IDT® Tsi574
Serial RapidIO Switch

User Manual

June 6, 2016

GENERAL DISCLAIMER

Integrated Device Technology, Inc. (“IDT”) reserves the right to make changes to its products or specifications at any time, without notice, in order to improve design or performance. IDT does not assume responsibility for use of any circuitry described herein other than the circuitry embodied in an IDT product. Disclosure of the information herein does not convey a license or any other right, by implication or otherwise, in any patent, trademark, or other intellectual property right of IDT. IDT products may contain errata which can affect product performance to a minor or immaterial degree. Current characterized errata will be made available upon request. Items identified herein as “reserved” or “undefined” are reserved for future definition. IDT does not assume responsibility for conflicts or incompatibilities arising from the future definition of such items. IDT products have not been designed, tested, or manufactured for use in, and thus are not warranted for, applications where the failure, malfunction, or any inaccuracy in the application carries a risk of death, serious bodily injury, or damage to tangible property. Code examples provided herein by IDT are for illustrative purposes only and should not be relied upon for developing applications. Any use of such code examples shall be at the user's sole risk.

The IDT logo is registered to Integrated Device Technology, Inc. IDT and CPS are trademarks of Integrated Device Technology, Inc.

Contents

Figures

Tables

About this Document

This section discusses the following topics:

Scope

The *Tsi574 User Manual* discusses the features, capabilities, and configuration requirements for the Tsi574. It is intended 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]

Differential Signal Notation

Differential signals consist of pairs of complement positive and negative signals that are measured at the same time to determine a signal's active or inactive state (they are denoted by "_p" and "_n", respectively). The following table illustrates the differential signal naming convention.

State	Single-line signal	Multi-line signal
Inactive	NAME_p = 0 NAME_n = 1	NAME_p[3] = 0 NAME_n[3] =1
Active	NAME_p = 1 NAME_n = 0	NAME_p[3] is 1 NAME_n[3] is 0

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

- 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

June 6, 2016, Formal

- Updated “Reserved Register Addresses and Fields”
- Updated the second caution in “RapidIO Error Management Extension Registers”
- Updated the description of bit 31 (Reserved) in the following registers: “SRIO MAC x SerDes Configuration Channel 0”, “SRIO MAC x SerDes Configuration Channel 1”, “SRIO MAC x SerDes Configuration Channel 2”, and “SRIO MAC x SerDes Configuration Channel 3”
- Removed Ordering Information from the manual. This information now resides solely in the *Tsi574 Hardware Manual*.

February 19, 2015, Formal

- Updated the "Ordering Information"

September 16, 2014, Formal

- Updated step 2 in the "Hot Extraction" procedure
- Added a new section, "Lane Sync Timer"
- Updated "Power-Down Options"
- Updated Figure 14: Drive Strength and Equalization Waveform
- Added a new section, "Multicast Operation with Multiple Tsi57x Switches"
- Updated steps 3 and 4 in "Control Symbol Example"
- Updated the description of Fatal Port Error in Table 13: Tsi574 Events
- Updated the description of "Per-Port Reset"
- Updated the description of "RapidIO Port x Error and Status CSR".PORT_ERR
- Updated "Tsi574_read_prbs_all.txt Script"

May 25, 2012, Formal

- Updated the second step in "Removing a Destination ID to Multicast Mask Association"
- Updated the second paragraph in "Payload"
- Updated "Port-writes and Multicast"
- Updated the registers listed in "Global Registers to Program after Port Power Down"
- Added a note about how SW_RST_b is the only external indicator that a reset request has been received to "System Control of Resets" and Table 31

November 19, 2010, Formal

- Added more information about "Lookup Table Entry States"
- Added more information about "Port Aggregation: 1x and 4x Modes"
- Added a note to the "SRIO MAC x SerDes Configuration Global" register
- Added more information about "SRIO MAC x Digital Loopback and Clock Selection Register".DLT_THRESH

July 2009, Formal

This is the production version of the *Tsi574 User Manual*. The document has been updated with IDT formatting. There have been no technical changes.

1. Functional Overview

This chapter describes the main features and functions of the Tsi574. This chapter includes the following information:

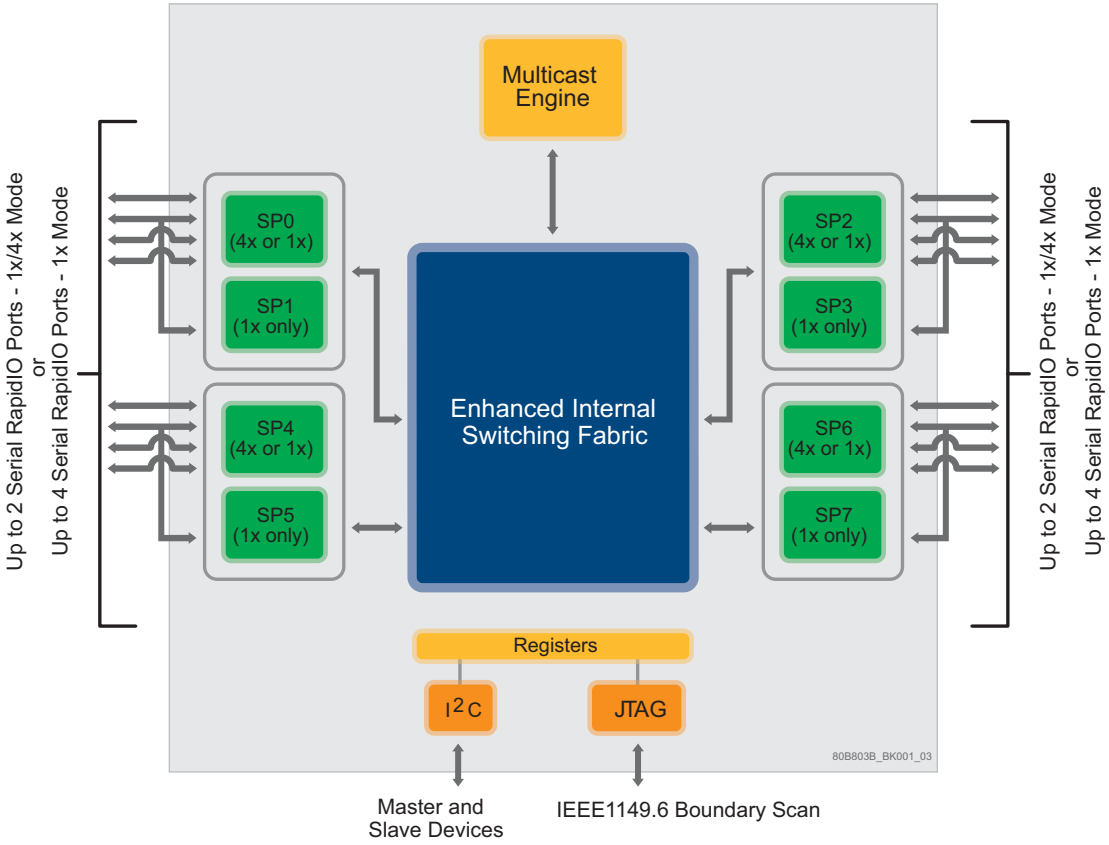
1.1 Overview

The IDT Tsi574^{TM} is a third generation RapidIO switch supporting 40 Gbits/s aggregate bandwidth. The Tsi574 is part of a family of switches that enable customers to develop systems with robust features and high performance at low cost.

The Tsi574 provides designers and architects with maximum scalability to design the device into a wide range of applications. Flexible port configurations can be selected through multiple port width and frequency options.

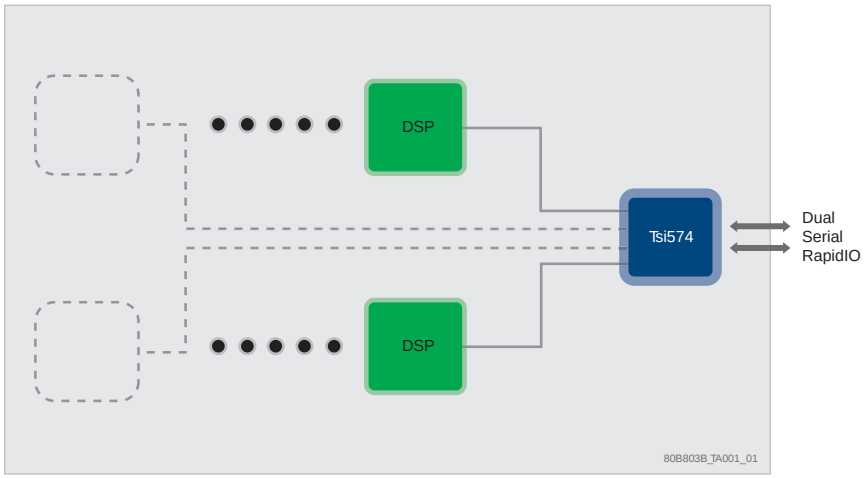
Building on the industry leading Tsi564A^{TM} 4/8 Port Serial RapidIO Switch, the Tsi574 contains all the benefits of its predecessor plus enhances the fabric switching capabilities through the addition of multicast, traffic management through scheduling algorithms, programmable buffer depth, and fabric performance monitoring to supervise and manage traffic flow.

