Tsi352™ PCI-to-PCI Bridge User Manual

80D6000_MA001_03

September 5, 2009

6024 Silver Creek Valley Road, San Jose, California 95138
Telephone: (800) 345-7015 • (408) 284-8200 • FAX: (408) 284-2775
Printed in U.S.A.

About this Document

This section discusses the following topics:

Scope

The *Tsi352 PCI-to-PCI Bridge User Manual* discusses the features, capabilities, and configuration requirements for the Tsi352. It is designed for hardware and software engineers who are designing system interconnect applications with the device.

Document Conventions

This document uses the following conventions.

Non-differential Signal Notation

Non-differential signals are either active-low or active-high. An active-low signal has an active state of logic 0 (or the lower voltage level), and is denoted by a lowercase "_b". An active-high signal has an active state of logic 1 (or the higher voltage level), and is not denoted by a special character. The following table illustrates the non-differential signal naming convention.

State	Single-line signal	Multi-line signal
Active low	NAME_b	NAME_b[3]
Active high	NAME	NAME[3]

Object Size Notation

- A *byte* is an 8-bit object.
- A *word* is a 16-bit object.
- A *doubleword* (Dword) is a 32-bit object.

Numeric Notation

- Hexadecimal numbers are denoted by the prefix *0x* (for example, 0x04).

- Binary numbers are denoted by the prefix *0b* (for example, 0b010).
- Registers that have multiple iterations are denoted by {x..y} in their names; where *x* is first register and address, and *y* is the last register and address. For example, REG{0..1} indicates there are two versions of the register at different addresses: REG0 and REG1.

Symbols

This symbol indicates a basic design concept or information considered helpful.

This symbol indicates important configuration information or suggestions.

This symbol indicates procedures or operating levels that may result in misuse or damage to the device.

Document Status Information

- Advance – Contains information that is subject to change, and is available once prototypes are released to customers.
- Preliminary – Contains information about a product that is near production-ready, and is revised as required.
- Formal – Contains information about a final, customer-ready product, and is available once the product is released to production.

Revision History

80D6000_MA001_03, Formal, September 2009

This document was rebranded as IDT. It does not include any technical changes.

80D6000_MA001_02, Formal, September 2007

This is the production version of the Tsi352 *User Manual*. The following changes have been made to this document:

- The section “Calculating the Prefetch Count” on page 30 has been added
- The description for the “Cacheline Size Register—Offset 0x0C” on page 131 has been clarified
- The information in “Read Transactions” on page 30 has been clarified

80D6000_MA001_01, Advance, July 2007

This was the first version of the Tsi352 *User Manual*.

Contents

Figures

Tables

1. Functional Overview

This chapter discusses the following topics about the Tsi352:

1.1 Overview

The IDT Tsi352 is a PCI-to-PCI bridge that is fully compliant with *PCI Local Bus Specification, Revision 2.3*. The Tsi352 has sufficient clock and arbitration pins to support four PCI bus master devices directly on its secondary interface.

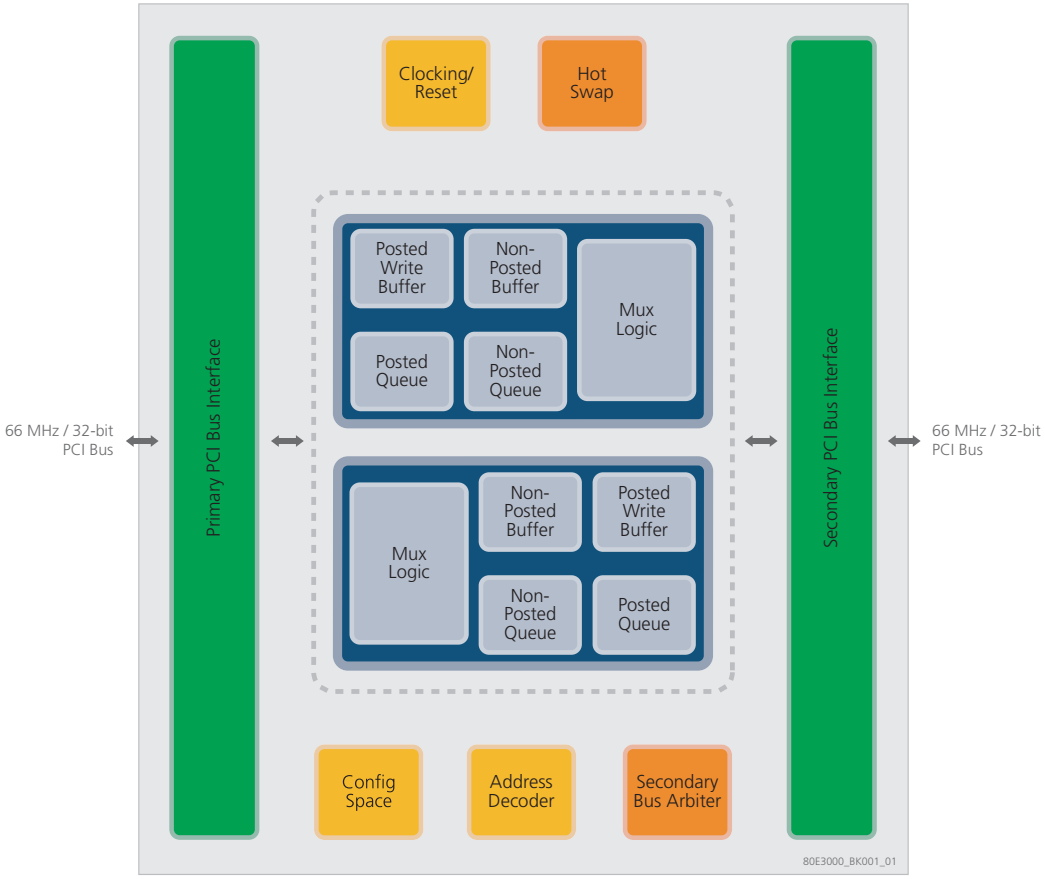
The Tsi352 allows the two PCI buses to operate concurrently. This means that a master and a target on the same PCI bus can communicate while the other PCI bus is busy. This traffic isolation can increase system performance in applications such as multimedia.

The Tsi352 makes it possible to extend a system's load capability limit beyond that of a single PCI bus by allowing designers to add more PCI devices or more PCI option card slots than a single PCI bus can support.

The Tsi352 has two identical PCI Interfaces that each handle PCI transactions for its respective bus, and, depending on the type of transaction, can act as either a bus master or a bus slave. These interfaces transfer data and control information flowing to and from the blocks.

The block diagram for Tsi352 is shown Figure 1 on page 16.

Figure 1: Tsi352 Block Diagram

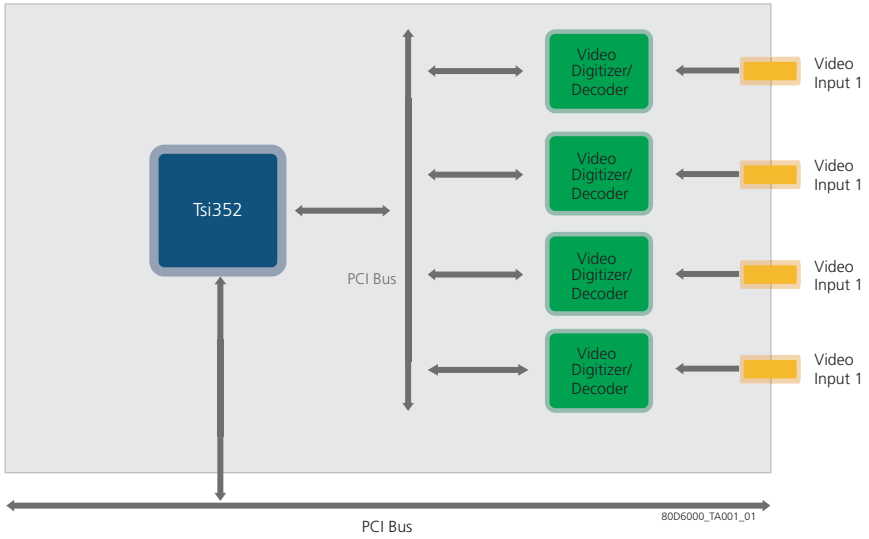


Typical Applications

Option card designers can use Tsi352 to implement multiple-device PCI option cards. Without a PCI-to- PCI bridge, PCI loading rules would limit option cards to one device. The *PCI Local Bus Specification* loading rules limit PCI option cards to a single connection per PCI signal in the option card connector. The Tsi352 overcomes this restriction by providing, on the option card, an independent PCI bus to which up to multiple devices can be attached.

The application diagram, shown in Figure 2, shows how the Tsi352 enables the design of a multi-component option card to expand existing PCI buses.

Figure 2: Application Diagram - Digital Video Recorder



1.2 Features

The following sections describe the features of Tsi352.

1.2.1 PCI features

- 32-bit Primary and Secondary interfaces, operating up to 66 MHz
- All I/O and memory commands
- Type 1 to Type 0 configuration conversion
- Type 1 to Type 1 configuration forwarding
- Type 1 configuration write to special cycle conversion
- Provides an internal arbiter for up to four secondary bus masters and supports an external secondary bus arbiter
- Programmable 2-level priority arbiter
- PCI Clock run support
- Propagates bus locking
- Supports posted write buffers in all directions
- 1Kbyte buffer (organized as 256x4 buffers)
- Enhanced address decoding
- 32-bit I/O address range
- 32-bit memory-mapped I/O address range
- 64-bit prefetchable address range

1.2.2 Compliance

- *PCI-to-PCI Bridge Architecture Specification (Revision 1.1)*
- *PCI Local Bus Specification (Revision 2.3)*
- *PCI bus Power Management Interface Specification (Revision 1.1)*
- *Advanced Configuration Power Interface (ACPI)*
- *PCI Mobile Design Guide (Revision 1.0)*

1.2.3 Physical

- Extended commercial temperature range 0^{o}C to 85^{o}C
- 31.20 mm x 31.20 mm, 160 PQFP package
- Available in RoHS compliant package
- 3.3 V I/O, 5 V tolerant

1.3 Functional Overview

Tsi352 has two PCI interfaces: a primary interface and a secondary interface. Each interface controls the PCI protocol for its respective bus. These interfaces transfer data/control information to and from the Buffer Logic Unit (BLU).

1.3.1 BLU

The BLU consists of a posted write buffer, posted write queue, non-posted buffer, and non-posted queue.

1.3.1.1 Posted Write Buffer

The Tsi352 handles the conventional PCI transactions of Memory Write, and Memory Write and Invalidate as posted transactions.

The posted write buffer is used for temporary storage of data flowing from the primary interface to the secondary interface and from the secondary interface to the primary interface. Each posted buffer has a capacity of 256 bytes. The amount of space assigned to each transaction is dynamic. A single transaction can use sizes ranging from one memory location (4 bytes) to 64-memory location (256 bytes).

When the Tsi352 determines that a memory write transaction must be forwarded across the bridge, it first checks for empty space in the posted write buffer. If space is available, the posted write buffer accepts data until the buffer is full or the transaction is terminated. If there is no space in the posted write buffer, the transaction is terminated with retry.

1.3.1.2 Posted Write Queue

The posted write queue is used to store the control information related to the transaction flowing from the primary interface to secondary interface or from the secondary interface to the primary interface. Each posted write queue has a four entry FIFO, which provides four active posted write transactions in each direction. Data related to each entry is stored in the posted write buffer.

The posted write queue accepts an entry from an external master as long as at least one entry is free and at least one Dword of space is available in the posted write buffer.

1.3.1.3 Non-Posted Buffer

The non-posted buffer is used for storing the data related to delayed transactions. The following list of transactions use the non-posted buffer:

- Memory Read
- Memory Read Line
- Memory Read Multiple
- I/O Read
- I/O Write
- Type-1 Configuration Read
- Type-1 Configuration Write

All the non-posted transactions are processed through the non-posted queues and non-posted buffers. Each non-posted buffer has a storage capacity of up to 256 bytes for storing data related to delayed transactions.

1.3.1.4 Non-Posted Queue

The non-posted queue is used to store the control information related to all the non-posted transactions. Each non-posted queue has a four entry FIFO, which provides four active non-posted transactions in each direction. Data related to each entry is stored in the non-posted buffer.

The non-posted queue accepts an entry from an external master if at least one entry is available. If all four entries are full the Tsi352 retries the external master until an entry becomes available.

1.3.2 Configuration Space

Tsi352 is a PCI-to-PCI bridge, and complies with the *PCI to PCI Bridge Architecture Specification, Revision 1.1*. The Tsi352's configuration space can only be accessed from the primary interface. The Tsi352 uses additional device specific configuration registers to support optional, device specific features.

For more information, refer to "Configuration Transactions" on page 35.

1.3.3 Address Decoding Logic

The Tsi352 is a transparent bridge. In transparent mode, the I/O, memory, and prefetchable memory base and limit define address ranges residing on the secondary bus. All other addresses are assumed to reside on the primary bus. Inverse address decoding determines when to forward the transaction up-stream.

For more information, refer to “Address Decoding” on page 51.

1.3.4 Secondary Bus Arbiter

The Tsi352 has an internal secondary bus arbiter. It provides bus arbitration for up to four additional masters. Each external master is assigned to either high or low priority, or can be masked off.

The internal arbiter provides a two-level arbitration scheme in which arbitration is divided into the following two groups: a high priority group and a low priority group. Each master can be assigned to either a high priority group or a low priority group through the configuration register. Tsi352 also supports an external arbiter on the secondary bus. When using an external arbiter, the Tsi352’s internal arbiter should be disabled by pulling S_CFN_b high.

For more information, refer to “PCI Bus Arbitration” on page 65.

1.3.5 Hot Swap Interface

The Tsi352 is designed with an interface for Hot Swap support. This allows the user to insert or extract the bridge card without powering down the system. During insertion and extraction process, the bridge indicates to system software about the Hot Swap event by driving HS_ENUM_b. It also provides a visual indication to the user through the HS_LED_OUT signal.

1.3.5.1 Multifunction Pins

The Tsi352 has two multifunction pins that can be configured as LOCK_b, CLKRUN_b or Hot Swap HS_ENUM_b and HS_SWITCH_b. The configuration of P_MFUNC and S_MFUNC is controlled by MS0 and MS1 and is shown in Table 1.

Table 1: MS0 and MS1 Configurations

MS0	MS1	P_MFUNC	S_MFUNC	MODE
0	0	HS_ENUM_b	HS_SWITCH_b	Hot swap mode
0	1	P_CLKRUN_b	S_CLKRUN_b	Clock Run mode
1	BPCC	P_LOCK_b	S_LOCK_b	Pericom mode

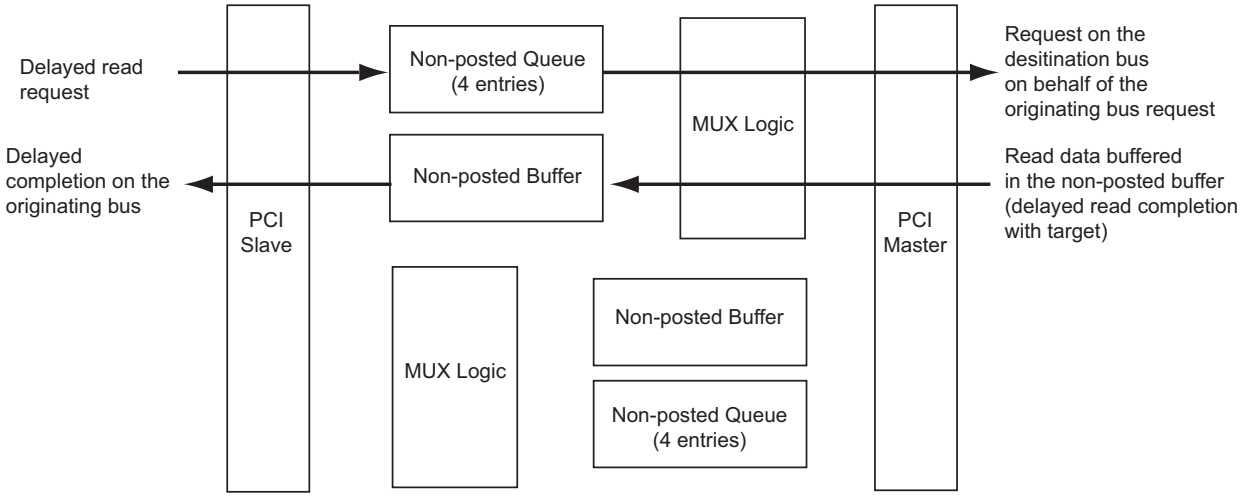
1.4 Data Flow

The following sections describe the data flow through the Tsi352 device.

1.4.1 Memory Read Transactions

The conventional PCI memory read, memory read line, and memory read multiple commands are used to transfer memory read data. Tsi352 completes all memory read transactions as delayed transactions. Figure 3 shows the Tsi352 memory read flow.

Figure 3: Memory Read Flow



The following steps show the memory read flow through the Tsi352:

- The read request from the initiator is posted/entered into the Non-Posted Queue
- The transaction is terminated by signaling target retry to the initiator
- When the target retry is received, the initiator is required to continue to repeat the same read transaction until at least one data transfer is completed or until a master/target abort is received
- The Tsi352 then arbitrates for the destination bus and initiates a read transaction using the exact read address and read command
  - If the Tsi352 receives retry 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.

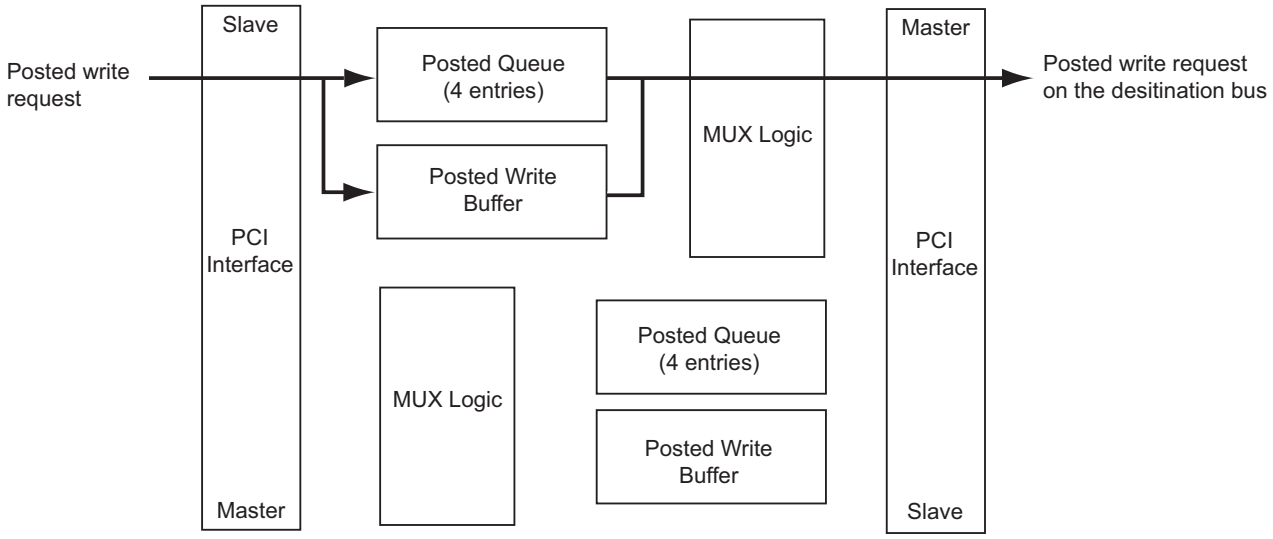
When a memory read transaction targets non-prefetchable address space, the Tsi352 prefetches a single DWORD of data when the memory read command is used. For all other read transactions, the Tsi352 prefetches data according to the prefetch algorithm (see “Read Transactions” on page 30).

- When the transaction is completed on the target bus, the Tsi352 transfers the data to the initiator when the initiator repeats the transaction

1.4.2 Posted Write Transaction Flow

The conventional PCI memory write and memory write and invalidate are posted transactions. Unlike non-posted transactions, these transactions are first completed on the originating bus and then completed on the destination bus. Figure 4 shows the Tsi352 posted write flow.

Figure 4: Memory Write Flow



The following steps show the posted write flow through the Tsi352:

- When Tsi352 determines that a memory write transaction is to be forwarded across the bridge, it first checks for empty space in the posted write buffer:
  - If space is available, Tsi352 accepts data until the buffer is full or the transaction is terminated.
  - If there is no space in the posted write buffer, the transaction is terminated with a retry.
- After buffering data into the posted buffer the Tsi352 arbitrates for the destination bus and writes the data to the destination.

2. PCI Interface

This chapter discusses the following topics about the Tsi352:

2.1 Overview

Tsi352 has two PCI interfaces: a primary interface and a secondary interface. Each interface controls the PCI protocol for its respective bus. These interfaces transfer data/control information to and from the Buffer Logic Unit (BLU). The BLU consists of a posted write buffer, posted write queue, non-posted buffer, and non-posted queue.

The following sections describe the how the Tsi352 handles PCI transactions, transaction forwarding across Tsi352, and transaction termination.

2.2 Transaction Types

This section provides a summary of PCI transactions performed by Tsi352. Table 2 lists the command code and name of each PCI transaction. The Master and Target columns indicate Tsi352 support for each transaction when Tsi352 initiates transactions as a master, on the primary bus and on the secondary bus, and when Tsi352 responds to transactions as a target, on the primary bus and on the secondary bus.

Table 2: Type of Transactions

Type of Transaction	Initiates as a Master		Responds as a Target	
	Primary	Secondary	Primary	Secondary
Interrupt Acknowledge (0000)	No	No	No	No
Special Cycle (0001)	Yes	Yes	No	No
I/O Read (0010)	Yes	Yes	Yes	Yes
I/O Write (0011)	Yes	Yes	Yes	Yes
Reserved (0100)	No	No	No	No
Reserved (0101)	No	No	No	No

Table 2: Type of Transactions

Type of Transaction	Initiates as a Master		Responds as a Target	
	Primary	Secondary	Primary	Secondary
Memory Read (0110)	Yes	Yes	Yes	Yes
Memory Write (0111)	Yes	Yes	Yes	Yes
Reserved (1000)	No	No	No	No
Reserved (1001)	No	No	No	No
Configuration Read (1010)	No	Yes	Yes	No
Configuration Write (1011)	Type-1	Yes	Yes	Type-1
Memory Read Multiple (1100)	Yes	Yes	Yes	Yes
Dual Address Cycle (1101)	Yes	Yes	Yes	Yes
Memory Read Line (1110)	Yes	Yes	Yes	Yes
Memory Write and Invalidate (1111)	Yes	Yes	Yes	Yes

2.2.1 Transaction Types Not Supported

The following PCI commands are not supported by the Tsi352:

- Tsi352 never initiates a PCI transaction with a reserved command code and, as a target, Tsi352 ignores reserved command codes.
- Tsi352 never initiates an interrupt acknowledge transaction and, as a target, Tsi352 ignores interrupt acknowledge transactions. Interrupt acknowledge transactions are expected to reside entirely on the primary PCI bus closest to the host bridge.
- Tsi352 does not respond to special cycle transactions. Tsi352 cannot guarantee delivery of a special cycle transaction to downstream buses because of the broadcast ability of the special cycle command and the inability to control the transaction as a target. To generate special cycle transactions on other PCI buses, either upstream or downstream, a type-1 configuration command must be used.
- Tsi352 does not generate Type 0 configuration transactions on the primary interface, nor does it respond to Type 0 configuration transactions on the secondary PCI interface. The *PCI-to-PCI Bridge Architecture Specification* does not support configuration from the secondary bus.

2.2.2 Address Phase

A standard PCI transaction consists of one or two address phases, followed by one or more data phases. The first address phase is designated by an asserting (falling) edge on the FRAME_b signal. The number of address phases depends on whether the address is 32 bits or 64 bits.

2.2.2.1 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 C/BE_b[3:0]. Tsi352 supports the linear increment address mode only for decoding memory address space, which is indicated when the lower two address bits are equal to 0. If either of the lower two address bits is nonzero, Tsi352 automatically disconnects the transaction after the first data transfer.

2.2.2.2 Dual Address Phase

Dual address transactions are PCI transactions that contain two address phases specifying a 64-bit address. The first address phase is denoted by the asserting edge of FRAME_b. The second address phase always follows on the next clock cycle.

For a 32-bit interface, the first address phase contains the dual address command code on the C/BE_b[3:0] lines, and the low 32 address bits on the AD[31:0] lines. The second address phase consists of the specific memory transaction command code on the C/BE_b[3:0] lines and the high 32 address bits on the AD[31:0]lines. In this way, 64-bit addressing can be supported on 32-bit PCI buses.

The *PCI-to-PCI Bridge Architecture Specification* supports the use of dual address transactions in the prefetchable memory range only. Tsi352 supports dual address transactions in both the upstream and the downstream direction. Tsi352 supports a programmable 64-bit address range in prefetchable memory for downstream forwarding of dual address transactions. Dual address transactions falling outside the prefetchable address range are forwarded upstream, but not downstream. Prefetching and posting are performed in a manner consistent with the guidelines given in this specification for each type of memory transaction in prefetchable memory space.

Tsi352 responds only to dual address transactions that use the following transaction command codes:

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

Use of other transaction codes can result in a master abort.

Any memory transactions addressing the first 4 GB space should use a single address phase; that is, the high 32 bits of a dual address transaction should never be 0.

2.2.3 Device Select (DEVSEL_b) Generation

Tsi352 always performs positive address decoding when accepting transactions on either the primary bus or secondary bus. Tsi352 never subtractively decodes.

2.2.4 Data Phase Transactions

The address phase, or phases, of a PCI transaction are followed by one or more data phases. A data phase is completed when IRDY_b and either TRDY_b or STOP_b are asserted. A transfer of data occurs only when both IRDY_b and TRDY_b are asserted during the same PCI clock cycle. The last data phase of a transaction is indicated when FRAME_b is deasserted and both TRDY_b and IRDY_b are asserted, or when IRDY_b and STOP_b are asserted.

Depending on the command type, Tsi352 can support multiple data phase PCI transactions.

2.2.4.1 Write Transactions

Write transactions are handled as either posted write or delayed write transactions.

Posted Write Transactions

Posted write forwarding is used for memory write and for memory write and invalidate transactions.

When Tsi352 determines that a memory write transaction is to be forwarded across the bridge, Tsi352 asserts DEVSEL_b (with medium timing) and TRDY_b in the next cycle, provided that enough buffer space is available in the posted data queue for the address and at least 8 words of data. This enables Tsi352 to accept write data without obtaining access to the target bus. Tsi352 can accept one Dword of write data every PCI clock cycle; that is, no target wait states are inserted. This write data is stored in internal posted write buffers and is subsequently delivered to the target.

