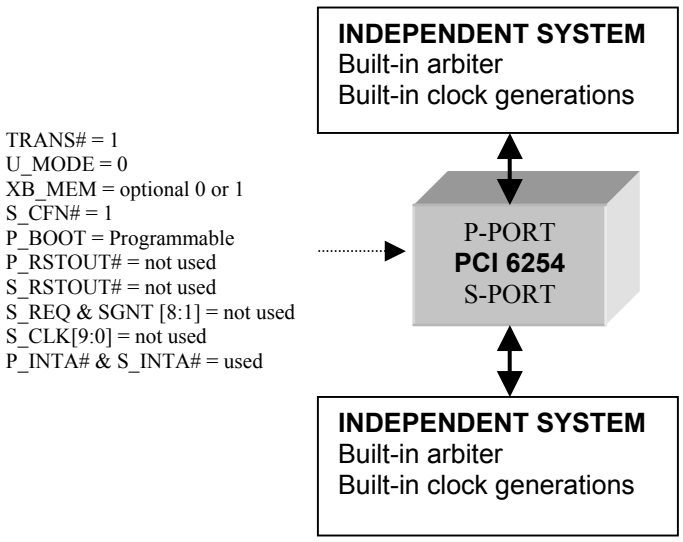
PCI 6254 is designed such that it allows the bridging of two totally independent systems. The only necessary option is to decide which host has higher boot priority.

The PCI 6254 External Arbiter Mode is selected so that the S_REQ0# and S_GNT0# act as the PCI GNT# and PCI REQ# respectively for handshaking with one of the host. All PCI signals should be connected to their respective host PCI signals. The boot priority is programmable. The higher priority boot system must either setup the BAR registers first or the XB_MEM option must be active for the lower boot priority system to be able to complete its system initialization sequence. The use of XB_MEM option forces PCI 6254 to declare a fixed 16M memory window for cross-bridge communication at power up. If necessary, this window size can be changed by EEPROM or software after power up.

The independent system should use either an external arbiter or an arbiter that is built-in to the North Bridge for its PCI bus use. System clock generation is generally achieved using a clock synthesizer to provide CPU clocks, PCI clocks to system PCI devices and the PCI 6254 PCI input.

Since the PCI 6254 Primary and Secondary ports are asynchronous to each other, the two independent systems can run at different frequencies.

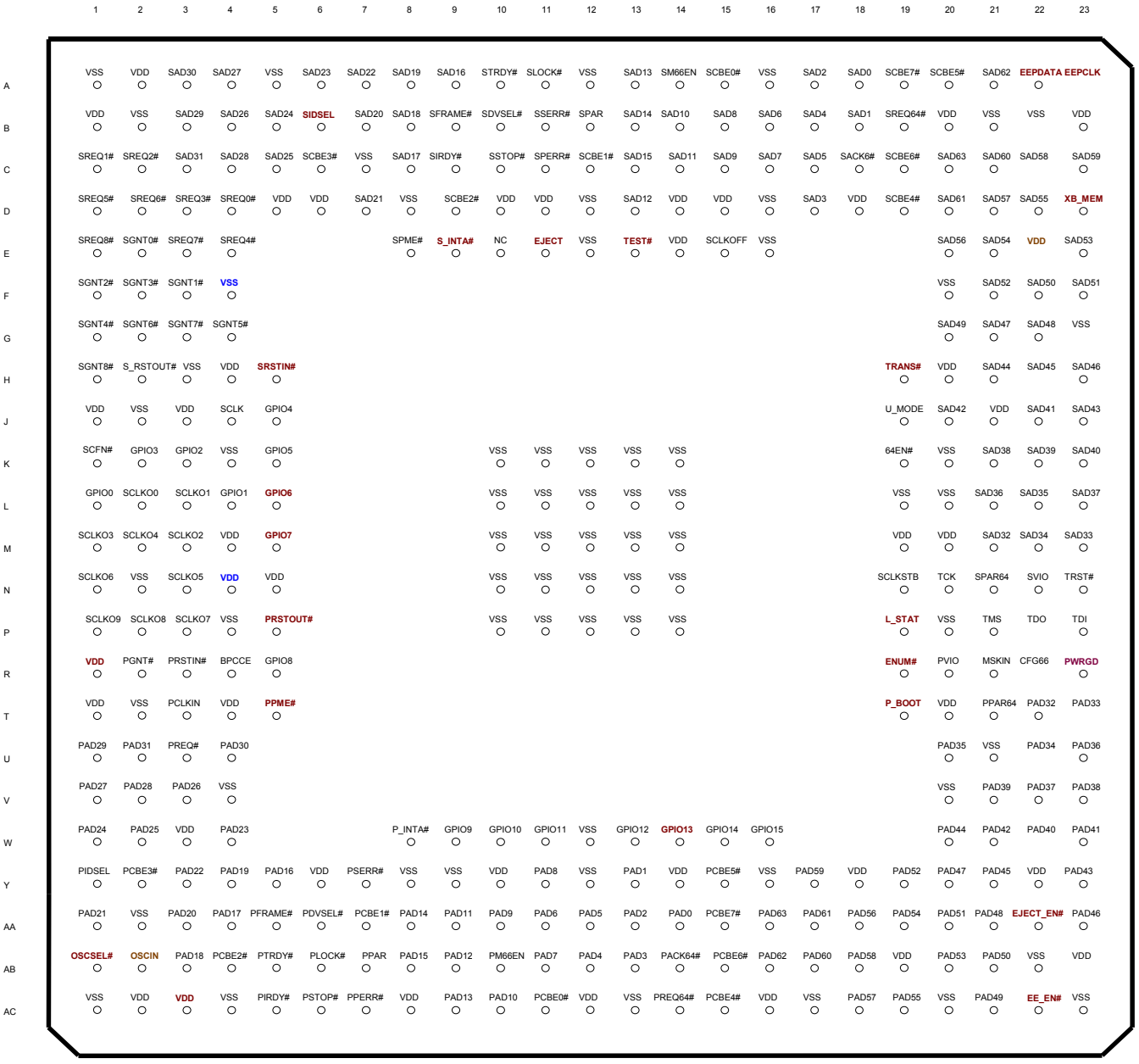
The basic design idea is optimized for the following:



# 4 Pin Diagram

This diagram depicts mainly Non-Transparent mode signal names. Please refer to pin description for Transparent mode signals that are multiplexed with Non-Transparent signals.

## PCI 6254 TOP VIEW



# 5 Signal Definition

**Signal Types**

| | |
|---|---|
| **PI** | PCI Input (5V signal input tolerant, I/O VDD=3.3V) |
| **PTS** | PCI Three-state bi-directional (5V signal input tolerant, I/O VDD=3.3V) |
| **PO** | PCI Output |
| **PSTS** | PCI Sustained Three-state Output. (5V signal input tolerant, I/O VDD=3.3V) |
| **I** | CMOS Input |
| **O** | CMOS Output |
| **IO** | CMOS Bi-direct |
| **PU** | Signal is pulled-up internally |
| **PD** | Signal is pulled-down internally |

## *5.1 Primary Bus Interface Signals*

| Name | Type | Description |
|---|---|---|
| P_AD[31:0] | PTS | **Primary address/data**: Multiplexed address and data bus. Address is indicated by P_FRAME# assertion. Write data is stable and valid when P_IRDY# is asserted and read data is stable and valid when P_TRDY# is asserted. Data is transferred on rising clock edges when both P_IRDY# and P_TRDY# are asserted. During bus idle, PCI 6254 drives P_AD to a valid logic level when P_GNT# is asserted. |
| P_CBE[3:0] | PTS | **Primary command/byte enables**: Multiplexed command field and byte enable field. During address phase, the initiator drives the transaction type on these pins. After that the initiator drives the byte enables during data phases. During bus idle, PCI 6254 drives P_CBE[3:0] to a valid logic level when P_GNT# is asserted. |
| P_PAR | PTS | **Primary Parity**: Parity is even across P_AD[31:0], P_CBE[3:0], and P_PAR (i.e. an even number of '1's). P_PAR is an input and is valid and stable one cycle after the address phase (indicated by assertion of P_FRAME#) for address parity. For write data phases, P_PAR is an input and is valid one clock after P_IRDY# is asserted. For read data phase, P_PAR is an output and is valid one clock after P_TRDY# is asserted. Signal P_PAR is three-stated one cycle after the PAD lines are three-stated. During bus idle, PCI 6254 drives PPAR to a valid logic level when P_GNT# is asserted. |
| P_FRAME# | PSTS | **Primary FRAME**: Driven by the initiator of a transaction to indicate the beginning and duration of an access. The deassertion of P_FRAME# indicates the final data phase requested by the initiator. Before being three-stated, it is driven to a deasserted state for one cycle. |
| P_IRDY# | PSTS | **Primary IRDY**: Driven by the initiator of a transaction to indicate its ability to complete the current data phase on the primary side. Once asserted in a data phase, it is not deasserted until end of the data phase. Before being three-stated, it is driven to a deasserted state for one cycle. |
| P_TRDY# | PSTS | **Primary TRDY**: Driven by the target of a transaction to indicate its ability to complete the current data phase on the primary side. Once asserted in a data phase, it is not deasserted until end of the data phase. Before being three-stated, it is driven to a deasserted state for one cycle. |

| | | |
|---|---|---|
| P_DEVSEL# | PSTS | **Primary Device Select**: Asserted by the target indicating that the device is accepting the transaction. As a master, PCI 6254 waits for the assertion of this signal within 5 cycles of P_FRAME# assertion; otherwise, terminate with master abort. Before being three-stated, it is driven to a deasserted state for one cycle. |
| P_STOP# | PSTS | **Primary STOP**: Asserted by the target indicating that the target is requesting the initiator to stop the current transaction. Before being three-stated, it is driven to a deasserted state for one cycle. |
| P_LOCK# | PI | **Primary LOCK**: Asserted by master for multiple transactions to complete. If LOCK function is not needed, as in the case that no secondary PCI devices support LOCK, this input should be pulled "HIGH" and should not be connected to the PCI bus. This function can also be disabled by setting register 46h bit 13 to 0. |
| P_IDSEL | PI | **Primary ID Select**. Used as chip select line for Type-0 configuration access to PCI 6254 configuration space. |
| P_PERR# | PSTS | **Primary Parity Error**: Asserted when a data parity error is detected for data received on the primary interface. Before being three-stated, it is driven to a deasserted state for one cycle. |
| P_SERR# | PSTS | **Primary System Error**: Can be driven LOW by any device to indicate a system error condition. PCI 6254 drives this pin during Transparent mode and when P_BOOT = 0 during Non-Transparent mode.<br>• Address parity error<br>• Posted write data parity error on target bus<br>• Secondary bus S_SERR# asserted<br>• Master abort during posted write transaction<br>• Target abort during posted write transaction<br>• Posted write transaction discarded<br>• Delayed write request discarded<br>• Delayed read request discarded<br>• Delayed transaction master timeout<br>This signal should be pulled up through an external resistor. |
| P_REQ# | PTS | **Primary Request**: This is asserted by PCI 6254 to indicate that it wants to start a transaction on the primary bus. PCI 6254 deasserts this pin for at least 2 PCI clock cycles before asserting it again. |
| P_GNT# | PI | **Primary Grant**: When asserted, PCI 6254 can access the primary bus. During idle and P_GNT# asserted, PCI 6254 will drive P_AD, P_CBE and P_PAR to valid logic level. |
| P_M66EN | PI | **Primary 66 MHz Enable:** Set high for 66MHz primary bus. This signal, along with the S_M66EN signal, controls the frequency output to the SCLKOUT pins. See Chapter 15 for more details |

## 5.2 Primary Bus Interface 64-bit Extension Signals

| Name | Type | Description |
|---|---|---|
| P_AD[63:32] | PTS | **Primary Address/Data Upper 32-bits**: Multiplexed address and data bus provide 32 extra pins. During address phase (when using the DAC command and when P_REQ64# is asserted), the upper 32-bits of a 64-bit address are transferred; otherwise, these bits are undefined. During a data phase, an additional 32-bit data is transferred if a 64-bit transaction is negotiated by the assertion of P_REQ64# and P_ACK64#. |
| P_CBE[7:4] | PTS | **Primary Command/byte Enables Upper 4-bits**: command and byte enable fields (Multiplexed). During address phase (when using the DAC command and when P_REQ64# is asserted), the actual bus command is transferred on these pins; otherwise, these bits are undefined. During a data phase, they indicate which byte lanes carry meaningful data if a 64-bit transaction is negotiated by the assertion of P_REQ64# and P_ACK64#. |
| P_REQ64# | PSTS | **Primary Request 64-bit Transfer**: When asserted by the current bus master, indicates it desire to transfer data using 64-bits. PCI 6254 ignores this input during reset and asserts P_REQ64# whenever the secondary PCI master transfers in 64 bit or the FORCE64 option is enabled. |
| P_ACK64# | PSTS | **Primary Acknowledge 64-bit Transfer**: When asserted by the target device, indicates it is willing to transfer data using 64-bits. It has the same timing as P_DEVSEL#. When deasserting, it is driven HIGH for one cycle before three-stated. |
| P_PAR64 | PTS | **Primary Interface Upper 32-bit Parity**: This provides the even parity bit that protects P_AD[63:32] and P_CBE[7:4]. It must be valid one clock after each address phase on any transaction in which P_REQ64# is asserted. |

## 5.3 Secondary Bus Interface Signals

| Name | Type | Description |
|---|---|---|
| S_AD[31:0] | PTS | **Secondary Address/Data**: Multiplexed address and data bus. Address is indicated by S_FRAME# assertion. Write data is stable and valid when S_IRDY# is asserted and read data is stable and valid when S_TRDY# is asserted. Data is transferred on rising clock edges when both S_IRDY# and S_TRDY# are asserted. During bus idle, PCI 6254 drives S_AD to a valid logic level when the S_GNT# is asserted. |
| S_CBE[3:0] | PTS | **Secondary Command/Byte Enables**: Multiplexed command field and byte enable field. During the address phase, the initiator drives the transaction type on these pins. After that the initiator drives the byte enables during data phases. During bus idle, PCI 6254 drives S_CBE[3:0] to a valid logic level when the internal grant is asserted. |
| S_PAR | PTS | **Secondary Parity**: Parity is even across S_AD[31:0], S_CBE[3:0], and S_PAR (i.e. an even number of '1's). S_PAR is an input and is valid and stable one cycle after the address phase (indicated by assertion of S_FRAME#) for address parity. For write data phases, S_PAR is an input and is valid one clock after S_IRDY# is asserted. For read data phase, S_PAR is an output and is valid one clock after S_TRDY# is asserted. Signal S_PAR is three-stated one cycle after the S_AD lines are three-stated. During bus idle, PCI 6254 drives S_PAR to a valid logic level when the internal grant is asserted. |
| S_FRAME# | PSTS | **Secondary FRAME**: Driven by the initiator of a transaction to indicate the beginning and duration of an access. The deassertion of S_FRAME# indicates the final data phase requested by the initiator. Before being three-stated, it is driven to a deasserted state for one cycle |
| S_IRDY# | PSTS | **Secondary IRDY**: Driven by the initiator of a transaction to indicate its ability to complete the current data phase on the primary side. Once asserted in a data phase, it is not deasserted until end of the data phase. Before being three-stated, it is driven to a deasserted state for one cycle. |
| S_TRDY# | PSTS | **Secondary TRDY**: Driven by the target of a transaction to indicate its ability to complete the current data phase on the primary side. Once asserted in a data phase, it is not deasserted until end of the data phase. Before being three-stated, it is driven to a deasserted state for one cycle. |
| S_DEVSEL# | PSTS | **Secondary Device Select**: Asserted by the target indicating that the device is accepting the transaction. As a master, PCI 6254 waits for the assertion of this signal within 5 cycles of S_FRAME# assertion; otherwise, terminate with master abort. Before being three-stated, it is driven to a deasserted state for one cycle. |
| S_STOP# | PSTS | **Secondary STOP**: Asserted by the target indicating that the target is requesting the initiator to stop the current transaction. Before being three-stated, it is driven to a deasserted state for one cycle. |
| S_LOCK# | PSTS | Secondary LOCK: Asserted by master to complete multiple transactions. |
| S_PERR# | PSTS | **Secondary Parity Error**: Asserted when a data parity error is detected for data received on the primary interface. Before being three-stated, it is driven to a deasserted state for one cycle. |

| | | |
|---|---|---|
| S_SERR# | PSTS | **Secondary System Error**: Can be driven LOW by any device to indicate a system error condition. PCI 6254 drives this pin only during Non-Transparent Mode with P_BOOT = 1 and if the following:<br>• Address parity error<br>• Posted write data parity error on target bus<br>• Primary P_SERR# asserted<br>• Master abort during posted write transaction<br>• Target abort during posted write transaction<br>• Posted write transaction discarded<br>• Delayed write request discarded<br>• Delayed read request discarded<br>• Delayed transaction master timeout<br>This signal should be pulled up through an external resistor. |
| S_REQ#[0] | PTS | **Secondary Request 0**: When External arbitration is not activated, this is asserted by external device to indicate that it wants to start a transaction on the Secondary bus. It must be externally pulled up through resistors to VDD.<br><br>When External arbitration is active, this becomes the External Grant input to PCI 6254.<br><br>When Universal Mode is active and when operated in Internal Arbitration mode, this is the PCI 6254 Secondary port Request output. |
| S_REQ#[8:1] | PI | **Secondary Requests**: This is asserted by external device to indicate that it wants to start a transaction on the Secondary bus. They must be externally pulled up through resistors to VDD. |
| S_GNT#[0] | PTS | **Secondary Grant 0**: This pin behaves like S_GNT#[8:1] under Transparent Mode and External arbitration is not activated.<br><br>When External arbitration is active, this becomes the External Bus Request output from PCI 6254.<br><br>When Universal Mode is active and when operated in Internal Arbitration mode, this is the PCI 6254 Secondary port Grant input. |
| S_GNT#[8:1] | PO | **Secondary Grants**: PCI 6254 asserts this pin to access the secondary bus. PCI 6254 deasserts this pin for at least 2 PCI clock cycles before asserting it again. During idle and S_GNT# asserted, PCI 6254 will drive S_AD, S_CBE and S_PAR to valid logic levels. |
| S_M66EN | PTS | **Secondary 66 MHz Enable:** Drive low if P_M66EN is low, otherwise driven from outside to select 66MHz or 33MHz. This signal, along with the P_M66EN signal, controls the frequency output to the S_CLKOUTn pins. This pin should be pulled High or Low externally.<br>See Chapter 15 for more details |

## 5.4 Secondary Bus Interface 64-bit Extension Signals

| Name | Type | Description |
|---|---|---|
| S_AD[63:32] | PTS | **Secondary Address/Data Upper 32-bits:** Multiplexed address and data bus provide 32 extra pins. During address phase (when using the DAC command and when S_REQ64# is asserted), the upper 32-bits of a 64-bit address are transferred; otherwise, these bits are undefined. During a data phase, an additional 32-bit data is transferred when a 64-bit transaction has been negotiated by the assertion of S_REQ64# and S_ACK64#. |
| S_CBE[7:4] | PTS | **Secondary Command/Byte Enables Upper 4-bits:** Multiplexed command field and byte enable field. During address phase (when using the DAC command and when S_REQ64# is asserted), the actual bus command is transferred on these pins; otherwise, these bits are undefined. During a data phase, they indicate which byte lanes carry meaningful data when a 64-bit transaction has been negotiated by the assertion of S_REQ64# and S_ACK64#. |
| S_REQ64# | PSTS | **Secondary Request 64-bit Transfer:** When asserted by the current bus master, indicates it desire to transfer data using 64-bits. PCI 6254 ignores this input during reset and asserts S_REQ64# whenever the primary PCI master transfers in 64 bit or the FORCE64 option is enabled. |
| S_ACK64# | PSTS | **Secondary Acknowledge 64-bit Transfer:** When asserted by the target device, indicates it is willing to transfer data using 64-bits. It has the same timing as S_DEVSEL#. When deasserting, it is driven HIGH for one cycle before three-stated. |
| S_PAR64 | PTS | **Secondary Interface Upper 32-bit Parity:** Provides even parity bit that protects S_AD[63:32] and S_CBE[7:4]. It must be valid one clock after each address phase on any transaction in which S_REQ64# is asserted. |

## 5.5 Clock Related Signals

| Name | Type | Description |
|---|---|---|
| P_CLKIN | I | **Primary CLK input:** Provides timing for all transactions on primary interface. |
| S_CLKIN | I | **Secondary CLK input:** Provides timing for all transactions on secondary interface **except** during Universal Non-Transparent Mode (U_MODE = 1 and TRANS# = 1). In Universal Non-Transparent mode, this input is ignored by PCI 6254 and the internal logic uses the input clock from the S_CLK0 pin. |
| S_CLK0 | I/O | **Secondary CLK output:** Provides S_CLKIN or OSCIN (if enabled) phase synchronous output clocks. This clock can be turned off using the S_CLKOFF pin.<br><br>**S_CLK0** output is three-stated during Universal Non-Transparent Mode. This allows S_CLK0 to take the CLK signal from a CPCI back plane for the Secondary port logic. When in CPCI SYSTEM slot, S_CLK0 drives the CPCI and when in PERIPHERAL slot, back plane clock drives S_CLK0 pin. |
| S_CLK[9:1] | O | **Secondary CLK output:** Provides S_CLKIN or OSCIN (if enabled) phase synchronous output clocks. These clocks can be turned off using the S_CLKOFF pin. |
| S_CLKOFF | I-PD | 0 = S_CLK[9:0] output is enabled. This enable can be over-ridded by MSKIN input controls and clock output control register bits.<br>1 = S_CLK[9:0] output are driven LOW. This disable can only be over-ridded by clock output control register bits. |
| S_CLKIN_ STB | I-PU | 1 = External Secondary Clock PLL and S_CLKIN clock are stable signals.<br>0 = S_RSTOUT# will not go inactive until S_CLKIN_STB is “1”.<br>If not used, this pin should be connected to 3.3V supply. |
| MSK_IN | I | **Secondary Clock Disable Serial Input.** This signal is used by the hardware mechanism to disable secondary clock outputs. The serial stream is received by MSK_IN, starting when P_RSTIN# is detected deasserted and S_RSTOUT# is detected asserted. This serial data is used for selectively disabling secondary clock outputs and is shifted into the secondary clock control configuration register.<br>This input can be tied low to enable all secondary clock outputs. If tied high, the clocks will be active until high after reset. After the “1”s have been shifted in, the clocks will be driven high. |
| OSCSEL# | I | **External Oscillator Enable.** Enables connection of external clock for the secondary interface. If low, secondary bus clock outputs will use the clock signal from OSCIN input instead of P_CLKIN to generate S_CLK[9:0]. If HIGH, P_CLKIN is used. This pin must NOT be left unconnected. |
| OSCIN | I | **External Oscillator Input.** External clock input used to generate secondary output clocks when enabled through OSCSEL# pin. |

## 5.6 Reset Signals

| Name | Type | Description |
|---|---|---|
| PWRGD | I | **Power Good Input: Important: PCI 6254 requires a clean low to high transition PWRGD input. The PWRGD is not internally debounced. It must be debounced externally and a HIGH input must reflect that the power is indeed stable.** When this input is low, all state machines and registers in PCI 6254 are reset and all outputs are three-stated. This HIGH input should be 3.3V.<br><br>**Important: The PWRGD is not internally debounced. It must be debounced externally and a HIGH input must reflect that the power is indeed stable.** Its asserting and deasserting edges can be asynchronous to P_CLK and S_CLK. |
| P_RSTOUT# | PO | **Primary Reset Output:** This is only valid in Non-Transparent Mode and it is asserted when any of the following conditions is met:<br><br>1. Signal S_RSTIN# is asserted when Primary Port has boot priority (PBOOT =1).<br><br>2. The Primary Reset bit in the Non-Transparent Diagnostic control register in configuration space is set. |
| P_RSTIN# | PI | **Primary Reset:** When P_RSTIN# is active, outputs are asynchronously three-stated and P_SERR# and P_GNT# floated. All primary port PCI standard configuration registers 0h-3Fh revert to their default state.<br><br>**When asserted, all primary PCI signals are three-stated.** |
| S_RSTOUT# | PTS | **Secondary Reset Output:** Asserted when any of the following conditions is met:<br><br>1. Signal P_RSTIN# is asserted.<br><br>2. The Secondary reset bit in the bridge control register (bit 6 Register 42h in Non-Transparent Mode and Register 3Eh in Transparent Mode) in configuration space is set.<br><br>In Universal Non-Transparent Mode (U_MODE = 1, TRANS# = 1), this output is disabled and S_RSTOUT# pin is used as the equivalent of S_RSTIN# input pin. |
| S_RSTIN# | I | **Secondary Reset Input:** In Non-Transparent Mode, active low input will cause all secondary port control logic to reset. Primary port control logic is not affected.<br><br>In Transparent or Universal mode, this pin is not used and should be pulled high.<br><br>In Universal Non-Transparent Mode (U_MODE = 1, TRANS# = 1), this input is ignored and S_RSTOUT# pin is used as the equivalent of S_RSTIN#. |

## 5.7 Hot Swap Signals

| Name | Type | Description |
|---|---|---|
| ENUM# | PTS | **Hot Swap Interrupt:** An open drain bussed signal to signal a change in status for the chip. This is an output only signal. It is asserted through the Hot Swap registers. |
| L_STAT | IO | **Hot Swap LED:** Output to indicate the status of software connection process. **If Hot Swap is not used, L_STAT must be at logic "0" if EJECT_EN# is "1".** |
| EJECT | I | **Hot Swap Eject:** Pin used to detect the insertion of Hot Swap device and is debounced internally. However some external debounce is still recommended. When signal is asserted, PCI 6254 asserts ENUM# pin after card initialization is finished. An external pull low resister is recommended. **If Hot Swap is not used, EJECT must be at logic "0" if EJECT_EN# is "0".** |
| EJECT_EN# | I | **Ejector Pin Use Enable:** This pin should always be connected to logic "0". 1 = Reserved and should not be used. |

## 5.8 Miscellaneous Signals

| Name | Type | Description |
|---|---|---|
| S_CFN# | I | **Internal Arbiter Enable:**<br>0 = Use internal arbiter.<br>1 = Use external arbiter.<br>If U_MODE = 0: S_REQ0# becomes External Arbiter GNT# input and S_GNT0# becomes REQ# output to External Arbiter.<br>If U_MODE = 1: S_REQ0# becomes REQ# output to External Arbiter and S_GNT0# becomes External Arbiter GNT# input. |
| CFG66 | I | **Primary Config 66MHz:** The state of this pin is reflected in the Secondary status register at offset 1Eh. Other than this, the pin has no effect on PCI 6254 operation. |
| BPCC_EN | I | **Bus/Power Clock Control Management pin.** When signal is tied high and the PCI 6254 is placed in the D3hot power state, the PCI 6254 places the secondary bus in the B2 power state. The PCI 6254 disables the secondary clocks and drives them to 0. When tied low, placing the PCI 6254 in the D3hot power state has no effect on the secondary bus clocks. |
| GPIO[3:0] | IO-PU | **General Purpose Input Output pins.** These 8 general purpose signals are programmable as either input-only or bi-directional signals by writing the GPIO output enable control register. During PRST# asserted, GPIO[3:0] are used to shift in the clock disable serial data. |
| GPIO[7:4] | IO-PU | **General Purpose Input Output pins.** These 4 general purpose signals are internally pulled up and are programmable as either input-only or bi-directional signals by writing the GPIO output enable control register.<br>During non-transparent mode, GPIO[5] can be enabled as an external interrupt source on the Primary port to trigger S_INTA#.<br>During non-transparent mode, GPIO[4] can be enabled as an external interrupt source on the Secondary port to trigger P_INTA#. |

| | | |
|---|---|---|
| GPIO[15:8] | IO | **General Purpose Input Output pins.** These 8 general purpose signals are programmable as either input-only or bi-directional signals by writing the GPIO output enable control register.<br><br>During power up reset, the status of these pins are latched in register xxh for general user defined use. They can be left unconnected if not used.<br><br>**Recommend Use (Must be 3.3V input):**<br><br>**GPIO15: Primary Power State:** 1 = Primary port power is stable.<br><br>**GPIO14: Secondary Power State:** 1 = Secondary port power is stable. |
| P_PME# | PTS | **Primary PME#:** Primary PCI Port Power management event interrupt. This is used by Secondary Port devices to wake up Primary port host. In Transparent mode, and in Non-Transparent with primary port has lower boot priority, P_PME# is always output and reflects the state of S_PME# input if enabled. |
| S_PME# | PTS | **Secondary PME#:** Secondary PCI Port Power management event interrupt. This is used by Primary Port devices to wake up Secondary port subsystem. In Non-Transparent with secondary port has lower boot priority, S_PME# is always output and reflects the state of P_PME# input if enabled. |
| EEPCLK | IO | **EEPROM Clock.** This pin is the clock signal to the EEPROM interface used during Autoload and for VPD functions. This pin is three-stated if EE_EN# = 1. |
| EEPDATA | IO | **EEPROM Serial Data.** This pin is serial data interface to the EEPROM. This pin is three-stated if EE_EN# = 1. |
| EE_EN# | I | **EEPROM Enable.** This input should be “0” to enable EEPROM use. Otherwise it should be connect to logic “1” state. |
| 64EN# | I-PU | **64 Bit Device Status**<br><br>This input has no hardware control function and is recommended for software read only as below. This status is reflected at register 52h bit 4.<br><br>0 = port uses 64 bit bus.<br><br>1 = Use 32 bit bus. |
| TRANS# | I-PD | **Enable Transparent Mode:**<br>0 = PCI 6254 is configured as a standard PCI-to-PCI bridge.<br>1 = PCI 6254 is in Non-Transparent mode.<br>In CPCI universal bridge applications, this can be connected directly to the SYSEN# pin. |
| U_MODE | I-PD | **Enable Universal Mode:**<br>1 = PCI 6254 is configured as a Universal Bridge. |
| TEST# | I-PU | 0 = PCI 6254 is set to test mode. During normal use, the pin should be left unconnected. |
| PVIO | I | **Primary Interface I/O Voltage** This signal must be tied to either 3.3V or 5V, depending on the signaling voltage of the primary interface. |
| SVIO | I | **Secondary Interface I/O Voltage** This signal must be tied to either 3.3V or 5V, depending on the signaling voltage of the secondary interface. |

## 5.9 Multiplexed Transparent and Non-Transparent Mode Signals

| Name | Type | Description |
|---|---|---|
| XB_MEM | I-PD | **Private Device or Cross Bridge Memory Space Enable**: After power up, the functions set by this bit can be modified by software at Chip Control Register bits 2 and 3. This input pin should not be left unconnected.<br><br>**(Non-Transparent mode) Cross-Bridge Memory Window Enable:**<br>When this bit is "1", PCI 6254 will automatically claim 16M of memory space. This allows the boot up of the Low priority Boot Port to move forward without waiting for the Priority Boot port to program the corresponding Memory BAR registers. Although the default claims 16M, the BAR registers can be changed by EEPROM or software to change the window size If this pin is "1", the P or S PORT_READY mechanism will NOT be relevant and access to BAR register will not be retried.<br><br>**(Transparent mode) Private Memory Enable:**<br>This function is not available and the input must be set to 0 during Transparent mode. |
| P_INTA# / P_CLKRUN# | PTS | In Non-Transparent and Universal mode, this is Primary port interrupt output **P_INTA#**.<br><br>In Transparent Mode, this is **Primary CLKRUN** used by the central resource to stop the PCI clock or to slow it down. This function is by default disabled and need to be enabled by software. If this feature is not enabled, this pin can be left unconnected. |
| S_INTA# / S_CLKRUN# | PTS | In Non-Transparent mode, this is Secondary port interrupt output **S_INTA#**.<br><br>In Transparent Mode, this is **Secondary CLKRUN** used by the central resource to stop the PCI clock or to slow it down. This function is by default disabled and need to be enabled by software. If this feature is not enabled, this pin can be left unconnected. |
| S_IDSEL | PI | **Secondary ID Select:** Used as chip select line for Type-0 configuration access to PCI 6254 secondary configuration space. Pin is used only during Non-transparent mode. |
| P_BOOT | I-PU | **Primary Port Boot indicator input:**<br><br>P_BOOT = 1: Primary Port has boot priority: Primary Port must set P_PORT_READY bit before Secondary Port can proceed to boot.<br><br>P_BOOT = 0: Secondary Port has boot priority: Secondary Port must set S_PORT_READY bit before Primary Port can proceed to boot. |

## 5.10 JTAG/Boundary Scan Interface Signals

| Name | Type | Description |
|---|---|---|
| TCK | I-PU | **Test Clock:** Used to clock state information and test data into and out of PCI 6254 during operation of the TAP.<br>This pin should be pulled high or pulled low to a known state using an external resistor. |
| TMS | I-PU | **Test Mode Select:** Used to control the state of the TAP controller in PCI 6254.<br>This pin should be pulled high or pulled low to a known state using an external resistor. |
| TDO | O | **Test Data Output:** Used to serially shift test data and test instructions out of PCI 6254 during TAP operation. |
| TDI | I-PU | **Test Data Input:** Used to serially shift test data and test instructions into PCI 6254 during TAP operation.<br>This pin should be pulled high or pulled low to a known state using an external resistor. |
| TRST# | I | **Test Reset:** It provides an asynchronous initialization of the TAP controller. This pin MUST be pulled high or pulled low to a known state using an external resistor. We recommend pulling low using a 330ohm resistor. |

## 5.11 Power Signals

| Name | Type | Description |
|---|---|---|
| VDD | | +3.3V |
| GND | | Ground |

## 5.12 Pin Assignment Sorted By Location

| LOCATION | PIN NAME |
|---|---|
| A01 | VSS |
| A02 | VDD |
| A03 | S_AD[30] |
| A04 | S_AD[27] |
| A05 | VSS |
| A06 | S_AD[23] |
| A07 | S_AD[22] |
| A08 | S_AD[19] |
| A09 | S_AD[16] |
| A10 | S_TRDY# |
| A11 | S_LOCK# |
| A12 | VSS |
| A13 | S_AD[13] |
| A14 | S_M66EN |
| A15 | S_CBE#[0] |
| A16 | VSS |
| A17 | S_AD[02] |
| A18 | S_AD[00] |
| A19 | S_CBE#[7] |
| A20 | S_CBE#[5] |
| A21 | S_AD[62] |
| A22 | EEPDATA |
| A23 | EEPCLK |
| B01 | VDD |
| B02 | VSS |
| B03 | S_AD[29] |
| B04 | S_AD[26] |
| B05 | S_AD[24] |
| **B06** | **SIDSEL** |
| B07 | S_AD[20] |
| B08 | S_AD[18] |
| B09 | S_FRAME# |
| B10 | S_DEVSEL# |
| B11 | S_SERR# |
| B12 | S_PAR |
| B13 | S_AD[14] |
| B14 | S_AD[10] |
| B15 | S_AD[08] |
| B16 | S_AD[06] |
| B17 | S_AD[04] |
| B18 | S_AD[01] |
| B19 | S_REQ64# |
| B20 | VDD |
| B21 | VSS |
| B22 | VSS |
| B23 | VDD |
| C01 | S_REQ#[1] |
| C02 | S_REQ#[2] |
| C03 | S_AD[31] |
| C04 | S_AD[28] |
| C05 | S_AD[25] |
| C06 | S_CBE#[3] |
| C07 | VSS |
| C08 | S_AD[17] |
| C09 | S_IRDY# |
| C10 | S_STOP# |
| C11 | S_PERR |
| C12 | S_CBE#[1] |
| C13 | S_AD[15] |
| C14 | S_AD[11] |
| C15 | S_AD[09] |
| C16 | S_AD[07] |
| C17 | S_AD[05] |
| C18 | S_ACK64# |
| C19 | S_CBE#[6] |
| C20 | S_AD[63] |
| C21 | S_AD[60] |
| C22 | S_AD[58] |
| C23 | S_AD[59] |
| D01 | S_REQ#[5] |
| D02 | S_REQ#[6] |
| D03 | S_REQ#[3] |
| D04 | S_REQ#[0] |
| D05 | VDD |
| D06 | VDD |
| D07 | S_AD[21] |
| D08 | VSS |
| D09 | S_CBE#[2] |
| D10 | VDD |
| D11 | VDD |
| D12 | VSS |
| D13 | S_AD[12] |
| D14 | VDD |
| D15 | VDD |
| D16 | VSS |
| D17 | S_AD[03] |
| D18 | VDD |
| D19 | S_CBE#[4] |
| D20 | S_AD[61] |
| D21 | S_AD[57] |
| D22 | S_AD[55] |
| **D23** | **XB_MEM** |
| E01 | S_REQ#[8] |
| E02 | S_GNT#[0] |
| E03 | S_REQ#[7] |
| E04 | S_REQ#[4] |
| **E08** | **S_PME#** |
| **E09** | **S_INTA#** |
| **E10** | **NC** |
| **E11** | **EJECT** |
| **E12** | **VSS** |
| **E13** | **TEST#** |
| **E14** | **VDD** |
| **E15** | **SCLKOFF** |
| **E16** | **VSS** |
| E20 | S_AD[56] |
| E21 | S_AD[54] |
| E22 | VDD |
| E23 | S_AD[53] |
| F01 | S_GNT#[2] |
| F02 | S_GNT#[3] |
| F03 | S_GNT#[1] |
| F04 | VSS |
| F20 | VSS |
| F21 | S_AD[52] |
| F22 | S_AD[50] |
| F23 | S_AD[51] |
| G01 | S_GNT#[4] |
| G02 | S_GNT#[6] |
| G03 | S_GNT#[7] |
| G04 | S_GNT#[5] |
| G20 | S_AD[49] |
| G21 | S_AD[47] |
| G22 | S_AD[48] |
| G23 | VSS |
| H01 | S_GNT#[8] |
| H02 | S_RSTOUT# |
| H03 | VSS |
| H04 | VDD |
| **H05** | **S_RSTIN#** |
| **H19** | **TRANS#** |
| H20 | VDD |
| H21 | S_AD[44] |
| H22 | S_AD[45] |
| H23 | S_AD[46] |
| J01 | VDD |
| J02 | VSS |
| J03 | VDD |
| J04 | S_CLK |
| **J05** | **GPIO4** |
| **J19** | **U_MODE** |
| J20 | S_AD[42] |
| J21 | VDD |
| J22 | S_AD[41] |
| J23 | S_AD[43] |
| K01 | S_CFN# |
| K02 | GPIO[03] |
| K03 | GPIO[02] |
| K04 | VSS |
| **K05** | **GPIO5** |
| **K10** | **VSS** |
| **K11** | **VSS** |
| **K12** | **VSS** |
| **K13** | **VSS** |
| **K14** | **VSS** |
| **K19** | **64EN#** |
| K20 | VSS |
| K21 | S_AD[38] |
| K22 | S_AD[39] |
| K23 | S_AD[40] |
| L01 | GPIO[0] |
| L02 | S_CLK_O[0] |
| L03 | S_CLK_O[1] |
| L04 | GPIO[01] |
| **L05** | **GPIO6** |
| **L10** | **VSS** |
| **L11** | **VSS** |
| **L12** | **VSS** |
| **L13** | **VSS** |
| **L14** | **VSS** |
| **L19** | **VSS** |
| L20 | VSS |
| L21 | S_AD[36] |
| L22 | S_AD[35] |
| L23 | S_AD[37] |
| M01 | S_CLK_O[3] |
| M02 | S_CLK_O[4] |
| M03 | S_CLK_O[2] |
| M04 | VDD |
| **M05** | **GPIO7** |
| **M10** | **VSS** |
| **M11** | **VSS** |
| **M12** | **VSS** |
| **M13** | **VSS** |
| **M14** | **VSS** |
| **M19** | **VDD** |
| M20 | VDD |
| M21 | S_AD[32] |
| M22 | S_AD[34] |
| M23 | S_AD[33] |
| N01 | S_CLK_O[6] |
| N02 | VSS |
| N03 | S_CLK_O[5] |
| N04 | VDD |
| **N05** | **VDD** |
| **N10** | **VSS** |
| **N11** | **VSS** |
| **N12** | **VSS** |
| **N13** | **VSS** |
| **N14** | **VSS** |
| **N19** | **SCLKSTB** |
| N20 | TCK |
| N21 | S_PAR64 |
| N22 | S_VIO |
| N23 | TRST# |
| P01 | S_CLK_O[9] |
| P02 | S_CLK_O[8] |
| P03 | S_CLK_O[7] |
| P04 | VSS |
| **P05** | **PRSTOUT#** |
| **P10** | **VSS** |
| **P11** | **VSS** |
| **P12** | **VSS** |
| **P13** | **VSS** |
| **P14** | **VSS** |
| **P19** | **L_STAT** |
| P20 | VSS |
| P21 | TMS |
| P22 | TDO |

| | |
|---|---|
| P23 | TDI |
| R01 | VDD |
| R02 | P_GNT# |
| R03 | P_RSTIN# |
| R04 | BPCCE |
| **R05** | **GPIO8** |
| **R19** | **ENUM#** |
| R20 | P_VIO |
| R21 | MSK_IN |
| R22 | CONFIG66 |
| R23 | PWRGD |
| T01 | VDD |
| T02 | VSS |
| T03 | P_CLKIN |
| T04 | VDD |
| **T05** | **P_PME#** |
| **T19** | **P_BOOT** |
| T20 | VDD |
| T21 | P_PAR64 |
| T22 | P_AD[32] |
| T23 | P_AD[33] |
| U01 | P_AD[29] |
| U02 | P_AD[31] |
| U03 | P_REQ# |
| U04 | P_AD[30] |
| U20 | P_AD[35] |
| U21 | VSS |
| U22 | P_AD[34] |
| U23 | P_AD[36] |
| V01 | P_AD[27] |
| V02 | P_AD[28] |
| V03 | P_AD[26] |
| V04 | VSS |
| V20 | VSS |
| V21 | P_AD[39] |
| V22 | P_AD[37] |
| V23 | P_AD[38] |
| W01 | P_AD[24] |

| | |
|---|---|
| W02 | P_AD[25] |
| W03 | VDD |
| W04 | P_AD[23] |
| **W08** | **P_INTA#** |
| **W09** | **GPIO9** |
| **W10** | **GPIO10** |
| **W11** | **GPIO11** |
| **W12** | **VSS** |
| **W13** | **GPIO12** |
| **W14** | **GPIO13** |
| **W15** | **GPIO14** |
| **W16** | **GPIO15** |
| W20 | P_AD[44] |
| W21 | P_AD[42] |
| W22 | P_AD[40] |
| W23 | P_AD[41] |
| Y01 | P_IDSEL |
| Y02 | P_CBE#[3] |
| Y03 | P_AD[22] |
| Y04 | P_AD[19] |
| Y05 | P_AD[16] |
| Y06 | VDD |
| Y07 | P_SERR# |
| Y08 | VSS |
| Y09 | VSS |
| Y10 | VDD |
| Y11 | P_AD[08] |
| Y12 | VSS |
| Y13 | P_AD[01] |
| Y14 | VDD |
| Y15 | P_CBE#[5] |
| Y16 | VSS |
| Y17 | P_AD[59] |
| Y18 | VDD |
| Y19 | P_AD[52] |
| Y20 | P_AD[47] |
| Y21 | P_AD[45] |
| Y22 | VDD |

| | |
|---|---|
| Y23 | P_AD[43] |
| AA01 | P_AD[21] |
| AA02 | VSS |
| AA03 | P_AD[20] |
| AA04 | P_AD[17] |
| AA05 | P_FRAME# |
| AA06 | P_DEVSEL# |
| AA07 | P_CBE#[1] |
| AA08 | P_AD[14] |
| AA09 | P_AD[11] |
| AA10 | P_AD[09] |
| AA11 | P_AD[06] |
| AA12 | P_AD[05] |
| AA13 | P_AD[02] |
| AA14 | P_AD[00] |
| AA15 | P_CBE#[7] |
| AA16 | P_AD[63] |
| AA17 | P_AD[61] |
| AA18 | P_AD[56] |
| AA19 | P_AD[54] |
| AA20 | P_AD[51] |
| AA21 | P_AD[48] |
| **AA22** | **EJECT_EN#** |
| AA23 | P_AD[46] |
| AB01 | OSCSEL# |
| AB02 | OSCIN |
| AB03 | P_AD[18] |
| AB04 | P_CBE#[2] |
| AB05 | P_TRDY# |
| AB06 | P_LOCK# |
| AB07 | P_PAR |
| AB08 | P_AD[15] |
| AB09 | P_AD[12] |
| AB10 | P_M66EN |
| AB11 | P_AD[07] |
| AB12 | P_AD[04] |
| AB13 | P_AD[03] |
| AB14 | P_ACK64# |

| | |
|---|---|
| AB15 | P_CBE#[6] |
| AB16 | P_AD[62] |
| AB17 | P_AD[60] |
| AB18 | P_AD[58] |
| AB19 | VDD |
| AB20 | P_AD[53] |
| AB21 | P_AD[50] |
| AB22 | VSS |
| AB23 | VDD |
| AC01 | VSS |
| AC02 | VDD |
| AC03 | VDD |
| AC04 | VSS |
| AC05 | P_IRDY# |
| AC06 | P_STOP# |
| AC07 | P_PERR# |
| AC08 | VDD |
| AC09 | P_AD[13] |
| AC10 | P_AD[10] |
| AC11 | P_CBE#[0] |
| AC12 | VDD |
| AC13 | VSS |
| AC14 | P_REQ64# |
| AC15 | P_CBE#[4] |
| AC16 | VDD |
| AC17 | VSS |
| AC18 | P_AD[57] |
| AC19 | P_AD[55] |
| AC20 | VSS |
| AC21 | P_AD[49] |
| **AC22** | **EE_EN#** |
| AC23 | VSS |

## 5.13 Pin Assignment Sorted By Pin Name

| LOCATION | PIN NAME |
|---|---|
| **K19** | **64EN#** |
| R04 | BPCCE |
| R22 | CONFIG66 |
| **AC22** | **EE_EN#** |
| A23 | EEPCLK |
| A22 | EEPDATA |
| **E11** | **EJECT** |
| **AA22** | **EJECT_EN#** |
| **R19** | **ENUM#** |
| L01 | GPIO[0] |
| L04 | GPIO[01] |
| K03 | GPIO[02] |
| K02 | GPIO[03] |
| **W10** | **GPIO10** |
| **W11** | **GPIO11** |
| **W13** | **GPIO12** |
| **W14** | **GPIO13** |
| **W15** | **GPIO14** |
| **W16** | **GPIO15** |
| **J05** | **GPIO4** |
| **K05** | **GPIO5** |
| **L05** | **GPIO6** |
| **M05** | **GPIO7** |
| **R05** | **GPIO8** |
| **W09** | **GPIO9** |
| **P19** | **L_STAT** |
| R21 | MSK_IN |
| **E10** | **NC** |
| AB02 | OSCIN |
| AB01 | OSCSEL# |
| AB14 | P_ACK64# |
| AA14 | P_AD[00] |
| Y13 | P_AD[01] |
| AA13 | P_AD[02] |
| AB13 | P_AD[03] |
| AB12 | P_AD[04] |
| AA12 | P_AD[05] |
| AA11 | P_AD[06] |
| AB11 | P_AD[07] |
| Y11 | P_AD[08] |
| AA10 | P_AD[09] |
| AC10 | P_AD[10] |
| AA09 | P_AD[11] |
| AB09 | P_AD[12] |
| AC09 | P_AD[13] |
| AA08 | P_AD[14] |
| AB08 | P_AD[15] |
| Y05 | P_AD[16] |
| AA04 | P_AD[17] |
| AB03 | P_AD[18] |
| Y04 | P_AD[19] |
| AA03 | P_AD[20] |
| AA01 | P_AD[21] |
| Y03 | P_AD[22] |
| W04 | P_AD[23] |
| W01 | P_AD[24] |
| W02 | P_AD[25] |
| V03 | P_AD[26] |
| V01 | P_AD[27] |
| V02 | P_AD[28] |
| U01 | P_AD[29] |
| U04 | P_AD[30] |
| U02 | P_AD[31] |
| T22 | P_AD[32] |
| T23 | P_AD[33] |
| U22 | P_AD[34] |
| U20 | P_AD[35] |
| U23 | P_AD[36] |
| V22 | P_AD[37] |
| V23 | P_AD[38] |
| V21 | P_AD[39] |
| W22 | P_AD[40] |
| W23 | P_AD[41] |
| W21 | P_AD[42] |
| Y23 | P_AD[43] |
| W20 | P_AD[44] |
| Y21 | P_AD[45] |
| AA23 | P_AD[46] |
| Y20 | P_AD[47] |
| AA21 | P_AD[48] |
| AC21 | P_AD[49] |
| AB21 | P_AD[50] |
| AA20 | P_AD[51] |
| Y19 | P_AD[52] |
| AB20 | P_AD[53] |
| AA19 | P_AD[54] |
| AC19 | P_AD[55] |
| AA18 | P_AD[56] |
| AC18 | P_AD[57] |
| AB18 | P_AD[58] |
| Y17 | P_AD[59] |
| AB17 | P_AD[60] |
| AA17 | P_AD[61] |
| AB16 | P_AD[62] |
| AA16 | P_AD[63] |
| **T19** | **P_BOOT** |
| AC11 | P_CBE#[0] |
| AA07 | P_CBE#[1] |
| AB04 | P_CBE#[2] |
| Y02 | P_CBE#[3] |
| AC15 | P_CBE#[4] |
| Y15 | P_CBE#[5] |
| AB15 | P_CBE#[6] |
| AA15 | P_CBE#[7] |
| T03 | P_CLKIN |
| AA06 | P_DEVSEL# |
| AA05 | P_FRAME# |
| R02 | P_GNT# |
| Y01 | P_IDSEL |
| **W08** | **P_INTA#** |
| AC05 | P_IRDY# |
| AB06 | P_LOCK# |
| AB10 | P_M66EN |
| AB07 | P_PAR |
| T21 | P_PAR64 |
| AC07 | P_PERR# |
| **T05** | **P_PME#** |
| U03 | P_REQ# |
| AC14 | P_REQ64# |
| R03 | P_RSTIN# |
| Y07 | P_SERR# |
| AC06 | P_STOP# |
| AB05 | P_TRDY# |
| R20 | P_VIO |
| **P05** | **PRSTOUT#** |
| **D23** | **XB_MEM** |
| R23 | PWRGD |
| C18 | S_ACK64# |
| A18 | S_AD[00] |
| B18 | S_AD[01] |
| A17 | S_AD[02] |
| D17 | S_AD[03] |
| B17 | S_AD[04] |
| C17 | S_AD[05] |
| B16 | S_AD[06] |
| C16 | S_AD[07] |
| B15 | S_AD[08] |
| C15 | S_AD[09] |
| B14 | S_AD[10] |
| C14 | S_AD[11] |
| D13 | S_AD[12] |
| A13 | S_AD[13] |
| B13 | S_AD[14] |
| C13 | S_AD[15] |
| A09 | S_AD[16] |
| C08 | S_AD[17] |
| B08 | S_AD[18] |
| A08 | S_AD[19] |
| B07 | S_AD[20] |
| D07 | S_AD[21] |
| A07 | S_AD[22] |
| A06 | S_AD[23] |
| B05 | S_AD[24] |
| C05 | S_AD[25] |
| B04 | S_AD[26] |
| A04 | S_AD[27] |
| C04 | S_AD[28] |
| B03 | S_AD[29] |
| A03 | S_AD[30] |
| C03 | S_AD[31] |
| M21 | S_AD[32] |
| M23 | S_AD[33] |
| M22 | S_AD[34] |
| L22 | S_AD[35] |
| L21 | S_AD[36] |
| L23 | S_AD[37] |
| K21 | S_AD[38] |
| K22 | S_AD[39] |
| K23 | S_AD[40] |
| J22 | S_AD[41] |
| J20 | S_AD[42] |
| J23 | S_AD[43] |
| H21 | S_AD[44] |
| H22 | S_AD[45] |
| H23 | S_AD[46] |
| G21 | S_AD[47] |
| G22 | S_AD[48] |
| G20 | S_AD[49] |
| F22 | S_AD[50] |
| F23 | S_AD[51] |
| F21 | S_AD[52] |
| E23 | S_AD[53] |
| E21 | S_AD[54] |
| D22 | S_AD[55] |
| E20 | S_AD[56] |
| D21 | S_AD[57] |
| C22 | S_AD[58] |
| C23 | S_AD[59] |
| C21 | S_AD[60] |
| D20 | S_AD[61] |
| A21 | S_AD[62] |
| C20 | S_AD[63] |
| A15 | S_CBE#[0] |
| C12 | S_CBE#[1] |
| D09 | S_CBE#[2] |
| C06 | S_CBE#[3] |
| D19 | S_CBE#[4] |
| A20 | S_CBE#[5] |
| C19 | S_CBE#[6] |
| A19 | S_CBE#[7] |
| K01 | S_CFN# |
| J04 | S_CLK |
| L02 | S_CLK_O[0] |
| L03 | S_CLK_O[1] |
| M03 | S_CLK_O[2] |
| M01 | S_CLK_O[3] |
| M02 | S_CLK_O[4] |
| N03 | S_CLK_O[5] |
| N01 | S_CLK_O[6] |
| P03 | S_CLK_O[7] |
| P02 | S_CLK_O[8] |
| P01 | S_CLK_O[9] |
| B10 | S_DEVSEL# |
| B09 | S_FRAME# |
| E02 | S_GNT#[0] |
| F03 | S_GNT#[1] |
| F01 | S_GNT#[2] |
| F02 | S_GNT#[3] |
| G01 | S_GNT#[4] |

| | |
|---|---|
| G04 | S_GNT#[5] |
| G02 | S_GNT#[6] |
| G03 | S_GNT#[7] |
| H01 | S_GNT#[8] |
| **E09** | **S_INTA#** |
| C09 | S_IRDY# |
| A11 | S_LOCK# |
| A14 | S_M66EN |
| B12 | S_PAR |
| N21 | S_PAR64 |
| C11 | S_PERR |
| **E08** | **S_PME#** |
| D04 | S_REQ#[0] |
| C01 | S_REQ#[1] |
| C02 | S_REQ#[2] |
| D03 | S_REQ#[3] |
| E04 | S_REQ#[4] |
| D01 | S_REQ#[5] |
| D02 | S_REQ#[6] |
| E03 | S_REQ#[7] |
| E01 | S_REQ#[8] |
| B19 | S_REQ64# |
| **H05** | **S_RSTIN#** |
| H02 | S_RSTOUT# |
| B11 | S_SERR# |
| C10 | S_STOP# |
| A10 | S_TRDY# |
| N22 | S_VIO |
| **E15** | **SCLKOFF** |
| **N19** | **SCLKSTB** |
| **B06** | **SIDSEL** |
| N20 | TCK |
| P23 | TDI |
| P22 | TDO |
| **E13** | **TEST#** |
| P21 | TMS |
| **H19** | **TRANS#** |
| N23 | TRST# |

| | |
|---|---|
| **J19** | **U_MODE** |
| A02 | VDD |
| B01 | VDD |
| B20 | VDD |
| B23 | VDD |
| D05 | VDD |
| D06 | VDD |
| D10 | VDD |
| D11 | VDD |
| D14 | VDD |
| D15 | VDD |
| D18 | VDD |
| **E14** | **VDD** |
| E22 | VDD |
| H04 | VDD |
| H20 | VDD |
| J01 | VDD |
| J03 | VDD |
| J21 | VDD |
| M04 | VDD |
| **M19** | **VDD** |
| M20 | VDD |
| N04 | VDD |
| **N05** | **VDD** |
| R01 | VDD |
| T01 | VDD |
| T04 | VDD |
| T20 | VDD |
| W03 | VDD |
| Y06 | VDD |
| Y10 | VDD |
| Y14 | VDD |
| Y18 | VDD |
| Y22 | VDD |
| AB19 | VDD |
| AB23 | VDD |
| AC02 | VDD |
| AC03 | VDD |

| | |
|---|---|
| AC08 | VDD |
| AC12 | VDD |
| AC16 | VDD |
| A01 | VSS |
| A05 | VSS |
| A12 | VSS |
| A16 | VSS |
| B02 | VSS |
| B21 | VSS |
| B22 | VSS |
| C07 | VSS |
| D08 | VSS |
| D12 | VSS |
| D16 | VSS |
| **E12** | **VSS** |
| **E16** | **VSS** |
| F04 | VSS |
| F20 | VSS |
| G23 | VSS |
| H03 | VSS |
| J02 | VSS |
| K04 | VSS |
| **K10** | **VSS** |
| **K11** | **VSS** |
| **K12** | **VSS** |
| **K13** | **VSS** |
| **K14** | **VSS** |
| K20 | VSS |
| **L10** | **VSS** |
| **L11** | **VSS** |
| **L12** | **VSS** |
| **L13** | **VSS** |
| **L14** | **VSS** |
| **L19** | **VSS** |
| L20 | VSS |
| **M10** | **VSS** |
| **M11** | **VSS** |
| **M12** | **VSS** |

| | |
|---|---|
| **M13** | **VSS** |
| **M14** | **VSS** |
| N02 | VSS |
| **N10** | **VSS** |
| **N11** | **VSS** |
| **N12** | **VSS** |
| **N13** | **VSS** |
| **N14** | **VSS** |
| P04 | VSS |
| **P10** | **VSS** |
| **P11** | **VSS** |
| **P12** | **VSS** |
| **P13** | **VSS** |
| **P14** | **VSS** |
| P20 | VSS |
| T02 | VSS |
| U21 | VSS |
| V04 | VSS |
| V20 | VSS |
| **W12** | **VSS** |
| Y08 | VSS |
| Y09 | VSS |
| Y12 | VSS |
| Y16 | VSS |
| AA02 | VSS |
| AB22 | VSS |
| AC01 | VSS |
| AC04 | VSS |
| AC13 | VSS |
| AC17 | VSS |
| AC20 | VSS |
| AC23 | VSS |

# 6 Configuration Registers

As a PCI bridge, PCI 6254 includes the standard Type-01h Configuration Space header defined in Bridge 1.1.