Figure 1: Block Diagram



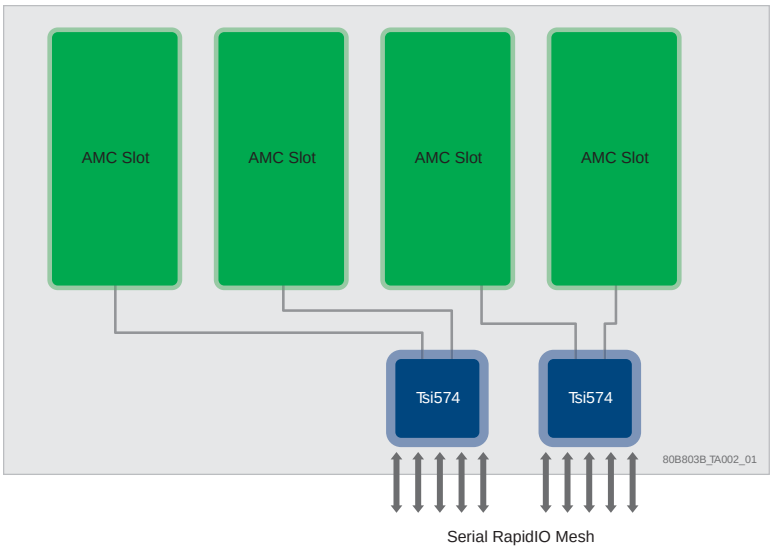
The Tsi574 can be used in many embedded communication applications. It provides chip-to-chip interconnect between I/O devices and can replace existing proprietary backplane fabrics for board-to-board interconnect which improves system cost and product time-to-market.

Figure 2: Processor Farm Mezzanine Diagram



The Tsi574 provides traffic aggregation through packet prioritization when it is used with RapidIO-enabled I/O devices. When it is in a system with multiple RapidIO-enabled processors it provides high performance peer-to-peer communication through its non-blocking switch fabric.

Figure 3: Switch Carrier Blade Diagram



1.1.1 Features

The Tsi574 contains the following features:

Electrical Layer Serial RapidIO Features

- Up to 4 ports in 4x Serial mode
- Up to 8 ports in 1x Serial mode (each 4x port can be configured independently as two 1x ports)
- Operating baud rate per data lane: 1.25 Gbit/s, 2.5 Gbit/s, or 3.125Gbit/s
- Full duplex bandwidth:
  - 12.5 Gbit/s inbound and 12.5 Gbit/s outbound bandwidth at 3.125 GHz for a port configured for 4x mode[1]
  - 3.125 Gbit/s inbound and 3.125 Gbit/s outbound bandwidth at 3.125 GHz for a port configured for 1x mode[2]
- Programmable serial transmit current with pre-emphasis equalization
- Loopback support for system testing

1. Usable data rate is 10 Gbit/s rather than 12.5 Gbit/s due to 8B/10B physical layer encoding.
2. Usable data rate is 2.5 Gbit/s rather than 3.125 Gbit/s due to 8B/10B physical layer encoding.

- Hot-insertion capable I/Os and hardware support
- Per-port power down modes to reduce power consumption
- Ability to reverse the bit ordering of a 4x port to simplify PCB layout

Transport Layer RapidIO Features

- Dedicated destination ID lookup table per port, used to direct packets through the switch
- Supports both hierarchical lookup tables and flat mode lookup tables (512 destination IDs per lookup table)
- Supports an optional, unique hierarchical destination ID lookup table covering all 64K possible destinations ID
- Low-latency forwarding of the Multicast-Event control symbol
- Error management capability
- Performance monitoring capability
- Reset-system interrupt support
- Debug packet generation in debug mode

Multicast Engine Features

- One multicast engine provides dedicated multicast resources without impacting throughput on the ports
- Eight multicast groups
- Sustained multicast output bandwidth, up to 10 Gbit/s per egress port
- 10 Gbit/s of instantaneous multicast input bandwidth[1]
- Packets are replicated to each egress port in parallel
- The multicast engine can accept a bursts of traffic with different packet sizes
- Arbitration at the egress port to allow management of resource contention between multicast or non-multicast traffic.

System behavior when multicasting of packets which require responses is not defined in the *RapidIO Interconnect Specification (Revision 1.3) - Part 11 Multicast Specification*.

Other Device Interfaces

- Master and Slave mode I^2C port, supports up to 8 EEPROMs
- Optionally loads default configuration from ROMs during boot-up, through I^2C
- Ability to read and write EEPROMs through I^2C during system operation
- IEEE 1149.1 and 1149.6 boundary scan, with register access

1. All bandwidths assume the internal switching fabric is clocked at 156.25 MHz.

Internal switching fabric (ISF)

- Full-duplex,40 Gbps line rate, non-blocking switching fabric
- Prevents head-of-line blocking on each port
- Eight packet buffers per ingress port
- Eight packet buffers per egress port

Register Access

- Registers can be accessed from any RapidIO interface and both the JTAG interface and I^2C
- Optionally loads default configuration from ROMs during boot-up, through I^2C
- Supports one outstanding maintenance transaction per interface
- Supports 32-bit wide (4 byte) register access

1.2 Serial RapidIO Interface

The Tsi574 provides high-performance serial RapidIO interfaces that are used to provide connectivity for control plane and data plane applications. All RapidIO interfaces are compliant with the *RapidIO Interconnect Specification (Revision 1.3)*.

This section describes the transport layer features common to all Tsi574 RapidIO interfaces. The RapidIO interface has the following capabilities:

- RapidIO packet and control symbol transmission
- RapidIO packet and control symbol reception
- Register access through RapidIO maintenance requests

1.2.1 Features

The following features are supported:

- Up to four 4x-mode or up to 8 1x-mode serial RapidIO ports operating at up to 3.125 Gbits/s
- Per-port destination ID look-up table, used to direct packets through the switch

This is a IDT-specific implementation. The *RapidIO Interconnect Specification (Revision 1.3)* standard implementation of look-up tables is also supported.

- RapidIO error management extensions, including both hardware and software error recovery (described in *RapidIO Interconnect Specification (Revision 1.3) Part 8)*
- Low latency forwarding of the multicast control symbol
- Proprietary registers for performance monitoring and tuning
- Both cut-through and store-and-forward modes for performance tuning
- Debug packet generation and capture
- Multicast functionality (described in *RapidIO Interconnect Specification (Revision 1.3) Part 11)*
- Head-of-line blocking avoidance

1.2.2 Transaction Flow Overview

Packets and control symbols are received by the Serial RapidIO Electrical Interface (Serial MAC) and forwarded to the RapidIO Interface (for more information on the Serial MAC, refer to “Serial RapidIO Electrical Interface” on page 65). Received packets have their integrity verified by error checking. Once the packet’s integrity has been verified, the destination ID of the packet is used to access the routing lookup table to determine which port the packet should be forwarded to and whether the packet is a multicast packet. The packet is then buffered by the Internal Switch Fabric (ISF) for transmission to the port. After the packet is transferred to the egress port, the port transmits the packet. If a packet fails the CRC check, the packet is discarded and the transmitter is instructed to retransmit the packet through the use of control symbols.

The egress port receives packets to be transmitted from the ISF. The integrity of packets forwarded through the ISF is retained by sending the CRC code received with the packet. For more information on the input and ouput queues, refer to “Packet Queuing” on page 95.

The packet transmitter and the packet receiver cooperate to ensure that packets are never dropped (lost). A transmitter must retain a packet in its buffers until the port receives a packet accepted control symbol from the other end of the link.

1.2.3 Maintenance Requests

A maintenance packet is the only packet type that will be modified by the switch. If the hop count value of the maintenance request is 0, the maintenance request is forwarded to the register bus for processing. The register bus accesses the registers in the appropriate port. The response to the maintenance request is compiled into a maintenance response packet and queued by the port for transmission. Maintenance packets with a non-zero hop count value have their hop count decremented, CRC recomputed, and are then forwarded to the port selected by the destination ID value in the look up table.

1.2.4 Control Symbols

Control symbols received by the Tsi574 have their CRC validated, and their field values checked. If either the CRC is incorrect or the control symbol field values are incorrect, a packet-not-accepted control symbol is sent back and the control symbol is discarded. Otherwise, the control symbol is used by the port for purposes of packet management in the transmit port or link maintenance.

1.3 Multicast Engine

The Tsi574 multicast functionality is compliant to the *RapidIO Version 1.3 Part 11 Multicast Specification.*

1.3.1 Multicast Operation

In a multicast operation, packets are received at the speed of any ingress port (up to 10 Gbits/s) and broadcast at the speed of the egress ports (up to 10 Gbits/s for a 4x mode port operating at 3.125 Gbits/s) to multiple ports capable of accepting packets for transmission.

Packets are routed to the multicast engine based on their destinationID and Transaction Type (TT) field value. If no match is found for the destinationID and TT field, then the ingress lookup tables are used to route the packet. A maximum of eight different DestID/TT field combinations can be routed to the multicast engine. Each destinationID/TT set can be multicast to a different set of egress ports. A set of egress ports that packets are multicast to is called a multicast group and is represented by the multicast mask in the group table. A multicast packet is never sent out on the port that it was received on. Any number of ports can share the same multicast group.

Multicast packets are accepted by egress ports based on priority. In the event that multicast and unicast traffic are competing for resources in the egress port, multicast specific egress arbitration can be used to favour multicast or unicast traffic. This allows a group of endpoints that need to multicast to each other to share the same multicast mask.