Tsi352 continues to accept write data until one of the following events occurs:

- The initiator terminates the transaction by deasserting FRAME_b and IRDY_b
- An internal write address boundary is reached, such as a cacheline boundary or an aligned 4 KB boundary, depending on the transaction type
- The posted write data buffer is full

When one of the last two events occurs, Tsi352 returns a target disconnect to the requesting initiator on this data phase to terminate the transaction. Once the posted write data moves to the head of the posted data queue, Tsi352 asserts its request on the target bus. This can occur while the Tsi352 is still receiving data on the initiator bus. When the grant for the target bus is received and the target bus is detected in the idle condition, Tsi352 asserts FRAME_b and drives the stored write address out on the target bus. On the following cycle, Tsi352 drives the first Dword of write data 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, Tsi352 can drive one Dword of write data each PCI clock cycle.

Tsi352 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
  - Tsi352 starts another transaction to deliver the remainder of the write data
- The target returns a target abort
  - Tsi352 discards remaining write data

- The master latency timer expires, and Tsi352 no longer has the target bus grant
  - Tsi352 starts another transaction to deliver remaining write data).

Memory Write and Invalidate Transactions

Posted write forwarding is used for memory write and invalidate transactions. Memory write and invalidate transactions guarantee transfer of entire cachelines. If the write buffer fills before an entire cacheline is transferred, Tsi352 disconnects the transaction and converts it to a memory write transaction.

Tsi352 disconnects memory write and invalidate transactions at aligned cacheline boundaries. The cacheline size value in Tsi352 cacheline size register provides the number of Dwords in a cacheline (see “Cacheline Size Register—Offset 0x0C” on page 131). For Tsi352 to generate memory write and invalidate transactions, this cacheline size value must be written to a value that is a power of 2 and less than or equal to 16 (that is, 1, 2, 4, 8, or 16 Dwords).

If the cacheline size does not meet the memory write and invalidate conditions, that is, the value is 0, or is not a power of 2, or is greater than 16 Dwords, Tsi352 treats the memory write and invalidate command as a memory write command. In this case, when Tsi352 forwards the memory write and invalidate transaction to the target bus, it converts the command code to a memory write code and does not observe cacheline boundaries.

If the value in the cacheline size register does meet the memory write and invalidate conditions, that is, the value is a power of 2 less than or equal to 16 Dwords, Tsi352 returns a target disconnect to the initiator either on a cacheline boundary or when the posted write buffer fills. For a cacheline size of 16 Dwords, Tsi352 disconnects a memory write and invalidate transaction on every cacheline boundary. When the cacheline size is 1, 2, 4, or 8 Dwords, Tsi352 accepts another cacheline if at least 8 Dwords of empty space remains in the posted write buffer. If less than 8 Dwords of empty space remains, Tsi352 disconnects on that cacheline boundary. When the memory write and invalidate transaction is disconnected before a cacheline boundary is reached, typically because the posted write buffer fills, the transaction is converted to a memory write transaction.

Delayed Write Transactions

Delayed write forwarding is used for I/O write transactions and for Type 1 configuration write transactions.

A delayed write transaction guarantees that the actual target response is returned back to the initiator without holding the initiating bus in wait states. A delayed write transaction is limited to a single Dword data transfer.

When a write transaction is first detected on the initiator bus, and Tsi352 forwards it as a delayed transaction, Tsi352 claims the access by asserting DEVSEL_b and returns a target retry to the initiator. During the address phase, Tsi352 samples the bus command, address, and address parity one cycle later. After IRDY_b is asserted, Tsi352 also samples the first data Dword, byte enable bits, and data parity. This information is placed into the delayed transaction queue. The transaction is queued only if no other existing delayed transactions have the same address and command, and if the delayed transaction queue is not full. When the delayed write transaction moves to the head of the delayed transaction queue and all ordering constraints with posted data are satisfied, Tsi352 initiates the transaction on the target bus. Tsi352 transfers the write data to the target. If Tsi352 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 Tsi352 is unable to deliver write data after 2^{24} attempts, Tsi352 ceases further write attempts and returns a target abort to the initiator. The delayed transaction is removed from the delayed transaction queue. Tsi352 also asserts P_SERR_b if the primary SERR_b enable bit is set in the command register (see “Primary Command Register—Offset 0x04” on page 124).

When the initiator repeats the same write transaction (that is, the same command, address, byte enable bits, and data), and the completed delayed transaction is at the head of the queue, Tsi352 claims the access by asserting DEVSEL_b and returns TRDY_b to the initiator, to indicate that the write data was transferred. If the initiator requests multiple Dwords, Tsi352 also asserts STOP_b in conjunction with TRDY_b to signal a target disconnect.

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, Tsi352 returns a target retry to the initiator. Tsi352 continues to return a target retry to the initiator until write data is delivered to the target, or until an error condition is encountered. When the write transaction is repeated, Tsi352 does not make a new entry into the delayed transaction queue.

Discard Timer

Tsi352 uses 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 two values, selectable through both the primary and secondary master timeout bits in the bridge control register (see “Bridge Control Register—Offset 0x3C” on page 151). If the initiator does not repeat the delayed write transaction before the discard timer expires, Tsi352 discards the delayed write transaction from the delayed transaction queue. Tsi352 also conditionally asserts P_SERR_b.

Write Transaction Address Boundaries

Tsi352 imposes internal address boundaries when accepting write data. The aligned address boundaries are used to prevent Tsi352 from continuing a transaction over a device address boundary and to provide an upper limit on maximum latency. Tsi352 returns a target disconnect to the initiator when it reaches the aligned address boundaries under the conditions shown in Table 3.

Table 3: Posted Write Address Boundaries

Type of Transaction	Condition	Aligned Address Boundary[a]
Memory write	Disconnect control bit = 0 (“Chip Control Register/Diagnostic Control — Offset 0x40” on page 155)	4 KB aligned address boundary
Memory write	Disconnect control bit = 1 (“Chip Control Register/Diagnostic Control — Offset 0x40” on page 155)	Disconnects at n^{th} cacheline boundary
Memory write and invalidate	Cacheline size =1, 2, 4, 8, 16 (“Cacheline Size Register—Offset 0x0C” on page 131)	4 KB aligned address boundary
Memory write and invalidate	Cacheline size = 1, 2, 4, 8 (“Cacheline Size Register—Offset 0x0C” on page 131)	n^{th} cacheline boundary, where a cacheline boundary is reached and less than 8 free Dwords of posted write buffer space remains
Memory write and invalidate	Cacheline size = 16 (“Cacheline Size Register—Offset 0x0C” on page 131)	16-Dword aligned address boundary

a. n^{th} means the first, second, third, etc., cacheline boundary

Buffering Multiple Write Transactions

The memory write disconnect control bit is located in the “Chip Control Register/Diagnostic Control — Offset 0x40” on page 155 in configuration space. Tsi352 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. If the posted write data buffer fills before the initiator terminates the write transaction, Tsi352 returns a target disconnect to the initiator.

Delayed write transactions are handled as long as at least one open entry in Tsi352 delayed transaction queue exists. Therefore, several posted and delayed write transactions can exist in data buffers at the same time.

Fast Back-to-Back Write Transactions

Tsi352 can recognize fast back-to-back write transactions as a target on the PCI bus. When Tsi352 cannot accept the second transaction because of buffer space limitations, it returns a target retry to the initiator.

Tsi352 does not perform write combining or merging.

2.2.4.2 Read Transactions

Delayed read forwarding is used for all read transactions crossing the Tsi352. Delayed read transactions are handled as either prefetchable or non-prefetchable.

Table 4 shows the read behavior, prefetchable or non-prefetchable, for each type of read operation.

Table 4: Read Behavior

Type of Transaction	Read Behavior
I/O read	Prefetching is never completed
Configuration read	Prefetching is never completed
Memory read	Downstream: Prefetching is used if the address falls within a prefetchable region. If the address falls within a non-prefetchable region, one DWORD is fetched. Upstream: Prefetching used if prefetch disable is off (default)
Memory read line	Prefetching is always used
Memory read multiple	Prefetching is always used

Calculating the Prefetch Count

Tsi352 implements following registers to calculate prefetch count:

- “Cacheline Size Register—Offset 0x0C” on page 131
- “Memory Read Control Register — Offset 0x48” on page 158

The bits in the listed registers are used to calculate the prefetch count using the following equation:

- PRE_DW_CNT = (MPC - CLS) + CLB_CNT
  - Where:
    - PRE_DW_CNT is the prefetch dword count
    - MPC is the Maximum Prefetch Count field in the Memory Read Control Register
    - CLS is the Cache Line Size field in the Cacheline Size Register
    - CLB_CNT is the Cache Line Boundary Count

If the prefetch count has not reached the calculated prefetch count and flow through has been achieved, the Tsi352 keeps prefetching until one of the following occurs:

- The requesting master terminates the transaction
- Read data FIFO becomes full
- The read transaction reaches the 4 K address boundary
- A target disconnects the ongoing transaction

The following rules are true when determining the prefetch count behaviour of the Tsi352:

- Based on the formula, prefetch count is no less than 16 Dword count for any combination of cache line size and maximum prefetch count.
- If the maximum prefetch count is programmed less than Cache Line Size, the Tsi352 prefetches data up to first cache line boundary.
- If buffer is available, the Tsi352 can prefetch more than calculated prefetch count.

Prefetchable Read Transactions

A prefetchable read transaction is a read transaction where Tsi352 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. However, byte enable bits cannot be forwarded for all data phases as is done for the single data phase of the non-prefetchable read transaction. For prefetchable read transactions, Tsi352 forces all byte enable bits to be turned on for all data phases.

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 data that is prefetched depends on the type of transaction. The amount of prefetching can also be affected by the amount of free buffer space available in Tsi352, and by any read address boundaries encountered.

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.

Non-prefetchable Read Transactions

A non-prefetchable read transaction is a read transaction where Tsi352 requests one Dword from the target and disconnects the initiator after delivery of the first Dword of read data. Unlike prefetchable read transactions, Tsi352 forwards the read byte enable information for the data phase.

Non-prefetchable behavior is used for I/O and configuration read transactions, as well as for memory read transactions that fall into non-prefetchable memory space. If prefetching could have side effects, for example, when accessing a FIFO, use non-prefetchable read transactions to those locations. Accordingly, if it is important to retain the value of the byte enable bits during the data phase, use non-prefetchable read transactions. If these locations are mapped in memory space, use the memory read command and map the target into non-prefetchable (memory-mapped I/O) memory space to use non-prefetching behavior.

Read Prefetch Address Boundaries

Tsi352 imposes internal read address boundaries on read prefetching. When a read transaction reaches one of these aligned address boundaries, Tsi352 stops prefetching data, unless the target signals a target disconnect before the read prefetch boundary is reached. When Tsi352 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 remaining prefetched data is discarded.

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

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

Table 5: Prefetch Address Boundaries

Type of Transaction	Address Space	Cacheline Size	Prefetch Aligned Address Boundary
Configuration read	--	--	1 Dword (no prefetch)
I/O read	--	--	1 Dword (no prefetch)
Memory read	Non-prefetchable	--	1 Dword (no prefetch)
Memory read	Prefetchable	CLS ≠ 1, 2, 4, 8, 16, 32, and 64	16-Dword aligned address boundary
Memory read	Prefetchable	CLS = 1, 2, 4, 8, 16, 32, and 64	Cacheline address boundary
Memory read line	--	CLS ≠ 1, 2, 4, 8, 16, 32, and 64	16-Dword aligned address boundary
Memory read line	--	CLS = 1, 2, 4, 8, 16, 32, and 64	Cacheline address boundary
Memory read multiple	--	CLS ≠ 11, 2, 4, 8, 16, 32, and 64	Queue full
Memory read multiple	--	CLS = 1, 2, 4, 8, 16, 32, and 64	Second cacheline boundary

Delayed Read Requests

The Tsi352 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 the Tsi352 accepts a delayed read request, it first samples the read address, read bus command, and address parity. When IRDY_b is asserted, the Tsi352 then samples the byte enable bits for the first data phase. This information is entered into the delayed transaction queue.

The Tsi352 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.

Delayed Read Completion with Target

When the delayed read request reaches the head of the delayed transaction queue, and all previously queued posted write transactions have been delivered, Tsi352 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 non-prefetchable read, Tsi352 drives the captured byte enable bits during the next cycle. If the transaction is a prefetchable read transaction, it drives the captured byte enables for first data phase and drives all byte enable bits to 0 for all remaining data phases.

If Tsi352 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 through normal master termination or target disconnect after at least one data transfer has been completed, Tsi352 does not initiate any further attempts to read more data.

If Tsi352 is unable to obtain read data from the target after 2^{24} attempts, Tsi352 ceases further read attempts and returns a target abort to the initiator. The delayed transaction is removed from the delayed transaction queue. Tsi352 also asserts P_SERR_b if the primary SERR_b enable bit is set in the command register.

Once Tsi352 receives DEVSEL_b and TRDY_b from the target, it transfers the data read to the opposite direction read data queue, pointing toward the opposite interface, before terminating the transaction. For example, read data in response to a downstream read transaction initiated on the primary bus is placed in the upstream read data queue. Tsi352 can accept 1 Dword of read data each PCI clock cycle; that is, no master wait states are inserted. The number of Dwords transferred during a delayed read transaction depends on the conditions given in Table 5 (assuming no disconnect is received from the target).

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, Tsi352 transfers the data to the initiator when the initiator repeats the transaction. For memory read transactions, Tsi352 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. Tsi352 returns a target disconnect along with the transfer of the last Dword of read data to the initiator. If the 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 Tsi352 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 4-KB address boundary is reached, or until the buffer fills, whichever comes first. When the buffer empties, Tsi352 indicates the stalled condition to the initiator by deasserting TRDY_b until more read data is available; otherwise, Tsi352 does not insert any target wait states. When the initiator terminates the transaction, the deassertion of FRAME_b on the initiator bus is forwarded to the target bus. Any remaining read data is discarded.

Tsi352 uses 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 can be set to one of two values, selectable through both the primary and secondary master timeout value bits in the bridge control register. If the initiator does not repeat the read transaction before the discard timer expires Tsi352 discards the read transaction and the read data from its queues. Tsi352 also conditionally asserts P_SERR_b.

Tsi352 has the capability to post multiple delayed read requests, up to a maximum of three 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 posted as it is already contained in the delayed transaction queue.

2.3 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 Tsi352 registers are accessible in configuration space only.

In addition to accepting configuration transactions for initialization of its own configuration space, the Tsi352 also forwards configuration transactions for device initialization in hierarchical PCI systems, as well as for special cycle generation.

To support hierarchical PCI bus systems, two types of configuration transactions are specified: Type 0 and Type 1.

- 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.

Figure 5 and Figure 6 show the address formats for Type 0 and Type 1 configuration transactions.

Figure 5: Type 0 Configuration Transaction Address Format

31 11	10 8	7 2	1	0
Reserved	Function number	Register number	0	0

Figure 6: Type 1 Configuration Transaction Address Format

31 24	23 16	15 11	10 8	7 2	1	0
Reserved	Bus number	Device number	Function number	Register number	0	1

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.

2.3.1 Type 0 Access to Tsi352

Tsi352 configuration space is accessed by a Type 0 configuration transaction on the primary interface. Tsi352 configuration space cannot be accessed from the secondary bus. Tsi352 responds to a Type 0 configuration transaction by asserting P_DEVSEL_b when the following conditions are met during the address phase:

- The bus command is a configuration read or configuration write transaction
- The lower two address bits on P_AD[1:0] must be 00b
- The P_IDSEL signal must be asserted.

The function code is ignored because Tsi352 is a single-function device.

Tsi352 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 Tsi352 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 Tsi352 data buffers; that is, these transactions are completed immediately, regardless of the state of the data buffers.

Tsi352 ignores all Type 0 transactions initiated on the secondary interface.

2.3.2 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, such as the Tsi352, 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.

Tsi352 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. Tsi352 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, Tsi352 generates a Type 0 transaction only on the secondary bus, and never on the primary bus.

Tsi352 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.

2.3.2.1 Address Phase Requirements

Tsi352 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 lower two 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 Tsi352 configuration space.

- The bus command on P_CBE_b[3:0] is a configuration read or configuration write transaction.

When Tsi352 translates the Type 1 transaction to a Type 0 transaction on the secondary interface, it performs the following translations to the address:

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

2.3.2.2 Address and Device Mapping

Tsi352 asserts a unique address line based on the device number. These address lines can 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 6 presents the mapping used by the Tsi352.

Table 6: Device Number to IDSEL S_AD Mapping

Device Number	P_AD[15:11]	Secondary IDSEL S_AD[31:16]	S_AD Bit
0x0	00000	0000 0000 0000 0001	16
0x1	00001	0000 0000 0000 0010	17
0x2	00010	0000 0000 0000 0100	18
0x3	00011	0000 0000 0000 1000	19
0x4	00100	0000 0000 0001 0000	20
0x5	00101	0000 0000 0010 0000	21
0x6	00110	0000 0000 0100 0000	22
0x7	00111	0000 0000 1000 0000	23
0x8	01000	0000 0001 0000 0000	24
0x9	01001	0000 0010 0000 0000	25
0xA	01010	0000 0100 0000 0000	26
0xB	01011	0000 1000 0000 0000	27
0xC	01100	0001 0000 0000 0000	28
0xD	01101	0010 0000 0000 0000	29
0xE	01110	0100 0000 0000 0000	30
0xF	01111	1000 0000 0000 0000	31

Table 6: Device Number to IDSEL S_AD Mapping

Device Number	P_AD[15:11]	Secondary IDSEL S_AD[31:16]	S_AD Bit
0x10-0x1E	10000-11110	0000 0000 0000 0000	--
0x1F	11111	Generate special cycle (P_AD[7:2] = 0x00) 0000 0000 0000 0000 (P_AD[7:2] ≠ 0x00)	--

Tsi352 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.

2.3.3 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.

Tsi352 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.

When Tsi352 detects a Type 1 configuration transaction intended for a PCI bus downstream from the secondary bus, Tsi352 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.

2.3.3.1 Address Phase Requirements

Downstream Type 1-to-Type 1 forwarding occurs when the following conditions are met during the address phase:

- The lower two 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.

Tsi352 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 lower two 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.

Tsi352 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.

2.3.4 Special Cycles

The Type 1 configuration mechanism is used to generate special cycle transactions in hierarchical PCI systems. Special cycle transactions are ignored by a PCI-to-PCI bridge 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.

The Tsi352 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 lower two 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 C/BE_b is a configuration write command

When the Tsi352 initiates the transaction on the target interface, the bus command is changed from a configuration write to a special cycle and the following actions occur:

- 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, Tsi352 responds with TRDY_b to the next attempt of the configuration transaction from the initiator.
- If more than one data transfer is requested, the Tsi352 responds with a target disconnect operation during the first data phase.

2.4 Transaction Termination

This section describes how the Tsi352 returns transaction termination conditions back to the initiator.

The initiator can terminate transactions with one of the following types of terminations:

- Normal termination: Normal termination occurs when the initiator deasserts FRAME_b at the beginning of the last data phase, and deasserts IRDY_b at the end of the last data phase in conjunction with either TRDY_b or STOP_b
- Master abort: A master abort occurs when no target response is detected. When the initiator does not detect a DEVSEL_b from the target within five clock cycles after asserting FRAME_b, the initiator terminates the transaction with a master abort. If FRAME_b is still asserted, the initiator deasserts FRAME_b on the next cycle, and then deasserts IRDY_b on the following cycle. IRDY_b must be asserted in the same cycle in which FRAME_b deasserts. If FRAME_b is already deasserted, IRDY_b 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 terminations:

- Normal termination: Both TRDY_b and DEVSEL_b are asserted in conjunction with the deassertion of FRAME_b and assertion of IRDY_b.
- Target retry: Both STOP_b and DEVSEL_b are asserted, without TRDY_b, during the first data phase. No data transfers occur during the transaction. The transaction must be repeated.
- Target disconnect with data transfer: Both STOP_b and DEVSEL_b are asserted with TRDY_b. Asserting these signals show that this is the last data transfer of the transaction.
- Target disconnect without data transfer: Both STOP_b and DEVSEL_b are asserted, without TRDY_b, after previous data transfers have been made. Asserting these signals indicates that no more data transfers will be made during this transaction.
- Target abort: STOP_b is asserted, without both DEVSEL_b and TRDY_b. Asserting this signal indicates that the target will not be able to complete this transaction. DEVSEL_b must be asserted for at least one cycle during the transaction before target abort is signaled.

2.4.1 Master Termination Initiated by the Tsi352

The Tsi352, as an initiator, uses normal termination if DEVSEL_b is returned by the target within five clock cycles of Tsi352's assertion of FRAME_b on the target bus.

As an initiator, the Tsi352 terminates a transaction when the following conditions are met:

- During a delayed write transaction, a single Dword is delivered.
- During a non-prefetchable read transaction, a single Dword 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 Tsi352 data buffers to the target.
- For a burst transfer, with the exception of memory write and invalidate transactions, the master latency timer expires and Tsi352's bus grant is deasserted.
- The target terminates the transaction with a retry, disconnect, or target abort.

2.4.1.1 Master Latency Timer Expiration

If the Tsi352 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 the Tsi352 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.

2.4.2 Master Abort Received by Tsi352

If the Tsi352 initiates a transaction on the target bus and does not detect DEVSEL_b returned by the target within five clock cycles of the Tsi352's assertion of FRAME_b, the Tsi352 terminates the transaction with a master abort. The Tsi352 sets the received master abort bit in the status register corresponding to the target bus.

For delayed read and write transactions, when the master abort mode bit in the "Bridge Control Register—Offset 0x3C" on page 151 is 0, the Tsi352 returns TRDY_b on the initiator bus and, for read transactions, returns FFFF_FFFFh as data. When the master abort mode bit is 1, the Tsi352 returns target abort on the initiator bus. The Tsi352 also sets the signaled target abort bit in the register corresponding to the initiator bus.

When a master abort is received in response to a posted write transaction, the Tsi352 discards the posted write data and makes no more attempts to deliver the data. The Tsi352 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, the Tsi352 also asserts P_SERR_b. The functionality must be enabled by the SERR_b enable bit in the command register and not be disabled by the device-specific P_SERR_b disable bit for master abort during posted write transactions (that is, master abort mode = 1; SERR_b enable bit = 1; and P_SERR_b disable bit for master aborts = 0.)

When the Tsi352 performs a Type 1 to special cycle translation, a master abort is the expected termination for the special cycle on the target bus. In this case, the master abort received bit is *not* set, and the Type 1 configuration transaction is disconnected after the first data phase.

2.4.3 Target Termination Received by Tsi352

When the Tsi352 initiates a transaction on the target bus and the target responds with DEVSEL_b, the target can end the transaction with one of the following types of termination:

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

The Tsi352 handles these terminations in different ways, depending on the type of transaction being performed.

2.4.4 Delayed Write Target Termination Response

When the Tsi352 initiates a delayed write transaction, the type of target termination received from the target can be passed back to the initiator. Table 7 shows the Tsi352 response to each type of target termination that occurs during a delayed write transaction.

Table 7: Tsi352 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 • Set received target abort bit in target interface status register. • Set signaled target abort bit in initiator interface status register.

The Tsi352 repeats a delayed write transaction until one of the following conditions is met:

- The Tsi352 completes at least one data transfer
- The Tsi352 receives a master abort
- The Tsi352 receives a target abort

- The Tsi352 makes 2^{24} write attempts resulting in a response of target retry
  - After the Tsi352 makes 2^{24} attempts of the same delayed write transaction on the target bus, the Tsi352 asserts P_SERR_b if the primary SERR_b enable bit is set in the command register and the implementation-specific P_SERR_b disable bit for this condition is not set in the P_SERR_b event disable register. The Tsi352 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, the Tsi352 returns a target abort.

2.4.5 Posted Write Target Termination Response

When the Tsi352 initiates a posted write transaction, the target termination cannot be passed back to the initiator. Table 8 shows the Tsi352 response to each type of target termination that occurs during a posted write transaction.

Table 8: Tsi352 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. Assert P_SERR_b if enabled, and set the signaled system error bit in the primary status register.

When a target retry or target disconnect is returned and posted write data associated with that transaction remains in the write buffers, the Tsi352 initiates another write transaction to attempt to deliver the remainder of the write data. In the case of a target retry, the same address is 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, the Tsi352 uses the memory write command to deliver the remainder of the write data because less than a cacheline is transferred in the subsequent write transaction attempt.

After the Tsi352 makes 2^{24} write transaction attempts and fails to deliver all the posted write data associated with that transaction, the Tsi352 asserts P_SERR_b if the primary SERR_b enable bit is set in the command register and the device-specific P_SERR_b disable bit for this condition is not set in the P_SERR_b event disable register. The write data is discarded.

2.4.6 Delayed Read Target Termination Response

When the Tsi352 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 9 shows the Tsi352 response to each type of target termination that occurs during a delayed read transaction.

Table 9: Tsi352 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 non prefetchable, target disconnect on first data phase.
Target retry	Re-initiate 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. Set received target abort bit in the target interface status register. Set signaled target abort bit in the initiator interface status register.

The Tsi352 repeats a delayed read transaction until one of the following conditions is met:

- The Tsi352 completes at least one data transfer.
- The Tsi352 receives a master abort.
- The Tsi352 receives a target abort.
- The Tsi352 makes 2^{24} read attempts resulting in a response of target retry.
  - After the Tsi352 makes 2^{24} attempts of the same delayed read transaction on the target bus, the Tsi352 asserts P_SERR_b if the primary SERR_b enable bit is set in the command register and the implementation-specific P_SERR_b disable bit for this condition is not set in the P_SERR_b event disable register. The Tsi352 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, the Tsi352 returns a target abort.

2.4.7 Target Termination Initiated by the Tsi352

The Tsi352 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.

2.4.7.1 Target Retry

The Tsi352 returns a target retry to the initiator when it cannot accept write data or return read data as a result of internal conditions.

The Tsi352 returns a target retry to an initiator when any of the following conditions are 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 has not progressed to the head of the return queue
  - The delayed transaction queue is full, and the transaction cannot be queued
  - A transaction with the same address and command has been queued
- 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
  - The Tsi352 is currently discarding previously prefetched read data
- For posted write transactions:
  - The posted write data buffer does not have enough space for address and at least 8 Dwords of write data

Delayed 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 it is a write transaction, within the time frame specified by the master timeout value; otherwise, the transaction is discarded from the Tsi352 buffers.