## *6.1 Configuration Space Map – Transparent Mode*

**Superscript legend:**
1 = Writable when Read Only Register Write Enable bit is set
2 = EEPROM loadable

| Bits 31-24 | Bits 23-16 | Bits 15-8 | Bits 7-0 | Address |
|---|---|---|---|---|
| [1,2] Device ID | | [1,2] Vendor ID | | 00h |
| Primary Status | | Primary Command | | 04h |
| [1,2] Class Code | | | Revision ID | 08h |
| [1,2] BIST | [1,2] Header Type | Primary Latency Timer | Cache Line Size | 0Ch |
| Reserved | | | | 10h – 17h |
| Secondary Latency Timer | Subordinate Bus Number | Secondary Bus Number | Primary Bus Number | 18h |
| Secondary Status | | I/O Limit | I/O Base | 1Ch |
| Memory Limit | | Memory Base | | 20h |
| Prefetchable Memory Limit | | Prefetchable Memory Base | | 24h |
| Prefetchable Memory Base Upper 32 Bits | | | | 28h |
| Prefetchable Memory Limit Upper 32 Bits | | | | 2Ch |
| I/O Limit Upper 16 Bits | | I/O Base Upper 16 Bits | | 30h |
| Reserved | | | ECP Pointer | 34h |
| Reserved | | | | 38h |
| Bridge Control | | Interrupt Pin | Reserved | 3Ch |
| Arbiter Control | | Diagnostic Control | Chip Control | 40h |
| [2] Misc Options | | [2] Timeout Control | [2] Primary Flow Through Control | 44h |
| [2] Secondary Incremental Prefetch Count | [2] Primary Incremental Prefetch Count | [2] Secondary Prefetch Line Count | [2] Primary Prefetch Line Count | 48h |
| Reserved | [2] Secondary Flow Through Control | [2] Secondary Maximum Prefetch Count | [2] Primary Maximum Prefetch Count | 4Ch |
| Reserved | Test register | Internal Arbiter Control | | 50h |
| EEPROM Data | | EEPROM Address | EEPROM control | 54h |
| Reserved | | | | 58h-60h |

| | | | | |
|---|---|---|---|---|
| **GPIO[3-0] Input Data** | **GPIO[3-0] Output Enable Control** | **GPIO[3-0] Output Data** | **P_SERR# event disable** | 64h |
| **Clkrun Register** | **P_SERR# status** | **Clock Control** | | 68h |
| **Private Memory Limit** | | **Private Memory Base** | | 6Ch |
| **Private Memory Base Upper 32 Bits** | | | | 70h |
| **Private Memory Limit Upper 32 Bits** | | | | 74h |
| Reserved | | | | 78h-98h |
| **GPIO[7-4] Input Dataport** | **GPIO[7-4] Output Enable** | **GPIO[7-4] Output Data** | **Hot Swap switch ROR control** | 9Ch |
| **GPIO[15-8] Input Dataport** | **GPIO[15-8] Output Enable** | **GPIO[15-8] Output Data** | **Pwrup Status** | A0h |
| Reserved | | | | AC-CCh |
| **Extended Register Index** | Reserved | Reserved | Reserved | D0h |
| **Extended Registers Dataport** | | | | D4h |
| Reserved | | | | D8h |
| **[1,2] Power Management Capabilities** | | **Next Item Ptr = E4** | **Capability ID = 01** | DCh |
| **[1,2] Power Management Data** | **PMCSR Bridge Support** | **[1, 2] Power Management CSR** | | E0h |
| Reserved | **HSCSR = 00** | **Next Item Ptr = E8** | **Capability ID = 06** | E4h |
| **VPD Register = 0000** | | **Next Item Ptr = 00** | **Capability ID = 03** | E8h |
| **VPD Data Register = 0000_0000** | | | | ECh |
| **Reserved** | | | | F0h-FCh |

## *6.2 Extended Register Map*

| 31-24 | 23-16 | 15-8 | 7-0 | INDEX |
|---|---|---|---|---|
| **32 Bit Sticky Register 0** | | | | 0h |
| **32 Bit Sticky Register 1** | | | | 1h |
| **32 Bit Sticky Register 2** | | | | 2h |
| **32 Bit Sticky Register 3** | | | | 3h |
| **32 Bit Sticky Register 4** | | | | 4h |
| **32 Bit Sticky Register 5** | | | | 5h |
| **32 Bit Sticky Register 6** | | | | 6h |
| **32 Bit Sticky Register 7** | | | | 7h |
| **Address translation control registers: see following section** | | | | 8-Fh |

## 6.2.1 Address Translation Register Map

In order to use the Address Translation functions, the address window set in the BAR registers must fall in the range specified in the PCI-to-PCI bridge standard configuration registers at 20h-2Ch for Memory Base and Memory Limit or Prefetchable Memory Base and Prefetchable Memory Limit.

Registers definitions are defined in the Non-Transparent Mode Chapter. Description of the mechanism is in the **Address Decoding Chapter - Address Translation Section**.

These registers are normally shadowed and can only be accessed by setting the Downstream Translation BAR bit at register 9ch.

| | |
|---|---|
| **Downstream Memory 1 BAR** | 14h |
| **Downstream Memory 2 BAR or Downstream Memory 1 BAR Upper 32 bits** | 18h |

These registers are normally shadowed and can only be accessed by setting the Upstream Translation BAR bit at register 9ch.

| | |
|---|---|
| **Upstream I/O or Memory 0 BAR** | 10h |
| **Upstream Memory 1 BAR** | 14h |
| **Upstream Memory 2 BAR or Downstream Memory 1 BAR Upper 32 bits** | 18h |

**Extended registers**

| 31-24 | 23-16 | 15-8 | 7-0 | Extended Register INDEX |
|---|---|---|---|---|
| [2] **Upstream BAR 0 Translation Address** | | | | 8h |
| [2] **Upstream BAR 1 Translation Address** | | | | 9h |
| [2] **Upstream BAR 2 or Upstream BAR1 Upper 32 bits Translation Address** | | | | Ah |
| [2] **Upstream Translation Enable** | [2] **Upstream BAR 2 Translation Mask** | [2] **Upstream BAR 1 Translation Mask** | [2] **Upstream BAR 0 Translation Mask** | Bh |
| **Reserved** | | | | Ch |
| [2] **Downstream BAR 1 Translation Address** | | | | Dh |
| [2] **Downstream BAR 2 or Downstream BAR1 Upper 32 bits Translation Address** | | | | Eh |
| [2] **Downstream Translation Enable** | [2] **Downstream BAR 2 Translation Mask** | [2] **Downstream BAR 1 Translation Mask** | [2] **Downstream BAR 0 Translation Mask** | Fh |

## 6.3 Transparent Mode Configuration Register Description

### 6.3.1 PCI Standard Configuration Registers

**Vendor ID Register (Read Only) - Offset 0h**
Defaults to 3388(h).

**Device ID Register (Read Only) - Offset 2h**
Defaults to 0020(h) for Transparent Mode, 21h for Non-Transparent Mode.

(Note: R/W - Read/Write, R/O - Read Only, R/WC - Read/ Write 1 to clear)

**Primary Command Register (Read/Write) - Offset 4h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 0 | I/O Space Enable | R/W | Controls the bridge's response to I/O accesses on the primary interface.<br>0=ignore I/O transaction<br>1=enable response to I/O transaction<br>Reset to 0. |
| 1 | Memory Space Enable | R/W | Controls the bridge's response to memory accesses on the primary interface.<br>0=ignore all memory transaction<br>1=enable response to memory transaction<br>Reset to 0. |
| 2 | Bus Master Enable | R/W | Controls the bridge's ability to operate as a master on the primary interface.<br>0=do not initiate transaction on the primary interface and disable response to memory or I/O transactions on secondary interface<br>1=enable the bridge to operate as a master on the primary interface<br>Reset to 0. |
| 3 | Special Cycle Enable | R/O | No special cycle implementation (set to '0'). |
| 4 | Memory Write and Invalidate Enable | R/O | Memory write and invalidate not supported (set to '0'). |
| 5 | VGA Palette Snoop Enable | R/W | Controls the bridge's response to VGA compatible palette accesses.<br>0=ignore VGA palette accesses on the primary interface<br>1=enable response to VGA palette writes on the primary interface (I/O address AD[9:0]=3C6h, 3C8h and 3C9h)<br>Reset to 0. |
| 6 | Parity Error Enable | R/W | Controls the bridge's response to parity errors.<br>0=ignore any parity errors<br>1=normal parity checking performed<br>Reset to 0. |
| 7 | Wait Cycle Control | R/W | PCI 6254 performs address / data stepping (reset to '1'). |

| | | | |
|---|---|---|---|
| 8 | P_SERR# Enable | R/W | Controls the enable for the P_SERR# pin. |
| 9 | Fast Back to Back Enable | R/W | Controls the bridge's ability to generate fast back-to-back transactions to different devices on the primary interface.<br>0=no fast back to back transaction<br>1=reserved. PCI 6254 does not generate Fast Back to Back cycle.<br>Reset to 0. |
| 10-15 | Reserved | R/O | Reserved. Reset to 0. |

**Primary Status Register(Read/Write) – Offset 6h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 0-3 | Reserved | R/O | Reserved (set to '0's). |
| 4 | ECP | R/O | Enhanced Capabilities port. Reads as 1 to indicate PCI 6254 supports an enhanced capabilities list. |
| 5 | 66MHz | R/O | Reflects the state of CFG66 input pin. 1 = PCI 6254 is 66Mhz Capable. |
| 6 | UDF | R/O | No User-Definable Features (set to '0'). |
| 7 | Fast Back to Back Capable | R/O | Fast back-to-back write capable on primary side (set to '1'). |
| 8 | Data Parity Error Detected | R/WC | It is set when the following conditions are met:<br>1. P_PERR# is asserted<br>Bit 6 of Command Register is set<br>Reset to 0. |
| 9-10 | DEVSEL timing | R/O | DEVSEL# timing. Reads as 01b to indicate PCI 6254 responds no slower than with medium timing |
| 11 | Signaled Target Abort | R/WC | Should be set (by a target device) whenever a Target Abort cycle occurs. Reset to 0. |
| 12 | Received Target Abort | R/WC | Set to '1' (by a master device) when transactions are terminated with Target Abort. Reset to 0. |
| 13 | Received Master Abort | R/WC | Set to '1' (by a master) when transactions are terminated with Master Abort. Reset to 0. |
| 14 | Signaled System Error | R/WC | Should be set whenever P_SERR# is asserted. Reset to 0. |
| 15 | Detected Parity Error | R/WC | Should be set whenever a parity error is detected regardless of the state of the bit 6 of command register. Reset to 0. |

### Revision ID Register (Read Only) – Offset 8h
Defaults to 04h.

### Class Code Register (Read Only) – Offset 9h
Defaults to 060400h for Transparent Mode, 068000 for Non-Transparent Mode.

### Cache Line Size Register (Read/Write) – Offset 0Ch
This register is used when terminating memory write and invalidate transactions. Memory read prefetching is controlled by the prefetch count registers.

Only cache line sizes (in units of 32-bits words) which are power of two are valid. Resets to 0.

### Primary Latency Timer Register (Read/Write) – Offset 0Dh
This register sets the value for Master Latency Timer which starts counting when the master asserts FRAME#. Reset to 0.

### Header Type Register (Read Only) – Offset 0Eh
Defaults to 1 in Transparent Mode, 0 in Non-Transparent Mode.

### BIST Register (Read Only) – Offset 0Fh
This register can be written to by enabling the ROR Write Enable bit at register 9Ch bit 7. Reset to 0.

### Primary Bus Number Register (Read/Write) – Offset 18h
Programmed with the number of the PCI bus to which the primary bridge interface is connected. This value is set with configuration software. Reset to 0.

### Secondary Bus Number Register (Read/Write) – Offset 19h
Programmed with the number of the PCI bus to which the secondary bridge interface is connected. This value is set with configuration software. Reset to 0.

### Subordinate Bus Number Register (Read/Write) –Offset 1Ah
Programmed with the number of the PCI bus with the highest number that is subordinate to the bridge. This value is set with configuration software. Reset to 0.

### Secondary Latency Timer (Read/Write) – Offset 1Bh
This register is programmed in units of PCI bus clocks. Reset to 0. The latency timer checks for master accesses on the secondary side that remain unclaimed by any target.

### I/O Base Register (Read/Write) – Offset 1Ch
This register defines the bottom address of the I/O address range for the bridge. The upper four bits define the bottom address range used by the chip to determine when to forward I/O transactions from one interface to the other. These 4 bits correspond to address bits <15:12> and are writeable. The upper 16 bits corresponding to address bits <31:16> are defined in the I/O base address upper 16 bits register. The address bits <11:0> are assumed to be 000h. The lower four bits (3:0) of this register set to ‘0001’ (read-only) to indicate 32-bit I/O addressing. Reset to 0.

### I/O Limit Register (Read/Write) – Offset 1Dh
This register defines the top address of the I/O address range for the bridge. The upper four bits define the top address range used by the chip to determine when to forward I/O transactions from one interface to the other. These 4 bits correspond to address bits <15:12> and are writeable. The upper 16 bits corresponding to address bits <31:16> are defined in the I/O limit address upper 16 bits register. The address bits <11:0> are assumed to be FFFh. The lower four bits (3:0) of this register set to ‘0001’ (read-only) to indicate 32-bit I/O addressing. Reset to 0.

**Secondary Status Register (Read/Write) – Offset 1Eh**

| Bit | Function | Type | Description |
|---|---|---|---|
| 0-4 | reserved | R/O | Reserved (set to '0's). |
| 5 | 66MHz | R/O | Defaults to 1. PCI 6254 is 66Mhz Capable. |
| 6 | UDF | R/O | No User-Definable Features (set to '0'). |
| 7 | Fast Back to Back Capable | R/O | Fast back-to-back write capable on secondary port (set to '1'). |
| 8 | Data Parity Error Detected | R/WC | It is set when the following conditions are met:<br>1. SPERR# is asserted<br>2. Bit 6 of Command Register is set<br>Reset to 0. |
| 9-10 | DEVSEL timing | R/O | Medium DEVSEL# timing (set to '01') |
| 11 | Signaled Target Abort | R/WC | Should be set (by a target device) whenever a Target Abort cycle occurs. Should be '0' after reset.<br>Reset to 0. |
| 12 | Received Target Abort | R/WC | Set to '1' (by a master device) when transactions are terminated with Target Abort.<br>Reset to 0. |
| 13 | Received Master Abort | R/WC | Set to '1' (by a master) when transactions are terminated with Master Abort.<br>Reset to 0. |
| 14 | Received System Error | R/WC | Should be set whenever SSERR# is detected. Should be a '0 after reset.<br>Reset to 0. |
| 15 | Detected Parity Error | R/WC | Should be set whenever a parity error is detected regardless of the state of the bit 6 of command register.<br>Reset to 0. |

### Memory Base Register (Read/Write) – Offset 20h

This register defines the base address of the memory-mapped address range for forwarding the cycle through the bridge. The upper twelve bits corresponding to address bits <31:20> are writeable. The lower 20 address bits (19:0) are assumed to be 00000h. The 12 bits are reset to 0. The lower 4 bits are read only and set to 0.

### Memory Limit Register (Read/Write) – Offset 22h

This register defines the upper limit address of the memory-mapped address range for forwarding the cycle through the bridge. The upper twelve bits corresponding to address bits <31:20> are writeable. The 12 bits are reset to 0. The lower 4 bits are read only and are set to 0. The lower 20 address bits (19:0) are assumed to be FFFFFh. Reset to 0.

### Prefetchable Memory Base Register (Read/Write) - Offset 24h

This register defines the base address of the prefetchable memory-mapped address range for forwarding the cycle through the bridge. The upper twelve bits corresponding to address bits <31:20> are writeable. The 12 bits are reset to 0. The lower 4 bits are read only and are set to 0. The lower 20 address bits (19:0) are assumed to be 00000h. Reset to 0.

### Prefetchable Memory Limit Register (Read/Write) – Offset 26h

This register defines the upper limit address of the memory-mapped address range for forwarding the cycle through the bridge. The upper twelve bits correspond to address bits <31:20> are writeable. The 12 bits are reset to 0. The lower 4 bits are read only and are set to 0. The lower 20 address bits (19:0) are assumed to be FFFFFh. Reset to 0.

### Prefetchable Memory Base Register Upper 32 Bits (Read/Write) – Offset 28h

This register defines the upper 32 bit <63:32> memory base address of the prefetchable memory-mapped address for forwarding the cycle through the bridge. Reset to 0.

### Prefetchable Memory Limit Register Upper 32 Bits (Read/Write) – Offset 2Ch

This register defines the upper 32 bit <63:32> memory limit address of the prefetchable memory-mapped address for forwarding the cycle through the bridge. Reset to 0.

### I/O Base Address Upper 16 Bits Register (Read/Write) – Offset 30h

This register defines the upper 16 bits of a 32-bit base I/O address range used for forwarding the cycle through the bridge. Reset to 0.

### I/O Limit Address Upper 16 Bits Register (Read/Write) – Offset 32h

This register defines the upper 16 bits of a 32-bit limit I/O address range used for forwarding the cycle through the bridge. Reset to 0.

### ECP Pointer (Read/Only) – Offset 34h

| Bit | Function | Type | Description |
|---|---|---|---|
| 7-0 | ECP Pointer | R/O | Enhanced capabilities port offset pointer. This register reads as DCh to indicate the offset of the power management registers. |

### Interrupt Pin Register (Read Only) – Offset 3Dh

Reads as 0 to indicate that PCI 6254 does not use any interrupt pin.

### Bridge Control Register (Read/Write) – Offset 3Eh

| Bit | Function | Type | Description |
|---|---|---|---|
| 0 | Parity Error Response Enable | R/W | Controls the bridge's response to parity errors on the secondary interface.<br>0=ignore address and data parity errors on the secondary interface<br>1=enable parity error reporting and detection on the secondary interface<br>Reset to 0. |
| 1 | S_SERR# Enable | R/W | Controls the forwarding of S_SERR# to the primary interface.<br>0=disable the forwarding S_SERR# to primary<br>1=enable the forwarding of S_SERR# to primary<br>Reset to 0. |
| 2 | ISA Enable | R/W | Controls the bridge's response to ISA I/O addresses, which is limited to the first 64K.<br>0=forward all I/O addresses in the range defined by the I/O Base and I/O Limit registers<br>1=block forwarding of ISA I/O addresses in the range defined by the I/O Base and I/O Limit registers that are in the first 64K of I/O space that address the last 768 bytes in each 1Kbytes block. Secondary I/O transactions are forwarded upstream if the address falls within the last 768 bytes in each 1Kbytes block<br><br>There is an **ISA Enable Control bit Write Protect** mechanism control by EEPROM. When the **ISA Enable Control bit Write Protect** bit is set in the EEPROM and EEPROM initialization is enabled, PCI 6254 will change this bit to read only and ISA Enable feature will not be available.<br><br>Reset to 0. |
| 3 | VGA Enable | R/W | Controls the bridge's response to VGA compatible addresses.<br>0=do not forward VGA compatible memory and I/O addresses from primary to secondary<br>1=forward VGA compatible memory and I/O address from primary to secondary regardless of other settings<br>Reset to 0. |
| 4 | reserved | R/O | Reserved (set to 0). |
| 5 | Master Abort | R/W | Controls the bridge behavior in responding to master aborts on |

| | | | |
|---|---|---|---|
| | Mode | | secondary interface<br>0=do not report master aborts (return ffff_ffffh on reads and discards data on writes)<br>1=report master aborts by signaling target abort<br>Reset to 0.<br>Note: During lock cycles, PCI 6254 ignores this bit, and always completes the cycle as a target abort. |
| 6 | Secondary Reset | R/W | Forces the assertion of S_RSTOUT# signal pin on the secondary interface.<br>0=do not force the assertion of S_RSTOUT# pin<br>1=force the assertion of S_RSTOUT# pin<br>Reset to 0. |
| 7 | Fast Back to Back Enable | R/W | Controls the bridge's ability to generate fast back-to-back transactions to different devices on the secondary interface.<br>0 = no fast back to back transaction<br>1= reserved. PCI 6254 does not generate Fast Back to Back cycle.<br>Reset to 0. |
| 8 | Primary Master Timeout | R/W | Sets the maximum number of PCI clock for an initiator on the primary bus to repeat the delayed transaction request.<br>0=Timeout after $2^{15}$ PCI clocks<br>1=Timeout after $2^{10}$ PCI clocks<br>Reset to 0. |
| 9 | Secondary Master Timeout | R/W | Sets the maximum number of PCI clock for an initiator on the secondary bus to repeat the delayed transaction request.<br>0=Timeout after $2^{15}$ PCI clocks<br>1=Timeout after $2^{10}$ PCI clocks<br>Reset to 0. |
| 10 | Master Timeout Status | R/WC | Set to '1' when either primary master timeout or secondary master timeout.<br>Reset to 0. |
| 11 | Master Timeout P_SERR# enable | R/W | Enable P_SERR# assertion during master timeout.<br>0=P_SERR# not asserted on master timeout<br>1=P_SERR# asserted on either primary or secondary master timeout.<br>Reset to 0. |
| 15-12 | reserved | R/O | Reserved |

**Chip Control Register (Read/Write) – Offset 40h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 0 | Reserved | R/O | |
| 1 | Memory write disconnect control | R/W | Controls when the chip as a target disconnects memory transactions. When 0, disconnects on queue full or on a 4KB boundary. When 1, disconnects on a cache line boundary, as well as when the queue fills or on a 4 KB boundary. Reset value is 0. |
| 2 | Private or Cross Bridge Memory Enable | R/W | **(Transparent Mode)** 1 = Enable Private Memory block reserved only for Secondary Memory Space. The memory space can be programmed using the Private Memory Base/Limit registers. If Limit is smaller than Base, the Private memory space is disabled. Primary port cannot access this memory space through the bridge and the Secondary port will NOT respond to any memory cycles addressing this private memory space.<br><br>(In addition in Rev AA, the Cross-Bridge Memory Window, default at 0-16M space and programmable later by software, is also treated as a private memory block in Transparent Mode.)<br><br>**(Non-Transparent mode) Cross-Bridge Memory Window Enable:** When this bit is "1", PCI 6254 will automatically claim 16M of memory space. This allows the boot up of the Low priority Boot Port to move forward without waiting for the Priority Boot port to program the corresponding Memory BAR registers. If this bit is "1", the Primary or Secondary PORT_READY mechanism will NOT be relevant and access to BAR registers will not be retried. Although the default claims 16M, the BAR registers can be changed by EEPROM or software to change the window size.<br><br>In both Transparent and Non-Transparent modes, this bit resets to the value as presented at the XB_MEM input pin. After reset, this bit can be reprogrammed. |
| 3 | Reserved | R/O | |
| 4 | Secondary bus prefetch disable | R/W | Controls PCI 6254's ability to prefetch during upstream memory read transactions. When 0 the chip prefetches and does not forward byte enable bits during memory read transactions. When 1, PCI 6254 requests only one Dword from the target during memory read transactions and forwards read enable bits. PCI 6254 returns a target disconnect to the requesting master on the first data transfer. Memory read line and memory read multiple transactions are still prefetchable. Reset to 0. |
| 5 | Live Insertion mode | R/W | Enables hardware control of transaction forwarding in the PCI 6254. When 0, Pin GPIO[3] has no effect on the I/O, memory, and master enable bits.<br>When 1, if GPIO[3] is set as input, and GPIO[3] is driven high, I/O, memory and master enable bits are disabled. |
| 7:6 | Reserved | R/O | Reserved **(Set to 0).** |

**Diagnostic Control Register (Read/Write) – Offset 41h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 0 | Chip reset | R/W | Chip and Secondary bus reset. Setting this bit will do a chip reset, without asserting S_RSTOUT# and forcing secondary reset bit in bridge control register to be set. After resetting all bits except for the secondary reset bit in bridge control register, this bit will be cleared. Write 0 has no effect. |
| 2:1 | Test mode | R/W | Reserved |
| 3 | Reserved | R/W | Reserved (Set to 0). |
| 7:4 | Reserved | R/O | Reserved (Set to 0). |

**Arbiter Control Register (Read/Write) – Offset 42h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 8-0 | Arbiter Control | R/W | Each bit controls whether a secondary bus master is assigned to the high priority group or the low priority group. Bits <8:0> correspond to request inputs S_REQ#[8:0], respectively. Reset value is 0. |
| 9 | PCI 6254 priority | R/W | Defines whether the secondary port of PCI 6254 is in high priority group or the low priority group. Reset to 1.<br>0=low priority group<br>**1=high priority group**. |
| 11:10 | reserved | R/O | Reserved **(set to '0's)** |
| 12 | Primary Port Ordering Rule | R/W | Reserved **(default and should be set to '0')** |
| 13 | Secondary Port Ordering Rule | R/W | Reserved **(default and should be set to '0')** |
| 14 | Upstream 64 bit Cycle Control | R/W | Reserved **(default and should be set to '0')** |
| 15 | Downstream 64 bit Cycle Control | R/W | Reserved **(default and should be set to '0')** |

**Primary Flow Through Control Register - Offset 44h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 2-0 | Primary posted write completion wait count | R/W | Maximum number of clocks that PCI 6254 will wait for posted write data from initiator if delivering write data in flow through mode and internal post write queues are almost empty. If the count is exceeded without any additional data from the initiator, the cycle to target will be terminated to be completed later.<br>000 : PCI 6254 will terminate cycle if there is only 1 data entry left in the internal write queue.<br>001 : PCI 6254 will deassert IRDY#, and wait 1 clock for data before terminating cycle.<br>…<br>111 : PCI 6254 will wait 7 clocks for source data. |
| 3 | Reserved | R/O | Reserved. Returns 0 when read |
| 6-4 | Primary delayed read completion wait count | | Maximum number of clocks that PCI 6254 will wait for delayed read data from target if returning read data in flow through mode and internal delayed read queue is almost full. If the count is exceeded without any additional space in the queue, the cycle to target will be terminated, and completed when initiator retries the rest of the cycle.<br>000 : PCI 6254 will terminate cycle if only 1 data entry is left in the read queue.<br>001 : PCI 6254 will deassert TRDY#, and wait 1 clock for data before terminating cycle.<br>…<br>111 : PCI 6254 will wait 7 clocks for source data. |
| 7 | Reserved | R/O | Reserved. Returns 0 when read. |

**Timeout Control Register - Offset 45h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 2-0 | Maximum retry counter control | R/W | Controls maximum number of times that PCI 6254 will retry a cycle before signaling a timeout. This timeout applies to Read/Write retries and can be enabled to trigger SERR# on the primary or secondary port depending the SERR# events that are enabled.<br>Maximum number of retries to timeout =<br>0000 : $2^{24}$<br>0001 : $2^{18}$<br>0010 : $2^{12}$<br>0011 : $2^{6}$<br>0111 : $2^{0}$<br>Reset to 0. |
| 3 | Reserved | R/O | |
| 5:4 | Primary master timeout divider | R/W | Provides an additional option for the primary master timeout. Timeout counter can optionally be divided by 256, in addition to its original setting in the Bridge Control register.<br>Original setting is 32K by default and programmable to 1K.<br>11 : timeout counter = Primary master timeout / 256<br>10 : timeout counter = Primary master timeout / 16<br>01 : timeout counter = Primary master timeout / 8<br>00: counter = Primary master timeout / 1<br>defaults to 0 |
| 7:6 | Secondary master timeout divider | R/W | Provides an additional option for the secondary master timeout. Timeout counter can optionally be divided by 256, in addition to its original setting in the Bridge Control register.<br>Original setting is 32K by default and programmable to 1K.<br>11 : timeout counter = Primary master timeout / 256<br>10 : timeout counter = Primary master timeout / 16<br>01 : timeout counter = Primary master timeout / 8<br>00: counter = Primary master timeout / 1<br>defaults to 0 |

**Miscellaneous Options - Offset 46h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 0 | Write completion wait for PERR# | R/W | If 1, PCI 6254 will always wait for PERR# status of the target before completing a delayed write transaction to the initiator.<br>Defaults to 0 |
| 1 | Read completion wait for PAR | R/W | If 1, PCI 6254 will always wait for PAR status of the target before completing a delayed read transaction to the initiator.<br>Defaults to 0 |
| 2 | DTR out of order enable | R/W | If 1, PCI 6254 may return delayed read transactions in a different order than requested. Otherwise, delayed read transactions are returned in the same order as requested<br>Defaults to 0 |
| 3 | Generate parity enable | R/W | If 1, PCI 6254 as a master will generate the PAR and PAR64 to cycles going across the bridge, otherwise, PCI 6254 passes along the PAR/PAR64 of the cycle as stored in the internal buffers.<br>Defaults to 0 |
| 6-4 | Address step control | R/W | During configuration type 0 cycles, PCI 6254 will drive the address for the number of clocks specified in this register before asserting FRAME#.<br>000 : PCI 6254 will assert FRAME# at the same time as the address.<br>001 : PCI 6254 will assert FRAME# 1 clock after it drives the address on the bus.<br>…<br>111 : PCI 6254 will assert FRAME# 7 clocks after it drives the address on the bus. |
| 8-7 | Reserved | R/W | Defaults to 0 |

| | | | |
|---|---|---|---|
| 9 | Prefetch early termination | R/W | If 1, PCI 6254 will terminate prefetching at the current calculated count if flowthrough is not yet achieved, and another prefetchable read cycle is accepted by the PCI 6254.<br>If 0, PCI 6254 will always finish prefetching as programmed at the prefetch count registers, regardless of any other outstanding prefetchable reads in the transaction queue. |
| 10 | Read minimum enable | R/W | If 1, PCI 6254 will only initiate read cycles if there is available space in the FIFO as specified by the prefetch count registers. |
| 15,11 | Force 64 Bit Control | R/W | If set, 32-bit prefetchable reads or 32-bit posted memory write cycles on one side will be converted to 64-bit cycles on completion to target side if target supports 64-bit transfers. If set to 0, cycles are not converted.<br>When combined with the control of bit 15 of this register, the following control is provided: (Bit 15 is set to 0 for rev AA and cannot be changed)<br>Bit 15, 11<br>0, 0 Disable (Default)<br>0, 1 Convert to 64 bit command onto both ports<br>1, 0 Convert to 64 bit command onto Secondary Port<br>1, 1 Convert to 64 bit command onto Primary Port<br>Starting address for all cycles using this feature should be on the qword boundary.<br>Defaults to 0 |
| 12 | Memory write and invalidate control | R/W | If 1, PCI 6254 will pass memory write and invalidate commands if there is at least 1 cache line of FIFO space available, otherwise it will complete as a memory write cycle.<br>If 0, PCI 6254 will retry memory write and invalidate commands if there is no space for 1 cacheline of data in the internal queues.<br>Defaults to 0 |
| 13 | Primary Lock Enable | R/W | If 1, PCI 6254 will follow the LOCK protocol on the primary interface. Otherwise, LOCK is ignored.<br>Defaults to 0 (Rev AA defaults to 1) |
| 14 | Secondary Lock Enable | R/W | If 1, PCI 6254 will follow the LOCK protocol on the secondary interface. Otherwise, LOCK is ignored.<br>Defaults to 0 |
| 15,11 | Force 64 bit Control | R/W | See description in Bit 11 section. |

## 6.3.2 Prefetch Control Registers

Registers 44h, 48h – 4Dh are the prefetch control registers, and are used to fine-tune memory read prefetch behavior of the PCI 6254. Detailed descriptions of these registers can be found in Chapter 18 Flow Through Optimization.

**Primary Initial Prefetch Count - Offset 48h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 5-0 | Primary initial prefetch count | R/W | Controls initial prefetch count on the Primary bus during reads to prefetchable memory space. This register value should be a power of 2 (only one bit should be set to 1 at any time). Value is number of double words. Bit 0 is read only and is always 0.<br><br>Defaults to 10h. |
| 7-6 | Reserved | R/O | Reserved. Returns 0 when read. |

**Secondary Initial Prefetch Count - Offset 49h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 5-0 | Secondary initial prefetch count | R/W | Controls initial prefetch count on the Secondary bus during reads to prefetchable memory space. This register value should be a power of 2 (only one bit should be set to 1 at any time). Value is number of double words. Bit 0 is read only and is always 0.<br><br>Defaults to 10h. |
| 7-6 | Reserved | R/O | Reserved. Returns 0 when read. |

**Primary Incremental Prefetch Count - Offset 4Ah**

| Bit | Function | Type | Description |
|---|---|---|---|
| 5-0 | Primary incremental prefetch count | R/W | Controls incremental read prefetch count. When an entry's remaining prefetch Dword count falls below this value, the bridge will prefetch an additional "Primary incremental prefetch count" Dwords. This register value should be a power of 2 (only one bit should be set to 1 at any time). Value is number of double words. Bit 0 is read only and is always 0.<br><br>This register value must not exceed half the value programmed in the Primary Maximum Prefetch Count register. Otherwise, no incremental prefetch will be performed.<br><br>Defaults to 10h. |
| 7-6 | Reserved | R/O | Reserved. Returns 0 when read. |

**Secondary Incremental Prefetch Count - Offset 4Bh**

| Bit | Function | Type | Description |
|---|---|---|---|
| 5-0 | Secondary incremental prefetch count | R/W | This controls incremental read prefetch count. When an entry's remaining prefetch Dword count falls below this value, the bridge will prefetch an additional "Secondary incremental prefetch count" Dwords. This register value should be a power of 2 (only one bit should be set to 1 at any time). Value is number of double words. Bit 0 is read only and is always 0.<br><br>This register value must not exceed half the value programmed in the Secondary Maximum Prefetch Count register. Otherwise, no incremental prefetch will be performed.<br><br>Defaults to 10h |
| 7-6 | Reserved | R/O | Reserved. Returns 0 when read |

**Primary Maximum Prefetch Count - Offset 4Ch**

| Bit | Function | Type | Description |
|---|---|---|---|
| 5-0 | Primary maximum prefetch count | R/W | This value limits the cumulative maximum count of prefetchable Dwords that are allocated to one entry on the primary when flow through for that entry was not achieved. This register value should be an even number. Bit 0 is read only and is always 0.<br><br>Exception: 0h = 256 bytes = maximum programmable count<br><br>Defaults to 20h |
| 7-6 | Reserved | R/O | Reserved. Returns 0 when read |

**Secondary Maximum Prefetch Count - Offset 4Dh**

| Bit | Function | Type | Description |
|---|---|---|---|
| 5-0 | Secondary maximum prefetch count | R/W | Register limits the cumulative maximum count of prefetchable Dwords that are allocated to one entry on the secondary when flow through for that entry was not achieved. This register value should be an even number. Bit 0 is read only and is always 0.<br><br>Exception: 0h = 256 bytes = maximum programmable count<br><br>Defaults to 20h |
| 7-6 | Reserved | R/O | Reserved. Returns 0 when read |

**Secondary Flow Through Control Register - Offset 4Eh**

| Bit | Function | Type | Description |
|---|---|---|---|
| 2-0 | Secondary posted write completion wait count | R/W | Maximum number of clocks that PCI 6254 will wait for posted write data from initiator if delivering write data in flow through mode and internal post write queues are almost empty. If the count is exceeded without any additional data from the initiator, the cycle to target will be terminated, to be completed later.<br>000 : PCI 6254 will terminate cycle if there is only 1 data entry left in the internal write queue.<br>001 : PCI 6254 will deassert IRDY#, and wait 1 clock for data before terminating cycle.<br>…<br>111 : PCI 6254 will wait 7 clocks for source data. |
| 3 | Reserved | R/O | Reserved. Returns 0 when read |
| 6-4 | Secondary delayed read completion wait count | | Maximum number of clocks that PCI 6254 will wait for delayed read data from target if returning read data in flow through mode and internal delayed read queue is almost full. If the count is exceeded without any additional space in the queue, the cycle to target will be terminated, and completed when initiator retries the rest of the cycle.<br>000 : PCI 6254 will terminate cycle if only 1 data entry is left in the read queue.<br>001 : PCI 6254 will deassert TRDY#, and wait 1 clock for data before terminating cycle.<br>…<br>111 : PCI 6254 will wait 7 clocks for source data. |
| 7 | Reserved | R/O | Reserved. Returns 0 when read. |

**Internal Arbiter Control Register - Offset 50h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 0 | Low priority group fixed arbitration | R/W | If 1, the low priority group uses the fixed priority arbitration scheme, otherwise a rotating priority arbitration scheme is used<br>Defaults to 0 |
| 1 | Low priority group arbitration order | R/W | This bit is only valid when the low priority arbitration group is set to a fixed arbitration scheme. If 1, priority decreases in ascending numbers of the master, for example master #4 will have higher priority than master #3. If 0, the reverse is true. This order is relative to the master with the highest priority for this group, as specified in bits 7-4 of this register.<br>Defaults to 0 |
| 2 | High priority group fixed arbitration | R/W | If 1, the high priority group uses the fixed priority arbitration scheme, otherwise a rotating priority arbitration scheme is used<br>Defaults to 0 |
| 3 | High priority group arbitration order | R/W | This bit is only valid when the high priority arbitration group is set to a fixed arbitration scheme. If 1, priority decreases in ascending numbers of the master, for example master #4 will have higher priority than master #3. If 0, the reverse is true. This order is relative to the master with the highest priority for this group, as specified in bits 11-8 of this register.<br>Defaults to 0 |
| 7-4 | Highest priority master in low priority group | R/W | Controls which master in the low priority group has the highest priority. It is valid only if the group uses the fixed arbitration scheme.<br>0000 : master#0 has highest priority<br>0001 :<br>…<br>1001 : PCI 6254 has highest priority<br>1010-1111 : Reserved<br>Defaults to 0 |

| | | | |
|---|---|---|---|
| 11-8 | Highest priority master in high priority group | R/W | Controls which master in the high priority group has the highest priority. It is valid only if the group uses the fixed arbitration scheme.<br>0000 : master#0 has highest priority<br>0001 :<br>…<br>1001 : PCI 6254 has highest priority<br>1010-1111 : Reserved<br>Defaults to 0 |
| 12-15 | Bus Parking Control | R/W | Controls bus grant behavior during idle.<br>0000 : Last master granted is parked<br>0001 : Master #0 is parked<br>…<br>1001 : Master #8 is parked<br>1010 : PCI 6254 is parked<br>other : grant is deasserted<br>Defaults to 0 |

**PCI 6254 Test Register – Offset 52h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 0 | EEPROM Autoload control | R/W | If 1, disables EEPROM autoload.<br>**This is a testing feature only. In order to stop EEPROM load in Transparent Mode, 1 must be written into this register within 1200 clocks after P_RSTIN# goes HIGH. In Non-Transparent Mode, 1 must be written into this register within 1200 clocks after PWRGD goes HIGH.** |
| 1 | Fast EEPROM Autoload | R/W | If 1, speeds up EEPROM autoload by 32 times.<br>**This is a testing feature only. In order to enable Fast EEPROM load in Transparent Mode, 1 must be written into this register within 1200 clocks after P_RSTIN# goes HIGH. In Non-Transparent Mode, 1 must be written into this register within 1200 clocks after PWRGD goes HIGH.** |
| 2 | EEPROM autoload status | R/O | Status of EEPROM autoload. |
| 3 | Reserved | R/O | Reserved |
| 4 | 64EN# | R/O | Reflects the 64EN# pin status |
| 5 | S_CFN# | R/O | Reflects the S_CFN# pin status |
| 6 | TRANS# | R/O | Reflects the TRANS# pin status |
| 7 | U_MODE | R/O | Reflects the U_MODE pin status |

**EEPROM Control - Offset 54h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 0 | Start | R/W | Starts the EEPROM read or write cycle. |
| 1 | EEPROM command | R/W | Controls the command sent to the EEPROM<br>1 : write<br>0 : read |
| 2 | EEPROM Error | R/O | This bit is set to 1 if EEPROM ack was not received during EEPROM cycle. |
| 3 | EEPROM autoload successful | R/O | This bit is set to 1 if EEPROM autoload occurred successfully after reset, and some configuration registers were loaded with values programmed in the EEPROM. If zero, EEPROM autoload was unsuccessful or was disabled. |
| 5-4 | Reserved | R/O | Reserved. Returns '0' when read. |
| 7-6 | EEPROM clock rate | R/W | Controls frequency of EEPROM clock. EEPROM clock is derived from the primary PCI clock.<br>00 = PCLK/1024<br>01 = PCLK/512<br>10 = PCLK/256<br>11 = PCLK/32 (for test mode use)<br>Defaults to 10 (PCI 6254 Rev AA defaults to 00). |

**EEPROM Address - Offset 55h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 0 | Reserved | R/O | Starts the EEPROM read or write cycle. |
| 7-1 | EEPROM address | R/W | Word address for EEPROM cycle. |

**EEPROM Control - Offset 56h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 15-0 | EEPROM Data | R/W | Contains data to be written to the EEPROM. During reads, this register contains data received from the EEPROM after a read cycle has completed. |

**P_SERR# Event Disable Register - Offset 64h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 0 | Reserved | R/O | Reserved. Returns 0 when read |
| 1 | Posted write parity error | R/W | Controls ability of PCI 6254 to assert P_SERR# when a data parity error is detected on the target bus during a posted write transaction. P_SERR# is asserted if this event occurs when this bit is 0 and SERR# enable bit in the command register is set.<br>Reset value is 0. |
| 2 | Posted Memory write nondelivery | R/W | Controls ability of PCI 6254 to assert P_SERR# when it is unable to deliver posted write data after $2^{24}$ (or programmed Maximum Retry count at Timeout Control Register) attempts. P_SERR# is asserted if this event occurs when this bit is 0 and SERR# enable bit in the command register is set.<br>Reset value is 0. |
| 3 | Target abort during posted write | R/W | Controls ability of PCI 6254 to assert P_SERR# when it receives a target abort when attempting to deliver posted write data. P_SERR# is asserted if this event occurs when this bit is 0 and SERR# enable bit in the command register is set.<br>Reset value is 0. |
| 4 | Master abort on posted write | R/W | Controls ability of PCI 6254 to assert P_SERR# when it receives a master abort when attempting to deliver posted write data. P_SERR# is asserted if this event occurs when this bit is 0 and SERR# enable bit in the command register is set.<br>Reset value is 0. |
| 5 | Delayed Configuration or IO write nondelivery | R/W | Controls ability of PCI 6254 to assert P_SERR# when it is unable to deliver delayed write data after $2^{24}$ (or programmed Maximum Retry count at Timeout Control Register) attempts. P_SERR# is asserted if this event occurs when this bit is 0 and SERR# enable bit in the command register is set.<br>Reset value is 0. |
| 6 | Delayed read-no data from target | R/W | Controls ability of PCI 6254 to assert P_SERR# when it is unable to transfer any read data from the target after $2^{24}$ (or programmed Maximum Retry count at Timeout Control Register) attempts. P_SERR# is asserted if this event occurs when this bit is 0 and SERR# enable bit in the command register is set.<br>Reset value is 0. |
| 7 | Reserved | R/O | Reserved. Returns 0 when read. |

### GPIO[3:0] Output Data Register - Offset 65h

| Bit | Function | Type | Description |
|---|---|---|---|
| 3:0 | GPIO[3:0] output write 1 to clear | R/W1 TC | Writing 1 to any of these bits drives the corresponding bit low on the GPIO[3:0] bus if it is programmed as output. Writing 0 has no effect.<br>Read returns the last written value.<br>Resets to 0. |
| 7:4 | GPIO[3:0] output write 1 to set | R/W1 TC | Writing 1 to any of these bits drives the corresponding bit high on the GPIO[3:0] bus if it is programmed as output. Writing 0 has no effect.<br>Read returns the last written value.<br>Resets to 0. |

### GPIO[3:0] Output Enable Register - Offset 66h

| Bit | Function | Type | Description |
|---|---|---|---|
| 3:0 | GPIO output enable write 1 to clear | R/W1 TC | Writing 1 to any of these bits drives the corresponding bit on the GPIO[3:0] bus as input only. Writing 0 has no effect.<br>Read returns the last value written.<br>Resets to 0. |
| 7:4 | GPIO output enable write 1 to set | R/W1 TC | Writing 1 to any of these bits drives the corresponding bit on the GPIO[3:0] bus as output. GPIO[3:0] then drives the value set in the output data register (reg 65h). Writing 0 has no effect.<br>Read returns the last written value.<br>Resets to 0. |

### GPIO[3:0] Input Data Register - Offset 67h

| Bit | Function | Type | Description |
|---|---|---|---|
| 3:0 | Reserved | R/O | Reserved |
| 7:4 | GPIO[3:0] input data | R/O | This read-only register reads the state of the GPIO[3:0] pins. The state is updated on the PCI clock cycle following a change in the GPIO[3:0] state. |

**Clock Control Register (Read/Write) – Offset 68h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 1:0 | Clock 0 Disable | R/W | If either bit is 0, S_CLKOUT[0] is enabled.<br>When both bits are 1, S_CLKOUT[0] is disabled.<br>Upon secondary bus reset, this bit is initialized by shifting in a serial data stream. These bits are assigned to correspond to the PRSNT# pins for slot 0. |
| 3:2 | Clock 1 Disable | R/W | If either bit is 0, S_CLKO[1] is enabled.<br>When both bits are 1, S_CLKO[1] is disabled.<br>Upon secondary bus reset, this bit is initialized by shifting in a serial data stream. These bits are assigned to correspond to the PRSNT# pins for slot 1. |
| 5:4 | Clock 2 Disable | R/W | If either bit is 0, S_CLKO[2] is enabled.<br>When both bits are 1, S_CLKO[2] is disabled.<br>Upon secondary bus reset, this bit is initialized by shifting in a serial data stream. These bits are assigned to correspond to the PRSNT# pins for slot 2. |
| 7:6 | Clock 3 Disable | R/W | If either bit is 0, S_CLKO[3] is enabled.<br>When both bits are 1, S_CLKO[3] is disabled.<br>Upon secondary bus reset, this bit is initialized by shifting in a serial data stream. These bits are assigned to correspond to the PRSNT# pins for slot 3. |
| 8 | Clock 4 Disable | R/W | If 0, S_CLKO[4] is enabled.<br>When 1, S_CLKO[4] is disabled.<br>Upon secondary bus reset, this bit is initialized by shifting in a serial data stream. |
| 9 | Clock 5 Disable | R/W | If 0, S_CLKO[5] is enabled.<br>When 1, S_CLKO[5] is disabled.<br>Upon secondary bus reset, this bit is initialized by shifting in a serial data stream. |
| 10 | Clock 6 Disable | R/W | If 0, S_CLKO[6] is enabled.<br>When 1, S_CLKO[6] is disabled.<br>Upon secondary bus reset, this bit is initialized by shifting in a serial data stream. |
| 11 | Clock 7 Disable | R/W | If 0, S_CLKO[7] is enabled.<br>When 1, S_CLKO[7] is disabled.<br>Upon secondary bus reset, this bit is initialized by shifting in a serial data stream. |

| | | | |
|---|---|---|---|
| 12 | Clock 8 Disable | R/W | If 0, S_CLKO[8] is enabled.<br>When 1, S_CLKO[8] is disabled.<br>Upon secondary bus reset, this bit is initialized by shifting in a serial data stream. |
| 13 | Clock 9 Disable | R/W | If 0, S_CLKO[9] is enabled.<br>When 1, S_CLKO[9] is disabled.<br>Upon secondary bus reset, this bit is initialized by shifting in a serial data stream. |
| 15-14 | Reserved | R/O | Reserved |

**P_SERR# Status Register (Read/Write) – Offset 6Ah**

| Bit | Function | Type | Description |
|---|---|---|---|
| 0 | Address Parity error | R/WC | Signal P_SERR# was asserted due to address parity error on either side of the bridge. Reset to 0. |
| 1 | Posted Write Data Parity error | R/WC | Signal P_SERR# was asserted due to a posted write data parity error on the target bus. Reset to 0. |
| 2 | Post Write nondelivery | R/WC | Signal P_SERR# was asserted because PCI 6254 was unable to deliver posted write data to the target before timeout counter expires. Reset to 0. |
| 3 | Target abort during posted write | R/WC | Signal P_SERR# was asserted because PCI 6254 received a target abort when delivering posted write data. Reset to 0. |
| 4 | Master abort during posted write | R/WC | Signal P_SERR# was asserted because PCI 6254 received a master abort when delivering posted write data. Reset to 0. |
| 5 | Delayed write nondelivery | R/WC | Signal P_SERR# was asserted because PCI 6254 was unable to deliver delayed write data before time-out counter expires. Reset to 0. |
| 6 | Delayed read failed | R/WC | Signal P_SERR# was asserted because PCI 6254 was unable to read any data from the target before time-out counter expires. Reset to 0. |
| 7 | Delayed transaction master timeout | R/WC | Signal P_SERR# was asserted because a master did not repeat a read or write transaction before the master timeout counter expired on the initiator's bus. Reset to 0. |

**Clkrun Register (Read/Write) – Offset 6Bh**

| Bit | Function | Type | Description |
|---|---|---|---|
| 0 | Secondary Clock Stop Status | R/W | Secondary clock stop status<br>0 = Secondary clock not stopped<br>1 = Secondary clock stopped<br>Defaults to 0 |
| 1 | Secondary Clkrun Enable | R/W | Secondary clkrun protocol enable<br>0 = disable<br>1 = enable<br>Defaults to 0 |
| 2 | Primary Clock Stop | R/W | Primary clock stop<br>0 = allow primary clock to stop if secondary clock is stopped<br>1 = always keep primary clock running<br>Defaults to 0 |
| 3 | Primary Clkrun Enable | R/W | Primary clkrun protocol enable<br>0 = disable<br>1 = enable<br>Defaults to 0 |
| 4 | Clkrun Mode | R/W | Clkrun mode<br>0 = Stop the secondary clock only on request from the primary bus<br>1 = Stop the secondary clock whenever the secondary bus is idle and there are no requests from the primary bus.<br>Defaults to 0 |

### 6.3.3 Private Memory

Private Memory can be enabled via Chip Control Register bit 2 or by using the XB_MEM input pin.