1.3.2 Features

The Tsi574 supports multicast packet replication in accordance with *RapidIO Specification Version 1.3, Part 11 Multicast.*

The Tsi574 includes the following features:

- One multicast engine provides dedicated multicast resources without impacting throughput on the ports
- Eight multicast groups
- Sustained multicast output bandwidth, up to 10 Gbit/s per egress port
- 10 Gbit/s of instantaneous multicast input bandwidth[1]
- Packets are replicated to each egress port in parallel
- The multicast engine can accept bursts of traffic with different packet sizes
- Arbitration at the egress port to allow management of resource contention between multicast or unicast traffic

System behavior for the multicasting of packets which require responses is not defined in the *RapidIO Interconnect Specification (Revision 1.3) - Part 11 Multicast Specification*.

1.4 Serial RapidIO Electrical Interface

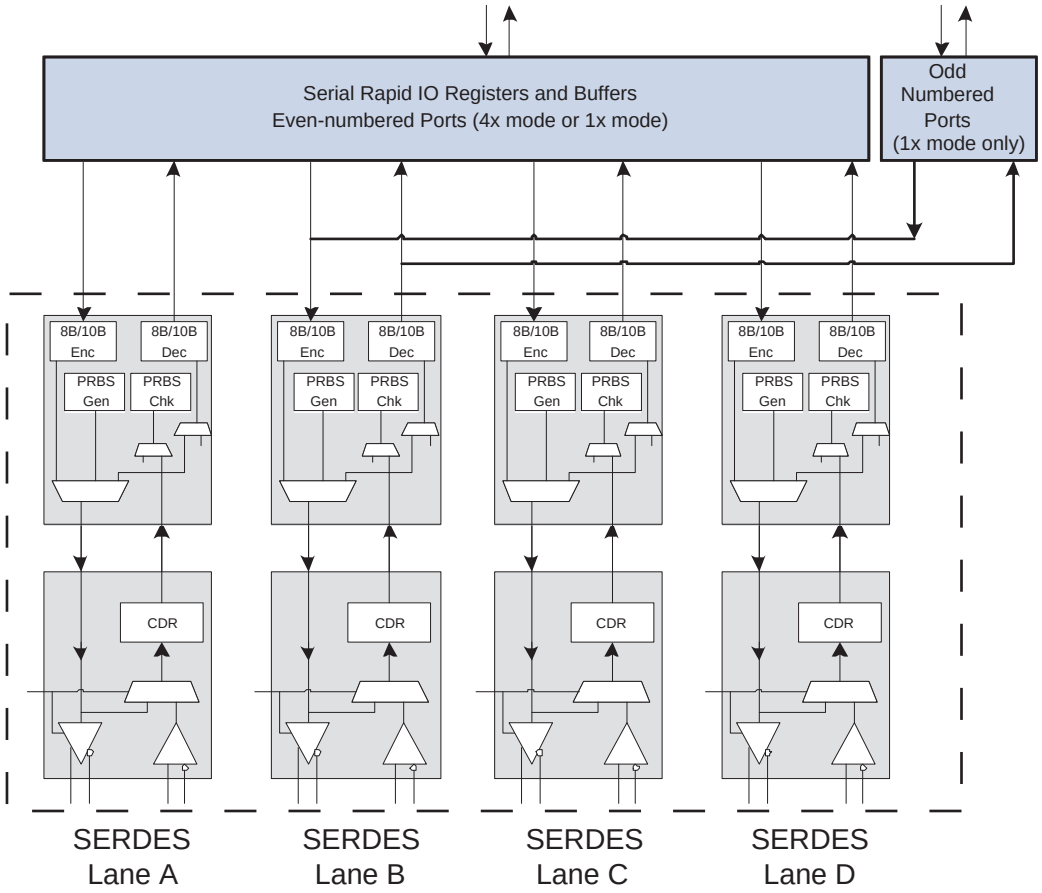
The Tsi574 has four Media Access Controllers (MAC) comprising the 8 Serial RapidIO ports. The 8 ports are grouped into pairs consisting of one even numbered port and one odd numbered port. Each port has flexible testing features including multiple loopback modes and bit error rate testing. Each pair of ports share four differential transmit lanes and four differential receive lanes.

Even and odd number ports have different capabilities. Even numbered ports can operate in either 4x or 1x mode, while odd numbered ports can only operate in 1x mode. When the even numbered port is operating in 4x mode, it has control over all four differential pairs (designated Lanes A, B, C and D). In 4x mode, the default state of the odd numbered port is powered on. All registers in the even and odd numbered port are accessible but the odd numbered port does not have access to the PHY. In order to decrease the power dissipation of the port, the odd numbered port can be powered down in this configuration. When the even numbered port is operating in 1x mode it uses only Lane A and the odd numbered port is permitted to operate in 1x mode using Lane B.

The Tsi574 MAC and SerDes interconnect block diagram is shown in the following figure.

1. All bandwidths assume the internal switching fabric is clocked at 156.25 MHz.

Figure 4: Tsi574 MAC Block Diagram



Each serial RapidIO MAC includes the following features:

- One port in 4x Serial mode
- Two ports in 1x Serial mode (each 4x port can be configured as two 1x ports)
- RapidIO standard operating baud rate per data lane: 1.25 Gbit/s, 2.5 Gbit/s, or 3.125 Gbit/s
  - 12.5 Gbit/s inbound and 12.5 Gbit/s outbound bandwidth at 3.125 Gbps for a port configured for 4x mode
  - 3.125 Gbit/s inbound and 3.125 Gbit/s outbound bandwidth at 3.125 Gbps for a port configured for 1x mode
- Adjustable receive equalization that is programmable per lane
- Serial loopback with a built-in testability
- Bit error rate testing (BERT)
- Scope function of eye signals
- Hot-insertion capable I/Os and hardware support

1.5 Internal Switching Fabric (ISF)

The Internal Switching Fabric (ISF) is the crossbar switching matrix at the core of the Tsi574. It transfers packets from ingress ports to egress ports and prioritizes traffic based on the RapidIO priority associated with a packet and port congestion.

The ISF has the following features:

- Full-duplex, non-blocking, crossbar-based switch fabric
- 10 Gbits/s fabric ports allow up to 10x internal speedup
- Manages head-of-line blocking on each port
- Cut-through and store-and-forward switching of variable-length packets

1.6 Internal Register Bus (AHB)

An internal multi-master Advanced High Performance Bus (AHB) allows any RapidIO port to configure and maintain the entire device. When the Tsi574 receives a RapidIO maintenance packet destined for itself, it translates the packet into register read or write request on the AHB.

The device registers can also be accessed through the JTAG interface or the I^2C interface.

1.7 I^2C Interface

The I^2C Interface provides a master and slave serial interface that can be used for the following purposes:

- Initializing device registers from an EEPROM after reset
- Reading and writing external devices on the I^2C bus
- Reading and writing Tsi574's internal registers for management purposes by an external I^2C master

The I^2C Interface has the following features:

- Operates as a master or slave on the I^2C bus
  - Multi-master support
    - Arbitrates among multiple masters for ownership of the I^2C bus
    - Automatically retries accesses if arbitration is lost
    - Provides timeout indication if the Tsi574 is unable to arbitrate for the I^2C bus

— I2C Interface: Master interface
  – Supports 7-bit device addressing
  – Supports 0, 1, or 2-byte peripheral addressing
  – Supports 0, 1, 2, 3, or 4-byte data transfers
  – Reverts to slave mode if arbitration is lost
  – Supports clock stretching by an external slave to limit bus speed to less than 100 kHz
  – Handles timeouts and reports them through interrupts

— I2C Interface: Slave interface
  – Slave address can be loaded from three sources: power-up signals, boot load from EEPROM, or by software configuration
  – Provides read and write accesses that are 32 bits in size to all Tsi574 registers
  – Ignores General-Call accesses
  – Ignores Start-Byte protocol
  – Provides a status register for determination of Tsi574's health
  – Slave operation enabled/disabled through power-up signal, boot load from EEPROM, or by software configuration
  – Provides mailbox registers for communicating between maintenance software operating on RapidIO based processors and external I2C masters

- Supports I2C operations up to 100 kHz
- Provides boot-time register initialization
  — Supports 1- and 2-byte addressing of the EEPROM selected by power-up signal
  — Verifies the number of registers to be loaded is legal before loading registers
  — Supports up to 2K byte address space and up to 255 address/data pairs for register configuration in 1-byte addressing mode, or up to 65 Kbyte address space and up to 8 K-1 address/data pairs in 2-byte addressing mode.
  — Supports chaining to a different EEPROM and/or EEPROM address during initialization.

The I2C Interface does not support the following features:

- START Byte protocol
  — Tsi574 does not provide a START Byte in transactions it masters
  — Tsi574 does not respond to START Bytes in transactions initiated by other devices. The Tsi574 will respond to the repeated start following the start byte provided the 7-bit address provided matches the Tsi574 device address.