2.4.7.2 Target Disconnect

The Tsi352 returns a target disconnect to an initiator when one of the following conditions is met:

- The Tsi352 hits an internal address boundary
- The Tsi352 cannot accept any more write data
- The Tsi352 has no more read data to deliver

2.4.7.3 Target Abort

The Tsi352 returns a target abort to an initiator when one of the following conditions is met:

- The Tsi352 is returning a target abort from the intended target.
- The Tsi352 is unable to obtain delayed read data from the target or to deliver delayed write data to the target after 2^{24} attempts.

When the Tsi352 returns a target abort to the initiator, it sets the signaled target abort bit in the status register corresponding to the initiator interface.

2.5 Exclusive Access

This section describes the use of the LOCK_b signal to implement exclusive access to a target for transactions that cross the Tsi352

2.5.1 Concurrent Locks

The primary and secondary bus lock mechanisms operate concurrently except when a locked transaction crosses the Tsi352. 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.

2.5.2 Exclusive Access across the Tsi352

For any PCI bus, before acquiring access to the LOCK_b 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_b signal must be deasserted

The initiator leaves the LOCK_b signal deasserted during the address phase (only the first address phase of a dual address transaction) and asserts LOCK_b one clock cycle later. Once a data transfer is completed from the target, the target lock has been achieved.

Locked transactions can cross the Tsi352 only in the downstream direction, from the primary bus to the secondary bus.

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 the Tsi352 detects, on the primary bus, an initial locked transaction intended for a target on the secondary bus, the Tsi352 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 apart of the locked transaction sequence are still posted. Memory read transactions that are apart of the locked transaction sequence are not prefetched.

When the locked delayed read request is queued, the Tsi352 does not queue any more transactions until the locked sequence is finished. The Tsi352 signals a target retry to all transactions initiated subsequent to the locked read transaction that are intended for targets on the other side of the Tsi352. The Tsi352 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, the Tsi352 initiates the transaction as a locked read transaction by deasserting S_LOCK_b on the secondary bus during the first address phase, and by asserting S_LOCK_b one cycle later. If S_LOCK_b is already asserted (used by another initiator), the Tsi352 waits to request access to the secondary bus until S_LOCK_b is sampled deasserted when the secondary bus is idle.

The existing lock on the secondary bus could not have crossed Tsi352 otherwise, the pending queued locked transaction would not have been queued.

When the Tsi352 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, the Tsi352 transfers the read data back to the initiator, and the lock is then also established on the primary bus.

For the Tsi352 to recognize and respond to the initiator, the initiator's subsequent attempts of the read transaction must use the locked transaction sequence (de-assert P_LOCK_b during address phase, and assert P_LOCK_b one cycle later). If the LOCK_b sequence is not used in subsequent attempts, a master timeout condition can result. When a master timeout condition occurs, P_SERR_N is conditionally asserted, the read data and queued read transaction are discarded, and the S_LOCK_b signal is deasserted on the secondary bus.

Once the intended target has been locked, any subsequent locked transactions initiated on the primary bus that are forwarded by the Tsi352 are driven as locked transactions on the secondary bus.

When the Tsi352 receives a target abort or a master abort in response to the delayed locked read transaction, a target abort is returned to the initiator, and no locks are established on either the target or the initiator bus. The Tsi352 resumes forwarding unlocked transactions in both directions.

When the Tsi352 detects, on the secondary bus, a locked delayed transaction request intended for a target on the primary bus, Tsi352 queues and forwards the transaction as an unlocked transaction. The Tsi352 ignores S_LOCK_b for upstream transactions and initiates all upstream transactions as unlocked transactions.

2.5.3 Ending Exclusive Lock

After the lock has been acquired on both the primary and secondary buses, the Tsi352 must maintain the lock on the secondary (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. The Tsi352 does not know whether the current transaction is the last one in a sequence of locked transactions until the initiator deasserts the P_LOCK_b signal at the end of the transaction.

When the last locked transaction is a delayed transaction, the Tsi352 has already completed the transaction on the secondary bus. In this case, as soon as the Tsi352 detects that the initiator has relinquished the P_LOCK_b signal by sampling it in the deasserted state while P_FRAME_b is deasserted, the Tsi352 deasserts the S_LOCK_b signal on the secondary bus as soon as possible.

Because of this behavior, S_LOCK_b can not be deasserted until several cycles after the last locked transaction has been completed on the secondary bus. As soon as the Tsi352 has deasserted S_LOCK_b 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, the Tsi352 deasserts S_LOCK_b on the secondary bus at the end of the transaction because the lock was relinquished at the end of the write transaction on the primary bus.

When the Tsi352 receives a target abort or a master abort in response to a locked delayed transaction, the Tsi352 returns a target abort when the initiator repeats the locked transaction. The initiator must then de-assert P_LOCK_b at the end of the transaction. The Tsi352 sets the appropriate status bits, flagging the abnormal target termination condition. Normal forwarding of unlocked posted and delayed transactions is resumed.

When the Tsi352 receives a target abort or a master abort in response to a locked posted write transaction, the Tsi352 cannot pass back that status to the initiator. The Tsi352 asserts P_SERR_b when a target abort or a master abort is received during a locked posted write transaction, if the SERR_b enable bit is set in the command register. Signal P_SERR_b is asserted for the master abort condition if the master abort mode bit is set in the bridge control register.

2.6 CompactPCI Hot Swap Support

The Tsi352 is Hot Swap friendly silicon that supports all of the Hot Swap capable features, contains support for software control, and integrates circuitry required by the *PICMG CompactPCI Hot-Swap Specification.*

To be Hot Swap capable, the Tsi352 supports the following:

- Compliance with *PCI Local Bus Specification.*
- Tolerance of V_{DD} from early power.
- Asynchronous reset.

- Tolerance of precharge voltage.
- I/O buffers that meet modified V/I requirements.
- Limited I/O terminal voltage at pre-charge voltage.
- Hot Swap control and status programming via extended PCI capabilities linked list.
- Hot Swap terminals: HS_ENUM_b, HS_SWITCH_b, and HS_LED_OUT, CompactPCI hot-swap defines a process for installing and removing PCI boards without adversely affecting a running system. The Tsi352 provides this functionality such that it can be implemented on a board that can be removed and inserted in a hot-swap system.

Tsi352 provides three terminals to support hot-swap when configured to be in hot-swap mode:

- HS_ENUM_b (output) - Indicates to the system that an insertion event occurred or that a removal event is about to occur.
- HS_SWITCH_b (input) - Indicates the state of a board ejector handle.
- HS_LED_OUT (output) - Drives a blue LED to signal insertion- and removal-ready status.

3. Address Decoding

This chapter discusses the following:

3.1 Overview of Address Decoding

The Tsi352 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 Tsi352 configuration space. This chapter describes these address ranges, as well as ISA-mode and VGA-addressing support.

3.2 Address Ranges

The Tsi352 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 PCI bus to the primary PCI bus:

- One 32-bit I/O address range
- One 32-bit memory-mapped I/O (non-prefetchable memory)
- One 64-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.

The Tsi352 uses a flat address space; that is, it does not perform any address translations. The address space has no *gaps* — addresses that are not marked for downstream forwarding are always forwarded upstream.

3.3 Address Forwarding

Tsi352 uses the following mechanisms that are defined in Tsi352 configuration space to specify the I/O address space for downstream and upstream forwarding:

- I/O base and limit address registers ("I/O Base Address Register—Offset 0x1C" on page 137)
- The ISA enable bit ("Bridge Control Register—Offset 0x3C" on page 151)
- The legacy ISA I/O enable bit ("Miscellaneous Control Register — Offset 0x58" on page 161)

- The VGA mode bit (“Bridge Control Register—Offset 0x3C” on page 151)
- The VGA snoop bit (“Bridge Control Register—Offset 0x3C” on page 151)

This section provides information on the I/O address registers and ISA mode. “VGA Mode” on page 59 provides information on the VGA modes.

To enable downstream forwarding of I/O transactions, the I/O enable bit must be set in the command register in Tsi352 configuration space (see “Primary Command Register—Offset 0x04” on page 124). 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, the Tsi352 ignores all I/O and memory transactions initiated on the secondary bus. Setting the master enable bit also allows upstream forwarding of memory transactions.

If any Tsi352 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 Tsi352 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.

3.3.1 Base and Limit Address Registers

The Tsi352 uses one set of I/O base and limit address registers in configuration space that define an I/O address range for downstream forwarding (see “I/O Base Address Register—Offset 0x1C” on page 137). The Tsi352 supports 32-bit I/O addressing, which allows I/O addresses downstream of the Tsi352 to be mapped anywhere in a 4 GB 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.

3.3.1.1 Turning off the I/O Range

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 Tsi352 I/O range has a minimum granularity of 4 kB and is aligned on a 4 kB boundary. The maximum I/O range is 4 GB in size.

3.3.1.2 I/O Base Register

The I/O base register consists of an 8-bit field at configuration address 0x1C, and a 16-bit field at address 0x30. 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 0x1 to indicate that the Tsi352 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 4 kB boundary.

The 16 bits contained in the I/O base upper 16 bits register at configuration offset 0x30 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 0x0000 0000.

3.3.1.3 I/O Limit Register

The I/O limit register consists of an 8-bit field at configuration offset 0x1D and a 16-bit field at offset 0x32. 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 0x0000 0FFF.

The initial states of the I/O base and I/O limit address registers define an I/O range of 0x0000 0000 to 0x0000 0FFF, which is the bottom 4 KB of I/O space. Write these registers with their appropriate values before either setting the I/O enable bit or the master enable bit in the command register in configuration space.

3.3.2 ISA Mode

The Tsi352 supports ISA mode by providing an ISA enable bit in the bridge control register in configuration space (“Bridge Control Register—Offset 0x3C” on page 151). ISA mode modifies the response of the Tsi352 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 the Tsi352 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 64 kB of I/O space (address bits [31:16] are 0x0000).

When the ISA Enable bit is set, the following conditions are true:

- The Tsi352 does not forward downstream any I/O transactions addressing the top 768 bytes of each aligned 1 kB block. Only those transactions addressing the bottom 256 bytes of an aligned 1 kB block inside the base and limit I/O address range are forwarded downstream. Transactions above the 64 kB I/O address boundary are forwarded as defined by the address range defined by the I/O base and limit registers.
- The Tsi352 forwards upstream those I/O transactions addressing the top 768 bytes of each aligned 1 kB block within the first 64 kB of I/O space. The master enable bit in the command configuration register must also be set to enable upstream forwarding (see “Primary Command Register—Offset 0x04” on page 124). 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 the Tsi352 can have I/O space mapped into the first 256 bytes of each 1 kB chunk below the 64 kB boundary, or anywhere in I/O space above the 64 kB boundary.

3.4 Memory Address Decoding

The Tsi352 has the following methods 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

To enable downstream forwarding of memory transactions, the memory enable bit must be set in the command register in the Tsi352 configuration space (see “Primary Command Register—Offset 0x04” on page 124). To enable upstream forwarding of memory transactions, the master enable bit must be set in the command register (see “Primary Command Register—Offset 0x04” on page 124). Setting the master enable bit also allows upstream forwarding of I/O transactions.

If any Tsi352 configuration state affecting memory transaction forwarding is changed by configuration write operation on the primary bus at the same time that memory transactions are ongoing on the secondary bus, the Tsi352 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.

3.4.1 Memory-Mapped I/O Base and Limit Address Registers

Memory-mapped I/O is also referred to as non-prefetchable memory. Memory addresses that cannot automatically be prefetched but that can conditionally prefetch based on command type should be mapped into this space. Read transactions to non-prefetchable space can exhibit side effects; this space can have non-memory-like behavior. The Tsi352 prefetches in this space only if the memory read line or memory read multiple commands are used; transactions using the memory read command are limited to a single data transfer.

The memory-mapped I/O base address and memory-mapped I/O limit address registers define an address range that the Tsi352 uses to determine when to forward memory commands. The Tsi352 forwards a memory transaction from the primary to the secondary interface if the transaction address falls within the memory-mapped I/O address range. The Tsi352 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 1 MB. The maximum memory-mapped I/O address range is 4 GB.

The memory-mapped I/O address range is defined by a 16-bit memory-mapped I/O base address register at configuration offset 0x20 and by a 16-bit memory-mapped I/O limit address register at offset 0x22. 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 0x0 0000, which results in a natural alignment to a 1 MB boundary. The low 20 bits of the memory-mapped I/O limit address are assumed to be 0xF FFFF, which results in an alignment to the top of a 1 MB block.

3.4.1.1 Turning off Memory-Mapped I/O

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.

The initial state of the memory-mapped I/O base address register is 0x0000 0000.The initial state of the memory-mapped I/O limit address register is 0x000F FFFF.

The initial state of these registers define a memory-mapped I/O range at the bottom 1 MB block of memory. 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.

3.4.2 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. The Tsi352 prefetches for all types of memory read commands in this address space.

The prefetchable memory base address and prefetchable memory limit address registers define an address range that the Tsi352 uses to determine when to forward memory commands. The Tsi352 forwards a memory transaction from the primary to the secondary interface if the transaction address falls within the prefetchable memory address range. The Tsi352 ignores memory transactions initiated on the secondary interface that fall into this address range. The Tsi352 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).

The prefetchable memory range supports 64-bit addressing and provides additional registers to define the upper 32 bits of the memory address range, the 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 handled 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. “Prefetchable Memory 64-Bit Addressing Registers” on page 58 describes 64-bit addressing support.

The prefetchable memory address range has a granularity and alignment of 1 MB. The maximum memory address range is 4 GB when 32-bit addressing is used, and above 4 GB 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 0x24 and by a 16-bit prefetchable memory limit address register at offset 0x28. 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 0x0 0000, which results in a natural alignment to a 1 MB boundary. The low 20 bits of the prefetchable memory limit address are assumed to be 0xF FFFF, which results in an alignment to the top of a 1 MB block.

The initial state of the base address register and the memory limit address register define a prefetchable memory range at the bottom 1 MB block of memory. 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.

3.4.3 Prefetchable Memory 64-Bit Addressing Registers

The Tsi352 supports 64-bit memory address decoding for forwarding of dual address memory transactions. The dual address cycle is used to support 64-bit addressing. The first address phase of a dual address transaction contains the lower 32 address bits, and the second address phase contains the upper 32 address bits. During a dual address cycle transaction, the upper 32 bits must never be 0 (use the single address cycle commands for transactions addressing the first 4 GB of memory space).

The Tsi352 uses the prefetchable memory base address upper 32 bits register and the prefetchable memory limit address upper 32 bits register to define a prefetchable memory address range greater than 4 GB.

The prefetchable address space can then be defined in the following ways:

- Residing entirely in the first 4 GB of memory
- Residing entirely above the first 4 GB of memory
- Crossing the first 4 GB memory boundary

3.4.3.1 Residing Entirely in the First 4 GB of Memory

If the prefetchable memory space on the secondary interface resides entirely in the first 4 GB of memory, both upper 32 bits registers must be set to 0. The Tsi352 ignores all dual address cycle transactions initiated on the primary interface and forwards all dual address transactions initiated on the secondary interface upstream.

3.4.3.2 Residing Entirely Above the First 4 GB of Memory

If the secondary interface prefetchable memory space resides entirely above the first 4 GB of memory, both the prefetchable memory base address upper 32 bits register and the prefetchable memory limit address upper 32 bits register must be initialized to nonzero values. The Tsi352 ignores all single address memory transactions initiated on the primary interface and forwards all single address memory transactions initiated on the secondary interface upstream (unless they fall within the memory-mapped I/O or VGA memory range).

A dual address memory transaction is forwarded downstream from the primary interface if it falls within the address range defined by the prefetchable memory base address, prefetchable memory base address upper 32 bits, prefetchable memory limit address, and prefetchable memory limit address upper 32 bits registers. If the dual address transaction initiated on the secondary interface falls outside this address range, it is forwarded upstream to the primary interface.

The Tsi352 does not respond to a dual address transaction initiated on the primary interface that falls outside this address range, or to a dual address transaction initiated on the secondary interface that falls within the address range.

3.4.3.3 Crosses the First 4 GB of Memory

If the secondary interface prefetchable memory space straddles the first 4 GB address boundary, the prefetchable memory base address upper 32 bits register is set to 0, while the prefetchable memory limit address upper 32 bits register is initialized to a nonzero value. Single address cycle memory transactions are compared to the prefetchable memory base address register only.

A transaction initiated on the primary interface is forwarded downstream if the address is greater than or equal to the base address. A transaction initiated on the secondary interface is forwarded upstream if the address is less than the base address.

Dual address transactions are compared to the prefetchable memory limit address and the prefetchable memory limit address upper 32 bits registers. If the address of the dual address transaction is less than or equal to the limit, the transaction is forwarded downstream from the primary interface and is ignored on the secondary interface. If the address of the dual address transaction is greater than this limit, the transaction is ignored on the primary interface and is forwarded upstream from the secondary interface.

The prefetchable memory base address upper 32 bits register is located at configuration Dword offset 0x28, and the prefetchable memory limit address upper 32 bits register is located at configuration Dword offset 0x2C. Both registers are reset to 0.

3.5 VGA Support

The Tsi352 has the following modes for VGA support:

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

3.5.1 VGA Mode

When a VGA-compatible device exists downstream from the Tsi352, set the VGA mode bit in the bridge control register in configuration space to enable VGA mode. When Tsi352 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 Tsi352 base and limit address registers. The Tsi352 ignores transactions initiated on the secondary interface addressing these locations.

The VGA frame buffer consists of the following memory address range: 0x000A 0000—0x000B FFFF

Read transactions to frame buffer memory are handled as non-prefetchable. The Tsi352 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:

- 0x3B0–0x3BB
- 0x3C0–0x3DF

These I/O addresses are aliased every 1 kB throughout the first 64 kB 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 zero.

VGA BIOS addresses starting at 0xC0000 are not decoded in VGA mode.

3.5.2 VGA Snoop Mode

The Tsi352 provides VGA snoop mode, allowing for VGA palette write transactions to be forwarded downstream. This mode is used when a graphics device downstream from the Tsi352 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 (see “Primary Command Register—Offset 0x04” on page 124).

The Tsi352 claims VGA palette write transactions by asserting DEVSEL_b in VGA snoop mode.

When the VGA snoop bit is set, the Tsi352 forwards downstream transactions with the following I/O addresses:

- 0xC6
- 0x3C8
- 0x3C9

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 1 kB throughout the first 64 kB of I/O space.

If both the VGA mode bit and the VGA snoop bit are set, the Tsi352 behaves in the same way as if only the VGA mode bit were set.

4. Transaction Ordering

This chapter discusses the following topics about the Tsi352:

4.1 Overview of Transaction Ordering

To maintain data coherency and consistency, the Tsi352 complies with the ordering rules described in the *PCI Local Bus Specification, Revision 2.3*, for transactions crossing the bridge.

This chapter describes the ordering rules that control transaction forwarding across Tsi352. For more information on transaction ordering, see the *PCI Local Bus Specification, Revision 2.3*.

4.2 Transaction Ordering Rules

Ordering relationships are established for the following classes of transactions crossing the Tsi352:

- 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
  - 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 Tsi352 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 Tsi352 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 bus to the initiator bus.

4.2.1 Combining or Merging Write Transactions

The Tsi352 does not combine or merge write transactions The following rules are used in regard to combining and merging:

- The Tsi352 does not combine separate write transactions into a single write transaction—this optimization is best implemented in the originating master.
- The Tsi352 does not merge bytes on separate masked write transactions to the same Dword address—this optimization is also best implemented in the originating master.
- The Tsi352 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.

4.3 General Ordering Guidelines

Independent transactions on the primary and secondary buses have a relationship only when those transactions cross the Tsi352.

The following general ordering guidelines govern transactions crossing the Tsi352:

- 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 a target retry.
- Requests terminated with a 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 a 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. Tsi352 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, non-posted transaction as a master. This is true of the Tsi352 and must be true of other bus agents; otherwise, a deadlock can occur.
- The Tsi352 accepts posted write transactions, regardless of the state of completion of any delayed transactions being forwarded across the Tsi352.

4.3.1 Ordering Rules

Table 10 shows the ordering relationships of all the transactions and refers by number to the ordering rules that follow.

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 select whether the transactions pass each other.

The entries without ordering rule references reflect the Tsi352's specific implementation.

Table 10: Summary of Transaction Ordering

Bus Operation	Can Row Pass Column?				
	Posted Write	Delayed Read Request	Delayed Write Request	Delayed Read Completion	Delayed Write Completion
Posted Write	No Ordering Rule: 1	Yes Ordering Rule: 5	Yes Ordering Rule: 5	Yes Ordering Rule: 5	Yes Ordering Rule: 5
Delayed Read Request	No Ordering Rule: 2	No	No	Yes	Yes
Delayed Write Request	No[4] Ordering Rule: 4	No	No	Yes	Yes
Delayed Read Completion	No[3] Ordering Rule: 3	Yes	Yes	No	No
Delayed Write Completion	Yes	Yes	Yes	No	No

The following ordering rules describe the transaction relationships. Each ordering rule is followed by an explanation. These ordering rules apply to posted write transactions, delayed write and read requests, and delayed write and read completion transactions crossing Tsi352 in the same direction. Note that delayed completion transactions cross the Tsi352 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 Tsi352 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 can 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.

5. PCI Bus Arbitration

This chapter discusses the following:

5.1 Overview of PCI Bus Arbitration

Tsi352 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 Tsi352, typically on the motherboard. For the secondary PCI bus, Tsi352 uses an internal arbiter.

This chapter describes primary and secondary bus arbitration.

5.2 Primary PCI Bus Arbitration

The Tsi352 uses a request output pin, P_REQ_b, and a grant input pin, P_GNT_b, for primary PCI bus arbitration. The Tsi352 asserts P_REQ_b 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, the Tsi352 keeps P_REQ_b asserted. However, if a target retry, target disconnect, or a target abort is received in response to a transaction initiated by the Tsi352 on the primary PCI bus, the Tsi352 de-asserts P_REQ_b for two PCI clock cycles.

5.2.1 Transactions

When P_GNT_b is asserted low by the primary bus arbiter after the Tsi352 has asserted P_REQ_b, the Tsi352 initiates a transaction on the primary bus during the next PCI clock cycle. If P_GNT_b is asserted to the Tsi352 and the Tsi352's P_REQ_b is not asserted, the Tsi352 parks P_AD, P_CBE_b, and P_PAR by driving them to valid logic levels. When the primary bus is parked at Tsi352 and the Tsi352 has a transaction to initiate on the primary bus, the Tsi352 starts the transaction (by asserting FRAME_b) if P_GNT_b was asserted during the previous cycle.

5.2.1.1 Posted Writes

For posted write transactions (see “Posted Write Transactions” on page 26), P_REQ_b is asserted a few cycles after S_DEVSEL_b is asserted.

5.2.1.2 Delayed Reads and Writes

For delayed read and write requests, P_REQ_b is not asserted until the transaction request has been completely queued in the delayed transaction queue (target retry has been returned to the initiator) and is at the head of the delayed transaction queue (see “Delayed Write Transactions” on page 28).

5.3 Secondary PCI Bus Arbitration

The Tsi352 uses an internal secondary PCI bus arbiter. This arbiter supports four external masters in addition to the Tsi352.

5.3.1 Secondary Bus Arbitration Using the Internal Arbiter

The Tsi352 has four secondary bus request input pins, S_REQ_b[3:0], and four secondary bus output grant pins, S_GNT_b[3:0], to support external secondary bus masters. The Tsi352 specific secondary bus request and grant signals are connected internally to the arbiter and are not brought out to external pins when S_CFN_b is low.

Tieing S_CFN_b low on the board enables the Tsi352 internal arbiter.

5.3.1.1 Arbiter Priorities

The secondary arbiter supports a programmable 2-level rotating algorithm. Two groups of masters are assigned, 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 rotates 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 bus master, including the Tsi352, can be configured to be in either the low priority group or the high priority group by setting the corresponding priority bit in the arbiter control register in device-specific configuration space. Each master has a corresponding bit. If the bit is set to 1, the master is assigned to the high priority group. If the bit is set to 0, the master is assigned to the low priority group. If all the masters are assigned to one group, the algorithm defaults to a straight rotating priority among all the masters. After reset, all external masters are assigned to the low priority group, and Tsi352 is assigned to the high priority group. The Tsi352 receives highest priority on the target bus every other transaction, and priority rotates evenly among the other masters.

5.3.1.2 Bus Contention

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 de-asserts another. It de-asserts 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_b or S_IRDY_b is asserted, the arbiter can de-assert one grant and assert another grant during the same PCI clock cycle.

5.3.2 Secondary Bus Arbitration Using the External Arbiter

The internal arbiter is disabled when the secondary bus central function control pin, S_CFN_b, is pulled high. An external arbiter must then be used.

When S_CFN_b is tied high, the Tsi352 reconfigures two pins to be external request and grant pins. The S_GNT_b[0] pin is reconfigured to be the Tsi352's external request pin because it is an output. The S_REQ_b[0] pin is reconfigured to be the external grant pin because it is an input. When an external arbiter is used, Tsi352 uses the S_GNT_b[0] pin to request the secondary bus. When the reconfigured S_REQ_b[0] pin is asserted low after Tsi352 has asserted S_GNT_b[0], the Tsi352 initiates a transaction on the secondary bus one cycle later. If S_REQ_b[0] is asserted and the Tsi352 has not asserted S_GNT_b[0], the Tsi352 parks the S_AD, S_CBE_b, and S_PAR pins by driving them to valid logic levels.

The unused secondary bus grant outputs, S_GNT_b[3:1], are driven high. Unused secondary bus request inputs, S_REQ_b[3:1], should be pulled high.

5.4 Bus Parking

Bus parking refers to driving the AD, C/BE_b, and PAR lines to a known value while the bus is idle. In general, the device implementing the bus arbiter is responsible for parking the bus or assigning another device to park the bus. A device parks the bus when the bus is idle, its bus grant is asserted, and the device’s request is not asserted. The AD and C/BE_b signals should be driven first, with the PAR signal driven one cycle later.

The Tsi352 parks the primary bus only when P_GNT_b is asserted, P_REQ_b is de-asserted, and the primary PCI bus is idle. When P_GNT_b is de-asserted, Tsi352 tristates the P_AD, P_CBE_b, and P_PAR signals on the next PCI clock cycle. If the Tsi352 is parking the primary PCI bus and needs to initiate a transaction on that bus, then the Tsi352 can start the transaction on the next PCI clock cycle by asserting P_FRAME_b if P_GNT_b is still asserted.

The internal secondary bus arbiter always parks the bus at the last master that used the PCI bus. That is, the Tsi352 keeps the secondary bus grant asserted to a specific master until a new secondary bus request comes along. After reset, the Tsi352 parks the secondary bus at itself until transactions start occurring on the secondary bus.