**Private Memory Base Register (Read/Write) – Offset 6Ch**

This register defines the base address of the Private Memory address range. The upper twelve bits corresponding to address bits <31:20> are writeable. The lower 20 address bits (19:0) are assumed to be 00000h. The 12 bits are reset to 0. The lower 4 bits are read only and set to b'0001'.

**Private Memory Limit Register (Read/Write) – Offset 6Eh**

This register defines the upper limit address of the Private Memory address range. The upper twelve bits corresponding to address bits <31:20> are writeable. The 12 bits are reset to 0. The lower 4 bits are read only and are set to 0. The lower 20 address bits (19:0) are assumed to be FFFFFh. Reset to b'0001'.

**Private Memory Base Register Upper 32 Bits (Read/Write) – Offset 70h**

This register defines the upper 32 bit <63:32> memory base address of the Private Memory address. Reset to 1.

**Private Memory Limit Register Upper 32 Bits (Read/Write) – Offset 74h**

This register defines the upper 32 bit <63:32> memory limit address of the Private Memory address. Reset to 0.

**Hot Swap Switch and ROR Control (R/W) – Offset 9Ch**

| Bit | Function | Type | Description |
|---|---|---|---|
| 0 | **Hot Swap extraction switch** | R/W | **Hot Swap extraction switch:** Software switch used to signal extraction of board. If set, board is in inserted state. Writing a '0' to this bit will signal the pending extraction of the board. |
| 4-1 | **Reserved** | R/O | **Reserved** |
| 5 | **Downstream Translation BAR Access** | R/W | **1 = Enable the shadowed Downstream Translation BAR registers to be accessed.**<br>**Reset to 0.** |
| 6 | **Upstream Translation BAR Access** | R/W | **1 = Enable the shadowed Upstream Translation BAR registers to be accessed.**<br>**Reset to 0.** |
| 7 | **ROR Write Enable** | R/W | **Read Only Registers Write Enable:** Subsystem Vender ID at Register 2Ch and Subsystem ID Register at 2Eh are normally Read Only. Setting this bit to 1 will enable write to such Read Only ID Registers.<br><br>Power Management Registers DEh, E0h, and E3h are normally Read Only. Setting this bit to 1 will enable write to all Read Only Power Management Registers.<br><br>This bit must be cleared after the desired values have been modified in the Read Only Registers. |

## 6.3.4 GPIO Registers

**GPIO[7:4] Output Data Register - Offset 9Dh**

| Bit | Function | Type | Description |
|---|---|---|---|
| 3:0 | GPIO[7:4] output write 1 to clear | R/W1 TC | Writing 1 to any of these bits drives the corresponding bit low on the GPIO[7:4] bus if it is programmed as output. Writing 0 has no effect.<br>Defaults to 0 |
| 7:4 | GPIO[7:4] output write 1 to set | R/W1 TC | Writing 1 to any of these bits drives the corresponding bit high on the GPIO[7:4] bus if it is programmed as output. Writing 0 has no effect |

**GPIO[7:4] Output Enable Register - Offset 9Eh**

| Bit | Function | Type | Description |
|---|---|---|---|
| 3:0 | GPIO[7:4] output enable write 1 to clear | R/W1 TC | Writing 1 to any of these bits drives the corresponding bit on the GPIO[7:4] bus as input only. Writing 0 has no effect, reads returns last value written.<br>Defaults to 0 |
| 7:4 | GPIO[7:4] output enable write 1 to set | R/W1 TC | Writing 1 to any of these bits drives the corresponding bit on the GPIO[7:4] bus as output. GPIO[7:4] then drives the value set in the output data register (reg 65h). Writing 0 has no effect, reads returns last value written.<br>Defaults to 0 |

**GPIO[7:4] Input Data Register - Offset 9Fh**

| Bit | Function | Type | Description |
|---|---|---|---|
| 3:0 | Reserved | R/O | Reserved |
| 7:4 | GPIO[7:4] input data | R/O | This read-only register reads the state of the GPIO[7:4] pins. The state is updated on the PCI clock cycle following a change in the GPIO[7:4] state. |

**Power Up Status Register - Offset A0h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 7-0 | Power up Status | R/O | **Power up latched Status bits:** Upon PWRGD (power good), the status of GPIO[15:8] are latched in this registers. User can choose to use such status for any desired option setting or checking.<br>**Some Recommended Use (Must be 3.3V input):**<br>**GPIO15: Primary Power State:** 1 = Primary port power is stable.<br>**GPIO14: Secondary Power State:** 1 = Secondary port power is stable. |

**GPIO[15:8] Output Data Register - Offset A1h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 7:0 | GPIO[15:8] output data | R/W | GPIO[15:8] output data. Defaults to 0 |

**GPIO[15:8] Output Enable Register - Offset A2h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 7:0 | GPIO[15:8] output enable | R/W | Writing 1 to any of these bits drives the corresponding bit on the GPIO[15:8] bus as output.<br>Defaults to 0 |

**GPIO[15:8] Input Data Register - Offset A3h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 7:0 | GPIO[15:8] input data | R/O | This read-only register reads the state of the GPIO[15:8] pins. The state is updated on the PCI clock cycle following a change in the GPIO[15:8] state. |

### 6.3.5 Extended Registers

Currently there are 8 32bit sticky scratch registers available in PCI 6254 and they are at extended address 0h-7h.

**Extended Register Index - Offset D3h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 7:0 | Extended Index address | R/W | Index address for extended registers |

**Extended Register Dataport - Offset D4h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 31:0 | Extended Registers Dataport | R/W | A Configuration WRITE will cause the data presented at this port to be written into the register addressed by the Extended Register Index.<br><br>A Configuration READ will cause the data from the register addressed by the Extended Register Index to be presented to this port. |

## Extended Registers

| Register | Index |
|---|---|
| 32 Bit Sticky Register 0 | 0h |
| 32 Bit Sticky Register 1 | 1h |
| 32 Bit Sticky Register 2 | 2h |
| 32 Bit Sticky Register 3 | 3h |
| 32 Bit Sticky Register 4 | 4h |
| 32 Bit Sticky Register 5 | 5h |
| 32 Bit Sticky Register 6 | 6h |
| 32 Bit Sticky Register 7 | 7h |

**32 Bit Sticky Scratch Registers - Extended Register Index 0h-7h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 31:0 | Scratch Register | R/W | Sticky Scratch register. Upon Power Good, their values are undefined. If Power is Good, P_RSTIN# and S_RSTIN# active inputs do not affect their pre-existing value. |

### 6.3.6 Power Management and Hot Swap Registers

Power Management Registers DEh, E0h, and E3h are normally Read Only. However their default value can be changed by firmware or software by setting the Read Only Registers Write Enable bit at Register. After any modifications to such registers, this Write Enable bit must be cleared to preserve their Read Only nature.

**Capability Identifier (R/O) – Offset DCh**
This register is set to 01h to indicate power management interface registers.

**Next Item Pointer (R/O) – Offset DDh**
Set to E4h. This field provides an offset into the function's PCI Configuration Space pointing to the location of next item in the function's capability list. In PCI 6254, this points to the Hot Swap registers.

**Power Management Capabilities(R/O) – Offset DEh**
This register is EEPROM or ROR Write controlled loadable, but is READ ONLY during normal operation.

| Bit | Function | Type | Description |
|---|---|---|---|
| 0-2 | Version | R/O | This register is set to 001b, indicating that this function complies with Rev 1.0 of the PCI Power Management Interface Specification |
| 3 | PME Clock | R/O | This bit is a '0', indicating that PCI 6254 does not support PME# signaling. |
| 4 | Auxiliary Power Source | R/O | This bit is set to ‘0’ since PCI 6254 does not support PME# signaling |
| 5 | DSI | R/O | Device Specific Initialization. Returns ‘0’ indicating that PCI 6254 does not need special initialization |
| 6-8 | Reserved | R/O | Reserved |
| 9 | D1 Support | R/O | Returns ‘1’ indicating that PCI 6254 supports the D1 device power state |
| 10 | D2 Support | R/O | Returns ‘1’ indicating that PCI 6254 supports the D2 device power state |
| 11-15 | PME Support | R/O | Set to “0601” in Revision AA. Set to “7E01” in Revision AB. |

### Power Management Control/ Status(R/W) – Offset E0h

This register is EEPROM or ROR Write controlled loadable, but is READ ONLY during normal operation.

| Bit | Function | Type | Description |
|---|---|---|---|
| 0-1 | Power State | R/W | This 2-bit field is used both to determine the current power state of a function and to set the function into a new power state. The definition of the field values is given below.<br>00b - D0<br>01b - D1<br>10b - D2<br>11b – D3hot |
| 2-7 | Reserved | R/O | Reserved |
| 8 | PME Enable | R/W | This bit is set to '0' since PCI 6254 does not support PME# signaling |
| 9-12 | Data Select | R/O | This field returns '0000b' indicating PCI 6254 does not return any dynamic data |
| 13-14 | Data Scale | R/O | Returns '00b' when read. PCI 6254 does not return any dynamic data. |
| 15 | PME Status | R/W | This bit is set to '0' since PCI 6254 does not support PME# signaling |

### PMCSR Bridge Support(R/W) – Offset E2h

| Bit | Function | Type | Description |
|---|---|---|---|
| 0-5 | Reserved | R/O | Reserved |
| 6 | B2/B3 Support for D3hot | R/O | This bit reflects the state of the BPCC input pin. A '1' indicates that when PCI 6254 is programmed to D3hot state the secondary bus's clock is stopped. |
| 7 | Bus Power Control Enable | R/O | This bit reflects the state of the BPCC input pin. A '1' indicates that the power management state of the secondary bus follows that of PCI 6254 with one exception, D3hot state. |

### Power Management Data Register (RO) – Offset E3h

This register is EEPROM or ROR Write controlled loadable, but is READ ONLY during normal operation.

### Capability Identifier (R/O) – Offset E4h

This register is **set to 06h** to indicate Hot Swap interface registers.

### Next Item Pointer (R/O) - Offset E5h

**Set to E8h.** This field provides an offset into the function's PCI Configuration Space pointing to the location of next item in the function's capability list. In PCI 6254, this points to the Vital Product Data (VPD) registers.

### Hot Swap Register(R/W) – Offset E6h

| Bit | Function | Type | Description |
|---|---|---|---|
| 0 | DHA | R/W | Device Hiding Arm. Reset to 0.<br>1 = Arm Device Hiding<br>0 = Disarm Device Hiding<br>DHA is set to 1 by hardware during Hot Swap port PCI RSTIN# going inactive and handle switch is still unlocked. The locking of the handle will clear this bit. |
| 1 | EIM<br>ENUM# Mask Status | R/W | Enables or disables ENUM# assertion. Reset to 0.<br>0 = enable ENUM# signal<br>1 = mask off ENUM# signal |
| 2 | PIE | R/O | Pending INSert or EXTract: This bit is set when either INS or EXT is "1" or INS is armed (Write 1 to EXT bit).<br>1 = either an insertion or an extraction is in progress.<br>0 = Neither is pending |
| 3 | LOO<br>LED status | R/W | Indicates if LED is on or off. Reset to 0.<br>0 = LED is off<br>1 = LED is on |
| 5-4 | PI | R/W | Programming Interface:<br>Hardcode at 01: INS, EST, LOO, EIM and PIE, Device Hiding are supported. |
| 6 | EXT<br>Extraction State | R/W1C | This bit is set by hardware when the ejector handle is unlocked and INS = 0. |
| 7 | INS<br>Insertion State | R/W1C | This bit is set by hardware when Hot Swap port RSTIN# is deasserted, EEPROM autoload is completed and the ejector handle is locked.<br>Writing 1 to EXT bit also arms INS. |
| 15:8 | Reserved | R/O | Reserved and a read returns all 0. Write has no effect. |

## 6.3.7 VPD Registers

### Capability Identifier (R/O) - Offset E8h

This register is set to 03h to indicate VPD registers.

### Next Item Pointer (R/O) - Offset E9h

Set to 00h.

### VPD Register (R/W) – Offset EAh

| Bit | Function | Type | Description |
|---|---|---|---|
| 1-0 | Reserved | R/O | Reserved |
| 7-2 | VPD Address | R/W | **VPD operation**: Writing a '0' to this bit generates a read cycle from the EEPROM at the VPD address specified in bits 7-2 of this register. This bit will remain at a logic '0' value until EEPROM cycle is finished, then it be set to '1'. Data for reads is available at register ECh<br><br>Writing a '1' to this bit generates a write cycle to the EEPROM at the VPD address specified in bits 7-2 of this register. This bit will remain at a logic '1' value until EEPROM cycle is finished, then it be cleared to '0'. |
| 14-8 | Reserved | R/O | Reserved |
| 15 | VPD Operation | R/W | **VPD operation**: Writing a '0' to this bit generates a read cycle from the EEPROM at the VPD address specified in bits 7-2 of this register. This bit will remain at a logic '0' value until EEPROM cycle is finished, then it be set to '1'. Data for reads is available at register ECh<br><br>Writing a '1' to this bit generates a write cycle to the EEPROM at the VPD address specified in bits 7-2 of this register. This bit will remain at a logic '1' value until EEPROM cycle is finished, then it be cleared to '0'. |

### VPD Data Register (R/W) – Offset ECh

| Bit | Function | Type | Description |
|---|---|---|---|
| 31-0 | VPD Data | R/W | **VPD Data** (EEPROM data[addr + 0x40]) - The least significant byte of this register corresponds to the byte of VPD at the address specified by the VPD Address register. The data read from or written to this register uses the normal PCI byte transfer capabilities. |

# 7 PCI Bus Operation

This chapter presents detailed information about PCI transactions PCI 6254 responds to and PCI transactions initiated by PCI 6254.

## 7.1 PCI Transactions

Table 7–1 lists the command code and name of each PCI transaction that PCI 6254 initiates and responds to. The Master and Target columns indicate support for each transaction when PCI 6254 initiates transactions as a master, on the primary bus and on the secondary bus, and when PCI 6254 responds to transactions as a target, on the primary bus and on the secondary bus.

**Table 7–1, PCI Transactions**

| Type of transaction | | Initiates as Master | | Responds as Target | |
|---|---|---|---|---|---|
| | | Primary | Secondary | Primary | Secondary |
| 0000 | Interrupt acknowledge | N | N | N | N |
| 0001 | Special cycle | Y | Y | N | N |
| 0010 | I/O read | Y | Y | Y | Y |
| 0011 | I/O write | Y | Y | Y | Y |
| 0100 | Reserved | N | N | N | N |
| 0101 | Reserved | N | N | N | N |
| 0110 | Memory read | **Y** | Y | Y | Y |
| 0111 | Memory write | Y | Y | Y | Y |
| 1000 | Reserved | N | N | N | N |
| 1001 | Reserved | N | N | N | N |
| 1010 | Configuration read | N | Y | Y | N |
| 1011 | Configuration write | Type-1 | Y | Y | Type-1 |
| 1100 | Memory read multiple | Y | Y | Y | Y |
| 1101 | Dual address cycle | Y | Y | Y | Y |
| 1110 | Memory read line | Y | Y | Y | Y |
| 1111 | Memory write and invalidate | Y | Y | Y | Y |

As indicated in Table 7–1, the following PCI commands are not supported by PCI 6254:

- PCI 6254 ignores reserved command codes and does not generate any reserved commands.
- PCI 6254 never initiates an interrupt acknowledge transaction and, as a target, ignores interrupt acknowledge transactions. Interrupt acknowledge transactions are expected to reside entirely on the primary PCI bus closest to the host bridge.
- PCI 6254 does not respond to special cycle transactions. To generate special cycle transactions on other PCI buses, either upstream or downstream, a Type-1 configuration command must be used.
- PCI 6254 does not generate Type-0 configuration transactions on the primary interface. It will respond to Type-0 configuration transactions on the secondary PCI interface only if non-transparent mode is enabled.

## 7.2 Single Address Phase

A 32-bit address uses a single address phase. This address is driven on AD[31:0], and the bus command is driven on P_CBE[3:0]

PCI 6254 supports the linear increment address mode only, which is indicated when the low 2 address bits are equal to 0. If either of the low 2 address bits is nonzero, PCI 6254 automatically disconnects the transaction after the first data transfer.

## 7.3 Dual Address Phase

PCI 6254 supports the Dual Address Cycle (DAC) bus command to transfer 64-bit addresses. In DAC transactions, the first address phase is during the initial assertion of frame, and the second address phase is one

clock later. During the first address phase, the DAC command is presented on CBE[3:0], the lower 32 bits of the address on AD[31:0]. The second address phase has the cycle command on CBE[3:0], and the upper 32 bits of the address on AD[31:0]. When a 64-bit master uses DAC, it must provide the upper 32 bits of the address on AD[63:32] to and the command on CBE[7:4] during both address phases of the transaction to allow 64-bit targets additional time to decode the transaction.

DACs are used to access locations that are not in the first 4GB of PCI memory space. Addresses in the first 4GB of memory space always use a Single Address Cycle (SAC).
PCI 6254 supports DAC in the upstream and downstream direction.
PCI 6254 responds to DAC for the following commands only:

- Memory Write
- Memory Write and Invalidate
- Memory Read
- Memory Read Line
- Memory Read Multiple

## *7.4 Device Select (DEVSEL#) Generation*

PCI 6254 always performs positive address decoding when accepting transactions on either the primary or secondary buses. PCI 6254 never subtractively decodes. Medium DEVSEL# timing is used for 33MHz operation and Slow DEVSEL# timing is used for 66MHz operation.

## *7.5 Data Phase*

Depending on the command type, PCI 6254 can support multiple data phase PCI transactions. Write transactions are treated as either posted write or delayed write transactions.
Table 7–2 shows the method of forwarding used for each type of write operation.

**Table 7–2, Write Transaction Forwarding**

| Type of Transaction | Type of Forwarding |
|---|---|
| Memory write | Posted |
| Memory write and invalidate | Posted |
| I/O write | Delayed |
| Type-1 configuration write | Delayed |

### 7.5.1 Posted Write Transactions

When PCI 6254 determines that a memory write transaction is to be forwarded across the bridge, PCI 6254 asserts DEVSEL# with slow timing and TRDY# in the same cycle, provided that enough buffer space is available in the posted write data queue, and there are less than 4 outstanding posted transactions in the queue. PCI 6254 can accept 1 QUAD/DWORD of write data every PCI clock cycle; that is, no target wait states are inserted. Up to 256 bytes of posted write data is stored in internal posted write buffers and is eventually delivered to the target.

PCI 6254 continues to accept write data until one of the following events occurs:

- The initiator terminates the transaction normally.
- A cache line boundary or an aligned 4KB boundary is reached, depending on the transaction type.
- The posted write data buffer fills

When one of the last two events occurs, PCI 6254 returns a target disconnect to the requesting initiator on this data phase to terminate the transaction.
Once the posted write transaction is selected for completion, PCI 6254 requests ownership of the target bus. This can occur while PCI 6254 is still receiving data on the initiator bus. Once PCI 6254 has ownership of the target bus, and the target bus is detected in the idle condition, PCI 6254 generates the write cycle and continues to transfer write data until all write data corresponding to that transaction is delivered, or until a target termination is received. As long as write data exists in the queue, PCI 6254 can drive 1 QUAD/DWORD of write data each PCI clock cycle. If write data is flowing through PCI 6254 and the initiator stalls, PCI 6254 will insert wait states on the target bus if the queue empties.

PCI 6254 ends the transaction on the target bus when one of the following conditions is met:

- All posted write data has been delivered to the target.
- The target returns a target disconnect or target retry (PCI 6254 starts another transaction to deliver the rest of the write data).
- The target returns a target abort (PCI 6254 discards remaining write data).

The master latency timer expires, and PCI 6254 no longer has the target bus grant (PCI 6254 starts another transaction to deliver the remaining write data).

## 7.5.2 Memory Write and Invalidate Transactions

Memory Write and Invalidate transactions guarantee transfer of entire cache lines. By default, PCI 6254 will retry a Memory Write and Invalidate cycle until there is space for at least 1 cache line of data in the internal buffers. It will then complete the transaction on the secondary bus as a Memory Write and Invalidate cycle. PCI 6254 can also be programmed to accept Memory Write and Invalidate cycles under the same conditions as normal memory writes. In this case, if the write buffer fills before an entire cache line is transferred, PCI 6254 will disconnect and complete the write cycle on the secondary bus as a normal Memory Write cycle. (Register 46, bit12). PCI 6254 disconnects Memory Write and Invalidate commands at aligned cache line boundaries. The cache line size value in the cache line size register gives the number of DWORDs in a cache line. For PCI 6254, to generate Memory Write and Invalidate transactions, this cache line size value must be written to a value that is 8h, 10h, or 20h. If an invalid cache line size is programmed, wherein the value is 0, or is not a power of 2, or is greater than 20h DWORDs, PCI 6254 sets the cache line size to the minimum value of 8h. PCI 6254 always disconnects on the cache line boundary.

When the Memory Write and Invalidate transaction is disconnected before a cache line boundary is reached, typically because the posted write data buffer fills, the transaction is converted to a Memory Write transaction.

## 7.5.3 Delayed Write Transactions

A Delayed Write transaction is used to forward I/O Write and Type-1 configuration cycles through PCI 6254, and is limited to a single QUAD/DWORD data transfer.

When a write transaction is first detected on the initiator bus, PCI 6254 claims the access and returns a target retry to the initiator. During the cycle, PCI 6254 samples the bus command, address, and address parity bits. PCI 6254 also samples the first data QUAD/DWORD, byte enable bits, and data parity. Cycle information is placed into the delayed transaction queue if there are no other existing delayed transactions with the same cycle information, and if the delayed transaction queue is not full. When PCI 6254 schedules delayed write transaction to be the next cycle to be completed based on its ordering constraints, PCI 6254 initiates the transaction on the target bus. PCI 6254 transfers the write data to the target.

If PCI 6254 receives a target retry in response to the write transaction on the target bus, it continues to repeat the write transaction until the data transfer is completed, or until an error condition is encountered. If PCI 6254 is unable to deliver write data after $2^{24}$ attempts (programmable through register 45, bits 3-0), PCI 6254 ceases further write attempts and returns a target abort to the initiator. The delayed transaction is removed from the delayed transaction queue. PCI 6254 also asserts P_SERR# if the primary SERR# enable bit is set in the command register. When the initiator repeats the same write transaction (same command, address, byte enable bits, and data), after PCI 6254 has completed data delivery, and has all the complete cycle information in the queue, PCI 6254 claims the access returns TRDY# to the initiator, to indicate that the write data was transferred. If the initiator requests multiple QUAD/DWORD, PCI 6254 asserts STOP# in conjunction with TRDY# to signal a target disconnect. Note that only those bytes of write data with valid byte enable bits are compared. If any of the byte enable bits are turned off (driven HIGH), the corresponding byte of write data is not compared.

If the initiator repeats the write transaction before the data has been transferred to the target, PCI 6254 returns a target retry to the initiator. PCI 6254 continues to return a target retry to the initiator until write data is delivered to the target or an error condition is encountered. When the write transaction is repeated, PCI 6254 does not make a new entry into the delayed transaction queue.

PCI 6254 implements a discard timer that starts counting when the delayed write completion is at the head of the delayed transaction queue. The initial value of this timer can be set to one of four values, selectable through both the primary and the secondary master timeout bits in the bridge control register as well as the master timeout divider bits in register 45h. If the discard timer expires before the write cycle is retried, PCI 6254 discards the delayed write transaction from the delayed transaction queue. PCI 6254 also conditionally asserts P_SERR#.

## 7.5.4 Write Transaction Address Boundaries

PCI 6254 imposes internal address boundaries when accepting write data. The aligned address boundaries are used to prevent PCI 6254 from continuing a transaction over a device address boundary and to provide an upper limit on maximum latency. PCI 6254 returns a target disconnects to the initiator when it reaches the aligned address boundaries under the conditions shown in Table 7–3.

**Table 7–3, Write Transaction Disconnect Address Boundaries**

| Type of Transaction | Condition | Aligned Address Boundary |
|---|---|---|
| Delayed write | All | Disconnects after one data transfer |
| Posted memory write | Memory write disconnect control bit = 0[1] | 4KB aligned address boundary |
| Posted memory write | Memory write disconnect control bit = 1[1] | Disconnects at cache line boundary |
| Posted memory write and invalidate | **Cache line size = 8,** | 8h-DWORD aligned address boundary |
| Posted memory write and invalidate | Cache line size = 10h | 10h-DWORD aligned address boundary |
| **Posted memory write and invalidate** | **Cache line size = 20h** | **20h-DWORD aligned address boundary** |

[1]- Memory Write disconnect control bit is located in the chip control register at offset 40h in configuration space.

## 7.5.5 Buffering Multiple Write Transactions

PCI 6254 continues to accept posted memory write transactions as long as space for at least 1 DWORD of data in the posted write data buffer remains and there are less than 4 outstanding posted memory write cycles. If the posted write data buffer fills before the initiator terminates the write transaction, PCI 6254 returns a target disconnect to the initiator.

Delayed write transactions are posted as long as at least one open entry in the delayed transaction queue exists. PCI 6254 can queue up to four posted write transactions and four delayed transactions in both upstream and downstream directions.

## 7.5.6 Read Transactions

Delayed read forwarding is used for all read transactions crossing PCI 6254.
Delayed read transactions are treated as either prefetchable or nonprefetchable.
Table 7–4 shows the read behavior, prefetchable or nonprefetchable, for each type of read operation.

**Table 7–4: Read Transaction Prefetching**

| Type of transaction | Read behavior |
|---|---|
| I/O read | Prefetching never done |
| Configuration read | Prefetching never done |
| Memory read | Downstream: prefetching used if address in prefetchable space<br>Upstream: prefetching used if prefetch disable is off (default) |
| Memory read line | Prefetching always used if request is for more than one data transfer. |
| Memory read multiple | Prefetching always used if request is for more than one data transfer. |

## 7.5.7 Prefetchable Read Transactions

A prefetchable read transaction is a read transaction where PCI 6254 performs speculative DWORD reads, transferring data from the target before it is requested from the initiator. This behavior allows a prefetchable read transaction to consist of multiple data transfers. Only the first byte enable bits can be forwarded. PCI 6254 forces all byte enable bits of subsequent transfers to be enabled.

Prefetchable behavior is used for memory read line and memory read multiple transactions, as well as for memory read transactions that fall into prefetchable memory space.

The amount of the prefetched data depends on the type of transaction. The amount of prefetching may also be affected by the amount of free buffer space available in PCI 6254, and by any read address boundaries encountered. In addition, there are several PCI 6254-specific registers that can be used to optimize read prefetch behavior.

Prefetching should not be used for those read transactions that have side effects in the target device, that is, control and status registers, FIFOs, and so on. The target device's base address register or registers indicate if a memory address region is prefetchable.

## 7.5.8 Nonprefetchable Read Transactions

A nonprefetchable read transaction is read transaction by the initiator into a nonprefetchable region, and is used for I/O and configuration read transactions, as well as for memory reads from nonprefetchable memory space. In this case, PCI 6254 requests 1 and only 1 DWORD from the target and disconnects the initiator after delivery of the first DWORD of read data.

Nonprefetchable read transactions should not be used for regions where extra read transactions could have side effects, such as FIFO memory, or control registers. Accordingly, if it is important to retain the value of the byte enable bits during the data phase, use nonprefetchable read transactions. If these locations are mapped in memory space, use the memory read command and map the target into nonprefetchable (memory-mapped I/O) memory space to utilize nonprefetching behavior.

## 7.5.9 Read Prefetch Address Boundaries

PCI 6254 imposes internal read address boundaries on read prefetching. The address boundary is used by PCI 6254 to calculate the initial amount of data that it will prefetch. During read transactions to prefetchable regions, PCI 6254 will prefetch data until it reaches one of these aligned address boundaries, unless the target signals a target disconnect before the read prefetch boundary is reached. Once the aligned address boundary is reached, PCI 6254 may optionally continue prefetching data, depending on certain conditions (see section on flow-through optimization). When PCI 6254 finishes transferring this read data to the initiator, it returns a target disconnect with the last data transfer, unless the initiator completes the transaction before all prefetched read data is delivered. Any leftover prefetched data is discarded.

Prefetchable read transactions in flow-through mode prefetch to the nearest aligned 4KB address boundary, or until the initiator deasserts FRAME#.

Table 7–5 shows the read prefetch address boundaries for read transactions during non-flow-through mode.

**Table 7–5: Read Prefetch Address Boundaries**

| Type of transaction | Address space | Prefetch aligned address boundary |
|---|---|---|
| Configuration read | - | 1 DWORD (no prefetch) |
| I/O read | - | 1 DWORD (no prefetch) |
| Memory read | Nonprefetchable | 1 DWORD (no prefetch) |
| Memory read | Prefetchable | Configured through prefetch count registers |
| Memory read line | Prefetchable | Configured through prefetch count registers |
| Memory read multiple | Prefetchable | Configured through prefetch count registers |

## 7.5.10 Delayed Read Requests

PCI 6254 treats all read transactions as delayed read transactions, which means that the read request from the initiator is posted into a delayed transaction queue. Read data from the target is placed in the read data queue directed toward the initiator bus interface and is transferred to the initiator when the initiator repeats the read transaction.
When PCI 6254 accepts a delayed read request, it first samples the read address, read bus command, and address parity. When IRDY# is asserted, PCI 6254 then samples the byte enable bits for the first data phase. This information is entered into the delayed transaction queue. PCI 6254 terminates the transaction by signaling a target retry to the initiator. Upon reception of the target retry, the initiator is required to continue to repeat the same read transaction until at least one data transfer is completed, or until a target response other than a target retry (target abort, or master abort) is received.

## 7.5.11 Delayed Read Completion with Target

When a delayed read request is scheduled by PCI 6254 to be executed, PCI 6254 arbitrates for the target bus and initiates the read transaction, using the exact read address and read command captured from the initiator during the initial delayed read request. If the read transaction is a nonprefetchable read, PCI 6254 drives the captured byte enable bits during the next cycle. If the transaction is a prefetchable read transaction, it drives the captured first byte enable bits followed by 0 for the subsequent data phases. If PCI 6254 receives a target retry in response to the read transaction on the target bus, it continues to repeat the read transaction until at least one data transfer is completed, or until an error condition is encountered. If the transaction is terminated via normal master termination or target disconnect after at least one data transfer has been completed, PCI 6254 does not initiate any further attempts to read more data.

If PCI 6254 is unable to obtain read data from the target after $2^{24}$ attempts (default), PCI 6254 ceases further read attempts and returns a target abort to the initiator. The delayed transaction is removed from the delayed transaction queue. PCI 6254 also asserts P_SERR# if the primary SERR# enable bit is set in the command register.
Once PCI 6254 receives DEVSEL# and TRDY# from the target, it transfers the data stored in the internal read FIFO, before terminating the transaction. PCI 6254 can accept 1 DWORD/QWORD of read data each PCI clock cycle; no master wait states are inserted. The number of DWORD/QWORD transferred during a delayed read transaction depends on the conditions given in Table 7–5 (assuming no disconnect is received from the target).

## 7.5.12 Delayed Read Completion on Initiator Bus

When the transaction has been completed on the target bus, and the delayed read data is at the head of the read data queue, and all ordering constraints with posted write transactions have been satisfied, PCI 6254 transfers the data to the initiator when the initiator repeats the transaction. For memory read transactions, PCI 6254 aliases the memory read, memory read line, and memory read multiple bus commands when matching the bus command

of the transaction to the bus command in the delayed transaction queue. PCI 6254 returns a target disconnect along with the transfer of the last DWORD of read data to the initiator. If PCI 6254 initiator terminates the transaction before all read data has been transferred, the remaining read data left in data buffers is discarded.

When the master repeats the transaction and starts transferring prefetchable read data from data buffers while the read transaction on the target bus is still in progress and before a read boundary is reached on the target bus, the read transaction starts operating in flow-through mode. Because data is flowing through the data buffers from the target to the initiator, long read bursts can then be sustained. In this case, the read transaction is allowed to continue until the initiator terminates the transaction, or until an aligned 4KB address boundary is reached, or until the buffer fills, whichever comes first. When the buffer empties, PCI 6254 reflects the stalled condition to the initiator by deasserting TRDY# for a maximum of 8 clock periods until more read data is available; otherwise, PCI 6254 will disconnect the cycle. When the initiator terminates the transaction, PCI 6254 deassertion of FRAME# on the initiator bus is forwarded to the target bus. Any remaining read data is discarded.

PCI 6254 implements a discard timer that starts counting when the delayed read completion is at the head of the delayed transaction queue, and the read data is at the head of the read data queue. The initial value of this timer is programmable through configuration register. If the initiator does not repeat the read transaction before Discard Timer expires, PCI 6254 discards the read transaction (and the read data from its queues). PCI 6254 also conditionally asserts P_SERR#.

PCI 6254 has the capability to post multiple delayed read requests, up to a maximum of four in each direction. If an initiator starts a read transaction that matches the address and read command of a read transaction that is already queued, the current read command is not stored as it is already contained in the delayed transaction queue.

### 7.5.13 Configuration Transactions

Configuration transactions are used to initialize a PCI system. Every PCI device has a configuration space that is accessed by configuration commands. All registers are accessible in configuration space only.
In addition to accepting configuration transactions for initialization of its own configuration space, PCI 6254 forwards configuration transactions for device initialization in hierarchical PCI systems, as well as for special cycle generation.

During non-transparent mode, PCI 6254 can also accept configuration transactions on its secondary interface (see non-transparent operation section).

To support hierarchical PCI bus systems, Type-0 and Type-1 configuration transactions are specified.
Type-0 configuration transactions are issued when the intended target resides on the same PCI bus as the initiator. A Type-0 configuration transaction is identified by the configuration command and the Lowest 2 bits of the address set to 00b.

Type-1 configuration transactions are issued when the intended target resides on another PCI bus, or when a special cycle is to be generated on another PCI bus. A Type-1 configuration command is identified by the configuration command and the Lowest 2 address bits set to 01b.

The register number is found in both Type-0 and Type-1 formats and gives the DWORD address of the configuration register to be accessed. The function number is also included in both Type-0 and Type-1 formats and indicates which function of a multifunction device is to be accessed. For single-function devices, this value is not decoded. Type-1 configuration transaction addresses also include a 5-bit field designating the device number that identifies the device on the target PCI bus that is to be accessed. In addition, the bus number in Type-1 transactions specifies the PCI bus to which the transaction is targeted.

### 7.5.14 Type-0 Access to PCI 6254

The configuration space is accessed by a Type-0 configuration transaction on the primary interface. The configuration space cannot be accessed from the secondary bus. PCI 6254 responds to a Type-0 configuration transaction by asserting P_DEVSEL# when the following conditions are met during the address phase:

- The bus command is a configuration read transaction or configuration write transaction.
- Low 2 address bits P_AD[1:0] must be 00b.
- Signal P_IDSEL must be asserted.

PCI 6254 limits all configuration accesses to a single DWORD data transfer and returns a target disconnect with the first data transfer if additional data phases are requested. Because read transactions to configuration space do not have side effects, all bytes in the requested DWORD are returned, regardless of the value of the byte enable bits.

Type-0 configuration write and read transactions do not use data buffers; that is, these transactions are completed immediately, regardless of the state of the data buffers.
PCI 6254 ignores all Type-0 transactions initiated on the secondary interface.

### 7.5.15 Type-1 to Type-0 Translation

Type-1 configuration transactions are used specifically for device configuration in a hierarchical PCI bus system. A PCI-to-PCI bridge is the only type of device that should respond to a Type-1 configuration command. Type-1 configuration commands are used when the configuration access is intended for a PCI device that resides on a PCI bus other than the one where the Type-1 transaction is generated.

PCI 6254 performs a Type-1 to Type-0 translation when the Type-1 transaction is generated on the primary bus and is intended for a device attached directly to the secondary bus. PCI 6254 must convert the configuration command to a Type-0 format so that the secondary bus device can respond to it. Type-1 to Type-0 translations are performed only in the downstream direction; that is, PCI 6254 generates a Type-0 transaction only on the secondary bus, and never on the primary bus.

PCI 6254 responds to a Type-1 configuration transaction and translates it into a Type-0 transaction on the secondary bus when the following conditions are met during the address phase:

- The low 2 address bits on P_AD[1:0] are 01b.
- The bus number in address field P_AD[23:16] is equal to the value in the secondary bus number register in configuration space.
- The bus command on P_CBE[3:0] is a configuration read or configuration write transaction.

When PCI 6254 translates the Type-1 transaction to a Type-0 transaction on the secondary interface, it performs the following translations to the address:

- Sets the low 2 address bits on S_AD[1:0] to 00b.
- Decodes the device number and drives the bit pattern specified in Table 7–6 on S_AD[31:16] for the purpose of asserting the device's IDSEL signal.
- Sets S_AD[15:11] to 0.
- Leaves unchanged the function number and register number fields.

PCI 6254 asserts a unique address line based on the device number. These address lines may be used as secondary bus IDSEL signals. The mapping of the address lines depends on the device number in the Type-1 address bits P_AD[15:11]. Table 7–6 presents the mapping that PCI 6254 uses.

**Table 7–6: Device Number to IDSEL S_AD Pin Mapping**

| Device Number | P_AD[15:11] | Secondary IDSEL S_AD[31:16] | S_AD Bit |
|---|---|---|---|
| 0 | 00000 | 0000 0000 0000 0001 | 16 |
| 1 | 00001 | 0000 0000 0000 0010 | 17 |
| 2 | 00010 | 0000 0000 0000 0100 | 18 |
| 3 | 00011 | 0000 0000 0000 1000 | 19 |
| 4 | 00100 | 0000 0000 0001 0000 | 20 |
| 5 | 00101 | 0000 0000 0010 0000 | 21 |
| 6 | 00110 | 0000 0000 0100 0000 | 22 |
| 7 | 00111 | 0000 0000 1000 0000 | 23 |
| 8 | 01000 | 0000 0001 0000 0000 | 24 |
| 9 | 01001 | 0000 0010 0000 0000 | 25 |
| 10 | 01010 | 0000 0100 0000 0000 | 26 |
| 11 | 01011 | 0000 1000 0000 0000 | 27 |
| 12 | 01100 | 0001 0000 0000 0000 | 28 |
| 13 | 01101 | 0010 0000 0000 0000 | 29 |
| 14 | 01110 | 0100 0000 0000 0000 | 30 |
| 15 | 01111 | 1000 0000 0000 0000 | 31 |
| Special Cycle | 1XXXX | 0000 0000 0000 0000 | None |

PCI 6254 can assert up to 16 unique address lines to be used as IDSEL signals for up to 16 devices on the secondary bus, for device numbers ranging from 0 through 15. Because of electrical loading constraints of the PCI bus, more than 16 IDSEL signals should not be necessary. However, if device numbers greater than 15 are desired, some external method of generating IDSEL lines must be used, and no upper address bits are then asserted. The configuration transaction is still translated and passed from the primary bus to the secondary bus. If no IDSEL pin is asserted to a secondary device, the transaction ends in a master abort.

PCI 6254 forwards Type-1 to Type-0 configuration read or write transactions as delayed transactions. Type-1 to Type-0 configuration read or write transactions are limited to a single 32-bit data transfer. When Type-1 to Type-0 configurations cycles are forwarded, address stepping is used, valid address is driven on the bus before the FRAME# is asserted. Type-0 configuration address stepping is programmable through register 46h, bits 6-4.

### 7.5.16 Type-1 to Type-1 Forwarding

Type-1 to Type-1 transaction forwarding provides a hierarchical configuration mechanism when two or more levels of PCI-to-PCI bridges are used.

When PCI 6254 detects a Type-1 configuration transaction intended for a PCI bus downstream from the secondary bus, PCI 6254 forwards the transaction unchanged to the secondary bus. Ultimately, this transaction is translated to a Type-0 configuration command or to a special cycle transaction by a downstream PCI-to-PCI bridge. Downstream Type-1 to Type-1 forwarding occurs when the following conditions are met during the address phase:

- The low 2 address bits are equal to 01b.
- The bus number falls in the range defined by the lower limit (exclusive) in the secondary bus number register and the upper limit (inclusive) in the subordinate bus number register.
- The bus command is a Configuration Read or Write transaction.

PCI 6254 also supports Type-1 to Type-1 forwarding of configuration write transactions upstream to support upstream special cycle generation. A Type-1 configuration command is forwarded upstream when the following conditions are met:

- The low 2 address bits are equal to 01b.
- The bus number falls outside the range defined by the lower limit (inclusive) in the secondary bus number register and the upper limit (inclusive) in the subordinate bus number register.
- The device number in address bits AD[15:11] is equal to 11111b.
- The function number in address bits AD[10:8] is equal to 111b.
- The bus command is a Configuration Write transaction.

PCI 6254 forwards Type-1 to Type-1 configuration write transactions as delayed transactions. Type-1 to Type-1 configuration write transactions are limited to a single data transfer.

### 7.5.17 Special Cycles

The Type-1 configuration mechanism is used to generate special cycle transactions in hierarchical PCI systems. Special cycle transactions are ignored by acting as a target and are not forwarded across the bridge. Special cycle transactions can be generated from Type-1 configuration write transactions in either the upstream or the downstream direction.

PCI 6254 initiates a special cycle on the target bus when a Type-1 Configuration Write transaction is detected on the initiating bus and the following conditions are met during the address phase:

- The low 2 address bits on AD[1:0] are equal to 01b.
- The device number in address bits AD[15:11] is equal to 11111b.
- The function number in address bits AD[10:8] is equal to 111b.
- The register number in address bits AD[7:2] is equal to 000000b.
- The bus number is equal to the value in the secondary bus number register in configuration space for downstream forwarding or equal to the value in the primary bus number register in configuration space for upstream forwarding.
- The bus command on CBE is a Configuration Write command.

When PCI 6254 initiates the transaction on the target interface, the bus command is changed from configuration write to special cycle. The address and data are forwarded unchanged. Devices that use special cycles ignore the address and decode only the bus command. The data phase contains the special cycle message. The transaction is forwarded as a delayed transaction, but in this case the target response is not forwarded back (because special cycles result in a master abort). Once the transaction is completed on the target bus, through detection of the master abort condition, PCI 6254 responds with TRDY# to the next attempt of the configuration transaction from the initiator. If more than one data transfer is requested, PCI 6254 responds with a target disconnect operation during the first data phase.

## *7.6 Transaction Termination*

This section describes how PCI 6254 returns transaction termination conditions back to the initiator.
The initiator can terminate transactions with one of the following types of termination:

- Normal termination: It occurs when the initiator deasserts FRAME# at the beginning of the last data phase, and deasserts IRDY# at the end of the last data phase in conjunction with either TRDY# or STOP# assertion from the target.
- Master abort: It occurs when no target response is detected. When the initiator does not detect a DEVSEL# from the target within five clock cycles after asserting FRAME#, the initiator terminates the transaction with a master abort. If FRAME# is still asserted, the initiator deasserts FRAME# on the next cycle, and then deasserts IRDY# on the following cycle. IRDY# must be asserted in the same cycle in which FRAME# deasserts. If FRAME# is already deasserted, IRDY# can be deasserted on the next clock cycle following detection of the master abort condition.

The target can terminate transactions with one of the following types of termination:

- Normal termination: TRDY# and DEVSEL# asserted in conjunction with FRAME# deasserted and IRDY# asserted.
- Target retry: STOP# and DEVSEL# asserted without TRDY# during the first data phase. No data transfers occur during the transaction. This transaction must be repeated.
- Target disconnect (with data transfer): STOP# and DEVSEL# asserted with TRDY#. Signals that this is the last data transfer of the transaction.
- Target disconnect (without data transfer): STOP# and DEVSEL# asserted without TRDY# after previous data transfers have been made. Indicates that no more data transfers will be made during this transaction.
- Target abort: STOP# asserted without DEVSEL# and without TRDY#. Indicates that the target will never be able to complete this transaction. DEVSEL# must be asserted for at least one cycle during the transaction before the target abort is signaled.

### 7.6.1 Master Termination Initiated by PCI 6254

PCI 6254, as an initiator, uses normal termination if DEVSEL# is returned by the target within five clock cycles of PCI 6254's assertion of FRAME# on the target bus. As an initiator, PCI 6254 terminates a transaction when the following conditions are met:

- During a delayed write transaction, a single DWORD/QWORD is delivered.
- During a nonprefetchable read transaction, a single DWORD/QWORD is transferred from the target.
- During a prefetchable read transaction, a prefetch boundary is reached.
- For a posted write transaction, all write data for the transaction is transferred from data buffers to the target.
- For a burst transfer, with the exception of memory write and invalidate transactions, the master latency timer expires and PCI 6254's bus grant is deasserted.
- The target terminates the transaction with a retry, disconnect, or target abort.

If PCI 6254 is delivering posted write data when it terminates the transaction because the master latency timer expires, it initiates another transaction to deliver the remaining write data. The address of the transaction is updated to reflect the address of the current DWORD to be delivered.
If PCI 6254 is prefetching read data when it terminates the transaction because the master latency timer expires, it does not repeat the transaction to obtain more data.

### 7.6.2 Master Abort Received by PCI 6254

If the initiator initiates a transaction on the target bus and does not detect DEVSEL# returned by the target within five clock cycles of PCI 6254's assertion of FRAME#, PCI 6254 terminates the transaction as specified through the master abort mode bit of the bridge control register.

For delayed read and write transactions, PCI 6254 can either assert TRDY# and return FFFF_FFFFh for reads, or return target abort. SERR# is also optionally asserted.

When a master abort is received in response to a posted write transaction, PCI 6254 discards the posted write data and makes no more attempts to deliver the data. PCI 6254 sets the received master abort bit in the status register when the master abort is received on the primary bus, or it sets the received master abort bit in the secondary status register when the master abort is received on the secondary interface. When a master abort is detected in response to a posted write transaction, and the master abort mode bit is set, PCI 6254 also asserts P_SERR# if enabled by the SERR# enable bit in the command register and if not disabled by the device-specific P_SERR# disable bit for master abort during posted write transactions (that is, master abort mode = 1; SERR# enable bit = 1; and P_SERR# disable bit for master aborts = 0.

## 7.6.3 Target Termination Received by PCI 6254

When PCI 6254 initiates a transaction on the target bus and the target responds with DEVSEL#, the target can end the transaction with one of the following types of termination:

- Normal termination (upon deassertion of FRAME#)
- Target retry
- Target disconnect
- Target abort

PCI 6254 handles these terminations in different ways, depending on the type of transaction being performed.

### 7.6.3.1 Delayed Write Target Termination Response

When PCI 6254 initiates a delayed write transaction, the type of target termination received from the target can be passed back to the initiator. Table 7–7 shows the response to each type of target termination that occurs during a delayed write transaction.

PCI 6254 repeats a delayed write transaction until one of the following conditions is met:

- PCI 6254 completes at least one data transfer.
- PCI 6254 receives a master abort.
- PCI 6254 receives a target abort.

PCI 6254 makes $2^{24}$ (default) write attempts resulting in a response of target retry.