- CBUS compatibility
  - Tsi574 does not provide the DLEN signal
  - Tsi574 does not respond as a CBUS device when addressed with the CBUS address. The Tsi574 will interpret the CBUS address like any other 7-bit address and compare it to its device address without consideration for any other meaning.
- Fast Mode or High-Speed Mode (HS-MODE)
- Reserved 7-bit addresses should not be used as the Tsi574's 7-bit address. If a reserved address is programmed, the Tsi574 will respond to that address as though it were any other 7-bit address with no consideration of any other meaning.
- 10-bit addressing
  - Tsi574 must not have its device address programmed to the 10-bit address selection (11110XXb) in systems that use 10-bit addressing. The Tsi574 will interpret this address like any other 7-bit address and compare it to its device address without consideration for any other meaning.
- General Call. The general call address will be NACK'd and the remainder of the transaction ignored up to a subsequent Restart or Stop.

1.8 JTAG Interface

The JTAG interface in Tsi574 is fully compliant with IEEE 1149.6 B*oundary Scan Testing of Advanced Digital Networks* as well as IEEE 1149.1 *Standard Test Access Port and Boundary Scan Architecture* standards. There are five standard pins associated with the interface (TMS, TCK, TDI, TDO and TRST_b) which allow full control of the internal TAP (Test Access Port) controller.

The JTAG Interface has the following features:

- Contains a 5-pin Test Access Port (TAP) controller, with support for the following registers:
  - Instruction register (IR)
  - Boundary scan register
  - Bypass register
  - Device ID register
  - User test data register (DR)
- IDT-specific pin (BCE) which allows full 1149.6 compliant boundary-scan tests. This pin should be held high on the board.
- Supports debug access of Tsi574's configuration registers
- Supports the following instruction opcodes:
  - Sample/Preload
  - Extest
  - EXTEST_PULSE *(1149.6)*
  - EXTEST_TRAIN *(1149.6)*

- Bypass
- Hi-Z
- IDCODE
- Clamp
- User data select

2. Serial RapidIO Interface

This chapter describes the serial RapidIO interface of the Tsi574. It includes the following information:

2.1 Overview

The Tsi574 provides high-performance serial RapidIO interfaces that are used to provide connectivity for control plane and data plane applications. All RapidIO interfaces are compliant with the *RapidIO Interconnect Specification (Revision 1.3)*.

This section describes the transport layer features common to all Tsi574 RapidIO interfaces. The RapidIO interface has the following capabilities:

- RapidIO packet and control symbol transmission
- RapidIO packet and control symbol reception
- Register access through RapidIO maintenance requests

2.1.1 Features

The following features are supported:

- Up to four 4x-mode or up to 8 1x-mode serial RapidIO ports operating at up to 3.125 Gbits/s
- Per-port destination ID look-up table, used to direct packets through the switch

This is a IDT-specific implementation. The *RapidIO Interconnect Specification (Revision 1.3)* standard implementation of look-up tables is also supported.

- RapidIO error management extensions, including both hardware and software error recovery (described in *RapidIO Interconnect Specification (Revision 1.3) Part 8)*
- Low latency forwarding of the multicast control symbol

- Proprietary registers for performance monitoring and tuning
- Both cut-through and store-and-forward modes for performance tuning
- Debug packet generation and capture
- Multicast functionality (described in *RapidIO Interconnect Specification (Revision 1.3) Part 11)*
- Head-of-line blocking avoidance

2.1.2 Transaction Flow Overview

Packets and control symbols are received by the Serial RapidIO Electrical Interface (Serial MAC) and forwarded to the RapidIO Interface (for more information on the Serial MAC, refer to “Serial RapidIO Electrical Interface” on page 65). Received packets have their integrity verified by error checking. Once the packet’s integrity has been verified, the destination ID of the packet is used to access the routing lookup table to determine which port the packet should be forwarded to and whether the packet is a multicast packet. The packet is then buffered by the Internal Switch Fabric (ISF) for transmission to the port. After the packet is transferred to the egress port, the port transmits the packet. If a packet fails the CRC check, the packet is discarded and the transmitter is instructed to retransmit the packet through the use of control symbols.

The egress port receives packets to be transmitted from the ISF. The integrity of packets forwarded through the ISF is retained by sending the CRC code received with the packet. For more information on the input and ouput queues, refer to “Packet Queuing” on page 95.

The packet transmitter and the packet receiver cooperate to ensure that packets are never dropped (lost). A transmitter must retain a packet in its buffers until the port receives a packet accepted control symbol from the other end of the link.

2.1.3 Maintenance Requests

A maintenance packet is the only packet type that will be modified by the switch. If the hop count value of the maintenance request is 0, the maintenance request is forwarded to the register bus for processing. The register bus accesses the registers in the appropriate port. The response to the maintenance request is compiled into a maintenance response packet and queued by the port for transmission. Maintenance packets with a non-zero hop count value have their hop count decremented, CRC recomputed, and are then forwarded to the port selected by the destination ID value in the look up table.

2.1.4 Control Symbols

Control symbols received by the Tsi574 have their CRC validated, and their field values checked. If either the CRC is incorrect or the control symbol field values are incorrect, a packet-not-accepted control symbol is sent back and the control symbol is discarded. Otherwise, the control symbol is used by the port for purposes of packet management in the transmit port or link maintenance.

2.2 Transaction Flow

The Tsi574 receives a RapidIO packet on one of its RapidIO ports. After performing integrity checks, such as validating a CRC, the interface logic locates the destination ID in the packet. The Tsi574 uses this information to determine to which egress port the packet must be sent and whether it is a multicast packet. It consults a user-configurable lookup table, which maps destination ID into egress port numbers.

The RapidIO port transfers the packet to the Switch ISF where it is buffered and transferred to an egress port or to the Multicast Engine. The Switch ISF is non-blocking, which means that all ports can switch data at the same time as long as they are not switching data from multiple ports to a single port. The Switch ISF manages head-of-line blocking, which means that when a packet cannot be moved to an egress port (for example, because multiple ingress ports are trying to send to the same egress port), the Switch ISF selects another packet to service from the same ingress port.

The ingress queue of Tsi574 can operate in two modes: store-and-forward and cut-through modes (see “RapidIO Port x Control Independent Register” on page 311). In store-and-forward mode, the ingress port of the device waits for the arrival of the whole packet before sending it to the ISF. In cut-through mode, the ingress port transmits the packet as soon as the ISF grants access (when the routing information is received). However, in both modes the egress port always operates in cut-through mode: the packet is immediately forwarded. A copy of the packet is saved at the egress port so that it can be retransmitted should an error occur.

RapidIO provides a *stomp* function to abort partially transmitted packets that are later determined to have data integrity errors or similar errors. This means if the Tsi574 finds that a packet that is being cut-through has an error, it may send a stomp control symbol to notify the receiver that the packet was in error and all received data of the erred packet should be dropped.

Packets delivered to a Multicast Engine (MCE) are replicated, based on user-configured *multicast groups*. The MCE sends copies of the original packet to the egress ports in a parallel fashion.

Packets can cut-through from the ingress port to the Multicast Work Queue and from the Multicast Work Queue to the Broadcast Buffers. A complete packet copy must be received by a Broadcast Buffer before it attempts to forward the packet copy to the egress port.

2.3 Lookup Tables

Lookup tables (LUTs) are used to direct incoming packets to output ports. An ingress port performs this routing operation by mapping the destination ID field of an incoming packet to an egress port number on the RapidIO switch. The ingress port does this by using the destination ID as an index to a lookup table containing user-defined egress port numbers.

Each RapidIO port has its own uniquely configurable lookup table. Configuration and maintenance of the LUTs is compliant with the *RapidIO Interconnect Specification (Revision 1.3)*. All LUTs are written simultaneously by these registers. Additionally, the LUT of each port can be accessed using device-specific registers.

The LUTs support two modes of operation, selectable on a per-port basis: “Flat Mode” on page 40 and “Hierarchical Mode” on page 45. Flat mode is the default mode and it supports destination IDs in the range of 0 to 511, with a default port for destination IDs outside this range. The hierarchical model covers the full large system range of 64-KB destination IDs, with some limitations.

To ensure high system reliability, the lookup tables are parity protected. System software must intervene when a parity error is detected. The Tsi574 guarantees that packets are not incorrectly delivered when the lookup table incurs single bit errors.

When a packet arrives at the ingress port, the destination ID of the packet is examined against the Multicast Group Table to determine if the packet is a *multicast* packet (see “Multicast” on page 101).

2.3.1 Filling the Lookup Tables

The process of filling in the LUT is composed of the following series of register writes:

- The “RapidIO Route Configuration DestID CSR” on page 253 is loaded with the destination ID value to be routed
- The “RapidIO Route Configuration Output Port CSR” on page 254 is written with the desired egress port number

If there is an attempt to write a destination ID with a value of greater than 511 into the “RapidIO Route Configuration DestID CSR” on page 253 using the LRG_CFG_DESTID and CFG_DESTID fields, the upper seven bits of the destination ID in the LRG_CFG_DESTID field is truncated.