6. Error Handling

This chapter discusses the following topics about Tsi352's error handling capabilities:

6.1 Overview

The Tsi352 checks, forwards, and generates parity on both the primary and secondary interfaces. To maintain transparency, the Tsi352 always forwards the existing parity condition on one bus to the other bus, along with address and data. The Tsi352 always attempts to be transparent when reporting errors, but this is not always possible, given the presence of posted data and delayed transactions.

The Tsi352 has the following features to support error reporting on the PCI bus:

- PERR_b and SERR_b signals on both the primary and secondary interfaces (see "Signal Descriptions" on page 85)
- Primary status register and secondary status register (see "Primary Status Register—Offset 0x04" on page 127 and "Secondary Status Register—Offset 0x1C" on page 139)
- The device-specific P_SERR_b event disable register (see "P_SERR_b Event Enable Register—Offset 0x64" on page 163)
- The device-specific P_SERR_b status register (see "P_SERR_b Status Register — Offset 0x68" on page 166)

This chapter provides information about how the Tsi352 handles errors. It also describes error status reporting and error operation disabling.

6.2 Address Parity Errors

The Tsi352 checks address parity for all transactions on both buses, for all address and all bus commands.

When the Tsi352 detects an address parity error on the primary interface, the following events occur:

- If the parity error response bit is set in the “Primary Command Register—Offset 0x04” on page 124, the Tsi352 does not claim the transaction with P_DEVSEL_b; this can allow the transaction to terminate in a master abort. If the parity error response bit is not set, the Tsi352 proceeds normally and accepts the transaction if it is directed to or across the Tsi352.
- The Tsi352 sets the detected parity error bit in the “Primary Status Register—Offset 0x04” on page 127.
- The Tsi352 asserts P_SERR_b and sets the signaled system error bit in the status register, if both of the following conditions are met:
  - The SERR_b enable bit is set in the command register.
  - The parity error response bit is set in the command register

When the Tsi352 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—Offset 0x3C” on page 151, the Tsi352 does not claim the transaction with S_DEVSEL_b; this can allow the transaction to terminate in a master abort. If the parity error response bit is not set, Tsi352 proceeds normally and accepts the transaction if it is directed to or across Tsi352.
- The Tsi352 sets the detected parity error bit in the secondary status register.
- The Tsi352 asserts P_SERR_b and sets the signaled system error bit in the “Secondary Status Register—Offset 0x1C” on page 139, if both of the following conditions are met:
  - The SERR_b enable bit is set in the command register.
  - The parity error response bit is set in the “Bridge Control Register—Offset 0x3C” on page 151.

6.3 Data Parity Errors

When forwarding transactions, the Tsi352 attempts to pass the data parity condition from one interface to the other unchanged 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 Tsi352.

6.3.1 Configuration Write Transactions to the Tsi352 Configuration Space

When the Tsi352 detects a data parity error during a Type 0 configuration write transaction to the Tsi352 configuration space, the following events occur:

- If the parity error response bit is set in the “Primary Command Register—Offset 0x04” on page 124, the Tsi352 asserts P_TRDY_b and writes the data to the configuration register. The Tsi352 also asserts P_PERR_b.
- If the parity error response bit is not set, the Tsi352 does not assert P_PERR_b.
- The Tsi352 sets the detected parity error bit in the status register, regardless of the state of the parity error response bit.

6.3.2 Read Transactions

When the Tsi352 detects a parity error during a read transaction, the target drives data and data parity, and the initiator checks parity and conditionally asserts PERR_b.

6.3.2.1 Downstream Transaction

For downstream transactions, when the Tsi352 detects a read data parity error on the secondary bus, the following events occur:

- The Tsi352 asserts S_PERR_b two cycles following the data transfer, if the secondary interface parity error response bit is set in the “Bridge Control Register—Offset 0x3C” on page 151.
- The Tsi352 sets the detected parity error bit in the “Secondary Status Register—Offset 0x1C” on page 139.
- The Tsi352 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—Offset 0x3C” on page 151.
- The Tsi352 forwards the bad parity with the data back to the initiator on the primary bus.

Prefetched Reads

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. The Tsi352 completes the transaction normally.

6.3.2.2 Upstream Transaction

For upstream transactions, when the Tsi352 detects a read data parity error on the primary bus, the following events occur:

- The Tsi352 asserts P_PERR_b two cycles following the data transfer, if the primary interface parity error response bit is set in the “Primary Command Register—Offset 0x04” on page 124
- The Tsi352 sets the detected parity error bit in the “Primary Status Register—Offset 0x04” on page 127
- The Tsi352 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.
- The Tsi352 forwards the bad parity with the data back to the initiator on the secondary bus.

Prefetched Read

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. The Tsi352 completes the transaction normally.

6.3.2.3 PERR_b

The Tsi352 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_b two cycles after the data transfer occurs. It is assumed that the initiator takes responsibility for handling a parity error condition; therefore, when the Tsi352 detects PERR_b asserted while returning read data to the initiator, the Tsi352 does not take any further action and completes the transaction normally.

6.3.3 Delayed Write Transactions

When the Tsi352 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_b.

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 the Tsi352 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.

6.3.3.1 Parity Error on Initial Delayed Write Request

When the Tsi352 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, the Tsi352 asserts TRDY_b to the initiator and the transaction is not queued. If multiple data phases are requested, STOP_b is also asserted to cause a target disconnect. Two cycles after the data transfer, the Tsi352 also asserts PERR_b.

  If the parity error response bit is not set, Tsi352 returns a target retry and queues the transaction as usual. The PERR_b signal is not asserted. In this case, the initiator repeats the transaction.

- The Tsi352 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.

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 attempts of the write transaction can not match the original queued delayed write information contained in the delayed transaction queue. In this case, a master timeout condition can occur, possibly resulting in a system error (P_SERR_b asserted).

Downstream Transactions

For downstream transactions, when the Tsi352 is delivering data to the target on the secondary bus and S_PERR_b is asserted by the target, the following events occur:

- The Tsi352 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—Offset 0x3C" on page 151.
- The Tsi352 captures the parity error condition to forward it back to the initiator on the primary bus.

Upstream Transactions

For upstream transactions, when the Tsi352 is delivering data to the target on the primary bus and P_PERR_b is asserted by the target, the following events occur:

- The Tsi352 sets the primary interface data parity detected bit in the status register, if the primary parity error response bit is set in the "Primary Command Register—Offset 0x04" on page 124.
- The Tsi352 captures the parity error condition to forward it back to the initiator on the secondary bus.

6.3.3.2 Parity Error on Delayed Write Request on a Reattempt or Forwarded Transaction

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.

The parity bit is not compared when determining whether the transaction matches those in the delayed transaction queues.

Two situations 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.

Downstream Transactions

For downstream delayed write transactions, when the parity error is detected on the initiator bus and the Tsi352 has write status to return, the following events occur:

- The Tsi352 first asserts P_TRDY_b and then asserts P_PERR_b two cycles later, if the primary interface parity error response bit is set in the “Primary Command Register—Offset 0x04” on page 124.
- The Tsi352 sets the primary interface parity error detected bit in the “Primary Status Register—Offset 0x04” on page 127.
- The write status is not returned and the transaction remains in the queue because there was not an exact data and parity match.

Upstream Transactions

For upstream delayed write transactions, when the parity error is detected on the initiator bus and the Tsi352 has write status to return, the following events occur:

- The Tsi352 first asserts S_TRDY_b and then asserts S_PERR_b two cycles later, if the secondary interface parity error response bit is set in the “Bridge Control Register—Offset 0x3C” on page 151.
- The Tsi352 sets the secondary interface parity error detected bit in the “Secondary Status Register—Offset 0x1C” on page 139.
- The write status is not returned and the transaction remains in the queue because there was not an exact data and parity match.

Parity Error on a Forwarded Transaction from the Target Bus to the Initiator Bus

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:

- The Tsi352 asserts P_PERR_b two cycles after the data transfer, if both of the following are true:
  - The primary interface parity error response bit is set in the "Primary Command Register—Offset 0x04" on page 124.
  - The secondary interface parity error response bit is set in the "Bridge Control Register—Offset 0x3C" on page 151.
- Tsi352 completes the transaction normally.

For upstream transactions, in the case where the parity error is being passed back from the target bus and the initiator bus, the following events occur:

- The Tsi352 asserts S_PERR_b two cycles after the data transfer, if both of the following are true:
  - The primary interface parity error response bit is set in the "Primary Command Register—Offset 0x04" on page 124.
  - The secondary interface parity error response bit is set in the "Bridge Control Register—Offset 0x3C" on page 151.
- The Tsi352 completes the transaction normally.

6.3.4 Posted Write Transactions

The following sections show different types of posted write transactions and error conditions.

6.3.4.1 Data Parity Error on the Initiator (Primary) Bus

When the Tsi352, responding as a target on a downstream-posted write transactions, detects a data parity error on the initiator (primary) bus, the following events occur:

- The Tsi352 asserts P_PERR_b two cycles after the data transfer, if the primary interface parity error response bit is set in the "Primary Command Register—Offset 0x04" on page 124.
- The Tsi352 sets the primary interface parity error detected bit in the "Primary Status Register—Offset 0x04" on page 127.
- The Tsi352 captures and forwards the bad parity condition to the secondary bus.
- The Tsi352 completes the transaction normally.

Similarly, during upstream posted write transactions, when Tsi352, responding as a target, detects a data parity error on the initiator (secondary) bus, the following events occur:

- The Tsi352 asserts S_PERR_b two cycles after the data transfer, if the secondary interface parity error response bit is set in the "Bridge Control Register—Offset 0x3C" on page 151.
- The Tsi352 sets the secondary interface parity error detected bit in the "Secondary Status Register—Offset 0x1C" on page 139.
- The Tsi352 captures and forwards the bad parity condition to the primary bus.

- The Tsi352 completes the transaction normally.

6.3.4.2 Data Parity Error Reported on the Target (Secondary) Bus

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

- The Tsi352 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—Offset 0x3C" on page 151.
- The Tsi352 asserts P_SERR_b and sets the signaled system error bit in the status register, if all of the following conditions are met:
  - The SERR_b enable bit is set in the "Primary Command Register—Offset 0x04" on page 124.
  - The device-specific P_SERR_b disable bit for posted write parity errors is not set.
  - The secondary interface parity error response bit is set in the "Bridge Control Register—Offset 0x3C" on page 151.
  - The primary interface parity error response bit is set in the command register.
  - The Tsi352 did not detect the parity error on the primary (initiator) bus; that is, the parity error was not forwarded from the primary bus.

6.3.4.3 Data Parity Error Reported on the Target (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_b, the following events occur:

- The Tsi352 sets the data parity detected bit in the status register, if the primary interface parity error response bit is set in the command register.
- The Tsi352 asserts P_SERR_b and sets the signaled system error bit in the status register, if all of the following conditions are met:
  - The SERR_b 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.
  - The Tsi352 did not detect the parity error on the secondary (initiator) bus; that is, the parity error was not forwarded from the secondary bus.

6.3.4.4 Assertion of P_SERR_b

The assertion of P_SERR_b is used to signal the parity error condition when 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 is forwarded from the initiating bus to the target bus, P_SERR_b is not asserted.

6.4 System Error (SERR_b) Reporting

The Tsi352 uses the P_SERR_b signal to report a number of system error conditions in addition to the special case parity error conditions described in “Data Parity Errors” on page 71.

In order for the Tsi352 to assert P_SERR_b, the following must be true:

- For Tsi352 to assert P_SERR_b for any reason, the SERR_b enable bit must be set in the command register.
- Whenever Tsi352 asserts P_SERR_b, Tsi352 must also set the signaled system error bit in the status register.

6.4.1 Assertion of P_SERR_b

In compliance with the *PCI-to-PCI Bridge Architecture Specification*, Tsi352 asserts P_SERR_b in the following situations:

- The secondary SERR_b input, S_SERR_b, is asserted
- The SERR_b forward enable bit is set in the “Bridge Control Register—Offset 0x3C” on page 151.

When the Tsi352 assets P_SERR_b it also sets the received system error bit in the secondary status register.

Tsi352 can also assert P_SERR_b for any of the following reasons:

- Target abort detected during a posted write transaction
- Master abort detected during a 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 “Posted Write Transactions” on page 75)
- 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

6.4.2 Device Specific Reporting

The Tsi352 device-specific P_SERR_b status register reports the reason for Tsi352’s assertion of P_SERR_b (see “P_SERR_b Status Register — Offset 0x68” on page 166).

Most of the events listed in “Assertion of P_SERR_b” on page 77 have additional device-specific disable bits in the P_SERR_b event disable register (see “P_SERR_b Event Enable Register—Offset 0x64” on page 163) that make it possible to mask out P_SERR_b assertion for specific events. However, in the case of he master timeout condition, it has a SERR_b enable bit for in the bridge control register and therefore does not have a device-specific disable bit.

7. PCI Power Management

This chapter discusses the following topics about the Tsi352:

7.1 PCI Power Management

The Tsi352 incorporates functionality that meets the requirements of the *PCI Power Management Specification, Revision 1.1*. These features include:

- PCI Power Management registers using the Enhanced Capabilities Port (ECP) address mechanism (see "Power Management Capabilities Register—Offset 0xDC" on page 169 and "Power Management Data Register—Offset 0xE0" on page 170)
- Support for D0, D3hot and D3cold power management states
- Support for D0, D1, D2, D3hot, and D3cold power management states for devices behind the bridge
- Support of the B2 secondary bus power state when in the D3hot power management state

Table 11 shows the states and related actions that Tsi352 performs during power management transitions.No other transactions are permitted.

Table 11: Power Management Transitions

Current State	Next State	Action
D0	D3cold	Power has been removed from the Tsi352. A power-up reset must be performed to bring the Tsi352 to D0.
D0	D3hot	The Tsi352 disables the secondary clocks and drive them high (if enabled to do so by the BPCC pin)
D0	D2	Unimplemented power state. The Tsi352 ignores the write to the power state bits (power state remains at D0).
D0	D1	Unimplemented power state. The Tsi352 ignore the write to the power state bits (power state remains at D0).
D3hot	D0	Tsi352 enables secondary clock outputs and performs an internal chip reset. Signal S_RST_b is not asserted. All registers are returned to the reset values and buffers are cleared.
D3hot	D3cold	Power has been removed from the Tsi352. A power-up reset must be performed to bring the Tsi352 to D0.
D3cold	D0	Power-up reset. The Tsi352 performs the standard power-up reset functions

PME_b signals are routed from downstream devices around PCI-to-PCI bridges. PME_b signals do not pass through PCI-to-PCI bridges.

8. Reset, Clock, and Initialization

This chapter discusses the following topics about the Tsi352:

8.1 Clocking

The Tsi352 has a primary and secondary clock. The Tsi352 is a synchronous device because the secondary clock outputs are synchronous to the primary clock input.

8.1.1 Primary and Secondary Clock Inputs

The Tsi352 implements a separate clock input for each PCI interface. The primary interface is synchronized to the primary clock input, P_CLK, and the secondary interface is synchronized to the secondary clock input, S_CLK_IN. The Tsi352 operates at a maximum frequency of 66 MHz, and S_CLK_IN always operates at the same frequency as P_CLK.

The primary and secondary clock inputs must always maintain a synchronous relationship to each other; that is, their edge relationships to each other are well defined. The maximum skew between P_CLK and S_CLK_IN edges is 7 ns. The minimum skew between P_CLK and S_CLK edges is 0 ns. The secondary clock edge must never precede the primary clock edge.

8.1.2 Secondary Clock Outputs

Tsi352 has five secondary clock outputs, S_CLKOUT[4:0], that can be used as clock inputs for up to four external secondary bus devices and for the Tsi352 secondary clock input S_CLK_IN. One of the S_CLKOUT must be used for S_CLK_IN

The S_CLKOUT outputs are derived from P_CLK and have same frequency as P_CLK. The S_CLKOUT edges are delayed from P_CLK edges by a minimum of 0 ns and a maximum of 5 ns. The maximum skew between S_CLKOUT edges is 500 ps. Therefore, to meet the P_CLK and S_CLK_IN requirements (see "Primary and Secondary Clock Inputs" on page 81) no more than 2 ns of delay is allowed for secondary clocks returning to the device secondary clock inputs.

The following rules should be followed regarding the secondary output clocks:

- Each secondary clock output is limited to one load
- S_CLKOUT[4] should be used for the Tsi352 S_CLK_IN input

Although any of the S_CLKOUT[4:0] signals can be used for the S_CLK_IN input, IDT recommends using the S_CLKOUT[4] signal.

- Unused secondary clocks should be disabled through software by writing to the "Secondary Clock Control Register—Offset 0x68" on page 165

8.1.3 Clock Run

The Tsi352 supports the PCI clock run protocol defined in the *PCI Mobile Design Guide 1.0*. The P_CLKRUN_b signal is high when the system's central resource initiates to stop the primary clock (P_CLK). The Tsi352 then signals that it allows the PCI clock to be stopped by keeping P_CLKRUN_b high, or it initiates P_CLK to remain operating by driving P_CLKRUN_b low for two clocks. After the two clocks have elapsed, the system's central resource keeps P_CLKRUN_b low.

The Tsi352 keeps the primary clock operating under the following conditions:

- P_CLK_STOP bit is set to 1 in the "CLKRUN Register — Offset 0x58" on page 162
- There is a pending transaction operating through the bridge
- A secondary device requires the clock
- The secondary clock run protocol is enabled by setting the Secondary CLKRUN Enable bit in the "CLKRUN Register — Offset 0x58" on page 162. The primary is responsible for the initiation of stopping or slowing down the secondary clock. However, if the CLKRUN Mode bit is set to 1, the secondary clock is stopped when the bus is idle and there are no other cycles from the primary bus.

8.2 Reset

The Tsi352 can either be reset by hardware by asserting the P_RST_b signal or software by programming the Chip Reset bit in the "Chip Control Register/Diagnostic Control — Offset 0x40" on page 155.

8.2.1 Primary Interface Reset

The Tsi352 has one reset input, P_RST_b. When P_RST_b is asserted, the following events occur:

- The Tsi352 immediately tristates all primary and secondary PCI interface signals.
- The Tsi352 performs a chip reset.
- Registers that have default values are reset.

The P_RST_b asserting and de-asserting edges can be asynchronous to P_CLK and S_CLK.

8.2.2 Secondary Interface Reset

The Tsi352 is responsible for driving the secondary bus reset signal, S_RST_b. The Tsi352 asserts S_RST_b when any of the following conditions is met:

- P_RST_b is asserted
  - S_RST_b is asserted when P_RST_b is asserted on the primary interface of the Tsi352.
- Programming Secondary Bus Reset, in the "Bridge Control Register—Offset 0x3C" on page 151
  - Software can force S_RST_b on the secondary bus by programming this bit to 1. The software must clear this bit to 0 to de-assert S_RST_b. Following the clear operation, the Tsi352 de-asserts S_RST_b after a 100 us period.
- Chip Reset in the "Chip Control Register/Diagnostic Control — Offset 0x40" on page 155.

— Software can program this bit to 1 to reset assert both P_RST_b and S_RST_b on the secondary interface. The Tsi352 de-asserts S_RST_b automatically after 100 us and clears the bit to 0, provided the secondary reset bit is cleared in the “Bridge Control Register—Offset 0x3C” on page 151. Signal S_RST_b remains asserted until a configuration write operation clears the secondary reset bit.

Writing a 1 to the Chip Reset bit sets the Secondary Bus Reset bit in the Bridge Control register.

8.2.2.1 S_RST_b Impact

When S_RST_b is asserted by means of the secondary reset bit in the “Bridge Control Register—Offset 0x3C” on page 151, the Tsi352 remains accessible during secondary interface reset and continues to respond to accesses to its configuration space from the primary interface.

When S_RST_b is asserted, all secondary PCI interface control signals, including the secondary grant outputs, are immediately tristated. Signals S_AD, S_CBE_b, and S_PAR are driven low for the duration of S_RST_b assertion. All posted write and delayed transaction data buffers are reset; therefore, any transactions residing in the Tsi352 buffers at the time of secondary reset are discarded.

8.2.3 Chip Reset

The Chip Reset bit in the “Chip Control Register/Diagnostic Control — Offset 0x40” on page 155 can be used to reset Tsi352 and the secondary bus. During chip reset, Tsi352 is not accessible.

When the Chip Reset bit is set, the entire Tsi352 is reset and all signals are tristated. In addition, S_RST_b is asserted, and the secondary reset bit is automatically set. The S_RST_b signal remains asserted until a configuration write operation clears the secondary reset bit.

As soon as chip reset completes, within 20 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.

9. Signal Descriptions

This chapter discusses the following topics about the Tsi352's signals:

9.1 Overview

This chapter provides descriptions of the Tsi352 signal pins, grouped by function.

Table 12: Signal Types

Signal Type	Description
I	Standard input only
O	Standard output only
TS	Tristate bidirectional
STS	Sustained tristate. Active low signal must be pulled high for one cycle when deasserting.
OD	Standard open drain

The _b signal name suffix indicates that the signal is asserted when it is at a low voltage level and corresponds to the # suffix in the *PCI Local Bus Specification*. If this suffix is not present, the signal is asserted when it is at a high voltage level.

9.2 Primary PCI Interface Signals

The Primary PCI Interface signals are described in Table 13.

Table 13: Primary PCI Interface Signals

Signal Name	Pin Number	Pin Type	Description
P_AD[31:0]	70,72,73,74,76, 77,78,79,84,85, 87,88,89,91,92, 93,109,110,111, 113,114,115,117, 118,123,124,126, 127,129,130,132, 133	TS	Primary PCI Interface address/data These signals are a multiplexed address and data bus. During the address phase, or phases, of a transaction, the initiator drives a physical address on P_AD[31:0]. During the data phases of a transaction, the initiator drives write data, or the target drives read data, on P_AD[31:0]. When the primary PCI bus is idle, the Tsi352 drives P_AD to a valid logic level when P_GNT_b is asserted.
P_CBE_b[3:0]	82,95,107,122	TS	Transaction type on P_CBE_b[3:0] When there are two address phases, the first address phase carries the dual address command and the second address phase carries the transaction type. For both read and write transactions, the initiator drives byte enables on P_CBE_b[3:0] during the data phases. When the primary PCI bus is idle, the Tsi352 drives P_CBE_b to a valid logic level when P_GNT_b is asserted.
P_PAR	106	TS	Primary PCI Interface parity Signal P_PAR carries the even parity of the 36 bits of P_AD[31:0] and P_CBE_b[3:0] for both address and data phases. Signal P_PAR is driven by the same agent that has driven the address (for address parity) or the data (for data parity). The P_PAR signal contains valid parity one cycle after the address is valid (indicated by assertion of P_FRAME_b, or one cycle after data is valid (indicated by assertion of P_IRDY_b for write transactions and P_TRDY_b for read transactions). The P_PAR signal is driven by the device-driving read or write data one cycle after P_AD is driven. The P_PAR signal is tristated one cycle after the P_AD lines are tristated. Devices receiving data sample P_PAR as an input to check for possible parity errors. When the primary PCI bus is idle, the Tsi352 drives P_PAR to a valid logic level when P_GNT_b is asserted (one cycle after the P_AD bus is parked).
P_FRAME_b	96	STS	Primary PCI Interface FRAME_b The P_FRAME_b signal is driven by the initiator of a transaction to indicate the beginning and duration of an access on the primary PCI bus. The P_FRAME_b signal assertion (falling edge) indicates the beginning of a PCI transaction. While P_FRAME_b remains asserted, data transfers can continue. The deassertion of P_FRAME_b indicates the final data phase requested by the initiator. When the primary PCI bus is idle, P_FRAME_b is driven to a deasserted state for one cycle and then is sustained by an external pull-up resistor.

Table 13: Primary PCI Interface Signals

Signal Name	Pin Number	Pin Type	Description
P_IRDY_b	97	STS	Primary PCI Interface IRDY_b. The P_IRDY_b signal is driven by the initiator of a transaction to indicate the initiator's ability to complete the current data phase on the primary PCI bus. During a write transaction, assertion of P_IRDY_b indicates that valid write data is being driven on the P_AD bus. During a read transaction, assertion of P_IRDY_b indicates that the initiator is able to accept read data for the current data phase. Once asserted during a given data phase, P_IRDY_b is not deasserted until the data phase completes. When the primary bus is idle, P_IRDY_b is driven to a deasserted state for one cycle and then is sustained by an external pull-up resistor.
P_TRDY_b	99	STS	Primary PCI Interface TRDY_b. The P_TRDY_b signal is driven by the target of a transaction to indicate the target's ability to complete the current data phase on the primary PCI bus. During a write transaction, assertion of P_TRDY_b indicates that the target is able to accept write data for the current data phase. During a read transaction, assertion of P_TRDY_b indicates that the target is driving valid read data on the P_AD bus. Once asserted during a given data phase, P_TRDY_b is not deasserted until the data phase completes. When the primary bus is idle, P_TRDY_b is driven to a deasserted state for one cycle and then is sustained by an external pull-up resistor.
P_DEVSEL_b	100	STS	Primary PCI Interface DEVSEL_b. P_DEVSEL_b is asserted by the target, indicating that the device is accepting the transaction. As a target, Tsi352 performs positive decoding on the address of a transaction initiated on the primary bus to determine whether to assert P_DEVSEL_b. As an initiator of a transaction on the primary bus, Tsi352 looks for the assertion of P_DEVSEL_b within five cycles of P_FRAME_b assertion; otherwise, Tsi352 terminates the transaction with a master abort. When the primary bus is idle, P_DEVSEL_b is driven to a deasserted state for one cycle and then is sustained by an external pull-up resistor.
P_STOP_b	101	STS	Primary PCI Interface STOP_b. The P_STOP_b signal is driven by the target of the current transaction, indicating that the target is requesting the initiator to stop the current transaction on the primary bus. • When P_STOP_b is asserted in conjunction with P_TRDY_b and P_DEVSEL_b assertion, a disconnect with data transfer is being signaled. • When P_STOP_b and P_DEVSEL_b are asserted, but P_TRDY_b is deasserted, a target disconnect without data transfer is being signaled. When this occurs on the first data phase, that is, no data is transferred during the transaction, this is referred to as a target retry. • When P_STOP_b is asserted and P_DEVSEL_b is deasserted, the target is signaling a target abort. When the primary bus is idle, P_STOP_b is driven to a deasserted state for one cycle and then is sustained by an external pull-up resistor.