**Table 7–7: Response to Delayed Write Target Termination**

| Target Termination | Response |
|---|---|
| Normal | Return disconnect to initiator with first data transfer only if multiple data phases requested. |
| Target retry | Return target retry to initiator. Continue write attempts to target. |
| Target disconnect | Return disconnect to initiator with first data transfer only if multiple data phases requested. |
| Target abort | Return target abort to initiator.<br>Set received target abort bit in target interface status register.<br>Set signaled target abort bit in initiator interface status register. |

After PCI 6254 makes $2^{24}$ attempts of the same delayed write transaction on the target bus, PCI 6254 asserts P_SERR# if the primary SERR# enable bit is set in the command register and the implementation-specific P_SERR# disable bit for this condition is not set in the P_SERR# event disable register. PCI 6254 stops initiating transactions in response to that delayed write transaction. The delayed write request is discarded. Upon a subsequent write transaction attempt by the initiator, PCI 6254 returns a target abort.

### 7.6.3.2 Posted Write Target Termination Response

When PCI 6254 initiates a posted write transaction, the target termination cannot be passed back to the initiator. Table 7–8 shows the response to each type of target termination that occurs during a posted write transaction.

**Table 7–8: Response to Posted Write Target Termination**

| Target termination | Response |
|---|---|
| Normal | No additional action. |
| Target retry | Repeat write transaction to target. |
| Target disconnect | Initiate write transaction to deliver remaining posted write data. |
| Target abort | Set received target abort bit in the target interface status register.<br>Assert P_SERR# if enabled, and set the signaled system error bit in primary status register. |

Note that when a target retry or target disconnect is returned and posted write data associated with that transaction remains in the write buffers, PCI 6254 initiates another write transaction to attempt to deliver the rest of the write data. In the case of a target retry, the exact same address will be driven as for the initial write transaction attempt. If a target disconnect is received, the address that is driven on a subsequent write transaction attempt is updated to reflect the address of the current DWORD. If the initial write transaction is a memory write and invalidate transaction, and a partial delivery of write data to the target is performed before a target disconnect is received, PCI 6254 uses the memory write command to deliver the rest of the write data because less than a cache line will be transferred in the subsequent write transaction attempt.

After PCI 6254 makes $2^{24}$ write transaction attempts and fails to deliver all the posted write data associated with that transaction, PCI 6254 asserts P_SERR# if the primary SERR# enable bit is set in the command register and the device-specific P_SERR# disable bit for this condition is not set in the P_SERR# event disable register. The write data is discarded.

### 7.6.3.3 Delayed Read Target Termination Response

When PCI 6254 initiates a delayed read transaction, the abnormal target responses can be passed back to the initiator. Other target responses depend on how much data the initiator requests. Table 7–9 shows the response to each type of target termination that occurs during a delayed read transaction.

**Table 7–9: Response to Delayed Read Target Termination**

| Target termination | Response |
|---|---|
| Normal | If prefetchable, target disconnect only if initiator requests more data than read from target. If nonprefetchable, target disconnect on first data phase. |
| Target retry | Reinitiate read transaction to target |
| Target disconnect | If initiator requests more data than read from target, return target disconnect to initiator |
| Target abort | Return target abort to initiator.<br>Set received target abort bit in the target interface status register.<br>Set signaled target abort bit in the initiator interface status register. |

PCI 6254 repeats a delayed read transaction until one of the following conditions is met:

- PCI 6254 completes at least one data transfer.
- PCI 6254 receives a master abort.
- PCI 6254 receives a target abort.
- PCI 6254 makes $2^{24}$ read attempts resulting in a response of target retry.

After PCI 6254 makes $2^{24}$ attempts of the same delayed read transaction on the target bus, PCI 6254 asserts P_SERR# if the primary SERR# enable bit is set in the command register and the implementation-specific P_SERR# disable bit for this condition is not set in the P_SERR# event disable register. PCI 6254 stops initiating transactions in response to that delayed read transaction. The delayed read request is discarded. Upon a subsequent read transaction attempt by the initiator, PCI 6254 returns a target abort.

### 7.6.4 Target Termination Initiated by PCI 6254

PCI 6254 can return a Target Retry, Target Disconnect, or Target-Abort to an initiator for reasons other than detection of that condition at the target interface.

#### 7.6.4.1 Target Retry

PCI 6254 returns a Target Retry to the initiator when it cannot accept write data or return read data as a result of internal conditions. PCI 6254 returns a target retry to an initiator when any of the following conditions is met:

For delayed write transactions:

- The transaction is being entered into the delayed transaction queue.
- The transaction has already been entered into the delayed transaction queue, but target response has not yet been received.
- Target response has been received but the posted memory write ordering rule prevents the cycle from being completed.
- The delayed transaction queue is full, and the transaction cannot be queued.
- A transaction with the same address and command has been queued.
- A locked sequence is being propagated across PCI 6254, and the write transaction is not a locked transaction.
- The target bus is locked and the write transaction is a locked transaction.

For delayed read transactions:

- The transaction is being entered into the delayed transaction queue.
- The read request has already been queued, but read data is not yet available.
- Data has been read from the target, but it is not yet at the head of the read data queue, or a posted write transaction precedes it.
- The delayed transaction queue is full, and the transaction cannot be queued.
- A delayed read request with the same address and bus command has already been queued.
- A locked sequence is being propagated across PCI 6254, and the read transaction is not a locked transaction.
- The target bus is locked and the write transaction is a locked transaction.

For posted write transactions:

- The posted write data buffer does not have enough space for the address and at least two QWORDs of write data.
- A locked sequence is being propagated across PCI 6254, and the write transaction is not a locked transaction.

When a target retry is returned to the initiator of a delayed transaction, the initiator must repeat the transaction with the same address and bus command as well as the data if this is a write transaction, within the time frame specified by the master timeout value; otherwise, the transaction is discarded from the buffers.

### 7.6.4.2 Target Disconnected

PCI 6254 returns a Target Disconnect to an initiator when one of the following conditions is met:

- PCI 6254 hits an internal address boundary
- PCI 6254 cannot accept any more write data
- PCI 6254 has no more read data to deliver

### 7.6.4.3 Target-Abort

PCI 6254 returns a Target-Abort to an initiator when one of the following conditions is met:

- PCI 6254 is returning a target abort from the intended target.
- PCI 6254 detects a master abort on the target, and the master abort mode bit is set.
- PCI 6254 is unable to obtain delayed read data from the target or to deliver delayed write data to the target after $2^{24}$ attempts.

When PCI 6254 returns a target abort to the initiator, it sets the signaled target abort bit in the status register corresponding to the initiator interface.

# 8 Address Decoding

PCI 6254 uses three address ranges that control I/O and memory transaction forwarding. These address ranges are defined by base and limit address registers in the configuration space. This chapter describes these address ranges, as well as ISA-mode and VGA-addressing support.

## *8.1 Address Ranges*

PCI 6254 uses the following address ranges that determine which I/O and memory transactions are forwarded from the primary PCI bus to the secondary PCI bus, and from the secondary bus to the primary bus:

- One 32-bit I/O address range
- One 32-bit memory-mapped I/O (nonprefetchable memory) range
- One 32-bit prefetchable memory address range

Transactions falling within these ranges are forwarded downstream from the primary PCI bus to the secondary PCI bus. Transactions falling outside these ranges are forwarded upstream from the secondary PCI bus to the primary PCI bus.

## *8.2 I/O Address Decoding*

PCI 6254 uses the following mechanisms that are defined in the configuration space to specify the I/O address space for downstream and upstream forwarding:

- I/O base and limit address registers
- The ISA enable bit
- The VGA mode bit
- The VGA snoop bit

This section provides information on the I/O address registers and ISA mode.

To enable downstream forwarding of I/O transactions, the I/O enable bit must be set in the command register in configuration space. If the I/O enable bit is not set, all I/O transactions initiated on the primary bus are ignored. To enable upstream forwarding of I/O transactions, the master enable bit must be set in the command register. If the master enable bit is not set, PCI 6254 ignores all I/O and memory transactions initiated on the secondary bus. Setting the master enable bit also allows upstream forwarding of memory transactions.

**Caution:** If any configuration state affecting I/O transaction forwarding is changed by a configuration write operation on the primary bus at the same time that I/O transactions are ongoing on the secondary bus, the PCI 6254 response to the secondary bus I/O transactions is not predictable. Configure the I/O base and limit address registers, ISA enable bit, VGA mode bit, and VGA snoop bit before setting the I/O enable and master enable bits, and change them subsequently only when the primary and secondary PCI buses are idle.

### 8.2.1 I/O Base and Limit Address Registers

PCI 6254 implements one set of I/O base and limit address registers in configuration space that define an I/O address range downstream forwarding. PCI 6254 supports 32-bit I/O addressing, which Allows I/O addresses downstream of PCI 6254 to be mapped anywhere in a 4GB I/O address space.

I/O transactions with addresses that fall inside the range defined by the I/O base and limit registers are forwarded downstream from the primary PCI bus to the secondary PCI bus. I/O transactions with addresses that fall outside this range are forwarded upstream from the secondary PCI bus to the primary PCI bus.
The I/O range can be turned off by setting the I/O base address to a value greater than that of the I/O limit address. When the I/O range is turned off, all I/O transactions are forwarded upstream, and no I/O transactions are forwarded downstream.

The I/O range has a minimum granularity of 4KB and is aligned on a 4KB boundary. The maximum I/O range is 4GB in size.

The I/O base register consists of an 8-bit field at configuration address 1Ch, and a 16-bit field at address 30h. The top 4 bits of the 8-bit field define bits [15:12] of the I/O base address. The bottom 4 bits read only as 1h to indicate that PCI 6254 supports 32-bit I/O addressing. Bits [11:0] of the base address are assumed to be 0, which naturally aligns the base address to a 4KB boundary. The 16 bits contained in the I/O base upper 16 bits register at configuration offset 30h define AD[31:16] of the I/O base address. All 16 bits are read/write. After primary bus reset or chip reset, the value of the I/O base address is initialized to 0000_0000h.

The I/O limit register consists of an 8-bit field at configuration offset 1Dh and a 16-bit field at offset 32h. The top 4 bits of the 8-bit field define bits [15:12] of the I/O limit address. The bottom 4 bits read only as 1h to indicate that 32-bit I/O addressing is supported. Bits [11:0] of the limit address are assumed to be FFFh, which naturally aligns the limit address to the top of a 4KB I/O address block. The 16 bits contained in the I/O limit upper 16 bits register at configuration offset 32h define AD[31:16] of the I/O limit address. All 16 bits are read/write. After primary bus reset or chip reset, the value of the I/O limit address is reset to 0000 0FFFh.

**Note**
Write these registers with their appropriate values before setting either the I/O enable bit or the master enable bit in the command register in configuration space.

## *8.3 ISA Mode*

PCI 6254 supports ISA mode by providing an ISA enable bit in the bridge control register in configuration space. ISA mode modifies the response of PCI 6254 inside the I/O address range in order to support mapping of I/O space in the presence of an ISA bus in the system. This bit only affects the response of PCI 6254 when the transaction falls inside the address range defined by the I/O base and limit address registers, and only when this address also falls inside the first 64KB of I/O space (address bits [31:16] are 0000h).

When the ISA enable bit is set, PCI 6254 does not forward downstream any I/O transactions addressing the top 768 bytes of each aligned 1KB block. Only those transactions addressing the bottom 256 bytes of an aligned 1KB block inside the base and limit I/O address range are forwarded downstream. Transactions above the 64KB I/O address boundary are forwarded as defined by the address range defined by the I/O base and limit registers.

Accordingly, if the ISA enable bit is set, PCI 6254 forwards upstream those I/O transactions addressing the top 768 bytes of each aligned 1KB block within the first 64KB of I/O space. The master enable bit in the command configuration register must also be set to enable upstream forwarding. All other I/O transactions initiated on the secondary bus are forwarded upstream only if they fall outside the I/O address range.

When the ISA enable bit is set, devices downstream of PCI 6254 can have I/O space mapped into the first 256 bytes of each 1KB chunk below the 64KB boundary, or anywhere in I/O space above the 64KB boundary.

## *8.4 Memory Address Decoding*

PCI 6254 has three mechanisms for defining memory address ranges for forwarding of memory transactions:

- Memory-mapped I/O base and limit address registers
- Prefetchable memory base and limit address registers
- VGA mode

This section describes the first two mechanisms.
To enable downstream forwarding of memory transactions, the memory enable bit must be set in the command register in configuration space. To enable upstream forwarding of memory transactions, the master enable bit must be set in the command register. Setting the master enable bit also allows upstream forwarding of I/O transactions.

**Caution**
If any configuration state affecting memory transaction forwarding is changed by a configuration write operation on the primary bus at the same time that memory transactions are ongoing on the secondary bus, response to the secondary bus memory transactions is not predictable. Configure the memory-mapped I/O base and limit address registers, prefetchable memory base and limit address registers, and VGA mode bit before setting the memory enable and master enable bits, and change them subsequently only when the primary and secondary PCI buses are idle.

### 8.4.1 Memory-Mapped I/O Base and Limit Address Registers

Memory-mapped I/O is also referred to as nonprefetchable memory. The memory-mapped I/O base address and memory-mapped I/O limit address registers define an address range that PCI 6254 uses to determine when to forward memory commands. PCI 6254 forwards a memory transaction from the primary to the secondary interface if the transaction address falls within the memory-mapped I/O address range. PCI 6254 ignores memory transactions initiated on the secondary interface that fall into this address range. Any transactions that fall outside this address range are ignored on the primary interface and are forwarded upstream from the secondary interface (provided that they do not fall into the prefetchable memory range or are not forwarded downstream by the VGA mechanism).

The memory-mapped I/O range supports 32-bit addressing only. The PCI-to-PCI Bridge Architecture Specification does not provide for 64-bit addressing in the memory-mapped I/O space. The memory-mapped I/O address range has a granularity and alignment of 1MB. The maximum memory-mapped I/O address range is 4GB.
The memory-mapped I/O address range is defined by a 16-bit memory-mapped I/O base address register at configuration offset 20h and by a 16-bit memory-mapped I/O limit address register at offset 22h. The top 12 bits of each of these registers correspond to bits [31:20] of the memory address. The low 4 bits are hardwired to 0. The low 20 bits of the memory-mapped I/O base address are assumed to be 0 0000h, which results in a natural alignment to a 1MB boundary. The low 20 bits of the memory-mapped I/O limit address are assumed to be F FFFFh, which results in an alignment to the top of a 1MB block.

**Note**
Write these registers with their appropriate values before setting either the memory enable bit or the master enable bit in the command register in configuration space.
To turn off the memory-mapped I/O address range, write the memory-mapped I/O base address register with a value greater than that of the memory-mapped I/O limit address register.

## 8.4.2 Prefetchable Memory Base and Limit Address Registers

Locations accessed in the prefetchable memory address range must have true memory-like behavior and must not exhibit side effects when read. This means that extra reads to a prefetchable memory location must have no side effects. PCI 6254 prefetches for all types of memory read commands in this address space.

PCI 6254 prefetchable memory base address and prefetchable memory limit address registers define an address range that PCI 6254 uses to determine when to forward memory commands. PCI 6254 forwards a memory transaction from the primary to the secondary interface if the transaction address falls within the prefetchable memory address range. PCI 6254 ignores memory transactions initiated on the secondary interface that fall into this address range. PCI 6254 does not respond to any transactions that fall outside this address range on the primary interface and forwards those transactions upstream from the secondary interface (provided that they do not fall into the memory-mapped I/O range or are not forwarded by the VGA mechanism).

PCI 6254 prefetchable memory range supports 64-bit addressing and provides additional registers to define the upper 32 bits of the memory address range, PCI 6254 prefetchable memory base address upper 32 bits register, and the prefetchable memory limit address upper 32 bits register. For address comparison, a single address cycle (32-bit address) prefetchable memory transaction is treated like a 64-bit address transaction where the upper 32 bits of the address are equal to 0. This upper 32-bit value of 0 is compared to the prefetchable memory base address upper 32 bits register and the prefetchable memory limit address upper 32 bits register. The prefetchable memory base address upper 32 bits register must be 0 in order to pass any single address cycle transactions downstream.
The prefetchable memory address range has a granularity and alignment of 1MB. The maximum memory address range is 4GB when 32-bit addressing is used, and $2^{64}$ bytes when 64-bit addressing is used.

The prefetchable memory address range is defined by a 16-bit prefetchable memory base address register at configuration offset 24h and by a 16-bit prefetchable memory limit address register at offset 26h. The top 12 bits of each of these registers correspond to bits [31:20] of the memory address. The low 4 bits are hardwired to 1h, indicating 64-bit address support. The low 20 bits of the prefetchable memory base address are assumed to be 0 0000h, which results in a natural alignment to a 1MB boundary. The low 20 bits of the prefetchable memory limit address are assumed to be F_FFFFh, which results in an alignment to the top of a 1MB block.

**Note**

Write these registers with their appropriate values before setting either the memory enable bit or the master enable bit in the command register in configuration space.

To turn off the prefetchable memory address range, write the prefetchable memory base address register with a value greater than that of the prefetchable memory limit address register. The entire base value must be greater than the entire limit value, meaning that the upper 32 bits must be considered. Therefore, to disable the address range, the upper 32 bits registers can both be set to the same value, while the lower base register is set greater than the lower limit register; otherwise, the upper 32-bit base must be greater than the upper 32-bit limit.

## *8.5 VGA Support*

PCI 6254 provides two modes for VGA support:

- VGA mode, supporting VGA-compatible addressing
- VGA snoop mode, supporting VGA palette forwarding

### 8.5.1 VGA Mode

When a VGA-compatible device exists downstream from PCI 6254, set the VGA mode bit in the bridge control register in configuration space to enable VGA mode. When PCI 6254 is operating in VGA mode, it forwards downstream those transactions addressing the VGA frame buffer memory and VGA I/O registers, regardless of the values of the base and limit address registers. PCI 6254 ignores transactions initiated on the secondary interface addressing these locations.

The VGA frame buffer consists of the following memory address range: 000A_0000h – 000B_FFFFh
Read transactions to frame buffer memory are treated as nonprefetchable. PCI 6254 requests only a single data transfer from the target, and read byte enable bits are forwarded to the target bus.
The VGA I/O addresses consist of the following I/O addresses:

- 3B0h–3BBh
- 3C0h–3DFh

These I/O addresses are aliased every 1KB throughout the first 64KB of I/O space. This means that address bits [15:10] are not decoded and can be any value, while address bits [31:16] must be all 0s.
VGA BIOS addresses starting at C0000h are not decoded in VGA mode.

### 8.5.2 VGA Snoop Mode

PCI 6254 provides VGA snoop mode, Allowing for VGA palette write transactions to be forwarded downstream. This mode is used when a graphics device downstream from PCI 6254 needs to snoop or respond to VGA palette write transactions. To enable the mode, set the VGA snoop bit in the command register in configuration space.
Note that PCI 6254 claims VGA palette write transactions by asserting DEVSEL# in VGA snoop mode.
When the VGA snoop bit is set, PCI 6254 forwards downstream transactions with the following I/O addresses:

- 3C6h
- 3C8h
- 3C9h

Note that these addresses are also forwarded as part of the VGA compatibility mode previously described. Again, address bits [15:10] are not decoded, while address bits [31:16] must be equal to 0, which means that these addresses are aliased every 1KB throughout the first 64KB of I/O space.

**Note**
If both the VGA mode bit and the VGA snoop bit are set, PCI 6254 behaves in the same way as if only the VGA mode bit were set.

## *8.6 Private Device Support*

In Transparent Mode, PCI 6254 can support PCI devices that are not visible at all to primary port hosts or masters.

By connecting XB_MEM input pin to “1”, PCI 6254 enable the use of Private memory at power up. The private memory range need to be setup by driver software. PCI 6254 will not respond to any access to this private memory range on either the primary or the secondary port.

## *8.7 Address Translation*

PCI 6254 has an address translation mechanism. Address translation is supported for both upstream and downstream PCI cycles. When enabled, PCI cycles accessing a specific address range on the initiator bus will pass through to the target bus as the same cycle, but with a different address, as specified by the address translation registers.

### 8.7.1 Base Address Registers

PCI 6254 supports a maximum of three address ranges that can be translated. The resource configuration is done through three base address registers, BAR 0, BAR 1 and BAR 2. Each BAR has a 32-bit translation register and an 8-bit configuration register associated with it. BAR 0 can be configured as a 32-bit I/O or memory BAR. BAR1 and BAR2 are 32-bit bars, but can optionally be configured as a single 64-bit memory BAR. Each BAR follows the standard base address register definition described in the PCI specification. There are two sets of these registers, one for upstream translation and one for downstream translation.

Each base address register has a programmable translation address register. PCI 6254 uses these registers to translate each cycle accessing memory or I/O space specified in one of the base address registers, if translation is enabled.

### 8.7.2 Configuration Address Translation Operation

This section will provide more specific details on programming the address translation registers of the PCI 6254. As an example, it will show the typical sequence that would be performed by the secondary host.
The secondary host first determines which resources it will make accessible to the host on the primary side of the bridge. It can provide any of the following combinations:

- One 32-bit I/O translated address range and one 64-bit Memory translated address range
- One 32-bit I/O translated address range and two 32-bit Memory translated address range
- One 32-bit memory translated address range and one 64-bit Memory translated address range
- Three 32-bit memory translated address range

To specify the resources for the primary host, the secondary host can program the Downstream BAR Translation mask registers. These registers are used by the primary interface to configure its base address registers.

The first field in the translation mask register is used to specify the amount of address space the device requires. The value programmed in this field is interpreted as a bit position into the corresponding BAR in the primary configuration space. When BIOS tries to determine amount of address space requested by the BAR by writing the value 0f FFFFh and reading back the register, the read value will return zeroes in all bit positions above the value specified in the translation mask register. For example, to request 4KB of address space, the BAR should return FFFF_F000h. This would be specified in the translation mask register by programming a value of Bh (1011b) in the ‘MSB position of address mask field’.

Downstream BAR0 translation mask register has a BAR type bit, bit 6, which can be used to specify if BAR0 will specify an I/O or a memory range. This bit becomes reflected in bit 0 of the base address register at offset 10h of the primary configuration space. In addition, bit 7 specifies if the address register points to prefetchable address space or not, and is reflected in bits 1 and 2 of the corresponding address register if specified as a memory range.

Downstream BAR1 translation mask register can only configure a memory base address register in primary configuration register 14h. It can be configured as prefetchable or non-prefetchable through bit 7. In addition to this, bit 6 allows configuration as a 32-bit register or a 64-bit register. If programmed as a 64-bit register, then there is no need to program BAR2 translation mask register.

Downstream BAR2 translation mask register can configure the third base address register as a 32-bit base address register only, and can be selected as prefetchable or non prefetchable address range.

After programming the mask registers, the secondary host programs the Downstream BAR0, 1, and 2 translation address registers. The address programmed here would be the starting address of each of the shared address space on the secondary interface address map.

At this point, the secondary host has completed its programming, and it will then allow the primary side to be configured. The primary side base address registers can be configured by BIOS, but will still need software to enable translation by programming the Downstream translation enable register at primary configuration register.

Alternately, all these registers can be programmed into the EEPROM device, to be autoloaded during power-up.

# 9 Transaction Ordering

To maintain data coherency and consistency, PCI 6254 complies with the ordering rules set forth in the PCI Local Bus Specification, Revision 2.2.

## *9.1 Transaction Ordering*

This section describes the ordering rules that control transaction forwarding across PCI 6254. For a more detailed discussion of transaction ordering, see Appendix E of the PCI Local Bus Specification, Revision 2.2.

### 9.1.1 Transactions Governed by Ordering Rules

Ordering relationships are established for the following classes of transactions crossing PCI 6254:

- Posted write transactions, comprised of memory write and memory write and invalidate transactions
  Posted write transactions complete at the source before they complete at the destination; that is, data is written into intermediate data buffers before it reaches the target.
- Delayed write request transactions, comprised of I/O write and configuration write transactions
  Delayed write requests are terminated by target retry on the initiator bus and are queued in the delayed transaction queue. A delayed write transaction must complete on the target bus before it completes on the initiator bus.
- Delayed write completion transactions, also comprised of I/O write and configuration write transactions.
  Delayed write completion transactions have been completed on the target bus, and the target response is queued in the buffers. A delayed write completion transaction proceeds in the direction opposite that of the original delayed write request; that is, a delayed write completion transaction proceeds from the target bus to the initiator bus.
- Delayed read request transactions, comprised of all memory read, I/O read, and configuration read transactions.
  Delayed read requests are terminated by target retry on the initiator bus and are queued in the delayed transaction queue.
- Delayed read completion transactions, comprised of all memory read, I/O read, and configuration read transactions.
  Delayed read completion transactions have been completed on the target bus, and the read data has been queued in the read data buffers. A delayed read completion transaction proceeds in the direction opposite that of the original delayed read request; that is, a delayed read completion transaction proceeds from the target's bus to the initiator's bus.

PCI 6254 does not combine or merge write transactions:

- PCI 6254 does not combine separate write transactions into a single write transaction—this optimization is best implemented in the originating master.
- PCI 6254 does not merge bytes on separate masked write transactions to the same DWORD address—this optimization is also best implemented in the originating master.
- PCI 6254 does not collapse sequential write transactions to the same address into a single write transaction—the PCI Local Bus Specification does not permit this combining of transactions.

### 9.1.2 General Ordering Guidelines

Independent transactions on the primary and secondary buses have a relationship only when those transactions cross PCI 6254.

The following general ordering guidelines govern transactions crossing PCI 6254:

- The ordering relationship of a transaction with respect to other transactions is determined when the transaction completes, that is, when a transaction ends with a termination other than target retry.
- Requests terminated with target retry can be accepted and completed in any order with respect to other transactions that have been terminated with target retry. If the order of completion of delayed requests is important, the initiator should not start a second delayed transaction until the first one has been completed. If more than one delayed transaction is initiated, the initiator should repeat all the delayed transaction requests, using some fairness algorithm. Repeating a delayed transaction cannot be contingent on completion of another delayed transaction; otherwise, a deadlock can occur.

- Write transactions flowing in one direction have no ordering requirements with respect to write transactions flowing in the other direction. PCI 6254 can accept posted write transactions on both interfaces at the same time, as well as initiate posted write transactions on both interfaces at the same time.
- The acceptance of a posted memory write transaction as a target can never be contingent on the completion of a non-locked, nonposted transaction as a master. This is true of PCI 6254 and must also be true of other bus agents; otherwise, a deadlock can occur.
- PCI 6254 accepts posted write transactions, regardless of the state of completion of any delayed transactions being forwarded across PCI 6254.

## 9.1.3 Ordering Rules

Table 9–1 shows the ordering relationships of all the transactions and refers by number to the ordering rules that follow.

**Table 9–1: Summary of Transaction Ordering**

| Pass | Posted Write | Delayed read Request | Delayed Write Request | Delayed Read Completion | Delayed Write Completion |
|---|---|---|---|---|---|
| Posted write | N[1] | Y[5] | Y[5] | Y[5] | Y[5] |
| Delayed read request | N[2] | Y | Y | Y | Y |
| Delayed write request | N[4] | Y | Y | Y | Y |
| Delayed read completion | N[3] | Y | Y | Y | Y |
| Delayed write completion | Y | Y | Y | Y | Y |

**Note**
The superscript accompanying some of the table entries refers to any applicable ordering rule listed in this section. Many entries are not governed by these ordering rules; therefore, the implementation can choose whether or not the transactions pass each other.

**The entries without superscripts reflect PCI 6254's implementation choices.**
The following ordering rules describe the transaction relationships. Each ordering rule is followed by an explanation, and the ordering rules are referred to by number in Table 9–1. These ordering rules apply to posted write transactions, delayed write and read requests, and delayed write and read completion transactions crossing PCI 6254 in the same direction. Note that delayed completion transactions cross PCI 6254 in the direction opposite that of the corresponding delayed requests.

1. Posted write transactions must complete on the target bus in the order in which they were received on the initiator bus.
   The subsequent posted write transaction can be setting a flag that covers the data in the first posted write transaction; if the second transaction were to complete before the first transaction, a device checking the flag could subsequently consume stale data.
2. A delayed read request traveling in the same direction as a previously queued posted write transaction must push the posted write data ahead of it. The posted write transaction must complete on the target bus before the delayed read request can be attempted on the target bus.
   The read transaction can be to the same location as the write data, so if the read transaction were to pass the write transaction, it would return stale data.
3. A delayed read completion must ''pull'' ahead of previously queued posted write data traveling in the same direction. In this case, the read data is traveling in the same direction as the write data, and the initiator of the read transaction is on the same side of the  as the target of the write transaction. The posted write transaction must complete to the target before the read data is returned to the initiator.
   The read transaction can be to a status register of the initiator of the posted write data and therefore should not complete until the write transaction is complete.
4. Delayed write requests cannot pass previously queued posted write data. As in the case of posted memory write transactions, the delayed write transaction can be setting a flag that covers the data in the posted write transaction; if the delayed write request were to complete before the earlier posted write transaction, a device checking the flag could subsequently consume stale data.

5. Posted write transactions must be given opportunities to pass delayed read and write requests and completions. Otherwise, deadlocks may occur when bridges that support delayed transactions are used in the same system with bridges that do not support delayed transactions. A fairness algorithm is used to arbitrate between the posted write queue and the delayed transaction queue.
   PCI 6254 can generate cycles across the bridge in the same order than requested if bit 2 of register 46h is set. By default, requested cycles can execute out of order across the bridge, if all other ordering rules are satisfied. So, if PCI 6254 starts a delayed transaction that is retried by the target, it can execute another transaction in the delayed transaction request queue. Also, if there is both delayed read and delayed write requests in the queue, and the read data FIFO's are full, PCI 6254 may execute the delayed write request before the delayed read request.
   On cycle completion, PCI 6254 may complete cycles in a different order than requested by the initiator.

### 9.1.4 Data Synchronization

Data synchronization refers to the relationship between interrupt signaling and data delivery. The PCI Local Bus Specification, Revision 2.2, provides the following alternative methods for synchronizing data and interrupts:

- The device signaling the interrupt performs a read of the data just written (software).
- The device driver performs a read operation to any register in the interrupting device before accessing data written by the device (software).
- System hardware guarantees that write buffers are flushed before interrupts are forwarded.

PCI 6254 does not have a hardware mechanism to guarantee data synchronization for posted write transactions. Therefore, all posted write transactions must be followed by a read operation, either from the device to the location just written (or some other location along the same path), or from the device driver to one of the device registers.

# 10 Error Handling

PCI 6254 checks, forwards, and generates parity on both the primary and secondary interfaces. To maintain transparency, PCI 6254 always tries to forward the existing parity condition on one bus to the other bus, along with address and data. PCI 6254 always attempts to be transparent when reporting errors, but this is not always possible, given the presence of posted data and delayed transactions.

To support error reporting on the PCI bus, PCI 6254 implements the following:

- PERR# and SERR# signals on both the primary and secondary interfaces
- Primary status and secondary status registers
- The device-specific P_SERR# event disable register
- The device-specific P_SERR# status register
- For Non-Transparent Mode, the device-specific S_SERR# event disable register
- For Non-Transparent Mode, the device-specific S_SERR# status register

This chapter provides detailed information about how PCI 6254 handles errors. It also describes error status reporting and error operation disabling.

## *10.1 Address Parity Errors*

PCI 6254 checks address parity for all transactions on both buses, for all address and all bus commands.
When PCI 6254 detects an address parity error on the primary interface, the following events occur:

- If the parity error response bit is set in the command register, PCI 6254 does not claim the transaction with P_DEVSEL#; this may allow the transaction to terminate in a Master-Abort. If the parity error response bit is not set, PCI 6254 proceeds normally and accepts the transaction if it is directed to or across PCI 6254.
- PCI 6254 sets the detected parity error bit in the status register.
- PCI 6254 asserts P_SERR# and sets the signaled system error bit in the status register, if both of the following conditions are met:
  - The SERR# enable bit is set in the command register.
  - The parity error response bit is set in the command register.

When PCI 6254 detects an address parity error on the secondary interface, the following events occur:

- If the parity error response bit is set in the bridge control register, PCI 6254 does not claim the transaction with S_DEVSEL#; this may allow the transaction to terminate in a master abort. If the parity error response bit is not set, PCI 6254 proceeds normally and accepts the transaction if it is directed to or across PCI 6254.
- PCI 6254 sets the detected parity error bit in the secondary status register.
- PCI 6254 asserts P_SERR# and sets the signaled system error bit in the status register, if both of the following conditions are met:
  - The SERR# enable bit is set in the command register.
  - The parity error response bit is set in the bridge control register.

## *10.2 Data Parity Errors*

When forwarding transactions, PCI 6254 attempts to pass the data parity condition from one interface to the other unchanged, whenever possible, to Allow the master and target devices to handle the error condition.
The following sections describe, for each type of transaction, the sequence of events that occurs when a parity error is detected and the way in which the parity condition is forwarded across the bridge.

### 10.2.1 Configuration Write Transactions to Configuration Space

When PCI 6254 detects a data parity error during a Type-0 configuration write transaction to configuration space, the following events occur:

- If the parity error response bit is set in the command register, PCI 6254 asserts P_TRDY# and writes the data to the configuration register. PCI 6254 also asserts P_PERR#.
  If the parity error response bit is not set, PCI 6254 does not assert P_PERR#.
- PCI 6254 sets the detected parity error bit in the status register, regardless of the state of the parity error response bit.

### 10.2.2 Read Transactions

When PCI 6254 detects a parity error during a read transaction, the target drives data and data parity, and the initiator checks parity and conditionally asserts PERR#.

For downstream transactions, when PCI 6254 detects a read data parity error on the secondary bus, the following events occur:

- PCI 6254 asserts S_PERR# two cycles following the data transfer, if the secondary interface parity error response bit is set in the bridge control register.
- PCI 6254 sets the detected parity error bit in the secondary status register.
- PCI 6254 sets the data parity detected bit in the secondary status register, if the secondary interface parity error response bit is set in the bridge control register.
- PCI 6254 forwards the bad parity with the data back to the initiator on the primary bus.
  If the data with the bad parity is prefetched and is not read by the initiator on the primary bus, the data is discarded and the data with bad parity is not returned to the initiator.
- PCI 6254 completes the transaction normally. For upstream transactions, when PCI 6254 detects a read data parity error on the primary bus, the following events occur:
- PCI 6254 asserts P_PERR# two cycles following the data transfer, if the primary interface parity error response bit is set in the command register.
- PCI 6254 sets the detected parity error bit in the primary status register.
- PCI 6254 sets the data parity detected bit in the primary status register, if the primary interface parity error response bit is set in the command register.
- PCI 6254 forwards the bad parity with the data back to the initiator on the secondary bus.
  If the data with the bad parity is prefetched and is not read by the initiator on the secondary bus, the data is discarded and the data with bad parity is not returned to the initiator.
- PCI 6254 completes the transaction normally.

PCI 6254 returns to the initiator the data and parity that was received from the target. When the initiator detects a parity error on this read data and is enabled to report it, the initiator asserts PERR# two cycles after the data transfer occurs. It is assumed that the initiator takes responsibility for handling a parity error condition; therefore, when PCI 6254 detects PERR# asserted while returning read data to the initiator, PCI 6254 does not take any further action and completes the transaction normally.

### 10.2.3 Delayed Write Transactions

When PCI 6254 detects a data parity error during a delayed write transaction, the initiator drives data and data parity, and the target checks parity and conditionally asserts PERR#.

For delayed write transactions, a parity error can occur at the following times:

- During the original delayed write request transaction
- When the initiator repeats the delayed write request transaction
- When PCI 6254 completes the delayed write transaction to the target

When a delayed write transaction is normally queued, the address, command, address parity, data, byte enable bits, and data parity are all captured and a target retry is returned to the initiator. When PCI 6254 detects a parity error on the write data for the initial delayed write request transaction, the following events occur:

- If the parity error response bit corresponding to the initiator bus is set, PCI 6254 asserts TRDY# to the initiator and the transaction is not queued. If multiple data phases are requested, STOP# is also asserted to cause a target disconnect. Two cycles after the data transfer, PCI 6254 also asserts PERR#.
  If the parity error response bit is not set, PCI 6254 returns a target retry and queues the transaction as usual. Signal PERR# is not asserted. In this case, the initiator repeats the transaction.
- PCI 6254 sets the detected parity error bit in the status register corresponding to the initiator bus, regardless of the state of the parity error response bit.

**Note**

If parity checking is turned off and data parity errors have occurred for queued or subsequent delayed write transactions on the initiator bus, it is possible that the initiator's reattempts of the write transaction may not match the original queued delayed write information contained in the delayed transaction queue. In this case, a master timeout condition may occur, possibly resulting in a system error (P_SERR# assertion).

For downstream transactions, when PCI 6254 is delivering data to the target on the secondary bus and S_PERR# is asserted by the target, the following events occur:

- PCI 6254 sets the secondary interface data parity detected bit in the secondary status register, if the secondary parity error response bit is set in the bridge control register.
- PCI 6254 captures the parity error condition to forward it back to the initiator on the primary bus.

Similarly, for upstream transactions, when the is delivering data to the target on the primary bus and P_PERR# is asserted by the target, the following events occur:

- PCI 6254 sets the primary interface data parity detected bit in the status register, if the primary parity error response bit is set in the command register.
- PCI 6254 captures the parity error condition to forward it back to the initiator on the secondary bus.

A delayed write transaction is completed on the initiator bus when the initiator repeats the write transaction with the same address, command, data, and byte enable bits as the delayed write command that is at the head of the posted data queue. Note that the parity bit is not compared when determining whether the transaction matches those in the delayed transaction queues.

Two cases must be considered:

- When parity error is detected on the initiator bus on a subsequent reattempt of the transaction and was not detected on the target bus
- When parity error is forwarded back from the target bus
- For downstream delayed write transactions, when the parity error is detected on the initiator bus and PCI 6254 has write status to return, the following events occur:
- PCI 6254 first asserts P_TRDY# and then asserts P_PERR# two cycles later, if the primary interface parity error response bit is set in the command register.
- PCI 6254 sets the primary interface parity error detected bit in the status register.
- Because there was not an exact data and parity match, the write status is not returned and the transaction remains in the queue.

Similarly, for upstream delayed write transactions, when the parity error is detected on the initiator bus and PCI 6254 has write status to return, the following events occur:

- PCI 6254 first asserts S_TRDY# and then asserts S_PERR# two cycles later, if the secondary interface parity error response bit is set in the bridge control register.
- PCI 6254 sets the secondary interface parity error detected bit in the secondary status register.
- Because there was not an exact data and parity match, the write status is not returned and the transaction remains in the queue.

For downstream transactions, in the case where the parity error is being passed back from the target bus and the parity error condition was not originally detected on the initiator bus, the following events occur:

- PCI 6254 asserts P_PERR# two cycles after the data transfer, if both of the following are true:
  - The primary interface parity error response bit is set in the command register.
  - The secondary interface parity error response bit is set in the bridge control register.
- PCI 6254 completes the transaction normally.

For upstream transactions, in the case where the parity error is being passed back from the target bus and the parity error condition was not originally detected on the initiator bus, the following events occur:

- PCI 6254 asserts S_PERR# two cycles after the data transfer, if both of the following are true:
  - The primary interface parity error response bit is set in the command register.
  - The secondary interface parity error response bit is set in the bridge control register.
- PCI 6254 completes the transaction normally.

### 10.2.4 Posted Write Transactions

During downstream posted write transactions, when the PCI 6254, responding as a target, detects a data parity error on the initiator (primary) bus, the following events occur:

- PCI 6254 asserts P_PERR# two cycles after the data transfer, if the primary interface parity error response bit is set in the command register.
- PCI 6254 sets the primary interface parity error detected bit in the status register.
- PCI 6254 captures and forwards the bad parity condition to the secondary bus.
- PCI 6254 completes the transaction normally.

Similarly, during upstream posted write transactions, when the PCI 6254, responding as a target, detects a data parity error on the initiator (secondary) bus, the following events occur:

- PCI 6254 asserts S_PERR# two cycles after the data transfer, if the secondary interface parity error response bit is set in the bridge control register.
- PCI 6254 sets the secondary interface parity error detected bit in the secondary status register.
- PCI 6254 captures and forwards the bad parity condition to the primary bus.
- PCI 6254 completes the transaction normally.

During downstream write transactions, when a data parity error is reported on the target (secondary) bus by the target's assertion of S_PERR#, the following events occur:

- PCI 6254 sets the data parity detected bit in the secondary status register, if the secondary interface parity error response bit is set in the bridge control register.
- PCI 6254 asserts P_SERR# and sets the signaled system error bit in the status register, if all of the following conditions are met:
  - The SERR# enable bit is set in the command register.
  - The device-specific P_SERR# disable bit for posted write parity errors is not set.
  - The secondary interface parity error response bit is set in the bridge control register.
  - The primary interface parity error response bit is set in the command register.
  - PCI 6254 did not detect the parity error on the primary (initiator) bus; that is, the parity error was not forwarded from the primary bus.

During upstream write transactions, when a data parity error is reported on the target (primary) bus by the target's assertion of P_PERR#, the following events occur:

- PCI 6254 sets the data parity detected bit in the status register, if the primary interface parity error response bit is set in the command register.
- PCI 6254 asserts P_SERR# and sets the signaled system error bit in the status register, if all of the following conditions are met:
  - The SERR# enable bit is set in the command register.
  - The secondary interface parity error response bit is set in the bridge control register.
  - The primary interface parity error response bit is set in the command register.
  - PCI 6254 did not detect the parity error on the secondary (initiator) bus; that is, the parity error was not forwarded from the secondary bus.

The assertion of P_SERR# is used to signal the parity error condition in the case where the initiator does not know that the error occurred. Because the data has already been delivered with no errors, there is no other way to signal this information back to the initiator.
If the parity error was forwarded from the initiating bus to the target bus, P_SERR# is not asserted.

## *10.3 Data Parity Error Reporting Summary*

In the previous sections, PCI 6254's responses to data parity errors are presented according to the type of transaction in progress. This section organizes PCI 6254's responses to data parity errors according to the status bits that PCI 6254 sets and the signals that it asserts.

Table 10–1 shows setting the detected parity error bit in the status register, corresponding to the primary interface. This bit is set when PCI 6254 detects a parity error on the primary interface.

**Table 10–1: Setting the Primary Interface Detected Parity Error Bit**

| Primary detected parity error bit | Transaction type | Direction | Bus where error was detected | Prim./Sec. parity error response bits |
|---|---|---|---|---|
| 0 | Read | Downstream | Primary | x/x[1] |
| 0 | Read | Downstream | Secondary | x/x |
| 1 | Read | Upstream | Primary | x/x |
| 0 | Read | Upstream | Secondary | x/x |
| 1 | Posted write | Downstream | Primary | x/x |
| 0 | Posted write | Downstream | Secondary | x/x |
| 0 | Posted write | Upstream | Primary | x/x |
| 0 | Posted write | Upstream | Secondary | x/x |
| 1 | Delayed write | Downstream | Primary | x/x |
| 0 | Delayed write | Downstream | Secondary | x/x |
| 0 | Delayed write | Upstream | Primary | x/x |
| 0 | Delayed write | Upstream | Secondary | x/x |

[1] x = don't care

Table 10–2 shows setting the detected parity error bit in the secondary status register, corresponding to the secondary interface. This bit is set when PCI 6254 detects a parity error on the secondary interface.

**Table 10–2: Setting the Secondary Interface Detected Parity Error Bit**

| Sec. Detected Parity Error Bit | Transaction Type | Direction | Bus Where Error Was Detected | Prim./Sec. parity error response bits |
|---|---|---|---|---|
| 0 | Read | Downstream | Primary | x/x[1] |
| 1 | Read | Downstream | Secondary | x/x |
| 0 | Read | Upstream | Primary | x/x |
| 0 | Read | Upstream | Secondary | x/x |
| 0 | Posted write | Downstream | Primary | x/x |
| 0 | Posted write | Downstream | Secondary | x/x |
| 0 | Posted write | Upstream | Primary | x/x |
| 1 | Posted write | Upstream | Secondary | x/x |
| 0 | Delayed write | Downstream | Primary | x/x |
| 0 | Delayed write | Downstream | Secondary | x/x |
| 0 | Delayed write | Upstream | Primary | x/x |
| 1 | Delayed write | Upstream | Secondary | x/x |

[1]x = don't care

Table 10–3 shows setting the data parity detected bit in the status register, corresponding to the primary interface. This bit is set under the following conditions:

- PCI 6254 must be a master on the primary bus.
- The parity error response bit in the command register, corresponding to the primary interface, must be set.
- The P_PERR# signal is detected asserted or a parity error is detected on the primary bus.

**Table 10–3: Setting the Primary Interface Data Parity Detected Bit**

| Primary data parity detected bit | Transaction type | Direction | Bus where error was detected | Prim./Sec. parity error response bits |
|---|---|---|---|---|
| 0 | Read | Downstream | Primary | x/x[1] |
| 0 | Read | Downstream | Secondary | x/x |
| 1 | Read | Upstream | Primary | 1/x |
| 0 | Read | Upstream | Secondary | x/x |
| 0 | Posted write | Downstream | Primary | x/x |
| 0 | Posted write | Downstream | Secondary | x/x |
| 1 | Posted write | Upstream | Primary | 1/x |
| 0 | Posted write | Upstream | Secondary | x/x |
| 0 | Delayed write | Downstream | Primary | x/x |
| 0 | Delayed write | Downstream | Secondary | x/x |
| 1 | Delayed write | Upstream | Primary | 1/x |
| 0 | Delayed write | Upstream | Secondary | x/x |

[1]x = don't care

Table 10–4 shows setting the data parity detected bit in the secondary status register, corresponding to the secondary interface. This bit is set under the following conditions:

- PCI 6254 must be a master on the secondary bus.
- The parity error response bit in the bridge control register, corresponding to the secondary interface, must be set.
- The S_PERR# signal is detected asserted or a parity error is detected on the secondary bus.

**Table 10–4: Setting the Secondary Interface Data Parity Detected Bit**

| Secondary data parity detected bit | Transaction type | Direction | Bus where error was detected | Prim./Sec. parity error response bits |
|---|---|---|---|---|
| 0 | Read | Downstream | Primary | x/x[1] |
| 1 | Read | Downstream | Secondary | x/1 |
| 0 | Read | Upstream | Primary | x/x |
| 0 | Read | Upstream | Secondary | x/x |
| 0 | Posted write | Downstream | Primary | x/x |
| 1 | Posted write | Downstream | Secondary | x/1 |
| 0 | Posted write | Upstream | Primary | x/x |
| 0 | Posted write | Upstream | Secondary | x/x |
| 0 | Delayed write | Downstream | Primary | x/x |
| 1 | Delayed write | Downstream | Secondary | x/1 |
| 0 | Delayed write | Upstream | Primary | x/x |
| 0 | Delayed write | Upstream | Secondary | x/x |

[1]x =don't care

Table 10–5 shows assertion of P_PERR#. This signal is set under the following conditions:

- PCI 6254 is either the target of a write transaction or the initiator of a read transaction on the primary bus.
- The parity error response bit in the command register, corresponding to the primary interface, must be set.
- PCI 6254 detects a data parity error on the primary bus or detects S_PERR# asserted during the completion phase of a downstream delayed write transaction on the target (secondary) bus.

**Table 10–5: Assertion of P_PERR#**

| P_PERR# | Transaction type | Direction | Bus where error was detected | Prim./Sec. parity error response bits |
|---|---|---|---|---|
| 1 (deasserted) | Read | Downstream | Primary | x/x[1] |
| 1 | Read | Downstream | Secondary | x/x |
| 0 (asserted) | Read | Upstream | Primary | 1/x |
| 1 | Read | Upstream | Secondary | x/x |
| 0 | Posted write | Downstream | Primary | 1/x |
| 1 | Posted write | Downstream | Secondary | x/x |
| 1 | Posted write | Upstream | Primary | x/x |
| 1 | Posted write | Upstream | Secondary | x/x |
| 0 | Delayed write | Downstream | Primary | 1/x |
| 0[2] | Delayed write | Downstream | Secondary | 1/1 |
| 1 | Delayed write | Upstream | Primary | x/x |
| 1 | Delayed write | Upstream | Secondary | x/x |

[1]x = don't care

[2]The parity error was detected on the target (secondary) bus but not on the initiator (primary) bus.

Table 10–6 shows assertion of S_PERR#. This signal is set under the following conditions:

- PCI 6254 is either the target of a write transaction or the initiator of a read transaction on the secondary bus.
- The parity error response bit in the bridge control register, corresponding to the secondary interface, must be set.
- PCI 6254 detects a data parity error on the secondary bus or detects P_PERR# asserted during the completion phase of an upstream delayed write transaction on the target (primary) bus.

**Table 10-6: Assertion of S_PERR#**

| S_PERR# | Transaction type | Direction | Bus where error was detected | Prim./Sec. parity error response bits |
|---|---|---|---|---|
| 1 (deasserted) | Read | Downstream | Primary | x/x[1] |
| 0 (asserted) | Read | Downstream | Secondary | x/1 |
| 1 | Read | Upstream | Primary | x/x |
| 1 | Read | Upstream | Secondary | x/x |
| 1 | Posted write | Downstream | Primary | x/x |
| 1 | Posted write | Downstream | Secondary | x/x |
| 1 | Posted write | Upstream | Primary | x/x |
| 0 | Posted write | Upstream | Secondary | x/1 |
| 1 | Delayed write | Downstream | Primary | x/x |
| 1 | Delayed write | Downstream | Secondary | x/x |
| 0[2] | Delayed write | Upstream | Primary | 1/1 |
| 0 | Delayed write | Upstream | Secondary | x/1 |

[1]x =don't care
[2]The parity error was detected on the target (secondary) bus but not on the initiator (primary) bus.

Table 10–7 shows assertion of P_SERR#. This signal is set under the following conditions:

- PCI 6254 has detected P_PERR# asserted on an upstream posted write transaction or S_PERR# asserted on a downstream posted write transaction.
- PCI 6254 did not detect the parity error as a target of the posted write transaction.
- The parity error response bit on the command register and the parity error response bit on the bridge control register must both be set.
- The SERR# enable bit must be set in the command register.

**Table 10–7: Assertion of P_SERR# for Data Parity Errors**

| P_PERR# | Transaction Type | Direction | Bus where error was detected | Prim./Sec. parity error response bits |
|---|---|---|---|---|
| 1 (deasserted) | Read | Downstream | Primary | x/x[1] |
| 1 | Read | Downstream | Secondary | x/x |
| 1 | Read | Upstream | Primary | x/x |
| 1 | Read | Upstream | Secondary | x/x |
| 1 | Posted write | Downstream | Primary | x/x |
| 0[2] (asserted) | Posted write | Downstream | Secondary | 1/1 |
| 0[3] | Posted write | Upstream | Primary | 1/1 |
| 1 | Posted write | Upstream | Secondary | x/x |
| 1 | Delayed write | Downstream | Primary | x/x |
| 1 | Delayed write | Downstream | Secondary | x/x |
| 1 | Delayed write | Upstream | Primary | x/x |
| 1 | Delayed write | Upstream | Secondary | x/x |

[1]x = don't care
[2]The parity error was detected on the target (secondary) bus but not on the initiator (primary) bus.
[3]The parity error was detected on the target (primary) bus but not on the initiator (secondary) bus.

## *10.4 System Error (SERR#) Reporting*

Whenever the assertion of P_SERR# is discussed in this document, it is assumed that the following conditions apply:

- For PCI 6254 to assert P_SERR# for any reason, the SERR# enable bit must be set in the command register.
- Whenever PCI 6254 asserts P_SERR#, PCI 6254 must also set the signaled system error bit in the status register.