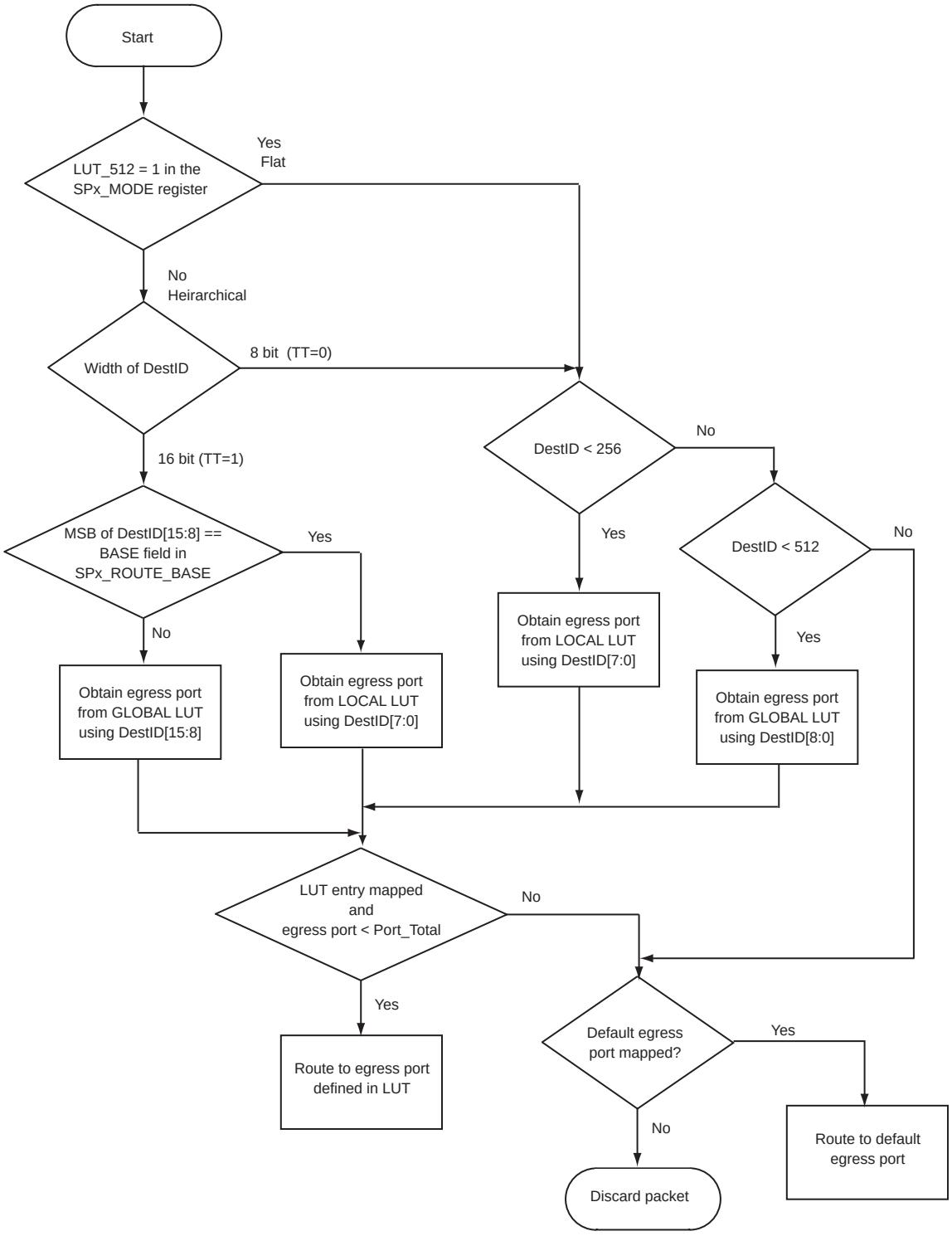
“RapidIO Route LUT Size CAR” on page 249 only advertises the switch can map 512 destination IDs. This is due to the fact this register is global in scope, whereas the ports can be independently configured for either flat mode or hierarchical mode lookup tables.

The LUT of all the ports can be loaded simultaneously if it is desired to have the same routing entries in all of the ports required. The loading process is similar to loading an individual port's LUT, however alternative registers are used. The register addresses are:

- “RapidIO Route Configuration DestID CSR” on page 253 at 0x0070 or
- “RapidIO Port x Route Config DestID CSR” on page 306 at 0x10070
- “RapidIO Route Configuration Output Port CSR” on page 254 at 0x0074 or
- “RapidIO Port x Route Config Output Port CSR” on page 307 at 0x10074

The register sets are identical except that SPx_ROUTE_CFG_PORT are per-port configuration registers and include an auto-increment bit to increment the contents of SPx_ROUTE_CFG_DESTID after a read or write operation.

Figure 5: LUT Mode of Operation



2.3.2 LUT Modes

The LUT mode, flat or hierarchical, is selected on a per-port basis through the LUT_512 field value in the “RapidIO Port x Mode CSR” on page 302.

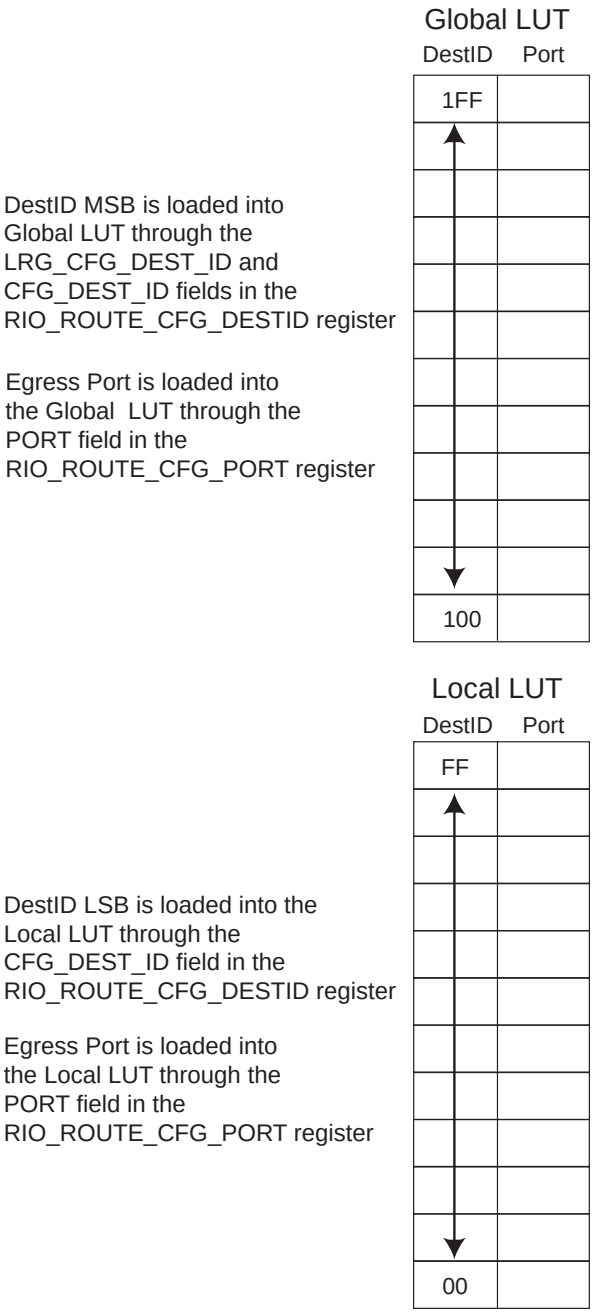
2.3.3 Flat Mode

A flat mode LUT is a table that maps destination IDs 0 to 511 to user selectable egress ports. Destination IDs that fall outside this range are sent to the egress port identified in the RIO Route LUT Attributes CSR (see “RapidIO Route LUT Attributes (Default Port) CSR”).

Flat mode is the default mode of operation of the LUT.

Figure 6 shows the configuration of the Local and Global Lookup tables (LUT) in Flat mode.