Table 13: Primary PCI Interface Signals

Signal Name	Pin Number	Pin Type	Description
P_IDSEL	83	I	Primary PCI Interface IDSEL_b. The P_IDSEL signal is used as the chip select line for Type 0 configuration accesses to the Tsi352 configuration space. When P_IDSEL is asserted during the address phase of a Type 0 configuration transaction, the Tsi352 responds to the transaction by asserting P_DEVSEL_b.
P_PERR_b	104	STS	Primary PCI Interface PERR_b. The P_PERR_b signal is asserted when a data parity error is detected for data received on the primary interface. The timing of P_PERR_b corresponds to P_PAR driven one cycle earlier and P_AD and P_CBE_b driven two cycles earlier. The P_PERR_b signal is asserted by the target during write transactions, and by the initiator during read transactions. When the primary bus is idle, P_PERR_b is driven to a deasserted state for one cycle and then is sustained by an external pull-up resistor.
P_SERR_b	105	OD	Primary PCI Interface SERR_b. The P_SERR_b signal can be driven low by any device on the primary bus to indicate a system error condition. The Tsi352 can assert P_SERR_b for the following reasons: • Address parity error • Posted write data parity error on target bus • Secondary bus S_SERR_b assertion • Master abort during posted write transaction • Target abort during posted write transaction • Posted write transaction discarded • Delayed write request discarded • Delayed read request discarded • Delayed transaction master timeout The P_SERR_b signal is pulled up through an external resistor.
P_REQ_b	69	TS	Primary PCI bus REQ_b. The P_REQ_b signal is asserted by the Tsi352 to indicate to the primary bus arbiter that it needs to start a transaction on the primary bus. When the Tsi352 receives a target retry or disconnect in response to initiating a transaction, the Tsi352 deasserts P_REQ_b for at least two PCI clock cycles before asserting it again.
P_GNT_b	68	I	Primary PCI bus GNT_b. When asserted, P_GNT_b indicates to the Tsi352 that access to the primary bus is granted. The Tsi352 can start a transaction on the primary bus when the bus is idle and P_GNT_b is asserted. When the Tsi352 has not requested use of the bus and P_GNT_b is asserted, the Tsi352 must drive P_AD, and P_PAR to valid logic levels.

9.3 Secondary PCI Interface Signals

The Secondary PCI Interface signals are described in Table 14.

Table 14: Secondary PCI Interface Signals

Signal Name	Pin Number	Type	Description
S_AD[31:0]	36,35,33,32,31,29, 28,26,24,22,21,20, 18,17,16,14,156,155, 153,152,150,149, 148,146,144,142, 141,140,138,137, 136,134	TS	Secondary PCI Interface address/data. These signals are a multiplexed address and data bus. During the address phase or phases of a transaction, the initiator drives a physical address on S_AD[31:0]. During the data phases of a transaction, the initiator drives write data, or the target drives read data, on S_AD[31:0]. When the secondary PCI bus is idle, the Tsi352 drives S_AD to a valid logic level when its secondary bus grant is asserted.
S_CBE_b[3:0]	25,13,158,145	TS	Secondary PCI Interface command/byte enables. These signals are a multiplexed command field and byte enable field. During the address phase or phases of a transaction, the initiator drives the transaction type on S_CBE_b[3:0]. When there are two address phases, the first address phase carries the dual address command and the second address phase carries the transaction type. For both read and write transactions, the initiator drives byte enables on S_CBE_b[3:0] during the data phases. When the secondary PCI bus is idle, the Tsi352 drives S_CBE_b to a valid logic level when its secondary bus grant is asserted.
S_PAR	2	TS	Secondary PCI Interface parity. The S_PAR signal carries the even parity of the 36 bits of S_AD[31:0] and S_CBE_b[3:0] for both address and data phases. The S_PAR signal is driven by the same agent that has driven the address (for address parity) or the data (for data parity). The S_PAR signal contains valid parity one cycle after the address is valid (indicated by assertion of S_FRAME_b), or one cycle after data is valid (indicated by assertion of S_IRDY_b for write transactions and S_TRDY_b for read transactions). The S_PAR signal is driven by the device driving read or write data one cycle after S_AD is driven. Signal S_PAR is tristated one cycle after the S_AD lines are tristated. Devices receiving data sample S_PAR as an input in order to check for possible parity errors. When the secondary PCI bus is idle, the Tsi352 drives S_PAR to a valid logic level when its secondary bus grant is asserted (one cycle after the S_AD bus is parked).

Table 14: Secondary PCI Interface Signals

Signal Name	Pin Number	Type	Description
S_FRAME_b	11	STS	Secondary PCI Interface FRAME_b. The S_FRAME_b signal is driven by the initiator of a transaction to indicate the beginning and duration of an access on the secondary PCI bus. The S_FRAME_b signal assertion (falling edge) indicates the beginning of a PCI transaction. While S_FRAME_b remains asserted, data transfers can continue. The deassertion of S_FRAME_b indicates the final data phase requested by the initiator. When the secondary PCI bus is idle, S_FRAME_b is driven to a deasserted state for one cycle and then is sustained by an external pull-up resistor.
S_IRDY_b	10	STS	Secondary PCI Interface IRDY_b. The S_IRDY_b signal is driven by the initiator of a transaction to indicate the initiator's ability to complete the current data phase on the secondary PCI bus. During a write transaction, assertion of S_IRDY_b indicates that valid write data is being driven on the S_AD bus. During a read transaction, assertion of S_IRDY_b indicates that the initiator is able to accept read data for the current data phase. Once asserted during a given data phase, S_IRDY_b is not deasserted until the data phase completes. When the secondary bus is idle, S_IRDY_b is driven to a deasserted state for one cycle and then is sustained by an external pull-up resistor.
S_TRDY_b	9	STS	Secondary PCI Interface TRDY. The S_TRDY_b signal is driven by the target of a transaction to indicate the target's ability to complete the current data phase on the secondary PCI bus. During a write transaction, assertion of S_TRDY_b indicates that the target is able to accept write data for the current data phase. During a read transaction, assertion of S_TRDY_b indicates that the target is driving valid read data on the S_AD bus. Once asserted during a given data phase, S_TRDY_b is not deasserted until the data phase completes. When the secondary bus is idle, S_TRDY_b is driven to a deasserted state for one cycle and then is sustained by an external pull-up resistor.
S_DEVSEL_b	7	STS	Secondary PCI Interface DEVSEL_b. The S_DEVSEL_b signal is asserted by the target, indicating that the device is accepting the transaction. As a target, the Tsi352 performs positive decoding on the address of a transaction initiated on the secondary bus in order to determine whether to assert S_DEVSEL_b. As an initiator of a transaction on the secondary bus, the Tsi352 looks for the assertion of S_DEVSEL_b within five cycles of S_FRAME_b assertion; otherwise, the Tsi352 terminates the transaction with a master abort. When the secondary bus is idle, S_DEVSEL_b is driven to a deasserted state for one cycle and then is sustained by an external pull-up resistor.

Table 14: Secondary PCI Interface Signals

Signal Name	Pin Number	Type	Description
S_STOP_b	6	STS	Secondary PCI Interface STOP_b. The S_STOP_b signal is driven by the target of the current transaction, indicating that the target is requesting the initiator to stop the current transaction on the secondary bus. • When S_STOP_b is asserted in conjunction with S_TRDY_b and S_DEVSEL_b assertion, a disconnect with data transfer is being signaled. • When S_STOP_b and S_DEVSEL_b are asserted, but S_TRDY_b is deasserted, a target disconnect without data transfer is being signaled. When this occurs on the first data phase, that is, no data is transferred during the transaction, this is referred to as a target retry. • When S_STOP_b is asserted and S_DEVSEL_b is deasserted, the target is signaling a target abort. When the secondary bus is idle, S_STOP_b is driven to a deasserted state for one cycle and then is sustained by an external pull-up resistor.
S_PERR_b	4	STS	Secondary PCI Interface PERR_b. The S_PERR_b signal is asserted when a data parity error is detected for data received on the secondary interface. The timing of S_PERR_b corresponds to S_PAR driven one cycle earlier and S_AD driven two cycles earlier. Signal S_PERR_b is asserted by the target during write transactions, and by the initiator during read transactions. When the secondary bus is idle, S_PERR_b is driven to a deasserted state for one cycle and then is sustained by an external pull-up resistor.
S_SERR_b	3	I	Secondary PCI Interface SERR_b. The S_SERR_b can be driven low by any device except the Tsi352 on the secondary bus to indicate a system error condition. The Tsi352 samples S_SERR_b as an input and conditionally forwards it to the primary bus on P_SERR_b. The Tsi352 does not drive S_SERR_b. Signal S_SERR_b is pulled up through an external resistor.
S_REQ_b[3:0]	42,39,38,37	I	Secondary PCI Interface REQ_b. The Tsi352 accepts four request inputs, S_REQ_b[3:0], into its secondary bus arbiter. The Tsi352 request input to the arbiter is an internal signal. Each request input can be programmed to be in either a high priority rotating group or a low priority rotating group. An asserted level on an S_REQ_b pin indicates that the corresponding master needs to initiate a transaction on the secondary PCI bus.
S_GNT_b[3:0]	47,45,44,43	TS	Secondary PCI Interface GNT_b. The Tsi352 secondary bus arbiter can assert one of four secondary bus grant outputs, S_GNT_b[3:0], to indicate that an initiator can start a transaction on the secondary bus if the bus is idle.

9.4 Clocks and Resets

The clocking and reset signals are described in Table 15.

Table 15: Clocks and Resets

Signal Name	Pin Number	Type	Description
P_CLK	66	I	Primary Clock Input This signal provides timing for all transactions on the primary interface.
S_CLKOUT[4:0]	61,59,57,55,53	O	Secondary Clock Output The S_CLKOUT[4:0] signals are five clock outputs generated from the primary interface clock input, P_CLK. These clocks operate at the same frequency of P_CLK. When these clocks are used, S_CLKOUT[4] must be fed back to the secondary clock input, S_CLK_IN. Unused clock outputs can be disabled by writing the secondary clock disable bits in configuration space.
S_CLK_IN	51	I	Secondary Clock Input This clock synchronizes the Tsi352 to secondary bus clocks
P_RST_b	64	I	Primary reset When P_RST_b is active, all PCI signals should be asynchronously tri-stated.
S_RST_b	48	O	Secondary reset Asserted when any of the following conditions are met: • Signal P_RST_b is asserted. • Chip Reset bit in bridge control register in is set (see “Bridge Control Register—Offset 0x3C” on page 151) When this signal is asserted, all control signals are tri-stated and zeroes are driven on S_AD, S_CBE, and S_PAR.

9.5 Miscellaneous Signals

Miscellaneous signals are described in Table 16.

Table 16: Miscellaneous Signals

Signal Name	Pin Number	Type	Description
MS0	120	I	Mode Select 0
MS1/BPCC	159	I	Mode Select 1 when MS0 is low, BPCC when MS0 is high
P_MFUNC	102	I/O	Primary multifunction terminal This terminal can be configured as P_CLKRUN_b, P_LOCK_b or HS_ENUM_b depending on the values of MS0 and MS1
S_MFUNC	5	I/O	Secondary multifunction terminal This terminal can be configured as S_CLKRUN_b, S_LOCK_b or HS_SWITCH_b depending on the values of MS0 and MS1
S_CFN_b	49	I	Secondary external arbiter enable When tied low, S_CFN_b enables the Tsi352 secondary bus arbiter. When tied high, S_CFN_b disables the internal arbiter. An external secondary bus arbiter must then be used. Signal S_REQ_b[0] is reconfigured to be the Tsi352 secondary bus grant input, and S_GNT_b[0] is reconfigured to be the Tsi352 secondary bus request output, when an external arbiter is used. Secondary bus parking is completed when S_REQ_b[0] is asserted, the secondary bus is idle, and the Tsi352 does not need to initiate a transaction.
SCAN_TM_b	63	I	Scan Test Mode Enable (Manufacturing test pin) For normal operation, pull SCAN_TM_b to high. This signal has an internal pull-up.
SCAN_EN/ HS_LED_OUT	62	I/O	Scan output when SCAN_TM_b is asserted and HS_LED_OUT when SCAN_TM_b is deasserted.

9.6 Power Supply Signals

Table 17: Power Supply Signals

Name	Pin Type	Description
VDD	Core power	3.3V core power
P_V_{CCP}	Primary Interface I/O Voltage	This signal must be tied to either 3.3V or 5.0V, depending on the signaling voltage of the primary interface.
S_V_{CCP}	Secondary Interface I/O Voltage	This signal must be tied to either 3.3V or 5.0V, depending on the signaling voltage of the secondary interface.
VSS	GND	Ground

10. Electrical Characteristics

Topics discussed include the following:

10.1 Absolute Maximum Ratings

The following tables contain the absolute maximum ratings for the Tsi352.

Table 18: Absolute Maximum Ratings

Symbol	Parameter	Minimum	Maximum	Units
T_{STG}	Storage temperature	-40	125	^{o}C
T_C	Case temperature under bias	-40	120	^{o}C
Voltage with respect with ground				
V_{DD}	3.3V DC I/O supply voltage	-0.3	3.63	V
V_{IL}	Minimum signal input voltage	-0.5	-	V
V_{IH}	Maximum signal input voltage	-	V_{DD}[a] + 0.5	V

a. The V_{DD} reference is dependent on the input pad supply rail.

Table 19: Absolute Maximum Ratings — PCI

Symbol	Parameter	Min	Max	Unit
V_{ESD_HBM}	Maximum ESD Voltage Discharge Tolerance for Human Body Model (HBM). [Test Conditions per JEDEC standard - JESD22-A114-B]	-	2000	V
V_{ESD_CDM}	Maximum ESD Voltage Discharge Tolerance for Charged Device Model (CDM). Test Conditions per JEDEC standard - JESD22-C101-A	-	500	V

10.2 Recommended Operating Conditions

The following table contains the recommended operating conditions for the Tsi352.

Table 20: Recommended Operating Conditions

Symbol	Parameter	Minimum	Maximum	Units	Notes
V_{DD}	3.3V DC I/O supply voltage	3.0	3.6	V	-
T_A	Ambient temperature	0.0	85	^oC	a, b
T_{JUNC}	Junction temperature	0.0	125	^oC	-

a. No heat sink, no air flow.

b. Higher ambient temperatures are permissible provided T_{JUNC} is not violated. For heat sink and air flow requirements for higher temperature operation, see .

10.3 Supply Current

Table 21 contains supply current characteristics for the Tsi352 for V_{DD} at 3.3V.

Table 21: Supply Current Characteristics at 3.3V

Supply Current at 66MHz				
$I_{DD\ max}$	Maximum V_{DD} current draw	650	mA	a
$I_{DD\ idle}$	Idle V_{DD} current draw	200	mA	b
Supply Current at 33MHz				
$I_{DD\ max}$	Maximum V_{DD} current draw	310	mA	a
$I_{DD\ idle}$	Idle V_{DD} current draw	100	mA	b

a. Measured with 4 loads on secondary side with each bit on the bus switching.

b. No bus switching,

10.4 Power Supply Sequencing

If either P_VCCP or S_VCCP is powered from a 5V supply, and if the 5V supply presents a low impedance path to ground when it is not active, IDT recommends powering this supply before, or concurrently, with the VDD rail.

10.5 DC Operating Characteristics

The following table contains DC operating characteristics for the Tsi352.

Table 22: DC Operating Characteristics

Symbol	Parameter	Condition	Minimum	Maximum	Units	Notes
V_{IL_MISC}	Miscellaneous Input Low Voltage	-	-0.5	$0.3V_{DD}$	V	a
V_{IH_MISC}	Miscellaneous Input High Voltage	-	2.0	V_{DD} + 0.5	V	a
P_V_{CCP}	Primary Interface I/O Voltage	-	3.0	5.5	V	-
S_V_{CCP}	Secondary Interface I/O Voltage	-	3.0	5.5	V	-
V_{IL}	Input Low Voltage	-	-0.5	$0.3V_{DD}$	V	
V_{IH}	PCI Input High Voltage	-	$0.5V_{DD}$	V_{DD} + 0.5	V	
V_{OL_MISC}	Misc Output Low Voltage	I_{OL} = 1.5mA	-	0.5	V	a
V_{OH_MISC}	Misc Output High Voltage	I_{OH} = -0.5mA	V_{DD} - 0.5	-	V	a, b
V_{OL}	Output Low Voltage	I_{OL} = 1.5mA	-	$0.1V_{DD}$	V	
V_{OL5V}	Output Low Voltage (5V Signaling)	IOH = 6mA	-	0.5	V	
V_{OH}	Output High Voltage	I_{OH} = -0.5mA	$0.9V_{DD}$	-	V	
V_{OH5V}	Output High Voltage (5V Signaling)	IOH = -2mA	2.4	-	V	
C_{IN}	Input Pin Capacitance	-	-	10	pF	-
C_{CLK}	Clock Pin Capacitance	-	5	12	pF	-
C_{IDSEL}	IDSEL Pin Capacitance	-	-	8	pF	-
L_{PIN}	Input Pin Inductance	-	-	20	nH	-

a. Miscellaneous (Misc) signals include all 3.3V signals that are not PCI.
b. V_{DD} = Min, I/O supply = Min.

10.6 AC Timing Specifications

This section discusses AC timing specifications for the Tsi352.

10.6.1 PCI Interface AC Signal Timing

Table 23: PCI Clock (PCI_CLK) Specification

Symbol	Parameter	PCI		Units	Notes
		Min	Max		
T_{C_PCI33}	33MHz PCI Clock Cycle Time	30	-	ns	-
T_{CH_PCI33}	33MHz PCI Clock High Time	11	-	ns	-
T_{CL_PCI33}	33MHz PCI Clock Low Time	11	-	ns	-
T_{C_PCI66}	66MHz PCI Clock Cycle Time	15	30	ns	-
T_{CH_PCI66}	66MHz PCI Clock High Time	6	-	ns	-
T_{CL_PCI66}	66MHz PCI Clock Low Time	6	-	ns	-
T_{SR_PCI}	PCI Clock Slew Rate	1	4	V/ns	-
T_{DELAY}	P_CLK (rising edge) to S_CLKOUT[4:0] (rising edge) Delay	2	4.2	ns	a

a. With 20pF load.

Table 24: AC Specifications for PCI Interface

Symbol	Parameter	PCI 66		PCI 33		Units	Notes
		Min	Max	Min	Max		
T_{OV1}	Clock to Output Valid Delay for bused signals	2	6	2	11	ns	a, b, c
T_{OV2}	Clock to Output Valid Delay for point to point signals	2	6	2	12	ns	a, b, c
T_{OF}	Clock to Output Float Delay	-	14	-	28	ns	a, d
T_{IS1}	Input Setup to clock for bused signals	3	-	7	-	ns	c, e, f
T_{IS2}	Input Setup to clock for point to point signals	5	-	10,12	-	ns	c, d
T_{IH1}	Input Hold time from clock	0	-	0	-	ns	d

Table 24: AC Specifications for PCI Interface *(Continued)*

Symbol	Parameter	PCI 66		PCI 33		Units	Notes
		Min	Max	Min	Max		
T_{RST}	Reset Active Time	1	-	1	-	ms	
T_{RF}	Reset Active to output float delay	-	40	-	40	ns	-

a. See the timing measurement conditions in Figure 8 on page 100.

b. See Figure 10 on page 101, Figure 11 on page 101, and Figure 12 on page 101.

c. Setup time for point-to-point signals applies to P[x]_REQ_B and P[x]_GNT_B only. All other signals are bused.

d. For purposes of Active/Float timing measurements, the HI-Z or “Off” state is defined to be when the total current delivered through the component pin is less than or equal to the leakage current specification.

e. See the timing measurement conditions in Figure 7 on page 100.

f. Setup time applies only when the device is not driving the pin. Devices cannot drive and receive signals at the same time.

10.7 AC Timing Waveforms

This section contains AC timing waveforms for the Tsi352.

Figure 7: Input Timing Measurement Waveforms

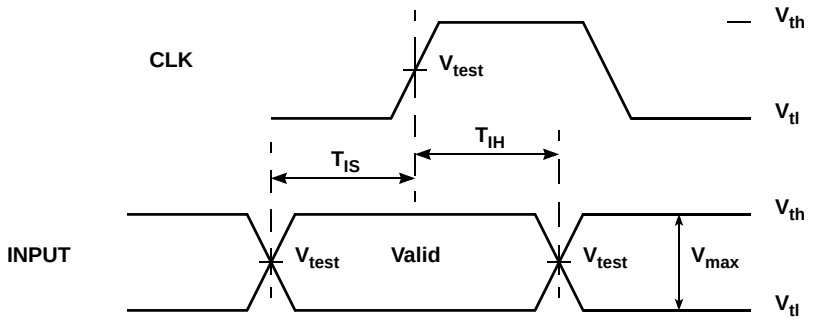


Figure 8: Output Timing Measurement Waveforms

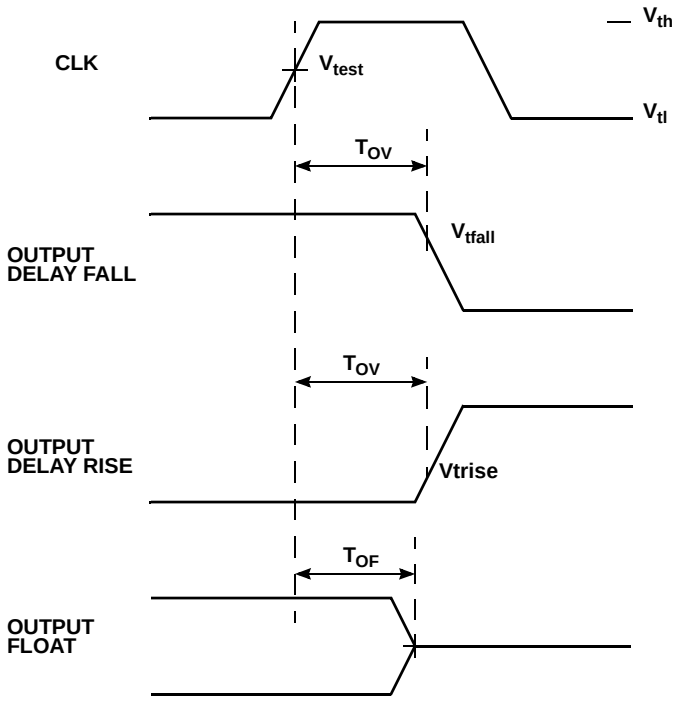


Figure 9: AC Test Load for All Signals Except PCI

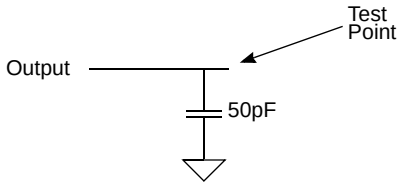


Figure 10: PCI $T_{OV\ (max)}$ Rising Edge AC Test Load

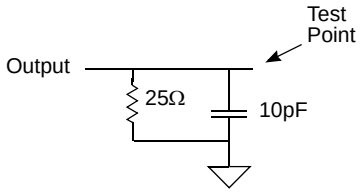


Figure 11: PCI $T_{OV\ (max)}$ Falling Edge AC Test Load

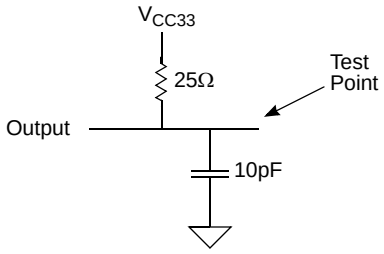
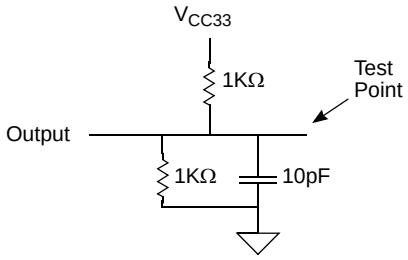


Figure 12: PCI $T_{OV\ (min)}$ AC Test Load



11. Packaging

This chapter discusses the following topics about the Tsi352's 160-pin PQFP package:

11.1 Mechanical Diagram

The 160-pin PQFP package figures have symbols that describe measurements of the package. Table 25 outlines the symbol values.