In compliance with the PCI-to-PCI Bridge Architecture Specification, PCI 6254 asserts P_SERR# when it detects the secondary SERR# input, S_SERR#, asserted and the SERR# forward enable bit is set in the bridge control register. In addition, PCI 6254 also sets the received system error bit in the secondary status register.
PCI 6254 also conditionally asserts P_SERR# for any of the following reasons:

- Target abort detected during posted write transaction
- Master abort detected during posted write transaction
- Posted write data discarded after $2^{24}$ attempts to deliver ($2^{24}$ target retries received)
- Parity error reported on target bus during posted write transaction (see previous section)
- Delayed write data discarded after $2^{24}$ attempts to deliver ($2^{24}$ target retries received)
- Delayed read data cannot be transferred from target after $2^{24}$ attempts ( $2^{24}$ target retries received)
- Master timeout on delayed transaction

The device-specific P_SERR# status register reports the reason for the assertion of P_SERR#.
Most of these events have additional device-specific disable bits in the P_SERR# event disable register that make it possible to mask out P_SERR# assertion for specific events. The master timeout condition has a SERR# enable bit for that event in the bridge control register and therefore does not have a device-specific disable bit.

# 11 Exclusive Access

This chapter describes the use of the LOCK# signal to implement exclusive access to a target for transactions that cross PCI 6254. **Current revision of the PCI 6254 does not support LOCKed BURST READ cycles**.

## *11.1 Concurrent Locks*

The primary and secondary bus lock mechanisms operate concurrently except when a locked transaction crosses PCI 6254. A primary master can lock a primary target without affecting the status of the lock on the secondary bus, and vice versa. This means that a primary master can lock a primary target at the same time that a secondary master locks a secondary target.

## *11.2 Acquiring Exclusive Access Across PCI 6254*

For any PCI bus, before acquiring access to the LOCK# signal and starting a series of locked transactions, the initiator must first check that both of the following conditions are met:

- The PCI bus must be idle.
- The LOCK# signal must be deasserted.

The initiator leaves the LOCK# signal deasserted during the address phase and asserts LOCK# one clock cycle later. Once a data transfer is completed from the target, the target lock has been achieved.
Locked transactions can cross PCI 6254 in the downstream and upstream directions, from the primary bus to the secondary bus and vice versa.

When the target resides on another PCI bus, the master must acquire not only the lock on its own PCI bus but also the lock on every bus between its bus and the target's bus. When PCI 6254 detects, on the primary bus, an initial locked transaction intended for a target on the secondary bus, PCI 6254 samples the address, transaction type, byte enable bits, and parity. It also samples the lock signal. Because a target retry is signaled to the initiator, the initiator must relinquish the lock on the primary bus, and therefore the lock is not yet established.

The first locked transaction must be a read transaction. Subsequent locked transactions can be read or write transactions. Posted Memory Write transactions that are a part of the locked transaction sequence are still posted. Memory read transactions that are a part of the locked transaction sequence are not prefetched.

When the locked delayed read request is queued, PCI 6254 does not queue any more transactions until the locked sequence is finished. PCI 6254 signals a target retry to all transactions initiated subsequent to the locked read transaction that are intended for targets on the other side of PCI 6254. PCI 6254 allows any transactions queued before the locked transaction to complete before initiating the locked transaction.

When the locked delayed read request transaction moves to the head of the delayed transaction queue, PCI 6254 initiates the transaction as a locked read transaction by deasserting LOCK# on the target bus during the first address phase, and by asserting LOCK# one cycle later. If LOCK# is already asserted (used by another initiator), PCI 6254 waits to request access to the secondary bus until LOCK# is sampled deasserted when the target bus is idle. Note that the existing lock on the target bus could not have crossed PCI 6254; otherwise, the pending queued locked transaction would not have been queued. When PCI 6254 is able to complete a data transfer with the locked read transaction, the lock is established on the secondary bus.

When the initiator repeats the locked read transaction on the primary bus with the same address, transaction type, and byte enable bits, PCI 6254 transfers the read data back to the initiator, and the lock is then also established on the primary bus.

For PCI 6254 to recognize and respond to the initiator, the initiator's subsequent attempts of the read transaction must use the locked transaction sequence (deassert LOCK# during address phase, and assert LOCK# one cycle later). If the LOCK# sequence is not used in subsequent attempts, a master timeout condition may result. When a master timeout condition occurs, SERR# is conditionally asserted, the read data and queued read transaction are discarded, and the LOCK# signal is deasserted on the target bus.

Once the intended target has been locked, any subsequent locked transactions initiated on the initiator bus that are forwarded by PCI 6254 are driven as locked transactions on the target bus.

When PCI 6254 receives a target abort or a master abort in response to the delayed locked read transaction, this status is passed back to the initiator, and no locks are established on either the target or the initiator bus. PCI 6254 resumes forwarding unlocked transactions in both directions.

## *11.3 Ending Exclusive Access*

After the lock has been acquired on both the initiator and target buses, PCI 6254 must maintain the lock on the target bus for any subsequent locked transactions until the initiator relinquishes the lock.

The only time a target retry causes the lock to be relinquished is on the first transaction of a locked sequence. On subsequent transactions in the sequence, the target retry has no effect on the status of the lock signal.

An established target lock is maintained until the initiator relinquishes the lock. PCI 6254 does not know whether the current transaction is the last one in a sequence of locked transactions until the initiator deasserts the LOCK# signal at the end of the transaction.

When the last locked transaction is a delayed transaction, PCI 6254 has already completed the transaction on the secondary bus. In this case, as soon as PCI 6254 detects that the initiator has relinquished the LOCK# signal by sampling it in the deasserted state while FRAME# is deasserted, PCI 6254 deasserts the LOCK# signal on the target bus as soon as possible. Because of this behavior, LOCK# may not be deasserted until several cycles after the last locked transaction has been completed on the target bus. As soon as PCI 6254 has deasserted LOCK# to indicate the end of a sequence of locked transactions, it resumes forwarding unlocked transactions.

When the last locked transaction is a posted write transaction, PCI 6254 deasserts LOCK# on the target bus at the end of the transaction because the lock was relinquished at the end of the write transaction on the initiator bus.

When PCI 6254 receives a target abort or a master abort in response to a locked delayed transaction, PCI 6254 returns a target abort or a master abort when the initiator repeats the locked transaction. The initiator must then deassert LOCK# at the end of the transaction. PCI 6254 sets the appropriate status bits, flagging the abnormal target termination condition. Normal forwarding of unlocked posted and delayed transactions is resumed.

When PCI 6254 receives a target abort or a master abort in response to a locked posted write transaction, PCI 6254 cannot pass back that status to the initiator. PCI 6254 asserts SERR# on the initiator bus when a target abort or a master abort is received during a locked posted write transaction, if the SERR# enable bit is set in the command register. Signal SERR# is asserted for the master abort condition if the master abort mode bit is set in the bridge control register.

**Note:**
PCI 6254 has an option to ignore the lock protocol, through register 46h, bits 13 and 14.

# 12 PCI Bus Arbitration

PCI 6254 must arbitrate for use of the primary bus when forwarding upstream transactions, and for use of the secondary bus when forwarding downstream transactions. The arbiter for the primary bus resides external to PCI 6254, typically on the motherboard. For the secondary PCI bus, PCI 6254 implements an internal arbiter. This arbiter can be disabled, and an external arbiter can be used instead.

This chapter describes primary and secondary bus arbitration.

## *12.1 Primary PCI Bus Arbitration*

PCI 6254 implements a request output pin, P_REQ#, and a grant input pin, P_GNT#, for primary PCI bus arbitration. PCI 6254 asserts P_REQ# when forwarding transactions upstream; that is, it acts as initiator on the primary PCI bus. As long as at least one pending transaction resides in the queues in the upstream direction, either posted write data or delayed transaction requests, PCI 6254 keeps P_REQ# asserted. However, if a target retry, target disconnect, or a target abort is received in response to a transaction initiated by PCI 6254 on the primary PCI bus, PCI 6254 deasserts P_REQ# for two PCI clock cycles. For all cycles through the bridge, P_REQ# is not asserted until the transaction request has been completely queued.

If P_GNT# is asserted (by the primary bus arbiter after PCI 6254 has asserted P_REQ#), PCI 6254 initiates a transaction on the primary bus during the next PCI clock cycle. When P_GNT# is asserted to PCI 6254 when P_REQ# is not asserted, PCI 6254 parks P_AD, P_CBE, and P_PAR by driving them to valid logic levels. When the primary bus is parked at PCI 6254 and PCI 6254 then has a transaction to initiate on the primary bus, PCI 6254 starts the transaction if P_GNT# was asserted during the previous cycle.

## *12.2 Secondary PCI Bus Arbitration*

PCI 6254 implements an internal secondary PCI bus arbiter. This arbiter supports nine external masters in addition to PCI 6254. The internal arbiter can be disabled, and an external arbiter can be used instead for secondary bus arbitration.

### 12.2.1 Secondary Bus Arbitration Using the Internal Arbiter

To use the internal arbiter, the secondary bus arbiter enable pin, S_CFN#, must be tied LOW. PCI 6254 has nine secondary bus request input pins, S_REQ#[8:0], and nine secondary bus output grant pins, S_GNT#[8:0], to support external secondary bus masters. The secondary bus request and grant signals are connected internally to the arbiter and are not brought out to external pins when S_CFN# is HIGH.

PCI 6254 has a 2-level arbitration scheme in which arbitration is divided into two groups, a high priority group and a low priority group. The low priority group as a whole represents one entry in the high priority group; that is, if the high priority group consists of n masters, then in at least every n+1 transactions the highest priority is assigned to the low priority group. Priority changes evenly among the low priority group. Therefore, members of the high priority group can be serviced n transactions out of n+1, while one member of the low priority group is serviced once every n+1 transactions. Each master can be assigned to either the low priority group or the high priority group, through configuration register 42h.

Each group can be programmed to use a rotating priority or a fixed priority scheme, through configuration register 50h.

### 12.2.1.1 Rotating Priority Scheme

The secondary arbiter supports a programmable 2-level rotating algorithm that takes care of 10 requests/grants, including the PCI 6254. Figure 12–1 shows an example of an internal arbiter where four masters, including PCI 6254, are in the high priority group, and six masters are in the low priority group. Using this example, if all requests are always asserted, the highest priority rotates among the masters in the following fashion (high priority members are given in italics, low priority members, in boldface type):

B, m0, m1, m2, B, m0, m1, m3, B, m0, m1, m4, B, m0, m1, m5, and so on.

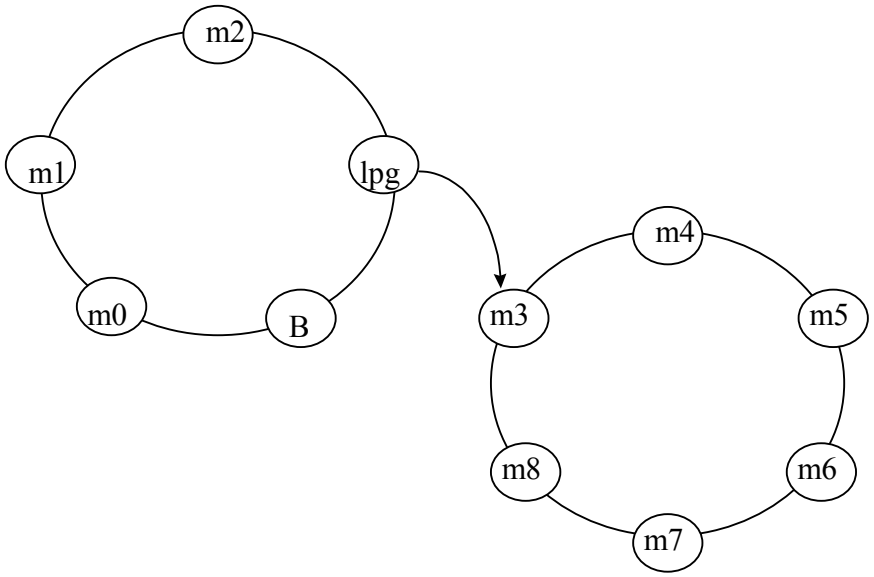



Figure 12-1 Secondary Arbiter Example

If all the masters are assigned to one group, the algorithm defaults to a rotating priority among all the masters. After reset, all external masters are assigned to the low priority group, and PCI 6254 is assigned to the high priority group. PCI 6254 receives highest priority on the target bus every other transaction, and priority rotates evenly among the other masters.

Priorities are reevaluated every time S_FRAME# is asserted, that is, at the start of each new transaction on the secondary PCI bus. From this point until the time that the next transaction starts, the arbiter asserts the grant signal corresponding to the highest priority request that is asserted. If a grant for a particular request is asserted, and a higher priority request subsequently asserts, the arbiter deasserts the asserted grant signal and asserts the grant corresponding to the new higher priority request on the next PCI clock cycle. When priorities are reevaluated, the highest priority is assigned to the next highest priority master relative to the master that initiated the previous transaction. The master that initiated the last transaction now has the lowest priority in its group. Priority is also re-evaluated if the requesting agent deasserts its request without generating any cycles while it was granted.

If PCI 6254 detects that an initiator has failed to assert S_FRAME# after 16 cycles of both grant assertion and a secondary idle bus condition, the arbiter will re-evaluate grant assignment.

### 12.2.1.2 Fixed Priority Scheme

PCI 6254 also has support for a fixed priority scheme within the two priority groups. In this case, configuration register 50h, bit 0 and 2 controls whether the low priority group or the high priority group will use fixed or rotating priority scheme. If a fixed priority scheme is used, then a master within the group is assigned to have the highest priority within its group, and then an option is set to control the priority of other masters relative to the highest priority master. This is controlled through bits 4-11 and bits 1 and 3 of the Internal Arbiter Control register (reg. 50h). For example, using the previous example with the groups at fixed priority, if master 7 has the highest priority of the low priority group, and PCI 6254 has the highest priority of the high priority group, and priority decreases in ascending order of masters for both groups, (bits 1,3 of register 50 are set to 1). The order of priority with the highest first, will be as follows:

B, m0, m1, m2, m7, m8, m3, m4, m5, m6

If bits 1, 3 or register 50 are set to 0, priority will decrease in the other direction, so the order becomes:

B, m2, m1, m0, m7, m6, m5, m4, m3, m8

To prevent bus contention, if the secondary PCI bus is idle, the arbiter never asserts one grant signal in the same PCI cycle in which it deasserts another. It deasserts one grant, and then asserts the next grant, no earlier than one PCI clock cycle later. If the secondary PCI bus is busy, that is, either S_FRAME# or S_IRDY# is asserted, the arbiter can deassert one grant and assert another grant during the same PCI clock cycle.

## 12.2.2 Secondary Bus Arbitration using an External Arbiter

The internal arbiter is disabled when the secondary bus central function control pin, S_CFN#, is tied HIGH. An external arbiter must then be used.

When S_CFN# is tied HIGH, PCI 6254 reconfigures two pins to be external request and grant pins. The S_GNT#[0] pin is reconfigured to be the PCI 6254's external request pin because it is output. The S_REQ#[0] pin is reconfigured to be the external grant pin because it is input. When an external arbiter is used, PCI 6254 uses the S_GNT#[0] pin to request the secondary bus. When the reconfigured S_REQ#[0] pin is asserted LOW after PCI 6254 has asserted S_GNT#[0], PCI 6254 initiates a transaction on the secondary bus one cycle later. If grant is asserted and PCI 6254 has not asserted the request, PCI 6254 parks the AD, CBE and PAR pins by driving them to valid logic levels.

The unused secondary bus grant outputs, S_GNT#<8:1> are driven HIGH. Unused secondary bus request inputs, S_REQ#<8:1> should be pulled HIGH.

## 12.2.3 Internal Arbitration Parking

If the internal secondary bus arbiter is enabled, the secondary arbiter can be optionally parked at the last active slot, or on any of the designated slots, and it can also be disabled.

PCI 6254 has the following options related to arbitration parking:

- No parking: all grants are deasserted if there are no requests asserted.
- Fixed parking: grant can be assigned to a specified master.
- Last master granted: grant is assign to the last granted master.

These options are selected through Internal Arbiter Control register at 50h, bits 12-15.

# 13 General Purpose I/O Interface

The PCI 6254 implements a 16 general-purpose I/O (GPIO) interface pins. During normal operation, the configuration registers control GPIO interface. In addition, the GPIO pins can be used for the following functions:

- During Secondary Reset, the GPIO interface can be used to shift in a 16-bit serial stream that serves as a secondary bus clock disable mask.
- A live insertion bit can be used, along with the GPIO[3] pin, to bring the PCI 6254 gracefully to a halt through hardware, permitting live insertion of option cards behind the PCI 6254.
- During non-transparent mode, some pins can be used as an interrupt for communication between the primary and secondary interfaces.

**Table 13-1: GPIO Pin Alternate Functions**

| GPIO Pin | Alternate Function |
|---|---|
| **GPIO0 – pull-up** | Functions as shift register clock output when PRST# asserted. |
| GPIO1 – pull-up | |
| GPIO2 – pull-up | Functions as shift/load control output to shift register when PRST# asserted |
| GPIO3 – pull-up | Can be used for live insertion function, if set as input and live insertion mode bit is set. If high, transaction forwarding is disabled. |
| GPIO4 – pull-up | If non-transparent mode is enabled, this input can be used as an active low level triggered external interrupt input to trigger P_INTA#. |
| GPIO5 – pull-up | If non-transparent mode is enabled, this input can be used as an active low level triggered external interrupt input to trigger S_INTA#. |
| GPIO6 – pull-up | |
| GPIO7 – pull-up | |
| GPIO8 | Power up PWRGD latched status |
| GPIO9 | Power up PWRGD latched status |
| GPIO10 | Power up PWRGD latched status |
| GPIO11 | Power up PWRGD latched status |
| GPIO12 | Power up PWRGD latched status |
| GPIO13 | Power up PWRGD latched status |
| GPIO14 | Power up PWRGD latched status |
| GPIO15 | Power up PWRGD latched status |

*** Important: Internal pull-low resistor is weak and when pull-low function is used, an external pull-low resistor is recommended.**

## 13.1 GPIO Control Registers

During normal operation, the GPIO interface is controlled by the following configuration registers:

- The GPIO output data register
- The GPIO output enable control register
- The GPIO input data register

GPIO7-0 consist of five 8-bit fields:

- Write-1-to-set output data field
- Write-1-to-clear output data field
- Write-1-to-set signal output enable control field
- Write-1-to-clear signal output enable control field
- Input data field

The output enable fields control whether each GPIO signal is input or output. Each signal is controlled independently by a bit in each output enable control field. If a 1 is written to the write-1-to-set field, the corresponding pin is activated as an output. If a 1 is written to the write-1-to-clear field, the output driver is three-stated, and the pin is then input only. Writing zeros to these registers has no effect. The reset state for these signals is input only. The input data field is read only and reflects the current value of the GPIO pins. A Type-0 configuration read operation to this address is used to obtain the values of these pins. All pins can be read at any time, whether configured as input only or as bi-directional. The output data fields also use the write-1-to-set and write-1-to-clear method. If a 1 is written to the write-1-to-set field and the pin is enabled as an output, the corresponding GPIO output is driven high. If a 1 is written to the write-1-to-clear field and the pin is enabled as an output, the corresponding GPIO output is driven low. Writing zeros to these registers has no effect. The value written to the output register will be driven only when the GPIO signal is configured as output. Type-0 configuration write operation is used to program these fields. The reset value for the output is 0.

GPIO15-8 consist of three 8-bit fields:

- Write- 0/1 as output data field
- Write- 0/1 to disable/enable output enable control field
- Input data field

# 14 Clocks

This chapter provides information about the clocks.

## *14.1 Primary and Secondary Clock Inputs*

PCI 6254 implements a separate clock input for each PCI interface. The primary interface is synchronized to the primary clock input, P_CLKIN, and the secondary interface is synchronized to the secondary clock input, S_CLKIN.

PCI 6254 operates at a maximum frequency of 66 MHz. Output clocks S_CLK[9:0] can be derived from P_CLKIN either at the same frequency, at P_CLKIN / 2, or at an asynchronous frequency.

PCI 6254 primary and secondary clock inputs can be asynchronous. The skew between P_CLKIN and S_CLKIN is irrelevant. PCI 6254 can tolerate a 1:2.5 or 2.5:1 frequency ratio between primary and secondary clock inputs.

## *14.2 Secondary Clock Outputs*

PCI 6254 has 10 secondary clock outputs that can be used as clock inputs for up to nine external secondary bus devices with one feedback to S_CLKIN.
The rules for using secondary clocks are:

- Each secondary clock output is limited to no more than 1 PCI load.
- All clock trace length, including feedback clock to the PCI 6254, must have equal length and impedance.
- Terminating or disabling unused secondary clock outputs is recommended to reduce power dissipation and noise in the system.

## *14.3 Disabling Unused Secondary Clock Outputs*

When secondary clock outputs are not used, both GPIO[3:0] and MSK_IN can be used to clock in a serial mask that selectively three-states secondary clock outputs. See GPIO section for details in this application.

After the serial mask has been shifted into the PCI 6254, the value of the mask is readable and can be changed in the secondary clock disable mask register. When the mask is modified by a configuration write operation to this register, the new clock mask disables the appropriate secondary clock outputs within a few cycles. This feature allows software to disable or enable secondary clock outputs based on the presence of option cards, and so on.
PCI 6254 delays deasserting the secondary reset signal, S_RSTOUT#, until the serial clock mask has been completely shifted in and the secondary clocks have been disabled or enabled, according to the mask. The delay between P_RSTIN# assertion and S_RSTOUT# deassertion is 16 cycles (32 cycles if S_CLK is operating at 66MHz).

### 14.3.1 Secondary Clock Control

The PCI 6254 uses the GPIO pins and the MSKIN signal to input a 16-bit serial data stream. This data stream is shifted into the secondary clock control register and is used for selectively disabling secondary clock outputs. The serial data stream is shifted in as soon as P_RSTIN# is detected deasserted and the secondary reset signal, S_RSTOUT#, is detected asserted. The deassertion of S_RSTOUT# is delayed until the PCI 6254 completes shifting in the clock mask data, which takes 16 clock cycles (32 cycles if operating at 66 MHz). After that, the GPIO pins can be used as general purpose I/O pins.

An external shift register should be used to load and shift the data. The GPIO pins are used for shift register control and serial data input. The data is input through the dedicated input signal, MSKIN. The shift register circuitry is not necessary for correct operation of the PCI 6254. The shift registers can be eliminated, and MSKIN can be tied low to enable all secondary clock outputs or tied high to force all secondary clock outputs high.

**GPIO Shift Register Operation**

| GPIO Pin | Operation |
|---|---|
| GPIO[0] | Shift register clock output at 33 MHz maximum frequency. |
| GPIO[2] | 0<br>1<br>Load<br>Shift<br><br>Shift Register control: |

**GPIO Serial Data Format**

| Bit Description | SCLK_O OUTPUT |
|---|---|
| **[1:0] Slot 0 PRSNT#<1:0> or device 0** | 0 |
| [3:2] Slot 1 PRSNT#<1:0> or device 1 | 1 |
| [5:4] Slot 2 PRSNT#<1:0> or device 2 | 2 |
| [7:6] Slot 3 PRSNT#<1:0> or device 3 | 3 |
| [8] Device 4 | 4 |
| [9] Device 5 | 5 |
| [10] Device 6 | 6 |
| [11] Device 7 | 7 |
| [12] Device 8 | 8 |
| [13] Can be used for PCI 6254 S_CLKIN input | 9 |
| [14] Reserved | Not applicable |
| [15] Reserved | Not applicable |

The first eight bits contain the PRSNT#[1:0] signal values for four slots, and these bits control the SCLKO[3:0] outputs. If one or both of the PRSNT#[1:0] signals are 0, that indicates that a card is present in the slot and therefore the secondary clock for that slot is not masked. If these clocks are connected to devices and not to slots, one or both of the bits should be tied low to enable the clock. The next five bits are the clock mask for devices; each bit enables or disables the clock for one device. These bits control the SCLKO[8:4] outputs: 0 enables the clock, and 1 disables the clock. Bit 13 is the clock enable bit for SCLKO[9], which is connected to the PCI 6254's SCLKIN input. If desired, the assignment of SCLKO clock outputs to slots, devices, and the PCI 6254's SCLKIN input can be rearranged from the assignment shown here. However, it is important that the serial data stream format match the assignment of SCLKO outputs. The GPIO pin serial protocol is designed to work with two 74F166 8-bit shift registers.

### 14.3.2 Force S_CLK[9:0] to LOW

There is also the S_CLKOFF input. When this pin is tied HIGH, all S_CLK[9:0] will be disabled and driven LOW.

## 14.4 Using an External Clock Source

PCI 6254 has 2 signals, OSC_SEL# and OSC_IN which can be used to connect an external clock source to the PCI 6254. During normal operation, PCI 6254 generates S_CLK[9:0] outputs based on the PCI CLK source. If OSCSEL# is low, PCI 6254 will derive S_CLK[9:0] outputs from the OSCIN signal instead. Clock division will still be performed on the OSCIN clock depending on the states of the P_M66EN and S_M66EN signals.

PCI 6254 also has S_CLK_STB input allowing designer to indicate the Secondary external clock source is stable. If this input is 0 indicating that the S_CLKIN is not yet stable, the PCI 6254 will not let S_RSTOUT# deassert.

## 14.5 Frequency Division Options

In Transparent mode, the PCI 6254 has built-in frequency division options to automatically adjust the output clocks S_CLK[9:0] for 66Mhz or 33Mhz operations. The following table depicts division conditions.

| P_M66EN | S_M66EN | DIVISION |
|---|---|---|
| 1 | 1 | 1/1 |
| 1 | 0 | 1/2 |
| 0 | 1 | 1/1 |
| 0 | 0 | 1/1 |

Notes
1. S_M66EN pin cannot be floating.
2. In Revision AA, the PCI 6154 does not drive S_M66EN to low when P_M66EN is low.
3. In Revision AB, the PCI 6154 drives S_M66EN to low when P_M66EN is low.

## 14.6 Running Secondary Port Faster than Primary Port

PCI 6254 allows running the Secondary port at a faster rate than the Primary port. PCI 6254 asynchronous design supports standard 66Mhz to 33MHz and faster secondary port speed such as 33Mhz to 66MHz conversion. Designers must provide the faster clock source either through an oscillator or a clock generator. If OSCIN is used to make use of the SCLKn outputs, the division control can be disabled by pulling the S_M66EN pin HIGH and not connecting this pin to the PCI slots. Otherwise external clock can be fed directly into the S_CLKIN.

## 14.7 Universal Mode Clock Behavior

There are some changes in clock behavior when PCI 6254 is in Universal mode.

| PIN | Descriptions |
|---|---|
| S_CLKIN | During Universal Non-Transparent Mode (U_MODE = 1 and TRANS# = 1). this input is ignored by PCI 6254 and the internal logic uses the input clock from the S_CLK0 pin. S_CLK0 acts as an input pin to provide clock for the Secondary interface. |
| S_CLK0 | **S_CLK0** output is three-stated during Universal Non-Transparent Mode. This allows S_CLK0 to take the CLK signal from a CPCI back plane for the Secondary port logic. When in CPCI SYSTEM slot, S_CLK0 drives the CPCI. When in PERIPHERAL slot, back plane clock drives S_CLK0 pin. |

# 15 Frequency Operation

PCI 6254 supports up to 66 MHz operations. CFG66 and PM66EN pin inputs should be HIGH for 66MHz operation.

## *15.1 66-MHZ operation*

Signal CFG66 must be tied high on the board to enable 66MHz operation and to set the 66MHz Capable bit in the Status register and the Secondary Status register in configuration space.

Signals P_M66EN and S_M66EN indicate whether the primary and secondary interfaces, respectively are operating at 66MHz. This information is needed to control the frequency of the secondary bus. Note that the PCI Local Bus Specification, Revision 2.2 restricts clock frequency changes above 33MHz to during PCI reset only.

The following primary and secondary bus frequency combinations are supported when using the Primary P_CLKIN to generate the secondary clock outputs:

- 66MHz primary bus, 66MHz secondary bus
- 66MHz primary bus, 33MHz secondary bus
- 33MHz primary bus, 33MHz secondary bus

If P_M66EN is low (66MHz capable, primary bus at 33MHz), then the PCI 6254 drives LOW S_M66EN to indicate that the secondary bus is operating at 33MHz. If Designers want to run the secondary bus at faster than primary bus, S_M66EN need not be connected to secondary PCI devices.

PCI 6254 can also generate S_CLK outputs from the OSCIN input if enabled. When PCI 6254 is running with external clock input that is not generated from S_CLK[9:0] outputs, the P_M66EN and S_M66EN controlled clock division will not apply.

When OSCIN or other external clock inputs are used for the secondary port, PCI 6254 can run with a maximum ratio of 1:2.5 or 2.5:1 between primary and secondary bus clocks.

# 16 Reset

This chapter describes the primary interface, secondary interface, and chip reset mechanisms. PCI 6254 has three reset inputs, PWRGD, P_RSTIN# and S_RSTIN#. There are also Power Management initiated internal reset and software controlled internal reset.

**Note:**
After any of the resets are deasserted, PCI 6254 requires 32 clocks to initialize the bridge functions.

Great care must be exercised when using P_RSTOUT# and S_RSTOUT# to feed to their corresponding RSTIN#. As P_RSTIN# can cause S_RSTOUT# and S_RSTIN# can cause P_RSTOUT#, there is a latching effect if both feedbacks are applied.

## *16.1 Power Good Reset*

For PCI 6254, a clean 3.3V PWRGD input signal must be provided. This input should NOT be simply connected to the 3.3V supply. When PWRGD is low, the following events occur:

- Registers 80h-ffh and extended registers that have default values are reset.
- All outputs are three-stated and all internal logic are reset. **(This is not true for PCI 6254 Rev AA)**
- (PCI 6254 Rev AA only: In Non-Transparent mode, PWRGD going HIGH causes EEPROM to be loaded.)

**Important: PCI 6254 requires a clean low to high transition PWRGD input. The PWRGD is not internally debounced. It must be debounced externally and a HIGH input must reflect that the power is indeed stable.** Its asserting and deasserting edges can be asynchronous to P_CLK and S_CLK.

### 16.1.1 PWRGD and Output Signals

Except for PCI 6254 Rev AA, as soon as PWRGD is 0, all outputs, including all clock and reset outputs, are three-stated.

## 16.2 Primary Reset Input

PCI 6254 requires at least 2 clocks before P_RSTIN# rising edge to reset properly.

When P_RSTIN# is asserted, the following events occur:

- In both Transparent and Non-Transparent modes, PCI 6254 three-states all primary PCI interface signals, except S_RSTOUT#, S_GNT#[8:0], SCLKO[9:0], and S_REQ64#, within 1 clock. In Non-Transparent Mode, S_REQ64# is also three-stated with 1 clock.
- In Non-Transparent Mode, Primary configuration registers 0-3Fh, **Secondary configuration registers 0-3Fh,** and shadow registers 40h-7fh are reset. In Transparent Mode, all registers, except sticky registers, are reset.
- In Non-Transparent Mode, S_RSTOUT# is driven active to indicate a Primary PCI reset if the Secondary Reset Output Mask bit is not set.
- In Non-Transparent Mode, Upon deassertion of P_RSTIN#, S_INTA# is driven active and status bit set if the Primary PCI Interrupt event is enabled.
- In Non-Transparent Mode, PCI 6254 performs a Primary Port state machine reset only if the Secondary Reset Output Mask bit at Register D9h bit 3 is not set. Secondary port state machines are not affected.
- In Transparent Mode, active P_RSTIN# will cause Secondary port reset. 43 clocks after P_RSTIN# going HIGH, S_RSTOUT# will go HIGH.
- Clock disable bits begin to be shifted in at the rising edge of P_RSTIN#.

The P_RSTIN# asserting and deasserting edges can be asynchronous to P_CLK and S_CLK. PCI 6254 requires 32 primary port PCI clocks after P_RSTIN# rising edge to reset its internal logic.

When P_RSTIN# is asserted, all posted write and delayed transaction data buffers are reset; therefore, any transactions residing in buffers at the active time of P_RSTIN# are discarded.

## 16.3 Primary Reset Output

Only in Non-Transparent Mode would PCI 6254 assert P_RSTOUT# when any of the following conditions is met:

- Secondary reset input (using S_RSTOUT# pin) is asserted in Universal Non-Transparent Mode (U_MODE = 1, TRANS# = 1).
- S_RSTIN# is asserted in standard Non-Transparent Mode with Primary port having boot priority (TRANS# = 1, P_BOOT = 1).
  Signal P_RSTOUT# remains asserted as long as S_RSTIN# is asserted.
- The Primary Reset bit in the Non-Transparent Diagnostic control register is set.
  Signal P_RSTOUT# remains asserted until the reset control bit is cleared.

## *16.4 Secondary Reset Input*

S_RSTIN# input pin is only used for Non-Transparent Operations and is not used in Transparent or Universal Mode. In Universal Non-Transparent Mode, S_RSTOUT# pin is used as the secondary port reset input pin. PCI 6254 requires 2 clocks before S_RSTIN# rising edge to reset properly.

When S_RSTIN# is asserted, the following events occur:

- PCI 6254 three-states all secondary PCI interface signals, except S_RSTOUT#, S_GNT#[8:0], SCLKO[9:0], and S_REQ64# within 1 clock.
- Non-Transparent Mode Secondary configuration registers 0-3Fh are **NOT** reset. There are no registers that are reset by an S_RSTIN# active input.
- P_RSTOUT# is driven active to indicate a Secondary PCI reset if the Primary Reset Output Mask bit is not set.
- Upon deassertion of S_RSTIN#, P_INTA# is driven active and status bit set if the Secondary PCI Interrupt event is enabled.
- PCI 6254 performs a Secondary Port state machine reset only. Primary port state machine are not affected.

The S_RSTIN# asserting and deasserting edges can be asynchronous to P_CLK and S_CLK. PCI 6254 requires 32 secondary port PCI clocks to reset its internal logic.

**Important: When not used, S_RSTIN# must be connected to high state.**

When S_RSTIN# is asserted, all secondary PCI interface signals, including the secondary grant outputs, are immediately three-stated. All posted write and delayed transaction data buffers are reset; therefore, any transactions residing in buffers at the active time of S_RSTIN# are discarded.

When S_RSTIN# is asserted by means of the secondary reset bit (bit 6 of register 3Eh in Transparent Mode and 42h in Non-Transparent Mode), PCI 6254 configuration space remains accessible from the primary interface.

In Transparent Mode and Universal Transparent Mode, secondary port reset is automatic whenever S_RSTOUT# is active. S_RSTIN# input is not used and it should be tied “HIGH”.

### 16.4.1 Universal Mode Secondary Reset Input

In Universal Non-Transparent Mode (U_MODE = 1, TRANS# = 1), S_RSTOUT# output is disabled and S_RSTOUT# pin is used as the equivalent of S_RSTIN# input pin. During this mode, a “LOW” input presented at S_RSTOUT# pin will cause a Secondary port reset. S_RSTIN# input is not used.

## *16.5 Secondary Reset Output*

PCI 6254 asserts S_RSTOUT# when any of the following conditions is met:

- Signal P_RSTIN# is asserted.
  Signal S_RSTOUT# remains asserted as long as P_RSTIN# is asserted and does not deassert until P_RSTIN# is deasserted, or while the secondary clock serial disable mask is being shifted in (16 clock cycles after P_RSTIN# deassertion) or the Secondary Clock input Stable S_CLKIN_STB input is “LOW”.
- The Secondary Reset bit in the bridge control register is set. Signal S_RSTOUT# remains asserted until the rest control bit is cleared.

## 16.6 Software Chip Reset

The chip reset bit (Transparent Mode register 41h and Non-Transparent Mode register D9h, bit 0) in the diagnostic control register can be used to reset the entire PCI 6254 like the PWRGD input except that it will NOT cause P_RSTOUT# or S_RSTOUT# to go active and it will NOT three-states the signals. However it will cause an EEPROM autoload if PCI 6254 is in Non-Transparent Mode.

When the chip reset bit is set, all registers and chip state are reset. Within 4 PCI clock cycles after completion of the configuration write operation that sets the chip reset bit, the chip reset bit automatically clears and the chip is ready for configuration.

During chip reset, PCI 6254 is inaccessible.

## 16.7 Power Management internal Reset

In Transparent Mode, or in Non-Transparent Mode with P_BOOT = 0, when there is a D3hot to D0 transition with Power Management Control registers bit 1:0 programmed to D0, an internal reset equivalent to P_RST input will be generated and all relevant registers will be reset. However P_RSTOUT# and S_RSTOUT# will NOT be active.

In Non-Transparent Mode and P_BOOT = 1, when there is a D3hot to D0 transition with Power Management Control registers bit 1:0 programmed to D0, the P_RSTOUT# will NOT be active.

## 16.8 Reset to First Cycle Access Latency

The PCI 6254 supports EEPROM initialization and needs time to determine if EEPROM data is valid before allowing the first cycle access to its registers, with the exception of Test Register 52h. During EEPROM loading, all access to PCI 6254 will be Retried.

Test Register 52h can be accessed within the first 1282 PCI clocks upon P_RSTIN# going HIGH in Transparent Mode and upon PWRGD going HIGH in Non-Transparent Mode. Subsequent EEPROM loading can be stopped if a configuration write can be completed within these 1282 clocks. EEPROM loading speed can also be accelerated by a factor of 32 times setting register 52h bit 1 to 1 with these 1282 clocks.

The following calculation is based on EEPROM clock speed being PCI clock speed divided by 1024.

In Transparent Mode, upon P_RSTIN# going HIGH, the PCI 6254 requires 56836 PCI clocks to determine if the EEPROM signature is valid. If the signature is not valid, the PCI 6254 will stop EEPROM load and access to PCI 6254 registers can begin. If the signature is valid, the PCI 6254 needs 9216 clocks to load each EEPROM register until the specified registers are completely loaded.

In Non-Transparent Mode, upon PWRGD going HIGH, the PCI 6254 requires PCI 56836 clocks to determine if the EEPROM signature is valid. If the signature is not valid, the PCI 6254 will stop EEPROM load and access to PCI 6254 registers can begin. If the signature is valid, the PCI 6254 needs 9216 clocks to load each EEPROM register until the specified registers are completely loaded.

## *16.9 Reset Inputs Table*

There are Transparent or Non-Transparent Mode related reset controls that can be used to generate Primary or Secondary reset outputs, as shown in the table below.

The following table depicts the effect of different PCI 6254 RESET INPUT sources.

| **Operating Mode**<br><br>**Reset Inputs** | Transparent Mode | Non-Transparent Mode | Universal Transparent Mode | Universal Non-Transparent Mode (**S_RSTOUT#** used as secondary reset input) |
|---|---|---|---|---|
| P_RSTIN# | **- Resets Primary and Secondary Ports<br>- Causes S_RSTOUT# active<br>- Causes EEPROM load** | **- Resets only Primary Port<br>- Causes S_RSTOUT# active** | **- Resets Primary and Secondary Ports<br>- Causes S_RSTOUT# active<br>- Causes EEPROM load** | **- Resets only Primary Port** |
| S_RSTIN# | **- Not used** | **- Resets only Secondary Port<br>- Causes P_RSTOUT# active** | **- Not used** | **- Not used** |
| S_RSTOUT# | **- Not used as input** | **- Not used as input** | **- Not used as input** | **- Used as reset input and resets only Secondary Port<br>- Causes P_RSTOUT# active** |
| S_CLK_STB not active | **- Resets only Secondary Port<br>- Causes S_RSTOUT# active** | **- Resets only Secondary Port<br>- Causes S_RSTOUT# active** | **- Resets only Secondary Port<br>- Causes S_RSTOUT# active** | **- No effect** |
| PWRGD not ready | **- Resets sticky registers** | **- Resets sticky registers<br>- Causes EEPROM load** | **- Resets sticky registers** | **- Resets sticky registers<br>- Causes EEPROM load** |
| Register 41h bit 0 Chip Reset | **- Resets internal state machines** | **- NA** | **- Resets internal state machines** | **- NA** |
| Register D9h bit 0 Chip Reset | **- NA** | **- Resets internal state machines<br>- Causes EEPROM load** | **- NA** | **- Resets internal state machines<br>- Causes EEPROM load** |
| Register 3Eh bit 6 Secondary Reset | **- Resets only Secondary Port<br>- Causes S_RSTOUT# active** | **- NA** | **- Resets only Secondary Port<br>- Causes S_RSTOUT# active** | **- NA** |
| Shadow Register 42h bit 6 Secondary Reset | **- NA** | **- Causes S_RSTOUT# active** | **- NA** | **- No effect** |
| Register D9h bit 5 Primary Port Software Reset | **- NA** | **- Causes P_RSTOUT# active** | **- NA** | **- Causes P_RSTOUT# active** |

## *16.10 Power Up and Reset Pin State Table*

There are PowerGood, P_RSTIN#, S_RSTIN# and Device Hiding support in the PCI 6254. The following depicts the pin state for each of such event. PCI 6254 Rev AA table is listed is Section 17.10.1.

**Legend:**
U = Undermined
I = Input Only
T = Three-stated
D1 = Drive 1 to output
D0 = Drive 0 to output
D01 = Can drive both 0 or 1 to output

| **Power Up / Reset**<br><br>**PCI 6254 Pins** | PWRGD = 0 With or without Clock | Trans Mode P_RSTIN # = 0 | Non-Trans Mode P_RSTIN# = 0 S_RSTIN# = 0 | Non-Univ Non-Trans Mode S_RSTIN# = 0 P_RSTIN# = 1 | Univ Non-Trans Mode S_RSTIN# = 0 P_RSTIN# = 1 | Device Hiding EJECTOR Switch Open P_RSTIN# = 1 S_RSTIN# = 1 |
|---|---|---|---|---|---|---|
| P_AD[63:0] | **T** | **T** | **T** | **T** | **T** | Follows the corresponding modes value to the left except for difference listed below |
| P_CBE[7:0] | **T** | **T** | **T** | **T** | **T** | |
| P_PAR | **T** | **T** | **T** | **T** | **T** | |
| P_PAR64 | **T** | **T** | **T** | **T** | **T** | |
| P_FRAME# | **T** | **T** | **T** | **T** | **T** | |
| P_IRDY# | **T** | **T** | **T** | **T** | **T** | |
| P_TRDY# | **T** | **T** | **T** | **T** | **T** | |
| P_DEVSEL# | **T** | **T** | **T** | **T** | **T** | |
| P_STOP# | **T** | **T** | **T** | **T** | **T** | |
| P_LOCK# | **T** | **T** | **T** | **T** | **T** | |
| P_IDSEL | **I** | **I** | **I** | **I** | **I** | |
| P_PERR# | **T** | **T** | **T** | **T** | **T** | |
| P_SERR# | **T** | **T** | **T** | **T** | **T** | |
| P_REQ# | **T** | **T** | **T** | **D1** | **D1** | |
| P_GNT# | **I** | **I** | **I** | **I** | **I** | |
| P_M66EN# | **I** | **I** | **I** | **I** | **I** | |
| P_REQ64# | **T** | **T** | **T** | **T** | **T** | |
| P_ACK64# | **T** | **T** | **T** | **T** | **T** | |
| P_INTA# | **T** | **T** | **T** | **T** | **T** | |
| | | | | | | |
| S_AD[63:0] | **T** | **T** | **T** | **T** | **T** | |
| S_CBE[7:0] | **T** | **T** | **T** | **T** | **T** | |
| S_PAR | **T** | **T** | **T** | **T** | **T** | |
| S_PAR64 | **T** | **T** | **T** | **T** | **T** | |
| S_FRAME# | **T** | **T** | **T** | **T** | **T** | |
| S_IRDY# | **T** | **T** | **T** | **T** | **T** | |
| S_TRDY# | **T** | **T** | **T** | **T** | **T** | |
| S_DEVSEL# | **T** | **T** | **T** | **T** | **T** | |
| S_STOP# | **T** | **T** | **T** | **T** | **T** | |
| S_LOCK# | **T** | **T** | **T** | **T** | **T** | |
| S_IDSEL | **I** | **I** | **I** | **I** | **I** | |
| S_PERR# | **T** | **T** | **T** | **T** | **T** | |
| S_SERR# | **T** | **T** | **T** | **T** | **T** | |
| S_REQ0# | **I** | **I** | **I** | **I** | **D1** | |
| S_GNT0# | **T** | **D1** | **D1** | **D1** | **I** | |
| S_REQ[8:1]# | **I** | **I** | **I** | **I** | **I** | |
| S_GNT[8:1]# | **T** | **D1** | **D1** | **D1** | **D1** | |
| S_M66EN# | **I** | **I** | **I** | **I** | **I** | |
| S_REQ64# | **T** | **D0** | **T** | **T** | **T** | |
| S_ACK64# | **T** | **T** | **T** | **T** | **T** | |
| S_INTA# | **T** | **T** | **T** | **T** | **T** | |

| **Power Up / Reset**<br><br>**PCI 6254 Pins** | PWRGD = 0 With or without Clock | Trans Mode P_RSTIN # = 0 | Non-Trans Mode P_RSTIN# = 0 S_RSTIN# = 0 | Non-Univ Non-Trans Mode S_RSTIN# = 0 P_RSTIN# = 1 | Univ Non-Trans Mode S_RSTIN# = 0 P_RSTIN# = 1 | Device Hiding EJECTOR Switch Open P_RSTIN# = 1 S_RSTIN# = 1 |
|---|---|---|---|---|---|---|
| P_CLKIN | **I** | **I** | **I** | **I** | **I** | |
| S_CLKIN | **I** | **I** | **I** | **T** | **I -not used** | |
| S_SCLK0<br>-S_CLKOFF=0<br>-S_CLKOFF=1 | **T**<br>**T** | **D01**<br>**D0** | **D01**<br>**D0** | **D01**<br>**D0** | **I**<br>**I** | |
| S_CLK[9:1]<br>-S_CLKOFF=0<br>-S_CLKOFF=1 | **T**<br>**T** | **D01**<br>**D0** | **D01**<br>**D0** | **D01**<br>**D0** | **D01**<br>**D0** | |
| S_CLKOFF | **I** | **I** | **I** | **I** | **I** | |
| S_CLKIN_STB | **I** | **I** | **I** | **I** | **I** | |
| MSK_IN | **I** | **I** | **I** | **I** | **I** | |
| OSCSEL# | **I** | **I** | **I** | **I** | **I** | |
| OSCIN | **I** | **I** | **I** | **I** | **I** | |
| | | | | | | |
| PWRGD | **I** | **I** | **I** | **I** | **I** | |
| P_RSTOUT# | **T** | **D1** | **D0** | **D0** | **D0** | **D1** |
| P_RSTIN# | **I** | **I** | **I** | **I** | **I** | |
| S_RSTOUT# | **T** | **D0** | **D0** | **D01** | **I** | **D1**<br>**I in Univ Non-Trans Mode** |
| S_RSTIN# | **I** | **I** | **I** | **I** | **I** | |
| | | | | | | |
| ENUM# | **T** | **T** | **T** | **T** | **T** | |
| L_STAT<br>-EJECT_EN#=0 | **T** | **D1** | **D1** | **D01** | **D01** | **D0** |
| EJECT | **I** | **I** | **I** | **I** | **I** | |
| EJECT_EN# | **I** | **I** | **I** | **I** | **I** | |
| | | | | | | |
| S_CFN# | **I** | **I** | **I** | **I** | **I** | |
| CFG66 | **I** | **I** | **I** | **I** | **I** | |
| BPCC_EN | **I** | **I** | **I** | **I** | **I** | |
| GPIO0 | **T** | **D1** | **D0** | **D01** | **D01** | **T**<br>**-if not used as output** |
| GPIO1 | **T** | **D0** | **D0** | **D01** | **D01** | **T**<br>**-if not used as output** |
| GPIO2 | **T** | **D0** | **D0** | **D01** | **D01** | **T**<br>**-if not used as output** |
| GPIO[15:3} | **T** | **T** | **T** | **T** | **T** | |
| P_PME# | **T** | **T** | **T** | **T** | **T** | |
| S_PME# | **T** | **T** | **T** | **T** | **T** | |
| EEPCLK | **T** | **D1** | **D1** | **D1** | **D1** | |
| EEPDATA | **T** | **D1** | **D1** | **D1** | **D1** | |
| EE_EN# | **I** | **I** | **I** | **I** | **I** | |
| 64EN# | **I** | **I** | **I** | **I** | **I** | |
| TRANS# | **I** | **I** | **I** | **I** | **I** | |
| U_MODE | **I** | **I** | **I** | **I** | **I** | |
| TEST# | **I** | **I** | **I** | **I** | **I** | |
| XB_MEM | **I** | **I** | **I** | **I** | **I** | |
| S_IDSEL | **I** | **I** | **I** | **I** | **I** | |
| P_BOOT | **I** | **I** | **I** | **I** | **I** | |

## 16.10.1 PCI 6254 Rev AA Power Up and Reset Pin State Table

In rev. AA, Power Good input does not affect pin state. P_RSTIN and valid clock are required to reach known initial pin state.

| **Power Up / Reset**<br><br>**PCI 6254 Pins** | Trans Mode P_RSTIN # = 0 | Non-Trans Mode P_RSTIN# = 0 S_RSTIN# = 0 | Non-Univ Non-Trans Mode S_RSTIN# = 0 P_RSTIN# = 1 | Univ Non-Trans Mode S_RSTIN# = 0 P_RSTIN# = 1 | Device Hiding EJECTOR Switch Open P_RSTIN# = 1 S_RSTIN# = 1 |
|---|---|---|---|---|---|
| P_AD[63:0] | **T** | **T** | **T** | **T** | Follows the corresponding modes value to the left except for difference listed below |
| P_CBE[7:0] | **T** | **T** | **T** | **T** | |
| P_PAR | **T** | **T** | **T** | **T** | |
| P_PAR64 | **T** | **T** | **T** | **T** | |
| P_FRAME# | **T** | **T** | **T** | **T** | |
| P_IRDY# | **T** | **T** | **T** | **T** | |
| P_TRDY# | **T** | **T** | **T** | **T** | |
| P_DEVSEL# | **T** | **T** | **T** | **T** | |
| P_STOP# | **T** | **T** | **T** | **T** | |
| P_LOCK# | **T** | **T** | **T** | **T** | |
| P_IDSEL | **I** | **I** | **I** | **I** | |
| P_PERR# | **T** | **T** | **T** | **T** | |
| P_SERR# | **T** | **T** | **T** | **T** | |
| P_REQ# | **T** | **T** | **D1** | **D1** | |
| P_GNT# | **I** | **I** | **I** | **I** | |
| P_M66EN# | **I** | **I** | **I** | **I** | |
| P_REQ64# | **T** | **T** | **T** | **T** | |
| P_ACK64# | **T** | **T** | **T** | **T** | |
| P_INTA# | **T** | **T** | **T** | **T** | |
| | | | | | |
| S_AD[63:0] | **T** | **T** | **T** | **T** | |
| S_CBE[7:0] | **T** | **T** | **T** | **T** | |
| S_PAR | **T** | **T** | **T** | **T** | |
| S_PAR64 | **T** | **T** | **T** | **T** | |
| S_FRAME# | **T** | **T** | **T** | **T** | |
| S_IRDY# | **T** | **T** | **T** | **T** | |
| S_TRDY# | **T** | **T** | **T** | **T** | |
| S_DEVSEL# | **T** | **T** | **T** | **T** | |
| S_STOP# | **T** | **T** | **T** | **T** | |
| S_LOCK# | **T** | **T** | **T** | **T** | |
| S_IDSEL | **I** | **I** | **I** | **I** | |
| S_PERR# | **T** | **T** | **T** | **T** | |
| S_SERR# | **T** | **T** | **T** | **T** | |
| S_REQ0# | **I** | **I** | **I** | **D1** | |
| S_GNT0# | **D1** | **D1** | **D1** | **I** | |
| S_REQ[8:1]# | **I** | **I** | **I** | **I** | |
| S_GNT[8:1]# | **D1** | **D1** | **D1** | **D1** | |
| S_M66EN# | **I** | **I** | **I** | **I** | |
| S_REQ64# | **D0** | **T** | **T** | **T** | |
| S_ACK64# | **T** | **T** | **T** | **T** | |
| S_INTA# | **T** | **T** | **T** | **T** | |

| **Power Up / Reset**<br><br>**PCI 6254 Pins** | Trans Mode P_RSTIN # = 0 | Non-Trans Mode P_RSTIN# = 0 S_RSTIN# = 0 | Non-Univ Non-Trans Mode S_RSTIN# = 0 P_RSTIN# = 1 | Univ Non-Trans Mode S_RSTIN# = 0 P_RSTIN# = 1 | Device Hiding EJECTOR Switch Open P_RSTIN# = 1 S_RSTIN# = 1 |
|---|---|---|---|---|---|
| P_CLKIN | **I** | **I** | **I** | **I** | |
| S_CLKIN | **I** | **I** | **T** | **I -not used** | |
| S_SCLK0<br>-S_CLKOFF=0<br>-S_CLKOFF=1 | <br>**D01**<br>**D0** | <br>**D01**<br>**D0** | <br>**D01**<br>**D0** | <br>**I**<br>**I** | |
| S_CLK[9:1]<br>-S_CLKOFF=0<br>-S_CLKOFF=1 | <br>**D01**<br>**D0** | <br>**D01**<br>**D0** | <br>**D01**<br>**D0** | <br>**D01**<br>**D0** | |
| S_CLKOFF | **I** | **I** | **I** | **I** | |
| S_CLKIN_STB | **I** | **I** | **I** | **I** | |
| MSK_IN | **I** | **I** | **I** | **I** | |
| OSCSEL# | **I** | **I** | **I** | **I** | |
| OSCIN | **I** | **I** | **I** | **I** | |
| | | | | | |
| PWRGD | **I** | **I** | **I** | **I** | |
| P_RSTOUT# | **D1** | **D0** | **D0** | **D0** | **D1** |
| P_RSTIN# | **I** | **I** | **I** | **I** | |
| S_RSTOUT# | **D0** | **D0** | **D1** | **I** | **D1**<br>**I in Univ Non-Trans Mode** |
| S_RSTIN# | **I** | **I** | **I** | **I** | |
| | | | | | |
| ENUM# | **T** | **T** | **T** | **T** | |
| L_STAT<br>-EJECT_EN#=0 | **D1** | **D1** | **D01** | **D01** | **D0** |
| EJECT | **I** | **I** | **I** | **I** | |
| EJECT_EN# | **I** | **I** | **I** | **I** | |
| | | | | | |
| S_CFN# | **I** | **I** | **I** | **I** | |
| CFG66 | **I** | **I** | **I** | **I** | |
| BPCC_EN | **I** | **I** | **I** | **I** | |
| GPIO0 | **D0** | **D0** | **D01** | **D01** | **T**<br>**-if not used as output** |
| GPIO1 | **D0** | **D0** | **D01** | **D01** | **T**<br>**-if not used as output** |
| GPIO2 | **D0** | **D0** | **D01** | **D01** | **T**<br>**-if not used as output** |
| GPIO[15:3} | **T** | **T** | **T** | **T** | |
| P_PME# | **T** | **T** | **T** | **T** | |
| S_PME# | **T** | **T** | **T** | **T** | |
| EEPCLK | **D1** | **D1** | **D1** | **D1** | |
| EEPDATA | **D1** | **D1** | **D1** | **D1** | |
| EE_EN# | **I** | **I** | **I** | **I** | |
| 64EN# | **I** | **I** | **I** | **I** | |
| TRANS# | **I** | **I** | **I** | **I** | |
| U_MODE | **I** | **I** | **I** | **I** | |
| TEST# | **I** | **I** | **I** | **I** | |
| XB_MEM | **I** | **I** | **I** | **I** | |
| S_IDSEL | **I** | **I** | **I** | **I** | |
| P_BOOT | **I** | **I** | **I** | **I** | |

# 17 Bridge Behavior

A PCI cycle is initiated by asserting the FRAME# signal. In a bridge, there are a number of possibilities. These are summarized in the table below. Table 17-1 shows PCI 6254's actions for different cycle types.