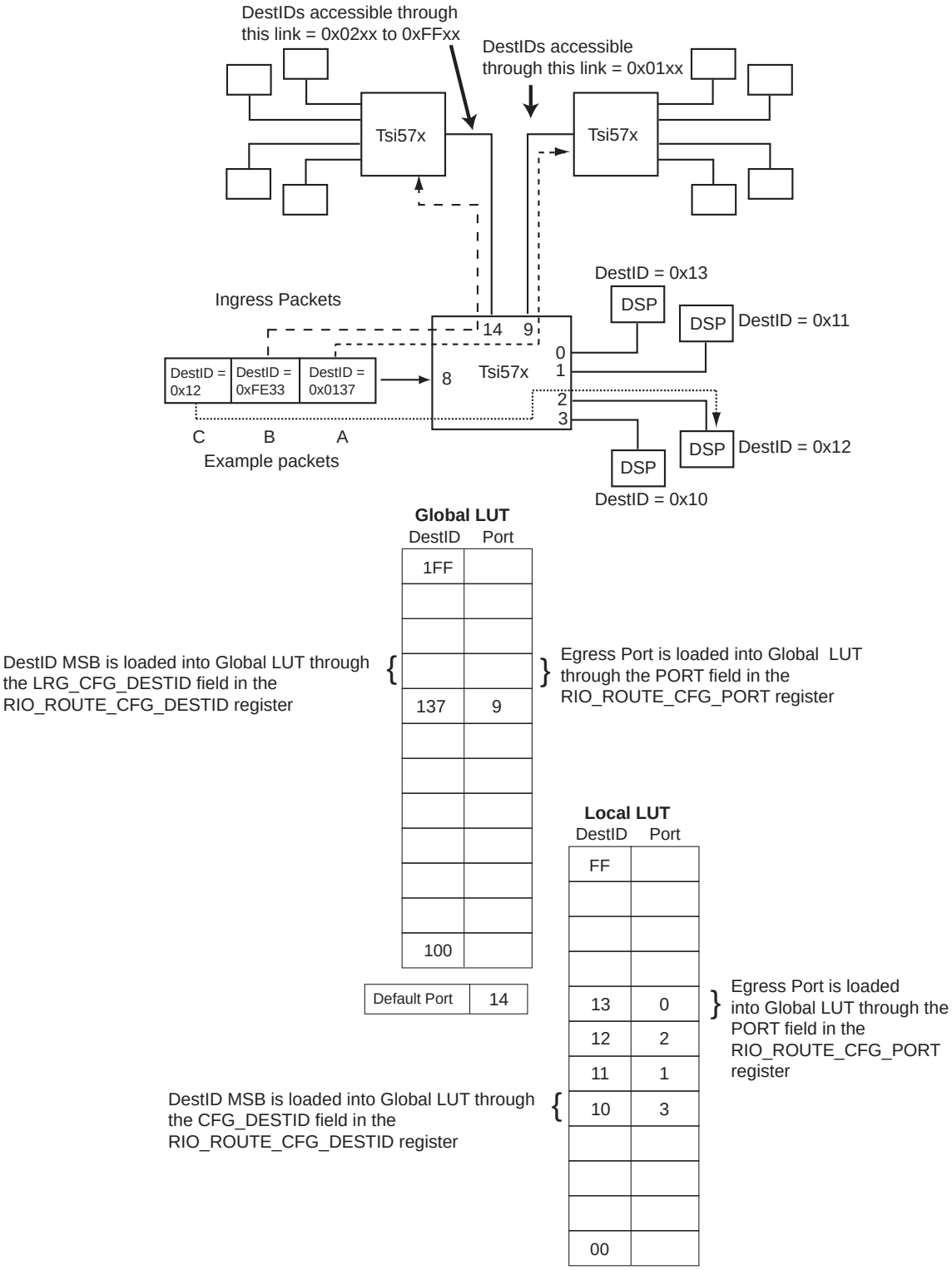
Figure 6: Flat Mode Routing



An incoming packet's destination ID is examined following the process in the flowchart in Figure 5. The egress port number is obtained from the LUT if there is a match between the destination ID in the packet header and the table. If there is no match, the packet is routed based on the default egress port programmed into "RapidIO Route LUT Attributes (Default Port) CSR" on page 255. If the default port is unmapped, the packet is discarded and the Tsi574 raises the IMP_SPEC_ERR bit in the "RapidIO Port x Error Detect CSR" on page 286.

Figure 7 shows an example of flat mode operation.

Figure 7: Flat Mode Routing Example



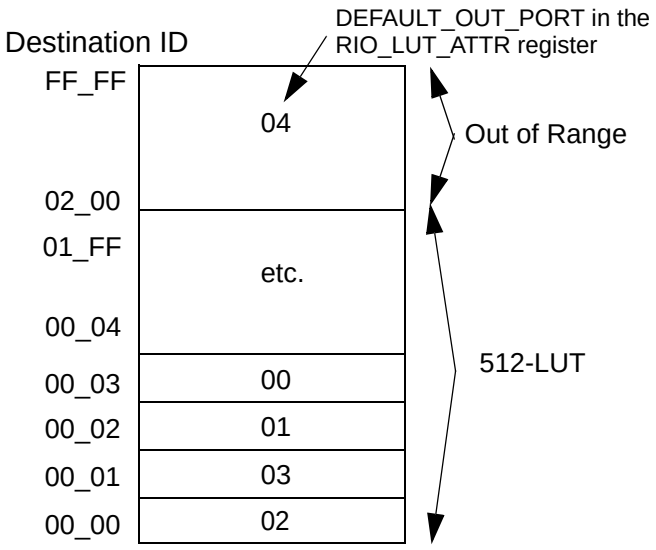
2.3.3.1 Flat LUT Programming

Each of the ports on the Tsi574 has its own lookup table. Each lookup table can be programmed with different values which allows each port to route packets differently. The lookup table maps the packet to the correct output port based on the destination ID. The capability of each port having their own LUT is functionality that is not required in the *RapidIO Interconnect Specification (Revision 1.3)*.

LUT entries can be reprogrammed at any time during normal system operation. However, software must ensure transactions have completed before reprogramming the LUTs.

Figure 8 shows an example of a LUT in flat mode. In this example, a destination ID of 0x0002, or 0x02, is routed by the switch to output port 1. A destination ID of 0x0003, or 0x03, is routed out port 0 and destination IDs greater than 0x1FF are routed out port 4.

Figure 8: Flat Mode LUT Configuration Example



Registers Used in Lookup Table Configuration

The Tsi574's RapidIO interfaces are compliant with the *RapidIO Interconnect Specification (Revision 1.3)*. The following standard RapidIO registers are used by the Tsi574 for programming the lookup tables:

- "RapidIO Route Configuration DestID CSR" on page 253
- "RapidIO Route Configuration Output Port CSR" on page 254
- "RapidIO Route LUT Size CAR" on page 249
- "RapidIO Route LUT Attributes (Default Port) CSR" on page 255
- "RapidIO Port x Local Routing LUT Base CSR" on page 308

Other related registers are IDT specific and include the following:

- "RapidIO Port x Route Config DestID CSR" on page 306

- "RapidIO Port x Route Config Output Port CSR" on page 307
- "RapidIO Port x Local Routing LUT Base CSR" on page 308

Other indirectly related (multicast) registers include:

- "RapidIO Multicast Mask Configuration Register" on page 256
- "RapidIO Multicast DestID Configuration Register" on page 258

All lookup table entries are in an unknown state after power-up. All entries should be programmed to a mapped or unmapped state to ensure predictable operation. IDT strongly recommends that the value 0xFF be used as the port value for writing unmapped lookup table entries. An unmapped lookup table entry returns the value of 0xFF as the port value when read.

Lookup Table Configuration Examples

The Tsi574 lookup tables can be configured through an external EEPROM or through software maintenance writes to the Tsi574 registers.

IDT strongly recommends that the entire lookup table be configured on each port to avoid undefined lookup table entries which can cause non-deterministic behavior.

The following sequence programs the lookup tables using the broadcast (BC) offset:

1. Write the destination ID to be configured or queried using the broadcast (BC) offset (0x10070) in the "RapidIO Port x Route Config DestID CSR" on page 306.
2. Read the "RapidIO Port x Route Config Output Port CSR" on page 307 register to determine the current egress port for the destination ID, or write this register to change the configuration of the destination ID.

Example One: Adding a Lookup Table Entry

In the following example, routing is added for all ports to route destination ID 0x98 to output port 0x4.To add a lookup table, perform the following steps:

1. Write to the "RapidIO Port x Route Config DestID CSR" on page 306, using the broadcast offset (0x10070), with a value of 0x00000098. This makes the destination ID 0x98.
2. Write to the "RapidIO Port x Route Config Output Port CSR" on page 307, using the broadcast (BC) offset (0x10074), a value of 0x00000004. This makes the egress Port 0x4 for destination ID 0x98.

Example Two: Adding a Lookup Table Entry

In the following example, routing is added for port 0x5 to route destination ID 0x20 to output port 0x3. To add a lookup table, complete the following steps:

1. Write to the "RapidIO Port x Route Config DestID CSR" on page 306, using the offset for port 5 (0x11570), with a value of 0x80000020. This makes the destination ID 0x20 and the Auto-increment 0x1
2. Write to the "RapidIO Port x Route Config Output Port CSR" on page 307, using the offset for port 5 (0x11574), with a value of 0x00000003. This programs Port 0x3.

In this example, if a further write to "RapidIO Port x Route Config DestID CSR" on page 306 (offset 0x11574) was performed the output port for destination ID 0x21 is configured

Example Three: Verifying / Reading a Lookup Table Entry

In the following example, output port for destination ID 0x54 is read. To verify and read a lookup table entry, perform the following steps:

1. Write to the "RapidIO Port x Route Config DestID CSR" on page 306, using the broadcast (BC) offset (0x10070), with a value of 0x00000054. This programs the destination ID 0x54.
2. Read to the value in "RapidIO Port x Route Config Output Port CSR" on page 307, using the broadcast (BC) offset (0x10074). This value represents the output port for packets with destination ID 0x54

The value reported back is assumed to be for all ports but it only reports back the value in port 0.

2.3.4 Hierarchical Mode

The hierarchical mode of operation of the LUT allows the full range of 65536 16-bit destination IDs to be mapped. This mode is enabled by setting RIO_SP_MODE.LUT_512 = 0. The hierarchical mode of operation uses two LUTs, each containing 256 entries.

- For packets with 8-bit destination IDs, the ingress port uses the ID as an index into the "local" LUT (see "Flat Mode").
- For packets with 16-bit destination IDs:
  - If the most significant 8 bits of the packet's destination ID match the value configured in SPx_ROUTE_BASE.BASE register field, the ingress port uses the least significant 8 bits of the packet's destination ID to index the "local LUT" and retrieve an egress port number.
  - If the most significant 8 bits of the packet's destination ID do not match the value configured in the SPx_ROUTE_BASE.BASE register field, the ingress port uses the most significant 8 bits of the packet's destination ID to index the "global LUT" and retrieve an egress port number. Thus, the majority of the 16-bit destination ID number space is covered by the global LUT, with groups of 256 destination IDs targeting the same egress switch port.