Table 25: PQFP Symbol Values

Symbol	Millimeter		
	Minimum	Nominal	Maximum
A	-	-	4.10
A1	0.25	-	-
A2	3.20	3.32	3.60
D E	31.20 BASIC		
D1 E1	28.00 BASIC		
R2	0.13	-	0.30
R1	0.13	-	-
Theta	0	-	7
Theta1	0	-	-
Theta2 Theta3	8 REF		
c	0.11	0.15	0.23
L	0.73	0.88	1.03
L1	1.60 REF		
S	0.20	-	-
b	0.22	0.30	0.38

Table 25: PQFP Symbol Values

Symbol	Millimeter		
	Minimum	Nominal	Maximum
e	0.65 BASIC		
D2	25.35		
E2	25.35		
aaa	0.25		
bbb	0.20		
ccc	-	0.10	-
ddd	-	0.13	-

The following figures show the Tsi352 mechanical diagram

Figure 13: Tsi352 Package - Top View

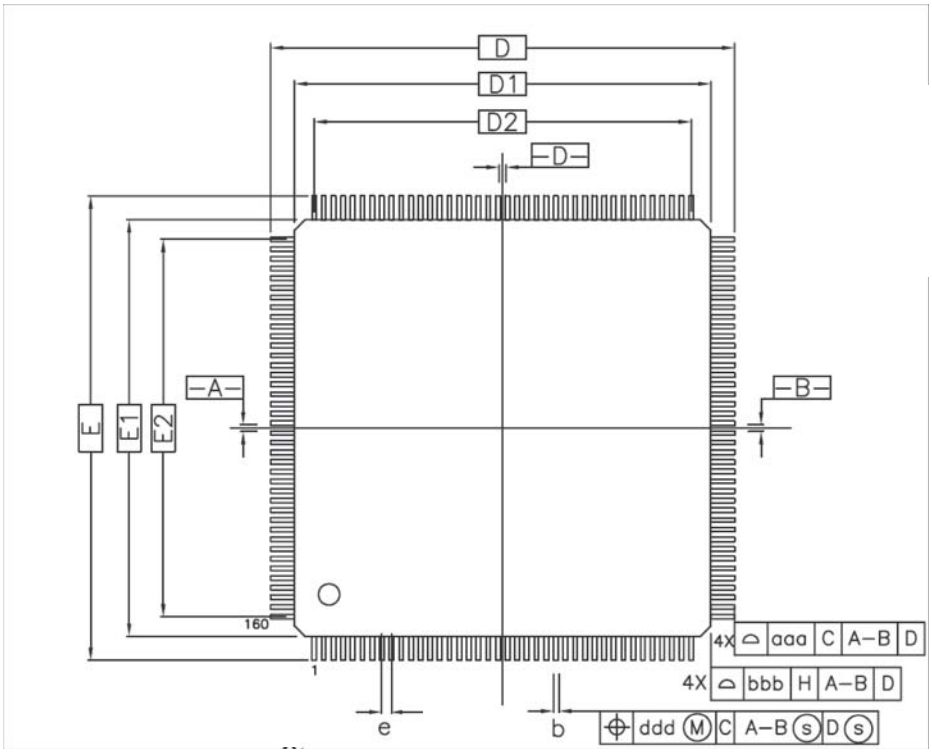


Figure 14: Tsi352 Package - Side View

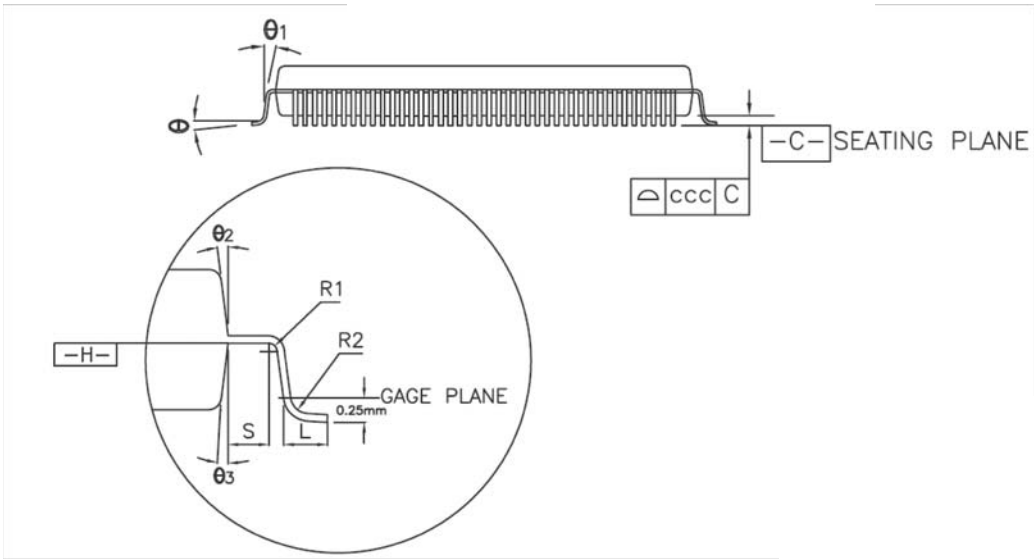
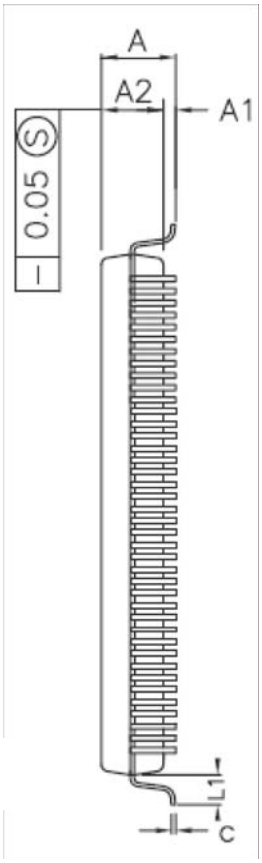


Figure 15: Tsi352 Package - Side View



11.2 Thermal Characteristics

Heat generated by the packaged IC has to be removed from the package to ensure that the IC is maintained within its functional and maximum design temperature limits. If heat buildup becomes excessive, the IC temperature can exceed the temperature limits. A consequence of this is that the IC can fail to meet the performance specifications and the reliability objectives can be affected.

Failure mechanisms and failure rate of a device have an exponential dependence of the IC operating temperatures. Thus, the control of the package temperature, and by extension the Junction Temperature, is essential to ensure product reliability. The Tsi352 is specified safe for operation when the Junction Temperature is within the recommended limits.

Table 26 shows the simulated $Theta_{jb}$ and $Theta_{jc}$ thermal characteristics of the Tsi352 packages.

Table 26: Thermal Characteristics of the Tsi352

Interface	Tsi352 PQFP Result
$Theta_{jb}$ (junction to board)	24.7 °C/watt
$Theta_{jc}$ (junction to case)	15.4 °C/watt

11.2.1 Junction-to-Ambient Thermal Characteristics (Theta ja)

Table 27 shows the simulated $Theta_{ja}$ thermal characteristic of the Tsi352 PQFP package. The results in Table 27 are based on a JEDEC Thermal Test Board configuration (JESD51-9) and do not factor in system level characteristics. As such, these values are for reference only.

The Theta $_{ja}$ thermal resistance characteristics of a package depend on multiple system level variables (see “System-level Characteristics” on page 107).

Table 27: Simulated Junction to Ambient Characteristics

Package	$Theta_{ja}$ at Specified Airflow (no Heat Sink)		
	0 m/s	1 m/s	2 m/s
Tsi352 PQFP	28.5 °C/watt	26.4 °C/watt	25.4 °C/watt

11.2.2 System-level Characteristics

The thermal resistance characteristics of a package depend on multiple variables other than the package. In an application, the following system-level characteristics and environmental issues must be taken into account:

- Package mounting (vertical / horizontal)
- System airflow conditions (laminar / turbulent)
- Heat sink design and thermal characteristics
- Heat sink attachment method
- PWB size, layer count and conductor thickness
- Influence of the heat dissipating components assembled on the PWB (neighboring effects)

11.2.3 Example on Thermal Data Usage

Based on the $Theta_{JA}$ data and specified conditions, the following formula can be used to derive the junction temperature (Tj) of the Tsi352 with a 0m/s airflow:

- $Tj = Theta_{JA} * P + Tamb$.

  Where: Tj is Junction Temperature, P is the Power consumption, Tamb is the Ambient Temperature

Assuming a power consumption (P) of 0.88 W and an ambient temperature (Tamb) of 85°C, the resulting junction temperature (Tj) for the PQFP package would be 110°C.

11.3 Moisture Sensitivity

The Tsi352's moisture sensitivity level (MSL) is three.

12. Pinlist Information

This chapter provides the following pinlist information for the Tsi352:

12.1 Ball Location to Signal Name

Table 28: Tsi352 160 PQFP Pinlist

Pin Number	Signal Name
1	VSS
2	S_PAR
3	S_SERR_b
4	S_PERR_b
5	S_MFUNC
6	S_STOP_b
7	S_DEVSEL_b
8	VDD
9	S_TRDY_b
10	S_IRDY_b
11	S_FRAME_b
12	VSS
13	S_CBE_b[2]
14	S_AD[16]
15	VDD
16	S_AD[17]
17	S_AD[18]
18	S_AD[19]
19	VSS
20	S_AD[20]
21	S_AD[21]
22	S_AD[22]
23	VDD
24	S_AD[23]
25	S_CBE_b[3]
26	S_AD[24]

Table 28: Tsi352 160 PQFP Pinlist

Pin Number	Signal Name
27	VSS
28	S_AD[25]
29	S_AD[26]
30	VDD
31	S_AD[27]
32	S_AD[28]
33	S_AD[29]
34	VSS
35	S_AD[30]
36	S_AD[31]
37	S_REQ_b[0]
38	S_REQ_b[1]
39	S_REQ_b[2]
40	VDD
41	VSS
42	S_REQ_b[3]
43	S_GNT_b[0]
44	S_GNT_b[1]
45	S_GNT_b[2]
46	VDD
47	S_GNT_b[3]
48	S_RST_b
49	S_CFN_b
50	VSS
51	S_CLK_IN
52	S_VCCP
53	S_CLKOUT[0]

Table 28: Tsi352 160 PQFP Pinlist

Pin Number	Signal Name
54	VSS
55	S_CLKOUT[1]
56	VDD
57	S_CLKOUT[2]
58	VSS
59	S_CLKOUT[3]
60	VDD
61	S_CLKOUT[4]
62	SCAN_EN/HS_LED
63	SCAN_TM_b
64	P_RST_b
65	VSS
66	P_CLK
67	P_VCCP
68	P_GNT_b
69	P_REQ_b
70	P_AD[31]
71	VSS
72	P_AD[30]
73	P_AD[29]
74	P_AD[28]
75	VDD
76	P_AD[27]
77	P_AD[26]
78	P_AD[25]
79	P_AD[24]
80	VDD

Table 28: Tsi352 160 PQFP Pinlist

Pin Number	Signal Name
81	VSS
82	P_CBE_b[3]
83	P_IDSEL
84	P_AD[23]
85	P_AD[22]
86	VSS
87	P_AD[21]
88	P_AD[20]
89	P_AD[19]
90	VDD
91	P_AD[18]
92	P_AD[17]
93	P_AD[16]
94	VSS
95	P_CBE_b[2]
96	P_FRAME_b
97	P_IRDY_b
98	VDD
99	P_TRDY_b
100	P_DEVSEL_b
101	P_STOP_b
102	P_MFUNC
103	VSS
104	P_PERR_b
105	P_SERR_b
106	P_PAR
107	P_CBE_b[1]

Table 28: Tsi352 160 PQFP Pinlist

Pin Number	Signal Name
108	VDD
109	P_AD[15]
110	P_AD[14]
111	P_AD[13]
112	VSS
113	P_AD[12]
114	P_AD[11]
115	P_AD[10]
116	VDD
117	P_AD[9]
118	P_AD[8]
119	VSS
120	MS0
121	VSS
122	P_CBE_b[0]
123	P_AD[7]
124	P_AD[6]
125	VDD
126	P_AD[5]
127	P_AD[4]
128	VSS
129	P_AD[3]
130	P_AD[2]
131	VDD
132	P_AD[1]
133	P_AD[0]
134	S_AD[0]

Table 28: Tsi352 160 PQFP Pinlist

Pin Number	Signal Name
135	VSS
136	S_AD[1]
137	S_AD[2]
138	S_AD[3]
139	VDD
140	S_AD[4]
141	S_AD[5]
142	S_AD[6]
143	VSS
144	S_AD[7]
145	S_CBE_b[0]
146	S_AD[8]
147	VDD
148	S_AD[9]
149	S_AD[10]
150	S_AD[11]
151	VSS
152	S_AD[12]
153	S_AD[13]
154	VDD
155	S_AD[14]
156	S_AD[15]
157	VSS
158	S_CBE_B[1]
159	MS1/BPCC
160	VDD

13. Register Descriptions

This chapter discusses the following topics about the Tsi352's registers:

13.1 Overview

This chapter provides a detailed description of the Tsi352 registers. The chapter is divided into the following sections: "PCI Configuration Space" on page 122 describes the standard Tsi352 PCI-to-PCI bridge configuration registers, and "Device Specific Register Descriptions" on page 155 describes Tsi352 device-specific configuration registers.

Tsi352 configuration space uses the PCI-to-PCI bridge standard format specified in the *PCI-to- PCI Bridge Architecture Specification*. The header type at configuration address 0x0E reads as 0x01, indicating that this device uses the PCI-to-PCI bridge format.

Tsi352 also contains device-specific registers, starting at address 0x40. Use of these registers is not required for standard PCI-to-PCI bridge implementations.

The configuration space registers can be accessed only from the primary PCI bus. To access a register, perform a Type 0 format configuration read or write operation to that register. During the Type 0 address phase, P_AD[7:2] indicates the Dword offset of the register. During the data phase, P_CBE_b[3:0] selects the bytes in the Dword that is being accessed.

Software changes the configuration register values that affect Tsi352 behavior only during initialization. Change these values subsequently only when both the primary and secondary PCI buses are idle, and the data buffers are empty; otherwise, the behavior of Tsi352 is unpredictable.

13.1.1 Reserved Register Addresses and Fields

Reserved register addresses should not be read or written. Reads to reserved register addresses will return unspecified data. Writes to reserved register addresses may lead to unpredictable results.

Reserved fields within registers return undefined data when read. Reserved fields should always be written as 0 unless noted otherwise.

Table 29 shows the defined register access types.

Table 29: Register Access Types

Abbreviation	Description
R	Read Only. Can be initialized or modified by pin-straps or serial EEPROM.
R/W	Read or Write
R/W1C	Read/Write 1 to clear; writing a 0 has no effect. These register bits are only set by the Tsi352.
RW1CS	Sticky Read Only, Write-1-to-Clear Not initialized or modified by hot reset.
R/WS	Sticky Read / Write Not initialized or modified by hot reset.
R/W1S	Read 0/Write 1 to set (writing a 1 triggers an event such as an interrupt). These register bits are only cleared by the Tsi352
R/W1TR	Read 0/Write 1 to reset
RC	Clear after read
RS	Sticky Read Only. Not initialized or modified by hot reset.
Reserved	Do not write any value other than 0 to this field. Undefined data will be returned when read.
Undefined	This value is undefined after reset because it is based on a bit setting, a pin setting, or a power-up setting.

13.2 Register Map

The following table lists the register map for the Tsi352.

Table 30: Register Map

Offset	Name	Bits	See
0x00	Device ID	31:16	“Device ID Register—Offset 0x00” on page 123
0x00	Vendor ID	15:0	“Vendor ID Register—Offset 0x00” on page 123
0x04	Primary Status Register	31:16	“Primary Status Register—Offset 0x04” on page 127
0x04	Primary Command Register	15:0	“Primary Command Register—Offset 0x04” on page 124
0x08	Base Class Code	31:24	“Base Class Code Register—Offset 0x08” on page 130
0x08	Subclass Code	23:16	“Subclass Code Register—Offset 0x08” on page 130
0x08	Programming Interface Register	15:8	“Programming Interface Register—Offset 0x08” on page 129
0x08	Revision ID	7:0	“Revision ID Register—Offset 0x08” on page 129
0x0C	Reserved	-	-
0x0C	Header Type	23:16	“Header Type Register—Offset 0x0C” on page 133
0x0C	Primary Latency Timer	15:8	“Primary Latency Timer Register—Offset 0x0C” on page 132
0x0C	Cacheline Size	7:0	“Cacheline Size Register—Offset 0x0C” on page 131
0x10-0x14	Reserved	-	-
0x18	Secondary Latency timer	31:24	“Secondary Latency Timer Register—Offset 0x18” on page 136
0x18	Subordinate Bus Number	23:16	“Subordinate Bus Number Register—Offset 0x18” on page 135
0x18	Secondary Bus Number	15:8	“Secondary Bus Number Register—Offset 0x18” on page 134
0x18	Primary Bus Number	7:0	“Primary Bus Number Register—Offset 0x18” on page 134
0x1C	Secondary Status	31:16	“Secondary Status Register—Offset 0x1C” on page 139
0x1C	I/O Limit	15:8	“I/O Limit Address Register—Offset 0x1C” on page 138
0x1C	I/O Base	7:0	“I/O Base Address Register—Offset 0x1C” on page 137
0x20	Memory Limit Address	31:16	“Memory Limit Address Register—Offset 0x20” on page 142
0x20	Memory Base Address	15:0	“Memory Base Address Register—Offset 0x20” on page 141

Table 30: Register Map *(Continued)*

Offset	Name	Bits	See
0x24	Prefetchable Memory Limit	31:16	“Prefetchable Memory Limit Address Register—Offset 0x24” on page 144
0x24	Prefetchable Memory Base	15:0	“Prefetchable Memory Base Address Register—Offset 0x24” on page 143
0x28	Prefetchable Memory Base Upper 32 Bits	31:0	“Prefetchable Memory Base Address Upper 32 Bits Register—Offset 0x28” on page 145
0x2C	Prefetchable Memory Limit Upper 32 Bits	31:0	“Prefetchable Memory Limit Address Upper 32 Bits Register—Offset 0x2C” on page 146
0x30	I/O Limit Upper 16 Bits	31:16	“I/O Limit Address Upper 16 Bits Register—Offset 0x30” on page 148
0x30	I/O Base Upper 16 Bits	15:0	“I/O Base Address Upper 16 Bits Register—Offset 0x30” on page 147
0x34	Reserved	-	-
0x34	ECP Pointer	7:0	“ECP Pointer Register—Offset 0x34” on page 149
0x38	Reserved	-	-
0x3C	Bridge Control	31:16	“Bridge Control Register—Offset 0x3C” on page 151
0x3C	Interrupt Pin	15:8	“Interrupt Pin Register—Offset 0x3C” on page 150
0x3C	Interrupt Line	7:0	“Interrupt Line Register – Offset 0x3C” on page 150
0x40	Arbiter Control	31:16	“Arbiter Control Register—Offset 0x40” on page 157
0x40	Diagnostic Control	15:8	
0x40	Chip Control	7:0	“Chip Control Register/Diagnostic Control — Offset 0x40” on page 155
0x44	Reserved	-	-
0x48	Reserved	-	-
0x48	Memory Read Control	7:0	“Memory Read Control Register — Offset 0x48” on page 158
0x4C-0x54	Reserved	-	-
0x58	CLKRUN	31:24	“CLKRUN Register — Offset 0x58” on page 162
0x58	Miscellaneous Control	23:16	“Miscellaneous Control Register — Offset 0x58” on page 161
0x58	Port Option	15:0	“Port Option Register — Offset 0x58” on page 159
0x5C-0x60	Reserved	-	-

Table 30: Register Map *(Continued)*

Offset	Name	Bits	See
0x64	Reserved	-	-
0x64	P_SERR_b Event Disable	7:0	“P_SERR_b Event Enable Register—Offset 0x64” on page 163
0x68	Reserved	-	
0x68	P_SERR_b Status	23:16	“P_SERR_b Status Register — Offset 0x68” on page 166
0x68	Secondary Clock Control	15:0	“Secondary Clock Control Register—Offset 0x68” on page 165
0x6C-0x80	Reserved	-	-
0x84-0xB0	Reserved	-	-
0xB4-0xD8	Reserved	-	-
0xDC	Power Management Capabilities	31:16	“Power Management Capabilities Register—Offset 0xDC” on page 169
0xDC	Next Item Pointer	15:8	“Next Item Pointer Register—Offset 0xDC” on page 168
0xDC	Capability ID	7:0	“ Capability ID Register—Offset 0xDC” on page 167
0xE0	Reserved	-	-
0xE0	PMCSR_BSE	23:16	“PMCSR_BSE Register — Offset 0xE0” on page 171
0xE0	Power Management CSR	15:0	“Power Management Data Register—Offset 0xE0” on page 170
0xE4	Reserved	-	-
0xE4	Hot Swap Control Status	23:16	“HS Control Status Register — Offset 0xE4” on page 173
0xE4	HS Next Item Pointer	15:8	“HS Next Item Pointer Register — Offset 0xE4” on page 172
0xE4	HS Capability ID	7:0	“HS Capability ID Register — Offset 0xE4” on page 172
0xE8-0xFF	Reserved	-	-

13.3 PCI Configuration Space

The Tsi352 configuration space uses the PCI-to-PCI bridge standard format specified in the *PCI-to-PCI Bridge Architecture Specification.* The header type at configuration address 0x0E reads as 0x01, indicating that this device uses the PCI-to-PCI bridge format.

Tsi352 also contains device-specific registers, starting at address 0x40. Use of these registers is not required for standard PCI-to-PCI bridge implementations.

13.3.1 Accessing Configuration Space Registers

The configuration space registers can be accessed only from the primary PCI bus. To access a register, perform a Type 0 format configuration read or write operation to that register. During the Type 0 address phase, P_AD[7:2] indicates the Dword offset of the register. During the data phase, P_CBE_b[3:0] selects the bytes in the Dword that is being accessed.

Software changes the configuration register values that affect the Tsi352 behavior only during initialization. Change these values subsequently only when both the primary and secondary PCI buses are idle, and the data buffers are empty; otherwise, the behavior of the Tsi352 is unpredictable.

13.3.2 PCI-to-PCI Bridge Standard Register Descriptions

The Tsi352 configuration space uses the PCI-to-PCI bridge standard format specified in the *PCI-to-PCI Bridge Architecture Specification.* The header type at configuration address 0x0E reads as 0x01, indicating that this device uses the PCI-to-PCI bridge format.

13.3.3 Vendor ID Register—Offset 0x00

This section describes the vendor ID register.

Byte enable P_CBE_b[3:0] = xx00b

Register name: PCI_VID Reset value: 0x10E3	Register offset: 0x00

Bits	7	6	5	4	3	2	1	0
15:08	VID							
07:00	VID							

Bits	Name	Description	Type	Reset value
15:00	VID	Vendor ID Identifies the vendor of this device	R	0x10E3

13.3.4 Device ID Register—Offset 0x00

This section describes the device ID register.

Byte enable P_CBE_b[3:0] = 00xxb

Register name: PCI_DID Reset value: 0x0352	Register offset: 0x00

Bits	7	6	5	4	3	2	1	0
31:24	DID							
23:16	DID							

Bits	Name	Description	Type	Reset value
31:16	DID	Device ID Identifies this device as the Tsi352	R	0x0352

13.3.5 Primary Command Register—Offset 0x04

This section describes the primary command register.

These bits affect the behavior of the Tsi352 primary interface, except where noted. Some of the bits are repeated in the bridge control register, to act on the secondary interface.

This register must be initialized by configuration software.

Byte enable P_CBE_b[3:0] = xx00b

Register name: PCI_COMD **Reset value: 0x0000**	**Register offset: 0x04**

Bits	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
15:08	Reserved					INT_DIS	MFBBC	SERR_EN
07:00	WAIT	PERESP	VGAPS	MWI_EN	SC	MAST_EN	MS	IOS

Bits	**Name**	**Description**	**Type**	**Reset value**
15:10	Reserved	Reserved. Returns 0 when read.	R	0x0
09	MFBBC	Reads as 0 to indicate that Tsi352 does not generate fast back-to-back transactions on the primary bus.	R	0x0
08	SERR_EN	SERR_b Enable Controls the enable for P_SERR_b on the primary interface. 0 = Signal P_SERR_b cannot be driven by Tsi352. 1 = Signal P_SERR_b can be driven low by Tsi352 under the conditions described in “System Error (SERR_b) Reporting” on page 77. Reset value: 0.	R/W	0x0
07	WAIT	Wait Cycle Control Reads as 0 to indicate that Tsi352 does not perform address or data stepping.	R	0x0
06	PERESP	Parity Error Response Controls Tsi352’s response when a parity error is detected on the primary interface. 0 = Tsi352 does not assert P_PERR_b, nor does it set the data parity reported bit in the status register. Tsi352 does not report address parity errors by asserting P_SERR_b. 1 = Tsi352 drives P_PERR_b and conditionally sets the data parity reported bit in the status register when a data parity error is detected. Tsi352 allows P_SERR_b assertion when address parity errors are detected on the primary interface.	R/W	0x0

Bits	Name	Description	Type	Reset value
05	VGAPS	VGA Snoop Enable Controls Tsi352's response to VGA compatible palette write transactions. VGA palette write transactions correspond to I/O transactions whose address bits are as follows: • P_AD[9:0] are equal to 3C6h, 3C8h, and 3C9h. • P_AD[15:10] are not decoded. • P_AD[31:16] must be 0. 0 = VGA palette write transactions on the primary interface are ignored unless they fall inside Tsi352's I/O address range. 1 = VGA palette write transactions on the primary interface are positively decoded and forwarded to the secondary interface.	R/W	0x0
04	MWI_EN	Memory Write and Invalidate Enable Tsi352 generates memory write and invalidate transactions only when operating on behalf of another master whose memory write and invalidate transaction is crossing Tsi352. This bit is read only and returns 0.	R	0x0
03	SC	Special Cycle Enable Tsi352 ignores special cycle transactions. This bit is read only and returns 0.	R	0x0
02	MAST_EN	Master Enable Controls Tsi352's ability to initiate memory and I/O transactions on the primary bus on behalf of an initiator on the secondary bus. Forwarding of configuration transactions is not affected. 0 = Tsi352 does not respond to I/O or memory transactions on the secondary interface and does not initiate I/O or memory transactions on the primary interface. 1 = Tsi352 is enabled to operate as an initiator on the primary bus and responds to I/O and memory transactions initiated on the secondary bus.	R/W	0x0
01	MS	Memory Space Enable Controls Tsi352's response to memory transactions on Tsi352 primary interface. 0 = Tsi352 does not respond to memory transactions initiated on the primary bus. 1 = Tsi352 response to memory transactions initiated on the primary bus is enabled.	R/W	0x0

Bits	Name	Description	Type	Reset value
00	IOS	I/O Space Enable Controls Tsi352's response to I/O transactions on the primary interface. 0 = Tsi352 does not respond to I/O transactions initiated on the primary bus. 1 = Tsi352 response to I/O transactions initiated on the primary bus is enabled.	R/W	0x0

13.3.6 Primary Status Register—Offset 0x04

This section describes the primary status register.

These bits affect the status of Tsi352 primary interface. Bits reflecting the status of the secondary interface are found in the secondary status register. W1TC indicates that writing 1 to a bit sets that bit to 0. Writing 0 has no effect.

Byte enable P_CBE_b[3:0] = 00xxb

Register name: PCI_STAT **Reset value: 0x02B0**	**Register offset: 0x04**

Bits	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
31:24	D_PE	S_SERR	R_MA	R_TA	S_TA	DEVSEL		MDP_D
23:16	TFBBC	Reserved	DEV66	ECP	Reserved			

Bits	**Name**	**Description**	**Type**	**Reset value**
31	D_PE	Detected Parity Error This bit is set to 1 when Tsi352 detects an address or data parity error on the primary interface.	R/W1TC	0x0
30	S_SERR	Signaled System Error This bit is set to 1 when Tsi352 has asserted P_SERR_b.	R/W1TC	0x0
29	R_MA	Received Master-Abort This bit is set to 1 when Tsi352 is acting as a master on the primary bus and receives a master abort.	R/W1TC	0x0
28	R_TA	Received Target-Abort This bit is set to 1 when Tsi352 is acting as a master on the primary bus and receives a target abort from the primary target.	R/W1TC	0x0
27	S_TA	Signaled Target-Abort This bit is set to 1 when Tsi352 is acting as a target on the primary bus and returns a target abort to the primary master.	R/W1TC	0x0
26:25	DEVSEL	DEVSEL_b Timing Indicates slowest response to a non-configuration command on the primary interface. Indicates that Tsi352 responds no slower than with medium timing.	R	0x1

Bits	Name	Description	Type	Reset value
24	DP_D	Data Parity Detected This bit is set to 1 when all of the following are true: • Tsi352 is a master on the primary bus. • Signal P_PERR_b is detected asserted, or a parity error is detected on the primary bus. • The parity error response bit is set in the command register.	R/W1TC	0x0
23	TFBBC	Fast Back-to-Back Capable Set to 1 to indicate that Tsi352 is able to respond to fast back-to-back transactions on the primary interface.	R	0x1
22	Reserved	Reserved. Returns 0 when read.	R	0x0
21	DEV66	66-MHz Capable Set to 1 to indicates that primary interface is 66- MHz capable.	R	0x1
20	ECP	Enhanced Capabilities Port (ECP) Enable. Reads as 1 in Tsi352 to indicate that Tsi352 supports an enhanced capabilities list.	R	0x1
16:19	Reserved	Reserved. Returns 0 when read.	R	0x0

13.3.7 Revision ID Register—Offset 0x08

This section describes the revision ID register.