**Table 17-1: Bridge Actions for Various Cycle Types**

| Initiator | Target | Response |
|---|---|---|
| Master on primary | Target on Primary | PCI 6254 does not respond. It detects this situation by decoding the address as well as monitoring the P_DEVSEL# for other fast and medium devices on the primary port. |
| Master on primary | Target on secondary | PCI 6254 asserts P_DEVSEL#, terminate the cycle normally if able to posted, otherwise return with a retry. Then passes the cycle to the appropriate port. When cycle is complete on the target port, it will wait for the initiator to repeat the same cycle and end with normal termination. |
| Master on primary | Target not on primary nor secondary port | PCI 6254 does not respond and the cycle will terminate as master abort. |
| Master on secondary | Target on the same secondary port | PCI 6254 does not respond. |
| Master on secondary | Target on primary or the other secondary port | PCI 6254 asserts S_DEVSEL#, terminate the cycle normally if able to posted, otherwise return with a retry. Then passes the cycle to the appropriate port. When cycle is complete on the target port, it will wait for the initiator to repeat the same cycle and end with normal termination. |
| Master on secondary | Target not on primary nor the other secondary | PCI 6254 does not respond. |

A target then has up to three cycles to respond before subtractive decoding is initiated. If the target detects an address hit, it should assert its DEVSEL# signal in the cycle corresponding to the values of bits 9 and 10 in the Configuration Status Register.

Termination of a PCI cycle can occur in a number of ways. Normal termination begins by the initiator (master) deasserting FRAME# with IRDY# being asserted (or remaining asserted) on the same cycle. The cycle completes when TRDY# and IRDY# are both asserted simultaneously. The target should deassert TRDY# for one cycle following final assertion (sustained three-state signal).

## *17.1 Abnormal Termination (Initiated by Bridge Master)*

### 17.1.1 Master Abort

Master abort indicates that PCI 6254 acting as a master receives no response (i.e., no target asserts P_DEVSEL# or S_DEVSEL#) from a target. The bridge deasserts FRAME# and then deasserts IRDY#.

## *17.2 Parity and Error Reporting*

Parity must be checked for all addresses and write data. Parity is defined on the P_PAR/P_PAR64 and S_PAR/S_PAR64 signals. Parity should be even (i.e. an even number of '1's) across AD, CBE, and PAR. Parity information on PAR is valid the cycle after AD and CBE are valid.

### 17.2.1 Reporting Parity Errors

For all address phases, if a parity error is detected, the error is reported on the P_SERR# signal by asserting P_SERR# for one cycle and then three-stating two cycles after the bad address. P_SERR# can only be asserted if bit 6 and 8 in the Command Register are both set to 1. For write data phases, a parity error is reported by asserting the P_PERR# signal two cycles after the data phase and should remain asserted for one cycle when bit 8 in the Command register is set to a 1. The target reports any type of data parity errors during write cycles, while the master reports data parity errors during read cycles.

Detection of an address parity error will cause the PCI bridge target to not claim the bus (P_DEVSEL# remains inactive) and the cycle will then terminate with a Master Abort. When the bridge is acting as master, a data parity error during a read cycle results in the bridge master initiating a Master Abort.

## *17.3 Secondary IDSEL Mapping*

When PCI 6254 detects a Type-1 configuration transaction for a device connected to the secondary, it translates the Type-1 transaction to Type-0 transaction on the downstream interface. Type-1 configuration format uses a 5-bit field at P_AD[15:11] as a device number. This is translated to S_AD[31:16] by PCI 6254. table 18-2 explains how PCI 6254 generates the secondary IDSELs. No device is permitted to connect IDSEL to AD[16] (the source bridge is device number 0). PCI 6254 is the source bridge for its secondary bus.

**Table 17-2: Generation of S_IDSEL.**

| P_AD[15:11] | S_AD[31:16] | Device Number | S_AD bit |
|---|---|---|---|
| 0 0000b | 0000 0000 0000 0001b | 0 (Source Bridge) | 16 |
| 0 0001b | 0000 0000 0000 0010b | 1 | 17 |
| 0 0010b | 0000 0000 0000 0100b | 2 | 18 |
| … | … | … | … |
| 0 1011b | 0000 1000 0000 0000b | 11 | 27 |
| 0 1100b | 0001 0000 0000 0000b | 12 | 28 |
| … | … | … | … |
| 0 1111b | 1000 0000 0000 0000b | 15 | 31 |
| 1 xxxxb | 0000 0000 0000 0000b | Special Cycle | N/A |

## *17.4 32-bit to 64-bit Cycle Conversion*

When a 32-bit device generates a request to a 64-bit PCI target, PCI 6254 can optionally convert this cycle to a 64-bit cycle on the target bus. The conversion is only used on 32-bit prefetchable read memory cycles and posted memory write cycles with more than 2 data transfers. This function is set through **bit 11 and 15 of register 46h**.

If the force 64-bit option is set, all posted memory write and prefetchable memory read cycles are internally stored in the PCI 6254 as 64-bit cycles if the data transfer will be more than 2 DWORDs cycles. These cycles execute on the target following the standard 64-bit PCI protocol.

PCI 6254 asserts REQ64#, and if the target responds with ACK64# active, PCI 6254 will generate a 64-bit cycle. For memory write cycles, if the initial DWORD address is on an odd boundary, PCI 6254 will generate a 64-bit cycle with a value of Fh for the low DWORD of the initial write data transfer. If the target of a posted memory write is a 32-bit device, and it retries with data transfer on an odd DWORD boundary, the rest of the cycle will be completed later as a 32-bit cycle when PCI 6254 retries it. Otherwise, PCI 6254 will continue to retry the cycle as a 64-bit transaction.

# 18 Flow Through Optimization

PCI 6254 has several options that can be used to optimize PCI transfers.

During flow through posted write cycles, if there is only 1 data transfer pending in the internal post memory write queue, PCI 6254 can be programmed to wait for a specified number of clocks before disconnecting. PCI 6254 will deassert IRDY# on the target side and wait for up to 7 clocks for more data from the initiator. If during this period new write data is received from the initiator, PCI 6254 will reassert IRDY# and continue with the write cycle. If no new write data is received during this period, the PCI 6254 will terminate the cycle to the target with the last data from the queue and finish the cycle at a later time.

For flow through delayed read cycles, if the internal read queue is almost full, PCI 6254 can be programmed to wait for a specified number of clocks for read data from the target before disconnecting. During this time, the PCI 6254 will deassert IRDY# from the read source and wait for up to 7 clocks. If additional space becomes available in the read queue before the end of IRDY# inactive period, PCI 6254 will reassert IRDY# and proceed with the next read data phase. If no additional space becomes available in the read queue, the current data phase will become the last one and the cycle will terminate at the end of the data phase.

## *18.1 Cautions with Non-Optimized PCI Master Devices*

PCI 6254 is capable of very high performance prefetching. However, for some PCI masters that cannot prefetch a lot of data due to limited buffers size or other reasons, the default aggressive prefetching may affect the overall performance. In this case, we recommend tuning PCI 6254 default aggressive prefetching and require that the PCI 6254 prefetching registers be programmed as the following:

| Register in Configuration Space | Data |
|---|---|
| 0x48 | Same value as in register 0x0C *<br>* Most PC set this value to 08h. |
| 0x49 | Same value as in register 0x0C *<br>* Most PC set this value to 08h. |
| 0x4A | 0 |
| 0x4B | 0 |
| 0x4C | 0 |
| 0x4D | 0 |

EEPROM can also be used to program the configuration space upon reset. Please refer to PCI 6254 data book for detail description of each field.

## *18.2 Read Cycle Optimization*

The main function is to increase the probability of flow-through occurring during read access to prefetchable memory regions. In the case that flow-through does not occur, it would be inefficient for the PCI 6254 to prefetch too little or too much data. If PCI 6254 prefetches too little and flow-through does not occur, then the read cycles become divided into multiple cycles. If PCI 6254 prefetches too much data and the internal FIFOs fill, it has to wait for the initiator to retry the previous read cycle, and then flush unclaimed data before it can enqueue subsequent cycles. The prefetch count registers can be used to tune the PCI 6254 for various PCI masters.

The initial count is equivalent to the cache line size, and assumes that a master will want at least 1 cache line of data. The incremental count is only used when PCI 6254 does not detect flow-through for the current cycle being prefetched during the initial fetch count. PCI 6254 will continue prefetching in increments until the maximum count is reached, then disconnect the cycle.

For read prefetching, the PCI 6254 implements several registers that control the amount of data prefetched on the secondary and primary PCI bus. The following registers can be used to optimize PCI 6254 performance during read cycles:

- Primary Initial Prefetch Maximum Count
- Primary Incremental Prefetch Count
- Primary Maximum Prefetch Count
- Secondary Initial Prefetch Maximum Count
- Secondary Incremental Prefetch Count
- Secondary Maximum Prefetch Count

The PCI 6254 will prefetch either until flow-through or until prefetch must stop based on the following conditions:

(IPMC + IPC + IPC + … + IPC) < MPC  where IPC < ½ MPC

IPMC = Initial Prefetch Maximum Count
IPC = Incremental Prefetch Count
MPC = Maximum Prefetch Count

If the prefetch count has not reached MPC and flow through has been achieved, the PCI 6254 will keep on prefetching until the requesting master terminates the prefetch request. Otherwise, when MPC has been reached, the PCI 6254 will not prefetch any more.

The incremental prefetch can be disabled by setting IPC >= MPC.

### 18.2.1 Primary/Secondary Initial Prefetch Count

This count controls the amount of data initially prefetched by the PCI 6254 on the primary/secondary bus during reads to the prefetchable memory region. This assumes there is enough space in the internal FIFO. If flow through is achieved during this initial prefetch, PCI 6254 will continue prefetching beyond this count.

### 18.2.2 Primary/Secondary Incremental Prefetch Count

This register controls the amount of prefetching done after the initial prefetch. If flow-through was not achieved during the initial prefetch, PCI 6254 will try to prefetch more data, until the FIFO fills, or until the maximum prefetch count is reached.

### 18.2.3 Primary/Secondary Maximum Prefetch Count

This register limits the amount of prefetched data for a single entry available in the internal FIFO at any time. During subsequent read prefetch cycles, the PCI 6254 will disconnect the cycle when the count of data in the FIFO for the current cycle reaches this value, and flow-through has not been achieved.

## *18.3 Read Prefetch Boundaries*

For **memory read** and **memory read line** commands, PCI 6254 prefetches from the starting address up until the address with offset equals to the Initial Prefetch count. For example, if Initial Prefetch count equals 20H and the starting address is 10H, PCI 6254 will only prefetch 10H (20H-10H) count and afterward will start incremental prefetch until the Maximum Prefetch count is reached, or flow through is achieved. The exception is that if the Initial Prefetch Count (IPC) and starting address offset difference (IPC – Starting address offset) is less than 3 transfers apart, the PCI 6254 will not activate incremental prefetch.

For **memory read multiple** commands, if the starting address is not 0, PCI 6254 will first prefetch from the starting address up until the address with offset equals to the Initial Prefetch count. Afterwards the PCI 6254 will additionally prefetch one “Initial Prefetch count” count. For example, if Initial Prefetch count equals 20H and the starting address is 10H, PCI 6254 will first prefetch 10H (20H-10H) count and then continues to prefetch another 20H count. Afterwards, incremental prefetch is invoked until the Maximum Prefetch count is reached, or flow through is achieved.

# 19 Non-Transparent Mode

PCI 6254 is a dual mode (Transparent and Non-Transparent) Universal PCI-to-PCI bridge.

**Important: The PWRGD is not internally debounced. When in Non-Transparent mode, it must be debounced externally and a HIGH input must reflect that the power is indeed stable.** Its asserting and deasserting edges can be asynchronous to P_CLK and S_CLK.

## *19.1 Non-Transparent Mode Configuration Space Map*

PCI 6254 can be configured to act as either a transparent or non-transparent PCI-to-PCI bridge. The mode is selectable through the TRANS# pin.

**Superscript legend:**
1 = Writable when Read Only Register Write Enable bit is set
2 = EEPROM loadable for part or all of the bits
3 = Shared registers for Primary and Secondary ports

| 31-24 | 23-16 | 15-8 | 7-0 | Primary Offset | Secondary Offset |
|---|---|---|---|---|---|
| [1,2,3] Device ID | | [1,2,3] Vendor ID | | 00h | 40h |
| Primary Status | | Primary Command | | 04h | 44h |
| [1,2,3] Class Code | | | [3] Revision ID | 08h | 48h |
| [1,2,3] BIST | [1,2,3] Header Type | Primary Latency Timer | Primary Cache Line Size | 0Ch | 4Ch |
| Downstream I/O or Memory 0 BAR | | | | 10h | 50h |
| Downstream Memory 1 BAR | | | | 14h | 54h |
| Downstream Memory 2 BAR or Downstream Memory 1 BAR Upper 32 bits | | | | 18h | 58h |
| Reserved | | | | 1Ch – 2Bh | 5Ch-6Bh |
| [1,2,3] Subsystem ID | | [1,2,3] Subsystem Vendor ID | | 2Ch | 6Ch |
| Reserved | | | | 30h | 70h |
| Reserved | | | [3] Capability Ptr | 34h | 74h |
| Reserved | | | | 38h | 78h |
| [1] Primary Maximum Latency | [1] Primary Minimum Grant | Primary Interrupt Pin | Primary Interrupt Line | 3Ch | 7Ch |
| [1,2,3] Device ID | | [1,2,3] Vendor ID | | 40h | 00h |
| Secondary Status | | Secondary Command | | 44h | 04h |
| [1,2,3] Class Code | | | [3] Revision ID | 48h | 08h |
| [1,2,3] BIST | [1,2,3] Header Type | Secondary Latency Timer | Secondary Cache Line Size | 4Ch | 0Ch |
| Upstream I/O or Memory 0 BAR | | | | 50h | 10h |
| Upstream Memory 1 BAR | | | | 54h | 14h |
| Upstream Memory 2 BAR or Upstream Memory 1 BAR Upper 32 bits | | | | 58h | 18h |
| Reserved | | | | 5Ch – 6Bh | 1Ch-2Bh |

| | | | | | |
|---|---|---|---|---|---|
| [1,2,3] **Subsystem ID** | | [1,2,3] **Subsystem Vendor ID** | | 6Ch | 2Ch |
| Reserved | | | | 70h | 30h |
| Reserved | | | [3] **Capability Ptr** | 74h | 34h |
| Reserved | | | | 78h | 38h |
| [1] **Secondary Maximum Latency** | [1] **Secondary Minimum Grant** | **Secondary Interrupt Pin** | **Secondary Interrupt Line** | 7Ch | 3Ch |

### 19.1.1 Configuration 80h-FFh, Shadow and Extended Registers

**Important Note:**
Registers 80h-FFh, Shadow Registers and Extended Registers are shared registers and can be accessed by both Primary and Secondary port. There should be extreme care in accessing such registers by both Primary and Secondary port masters. The use of the built-in semaphore mechanisms is recommended.

#### 19.1.1.1 Configuration 80h-FFh Registers

Registers 80h-FFh, and the extended registers, are set to their default values upon Power-up of the PCI 6254. Subsequent PCI resets from P_RSTIN# and S_RSTIN# will not affect their values.

| | | | | | |
|---|---|---|---|---|---|
| **XB Downstream Configuration Address** | | | | 80h | 80h |
| **XB Downstream Configuration Dataport** | | | | 84h | 84h |
| **XB Upstream Configuration Address** | | | | 88h | 88h |
| **XB Upstream Configuration Dataport** | | | | 8Ch | 8Ch |
| Reserved | **XB Configuration Access Semphore Status** | **XB Upstream Configuration Own Semaphore** | **XB Downstream Configuration Own Semaphore** | 90h | 90h |
| **Reserved** | **SERR# event disable** | **Clock Control** | | 94h | 94h |
| **GPIO[3:0] Input Data** | **GPIO{3:0] Output Enable Control** | **GPIO{3:0] Output Data** | **SERR# status** | 98h | 98h |
| **GPIO[7-4] Input Register** | **GPIO[7-4] Output Enable** | **GPIO[7-4] Output Data** | **Hot Swap switch ROR Control** | 9Ch | 9Ch |
| **GPIO[15-8] Input Register** | **GPIO[15-8] Output Enable** | **GPIO[15-8] Output Data** | **Pwrup Status** | A0h | A0h |
| **Upstream Message 3** | **Upstream Message 2** | **Upstream Message 1** | **Upstream Message 0** | A4h | A4h |
| **Downstream Message 3** | **Downstream Message 2** | **Downstream Message 1** | **Downstream Message 0** | A8h | A8h |
| **MSI Control** | | **Next Item Ptr = 00** | [1,2,3] **MSI Cap. ID =5 (rev AA = 8)** | **ACh** | **ACh** |
| **MSI Address** | | | | **B0h** | **B0h** |
| **MSI Upper Address** | | | | **B4h** | **B4h** |
| Reserved | | **MSI Data** | | **B8h** | **B8h** |

| | | | | | |
|---|---|---|---|---|---|
| Reserved | | | | **BCh** | **BCh** |
| **Downstream Doorbell Request** | | **Downstream Doorbell Enable** | | C0h | C0h |
| **Upstream Doorbell Request** | | **Upstream Doorbell Enable** | | C4h | C4h |
| **Upstream Interrupt Enable** | **Downstream Interrupt Status** | **Downstream Doorbell Status** | | C8h | C8h |
| **Downstream Interrupt Enable** | **Upstream Interrupt Status** | **Upstream Doorbell Status** | | CCh | CCh |
| **Extended Register Index** | **NT Configuration Own Semaphore** | Reserved | Reserved | D0h | D0h |
| **Extended Registers Dataport** | | | | D4h | D4h |
| **Arbiter Control** | | **Diagnostic Control** | **Chip Control** | D8h | D8h |
| **[1,2] Power Management Capabilities** | | **Next Item Ptr = E4** | **Capability ID = 01** | DCh | DCh |
| **[1,2] Power Management Data** | **PMCSR Bridge Support** | **[1,2] Power Management CSR** | | E0h | E0h |
| Reserved | **HSCSR = 00** | **Next Item Ptr = E8** | **Capability ID = 06** | E4h | E4h |
| **VPD Register = 0000** | | **Next Item Ptr = 00** | **Capability ID = 03** | E8h | E8h |
| **VPD Data Register = 0000_0000** | | | | ECh | ECh |
| **Reserved** | | | | F0h-FCh | F0h-FCh |

### 19.1.1.2 Extended Register Map

| 31-24 | 23-16 | 15-8 | 7-0 | INDEX |
|---|---|---|---|---|
| **32 Bit Sticky Register 0** | | | | 0h |
| **32 Bit Sticky Register 1** | | | | 1h |
| **32 Bit Sticky Register 2** | | | | 2h |
| **32 Bit Sticky Register 3** | | | | 3h |
| **32 Bit Sticky Register 4** | | | | 4h |
| **32 Bit Sticky Register 5** | | | | 5h |
| **32 Bit Sticky Register 6** | | | | 6h |
| **32 Bit Sticky Register 7** | | | | 7h |
| **[2] Upstream BAR 0 Translation Address** | | | | 8h |
| **[2] Upstream BAR 1 Translation Address** | | | | 9h |
| **[2] Upstream BAR 2 or Upstream BAR1 Upper 32 bits Translation Address** | | | | Ah |
| **[2] Upstream Translation Enable** | **[2] Upstream BAR 2 Translation Mask** | **[2] Upstream BAR 1 Translation Mask** | **[2] Upstream BAR 0 Translation Mask** | Bh |
| **[2] Downstream BAR 0 Translation Address** | | | | Ch |
| **[2] Downstream BAR 1 Translation Address** | | | | Dh |
| **[2] Downstream BAR 2 or Downstream BAR1 Upper 32 bits Translation Address** | | | | Eh |
| **[2] Downstream Translation Enable** | **[2] Downstream BAR 2 Translation Mask** | **[2] Downstream BAR 1 Translation Mask** | **[2] Downstream BAR 0 Translation Mask** | Fh |

### 19.1.1.3 Primary Configuration Shadow Registers

The following registers are normally shadowed and can only be accessed when Chip Control register D8h bit 6 is set to "1". These registers restored to their default value upon P_RSTIN# and S_RSTIN# and are accessible by both Primary and Secondary ports.

| 31-24 | 23-16 | 15-8 | 7-0 | Offset |
|---|---|---|---|---|
| **Bridge Control** | | Reserved | | 40h |
| **[2] Misc Options** | | **[2] Timeout Control** | **[2] Primary Flow Through Control** | 44h |
| **[2] Secondary Incremental Prefetch Count** | **[2] Primary Incremental Prefetch Count** | **[2] Secondary Prefetch Line Count** | **[2] Primary Prefetch Line Count** | 48h |
| **Reserved** | **[2] Secondary Flow Through Control** | **[2] Secondary Maximum Prefetch Count** | **[2] Primary Maximum Prefetch Count** | 4Ch |
| Reserved | **Test register** | **[2] Internal Arbiter Control** | | 50h |
| **EEPROM Data** | | **EEPROM Address** | **EEPROM control** | 54h |
| Reserved | | | | 58h-74h |

## *19.2 Non-Transparent Mode Primary Configuration Registers Description*

Except for the declared shared registers in the configuration space map above, Primary and Secondary ports have independent Registers 0-3Fh.

Non-Transparent Configuration Space 80h-FFh are accessible by both Primary and Secondary masters. In order to avoid corruptions by the other master, PCI 6254 implements an NT-Configuration Semaphore. It is recommended that any Configuration Write to the common space should use the semaphore implemented in register D2h.

User must use the correct byte size when accessing the configuration registers in order to prevent unintended corruption of configuration registers.

### 19.2.1 PCI Standard Configuration Registers

**Vendor ID Register (Read Only) – Offset 0h**
Defaults to 3388(h).

**Device ID Register (Read Only) – Offset 2h**
Defaults to 0021(h) for Non-Transparent Mode, 20h for Transparent Mode.

(Note: R/W - Read/Write, R/O - Read Only, R/W1C - Read/ Write 1 to clear)

**Command Register (Read/Write) – Offset 4h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 0 | I/O Space Enable | R/W | Controls the bridge's response to I/O accesses on the primary interface.<br>**0=ignore I/O transaction**<br>1=enable response to I/O transaction<br>Reset to 0. |
| 1 | Memory Space Enable | R/W | Controls the bridge's response to memory accesses on the primary interface.<br>**0=ignore all memory transaction**<br>1=enable response to memory transaction<br>Reset to 0. |
| 2 | Bus Master Enable | R/W | Controls the bridge's ability to operate as a master on the primary interface.<br>**0=do not initiate transaction on the primary interface and disable response to memory or I/O transactions on secondary interface**<br>1=enable the bridge to operate as a master on the primary interface<br>Reset to 0. |
| 3 | Special Cycle Enable | R/O | No special cycle implementation (**set to '0'**). |
| 4 | Memory Write and Invalidate Enable | R/O | Memory write and invalidate not supported (**set to '0'**). |

| | | | |
|---|---|---|---|
| 5 | VGA Palette Snoop Enable | R/W | Controls the bridge's response to VGA compatible palette accesses.<br>**0=ignore VGA palette accesses on the primary interface**<br>1=enable response to VGA palette writes on the primary interface (I/O address AD[9:0]=3C6h, 3C8h and 3C9h)<br>Reset to 0. |
| 6 | Parity Error Enable | R/W | Controls the bridge's response to parity errors.<br>**0=ignore any parity errors**<br>1=normal parity checking performed<br>Reset to 0. |
| 7 | Wait Cycle Control | R/W | PCI 6254 performs address / data stepping (reset to '1'). |
| 8 | P_SERR# Enable | R/W | Controls the enable for the P_SERR# pin.<br>**0=disable the P_SERR# driver**<br>1=enable the P_SERR# driver<br>Reset to 0. |
| 9 | Fast Back to Back Enable | R/W | Controls the bridge's ability to generate fast back-to-back transactions to **different** devices on the primary interface.<br>**0=no fast back to back transaction**<br>1=reserved. PCI 6254 does not generate fast back to back transaction<br>Reset to 0. |
| 10-15 | Reserved | R/O | Reserved. Reset to 0. |

### Status Register(Read/Write) – Offset 6h

| Bit | Function | Type | Description |
|---|---|---|---|
| 0-3 | Reserved | R/O | Reserved (**set to '0's**). |
| 4 | ECP | R/O | Enhanced Capabilities port. Reads as 1 to indicate PCI 6254 supports an enhanced capabilities list. |
| 5 | 66MHz | R/O | Reflects the state of CFG66 input pin. 1 = PCI 6254 is 66Mhz Capable. |
| 6 | UDF | R/O | No User-Definable Features (**set to '0'**). |
| 7 | Fast Back to Back Capable | R/O | Fast back-to-back write capable on primary side (**set to '1'**). |
| 8 | Data Parity Error Detected | R/WC | It is set when the following conditions are met:<br>1. P_PERR# is asserted<br>2. Bit 6 of Command Register is set<br>Reset to 0. |
| 9-10 | DEVSEL timing | R/O | DEVSEL# timing. Reads as 01b to indicate PCI 6254 responds no slower than with medium timing |
| 11 | Signaled Target Abort | R/WC | Should be set (by a target device) whenever a Target Abort cycle occurs. Reset to 0. |
| 12 | Received Target Abort | R/WC | Set to '1' (by a master device) when transactions are terminated with Target Abort. Reset to 0. |
| 13 | Received Master Abort | R/WC | Set to '1' (by a master) when transactions are terminated with Master Abort. Reset to 0. |
| 14 | Signaled System Error | R/WC | Should be set whenever P_SERR# is asserted. Reset to 0. |
| 15 | Detected Parity Error | R/WC | Should be set whenever a parity error is detected regardless of the state of the bit 6 of command register. Reset to 0. |

### Revision ID Register (Read Only) - Offset 8h

Defaults to 04h.

### Class Code Register (Read Only) – Offset 9h

This register can be written to by enabling the ROR Write Enable bit at register 9Ch bit 7.

Defaults to 068000h for Non-Transparent Mode, 060400 for Transparent Mode.

### Cache Line Size Register (Read/Write) – Offset 0Ch

This register is used when terminating memory write and invalidate transactions and when prefetching.
Only cache line sizes (in units of 32-bits words), which are power of two are valid (only one bit can be set in this register). Reset to 0.

### Latency Timer Register (Read/Write) – Offset 0Dh

This register sets the value for Master Latency Timer, which starts counting when the master asserts FRAME#. Reset to 0.

### Header Type Register (Read Only) – Offset 0Eh

Defaults to 0 in Non-Transparent Mode.

### BIST Register (Read Only) – Offset 0Fh

This register can be written to by enabling the ROR Write Enable bit at register 9Ch bit 7.

Reset to 0.

### I/O or Memory BAR 0 - Offset 10h

| Bit | Function | Type | Description |
|---|---|---|---|
| 31:0 | I/O or Memory BAR 0 | R/W | This base address register follows the standard PCI base address register definition. Its Bar Type and Prefetchable Area control is located in the corresponding BAR Control register located in the Extended Register.<br><br>(For PCI 6254 Rev AA, when read, the I/O Type indication bit 0 will only show 1 when I/O command is enabled in the Command Register at 04h bit 0 and the Bar Type is programmed to I/O.) |

### Memory BAR 1 - Offset 14h

| Bit | Function | Type | Description |
|---|---|---|---|
| 31:0 | Memory BAR 1 | R/W | This base address register follows the standard PCI base address register definition. Its Bar Type and Prefetchable Area control is located in the corresponding BAR Control register located in the Extended Register. |

### Memory BAR 2 - Offset 18h

| Bit | Function | Type | Description |
|---|---|---|---|
| 31:0 | Memory BAR 2<br><br>Or Memory BAR1 Upper 32 bits | R/W | This base address register follows the standard PCI base address register definition. Its Bar Type and Prefetchable Area control is located in the corresponding BAR Control register located in the Extended Register. |

### 19.2.2 Subsystem Vendor ID and Subsystem ID

The values reflected here are normally Read Only. However their default value can be changed by firmware or software by first turning the ROR Write Enable bit at Register 9Ch bit 7. After any modifications to such registers, this Write Enable bit must be cleared to preserve their Read Only nature.

**Subsystem Vendor ID (Read Only) - Offset 2Ch**
Defaults to 3388h

**Subsystem ID (Read Only) - Offset 2Eh**
Defaults to 0028h

**ECP Pointer – Offset 34h(Read/Only)**

| Bit | Function | Type | Description |
|---|---|---|---|
| 7-0 | ECP Pointer | R/O | Enhanced capabilities port offset pointer. This register reads as DCh to indicate the offset of the power management registers. |

**Interrupt Line Register (R/W) – Offset 3Ch**
Reset to 0.

(For Rev AA, secondary port interrupt line register can only be written to using Word command, Not Byte command)

**Interrupt Pin Register (Read Only) – Offset 3Dh**
Reads as 01 to indicate that PCI 6254 uses interrupt pin.

**Minimum Grant Register (Read Only) – Offset 3Eh**
This register can be written to by enabling the ROR Write Enable bit at register 9Ch bit 7.
Reset to 0.

**Maximum Latency (Read Only) – Offset 3Fh**
This register can be written to by enabling the ROR Write Enable bit at register 9Ch bit 7.
Reset to 0.

### 19.2.3 Secondary Port Standard PCI Configuration Registers Shadow

Registers 40h-7Fh reflect the image of the opposite port Standard PCI configuration register 0h-3Fh. **Upon Transparent Mode Primary Reset deassertion or Non-Transparent Mode PowerGood assertion, and during EEPROM Autoload phase, the Shadow registers are presented instead of the Standard PCI configuration image of the opposite port.**

**However, the following Shadow Registers 42h-77h can be accessed by setting Chip Control register D8h bit 6 to "1". These are shared miscellaneous control registers that are also used in Transparent Mode.**

**Bridge Control Register (Read/Write) – Offset 42h (Reg 3Eh in Transparent Mode)**

| Bit | Function | Type | Description |
|---|---|---|---|
| 0 | Parity Error Response Enable | R/W | Controls the bridge's response to parity errors on the secondary interface.<br>0=ignore address and data parity errors on the secondary interface<br>1=enable parity error reporting and detection on the secondary interface<br>Reset to 0. |
| 1 | S_SERR# Enable | R/W | Controls the forwarding of S_SERR# to the primary interface.<br>0=disable the forwarding S_SERR# to primary<br>1=enable the forwarding of S_SERR# to primary<br>Reset to 0. |
| 2 | ISA Enable | R/W | Controls the bridge's response to ISA I/O addresses, which is limited to the first 64K.<br>0=forward all I/O addresses in the range defined by the I/O Base and I/O Limit registers<br>1=block forwarding of ISA I/O addresses in the range defined by the I/O Base and I/O Limit registers that are in the first 64K of I/O space that address the last 768 bytes in each 1Kbytes block. Secondary I/O transactions are forwarded upstream if the address falls within the last 768 bytes in each 1Kbytes block<br>Reset to 0. |
| 3 | VGA Enable | R/W | Controls the bridge's response to VGA compatible addresses.<br>0=do not forward VGA compatible memory and I/O addresses from primary to secondary<br>1=forward VGA compatible memory and I/O address from primary to secondary regardless of other settings<br>Reset to 0. |
| 4 | reserved | R/O | Reserved (set to 0). |
| 5 | Master Abort Mode | R/W | Controls the bridge behavior in responding to master aborts on secondary interface<br>0=do not report master aborts (return ffff_ffffh on reads and discards data on writes)<br>1=report master aborts by signaling target abort<br>Reset to 0.<br>Note: During lock cycles, PCI 6254 ignores this bit, and always completes the cycle as a target abort. |

| | | | |
|---|---|---|---|
| 6 | Secondary Reset | R/W | Forces the assertion of SRST# signal pin on the secondary interface.<br>0=do not force the assertion of S_RSTOUT# pin<br>1=force the assertion of S_RSTOUT# pin<br>Reset to 0. |
| 7 | Fast Back to Back Enable | R/W | Controls the bridge's ability to generate fast back-to-back transactions to different devices on the secondary interface.<br>0=no fast back to back transaction<br>1=reserved. PCI 6254 does not generate Fast Back to Back cycle.<br>Reset to 0. |
| 8 | Primary Master Timeout | R/W | Sets the maximum number of PCI clock for an initiator on the primary bus to repeat the delayed transaction request.<br>0=Timeout after $2^{15}$ PCI clocks<br>1=Timeout after $2^{10}$ PCI clocks<br>Reset to 0. |
| 9 | Secondary Master Timeout | R/W | Sets the maximum number of PCI clock for an initiator on the secondary bus to repeat the delayed transaction request.<br>0=Timeout after $2^{15}$ PCI clocks<br>1=Timeout after $2^{10}$ PCI clocks<br>Reset to 0. |
| 10 | Master Timeout Status | R/WC | Set to '1' when either primary master timeout or secondary master timeout.<br>Reset to 0. |
| 11 | Master Timeout P_SERR# enable | R/W | Enable P_SERR# assertion during master timeout.<br>0=P_SERR# not asserted on master timeout<br>1=P_SERR# asserted on either primary or secondary master timeout.<br>Reset to 0. |
| 12 | Master Timeout S_SERR# enable | R/W | Enable S_SERR# assertion during master timeout.<br>0=S_SERR# not asserted on master timeout<br>1=S_SERR# asserted on either primary or secondary master timeout.<br>Reset to 0. |
| 13 | P_SERR# Enable | R/W | Controls the forwarding of P_SERR# to the Secondary interface.<br>0=disable the forwarding P_SERR# to Secondary port<br>1=enable the forwarding of P_SERR# to Secondary port<br>Reset to 0. |
| 15-14 | reserved | R/O | Reserved (set to '0's). |

**Primary Flow Through Control Register - Offset 44h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 2-0 | Primary posted write completion wait count | R/W | Maximum number of clocks that PCI 6254 will wait for posted write data from initiator if delivering write data in flow through mode and internal post write queues are almost empty. If the count is exceeded without any additional data from the initiator, the cycle to target will be terminated to be completed later.<br>000 : PCI 6254 will terminate cycle if there is only 1 data entry left in the internal write queue.<br>001 : PCI 6254 will deassert IRDY#, and wait 1 clock for data before terminating cycle.<br>…<br>111 : PCI 6254 will wait 7 clocks for source data. |
| 3 | Reserved | R/O | Reserved. Returns 0 when read |
| 6-4 | Primary delayed read completion wait count | | Maximum number of clocks that PCI 6254 will wait for delayed read data from target if returning read data in flow through mode and internal delayed read queue is almost full. If the count is exceeded without any additional space in the queue, the cycle to target will be terminated, and completed when initiator retries the rest of the cycle.<br>000 : PCI 6254 will terminate cycle if only 1 data entry is left in the read queue.<br>001 : PCI 6254 will deassert TRDY#, and wait 1 clock for data before terminating cycle.<br>…<br>111 : PCI 6254 will wait 7 clocks for source data. |
| 7 | Reserved | R/O | Reserved. Returns 0 when read. |

**Timeout Control Register - Offset 45h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 2-0 | Maximum retry counter control | R/W | Controls maximum number of times that PCI 6254 will retry a cycle before signaling a timeout. This timeout applies to Read/Write retries and can be enabled to trigger SERR# on the primary or secondary port depending the SERR# events that are enabled.<br>Maximum number of retries to timeout =<br>0000 : $2^{24}$<br>0001 : $2^{18}$<br>0010 : $2^{12}$<br>0011 : $2^{6}$<br>0111 : $2^{0}$<br>Reset to 0. |
| 3 | Reserved | R/O | |
| 5:4 | Primary master timeout divider | R/W | Provides an additional option for the primary master timeout. Timeout counter can optionally be divided by 256, in addition to its original setting in the Bridge Control register.<br>Original setting is 32K by default and programmable to 1K.<br>11 : timeout counter = Primary master timeout / 256<br>10 : timeout counter = Primary master timeout / 16<br>01 : timeout counter = Primary master timeout / 8<br>00: counter = Primary master timeout / 1<br>defaults to 0 |
| 7:6 | Secondary master timeout divider | R/W | Provides an additional option for the secondary master timeout. Timeout counter can optionally be divided by 256, in addition to its original setting in the Bridge Control register.<br>Original setting is 32K by default and programmable to 1K.<br>11 : timeout counter = Primary master timeout / 256<br>10 : timeout counter = Primary master timeout / 16<br>01 : timeout counter = Primary master timeout / 8<br>00: counter = Primary master timeout / 1<br>defaults to 0 |

**Miscellaneous Options - Offset 46h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 0 | Write completion wait for PERR# | R/W | If 1, PCI 6254 will always wait for PERR# status of the target before completing a delayed write transaction to the initiator.<br>Defaults to 0 |
| 1 | Read completion wait for PAR | R/W | If 1, PCI 6254 will always wait for PAR status of the target before completing a delayed read transaction to the initiator.<br>Defaults to 0 |
| 2 | DTR out of order enable | R/W | If 1, PCI 6254 may return delayed read transactions in a different order than requested. Otherwise, delayed read transactions are returned in the same order as requested<br>Defaults to 0 |
| 3 | Generate parity enable | R/W | If 1, PCI 6254 as a master will generate the PAR and PAR64 to cycles going across the bridge, otherwise, PCI 6254 passes along the PAR/PAR64 of the cycle as stored in the internal buffers.<br>Defaults to 0 |
| 6-4 | Address step control | R/W | During configuration type 0 cycles, PCI 6254 will drive the address for the number of clocks specified in this register before asserting FRAME#.<br>000 : PCI 6254 will assert FRAME# at the same time as the address.<br>001 : PCI 6254 will assert FRAME# 1 clock after it drives the address on the bus.<br>…<br>111 : PCI 6254 will assert FRAME# 7 clocks after it drives the address on the bus. |
| 8-7 | Reserved | | |
| 9 | Prefetch early termination | R/W | If 1, PCI 6254 will terminate prefetching at the current calculated count if flowthrough is not yet achieved, and another prefetchable read cycle is accepted by the PCI 6254.<br>If 0, PCI 6254 will always finish prefetching as programmed at the prefetch count registers, regardless of any other outstanding prefetchable reads in the transaction queue. |
| 10 | Read minimum enable | R/W | If 1, PCI 6254 will only initiate read cycles if there is available space in the fifo as specified by the prefetch count registers. |

| | | | |
|---|---|---|---|
| 15,11 | Force 64 Bit Control | R/W | If set, 32-bit prefetchable reads or 32-bit posted memory write cycles on one side will be converted to 64-bit cycles on completion to target side if target supports 64-bit transfers. If set to 0, cycles are not converted.<br><br>When combined with the control of bit 15 of this register, the following control is provided: (Bit 15 is set to 0 for rev AA and cannot be changed)<br><br>Bit 15, 11<br><br>0, 0 Disable (Default)<br><br>0, 1 Convert to 64 bit command onto both ports<br><br>1, 0 Convert to 64 bit command onto Secondary Port<br><br>1, 1 Convert to 64 bit command onto Primary Port<br><br>Starting address for all cycles using this feature should be on the qword boundary.<br><br>Defaults to 0 |
| 12 | Memory write and invalidate control | R/W | If 1, PCI 6254 will pass memory write and invalidate commands if there is at least 1 cache line of FIFO space available, otherwise it will complete as a memory write cycle.<br><br>If 0, PCI 6254 will retry memory write and invalidate commands if there is no space for 1 cacheline of data in the internal queues.<br><br>Defaults to 0 |
| 13 | Primary Lock Enable | R/W | If 1, PCI 6254 will follow the LOCK protocol on the primary interface. Otherwise, LOCK is ignored.<br><br>Defaults to 1 |
| 14 | Secondary Lock Enable | R/W | If 1, PCI 6254 will follow the LOCK protocol on the secondary interface. Otherwise, LOCK is ignored.<br><br>Defaults to 0 |
| 15,11 | Force 64 bit Control | R/W | See description in Bit 11 section. |

### 19.2.3.1 Prefetch Control Registers

Registers 44h, 48h – 4Dh are the prefetch control registers, and are used to fine-tune memory read prefetch behavior of the PCI 6254. Detailed descriptions of these registers can be found in Chapter 18 Flow Through Optimization.

**Primary Initial Prefetch Count - Offset 48h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 5-0 | Primary initial prefetch count | R/W | Controls initial prefetch count on the Primary bus during reads to prefetchable memory space. This register value should be a power of 2 (only one bit should be set to 1 at any time). Value is number of double words. Bit 0 is read only and is always 0.<br>Defaults to 10h. |
| 7-6 | Reserved | R/O | Reserved. Returns 0 when read. |

**Secondary Initial Prefetch Count - Offset 49h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 5-0 | Secondary initial prefetch count | R/W | Controls initial prefetch count on the Secondary bus during reads to prefetchable memory space. This register value should be a power of 2 (only one bit should be set to 1 at any time). Value is number of double words. Bit 0 is read only and is always 0.<br>Defaults to 10h. |
| 7-6 | Reserved | R/O | Reserved. Returns 0 when read. |

**Primary Incremental Prefetch Count - Offset 4Ah**

| Bit | Function | Type | Description |
|---|---|---|---|
| 5-0 | Primary incremental prefetch count | R/W | This controls incremental read prefetch count. When an entry's remaining prefetch Dword count falls below this value, the bridge will prefetch an additional "Primary incremental prefetch count" Dwords. This register value should be a power of 2 (only one bit should be set to 1 at any time). Value is number of double words. Bit 0 is read only and is always 0.<br>This register value must not exceed half the value programmed in the Primary Maximum Prefetch Count register. Otherwise, no incremental prefetch will be performed.<br>Defaults to 10h. |
| 7-6 | Reserved | R/O | Reserved. Returns 0 when read. |

**Secondary Incremental Prefetch Count - Offset 4Bh**

| Bit | Function | Type | Description |
|---|---|---|---|
| 5-0 | Secondary incremental prefetch count | R/W | This controls incremental read prefetch count. When an entry's remaining prefetch Dword count falls below this value, the bridge will prefetch an additional "Secondary incremental prefetch count" Dwords. This register value should be a power of 2 (only one bit should be set to 1 at any time). Value is number of double words. Bit 0 is read only and is always 0.<br><br>This register value must not exceed half the value programmed in the Secondary Maximum Prefetch Count register. Otherwise, no incremental prefetch will be performed.<br><br>Defaults to 10h. |
| 7-6 | Reserved | R/O | Reserved. Returns 0 when read. |

**Primary Maximum Prefetch Count - Offset 4Ch**

| Bit | Function | Type | Description |
|---|---|---|---|
| 5-0 | Primary maximum prefetch count | R/W | This value limits the cumulative maximum count of prefetchable Dwords that are allocated to one entry on the primary when flow through for that entry was not achieved. This register value should be an even number. Bit 0 is read only and is always 0.<br><br>Exception: 0h = 256 bytes = maximum programmable count<br><br>Defaults to 20h. |
| 7-6 | Reserved | R/O | Reserved. Returns 0 when read. |

**Secondary Maximum Prefetch Count - Offset 4Dh**

| Bit | Function | Type | Description |
|---|---|---|---|
| 5-0 | Secondary maximum prefetch count | R/W | Register limits the cumulative maximum count of prefetchable Dwords that are allocated to one entry on the secondary when flow through for that entry was not achieved. This register value should be an even number. Bit 0 is read only and is always 0.<br><br>Exception: 0h = 256 bytes = maximum programmable count<br><br>Defaults to 20h. |
| 7-6 | Reserved | R/O | Reserved. Returns 0 when read. |

**Secondary Flow Through Control Register - Offset 4Eh**

| Bit | Function | Type | Description |
|---|---|---|---|
| 2-0 | Secondary posted write completion wait count | R/W | Maximum number of clocks that PCI 6254 will wait for posted write data from initiator if delivering write data in flow through mode and internal post write queues are almost empty. If the count is exceeded without any additional data from the initiator, the cycle to target will be terminated, to be completed later.<br>000 : PCI 6254 will terminate cycle if there is only 1 data entry left in the internal write queue.<br>001 : PCI 6254 will deassert IRDY#, and wait 1 clock for data before terminating cycle.<br>…<br>111 : PCI 6254 will wait 7 clocks for source data. |
| 3 | Reserved | R/O | Reserved. Returns 0 when read |
| 6-4 | Secondary delayed read completion wait count | | Maximum number of clocks that PCI 6254 will wait for delayed read data from target if returning read data in flow through mode and internal delayed read queue is almost full. If the count is exceeded without any additional space in the queue, the cycle to target will be terminated, and completed when initiator retries the rest of the cycle.<br>000 : PCI 6254 will terminate cycle if only 1 data entry is left in the read queue.<br>001 : PCI 6254 will deassert TRDY#, and wait 1 clock for data before terminating cycle.<br>…<br>111 : PCI 6254 will wait 7 clocks for source data. |
| 7 | Reserved | R/O | Reserved. Returns 0 when read. |

**Internal Arbiter Control Register - Offset 50h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 0 | Low priority group fixed arbitration | R/W | If 1, the low priority group uses the fixed priority arbitration scheme, otherwise a rotating priority arbitration scheme is used<br>Defaults to 0 |
| 1 | Low priority group arbitration order | R/W | This bit is only valid when the low priority arbitration group is set to a fixed arbitration scheme. If 1, priority decreases in ascending numbers of the master, for example master #4 will have higher priority than master #3. If 0, the reverse is true. This order is relative to the master with the highest priority for this group, as specified in bits 7-4 of this register.<br>Defaults to 0 |
| 2 | High priority group fixed arbitration | R/W | If 1, the high priority group uses the fixed priority arbitration scheme, otherwise a rotating priority arbitration scheme is used<br>Defaults to 0 |
| 3 | High priority group arbitration order | R/W | This bit is only valid when the high priority arbitration group is set to a fixed arbitration scheme. If 1, priority decreases in ascending numbers of the master, for example master #4 will have higher priority than master #3. If 0, the reverse is true. This order is relative to the master with the highest priority for this group, as specified in bits 11-8 of this register.<br>Defaults to 0 |
| 7-4 | Highest priority master in low priority group | R/W | Controls which master in the low priority group has the highest priority. It is valid only if the group uses the fixed arbitration scheme.<br>0000 : master#0 has highest priority<br>0001 :<br>…<br>1001 : PCI 6254 has highest priority<br>1010-1111 : Reserved<br>Defaults to 0 |
| 11-8 | Highest priority master in high priority group | R/W | Controls which master in the high priority group has the highest priority. It is valid only if the group uses the fixed arbitration scheme.<br>0000 : master#0 has highest priority<br>0001 :<br>…<br>1001 : PCI 6254 has highest priority<br>1010-1111 : Reserved<br>Defaults to 0 |

| | | | |
|---|---|---|---|
| 12-15 | Bus Parking Control | R/W | Controls bus grant behavior during idle.<br>0000 : Last master granted is parked<br>0001 : Master #0 is parked<br>…<br>1001 : Master #8 is parked<br>1010 : PCI 6254 is parked<br>other : grant is deasserted<br>Defaults to 0 |

**PCI 6254 Test Register – Offset 52h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 0 | EEPROM Autoload control | R/W | If 1, disables EEPROM autoload.<br>**This is a testing feature only. In order to stop EEPROM load in Transparent Mode, 1 must be written into this register within 1200 clocks after P_RSTIN# goes HIGH. In Non-Transparent Mode, 1 must be written into this register within 1200 clocks after PWRGD goes HIGH.** |
| 1 | Fast EEPROM Autoload | R/W | If 1, speeds up EEPROM autoload by 32 times.<br>**This is a testing feature only. In order to enable Fast EEPROM load in Transparent Mode, 1 must be written into this register within 1200 clocks after P_RSTIN# goes HIGH. In Non-Transparent Mode, 1 must be written into this register within 1200 clocks after PWRGD goes HIGH.** |
| 2 | EEPROM autoload status | R/O | Status of EEPROM autoload. |
| 3 | Reserved | R/O | Reserved |
| 4 | 64EN# | R/O | Reflects the 64EN# pin status |
| 5 | S_CFN# | R/O | Reflects the S_CFN# pin status |
| 6 | TRANS# | R/O | Reflects the TRANS# pin status |
| 7 | U_MODE | R/O | Reflects the U_MODE pin status |

**EEPROM Control - Offset 54h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 0 | Start | R/W | Starts the EEPROM read or write cycle. |
| 1 | EEPROM command | R/W | Controls the command sent to the EEPROM<br>1 : write<br>0 : read |
| 2 | EEPROM Error | R/O | This bit is set to 1 if EEPROM ACK was not received during EEPROM cycle. |
| 3 | EEPROM autoload successful | R/O | This bit is set to 1 if EEPROM autoload occurred successfully after reset, and some configuration registers were loaded with values programmed in the EEPROM. If zero, EEPROM autoload was unsuccessful or was disabled. |
| 5-4 | Reserved | R/O | Reserved. Returns '0' when read. |
| 7-6 | EEPROM clock rate | R/W | Controls frequency of EEPROM clock. EEPROM clock is derived from the primary PCI clock.<br>00 = PCLK/1024 (Used for 66Mhz PCI)<br>01 = PCLK/512<br>10 = PCLK/256<br>11 = PCLK/32 (for test mode use)<br>defaults to 00 |

**EEPROM Address - Offset 55h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 0 | Reserved | R/O | Starts the EEPROM read or write cycle. |
| 7-1 | EEPROM address | R/W | Word address for EEPROM cycle. |

**EEPROM Control - Offset 56h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 15-0 | EEPROM Data | R/W | Contains data to be written to the EEPROM. During reads, this register contains data received from the EEPROM after a read cycle has completed. |

### 19.2.4 Cross Bridge Configuration Access Control Registers

Registers 80h-87h, 90h cannot be written from Downstream side while 88h-8Fh, 91h cannot be written from Upstream side. Configuration address should always be first setup before access should be made to the Configuration Dataport.