If the result of a lookup yields an egress port number greater than the value in PORT_TOTAL (“RapidIO Switch Port Information CAR” on page 245), the incoming packet is routed to the Default Port defined by “RapidIO Route LUT Attributes (Default Port) CSR”. If the default port is unmapped, the packet is discarded and the Tsi574 raises the IMP_SPEC_ERR bit in the “RapidIO Port x Error Detect CSR” on page 286.

Register RIO_LUT_SIZE only advertises the switch can map 512 destination IDs. This is due to the fact that RIO_LUT_SIZE is a register with global scope, but the ports can be independently configured for either flat mode or hierarchical mode lookup.

Figure 9: Hierarchical Mode

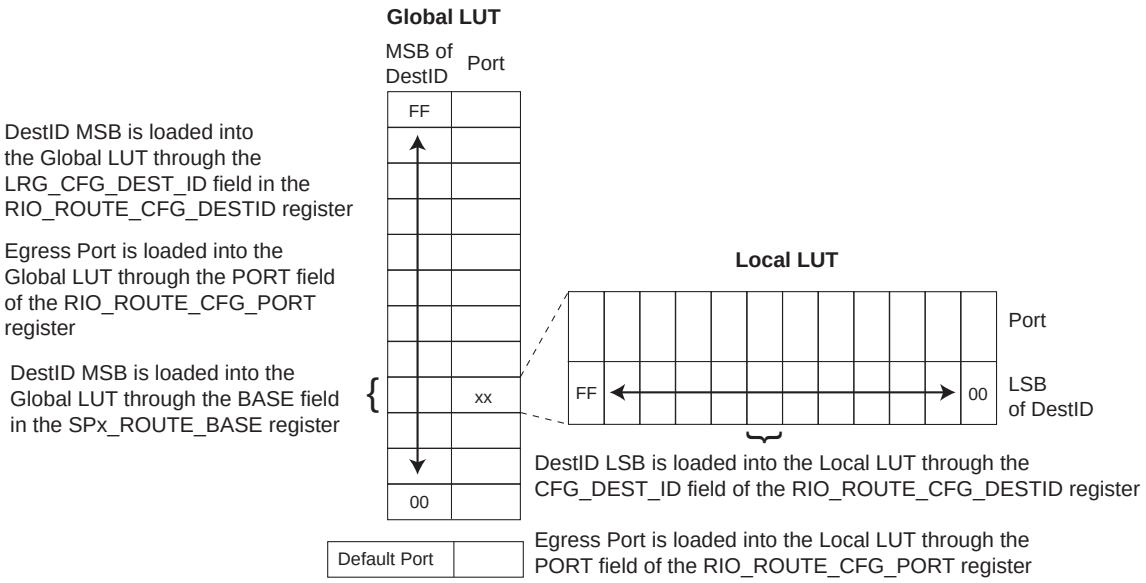
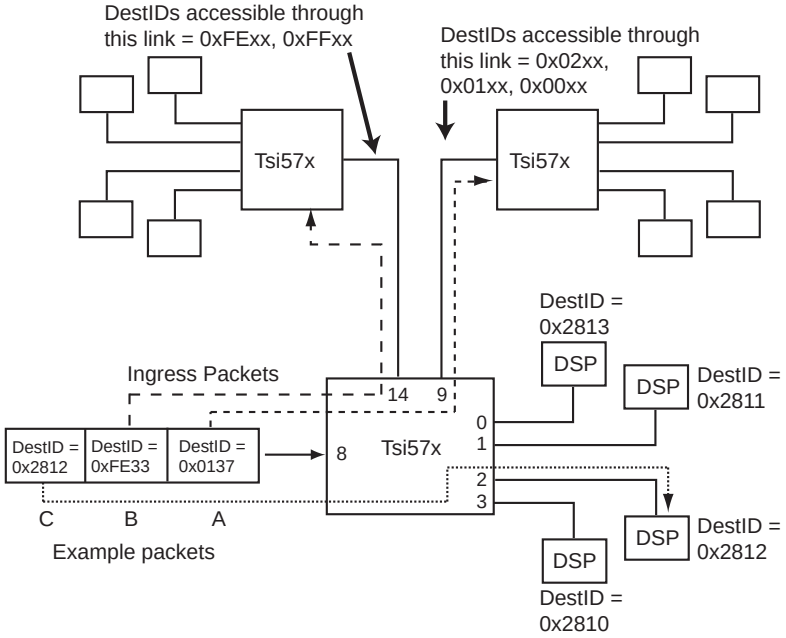
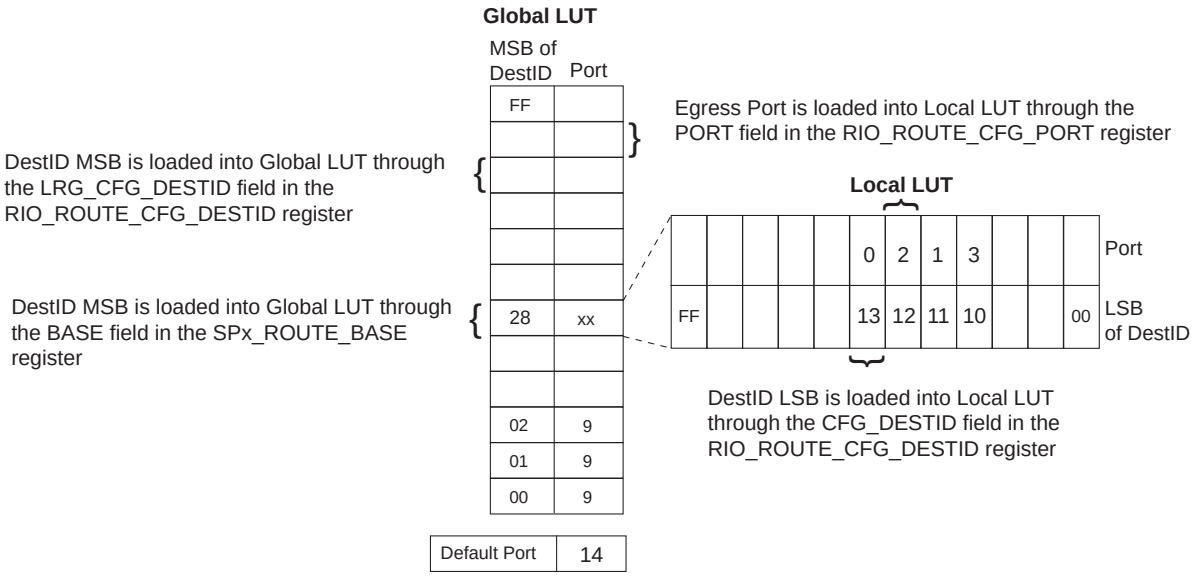


Figure 10 shows an example of hierarchical mode operation.

Figure 10: Hierarchical Mode Routing Example





2.3.4.1 Hierarchical LUT Programming

This example demonstrates the process used to program the LUT in Hierarchical mode and uses Figure 10 for reference.

The following example shows how to program the LUT in Port 8, but because all ports in the Tsi574 are capable of operating in hierarchical mode, this procedure can be easily modified to accommodate a different ingress port.

Steps 1 through 5 are in the JTAG script format for ease of re-use.

To add the required entries shown in Figure 10 after power-up, the following operations must be performed:

1. Program the LUT_512 bit = 0 in the SP8_MODE_CSR at offset 0x11804.

   w 11804 0x02000000

2. Program the Default Port = 0xA in the RIO_LUT_ATTR register at offset 0x0078. This operation may be done at any time before packet traffic starts

   w 0078 0x0A

3. Program the Global LUT with the MSB of the DEST_IDs to be routed using the following write operations:\

   W 11870 0x0 / SP8_ROUTE_CFG_DESTID

   W 11874 0x9 / SP8_ROUTE_CFG_PORT

   W 11870 0x0100

   W 11874 0x9

   W 11870 0x0200

   W 11874 0x9

   W 11870 0xFE00

   W 11874 0xE

   W 11870 0xFF00

   W 11874 0xE

4. Program the MSB of the DEST_ID that will be used to index into the Local LUT by programming the BASE field of the SP8_ROUTE_BASE register.

   W 11878 0x28000000

5. Program the Local LUT with the LSB values corresponding to an MSB of 0x28xx.

   W 11870 0x2810

   W 11874 0x3

   W 11870 0x2811

   W 11874 0x1

   W 11870 0x2812

   W 11874 0x2

   W 11870 0x2813

   W 11874 0x0

2.3.5 Mixed Mode of Operation

It is possible to operate a system in a mixed configured mode, with some ports in flat mode and some ports in hierarchical mode. Each port performs destination ID lookup consistent with its configured mode of operation.

2.3.6 Lookup Table Parity

Each entry in the lookup table is parity protected. A LUT parity error is detected in an entry when an incoming packet causes the ingress port to read that table entry. If the ingress port detects an error, it discards the packet and reports the error (see Table 1). Because the packet is discarded on the ingress port, the packet is never forwarded to the egress port and a stomp control symbol is not required when the packet is discarded.

The value of the LUT_VLD bit in the “RapidIO Port x LUT Parity Error Info CSR” on page 315 is unpredictable when there is a parity error in the LUT.

All LUT entries must be initialized before use to ensure that the parity bits are set appropriately.

2.3.7 Lookup Table Error Summary

Table 1 summarizes error conditions and resulting behaviors associated with the LUTs.