Byte enable P_CBE_b[3:0] = xxx0b

Register name: PCI_RID Reset value: 0x00	Register offset: 0x08

Bits	7	6	5	4	3	2	1	0
07:00	RID							

Bit	Name	Description	Type	Reset Value
07:0	RID	Revision ID Indicates the revision number of this device.	R	0x0

13.3.8 Programming Interface Register—Offset 0x08

This section describes the programming interface register.

Byte enable P_CBE_b[3:0] = xx0xb

Register name: PCI_PROG Reset value: 0x00	Register offset: 0x08

Bits	7	6	5	4	3	2	1	0
15:08	PROG							

Bit	Name	Description	Type	Reset Value
15:08	PROG	Programming Interface This field is not applicable for a bridge. It always reads 0.	R	0x0

13.3.9 Subclass Code Register—Offset 0x08

This section describes the subclass code register.

Byte enable P_CBE_b[3:0] = x0xxb

Register name: PCI_SUB Reset value: 0x04	Register offset: 0x08

Bits	7	6	5	4	3	2	1	0
23:16	SUB							

Bit	Name	Description	R/W	Reset Value
23:16	SUB	Subclass Code This field indicates the device is a PCI-to-PCI bridge.	R	0x04

13.3.10 Base Class Code Register—Offset 0x08

This section describes the base class code register.

Byte enable P_CBE_b[3:0] = 0xxxb

Register name: PCI_BASE Reset value: 0x06	Register offset: 0x08

Bits	7	6	5	4	3	2	1	0
23:16	Base							

Bit	Name	Description	Type	Reset Value
23:16	BASE	Base Class Code This field indicates the device is a bridge.	R	0x06

13.3.11 Cacheline Size Register—Offset 0x0C

This section describes the cacheline size register.

Byte enable P_CBE_b[3:0] = xxx0b

Register name: PCI_CLINE Reset value: 0x00	Register offset: 0x0C

Bits	7	6	5	4	3	2	1	0
07:00	CLINE							

Bit	Name	Description	Type	Reset Value
07:00	CLINE	Cacheline Size This field designates the cache line size for the system in units of Dwords (32-bits). It is used for prefetching memory read transactions and for terminating memory write and invalidate transactions. The cache line size should be written as a power of 2. If the value is not a power of 2, Tsi352 behaves as if the cache line size were 16 Dwords. Notes: • The supported values for prefetch are 1,2,4,8,16,32, and 64. 128 is not supported because maximum prefetch count value is 64 (see register 0x48). • Based on the prefetch count formula in section 2.2.4.2, the prefetch count is no less than 16 dwords for any combination of the cache line size and maximum prefetch count (see register 0x48). • For MWI, Tsi352 only supports 1,2,4,8, and 16.	R/W	0x0

13.3.12 Primary Latency Timer Register—Offset 0x0C

This section describes the primary latency timer register.

Byte enable P_CBE_b[3:0] = xx0xb

Register name: PCI_LAT_IMER **Reset value: 0x00**	**Register offset: 0x0C**

Bits	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
15:08	LAT_TIMER					Reserved		

Bit	Name	Description	Type	Reset Value
15:11	LAT_TIMER	Master Latency Timer for the Primary Interface Indicates the number of PCI clock cycles from the assertion of P_FRAME_b to the expiration of the timer when Tsi352 is acting as a master on the primary interface. All bits are writable, resulting in a granularity of eight PCI clock cycle. 0 = Tsi352 relinquishes the bus after the first data transfer when Tsi352's primary bus grant has been deasserted, with the exception of memory write and invalidate transactions.	R/W	0x0
10:08	Reserved	Set to 0 to force 8-cycle increments for the latency timer.	R	0x0

13.3.13 Header Type Register—Offset 0x0C

This section describes the header type register.

Byte enable P_CBE_b[3:0] = x0xxb

Register name: PCI_H_TYPE Reset value: 0x01	Register offset: 0x0C

Bits	7	6	5	4	3	2	1	0
31:24	Reserved							
23:16	H_TYPE							

Bit	Name	Description	Type	Reset Value
31:24	Reserved	Reserved. Returns 0 when read.	R	0x0
23:16	H_TYPE	Header Type This field indicates that the Tsi352 device is a single-function bridge. Defines the layout of addresses 0x10 through 0x3F in configuration space. Reads as 0x01 to indicate that the register layout conforms to the standard PCI-to-PCI bridge layout.	R	0x1

13.3.14 Primary Bus Number Register—Offset 0x18

This section describes the primary bus number register.This register must be initialized by configuration software.

Byte enable P_CBE_b[3:0] = xxx0b

Register name: PCI_P_BUS_NUM Reset value: 0x00	Register offset: 0x18

Bits	7	6	5	4	3	2	1	0
07:00	P_BUS_NUM							

Bit	Name	Description	Type	Reset Value
07:00	P_BUS_NUM	Primary Bus Number Indicates the number of the PCI bus to which the primary interface is connected. Tsi352 uses this register to decode Type 1 configuration transactions on the secondary interface that should either be converted to special cycle transactions on the primary interface or passed upstream unaltered.	R/W	0x0

13.3.15 Secondary Bus Number Register—Offset 0x18

This section describes the secondary bus number register. This register must be initialized by configuration software.

Byte enable P_CBE_b[3:0]= xx0xb

Register name: PCI_S_BUS_NUM Reset value: 0x00	Register offset: 0x18

Bits	7	6	5	4	3	2	1	0
15:08	S_BUS_NUM							

Bit	Name	Description	R/W	Reset Value
15:08	S_BUS_NUM	Secondary Bus Number Indicates the number of the PCI bus to which the secondary interface is connected. Tsi352 uses this register to determine when to respond to and forward Type 1 configuration transactions on the primary interface, and to determine when to convert them to Type 0 or special cycle transactions on the secondary interface.	R/W	0x0

13.3.16 Subordinate Bus Number Register—Offset 0x18

This section describes the subordinate bus number register. This register must be initialized by configuration software.

Byte enable P_CBE_b[3:0]= x0xxb

Register name: PCI_SUB_BUS_NUM **Reset value: 0x00**	**Register offset: 0x18**

Bits	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
23:16	SUB_BUS_NUM							

Bit	Name	Description	Type	Reset Value
23:16	SUB_BUS_NUM	Subordinate Bus Number Indicates the number of the highest numbered PCI bus that is behind (or subordinate to) Tsi352. Used in conjunction with the secondary bus number to determine when to respond to Type 1 configuration transactions on the primary interface and pass them to the secondary interface as a Type 1 configuration transaction.	R/W	0x0

13.3.17 Secondary Latency Timer Register—Offset 0x18

This section describes the secondary latency timer register.

Byte enable P_CBE_b[3:0] = 0xxxb

Register name: PCI_S_LTIMER **Reset value: 0x00**	**Register offset: 0x18**

Bits	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
31:24	S_LTIMER					Reserved		

Bit	**Name**	**Description**	**Type**	**Reset Value**
31:27	S_LTIMER	Secondary latency timer This field is the master latency timer for the secondary interface. It indicates the number of PCI clock cycles from the assertion of S_FRAME_b to the expiration of the timer when Tsi352 is acting as a master on the secondary interface. All bits are writable, resulting in a granularity of eight PCI clock cycle. 0 = Tsi352 ends the transaction after the first data transfer when Tsi352's secondary bus grant has been deasserted, with the exception of memory write and invalidate transactions. Reset value: 0.	R/W	0x0
26:24	Reserved	Set to 0 to force 8-cycle increments for the latency timer.	R	0x0

13.3.18 I/O Base Address Register—Offset 0x1C

This section describes the I/O base address register. This register must be initialized by configuration software.

Byte enable P_CBE_b[3:0] = xxx0b

Register name: PCI_BA **Reset value: 0x01**	**Register offset: 0x1C**

Bits	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
07:00	IO_BA				ADD_CAP2			

Bit	Name	Description	Type	Reset Value
07:04	IO_BA	I/O Base Address [15:12] Defines the bottom address of an address range used by Tsi352 to determine when to forward I/O transactions from one interface to the other. The upper 4 bits are write-able and correspond to address bits [15:12]. The lower 12 bits of the address are assumed 0. The upper 16 bits corresponding to address bits [31:16] are defined in the I/O base address upper 16 bits register. The I/O address range adheres to 4 KB alignment and granularity.	R/W	0x0
03:00	ADD_CAP2	Addressing Capability Bridge supports 32-bit I/O address decoding.	R	0x1

13.3.19 I/O Limit Address Register—Offset 0x1C

This section describes the I/O limit address register. This register must be initialized by configuration software.

Byte enable P_CBE_b[3:0] = xx0xb

Register name: PCI_LBA **Reset value: 0x01**	**Register offset: 0x1C**

Bits	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
15:08	IO_LA				ADD_CAP1			

Bit	Name	Description	R/W	Reset Value
15:12	IO_LA	I/O Limit Address Defines the top address of an address range used by Tsi352 to determine when to forward I/O transactions from one interface to the other. The upper 4 bits are writable and correspond to address bits [15:12]. The lower 12 bits of the address are assumed 0. The upper 16 bits corresponding to address bits [31:16] are defined in the I/O base address upper 16 bits register. The I/O address range adheres to 4 KB alignment and granularity. Reset value: 0.	R/W	0x0
11:8	ADD_CAP1	Addressing Capability Bridge supports 32-bit I/O address decoding.	R	0x1

13.3.20 Secondary Status Register—Offset 0x1C

This section describes the secondary status register.

These bits reflect the status of Tsi352 secondary interface. W1TC indicates that writing 1 to that bit sets the bit to 0. Writing 0 has no effect.

Byte enable P_CBE_b[3:0] = 00xxb

Register name: PCI_SEC_STAT **Reset value: 0x02A0**	**Register offset: 0x1C**

Bits	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
31:24	D_PE	S_SERR	R_MA	R_TA	S_TA	DEVSEL		MDP_D
23:16	TFBBC	Reserved	DEV66	ECP	Reserved			

Bit	**Name**	**Description**	**Type**	**Reset Value**
31	D_PE	Detected Parity Error This bit is set to 1 when Tsi352 detects an address or data parity error on the secondary interface.	R/W1TC	0x0
30	S_SERR	Received System Error This bit is set to 1 when Tsi352 detects the assertion of S_SERR_b on the secondary interface.	R/W1TC	0x0
29	R_MA	Received Master Abort This bit is set to 1 when Tsi352 is acting as an initiator on the secondary bus and receives a master abort.	R/W1TC	0x0
28	R_TA	Received target Abort This bit is set to 1 when Tsi352 is acting as a master on the secondary bus and receives a target abort from the secondary bus target.	R/W1TC	0x0
27	S_TA	Signaled Target Abort This bit is set to 1 when Tsi352 is acting as a target on the secondary bus and returns a target abort to the secondary bus master.	R/W1TC	0x0
26:25	DEVSEL	DEVSEL_b Timing Indicates slowest response to a Non configuration command on the secondary interface. Indicates that Tsi352 responds no slower than with medium timing.	R	0x1

Bit	Name	Description	Type	Reset Value
24	MDP_D	Data Parity Detected This bit is set to 1 when all of the following are true: • Tsi352 is a master on the secondary bus. • Signal S_PERR_b is detected asserted, or a parity error is detected on the primary bus. • The parity error response bit is set in the bridge control register.	R/W1TC	0x0
23	TFBBC	Fast Back-to-Back Capable Indicate that Tsi352 is able to respond to fast back-to-back transactions on the secondary interface.	R	0x1
22	Reserved	Reserved. Returns 0 when read.	R	0x0
21	DEV66	66-MHz Capable Indicates that secondary interface is 66- MHz capable.	R	0x1
20	ECP	Enhanced Capabilities Port (ECP) Enable Indicates that Tsi352 does not support an enhanced capabilities list.	R	0x0
19:16	Reserved	Reserved. Returns 0 when read.	R	0x0

13.3.21 Memory Base Address Register—Offset 0x20

This section describes the memory base address register. This register must be initialized by configuration software.

Byte enable P_CBE_b[3:0] = xx00b

Register name: PCI_MIO_BA **Reset value: 0x0000**	**Register offset: 0x20**

Bits	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
15:08	BA							
07:00	BA					Reserved		

Bits	**Name**	**Description**	**Type**	**Reset value**
15:04	BA	Memory Base Address This register is used to define the lower bound of the address range for forwarding memory-mapped I/O transactions across the bridge. These bits correspond to address bits [31:20] of the address range. The lower 20 address bits (19:0) are 20'h0. The memory address range adheres to 1MB alignment and granularity.	R/W	0
03:00	Reserved	Reserved	R	0

13.3.22 Memory Limit Address Register—Offset 0x20

This section describes the memory limit address register. This register must be initialized by configuration software.

Byte enable P_CBE_b[3:0] = 00xxb

Register name: PCI_MIO_LA **Reset value: 0x0000**	**Register offset: 0x20**

Bits	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
31:24	LA							
23:16	LA					Reserved		

Bits	**Name**	**Description**	**Type**	**Reset value**
31:20	LA	Memory Limit Address This register is used in conjunction with Memory Base Address register for forwarding memory-mapped I/O transactions. These bits define the upper bound for the memory address range. The upper twelve bits corresponding to address bits [31:20] of the address range. Bits [19:0] of the address range are 0xFFFFF. The memory address range adheres to 1MB alignment and granularity.	R/W	0
19:16	Reserved	Reserved	R	0

13.3.23 Prefetchable Memory Base Address Register—Offset 0x24

This section describes the prefetchable memory base address register. This register must be initialized by configuration software.

Byte enable P_CBE_b[3:0] = xx00b

Register name: PCI_MIO_PM_BA **Reset value: 0x0001**	**Register offset: 0x24**

Bits	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
15:08	PM_BA							
07:00	PM_BA				ADD_BA_64			

Bit	Name	Description	Type	Reset Value
15:4	PM_BA	Prefetchable memory base address [31:20] Defines the bottom address of an address range used by Tsi352 to determine when to forward memory read and write transactions from one interface to the other. The upper 12 bits are writable and correspond to address bits [31:20]. The lower 20 bits of the address are assumed 0. The memory base register upper 32 bits contain the upper half of the base address. The memory address range adheres to 1MB alignment and granularity. Reset value: 0.	R/W	0x0
3:0	ADD_BA_64	Addressing Capability — 64-bit indicator Bridge supports 64-bit addressing.	R	0x1

13.3.24 Prefetchable Memory Limit Address Register—Offset 0x24

This section describes the prefetchable memory limit address register. This register must be initialized by configuration software.

Byte enable P_CBE_b[3:0] = 00xxb

Register name: PCI_MIO_PM_LA **Reset value: 0x0001**	**Register offset: 0x24**

Bits	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
31:24	PM_LA							
23:16	PM_LA				ADD_LA_64			

Bit	**Name**	**Description**	**Type**	**Reset Value**
31:20	PM_LA	Prefetchable memory limit address [31:20] Defines the top address of an address range used by Tsi352 to determine when to forward memory read and write transactions from one interface to the other. The upper 12 bits are writable and correspond to address bits [31:20]. The lower 20 bits of the address are assumed to be 0xFFFFF. The memory limit upper 32 bits register contains the upper half of the limit address. The memory address range adheres to 1MB alignment and granularity.	R/W	0x0
19:16	ADD_LA_64	64-bit indicator Bridge supports 64-bit addressing.	R	0x1

13.3.25 Prefetchable Memory Base Address Upper 32 Bits Register—Offset 0x28

This section describes the prefetchable memory base address upper 32 bits register. This register must be initialized by configuration software.

Byte enable P_CBE_b[3:0]= 0000b

Register name: PCI_PM_BA_UPPER **Reset value: 0x0000_0000**	**Register offset: 0x28**

Bits	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
31:24	PM_BA_UPPER							
23:16	PM_BA_UPPER							
15:08	PM_BA_UPPER							
07:00	PM_BA_UPPER							

Bit	**Name**	**Description**	**Type**	**Reset Value**
31:00	PM_BA_UPPER	Upper 32 Prefetchable Memory Base Address [63:32] Defines the upper 32 bits of a 64-bit bottom address of an address range used by Tsi352 to determine when to forward memory read and write transactions from one interface to the other. The 32 bits relate to address bits [63:32] of the Prefetchable Base Address bits. The memory address range adheres to 1MB alignment and granularity.	R/W	0x0

13.3.26 Prefetchable Memory Limit Address Upper 32 Bits Register—Offset 0x2C

This section describes the prefetchable memory limit address upper 32 bits register. This register must be initialized by configuration software.

Byte enable P_CBE_b[3:0] = 0000b

Register name: PCI_PM_LA_UPPER **Reset value: 0x0000_0000**	**Register offset: 0x2C**

Bits	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
31:24	PM_LA_UPPER							
23:16	PM_LA_UPPER							
15:08	PM_LA_UPPER							
07:00	PM_LA_UPPER							

Bit	**Name**	**Description**	**Type**	**Reset Value**
31:00	PM_LA_UPPER	Upper 32 prefetchable memory limit address [63:32] Defines the upper 32 bits of a 64-bit top address of an address range used by the Tsi352 to determine when to forward memory read and write transactions from one interface to the other. Extra read transactions should have no side effects. The memory address range adheres to 1MB alignment and granularity.	R/W	0x0

13.3.27 I/O Base Address Upper 16 Bits Register—Offset 0x30

This section describes the I/O base address upper 16 bits register. This register must be initialized by configuration software.

Byte enable P_CBE_b[3:0] = xx00b

Register name: PCI_IO_BA_UPPER **Reset value: 0x0000**	**Register offset: 0x30**

Bits	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
15:08	IO_BA_UPPER							
07:00	IO_BA_UPPER							

Bit	**Name**	**Description**	**Type**	**Reset Value**
15:00	IO_BA_UPPER	I/O Base Address Upper 16-bits [31:16] Defines the upper 16 bits of a 32-bit bottom address of an address range used by Tsi352 to determine when to forward I/O transactions from one interface to the other. The I/O address range adheres to 4KB alignment and granularity.	R/W	0x0

13.3.28 I/O Limit Address Upper 16 Bits Register—Offset 0x30

This section describes the I/O limit address upper 16 bits register. This register must be initialized by configuration software.

Byte enable P_CBE_b[3:0] = 00xxb

Register name: PCI_IO_LA_UPPER **Reset value: 0x0000**	**Register offset: 0x30**

Bits	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
31:24	IO_LA_UPPER							
23:16	IO_LA_UPPER							

Bit	**Name**	**Description**	**Type**	**Reset Value**
31:16	IO_LA_UPPER	I/O Limit Address Upper 16-bits [31:16] Defines the upper 16 bits of a 32-bit top address of an address range used by Tsi352 to determine when to forward I/O transactions from one interface to the other. The I/O address range adheres to 4KB alignment and granularity.	R/W	0x0

13.3.29 ECP Pointer Register—Offset 0x34

This section describes the ECP pointer register.

Byte enable P_CBE_b[3:0] = 0000b

Register name: PCI_ECP **Reset value: 0x0000_00DC**	**Register offset: 0x34**

Bits	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
31:24	Reserved							
23:16	Reserved							
15:08	Reserved							
07:00	ECP_PTR							

Bit	**Name**	**Description**	**Type**	**Reset Value**
31:8	Reserved	Reserved. Return 0 when read.	R	0x0
7:0	ECP_PTR	Enhanced Capabilities Port (ECP) offset pointer. Reads as 0xDC in Tsi352 to indicate that the first item, which corresponds to the power management registers, resides at that configuration offset.	R	0xDC

13.3.30 Interrupt Line Register – Offset 0x3C

This section describes the interrupt line register.

Byte enable P_CBE_b[3:0] = xxx0b

Register name: PCI_INT_LINE Reset value: 0x00FF	Register offset: 0x3C

Bits	7	6	5	4	3	2	1	0
07:00	INT_LINE							

Bits	Name	Description	Type	Reset value
07:00	INT_LINE [7:0]	Interrupt Line The Tsi352 does not generate an interrupt. Therefore, the register value is written to 0xFF by initialization software.	R/W	0xFF

13.3.31 Interrupt Pin Register—Offset 0x3C

This section describes the interrupt pin register.

Byte enable P_CBE_b[3:0] = xx0xb

Register name: PCI_INT_PIN Reset value: 0x0000	Register offset: 0x3C

Bits	7	6	5	4	3	2	1	0
15:08	INT_PIN							

Bits	Name	Description	Type	Reset value
15:08	INT_PIN [7:0]	Interrupt Pin The bridge does not generate interrupts. Therefore, this register is hardwired to 0x00.	R	0x00

13.3.32 Bridge Control Register—Offset 0x3C

This section describes the bridge control register. This register must be initialized by configuration software.

Byte enable P_CBE_b[3:0]= 00xxb

Register name: PCI_BRIDGE_CONT **Reset value: 0x0000_0000**	**Register offset: 0x3C**

Bits	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
31:24	Reserved				TIMEOUT_ SERR_EN	TIMEOUT_ STATUS	S_TIMEOU T	P_TIMEOU T
23:16	S_FPTP_ EN	S_RESET	MA_ERR	Reserved	VGA_EN	ISA_EN	SERR_EN	S_ PERESP

Bit	**Name**	**Description**	**Type**	**Reset Value**
31:28	Reserved	Reserved. Returns 0 when read.	R	0x0
27	TIMEOUT_SERR_E N	Master Timeout SERR_b Enable Controls assertion of P_SERR_b during a master timeout. 0 = P_SERR_b is not asserted as a result of a master timeout. 1 = P_SERR_b is asserted when either the primary master timeout counter or the secondary master timeout counter expires and a delayed transaction is discarded from Tsi352's queues. The SERR_b enable bit in the command register must also be set.	R/W	0x0
26	TIMEOUT_STATUS	Master Timeout status This bit is set to 1 when either the primary master timeout counter or the secondary master timeout counter expires and a delayed transaction is discarded from Tsi352's queues.	R/W1TC	0x0
25	S_TIMEOUT	Secondary Master Timeout Sets the maximum number of PCI clock cycles that Tsi352 waits for an initiator on the secondary bus to repeat a delayed transaction request. The counter starts once the delayed transaction completion is at the head of the queue. If the master has not repeated the transaction at least once before the counter expires, Tsi352 discards the transaction from its queues. 0 = The primary master timeout value is 2^15 PCI clock cycles. 1= The value is 2^10 PCI clock cycles.	R/W	0x0

Bit	Name	Description	Type	Reset Value
24	P_TIMEOUT	Primary Master Timeout Sets the maximum number of PCI clock cycles that Tsi352 waits for an initiator on the primary bus to repeat a delayed transaction request. The counter starts once the delayed transaction completion is at the head of the queue. If the master has not repeated the transaction at least once before the counter expires, Tsi352 discards the transaction from its queues. 0 = The primary master timeout value is 2^15 PCI clock cycles. 1 = The value is 2^10 PCI clock.	R/W	0x0
23	S_FPTP_EN	Secondary Bus Fast Back-to-back Enable Indicates that Tsi352 does not generate fast back-to-back transactions on the secondary bus	R	0x0
22	S_RESET	Secondary Bus Reset Controls S_RST_b on the secondary interface. 0 = Tsi352 deasserts S_RST_b. 1 = Tsi352 asserts S_RST_b. When S_RST_b is asserted, the data buffers and the secondary interface are initialized back to reset conditions. The primary interface and configuration registers are not affected by the assertion of S_RST_b.	R/W	0x0
21	MA_ERR	Master Abort Mode Controls Tsi352's behavior when a master abort termination occurs in response to a transaction initiated by Tsi352 on either the primary or secondary PCI interface. 0 = Tsi352 asserts TRDY_b on the initiator bus for delayed transactions, and 0xFFFF FFFF for read transactions. For posted write transactions, P_SERR_b is not asserted. 1 = Tsi352 returns a target abort on the initiator bus for delayed transactions. For posted write transactions, Tsi352 asserts P_SERR_b if the SERR_b enable bit is set in the command register. Reset value: 0.	R/W	0x0
20	Reserved	Reserved. Returns 0 when read.	R	0x0

Bit	Name	Description	Type	Reset Value
19	VGA_EN	VGA Enable Modifies Tsi352's response to VGA compatible addresses. 0 = VGA transactions are ignored on the primary bus unless they fall within the I/O base and limit address registers and the ISA mode is 0. 1 = Tsi352 positively decodes and forwards the following transactions downstream, regardless of the values of the I/O base and limit registers, ISA mode bit, or VGA snoop bit: • Memory transactions addressing in the range 0x000A0000 to 0x000BFFFF • I/O transactions addressing in the following ranges: — P_AD[9:0] = 0x3B0–0x3BB and 0xC0–0x3DF — P_AD[15:10] are not decoded. — P_AD[31:16] = 0x0000. I/O and memory space enable bits must be set in the "Primary Command Register—Offset 0x04" on page 124. Note: The listed transactions are ignored by Tsi352 on the secondary bus.	R/W	0x0
18	ISA_EN	ISA enable Modifies Tsi352's response to ISA I/O addresses. Applies only to those addresses falling within the I/O base and limit address registers and within the first 64KB of PCI I/O space. 0 = Tsi352 forwards all I/O transactions downstream that fall within the I/O base and limit address registers. 1 = Tsi352 ignores primary bus I/O transactions within the I/O base and limit address registers and within the first 64KB of PCI I/O space that address the last 768 bytes in each 1KB block. Secondary bus I/O transactions are forwarded upstream if the address falls within the last 768 bytes in each 1KB block.	R/W	0x0
17	SERR_EN	SERR_b Forward Enable Controls whether Tsi352 asserts P_SERR_b when it detects S_SERR_b asserted. 0 = Tsi352 does not drive P_SERR_b in response to S_SERR_b assertion. 1 = Tsi352 asserts P_SERR_b when S_SERR_b is detected asserted (the primary SERR_b driver enable bit must also be set).	R/W	0x0

Bit	Name	Description	Type	Reset Value
16	S_PERESP	Parity Error Response Controls Tsi352's response when a parity error is detected on the secondary interface. 0 = Tsi352 does not assert S_PERR_b, nor does it set the data parity reported bit in the secondary status register. Tsi352 does not report address parity errors by asserting P_SERR_b. 1 = Tsi352 drives S_PERR_b and conditionally sets the data parity reported bit in the secondary status register when a data parity error is detected on the secondary interface (see Section 7.0). Also must be set to 1 to allow P_SERR_b assertion when address parity errors are detected on the secondary interface.	R/W	0x0

13.4 Device Specific Register Descriptions

Tsi352 contains device-specific registers, starting at address 0x40. Use of these registers is not required for standard PCI-to-PCI bridge implementations.