**XB Downstream Configuration Address - Offset 80h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 31:0 | Downstream Configuration Address | R/W | The data in this register is used as the Downstream configuration address.<br>Default = 0h |

**XB Downstream Configuration Dataport - Offset 84h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 31:0 | Downstream Configuration Dataport | R/W | The data presented at this register is used as the Downstream configuration Read/Write data.<br>Default = 0h |

**XB Upstream Configuration Address - Offset 88h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 31:0 | Upstream Configuration Address | R/W | The data in this register is used as the Upstream configuration address.<br>Default = 0h |

**XB Upstream Configuration Dataport - Offset 8Ch**

| Bit | Function | Type | Description |
|---|---|---|---|
| 31:0 | Upstream Configuration Dataport | R/W | The data presented at this register is used as the Upstream configuration Read/Write data.<br>Default = 0h |

**XB Downstream Configuration Ownership Semaphore Register – Offset 90h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 0 | Downstream Configurations Own | R/W1C | Only Upstream master can access this register using BYTE width access. When read as “0’ by Upstream interface with intention to access Downstream configuration registers, it indicates that Downstream configuration address and data registers are not owned and can be accessed. The read operation will automatically set this bit to “1” to indicated it will be owned. The owner should issue a configuration write “1” to clear this register after use. Software can check this semaphore status via register 92h bit 0 without taking ownership. |
| 7:1 | Reserved | RO | |

### XB Upstream Configuration Ownership Semaphore Register – Offset 91h

| Bit | Function | Type | Description |
|---|---|---|---|
| 0 | Upstream Configurations Own | R/W1C | Only Downstream master can access this register using BYTE width access. When read as "0' by Downstream interface with intention to access Upstream configuration registers, it indicates that Upstream configuration address and data registers are not owned and can be accessed. The read operation will automatically set this bit to "1" to indicated it will be owned. The owner should issue a configuration write "1" to clear this register after use. Software can check this semaphore status via register 92h bit 1 without taking ownership. |
| 7:1 | Reserved | RO | |

### XB Configuration Ownership Status Register (Read Only) – Offset 92h

| Bit | Function | Type | Description |
|---|---|---|---|
| 0 | Downstream Configurations Own bit | RO | Allows the examination of Downstream configuration OWN bit without setting it. |
| 1 | Upstream Configurations Own bit | RO | Allows the examination of Upstream configuration OWN bit without setting it. |
| 7:2 | Reserved | RO | |

### Clock Control Register (Read/Write) – Offset 94h

| Bit | Function | Type | Description |
|---|---|---|---|
| 1:0 | Clock 0 Disable | R/W | If either bit is 0, S_CLKOUT[0] is enabled.<br>When both bits are 1, S_CLKOUT[0] is disabled.<br>Upon secondary bus reset, this bit is initialized by shifting in a serial data stream. These bits are assigned to correspond to the PRSNT# pins for slot 0. |
| 3:2 | Clock 1 Disable | R/W | If either bit is 0, S_CLKO[1] is enabled.<br>When both bits are 1, S_CLKO[1] is disabled.<br>Upon secondary bus reset, this bit is initialized by shifting in a serial data stream. These bits are assigned to correspond to the PRSNT# pins for slot 1. |
| 5:4 | Clock 2 Disable | R/W | If either bit is 0, S_CLKO[2] is enabled.<br>When both bits are 1, S_CLKO[2] is disabled.<br>Upon secondary bus reset, this bit is initialized by shifting in a serial data stream. These bits are assigned to correspond to the PRSNT# pins for slot 2. |

| | | | |
|---|---|---|---|
| 7:6 | Clock 3 Disable | R/W | If either bit is 0, S_CLKO[3] is enabled.<br>When both bits are 1, S_CLKO[3] is disabled.<br>Upon secondary bus reset, this bit is initialized by shifting in a serial data stream. These bits are assigned to correspond to the PRSNT# pins for slot 3. |
| 8 | Clock 4 Disable | R/W | If 0, S_CLKO[4] is enabled.<br>When 1, S_CLKO[4] is disabled.<br>Upon secondary bus reset, this bit is initialized by shifting in a serial data stream. |
| 9 | Clock 5 Disable | R/W | If 0, S_CLKO[5] is enabled.<br>When 1, S_CLKO[5] is disabled.<br>Upon secondary bus reset, this bit is initialized by shifting in a serial data stream. |
| 10 | Clock 6 Disable | R/W | If 0, S_CLKO[6] is enabled.<br>When 1, S_CLKO[6] is disabled.<br>Upon secondary bus reset, this bit is initialized by shifting in a serial data stream. |
| 11 | Clock 7 Disable | R/W | If 0, S_CLKO[7] is enabled.<br>When 1, S_CLKO[7] is disabled.<br>Upon secondary bus reset, this bit is initialized by shifting in a serial data stream. |
| 12 | Clock 8 Disable | R/W | If 0, S_CLKO[8] is enabled.<br>When 1, S_CLKO[8] is disabled.<br>Upon secondary bus reset, this bit is initialized by shifting in a serial data stream. |
| 13 | Clock 9 Disable | R/W | If 0, S_CLKO[9] is enabled.<br>When 1, S_CLKO[9] is disabled.<br>Upon secondary bus reset, this bit is initialized by shifting in a serial data stream. |
| 15-14 | Reserved | R/O | Reserved |

### SERR# Event Disable Register - Offset 96h

If P_BOOT = 1, S_SERR# is driven when the following events occur. If P_BOOT = 0, P_SERR# is driven.

| Bit | Function | Type | Description |
|---|---|---|---|
| 0 | Address Parity error | R/W1C | Signal SERR# was asserted due to address parity error on either side of the bridge. Reset to 0. |
| 1 | Posted write parity error | R/W | Controls ability of PCI 6254 to assert P_SERR# when a data parity error is detected on the target bus during a posted write transaction. SERR# is asserted if this event occurs when this bit is 0 and SERR# enable bit in the command register is set. Reset value is 0. |
| 2 | Posted Memory write nondelivery | R/W | Controls ability of PCI 6254 to assert SERR# when it is unable to deliver posted write data after $2^{24}$ (or programmed Maximum Retry count at Timeout Control Register) attempts. SERR# is asserted if this event occurs when this bit is 0 and SERR# enable bit in the command register is set. Reset value is 0. |
| 3 | Target abort during posted write | R/W | Controls ability of PCI 6254 to assert SERR# when it receives a target abort when attempting to deliver posted write data. SERR# is asserted if this event occurs when this bit is 0 and SERR# enable bit in the command register is set. Reset value is 0. |
| 4 | Master abort on posted write | R/W | Controls ability of PCI 6254 to assert SERR# when it receives a master abort when attempting to deliver posted write data. SERR# is asserted if this event occurs when this bit is 0 and SERR# enable bit in the command register is set. Reset value is 0. |
| 5 | Delayed Configuration or IO write nondelivery | R/W | Controls ability of PCI 6254 to assert SERR# when it is unable to deliver delayed write data after $2^{24}$ (or programmed Maximum Retry count at Timeout Control Register) attempts. SERR# is asserted if this event occurs when this bit is 0 and SERR# enable bit in the command register is set. Reset value is 0. |
| 6 | Delayed read-no data from target | R/W | Controls ability of PCI 6254 to assert SERR# when it is unable to transfer any read data from the target after $2^{24}$ (or programmed Maximum Retry count at Timeout Control Register) attempts. SERR# is asserted if this event occurs when this bit is 0 and SERR# enable bit in the command register is set. Reset value is 0. |
| 7 | Posted Write Data Parity error | R/W1C | Signal SERR# was asserted due to a posted write data parity error on the target bus. Reset to 0. |

**SERR# Status Register (Read/Write) – Offset 98h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 0 | Primary Post Write nondelivery | R/W1C | Signal P_SERR# was asserted because PCI 6254 was unable to deliver posted write data to the target before timeout counter expires. Reset to 0. |
| 1 | Primary Delayed write nondelivery | R/W1C | Signal P_SERR# was asserted because PCI 6254 was unable to deliver delayed write data before time-out counter expires. Reset to 0. |
| 2 | Primary Delayed read failed | R/W1C | Signal P_SERR# was asserted because PCI 6254 was unable to read any data from the target before time-out counter expires. Reset to 0. |
| 3 | Primary Delayed transaction master timeout | R/W1C | Signal P_SERR# was asserted because a master did not repeat a read or write transaction before the master timeout counter expired on the initiator's bus. Reset to 0. |
| 4 | Secondary Post Write nondelivery | R/W1C | Signal S_SERR# was asserted because PCI 6254 was unable to deliver posted write data to the target before timeout counter expires. Reset to 0. |
| 5 | Secondary Delayed write nondelivery | R/W1C | Signal S_SERR# was asserted because PCI 6254 was unable to deliver delayed write data before time-out counter expires. Reset to 0. |
| 6 | Secondary Delayed read failed | R/W1C | Signal S_SERR# was asserted because PCI 6254 was unable to read any data from the target before time-out counter expires. Reset to 0. |
| 7 | Secondary Delayed transaction master timeout | R/W1C | Signal S_SERR# was asserted because a master did not repeat a read or write transaction before the master timeout counter expired on the initiator's bus. Reset to 0. |

## 19.2.5 GPIO Registers

**GPIO[3:0] Output Data Register - Offset 99h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 3:0 | GPIO[3:0] output write 1 to clear | R/W1 TC | Writing 1 to any of these bits drives the corresponding bit low on the GPIO[3:0] bus if it is programmed as output. Writing 0 has no effect.<br>Read returns the last written value.<br>Resets to 0. |
| 7:4 | GPIO[3:0] output write 1 to set | R/W1 TC | Writing 1 to any of these bits drives the corresponding bit high on the GPIO[3:0] bus if it is programmed as output. Writing 0 has no effect.<br>Read returns the last written value.<br>Resets to 0. |

**GPIO[3:0] Output Enable Register - Offset 9Ah**

| Bit | Function | Type | Description |
|---|---|---|---|
| 3:0 | GPIO output enable write 1 to clear | R/W1 TC | Writing 1 to any of these bits drives the corresponding bit on the GPIO[3:0] bus as input only. Writing 0 has no effect.<br>Read returns the last value written.<br>Resets to 0. |
| 7:4 | GPIO output enable write 1 to set | R/W1 TC | Writing 1 to any of these bits drives the corresponding bit on the GPIO[3:0] bus as output. GPIO[3:0] then drives the value set in the output data register (reg 99h). Writing 0 has no effect.<br>Read returns the last written value.<br>Resets to 0. |

**GPIO[3:0] Input Data Register - Offset 9Bh**

| Bit | Function | Type | Description |
|---|---|---|---|
| 3:0 | Reserved | R/O | Reserved |
| 7:4 | GPIO[3:0] input data | R/O | This read-only register reads the state of the GPIO[3:0] pins. The state is updated on the PCI clock cycle following a change in the GPIO[3:0] state. |

### Hot Swap Switch and ROR Register Control (R/W) – Offset 9Ch

| Bit | Function | Type | Description |
|---|---|---|---|
| 0 | **Hot Swap extraction switch** | R/W | **Hot Swap extraction switch :** Software switch used to signal extraction of board. If set, board is in inserted state. Writing a '0' to this bit will signal the pending extraction of the board. |
| 6-1 | **Reserved** | R/O | **Reserved** |
| 7 | **ROR Write Enable** | R/W | **Read Only Registers Write Enable:** Subsystem Vender ID at Register 2Ch and Subsystem ID Register at 2Eh are normally Read Only. Setting this bit to 1 will enable write to such Read Only ID Registers.<br><br>Power Management Registers DEh, E0h, and E3h are normally Read Only. Setting this bit to 1 will enable write to all Read Only Power Management Registers.<br><br>This bit must be cleared after the desired values have been modified in the Read Only Registers. |

### GPIO[7:4] Output Data Register - Offset 9Dh

| Bit | Function | Type | Description |
|---|---|---|---|
| 3:0 | GPIO[7:4] output write 1 to clear | R/W1 TC | Writing 1 to any of these bits drives the corresponding bit low on the GPIO[7:4] bus if it is programmed as output. Writing 0 has no effect.<br><br>Defaults to 0 |
| 7:4 | GPIO[7:4] output write 1 to set | R/W1 TC | Writing 1 to any of these bits drives the corresponding bit high on the GPIO[7:4] bus if it is programmed as output. Writing 0 has no effect |

### GPIO[7:4] Output Enable Register - Offset 9Eh

| Bit | Function | Type | Description |
|---|---|---|---|
| 3:0 | GPIO[7:4] output enable write 1 to clear | R/W1 TC | Writing 1 to any of these bits drives the corresponding bit on the GPIO[7:4] bus as input only. Writing 0 has no effect, reads returns last value written.<br><br>Defaults to 0 |
| 7:4 | GPIO[7:4] output enable write 1 to set | R/W1 TC | Writing 1 to any of these bits drives the corresponding bit on the GPIO[7:4] bus as output. GPIO[7:4] then drives the value set in the output data register (reg 65h). Writing 0 has no effect, reads returns last value written.<br><br>Defaults to 0 |

### GPIO[7:4] Input Data Register - Offset 9Fh

| Bit | Function | Type | Description |
|---|---|---|---|
| 3:0 | Reserved | R/O | Reserved |
| 7:4 | GPIO[7:4] input data | R/O | This read-only register reads the state of the GPIO[7:4] pins. The state is updated on the PCI clock cycle following a change in the GPIO[7:4] state. |

### Power Up Status Register - Offset A0h

| Bit | Function | Type | Description |
|---|---|---|---|
| 7-0 | Power up Status | R/O | **Power up latched Status bits:** Upon PWRGD (power good), the status of GPIO[15:8] are latched in this registers. User can choose to use such status for any desired option setting or checking.<br>**Some Recommended Use (Must be 3.3V input):**<br>**GPIO15: Primary Power State:** 1 = Primary port power is stable.<br>**GPIO14: Secondary Power State:** 1 = Secondary port power is stable. |

### GPIO[15:8] Output Data Register - Offset A1h

| Bit | Function | Type | Description |
|---|---|---|---|
| 7:0 | GPIO[15:8] output data | R/W | GPIO[15:8] output data. Defaults to 0 |

### GPIO[15:8] Output Enable Register - Offset A2h

| Bit | Function | Type | Description |
|---|---|---|---|
| 7:0 | GPIO[15:8] output enable | R/W | Writing 1 to any of these bits drives the corresponding bit on the GPIO[15:8] bus as output.<br>Defaults to 0 |

### GPIO[15:8] Input Data Register - Offset A3h

| Bit | Function | Type | Description |
|---|---|---|---|
| 7:0 | GPIO[15:8] input data | R/O | This read-only register reads the state of the GPIO[15:8] pins. The state is updated on the PCI clock cycle following a change in the GPIO[15:8] state. |

### 19.2.6 Direct Message Interrupt Registers

When enabled, a write command to the following message registers can cause a PCI interrupt. The Direct Message Interrupt Registers allow faster than doorbell register responds with the encoded interrupt message available. S_INTA# will be activated by Downstream messages while P_INTA# will be activated by Upstream messages.

**Upstream Message 0 Register - Offset A4h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 7:0 | Upstream 0 Register | R/W | Secondary port masters can write data to this register for Primary port devices to read. Reading this register clears the secondary message interrupt status bit. |

**Upstream Message 1 Register - Offset A5h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 15:8 | Upstream Message 1 Register | R/W | Secondary port masters can write data to this register for Primary port devices to read. Reading this register clears the secondary message interrupt status bit. |

**Upstream Message 2 Register - Offset A6h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 23:16 | Upstream Message 2 Register | R/W | Secondary port masters can write data to this register for Primary port devices to read. Reading this register clears the secondary message interrupt status bit. |

**Upstream Message 3 Register - Offset A7h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 31:24 | Upstream Message 3 Register | R/W | Secondary port masters can write data to this register for Primary port devices to read. Reading this register clears the secondary message interrupt status bit. |

### Downstream Message 0 Register - Offset A8h

| Bit | Function | Type | Description |
|---|---|---|---|
| 7:0 | Downstream Message 0 Register | R/W | Primary port masters can write data to this register for Secondary port devices to read. Reading this register clears the secondary message interrupt status bit. |

### Downstream Message 1 Register - Offset A9h

| Bit | Function | Type | Description |
|---|---|---|---|
| 15:8 | Downstream Message 1 Register | R/W | Primary port masters can write data to this register for Secondary port devices to read. Reading this register clears the secondary message interrupt status bit. |

### Downstream Message 2 Register - Offset AAh

| Bit | Function | Type | Description |
|---|---|---|---|
| 23:16 | Downstream Message 2 Register | R/W | Primary port masters can write data to this register for Secondary port devices to read. Reading this register clears the secondary message interrupt status bit. |

### Downstream Message 3 Register - Offset ABh

| Bit | Function | Type | Description |
|---|---|---|---|
| 31:24 | Downstream Message 3 Register | R/W | Primary port masters can write data to this register for Secondary port devices to read. Reading this register clears the secondary message interrupt status bit. |

## 19.2.7 Message Signal Interrupt Registers

### MSI Capability Identifier (R/O) - Offset ACh

This register is set to 08h. This register identifies this item in the Capabilities List as an MSI register set.

### Next Item Pointer (R/O) - Offset ADh

Set to 00h. It indicates the end of the Capabilities list.

### MSI Control - Offset AEh

This register provides system software control over MSI.

| Bit | Function | Type | Description |
|---|---|---|---|
| 0 | MSI Enable | R/W | System configuration software sets this bit to enable MSI. |
| 3-1 | Multiple Message Capable | R/O | System software reads this field to determine the number of requested messages. |
| 6-4 | Multiple Message Enable | R/W | System software writes to this field to indicate the number of allocated messages. |
| 7 | 64-bit Address Capable | R/O | System configuration software reads this bit. |
| 15-8 | Reserved | R/O | Reserved. Reset to 0 |

### MSI Address - Offset B0h

This register provides system software control over MSI.

| Bit | Function | Type | Description |
|---|---|---|---|
| 1-0 | Reserved | R/O | Reserved. Reset to 0. |
| 31-2 | Message Address | R/W | System-specified message address. |

### MSI Upper Address - Offset B4h

This register provides system software control over MSI.

| Bit | Function | Type | Description |
|---|---|---|---|
| 31-0 | Message Upper Address | R/W | System-specified message upper address. |

### MSI Data - Offset B8h

This register provides system software control over MSI.

| Bit | Function | Type | Description |
|---|---|---|---|
| 15-0 | Message Data | R/W | System-specified message. Each MSI function is allocated up to 32 unique messages. |
| 31-16 | Reserved | R/O | Reserved. Reset to 0. |

### 19.2.8 Doorbell and Miscellaneous Interrupt Registers

When there are active downstream interrupt sources, S_INTA# will be activated. When there are active upstream interrupt sources, P_INTA# will be activated.

**Downstream Doorbell Interrupt Enable - Offset C0h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 15:0 | Secondary interrupt requests Enable | R/W | Request to Secondary port - Software Interrupt requests enable. |

**Downstream Doorbell Interrupt Requests - Offset C2h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 15:0 | Doorbell interrupts | R/W | Primary masters setting these bits to 1 will cause interrupts on the Secondary Port. As long as this bit is 1, the corresponding interrupt status bit cannot be cleared and new interrupts will be caused. Therefore this bit should be cleared immediately after it has been set to generate an interrupt. |

**Upstream Doorbell Interrupt Enable - Offset C4h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 15:0 | Primary interrupt requests Enable | R/W | Request to Primary port - Software Interrupt requests enable. |

**Upstream Doorbell Interrupt Requests - Offset C6h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 15:0 | Doorbell interrupts | R/W | Secondary masters setting these bits to 1 will cause interrupts on the Primary Port. As long as this bit is 1, the corresponding interrupt status bit cannot be cleared and new interrupts will be caused. Therefore this bit should be cleared immediately after it has been set to generate an interrupt. |

**Downstream Doorbell Interrupt Status - Offset C8h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 15:0 | Secondary interrupt requests | R/W1C | Any set bit indicates the corresponding Secondary Software Interrupt request to the Secondary Host is detected. |

### Downstream Interrupt Status - Offset CAh

| Bit | Function | Type | Description |
|---|---|---|---|
| 0 | Downstream Message 0 | R/W1C | Primary to Secondary message 0 has been written. |
| 1 | Downstream Message 1 | R/W1C | Primary to Secondary message 1 has been written. |
| 2 | Downstream Message 2 | R/W1C | Primary to Secondary message 2 has been written. |
| 3 | Downstream Message 3 | R/W1C | Primary to Secondary message 3 has been written. |
| 4 | P_RSTIN# Deassertion | R/W1C | Deassertion of P_RSTIN# detected. |
| 5 | P_PME# Deassertion | R/W1C | Deassertion of P_PME# detected. |
| 6 | GPIO15 active Low Interrupt Recommended use: Primary Power is not available | RO | This reflects the inverted state of the GPIO15 pin if this interrupt is enabled. Otherwise it is always 0.<br>*(Rev AA: This feature is not available)* |
| 7 | GPIO5 Active Low Interrupt | RO | This reflects the inverted state of the GPIO5 pin if this interrupt is enabled. Otherwise it is always 0. *(Rev AA: This feature is not available)* |

### Upstream Interrupt Enable - Offset CBh

| Bit | Function | Type | Description |
|---|---|---|---|
| 0 | Upstream Message 0 interrupt Enable | R/W | Secondary to Primary message 0 event interrupt trigger enable. |
| 1 | Upstream Message 1 interrupt Enable | R/W | Secondary to Primary message 1 event interrupt trigger enable. |
| 2 | Upstream Message 2 interrupt Enable | R/W | Secondary to Primary message 2 event interrupt trigger enable. |
| 3 | Upstream Message 3 interrupt Enable | R/W | Secondary to Primary message 3 event interrupt trigger enable. |
| 4 | S_RSTIN# Deassertion Enable | R/W | Deassertion of S_RSTIN# detection enable. |
| 5 | S_PME# Deassertion Enable | R/W | Deassertion of S_PME# detection enable. |
| 6 | Secondary external interrupt at GPIO14 pin | R/W | GPIO14 pin LOW triggers interrupt enable<br>*(Rev AA: This feature is not available)* |
| 7 | Secondary external interrupt at GPIO4 pin | R/W | GPIO4 pin LOW triggers interrupt enable<br>*(Rev AA: This feature is not available)* |

### Upstream Doorbell Interrupt Status - Offset CCh

| Bit | Function | Type | Description |
|---|---|---|---|
| 15:0 | Primary interrupt requests | R/W1C | Any set bit indicates the corresponding Primary Interrupt request to the Primary Host is detected. |

### Upstream Interrupt Status - Offset CEh

| Bit | Function | Type | Description |
|---|---|---|---|
| 0 | Upstream Message 0 | R/W1C | Secondary to Primary message 0 has been written. |
| 1 | Upstream Message 1 | R/W1C | Secondary to Primary message 1 has been written. |
| 2 | Upstream Message 2 | R/W1C | Secondary to Primary message 2 has been written. |
| 3 | Upstream Message 3 | R/W1C | Secondary to Primary message 3 has been written. |
| 4 | S_RSTIN# Deassertion | R/W1C | Deassertion of S_RSTIN# detected. |
| 5 | S_PME# Deassertion | R/W1C | Deassertion of S_PME# detected. |
| 6 | GPIO14 active Low Interrupt<br>Recommended use:<br>Secondary Power is not available | RO | This reflects the inverted state of the GPIO14 pin if this interrupt is enabled. Otherwise it is always 0.<br>*(Rev AA: This feature is not available)* |
| 7 | GPIO4 Active Low Interrupt | RO | This reflects the inverted state of the GPIO4 pin if this interrupt is enabled. Otherwise it is always 0.<br>*(Rev AA: This feature is not available)* |

### Downstream Interrupt Enable - Offset CFh

| Bit | Function | Type | Description |
|---|---|---|---|
| 0 | Downstream Message 0 Interrupt Enable | R/W | Primary to Secondary message 0 event interrupt trigger enable. |
| 1 | Downstream Message 1 Interrupt Enable | R/W | Primary to Secondary message 1 event interrupt trigger enable. |
| 2 | Downstream Message 2 Interrupt Enable | R/W | Primary to Secondary message 2 event interrupt trigger enable. |
| 3 | Downstream Message 3 Interrupt Enable | R/W | Primary to Secondary message 3 event interrupt trigger enable. |
| 4 | P_RSTIN# Deassertion Enable | R/W | Deassertion of P_RSTIN# detection enable. |
| 5 | P_PME# Deassertion Enable | R/W | Deassertion of P_PME# detection enable. |
| 6 | Primary external interrupt at GPIO15 pin | R/W | GPIO15 pin LOW triggers interrupt enable<br>*(Rev AA: This feature is not available)* |
| 7 | Primary external interrupt at GPIO5 pin | R/W | GPIO5 pin LOW triggers interrupt enable<br>*(Rev AA: This feature is not available)* |

**NT Configuration Own Semaphore - Offset D2h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 0 | NT Configuration Own Semaphore | R/W1C | When either Primary or Secondary port does a configuration read to this semaphore bit, it returns "0" if there is no configuration read beforehand. Such a read will then cause this semaphore bit to become "1" automatically. Any further read by other Primary or Secondary masters will see a "1" (already owned). This bit must be cleared by the owner master using a configuration write "1" to this register.<br><br>Software can check this semaphore status via register D8h bit 0 without taking ownership. |
| 7:1 | Reserved | RO | Reserved |

## 19.2.9 Extended Registers

There are eight 32bit sticky scratch registers available in PCI 6254 and they are at extended address 0h-7h. Address translation registers are also located in the Extended Register area.

**Extended Register Index - Offset D3h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 7:0 | Extended Index address | R/W | Index address for extended registers |

**Extended Register Dataport - Offset D4h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 31:0 | Extended Registers Dataport | R/W | A Configuration WRITE will cause the data presented at this port to be written into the register addressed by the Extended Register Index.<br><br>A Configuration READ will cause the data from the register addressed by the Extended Register Index to be presented to this port. |

### Extended Registers

| Register | | | | Index |
|---|---|---|---|---|
| 32 Bit Sticky Register 0 | | | | 0h |
| 32 Bit Sticky Register 1 | | | | 1h |
| 32 Bit Sticky Register 2 | | | | 2h |
| 32 Bit Sticky Register 3 | | | | 3h |
| 32 Bit Sticky Register 4 | | | | 4h |
| 32 Bit Sticky Register 5 | | | | 5h |
| 32 Bit Sticky Register 6 | | | | 6h |
| 32 Bit Sticky Register 7 | | | | 7h |
| **Upstream BAR 0 Translation Address** | | | | 8h |
| **Upstream BAR 1 Translation Address** | | | | 9h |
| **Upstream BAR 2 Translation Address** | | | | Ah |
| **Upstream Translation Enable** | **Upstream BAR 2 Translation Mask** | **Upstream BAR 1 Translation Mask** | **Upstream BAR 0 Translation Mask** | Bh |
| **Downstream BAR 0 Translation Address** | | | | Ch |
| **Downstream BAR 1 Translation Address** | | | | Dh |
| **Downstream BAR 2 Translation Address** | | | | Eh |
| **Downstream Translation Enable** | **Downstream BAR 2 Translation Mask** | **Downstream BAR 1 Translation Mask** | **Downstream BAR 0 Translation Mask** | Fh |

**32 Bit Sticky Scratch Registers - Extended Register Index 0h-7h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 31:0 | Scratch Register | R/W | Sticky Scratch register. Upon Power Good, their values are undefined. If Power is Good, P_RSTIN# and S_RSTIN# active inputs do not affect their pre-existing value. |

### 19.2.9.1 Address Translation Control Registers

**Upstream BAR 0 Translation Address - Extended Register Index 8h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 31:0 | Upstream BAR 0 Translation Address | R/W | BAR 0 translation address. Bits 11:0 are read only and always 0. Only address A31:A12 are translated. Lower address bits will be passed. |

**Upstream BAR 1 Translation Address - Extended Register Index 9h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 31:0 | Upstream BAR 1 Translation Address | R/W | BAR 1 translation address. Bits 19:0 are read only and always 0. Only address A31:A20 are translated. Lower address bits will be passed. |

**Upstream BAR 2 Translation Address - Extended Register Index Ah**

| Bit | Function | Type | Description |
|---|---|---|---|
| 31:0 | Upstream BAR 2 Translation Address | R/W | BAR 2 translation address. Bits 19:0 are read only and always 0. Only address A31:A12 are translated. Lower address bits will be passed.<br>If BAR1 is configured as 64bit BAR, then this register contains the upper 32 bits of the BAR1 translation address. |

### Upstream BAR Control - Offset Bh

| Bit | Function | Type | Description |
|---|---|---|---|
| **Upstream BAR 0 Translation Mask** | | | |
| 4:0 | MSB position of address mask | R/W | Number of local address bits for BAR 0 mask.<br>Default = 1Fh (BAR Disabled)<br>This value must be at least 2 to indicate the masking of A3-A0 and must not exceed the value 1Eh to indicate the masking of A30-A0. |
| 5 | Reserved | R/O | Reserved |
| 6 | BAR Type | R/W | If 1, BAR 0 points to I/O space, else Memory<br>Defaults to 0<br>(For PCI 6254 Rev AA, when programmed to 1, IO SPACE must also be enabled in the PCI Command Register at 4h bit 0.) |
| 7 | Prefetchable | R/W | If 1, area pointed to by BAR 0 is in prefetchable area, else area is non-prefetchable.<br>Defaults to 0 |
| **Upstream BAR 1 Translation Mask** | | | |
| 13:8 | MSB position of address mask | R/W | Number of local address bits for BAR 1 mask.<br>Default = 3Fh (BAR disabled)<br>This value must be at least 2 to indicate the masking of A3-A0 and must not exceed the value 3Eh to indicate the masking of A62-A0. |
| 14 | BAR Type | R/W | If 0, BAR 1 is only a 32-bit BAR. Otherwise, BAR1 is a 64-bit BAR.<br>Defaults to 0 |
| 15 | Prefetchable | R/W | If 1, area pointed to by BAR 1 is in prefetchable area, else area is non-prefetchable.<br>Defaults to 0 |
| **Upstream BAR 2 Translation Mask** | | | |
| 20:16 | MSB position of address mask | R/W | Number of local address bits for BAR 2 mask.<br>Default = 1Fh (BAR disabled)<br>This value must be at least 2 to indicate the masking of A3-A0 and must not exceed the value 1Eh to indicate the masking of A30-A0. |
| 22:21 | Reserved | R/O | Reserved |
| 23 | Prefetchable | R/W | If 1, area pointed to by BAR 2 is in prefetchable area, else area is non-prefetchable.<br>Defaults to 0 |

| Upstream Translation Enable | | | |
|---|---|---|---|
| 24 | Upstream BAR 0 Enable | R/W | If 1, address translation using BAR 0 is enabled<br>Defaults to 0 |
| 25 | Upstream BAR 1 Enable | R/W | If 1, address translation using BAR 1 is enabled<br>Defaults to 0 |
| 26 | Upstream BAR 2 Enable | R/W | If 1, address translation using BAR 2 is enabled<br>Defaults to 0 |
| 30:27 | Reserved | R/O | Reserved |
| 31 | S_PORT READY | R/W | Upon S_RSTIN#, this bit is cleared. This bit should be set by Secondary port master upon completion of Secondary port initialization.<br>When P_BOOT = 0 (Secondary port has boot priority), Primary port master access to PCI Standard BAR configurations at 10h-1Bh will be retried until S_PORTREADY bit is set.<br>When P_BOOT = 0 and when this bit is "0", all cross bridge traffic initiated by Primary port will be returned with RETRY.<br>The PORT_READY mechanism does not have the above effect if the special fixed size cross bridge communication window is enabled when the XB_MEM input is "1". |

### Downstream BAR 0 Translation Address - Extended Register Index Ch

| Bit | Function | Type | Description |
|---|---|---|---|
| 31:0 | Downstream BAR 0 Translation Address | R/W | BAR 0 translation address. Bits 11:0 are read only and always 0. Only address A31:A12 are translated. Lower address bits will be passed. |

### Downstream BAR 1 Translation Address - Extended Register Index Dh

| Bit | Function | Type | Description |
|---|---|---|---|
| 31:0 | Downstream BAR 0 Translation Address | R/W | BAR 1 translation address. Bits 19:0 are read only and always 0. Only address A31:A20 are translated. Lower address bits will be passed. |

### Downstream BAR 2 Translation Address - Extended Register Index Eh

| Bit | Function | Type | Description |
|---|---|---|---|
| 31:0 | Downstream BAR 0 Translation Address | R/W | BAR 2 translation address. Bits 11:0 are read only and always 0. Only address A31:A12 are translated. Lower address bits will be passed.<br>If BAR1 is configured as 64bit BAR, then this register contains the upper 32 bits of the BAR1 translation address. |

**Downstream BAR Control - Offset Fh**

| Bit | Function | Type | Description |
|---|---|---|---|
| **Downstream BAR 0 Translation Mask** | | | |
| 4:0 | MSB position of address mask | R/W | Number of local address bits for BAR 0 mask.<br>Default = 1Fh (BAR Disabled)<br>This value must be at least 2 to indicate the masking of A3-A0 and must not exceed the value 1Eh to indicate the masking of A30-A0. |
| 5 | Reserved | R/O | Reserved |
| 6 | BAR Type | R/W | If 1, BAR 0 points to I/O space, else Memory<br>Defaults to 0<br>(For PCI 6254 Rev AA, when programmed to 1, IO SPACE must also be enabled in the PCI Command Register at 4h bit 0.) |
| 7 | Prefetchable | R/W | If 1, area pointed to by BAR 0 is in prefetchable area, else area is non-prefetchable.<br>Defaults to 0 |
| **Downstream BAR 1 Translation Mask** | | | |
| 13:8 | MSB position of address mask | R/W | Number of local address bits for BAR 1 mask.<br>Default = 3Fh (BAR Disabled)<br>This value must be at least 2 to indicate the masking of A3-A0 and must not exceed the value 3Eh to indicate the masking of A62-A0. |
| 14 | BAR Type | R/W | If 0, BAR 1 is only a 32-bit BAR 1. Otherwise, BAR1 is a 64-bit BAR.<br>Defaults to 0 |
| 15 | Prefetchable | R/W | If 1, area pointed to by BAR 1 is in prefetchable area, else area is non-prefetchable.<br>Defaults to 0 |
| **Downstream BAR 2 Translation Mask** | | | |
| 20:16 | MSB position of address mask | R/W | Number of local address bits for BAR 2 mask.<br>Default = 1Fh (BAR Disabled)<br>This value must be at least 2 to indicate the masking of A3-A0 and must not exceed the value 1Eh to indicate the masking of A30-A0. |
| 22:21 | Reserved | R/O | Reserved |
| 23 | Prefetchable | R/W | If 1, area pointed to by BAR 2 is in prefetchable area, else area is non-prefetchable.<br>Defaults to 0 |

| Downstream Translation Enable | | | |
|---|---|---|---|
| 24 | Downstream BAR 0 Enable | R/W | If 1, address translation using BAR 0 is enabled<br>Defaults to 0 |
| 25 | Downstream BAR 1 Enable | R/W | If 1, address translation using BAR 1 is enabled<br>Defaults to 0 |
| 26 | Downstream BAR 2 Enable | R/W | If 1, address translation using BAR 2 is enabled<br>Defaults to 0 |
| 30:27 | Reserved | R/O | Reserved |
| 31 | P_PORT READY | R/W | Upon P_RSTIN#, this bit is cleared. This bit should be set by Primary port master upon completion of Primary port initialization.<br>When P_BOOT = 1 (Primary port has boot priority), Secondary port master access to PCI Standard BAR configurations at 10h-1Bh will be retried until P_PORTREADY bit is set.<br>When P_BOOT = 1 and when this bit is “0”, all cross bridge traffic initiated by Secondary port will be returned with RETRY.<br>The PORT_READY mechanism does not have the above effect if the special fixed size cross bridge communication window is enabled when the XB_MEM input is “1”. |

## 19.2.10 General Control Registers

**Chip Control Register (Read/Write) – Offset D8h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 0 | NT configuration semaphore status | R/O | This is the NT-Configuration Semaphore bit status. Software can check the semaphore status via this bit without taking ownership. |
| 1 | Memory write disconnect control | R/W | Controls when the chip as a target disconnects memory transactions. When 0, disconnects on queue full or on a 4KB boundary. When 1, disconnects on a cache line boundary, as well as when the queue fills or on a 4 KB boundary. Reset value is 0. |
| 2 | Private or Cross Bridge Memory Enable | R/W | **(Transparent Mode)** 1 = Enable Private Memory block reserved only for Secondary Memory Space. The memory space can be programmed using the Private Memory Base/Limit registers. If Limit is smaller than Base, the Private memory space is disabled. Primary port cannot access this memory space through the bridge and the Secondary port will NOT respond to any memory cycles addressing this private memory space.<br><br>(In addition in Rev AA, the Cross-Bridge Memory Window, default at 0-16M space and programmable later by software, is also treated as a private memory block in Transparent Mode.)<br><br>**(Non-Transparent mode) Cross-Bridge Memory Window Enable:** When this bit is "1", PCI 6254 will automatically claim 16M of memory space. This allows the boot up of the Low priority Boot Port to move forward without waiting for the Priority Boot port to program the corresponding Memory BAR registers. If this bit is "1", the Primary or Secondary PORT_READY mechanism will NOT be relevant and access to BAR registers will not be retried. Although the default claims 16M, the BAR registers can be changed by EEPROM or software to change the window size.<br><br>In both Transparent and Non-Transparent modes, this bit resets to the value as presented at the XB_MEM input pin. After reset, this bit can be reprogrammed. |
| 3 | Reserved | R/O | |
| 4 | Secondary bus prefetch Disable | R/W | Controls PCI 6254's ability to prefetch during upstream memory read transactions. When written 0 the chip prefetches and does not forward byte enable bits during memory read transactions. When written 1, PCI 6254 requests only one Dword from the target during memory read transactions and forwards read enable bits. PCI 6254 returns a target disconnect to the requesting master on the first data transfer. Memory read line and memory read multiple transactions are still prefetchable. Reset to 0 in the register.<br><br>**Important: The READ value of this bit is the inverted value of the actual register value. Therefore upon reset, the READ value is 1.** |

| | | | |
|---|---|---|---|
| 5 | Live Insertion mode | R/W | Enables hardware control of transaction forwarding in the PCI 6254. When 0, Pin GPIO[3] has no effect on the I/O, memory, and master enable bits.<br>When 1, if GPIO[3] is set as input, and GPIO[3] is driven high, I/O, memory and master enable bits are disabled. |
| 6 | Transparent Access | R/W | Enables the access to Shadow Registers which are also Transparent Mode registers 44h-5Fh under Non-transparent mode. This defaults to "0". |
| 7 | Reserved | R/O | Reserved**(Set to 0)** |

**Diagnostic Control Register (Read/Write) – Offset D9h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 0 | Chip reset | R/W | Chip and Secondary bus reset. Setting this bit will do a chip reset, without asserting S_RSTOUT# and forcing secondary reset bit in bridge control register to be set. After resetting all bits except for the secondary reset bit in bridge control register, this bit will be cleared but not. Write 0 has no effect. |
| 2:1 | Test mode | R/W | Reserved |
| 3 | Secondary Reset output mask | R/W | 1 = Primary reset input P_RSTIN# active will not cause secondary reset output S_RSTOUT# to become active. If this bit is set, P_RSTIN# will not reset Primary Port control logic state machines. Power NOT Good (PWRGD = 0) clears this bit to 0. |
| 4 | Primary Reset output mask | R/W | 1 = Secondary reset input S_RSTIN# active will not cause primary reset output P_RSTOUT# to become active. Power NOT Good (PWRGD = 0) clears this bit to 0. |
| 5 | Primary Reset | R/W | Forces the assertion of P_RSTOUT# signal pin on the Primary interface. Reset to 0.<br>0=do not force the assertion of P_RSTOUT# pin<br>1=force the assertion "0" at P_RSTOUT# pin<br><br>Secondary Reset control bit is in Bridge Control register. |
| 7:6 | Reserved | R/O | Reserved **(Set to 0).** |

**Arbiter Control Register (Read/Write) – Offset DAh**

| Bit | Function | Type | Description |
|---|---|---|---|
| 8-0 | Arbiter Control | R/W | Each bit controls whether a secondary bus master is assigned to the high priority group or the low priority group. Bits <8:0> correspond to request inputs S_REQ#[8:0], respectively. Reset value is 0. |
| 9 | PCI 6254 priority | R/W | Defines whether the secondary port of PCI 6254 is in high priority group or the low priority group<br>0=low priority group<br>**1=high priority group.**<br>Reset to 1. |
| 15:10 | reserved | R/O | Reserved **(set to '0's)** |

## 19.2.11 Power Management Registers

Power Management registers DEh, E0h and E3h are EEPROM loadable or ROR Write Enable loadable, but is READ ONLY during normal operation.

### Capability Identifier (R/O) – Offset DCh

This register is set to 01h to indicate power management interface registers.

### Next Item Pointer (R/O) – Offset DDh

Set to E4h. This field provides an offset into the function's PCI Configuration Space pointing to the location of next item in the function's capability list. In PCI 6254, this points to the Hot Swap registers.

### Power Management Capabilities(R/W) – Offset DEh

| Bit | Function | Type | Description |
|---|---|---|---|
| 0-2 | Version | R/O | This register is set to 001b, indicating that this function complies with Rev 1.0 of the PCI Power Management Interface Specification |
| 3 | PME Clock | R/O | This bit is a '0', indicating that PCI 6254 does not support PME# signaling |
| 4 | Auxiliary Power Source | R/O | This bit is set to '0' since PCI 6254 does not support PME# signaling |
| 5 | DSI | R/O | Device Specific Initialization. Returns '0' indicating that PCI 6254 does not need special initialization |
| 6-8 | Reserved | R/O | Reserved |
| 9 | D1 Support | R/O | Returns '1' indicating that PCI 6254 supports the D1 device power state |
| 10 | D2 Support | R/O | Returns '1' indicating that PCI 6254 supports the D2 device power state |
| 11-15 | PME Support | R/O | Set to "0601" in Revision AA. Set to "7E01" in Revision AB. |

### Power Management Control/ Status(R/W) – Offset E0h

| Bit | Function | Type | Description |
|---|---|---|---|
| 0-1 | Power State | R/W | This 2-bit field is used both to determine the current power state of a function and to set the function into a new power state. The definition of the field values is given below.<br>00b - D0<br>01b - D1<br>10b - D2<br>11b – D3hot |
| 2-7 | Reserved | R/O | Reserved |
| 8 | PME Enable | R/W | This bit is set to '0' since PCI 6254 does not support PME# signaling. |
| 9-12 | Data Select | R/O | This field returns '0000b' indicating PCI 6254 does not return any dynamic data. |
| 13-14 | Data Scale | R/O | Returns '00b' when read. PCI 6254 does not return any dynamic data. |
| 15 | PME Status | R/W | This bit is set to '0' since PCI 6254 does not support PME# signaling. |

**PMCSR Bridge Support(R/W) – Offset E2h**

| Bit | Function | Type | Description |
|---|---|---|---|
| 0-5 | Reserved | R/O | Reserved |
| 6 | B2/B3 Support for D3hot | R/O | This bit reflects the state of the BPCC input pin. A '1' indicates that when PCI 6254 is programmed to D3hot state the secondary bus's clock is stopped. |
| 7 | Bus Power Control Enable | R/O | This bit reflects the state of the BPCC input pin. A '1' indicates that the power management state of the secondary bus follows that of PCI 6254 with one exception, D3hot state. |

**Power Management Data Register (R/W) – Offset E3h**

This register is EEPROM loadable or ROR Write Enable loadable, but is READ ONLY during normal operation.

## 19.2.12 Hot Swap Registers

### Capability Identifier (R/O) – Offset E4h

This register is **set to 06h** to indicate Hot Swap interface registers.

### Next Item Pointer (R/O) - Offset E5h

**Set to E8h.** This field provides an offset into the function's PCI Configuration Space pointing to the location of next item in the function's capability list. In PCI 6254, this points to the Vital Product Data (VPD) registers.

### Hot Swap Register(R/W) – Offset E6h

(For PCI 6254 rev AA, register E6h can only be accessed from the Primary Port. In Non-Transparent mode and when the Secondary port is connected to Hot Swap connector, in order to access this register, the host needs to establish handshake with the intelligent subsystem on the Primary port to instruct the primary port subsystem to write this register)

| Bit | Function | Type | Description |
|---|---|---|---|
| 0 | DHA | R/W | Device Hiding Arm. Reset to 0.<br>1 = Arm Device Hiding<br>0 = Disarm Device Hiding<br>DHA is set to 1 by hardware during Hot Swap port PCI RSTIN# going inactive and handle switch is still unlocked. The locking of the handle will clear this bit. |
| 1 | EIM<br>ENUM# Mask Status | R/W | Enables or disables ENUM# assertion. Reset to 0.<br>0 = enable ENUM# signal<br>1 = mask off ENUM# signal |
| 2 | PIE | R/O | Pending INSert or EXTract: This bit is set when either INS or EXT is “1” or INS is armed (Write 1 to EXT bit).<br>1 = either an insertion or an extraction is in progress.<br>0 = Neither is pending |
| 3 | LOO<br>LED status | R/W | Indicates if LED is on or off. Reset to 0.<br>0 = LED is off<br>1 = LED is on |
| 5-4 | PI | R/W | Programming Interface:<br>Hardcode at 01: INS, EST, LOO, EIM and PIE, Device Hiding are supported. |
| 6 | EXT<br>Extraction State | R/W1C | This bit is set by hardware when the ejector handle is unlocked and INS = 0. |
| 7 | INS<br>Insertion State | R/W1C | This bit is set by hardware when Hot Swap port RSTIN# is deasserted, EEPROM autoload is completed and the ejector handle is locked.<br>Writing 1 to EXT bit also arms INS. |
| 15:8 | Reserved | R/O | Reserved and a read returns all 0. Write has no effect. |

### 19.2.13 VPD Registers

**Capability Identifier (R/O) - Offset E8h**

This register is set to 03h to indicate VPD registers.

**Next Item Pointer (R/O) - Offset E9h**

Set to 00h.

**VPD Register (R/W) – Offset EAh**

| Bit | Function | Type | Description |
|---|---|---|---|
| 1-0 | **Reserved** | R/O | **Reserved** |
| 7-2 | **VPD Address** | R/W | **VPD operation**: Writing a '0' to this bit generates a read cycle from the EEPROM at the VPD address specified in bits 7-2 of this register. This bit will remain at a logic '0' value until EEPROM cycle is finished, then it be set to '1'. Data for reads is available at register ECh<br><br>Writing a '1' to this bit generates a write cycle to the EEPROM at the VPD address specified in bits 7-2 of this register. This bit will remain at a logic '1' value until EEPROM cycle is finished, then it be cleared to '0'. |
| 14-8 | **Reserved** | R/O | **Reserved** |
| 15 | **VPD Operation** | R/W | **VPD operation**: Writing a '0' to this bit generates a read cycle from the EEPROM at the VPD address specified in bits 7-2 of this register. This bit will remain at a logic '0' value until EEPROM cycle is finished, then it be set to '1'. Data for reads is available at register ECh<br><br>Writing a '1' to this bit generates a write cycle to the EEPROM at the VPD address specified in bits 7-2 of this register. This bit will remain at a logic '1' value until EEPROM cycle is finished, then it be cleared to '0'. |

**VPD Data Register (R/W) – Offset ECh**

| Bit | Function | Type | Description |
|---|---|---|---|
| 31-0 | **VPD Data** | R/W | **VPD Data** (EEPROM data[addr + 0x40]) - The least significant byte of this register corresponds to the byte of VPD at the address specified by the VPD Address register. The data read from or written to this register uses the normal PCI byte transfer capabilities. |

## 19.3 Non-Transparent Mode Operation

PCI 6254 can also operate as a Non-Transparent Universal bridge, which can be used for embedded type applications and for application on CPCI SYSTEM and PERIPHERAL slots. In this mode, PCI 6254 uses up to 3 base address registers on each side of the bridge to specify which cycles are passed downstream or upstream, after being translated using the values in the address translation registers. Non-Transparent mode is enabled through the TRANS# pin.

PCI 6254 has the following non-transparent capabilities:

- Downstream address translation
- Upstream address translation
- Separate configuration space for primary and secondary interfaces
- Up to three separate address ranges can be specified by using standard base address register definition
- Support for 32-bit I/O, 32-bit Memory and 64-bit memory address translation
- Translation registers are all EEPROM loadable, so PCI 6254 can operate in non-transparent mode without special software requirements
- Powerful Message register mechanism, doorbells, status and events with interrupt capability to pass information from one side of the bridge to the other side.

## *19.4 Interrupts*

MSI is a non-shared interrupt that enforces data consistency. The system guarantees that any data written by the device prior to sending the MSI has reached its final destination before the interrupt service routine accesses the data. MSI enables PCI 6254 to request service by writing a system-specified message to a system-specified address (PCI DWORD Memory Write transaction). The transaction address specifies the message destination and the transaction data specifies the message. System software initializes the message destination and message during device configuration.