13.4.1 Chip Control Register/Diagnostic Control — Offset 0x40

This section describes the chip control register.

Byte enable P_CBE_b[3:0] = xx00b

Register name: DEV_CONT_DIAG **Reset value: 0x0000**	**Register offset: 0x40**

Bits	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
15:08	Reserved							DEV_ RESET
07:00	Reserved			S_PRE_ DIS	Reserved		MW_DIS	Reserved

Bits	**Name**	**Description**	**Type**	**Reset value**
15:09	Reserved	Reserved. Returns 0 when read.	R	0x0
08	DEV_RESET	Chip and secondary bus reset control. 1 = Causes Tsi352 to perform a chip reset. Data buffers, configuration registers, and both the primary and secondary interfaces are reset to their initial state. Tsi352 clears this bit once chip reset is complete. Tsi352 can then be reconfigured. Secondary bus reset S_RST_b is asserted and the secondary reset bit in the bridge control register is set when this bit is set. The secondary reset bit in the bridge control Register must be cleared in order to de-assert S_RST_b	R/W1TR	0x0
07:05	Reserved	Reserved. Returns 0 when read.	R	0x0
04	S_PRE_DIS	Secondary Bus Prefetch Disable Controls Tsi352's ability to prefetch during upstream memory read transactions. 0 = Tsi352 prefetches and does not forward byte enable bits during memory read transactions. 1 = Tsi352 requests only one Dword from the target during memory read transactions and forwards read byte enable bits. Tsi352 returns a target disconnect to the requesting master on the first data transfer. Memory read line and memory read multiple transactions are still prefetchable.	R/W	0x0
03:02	Reserved	Reserved. Returns 0 when read.	R	0x0

(Continued)

Bits	Name	Description	Type	Reset value
01	MW_DIS	Memory Write Disconnect Control Controls when Tsi352, as a target, disconnects memory write transactions. 0 = Tsi352 disconnects on queue full or on a 4KB boundary. 1 = Tsi352 disconnects on a cacheline boundary, as well as when the queue fills or on a 4KB boundary.	R/W	0x0
00	Reserved	Reserved. Returns 0 when read.	R	0x0

13.4.2 Arbiter Control Register—Offset 0x40

This section describes the arbiter control register.

Byte enableP_CBE_b[3:0] = 00xxb

Register name: DEV_ARB_CONT Reset value: 0x0200	Register offset: 0x40

Bits	7	6	5	4	3	2	1	0
31:24	Reserved						S_PRIO	Reserved
23:16	Reserved				S_ARB_CONT			

Bits	Name	Description	Type	Reset value
31:26	Reserved	Reserved. Returns 0 when read.	R	0x0
25	S_PRIO	Priority of Secondary Interface Controls whether the secondary interface of the bridge is in the high priority group or the low priority group. 0 = Low priority 1 = High priority	R/W	0x1
24:20	Reserved	Reserved. Returns 0 when read.	R	0x0
19:16	S_ARB_CONT	Secondary Arbiter control Each bit controls whether a secondary bus master is assigned to the high priority arbiter group or the low priority arbiter group. Bits [19:16] correspond to request inputs S_REQ_b[3:0], respectively. 0 = Indicates that the master belongs to the low priority group. 1 = Indicates that the master belongs to the high priority group.	R/W	0x0

13.4.3 Memory Read Control Register — Offset 0x48

Based on the formula prefetch count formula, PRE_DW_CNT = (MPC - CLS) + CLB_CNT, the prefetch count is no less than 16 Dword count for any combination of cache line size and maximum prefetch count. For more information, see “Read Transactions” on page 30.

Register name: DEV_MEM_READ_CONT **Reset value: 0x00**	**Register offset: 0x48**

Bits	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
07:00	DOWN_PRE_COUNT		UP_PRE_COUNT		Reserved			FLUSH

Bits	Name	Description	Type	Reset value
07:06	DOWN_PRE_COUNT	Downstream Transaction Maximum Prefetch Count This register bits designates the maximum prefetch count for downstream Memory Read transactions. 00 = 16 Dwords 01 = 32 Dwords 10 = 48 Dwords 11 = 64 Dwords	R/W	0x0
05:04	UP_PRE_COUNT	Upstream Transaction Maximum Prefetch Count This register bits designates the maximum prefetch count for upstream Memory Read transactions. 00 = 16 Dwords 01 = 32 Dwords 10 = 48 Dwords 11 = 64 Dwords	R/W	0x0
03:01	Reserved	Reserved. Returns 0 when read.	R	0x0
00	FLUSH	Non-posted Flush 0 = Tsi352 follows PCI Bridge ordering rules. 1 = Clears the non-posted read completions from the primary interface, in the non-posted buffer, when a posted write is received from primary interface. The read request retry from the secondary device is registered as a new non-posted read request.	R/W	0x0

13.4.4 Port Option Register — Offset 0x58

Register name: DEV_PORT_OPTION Reset value: 0x006A	Register offset: 0x58

Bits	7	6	5	4	3	2	1	0
15:08	Reserved							S_MWI_DIS
07:00	P_MWI_DIS	S_MEMRL_EN	P_MEMRL_EN	Reserved	S_MEMR_EN	Reserved	P_MEMR_EN	Reserved

Bits	Name	Description	Type	Reset value
15:09	Reserved	Reserved. Returns 0 when read.	R	0x0
08	S_MWI_DIS	Secondary Memory Write and Invalidate Command Alias Disable Controls bridge detection mechanism for matching posted memory write and invalidate cycles from the initiator on the secondary interface 0 = When accepting MEMWI command at the secondary interface, bridge converts MEMWI to MEMW command on the destination interface 1 = When accepting MEMWI command at the secondary interface, bridge does not convert MEMWI to MEMW command on the destination interface	R/W	0x0
07	P_MWI_DIS	Primary Memory Write and Invalidate Command Alias Disable Controls bridge detection mechanism for matching posted memory write and invalidate cycles from the initiator on the primary interface 0 = When accepting MEMWI command at the primary interface, bridge converts MEMWI to MEMW command on the destination interface 1 = When accepting MEMWI command at the primary interface, bridge does not convert MEMWI to MEMW command on the destination interface	R/W	0x0
06	S_MEMRL_EN	Secondary Memory Read Line/Multiple Alias Enable Control bridge detection mechanism for matching memory read line/multiple cycles from the initiator on the secondary interface 0 = Exact matching for memory read line/multiple retry cycles from the initiator on the secondary interface 1 = Alias MEMRL to MEMRM or MEMRM to MEMRL for memory read retry cycles from the initiator on the secondary interface	R/W	0x1

(Continued)

Bits	Name	Description	Type	Reset value
05	P_MEMRL_EN	Primary Memory Read Line/Multiple Alias Enable Control bridge detection mechanism for matching memory read line/multiple cycles from the initiator on the primary interface 0 = Exact matching for memory read line/multiple retry cycles from the initiator on the primary interface 1 = Alias MEMRL to MEMRM or MEMRM to MEMRL for memory read retry cycles from the initiator on the primary interface	R/W	0x1
04	Reserved	Reserved. Returns 0 when read.	R	0x0
03	S_MEMR_EN	Secondary Memory Read Command Alias Enable Controls bridge detection mechanism for matching memory read retry cycles from the initiator on the secondary interface 0 = Exact matching memory read retry cycles from initiator on the secondary interface 1 = Alias MEMRL or MEMRM to MEMR for memory read retry cycles from the initiator on the secondary interface	R/W	0x1
02	Reserved	Reserved. Returns 0 when read.	R	0x0
01	P_MEMR_EN	Primary Memory Read Command Alias Enable Controls bridge detection mechanism for matching memory read retry cycles from the initiator on the primary interface. 0 = Exact matching memory read retry cycles from initiator on the primary interface 1 = Alias MEMRL or MEMRM to MEMR for memory read retry cycles from the initiator on the primary interface	R/W	0x1
00	Reserved	Reserved. Returns 0 when read.	R	0x0

13.4.5 Miscellaneous Control Register — Offset 0x58

Register name: DEV_MISC_CNTRL Reset value: 0x00	Register offset: 0x58

Bits	7	6	5	4	3	2	1	0
23:16	Reserved							ISA_EN

Bits	Name	Description	Type	Reset value
23:17	Reserved	Reserved. Returns 0 when read.	R	0x0
16	ISA_EN	Legacy ISA I/O Enable 0 = The following I/O addresses is not claimed by the bridge and is not be forwarded on to the secondary bus 1 = The following I/O addresses is forwarded on to the secondary bus • Game port: 0x0200 - 0x0207 • FM: 0x0388 - 0x038B • Audio: 0x0220 - 0x0233 • MIDI: 0x0330 - 0x0331	R/W	0x0

13.4.6 CLKRUN Register — Offset 0x58

Register name: DEV_CLKRUN Reset value: 0x00	Register offset: 0x58

Bits	7	6	5	4	3	2	1	0
31:24	Reserved			CLK_MODE	P_CLK_EN	P_CLK_STOP	S_CLK_EN	S_CLK_STAT

Bits	Name	Description	Type	Reset value
31:29	Reserved	Reserved. Returns 0 when read.	R	0x0
28	CLK_MODE	CLKRUN Mode 0 = Stop the secondary clock only on request from the primary bus 1 = Stop the secondary clock whenever the secondary bus is idle and there are no requests from the primary bus	R/W	0x0
27	P_CLK_EN	Primary CLKRUN Enable 0 = Disable primary CLKRUN 1 = Enable primary CLKRUN	R/W	0x0
26	P_CLK_STOP	Primary Clock Stop 0 = Allow primary clock to stop if secondary clock is stopped 1 = Always keep primary clock operating	R/W	0x0
25	S_CLK_EN	Secondary CLKRUN Enable 0 = Disable secondary CLKRUN 1 = Enable secondary CLKRUN	R/W	0x0
24	S_CLK_STAT	Secondary Clock Stop Status 0: Secondary clock not stopped 1: Secondary clock stopped.	R	0x0

13.4.7 P_SERR_b Event Enable Register—Offset 0x64

This section describes the P_SERR_b event disable register.

Byte enable P_CBE_b[3:0] = xxx0b

Register name: DEV_SERR_DISABLE Reset value: 0x00	Register offset: 0x64

Bits	7	6	5	4	3	2	1	0
07:00	Reserved	R_ERR_EN	W_DEL_ERR1_EN	W_MA_ERR_EN	W_TA_ERR_EN	W_DEL_ERR_EN	W_PAR_ERR_EN	Reserved

Bits	Name	Description	Type	Reset value
07	Reserved	Reserved. Returns 0 when read.	R	0x0
06	R_ERR_EN	Delayed Read Error (no data from target) Controls the Tsi352's ability to assert P_SERR_b when it is unable to transfer any read data from the target after 2^24 attempts. 0 = Signal P_SERR_b is asserted if this event occurs and the SERR_b enable bit in the command register is set. 1 = Signal P_SERR_b is not asserted if this event occurs.	R/W	0x0
05	W_DEL_ERR1_EN	Delayed Write Non-delivery Controls the Tsi352's ability to assert P_SERR_b when it is unable to deliver delayed write data after 2^24 attempts. 0 = Signal P_SERR_b is asserted if this event occurs and the SERR_b enable bit in the command register is set. 1 = Signal P_SERR_b is not asserted if this event occurs.	R/W	0x0
04	W_MA_ERR_EN	Master Abort on Posted Write Controls the Tsi352's ability to assert P_SERR_b when it receives a master abort when attempting to deliver posted write data. 0= Signal P_SERR_b is asserted if this event occurs and the SERR_b enable bit in the command register is set. 1 = Signal P_SERR_b is not asserted if this event occurs.	R/W	0x0
03	W_TA_ERR_EN	Target Abort During Posted Write Controls the Tsi352's ability to assert P_SERR_b when it receives a target abort when attempting to deliver posted write data. 0 = Signal P_SERR_b is asserted if this event occurs and the SERR_b enable bit in the command register is set. 1 = Signal P_SERR_b is not asserted if this event occurs.	R/W	0x0

(Continued)

Bits	Name	Description	Type	Reset value
02	W_DEL_ERR_EN	Posted Write Non-delivery Controls the Tsi352's ability to assert P_SERR_b when it is unable to deliver posted write data after 2^24 attempts. 0 = Signal P_SERR_b is asserted if this event occurs and the SERR_b enable bit in the command register is set. 1 = Signal P_SERR_b is not asserted if this event occurs.	R/W	0x0
01	W_PAR_ERR_EN	Posted Write Parity Error Controls the Tsi352's ability to assert P_SERR_b when a data parity error is detected on the target bus during a posted write transaction. 0 = Signal P_SERR_b is asserted if this event occurs and the SERR_b enable bit in the command register is set. 1 = Signal P_SERR_b is not asserted if this event occurs.	R/W	0x0
00	Reserved	Reserved. Returns 0 when read.	R	0x0

13.4.8 Secondary Clock Control Register—Offset 0x68

This section describes the secondary clock control register.

Byte enable P_CBE_b[3:0] = xx00b

Register name: DEV_S_CLK_CNTRL Reset value: 0x3E00	Register offset: 0x68

Bits	7	6	5	4	3	2	1	0
15:08	Reserved							S_CLKOUT4_DIS
07:00	S_CLKOUT3_DIS			S_CLKOUT2_DIS		S_CLKOUT1_DIS		S_CLKOUT0_DIS

Bits	Name	Description	Type	Reset value
15:14	Reserved	Reserved. Returns 0 when read.	R	0x0
13:09	Reserved	Reserved	R	0x1F
08	S_CLKOUT4_DIS	S_CLKOUT4 disable 0 = S_CLKOUT4 enabled (default) 1 = S_CLKOUT4 disabled and driven high	R/W	0x0
07:06	S_CLKOUT3_DIS	S_CLKOUT3 disable 00, 01, 10 = S_CLKOUT3 enabled (default) 11 = S_CLKOUT3 disabled and driven high	R/W	0x0
05:04	S_CLKOUT2_DIS	S_CLKOUT2 disable 00, 01, 10 = S_CLKOUT2 enabled (default) 11 = S_CLKOUT2 disabled and driven high	R/W	0x0
03:02	S_CLKOUT1_DIS	S_CLKOUT1 disable 00, 01, 10 = S_CLKOUT1 enabled (default) 11 = S_CLKOUT1 disabled and driven high	R/W	0x0
01:00	S_CLKOUT0_DIS	S_CLKOUT0 disable 00, 01, 10 = S_CLKOUT0 enabled (default) 11 = S_CLKOUT0 disabled and driven high	R/W	0x0

13.4.9 P_SERR_b Status Register — Offset 0x68

This status register indicates the reason for Tsi352's assertion of P_SERR_b.

Byte enable P_CBE_b[3:0] = x0xxb

Register name: DEV_SERR_STAT Reset value: 0x00	Register offset: 0x68

Bits	7	6	5	4	3	2	1	0
23:16	Reserved							ADD_PAR_ERR

Bits	Name	Description	Type	Reset value
16	ADD_PAR_ERR	Address Parity Error 1 = P_SERR_b was asserted because an address parity error was detected on either the primary or secondary PCI bus.	R/W1TC	0x0
17	W_DATA_PAR_ERR	Posted Write Data Parity Error 1= P_SERR_b was asserted because a posted write data parity error was detected on the target bus	R/W1TC	0x0
18	W_DEL_ERR1	Posted Write Non-delivery 1 = P_SERR_b was asserted because the Tsi352 was unable to deliver posted write data to the target after 2^24 attempts.	R/W1TC	0x0
19	W_TA_ERR	Target Abort During Posted Write 1 = P_SERR_b was asserted because the Tsi352 received a target abort when delivering posted write data.	R/W1TC	0x0
20	W_MA_ERR	Master abort during posted write 1 = P_SERR_b was asserted because the Tsi352 received a master abort when attempting to deliver posted write data	R/W1TC	0x0
21	W_DEL_ERR	Delayed Write Non-delivery 1 = P_SERR_b was asserted because the Tsi352 was unable to deliver delayed write data after 2^24attempts.	R/W1TC	0x0
22	R_ERR	Delayed Read Error (no data from target) 1 = P_SERR_b was asserted because the Tsi352 was unable to read any data from the target after 2^24 attempts.	R/W1TC	0x0
23	TIMEOUT_SERR	Delayed Transaction Master Time-out 1 = P_SERR_b was asserted because a master did not repeat a read or write transaction before the master timeout counter expired on the initiator's PCI bus.	R/W1TC	0x0

13.4.10 Capability ID Register—Offset 0xDC

This section describes the capability ID register.

Byte enable P_CBE_b[3:0] = xxx0b

Register name: DEV_CAP_ID Reset value: 0x01	Register offset: 0xDC

Bits	7	6	5	4	3	2	1	0
07:00	CAP_ID							

Bit	Name	Description	Type	Reset Value
07:00	CAP_ID	Enhanced Capabilities ID. Indicates that these are power management enhanced capability registers.	R	0x01

13.4.11 Next Item Pointer Register—Offset 0xDC

This section describes the next item pointer register.

Byte enable P_CBE_b[3:0] = xx0xb

Register name: DEV_NEXT_ITEM Reset value: 0x00	Register offset: 0x0DC

Bits	7	6	5	4	3	2	1	0
15:08	NEXT_ITEM							

Bit	Name	Description	Type	Reset Value
15:08	NEXT_ITEM	Next Item Pointer The next item pointer register is used to indicate the next item in the linked list of PCI power management capabilities. The next item pointer returns the following: 0xE4 = CompactPCI mode, indicating that the Tsi352 supports more than one extended capability 0x00 = All other modes, indicating that only one extended capability is supported.	R	0x00

13.4.12 Power Management Capabilities Register—Offset 0xDC

This section describes the power management capabilities register.

Byte enable P_CBE_b[3:0] = 00xxb

Register name: DEV_POWER_MAN_CAP Reset value: 0x0002	Register offset: 0xDC

Bits	7	6	5	4	3	2	1	0
31:24	PME_SUP					D2_SUP	D1_SUP	Reserved
23:16	Reserved		DSI	AUX_SUP	PME_CLOCK	PM_VER		

Bits	Name	Description	Type	Reset value
31:27	PME_SUP	PME_b Support Indicates this device does not support the PME_b pin.	R	0x0
26	D2_SUP	D2 Power State Support Indicates this device does not support the D2 power management state.	R	0x0
25	D1_SUP	D1 Power State Support Indicates this device does not support the D1 power management state.	R	0x0
24:22	Reserved	Reserved. Returns 0 when read.	R	0x0
21	DSI	Device Specific Initialization Indicate that this device does not have device specific initialization requirements.	R	0x0
20	AUX_SUP	Auxiliary Power Support. Indicates this device does not have PME_b support or an auxiliary power source.	R	0x0
19	PME_CLOCK	PME_b Clock Required Indicates this device does not support the PME_b clock	R	0x0
18:16	PM_VER	Power Management Revision Indicate this device is compliant with *PCI Power Management Interface Specification (Revision 1.1)*.	R	010

13.4.13 Power Management Data Register—Offset 0xE0

This section describes the power management control and status register.

Byte enable P_CBE_b[3:0] = xx00b

Register name: DEV_POWER_MAN_DATA Reset value: 0x0000	Register offset: 0xE0

Bits	7	6	5	4	3	2	1	0
15:08	PME_STAT	DATA_SCALE		DATA_SEL				PME_EN
07:00	Reserved						PWR_STATE	

Bits	Name	Description	Type	Reset value
15	PME_STAT	PME Status. Reads as 0 because the PME_b pin is not implemented.	R	0x0
14:13	DATA_SCALE	Data Scale This data field is not implemented.	R	0x0
12:9	DATA_SEL	Data Select This data field is not implemented.	R	0x0
8	PME_EN	PME_b Enable The PME_b pin is not implemented.	R	0x0
07:02	Reserved	Reserved. Returns 0 when read.	R	0x0
01:00	PWR_STATE	Power State Indicates the current power state of this device. If an unimplemented power state is written to this register, Tsi352 completes the write transaction, ignores the write data, and does not change the value of this field. Writing a value of D0 when the previous state was D3 causes a chip reset to occur (without asserting S_RST_b) 00 = D0 01 = D1 (not implemented) 10 = D2 (not implemented) 11 = D3	R/W	0x0

13.4.14 PMCSR_BSE Register — Offset 0xE0

Byte enable P_CBE_b[3:0] = x0xxb

Register name: DEV_PMCSR_BSE Reset value: Undefined	Register offset: 0xE0

Bits	7	6	5	4	3	2	1	0
23:16	BPCC_EN	B2_B3_SUP	Reserved					

Bits	Name	Description	Type	Reset value
23	BPCC_EN	Bus Power/Clock Control Enable. This bit returns value of MS1/BPCC input. 0 = Bus/Power Clock Control disable 1 = Bus/Power Clock Control enable	R	Undefined
22	B2_B3_SUP	B2_B3 Support for D3hot. This bit returns the value of MS1/BPCC input. 0 = the secondary clocks remain on in all device states. 1 = the secondary clocks are stopped when the device is placed in D3hot. Note: If the primary clock is stopped, then the secondary clocks stop because the primary clock is used to generate the secondary clocks	R	Undefined
21:16	Reserved	Reserved. Returns 0 when read.	R	0x0

13.4.15 HS Capability ID Register — Offset 0xE4

Byte enable P_CBE_b[3:0] = xxx0b

Register name: DEV_HS_CAP_ID Reset value: 0x06	Register offset: 0xE4

Bits	7	6	5	4	3	2	1	0
07:00	HS_CAP_ID							

Bits	Name	Description	Type	Reset value
07:00	HS_CAP_ID	Hot Swap Capability ID Indicates this register set of the capability list is a Hot Swap Register Set.	R	0x06

13.4.16 HS Next Item Pointer Register — Offset 0xE4

Register name: DEV_HS_NEXT_ITEM Reset value: 0x00	Register offset: 0xE4

Bits	7	6	5	4	3	2	1	0
15:08	HS_NEXT_ITEM							

Bits	Name	Description	Type	Reset value
15:08	HS_NEXT_ITEM	Hot Swap Next Item Pointer Indicates there are no more list items in the capabilities list.	R	0x0

13.4.17 HS Control Status Register — Offset 0xE4

Register name: DEV_HS_CSR Reset value: 0x00	Register offset: 0xE4

Bits	7	6	5	4	3	2	1	0
23:16	INS	Extraction	PI		LOO	PIE	EIM	DHA

Bits	Name	Description	Type	Reset value
23	INS	Insertion The bridge sets this bit to indicate software that a board has been freshly inserted. This bit is set only after the following events occur: • Ejector Handle is closed. • P_RST_b deasserted. The bridge asserts ENUM_b when this bit is set.	R/W1TC	0x0
22	Extraction	Extraction The bridge sets this bit to indicate software that a board is about to be removed from the system. The bridge asserts ENUM_b when this bit is set.	R/W1TC	0x0
21:20	PI	Returns 0 when read.	R	0x0
19	LOO	LED On Off This bit controls an external LED indicator for user feedback. When software writes a logical one to this register, the LED is illuminated. When a logical zero is written to this bit the LED is not illuminated.	R/W	0x0
18	PIE	Returns 0x0 when read.	R	0x0
17	EIM	ENUM Interrupt Mask This bit allows the ENUM_b pin to be masked by software. Writing a logical 1 to this bit masks the ENUM_b pin from being driven. Writing a logical 0 to this bit enables ENUM_b to be driven.	R/W	0x0
16	DHA	Reserved. Returns 0 when read	R	0x0

14. Ordering Information

Topics discussed include the following:

14.1 Part Numbers

The following table contains ordering information for the Tsi352.

Table 31: Part Numbers

Part Number	Frequency	Temperature	Package	Pin Count
Tsi352-66CQ	66 MHz	Extended Commercial (0°C to 85°C)	PQFP	160
Tsi352-66CQY	66 MHz	Extended Commercial (0°C to 85°C)	PQFP RoHS/Green	160

Glossary

Address decode window — The address range defined by a device's base address registers when operating in non-transparent addressing mode. If a transaction address on the bus falls within a device's address decode window, the device claims the transaction.

Base and limit register — A configuration register that stores memory or I/O address decode information in a device. If the address of a transaction falls within the window defined by a device's base and limit registers, the device claims the transaction. Base and Limit registers are used only by transparent bridges.

CompactPCI — cPCI. It is an adaptation of the *PCI Local Bus Specification (Revision 2.2)* for Industrial and/or embedded applications that require a more robust mechanical form factor than desktop PCI.

Configuration transaction — A read or write access of a PCI device's configuration registers.

Fairness algorithm — Arbitration logic that helps low and high priority devices gain fair access to a peripheral bus. This logic also helps prevent deadlocks among bus-mastering devices in a system.

Hot swap — This refers to the process of inserting and extracting CompactPCI boards from an active system without adversely affecting system operation.

Memory-mapped I/O — MIO. Memory-mapped I/O is used for non-prefetchable PCI memory transactions.

Message — A TLP used to communicate information outside of the memory, I/O, and configuration spaces. Message TLPs are always posted, and may or may not contain data.

Non-transparent addressing — This type of addressing is used by a PCI bridging device to isolate the primary address map from the secondary address map. It provides address translation for PCI devices located in separate address domains with multiple host processors. This mode of operation, which is sometimes called embedded bridging, allows for distinct PCI memory spaces to be connected through defined windows with address translation from one memory domain to another.

PCI extended capabilities — Optional features supported by the *PCI Local Bus Specification*. Some examples of extended capabilities include: Vital Product Data, Message Signaled Interrupts, and Slot Numbering. A device that supports extended capabilities uses a PCI capability list to access the features located in its PCI configuration space.

Prefetchable memory — The process of prefetching memory occurs when a line of information from memory is read before a bus master requests it. If a bus master later requests the memory line, the bus target can supply it immediately. This type of memory access minimizes the time required to retrieve target memory.

Transparent addressing — This type of PCI addressing is used by a bridging device to support configuration mapping but not perform address translation between two buses. When a device is configured in transparent mode, it provides standard PCI bus bridging support through its base and limit registers. These registers define address decode windows for multiple bridges so that transactions can be passed transparently in a system. This enables devices that are connected to multiple bridging devices to share a single, unified address space.

Index

Numerics

A

B

C

D

E

F

M

O

P

R

S

T

W

CORPORATE HEADQUARTERS
6024 Silver Creek Valley Road
San Jose, CA 95138

for SALES:
800-345-7015 or 408-284-8200
fax: 408-284-2775
www.idt.com

for Tech Support:
email: ssdhelp@idt.com
phone: 408-284-8208
Document: 80D6000_MA001_03