### 19.4.1 Direct Message Interrupts

The PCI 6254 has 4 upstream and 4 downstream message registers which when written to, can generate immediate interrupts to the other side. This is the fastest interrupt mechanisms that PCI 6254 has and is faster in latency than standard doorbell interrupts for software applications.

#### 19.4.1.1 Direct Message Interrupt Operations

When a PCI master wants to communicate with the host on the other side of the PCI 6254 bridge, it can make use any of the 4 message byte registers available. When the master writes an encoded message into this message register, the write action causes an interrupt be generated to the host. The interrupt service routine can first read the interrupt status registers to see which message status bit has been set and read the message register to get the interrupt message. The service routine should then write 1 to the corresponding status bit to clear the status. This allows the service routine to react to the encoded message quickly without performing to many polling of registers.

### 19.4.2 Doorbell Interrupts

The PCI 6254 has 16 upstream and 16 downstream doorbell interrupt registers which when written to, can generate immediate interrupts to the other side.

#### 19.4.2.1 Doorbell Interrupt Operations

When a PCI master wants to communicate with the host on the other side of the PCI 6254 bridge, it can make use any of the 16 doorbell interrupts. When the requesting master first writes 1, then writes 0 to its doorbell interrupt request bit, an interrupt will automatically be generated to the host. The interrupt service routine can read the doorbell status register to find out who is requesting the interrupt and can then go and inquire the corresponding device. The service routine should then write 1 to the corresponding status bit to clear the status.

### 19.4.3 Message Signaled Interrupts (MSI)

MSI is a non-shared interrupt that enforces data consistency. The system guarantees that any data written by the device prior to sending the MSI has reached its final destination before the interrupt service routine accesses the data. MSI enables PCI 6254 to request service by writing a system-specified message to a system-specified address (PCI DWORD Memory Write transaction). The transaction address specifies the message destination and the transaction data specifies the message. System software initializes the message destination and message during device configuration.

Interrupt latency (the time from interrupt signaling to interrupt serving) is system dependent.

### 19.4.3.1 MSI Operation

During configuration time, system software does the followings:

- Scan the function's capability list; the function implements MSI (capability ID of 05h exists).
- Reads the MSI capability structure's Message Control register to determine the function's capabilities.
- Reads the Multiple Message Capable field to determine the number of requested messages.
- Writes the Multiple Message Enable field to allocate either all or a subset of the requested messages.
- Initializes the MSI capability structure's Message Address register and Message Upper Address register with a system-specified message destination address.
- Initializes the MSI capability structure's Message Data register with a two bytes system-specified message (one WORD).

Once MSI is enabled, the function may send messages using a DWORD memory write to the address specified by the contents of the Message Address register. The DWORD that is written is made up of the value in the Message Data register. If the Multiple Message Enable field is non-zero, PCI 6254 can modify the low order bits of the message data to generate multiple messages.

If the MSI write transaction results in a data parity error, the master that originated the MSI write transaction is required to assert SERR# and set the appropriate bit in the Status register. The message receiver must complete the interrupt message transaction independent of when the CPU services the interrupt. If a device requires one interrupt message to be serviced before another, then the device must not send the second interrupt message until the first one has been serviced.

## *19.5 Non-Transparent Mode Boot Up Sequence*

PCI 6254 loads EEPROM on power up and during this time any configuration cycle will end up with retry. Depending on which port is set to have higher boot priority by the P_BOOT input, the lower priority boot master access to the PCI Standard BAR registers will be retired unless XB_MEM input is set to "1". Access to other configuration registers are not affected.

Upon reset, the corresponding port PORTREADY status bit is cleared to indicate that the port is not yet ready for access by the controlling host. High priority boot master (in general is the intelligent subsystem) can allocate a memory and/or I/O region that can be accessed by the low boot priority host (in general, the system control host). After that, the high boot priority master should set its corresponding PORTREADY BIT. Once this bit is set, the retried BAR access configuration cycle from the low boot priority boot master can proceed and therefore the low boot priority boot master can proceed with normal PCI initialization to set up the correct memory/IO space allocation.

There are semaphores that can be used to ensure exclusive access to shared registers.

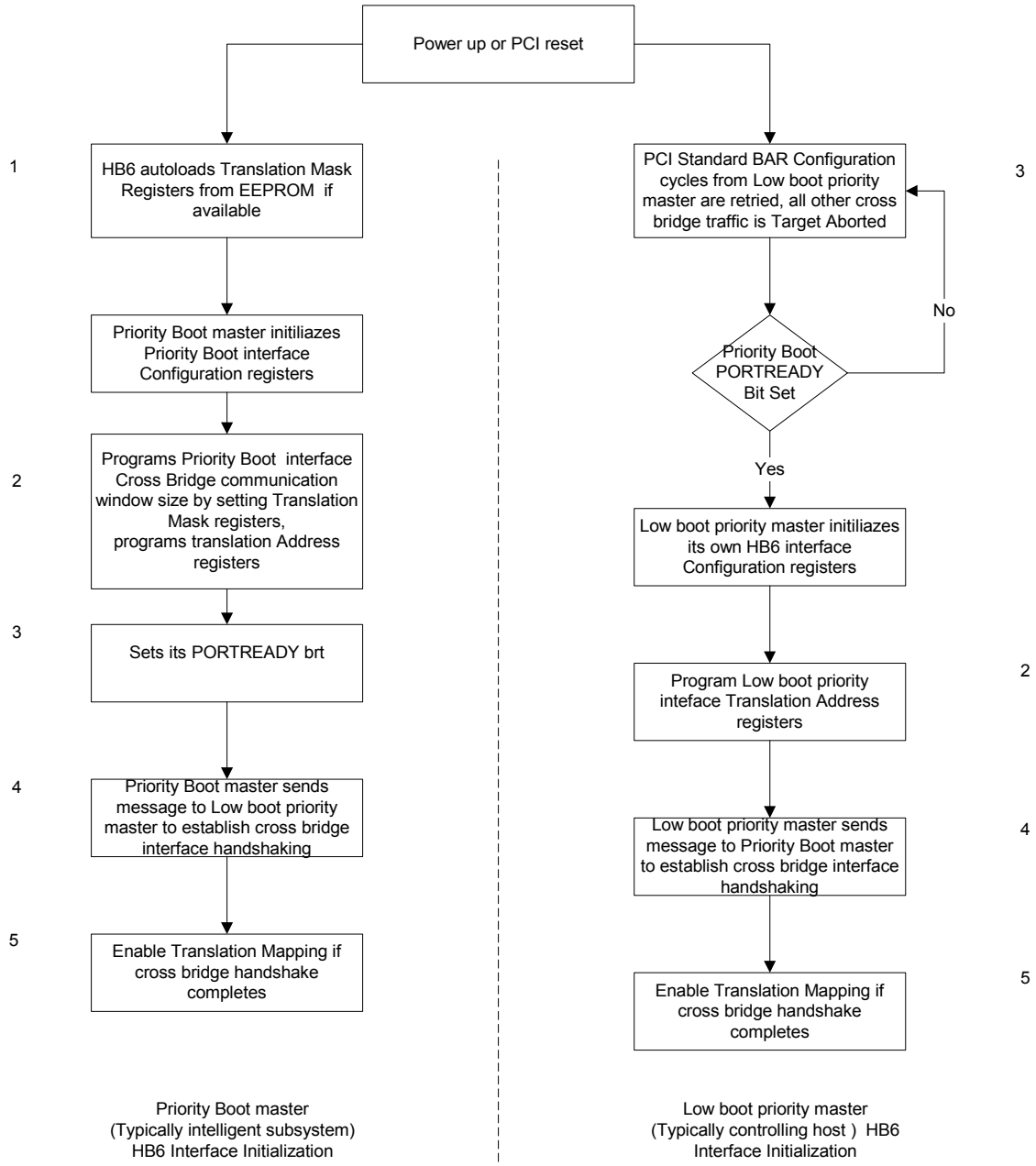
There are multiple cross bridge interrupt mechanisms for use. Direct interrupt mechanism allows user encode message to be written to registers that can cause interrupts. It is up to the designer to decide on the definitions used in the message registers. There are also 16 door bell registers for user to use for cross bridge communications.

Port reset, port power-down can also be configured to cause interrupts for the opposite port to respond.

Note that all the address translation registers can be stored into the EEPROM and then loaded during the EEPROM Autoload process, so that no software is required to setup the translation mechanism between the primary and secondary hosts.

### 19.5.1 Using XB_MEM Input to Avoid initial Retry Latency

The PORTREADY mechanism, which results in Retry for BAR access configuration cycles if subsystem is not yet set up, can be disabled if the XB_MEM input pin is connect to HIGH. In such event, the PCI 6254 hardcoded a fixed cross bridge communication window of 16MB memory space at power up. PCI 6254 will automatically claim such 16M of memory space. This allows the boot up of the Low priority Boot Port to move forward without waiting for the Priority Boot port to program the corresponding Memory BAR registers. When the XB_MEM (PRV_MEM pin in Transparent Mode) pin is "1", the P or S PORT_READY mechanism will NOT be relevant and access to BAR registers will not be retried. Although the default claims 16M, the BAR registers can be changed by EEPROM or software to change the window size.

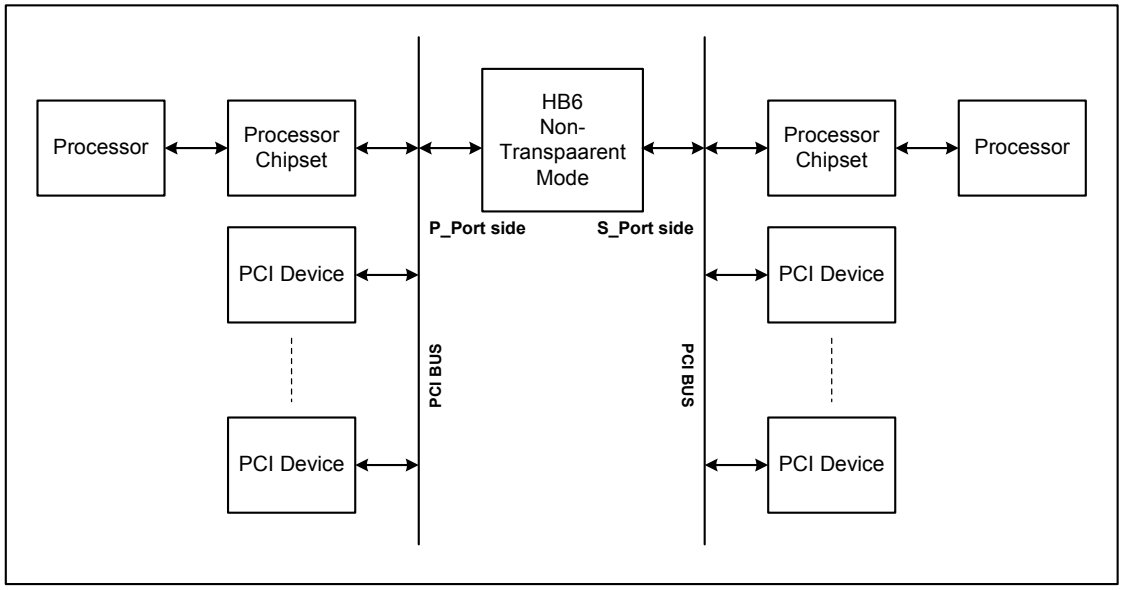



NOTES:
1.Translation mask register values are read from EEPROM offsets 32h, 33h, 3Eh, 3Fh
2. Extended registers at indexed address 8h-Ah for Secondary port, Ch-Eh for Primary port.
3. The PORT_READY bits, at extended registers at indexed address Bh secondary port and Fh for primary port, are cleared upon P_RSTIN_L and S_RSTIN_L..
4. Handshaking can be achieved using Direct Message Interrupts at Register A4h-A8h. Handshaking messages are user-defined status/ command information.
5.Translation can be enabled at Extended Registers at indexed address Fh( Primary ) and Bh( Secondary ).

## *19.6 Non-Transparent Application System Configuration Overview*

The following are some assumptions for this overview.

1. There is at least one processor on each side of PCI Bus.
2. P_Boot input is set to 1: P_Port has higher boot priority and need to finish basic boot up setup first.
3. Primary reset and secondary reset inputs have been applied properly.
4. EEPROM auto load sequence is completed after Primary reset going inactive if EEPROM is used.
5. Eject Handle is closed and Hot Swap procedure has been done if hot-plug application is applied.




### 19.6.1 Memory Allocation Registers Initialization

The following steps demonstrate the initialization procedure of system software to program the memory mapping registers. Since P_Boot is set to “1” in this application note, S_Port, the lower priority port, will be retried during configuration accesses to its BAR0, BAR1 and BAR2 configuration registers. When Primary port intelligent subsystem finishes its basic configuration setup and set the P_Port_Ready bit to “1”, the retried configuration accesses by S_Port master can proceed.

If the above mentioned retries are not desirable, XB_MEM pin input can be used to set a fixed 16MB cross bridge communication windows in BAR0 0x10h configuration register in both sides upon reset. This windows size can be changed by EEPROM or software afterward. This would allow lower priority boot port to move forward without having to wait for the higher priority port to set the corresponding memory BAR. Programming to the respective Upstream Address Translation Registers and Downstream Address Translation Registers is required when address translation mechanism is desired. When XB_MEM pin is “1”, the P_Port_ready and S_Port_ready mechanism will NOT be relevant and configuration accesses to BAR0, 1 and 2 will not be retried.

Many cross bridge related control registers are located in the extended registers area and programmed through configuration register 0xD3h (extended register Index) and 0xD4h (extended register Data Port).

### 19.6.2 Basic Initialization Sequence

**P_Port side (high boot priority port):**

1. Program the desired cross bridge memory window size at extended register 0x0Bh for BAR0, BAR1, and BAR2.
2. If address translation is desired, program the allocated memory location to the Translation Address registers at extended register 0x08h, 0x09h and 0x0Ah and enable the upstream translation enable bits in extended register 0x0Bh.
3. Set P_Port_Ready bit to "1" in bit 31 of extended register 0x0Fh.
4. Check if S_Port_Ready bit is set to "1" then go to step 3. Otherwise, go back to step 2. (Time-out mechanism, to prevent dead lock, can be implemented to safeguard against a faulty subsystem that never sets the S_Port_Ready.)
5. Checks the desired total memory size from configuration register 0x10h, 0x14h and 0x18h and then initialize these registers accordingly per PCI specification. Now the memory access is established in the space specified by configuration register 0x10h, 0x14h, and 0x18h on the P_Port side.
6. Program the rest of configuration register

**S_Port Side (need to wait until high boot priority port is setup):**

1. Check P_Port_Ready bit. If it is set to "1", go to step 2. Otherwise, go back to step 1. (Time-out mechanism, to prevent dead lock, can be implemented to safeguard against a faulty system that never sets the P_Port_Ready.)
2. Check the P_Port assigned total cross bridge memory size from configuration register 0x10h, 0x14h and 0x18h and then initialize these registers accordingly per PCI specification.
3. Program the S_Port allocated memory size to extended register 0x0Fh for BAR0, BAR1, and BAR2.
4. If address translation is desired, program the allocated memory location to the Translation Address registers at extended register 0x0Ch, 0x0Dh and 0x0Eh and enable the downstream translation enable bits in extended register 0x0Fh. Now the memory access is established in the space specified by configuration register 0x10h, 0x14h, and 0x18h on the S_Port side.
5. Set S_Port_Ready bit to "1" in bit 31 of extended register 0x0Bh and program the rest of configuration registers.

Note that all the Address Translation registers can be stored into the EEPROM and loaded during the EEPROM Autoload process. If desired, no software is required to setup the translation mechanism between the primary and secondary hosts. If XB_MEM input is not used, software setting of P_Port_Ready or S_Port_Ready is required to enable access to BAR registers by low boot priority port.

To avoid overlap accesses by software, there are semaphore bits that can be used to ensure exclusive access to shared registers. Please refer to the semaphore section in the PCI 6254 data book.

PCI 6254 provides multiple cross bridge interrupt mechanisms for use. Direct interrupt mechanism allows user encoded message to be written to registers that can cause interrupts. It is up to the designer to decide on the definitions used in these message registers. There are also 16 doorbell registers to use for cross bridge communications.

Port reset, port power-down can also be configured to cause interrupts for the opposite port to respond.

### 19.6.3 Example of Address Setup and Mapping

The following diagram demonstrates the relationships of BAR allocation register, Address Mask (region) and Translation Registers.

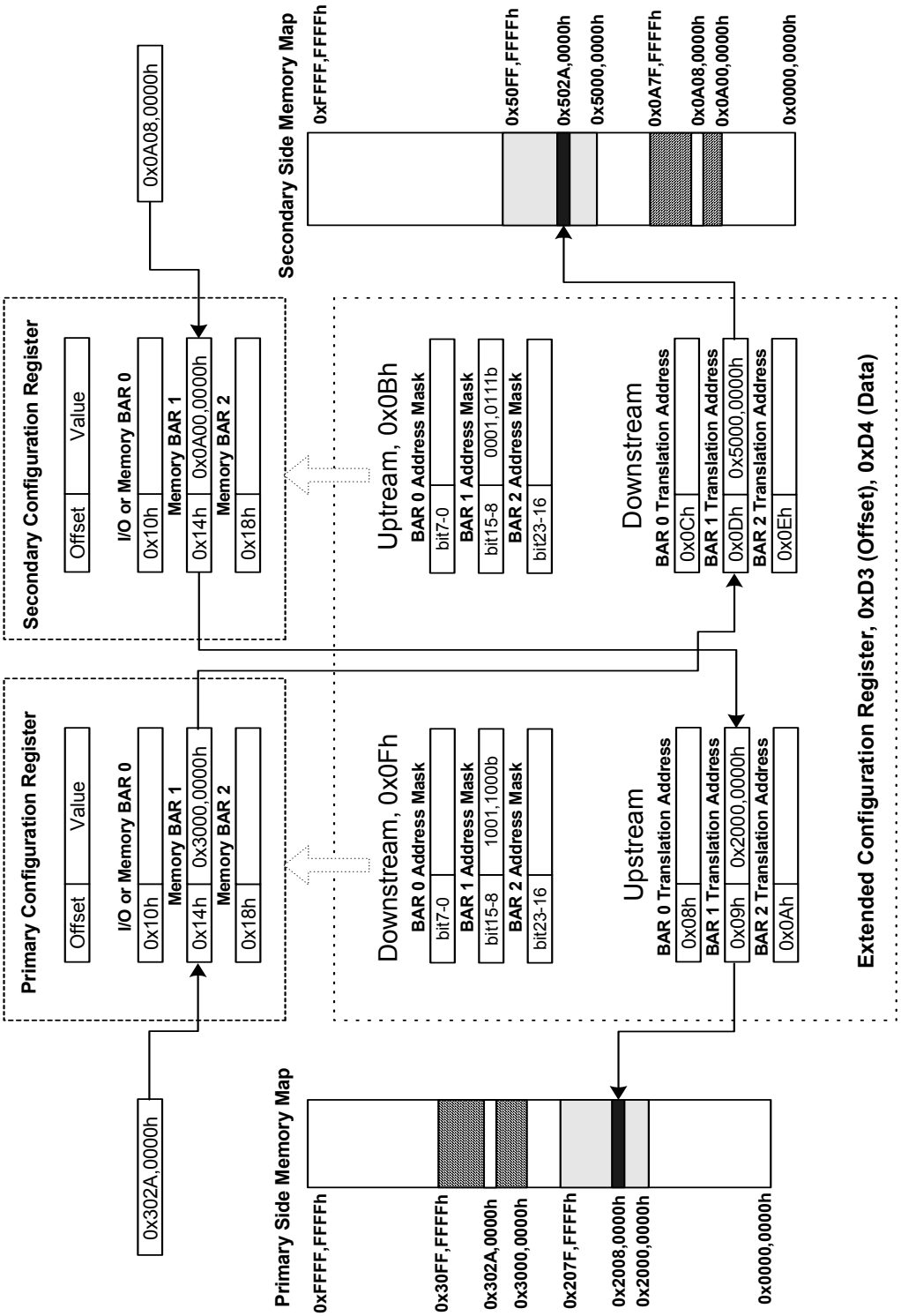
Please note that address translation is only available to higher order address bits. BAR0 and BAR2 translate only A31:A12 and BAR1 only translates A31:A20 or A63:A20.

**For upstream memory access:**

1. Bit 8-15 of extended register 0x0Bh been programmed to 0x0001,0111b to set 8MB memory space for secondary configuration register BAR1 0x14h.
2. Upstream Translation Address Register in extended register 0x09h been programmed to 0x2000,0000h to allocate the 8MB memory space from location 0x2000,000 to 0x207F,FFFFh in primary side of memory space.
3. 0x14h BAR1 register in secondary configuration register been programmed to 0x0A00,0000h.
4. Any memory access between location 0x0A00,0000h to 0x0A7F,FFFFh to PCI BUS in secondary side of memory space is transferred to access location 0x2000,0000h to 0x207F,FFFFh in primary side of memory space.

**For downstream memory access:**

1. Bit 8-15 of extended register 0x0Fh been programmed to 0x1001,1000b to set 16MB memory space for primary configuration register BAR1 0x14h. Bit 15 =1 to indicate the memory space is prefetchable.
2. Downstream Translation Address Register in extended register 0x0Dh been programmed to 0x5000,0000h to allocate the 16MB memory space from location 0x5000,000 to 0x50FF,FFFFh in secondary side of memory space.
3. 0x14h BAR1 register in primary configuration register been programmed to 0x3000,0000h.
4. Any memory access between location 0x3000,0000h to 0x30FF,FFFFh to PCI BUS in primary side of memory space is transferred to access location 0x5000,0000h to 0x50FF,FFFFh in secondary side of memory space.



# 20 IEEE 1149.1 Compatible JTAG Controller

An IEEE 1149.1 compatible Test Access Port (TAP) controller and the associated TAP pins are provided for board level continuity test and diagnostics. The TAP pins assigned are TCK, TDI, TDO, TMS and TRST#. All digital input, output, input/output pins are tested except the TAP pins and clock pin.

The IEEE 1149.1 Test Logic consists of a TAP controller, an instruction register, and a group of test data registers including Bypass, Device Identification and Boundary Scan registers. The TAP controller is a synchronous 16 state machine driven by the Test Clock (TCK) and the Test Mode Select (TMS) pins. An independent power on reset circuit is provided to ensure the machine is in RESET state at power-up.

PCI 6254 implements 3 basic instructions: BYPASS, SAMPLE/PRELOAD, and EXTEST.

# 21 EEPROM

***Important: Wrong EEPROM data can cause the PCI 6254 to lock the system. Designers should provide an optional switch to disable the EEPROM in their board design.***

PCI 6254 has an interface to EEPROM device. The interface can control an ISSI IS24C02 or compatible part, which is organized as 256X8 bits. The EEPROM is used to initialize the registers. After P_RSTIN# is deasserted, PCI 6254 will automatically load data from the EEPROM. The data structure is defined in the following section. The EEPROM interface is organized on 16-bit base in little-endian format, and PCI 6254 supplies a 7-bit EEPROM word address.

The following pins are used for the EEPROM interface:

- EEPCLK: EEPROM clock output
- EEPDATA: EEPROM bi-directional serial data pin
- EE_EN#: LOW input enables EEPROM access.

**Note: The PCI 6254 does not control the EEPROM address inputs. It can only access EEPROM with address inputs set to 0.**

## *21.1 Auto Mode EEPROM Access*

Using auto mode, PCI 6254 can access the EEPROM on a WORD basis via hardware sequencer. Users need only to access a WORD data via PCI 6254 configuration registers for EEPROM START control, address, read/write command. Before each access, software should check the Auto Mode Cycle in Progress status before issuing the next START.

## *21.2 EEPROM Mode at Reset*

Upon P_RSTIN# going high, PCI 6254 auto-loads input for EEPROM automatic load condition if input pin EE_EN_L = 0. (In Rev AA in Non-Transparent mode, EEPROM load is triggered by PWRGD going active instead)

The first offset in the EEPROM contains a signature. If the signature is recognized, register auto-load will commence right after RESET. During the auto-load, PCI 6254 will read sequential words from the EEPROM and write to the appropriate registers.

Before the PCI 6254 registers can be accessed through host, user should check the auto-load condition by reading the EEPAUTO bit. Host access is allowed only after EEPAUTO status becomes '0' which means that the auto load initialization sequence is complete.

## *21.3 PCI 6254 Rev AA only: EEPROM Autoload in Non-Transparent Mode*

In Non-Transparent Mode and upon PWRGD rising edge, the PCI 6254 initiates EEPROM autoload. However for registers that are also reset by P_RSTIN#, including vendor ID and subvendor ID, the autoload will NOT be effective if P_RSTIN# is active. Therefore in order to have effective autoload, the PWRGD rising edge should be aligned with the rising edge of P_RSTIN#.

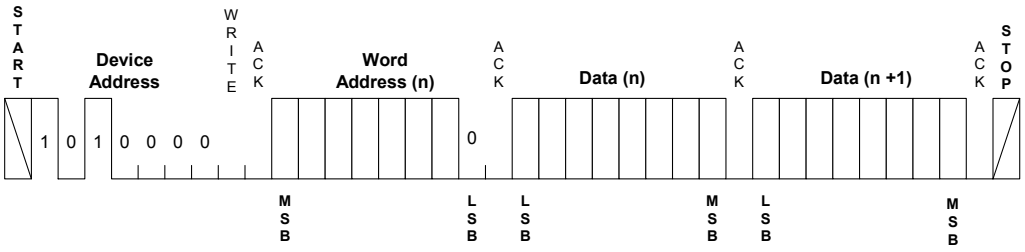
It is also important to note that the EEPROM initialized value will be cleared by any active P_RSTIN# or Power Management initiated internal reset. Only another PWRGD rising edge or a software initiated Chip Reset (register D9h, bit 0) will cause EEPROM load again.

## *21.4 EEPROM Data Structure*

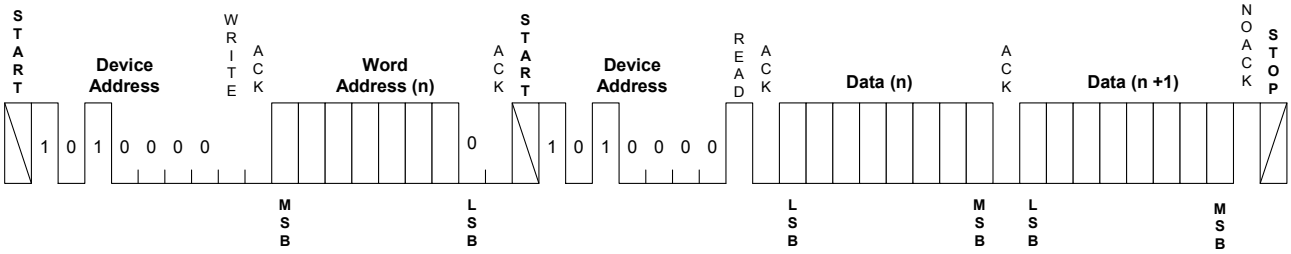
Following the reset, if the condition above is met, PCI 6254 will auto-load the registers with data from EEPROM. The following table describes the data structure used in EEPROM.

The PCI 6254 accesses the EEPROM one word at a time. It is important to note that in the data phase, bit orders are reverse of that of the address phase. PCI 6254 only supports EEPROM Device Address 0.

**Write:**




**Read:**



## 21.4.1 EEPROM Address and Corresponding PCI 6254 Register

| EEPROM Byte Address | PCI Configuration Offset | Description |
|---|---|---|
| 00-01h | | **EEPROM signature:** Autoload will only proceed if it reads a value of 1516h on the first word loaded.<br>0x1516=valid signature, otherwise disable autoloading. |
| 02h | | **Region Enable:** Enables or disables certain regions of the PCI configuration space from being loaded from the EEPROM. Valid combinations are:<br><br>Bit 0: Reserved<br><br>Bits 4-1: 0000 = stop autoload at offset 03h: Group 1<br>0001 = stop autoload at offset 13h: Group 2<br>0011 = stop autoload at offset 23h: Group 3<br>0111 = stop autoload at offset 27h: Group 4<br>1111 = autoload all EEPROM loadable registers: Group 5<br><br>Other combinations are undefined.<br>Bits7-5: reserved |
| 03h | | **Enable Miscellaneous functions:**<br>Bit0: **ISA Enable Control bit Write Protect**: When this bit is set, PCI 6254 will change the standard PCI-to-PCI Bridge Control Register 3Eh bit 2 into read only and ISA Enable feature will not be available.<br>Bit1-7: Reserved |
| **End of Group 1** | | |
| 04-05h | 00-01h | **Vendor ID** |
| 06-07h | 02-03h | **Transparent Device ID**<br>**In Rev AA, EEPROM loaded Non-Transparent Device ID bit 0 is not inverted.** |
| 08h | | **Reserved** |
| 09h | 09h | **Transparent mode Class Code**: Contains low byte of Class Code Register |
| 0Ah-0Bh | 0Ah-0Bh | **Transparent mode Class Code higher bytes**: Contains, upper bytes of Class Code Register |
| 0Ch | 0Eh | **Transparent Header Type** |
| 0Dh | 09h | **Non Transparent mode Class Code**: Contains low byte of Class Code Register |
| 0Eh-0Fh | 0Ah-0Bh | **Non Transparent mode Class Code higher bytes**: Contains, upper bytes of Class Code Register |
| 10h | 0Eh | **Non Transparent Header Type** |
| 11h | 0Fh | **BIST** |
| 12h-13h | 50h | **Internal Arbiter Control** |
| **End of Group 2** | | |
| 14h | 44h | **Primary flow through control** |
| 15h | 45h | **Timeout Control** |
| 16h-17h | 46-47h | **Miscellaneous Options** |
| 18h | 48h | **Primary Initial Prefetch count** |
| 19h | 49h | **Secondary Initial Prefetch count** |
| 1Ah | 4Ah | **Primary Incremental Prefetch Count** |
| 1Bh | 4Bh | **Secondary Incremental Prefetch Count** |
| 1Ch | 4Ch | **Primary Maximum Prefetch Count** |
| 1Dh | 4Dh | **Secondary Maximum Prefetch Count** |
| 1Eh | 4Eh | **Secondary flow through control** |
| 1Fh | E3h | **Power Management Data** |

| | | |
|---|---|---|
| 20h-21h | E0h | **Power Management CSR** |
| 22h- 23h | DEh | **Power management Capabilities** |
| **End of Group 3** | | |
| 24h-25h | 2Ch | **Subsystem Vendor ID**, 2Ch(non-transparent mode) |
| 26h-27h | 2Eh | **Subsystem ID**, 2Eh(non-transparent mode) |
| **End of Group 4** | | |
| 28h | | Reserved |
| 29h | | Bit 0-2: **Upstream address translation enable**<br>Bit 3: **Upstream BAR 0 I/O bit**<br><br>**Bit 7-4:** Bits 15-12 of Upstream BAR 0 translation address |
| 2Ah-2Bh | | **Bits 31-16 of Upstream BAR 0 translation address** |
| 2Ch | | Bit0: **Upstream BAR 0 prefetchable bit**<br>Bit1: **Upstream BAR 1 64 bit**<br>Bit2: **Upstream BAR 2 prefetchable bit**<br>Bit3: **Upstream BAR 1 prefetchable bit**<br>Bit 7-4: **Bits 23-20 of Upstream BAR 1 translation address** |
| 2Dh | | **Bits 31-24 of Upstream BAR 1 translation address** |
| 2Eh-2Fh | | **Bits 15:0 of Upstream BAR 2 translation address** |
| 30h-31h | | **Bits 31:16 of Upstream BAR 2 translation address** |
| 32h-33h | | Bits 4-0: **Upstream BAR 0 translation mask**<br>Bits 10-5: **Upstream BAR 1 translation mask**<br>Bits 15-11: **Upstream BAR 2 translation mask** |
| 34h | | Reserved |
| 35h | | Bit 0-2: **Downstream address translation enable**<br>Bit 3: **Downstream BAR 0 I/O bit**<br><br>**Bit 7-4:** Bits 15-12 of Downstream BAR 0 translation address |
| 36h-37h | | **Bits 31-16 of Downstream BAR 0 translation address** |
| 38h | | Bit0: **Downstream BAR 0 prefetchable bit**<br>Bit1: **Downstream BAR 1 prefetchable bit**<br>Bit2: **Downstream BAR 2 prefetchable bit**<br>Bit3: **Downstream BAR 1 64 bit**<br>Bit 7-4: **Bits 23-20 of Downstream BAR 1 translation address** |
| 39h | | **Bits 31-24 of Downstream BAR 1 translation address** |
| 3Ah-3Bh | | **Bits 15:0 of Downstream BAR 2 translation address** |
| 3Ch-3Dh | | **Bits 31:16 of Downstream BAR 2 translation address** |
| 3Eh-3Fh | | Bits 4-0: **Downstream BAR 0 translation mask**<br>Bits 10-5: **Downstream BAR 1 translation mask**<br>Bits 15-11: **Downstream BAR 2 translation mask** |

# 22 Vital Product Data

PCI 6254 contains the Vital Product Data (VPD) registers as specified in the PCI Local Bus Specification Revision 2.2.

- The VPD information is stored in the EEPROM device along with the Autoload information.

PCI 6254 provides for storage of 192 bytes of VPD data in the EEPROM device.

- VPD related registers are located starting at offset ECh of the PCI configuration space.
- VPD also uses the enhanced capabilities port address mechanism.

# 23 PCI Power Management

PCI 6254 incorporates functionality that meets the requirements of the PCI Power Management Specification, Revision 1.0. These features include:

- PCI power management registers using the enhanced capabilities port (ECP) address mechanism
- Support for D0, $D3_{hot}$ and $D3_{cold}$ power management states
- Support for D0, D1, D2, $D3_{hot}$ and $D3_{cold}$ power management states for devices behind the bridge
- Support of the B2 secondary bus power state when in the $D3_{hot}$ power management state

Table 23-1 below shows the states and related actions that the PCI 6254 performs during power management transitions. (no other transactions are permitted.)

**Table 23-1: States and Related Actions During Power Management Transitions**

| Current State | Next State | Action |
|---|---|---|
| D0 | $D3_{cold}$ | Power has been removed from the PCI 6254. A power-up reset must be performed to bring the PCI 6254 to D0. |
| D0 | $D3_{hot}$ | If enabled to do so by the BPCCE pin, the PCI 6254 will disable the secondary clocks and drive them low. |
| D0 | D2 | Unimplemented power state. The PCI 6254 will ignore the write to the power state bits (power state remains at D0). |
| D0 | D1 | Unimplemented power state. The PCI 6254 will ignore the write to the power state bits (power state remains at D0). |
| $D3_{hot}$ | D0 | The PCI 6254 enables secondary clock outputs and performs an internal chip reset. Signal S_RSTOUT# will not be asserted. All registers will be returned to the reset values and buffers will be cleared. |
| $D3_{hot}$ | $D3_{cold}$ | Power has been removed from the PCI 6254. A power-up reset must be performed to bring the PCI 6254 to D0. |
| $D3_{cold}$ | D0 | Power-up reset. The PCI 6254 performs the standard power-up reset functions. |

## *23.1 P_PME# and S_PME# signals.*

In Transparent Mode, S_PME# is passed through to P_PME#. The use is optional as some designers can choose to connect PME# signal directly from secondary PCI devices to the primary port.

In Non-Transparent mode, depending on the setting of P_BOOT, PME# will be passed from the high boot priority port to the low boot priority port.

The pass through mechanism can be enabled/disabled via the Power Management Control register.
.

# 24 Hot Swap

PCI 6254 incorporates functionality that meets the requirements of the CompactPCI Hot Swap specification PICMG 2.1 R2.0 with High Availability Programming Interface level 1 (PI=1). The CompactPCI Hot Swap register block is located at PCI configuration offset E4h. Designers should refer to the CompactPCI Hot Swap specification for detailed implementation guideline.

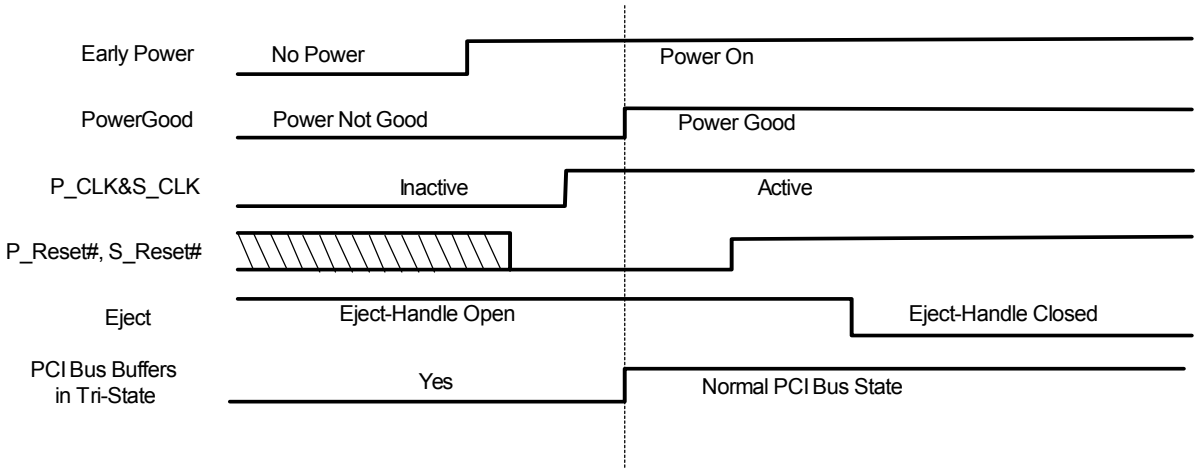
**Important Note:**
**If Hot-Swap feature is not needed, L_STAT and EJECT inputs must be connected to logic "0". Otherwise the PCI 6254 may not function. ENUM# can be left unconnected.**

## *24.1 Early Power Support*

PCI 6254 incorporate Early Power Support in the following ways:

- PCI 6254 can tolerate back end interface being unpowered when fully powered by Early Power. PCI 6254 three-states all PCI signals of a port until its corresponding RSTIN# is deasserted.
- When fully powered by back end power, PCI 6254 three-states all PCI signals of a port until its corresponding RSTIN# is deasserted.

**HB6-AB Hot-Insertion Power Up Sequence Recommendation**




## *24.2 Assignment of Hot Swap Port*

In Non Universal Mode, the Primary port is Hot Swap capable. In Universal Mode, Secondary Port is Hot Plug capable. All Hot Swap related control will be assigned to the corresponding Hot Swap capable port.

## 24.3 Hot Swap Signals

PCI 6254 has the following Hot Swap related pins:

- ENUM#: This is the output signal ENUM# to notify the system host that either a board has been freshly inserted or is about to be extracted. ENUM# is an open collector signal. ENUM# is asserted if INS or EXT bit is set and EIM is 0.
- L_STAT: This is the status BLUE LED. LED is illuminated if RSTIN# is asserted. LED is also illuminated when the LOO bit is asserted and RSTIN# is deasserted. LED is an active HIGH signal that allows other circuit to drive the BLUE LED.
- EJECT: This is the Handle Switching input. This signal should be debounced by external hardware and must be connected to logic "0" if Hot Swap function is not used. This signal can cause the assertion of ENUM#.

GPIO pins designated for recommended Hot Swap use:

- HEALTHY# (GPIO7): This can be used as the Board Healthy HEALTH# output. This pin has internal weak pull-up. Subsystem software can set the pin to output the desired Board Healthy status to the system and to control the custom logic generated Hot Swap port RSTIN#.

## 24.4 Hot Swap Register control and status

PCI 6254 Hot Swap register is located at E6h.

## 24.5 Avoiding Initially Retry or Initially Not Responding Requirement

The PORTREADY mechanism, which results in Retry for BAR access configuration cycles if subsystem is not yet set up, can be disabled if the XB_MEM input pin is connect to HIGH. In such event, the PCI 6254 hardcoded a fixed cross bridge communication window of 16MB memory space at power up. PCI 6254 will automatically claim such 16M of memory space. This allows the boot up of the Low priority Boot Port to move forward without waiting for the Priority Boot port to program the corresponding Memory BAR registers. When the XB_MEM (PRV_MEM pin in Transparent Mode) pin is "1", the P or S PORT_READY mechanism will NOT be relevant and access to BAR registers will not be retried. Although the default claims 16M, the BAR registers can be changed by EEPROM or software to change the window size

During reset, the PCI 6254 is a Not Responding device. Therefore Designer can use GPIO7, for example, to generate HEALTHY# control by the subsystem to control the LOCAL_PCI_RST# input to the PCI 6254 Hot Swap port.

## 24.6 Device Hiding

PCI 6254 implements Device Hiding to eliminate mid-transaction extractions.

PCI 6254 invokes Device Hiding by hardware upon Hot Swap after RSTIN# becomes inactive and ejector handle is still unlocked.

PCI 6254 will be quiesced by software before Device Hiding is invoked. Current transaction will be completed as early as possible. PCI 6254 will not initiate a transaction as a master and will not respond as a target to IO transactions and will not signal interrupts.

When Device Hiding is invoked, PCI 6254 shall terminate current configuration transaction by signaling Disconnect. Following the completion of the current transaction (which is disconnected), PCI 6254 shall not respond as a target to any subsequent transactions until Device Hiding is canceled.

If PCI 6254 is not participating in a transaction when Device Hiding is invoked, PCI 6254 shall not respond as a target to any subsequent transactions until Device Hiding is canceled.

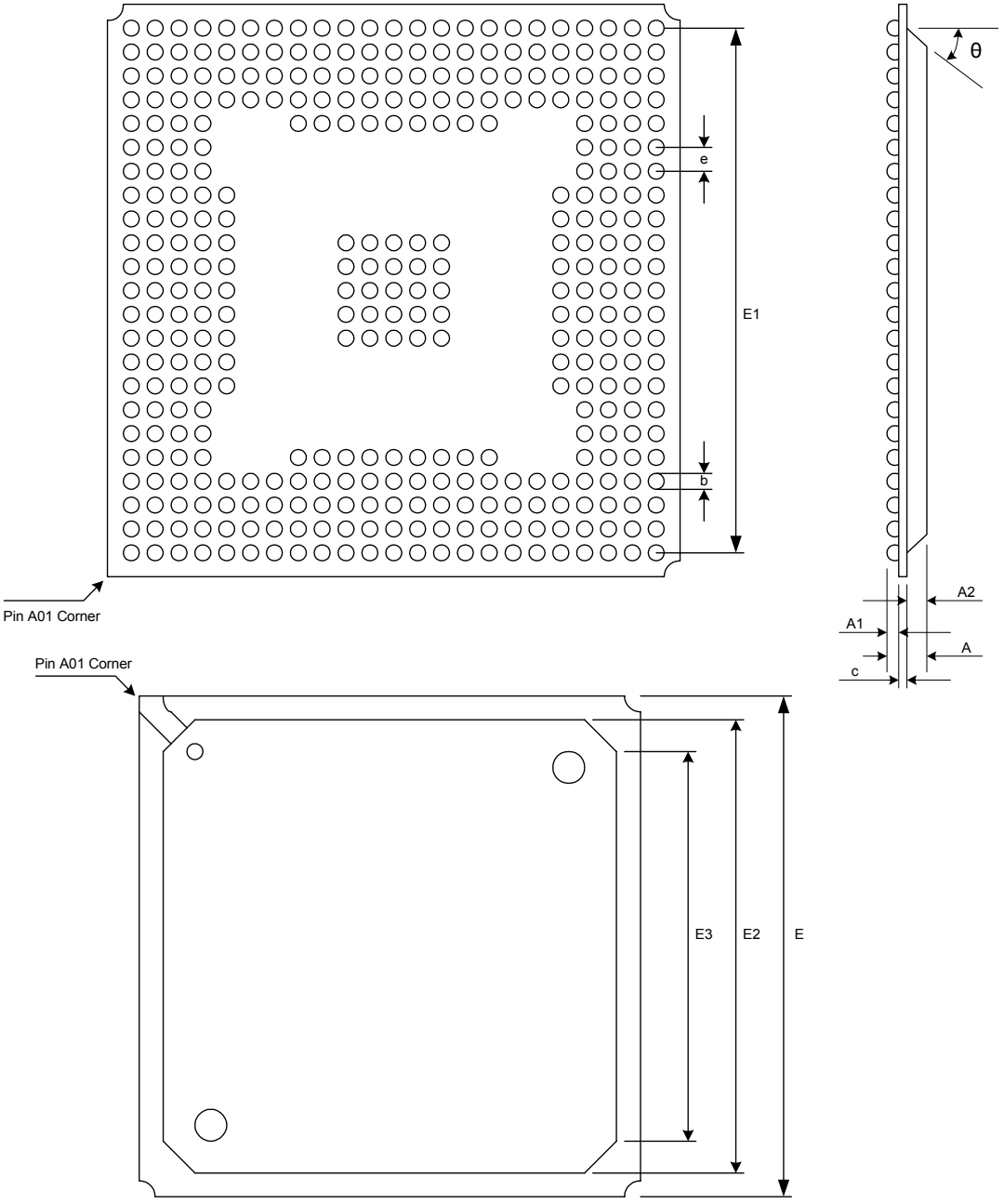
Device Hiding is cancelled when the handle switch is relocked.

## 24.7 Implementing Hot Swap Controller using PCI 6254 GPIO pins

In Transparent Mode, GPIO[15-8], which have weak internal pull-low, can be used for connection to radial signals BD_SEL#. GPIO[7:0], which have weak internal pull-up, can be used for radial signals HEALTHY#. ENUM# can be used to trigger HEALTH inquiry by reading the GPIO ports.

# 25 Package Specifications

This specification outlines the mechanical dimensions for PCI 6254.



The following table lists the package dimensions in millimeters. Tolerance is ±0.05mm unless otherwise specified

| Symbol | Dimension | Min | Nom | Max |
|---|---|---|---|---|
| A | Overall package height | 2.13 | 2.33 | 2.53 |
| A1 | Package standoff height | 0.50 | 0.60 | 0.70 |
| A2 | Encapsulation thickness | 1.12 | 1.17 | 1.22 |
| b | Ball diameter | 0.60 | 0.75 | 0.90 |
| c | Substrate thickness | 0.51 | 0.56 | 0.61 |
| e | Ball pitch | | 1.27 | |
| E | Overall package width | 30.80 | 31.00 | 31.20 |
| E1 | | | 27.94 | |
| E2 | Overall encapsulation width | 28.8 | 29.00 | 29.20 |
| E3 | | | 25.00 | |
| Θ | | | 30° | |

# 26 Electrical Specifications

## 26.1 Maximum Ratings

(Above which the useful life may be impaired. For user guidelines, not tested).

| Parameter | Minimum | Maximum |
|---|---|---|
| Storage Temperature Range | -55 °C | 125 °C |
| Junction Temperature | | 125 °C |
| Supply Voltage, $V_{DD}$ | | 3.9V |
| Maximum Voltage to signal pins | | 5.5V |
| Maximum Power | | 2.0W |

**Note:** Stresses greater than those listed under MAXIMUM RATINGS may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods of time may affect reliability.

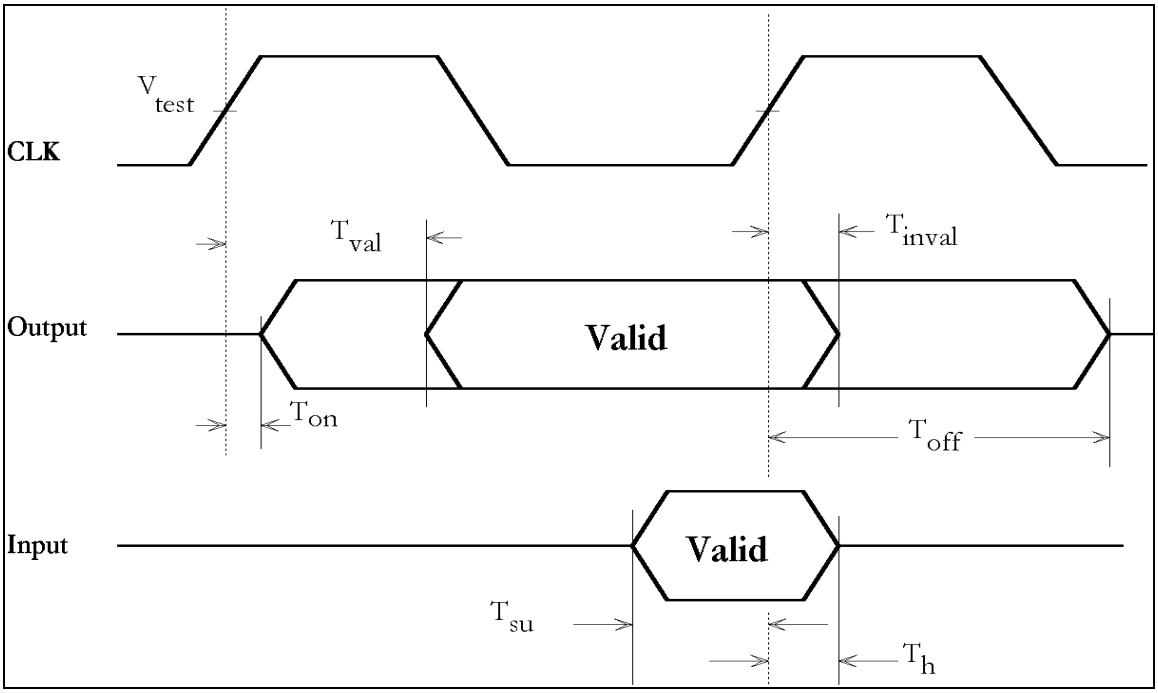
## 26.2 Functional Operating Range

| Parameter | Minimum | Maximum |
|---|---|---|
| Supply Voltage | 3.0 V | 3.6 V |
| Operating ambient temperature | 0 °C | 70 °C |

## 26.3 DC Electrical Characteristics

| Symbol | Parameter | Condition | Min | Max | Unit | Notes |
|---|---|---|---|---|---|---|
| $V_{DD}$ | Supply Voltage | | 3.0 | 3.6 | V | |
| $V_{IO}$ | **PVIO, SVIO** pin Interface I/O Voltage | | 3.0 | 5.5 | V | |
| $V_{ih}$ | Input HIGH Voltage | | 0.5 $V_{DD}$ | $V_{IO}$ | V | |
| $V_{il}$ | Input LOW Voltage | | -0.5 | 0.3 $V_{DD}$ | V | |
| $V_{ol}$ | Output LOW Voltage | $I_{iout}$ = 1500 µA | | 0.1 $V_{DD}$ | V | |
| $V_{oh}$ | Output HIGH Voltage | $I_{iout}$ = -500 µA | 0.9 $V_{DD}$ | | V | |
| $I_{il}$ | Input Leakage Current | 0 < $V_{in}$ < $V_{DD}$ | | ±2 | µA | |
| $C_{in}$ | Input Pin Capacitance | | | 7.0 | pF | |

## 26.4 PCI Signal Timing Specification




### 26.4.1 PCI Signal Timing

| Symbol | Parameter | Minimum | Maximum | Unit |
|---|---|---|---|---|
| $T_{val}$ | CLK to signal valid delay - bused signals | 2 | 8 | ns |
| $T_{val(ptp)}$ | CLK to signal valid delay – point to point | 2 | 8 | ns |
| $T_{on}$ | Float to active delay | 2 | - | ns |
| $T_{off}$ | Active to float delay | - | 14 | ns |
| $T_{su}$ | Input setup time to CLK – bused signals | 3 | - | ns |
| $T_{su(ptp)}$ | Input setup time to CLK – point to point | 5 | - | |
| $T_h$ | Input signal hold time from CLK | 0.5 | - | ns |