Table 1: Error Summary

Event	Behavior
Packet routed to a shut down port[a]	Packet discarded and no record of packet is kept
Packet routed to disabled port[b]	ISF time out occurs and a transaction error acknowledge (TEA) interrupt is asserted (if enabled) Note: The TEA signal is asserted when a timeout is detected on the ISF due to the requested destination being blocked. TEA is only asserted after the output port buffers are full. When this signal is asserted, it indicates to the source of the transaction that the requested transaction could not be completed and is removed from the request queue. The TEA error is reported through a port-write and/or an interrupt. Programmable in the "Fabric Control Register" and the interrupt status can be checked in the "Fabric Interrupt Status Register"
Packet routed to unconnected port[c]	Packet discarded and no record of packet is kept
Packet routed using an unmapped LUT entry, and the default egress port is also unmapped.	Packet header is recorded in the error capture registers and the packet is discarded: • IMP_SPEC_ERR bit is set in the "RapidIO Port x Error Detect CSR" • Port write can be generated (if enabled) • Interrupt can be generated (if enabled).
Parity error on LUT entry	Packet header is recorded in the error capture registers, packet discarded • IMP_SPEC_ERR bit is set in the "RapidIO Port x Error Detect CSR" • Port write can be generated • Interrupt can be generated • Port Number is also captured in "RapidIO Port x LUT Parity Error Info CSR"
Writes to LUT entries through the broadcast LUT registers to shutdown ports	Silently ignored
Access to LUT with a destination ID which exceeds LUT size (greater than 511 in flat mode)	Writes silently ignored and reads return 0xF

a. When a port is shut down, all clocks are off to that port. Reading to the registers returns 0 (except for 0x158 and 0x15C which give the correct values).

b. The Port is a healthy port. Packets routed to that port is disallowed to pass through to the Link Partner. All registers are still functional and when read, return the current operational values.

c. It's the same as a powered-on port except that the Link Partner was behaving as disconnected to the port. The port is healthy and when the link partner is resurrected, the link between the port and partner is be re-established.

If a LUT entry is unmapped for a particular port or the destination ID does not match any of the LUT entry, packets are routed to the default output port, as defined by “RapidIO Port x Route Config DestID CSR” on page 306. The IMP_SPEC_ERR bit is set in the “RapidIO Port x Error Detect CSR” on page 286. Note that if the default output port is unmapped, then the packet is discarded.

2.3.8 Lookup Table Entry States

A lookup table entry can be in one of the following states: mapped, unmapped, parity error, or unprogrammed. A lookup table entry that routes packets to a port that exists within the Tsi574 is *mapped*. A lookup table entry that routes packets to a port that does not exist with the Tsi574 *unmapped*.

After any reset, all lookup table entries are undefined (an unknown state). All lookup table entries must be programmed to a known value after reset to achieve predictable operation. When a lookup table entry’s parity is incorrect, the lookup table entry is in a parity error state.

Table 2 shows the possible lookup table states.

Table 2: Lookup Table States

Lookup Table Entry State	How to get into States	Action on Packet Arrival
Mapped	A lookup table entry that routes packets to a port that exists within the Tsi574 is *mapped*.	Packet is routed to the specified output port
Unmapped port value	A lookup table entry that routes packets to a port that does not exist with the Tsi574 is *unmapped*.	Default port is used for routing the packet Note: The default port is defined in RIO Route LUT Attributes CSR (see “RapidIO Route LUT Attributes (Default Port) CSR”).
Unmapped default port value	A lookup table entry that routes packets to a port that does not exist with the Tsi574 is *unmapped*.	• Packet Header recorded in error capture registers • Packet discarded • IMP_SPEC_ERR bit is set in the RIO Port x Error and Status CSR • Port write can be generated (if enabled) • Interrupt can be generated (if enabled)

Table 2: Lookup Table States

Lookup Table Entry State	How to get into States	Action on Packet Arrival
Parity Error	When a lookup table entry's parity is incorrect, the lookup table entry is in a parity error state.	• Packet Header recorded in error capture registers • Packet is discarded • IMP_SPEC_ERR bit is set in the RIO Port x Error and Status CSR • Port write can be generated • Interrupt can be generated • Port number is captured in RIO Port x LUT Parity Error Info CSR (see "RapidIO Port x LUT Parity Error Info CSR")
Unprogrammed (Undefined)	After reset all lookup table entries must be programmed to a known value to achieve predictable operation.	Non-deterministic operation can occur

When a port value for a lookup table entry is unmapped, the default port is used for routing the packet as defined in RIO Route LUT Attributes CSR register. If the default port value is unmapped, packets routed using the default port value are discarded and the IMP_SPEC_ERR bit is set in the "RapidIO Port x Error Detect CSR" on page 286.

Lookup table entries can be programmed through the standard RapidIO compliant interface or through a IDT-specific interface.

A LUT parity error can also be left over from initializing the LUTs. The Tsi574's design always checks the routing table entry for routing maintenance packets it receives, even though when the hop count is zero the receiving port is automatically used to return the response. If the destID in the maintenance packet is not a programmed LUT entry, though the routing table entry is not used to determine where the response packet is to be sent, parity errors are still detected and flagged if they occur. Since the LUTs power up in a random state, the occurrence of a LUT_PAR_ERR will be a random occurrence until all LUT entries are programmed with values to support all destIDs that the switch encounters.

2.4 Maintenance Packets

Maintenance packets are handled differently than other packets by the Tsi574. In a system the Tsi574 can be the destination of the maintenance packet.

Maintenance packet processing is based on the maintenance packet's hop count value. The hop count value controls how many hops the maintenance packet travels before it reaches its destination. The routing of the maintenance packet is controlled by the destination ID of the packet, the lookup table, and other values programmed in the intervening devices.

Ensure the destination IDs of the maintenance packet does not match the destination ID of a multicast packet. If there is a match, system behavior is undefined.

If a maintenance packet has a hop count greater than zero, the Tsi574 decrements the hop count, recalculates the CRC, and routes the packet out the port selected by the LUT. For this reason, all maintenance packets must contain routeable source and destination addresses and the routing LUT must be programmed to route both the maintenance transaction and its response.

If a maintenance read or maintenance write request packet has a hop count of 0, the port processes the maintenance request and sends a maintenance response packet. The maintenance request is passed to the register bus as a read or write transaction, an address offset, and any data associated with the request. The maintenance response packet is generated by the Tsi574 using the success or failure of the access and data from a read operation. CRC is computed and the packet is enqueued for transmission on the port that received the maintenance request.

Each port can have only one outstanding maintenance request at a time. A maintenance request received while another maintenance is being processed is retried by the RapidIO port.

The Tsi574 supports 4 byte maintenance requests only. With hop count equals 0, any Maintenance Requests larger than 4 bytes, as well as maintenance packets that are not read or write requests, are dropped and an error is noted in the IMP_SPEC_ERR bit in the "RapidIO Port x Error Detect CSR" on page 286. Examples of maintenance packets that are dropped are maintenance response and port-write packets received with a hop count of 0.

Table 3: Examples of Maintenance Packets with Hop Count = 0 and Associated Tsi574 Responses

Transaction Type	Size Field	Action taken by Tsi574	Error Logging	Notes
Read or Write Request	4 bytes	Response generated with status ok	N/A	Accepted address space = 00000 to 1FFFF
	4 bytes	Send Maintenance Response with Status Error (0111)	N/A	Address space specified > 1FFFF
	≠ 4 bytes	Send Maintenance Response with Status Error (0111)	N/A	Not supported by Tsi574

Table 3: Examples of Maintenance Packets with Hop Count = 0 and Associated Tsi574 Responses

Transaction Type	Size Field	Action taken by Tsi574	Error Logging	Notes
Write Request with no payload	Do not care	Send Maintenance Response with Status Error (0111)	N/A	Erred Write Request
Read Request with payload	Do not care	Send Maintenance Response with Status Error (0111)	N/A	Erred Read Request
Write Request with payload	4 bytes	Send Maintenance Response with Status Error (0111)	N/A	Size field reports incorrect payload size
Write Response • Hop count is 0	Do not care	Send port-write and set interrupt, if enabled	Bit 8 in "RapidIO Logical and Transport Layer Error Detect CSR"	Tsi574 is not an endpoint device
Read Response • Hop count is 0	Do not care	Send port-write and set interrupt, if enabled	Bit 8 in "RapidIO Logical and Transport Layer Error Detect CSR"	Tsi574 is not an endpoint device
Port Write • Hop count is 0	Do not care	Send port-write and set interrupt, if enabled	Bit 9 in "RapidIO Logical and Transport Layer Error Detect CSR"	Tsi574 is not an endpoint device
Reserved Transaction Type • Hop count is 0	Do not care	Send port-write and set interrupt, if enabled	Bit 4 in "RapidIO Logical and Transport Layer Error Detect CSR"	Tsi574 is not an endpoint device
Reserved TT Type • Hop count is 0	Do not care	Send port-write and set interrupt, if enabled	Bit 4 in "RapidIO Logical and Transport Layer Error Detect CSR"	Tsi574 is not an endpoint device

2.5 Multicast Event Control Symbols

Multicast-Event Control Symbol (MCS or MCES) forwarding describes the process where an MCS received on one RapidIO port is propagated out other RapidIO ports.

When a RapidIO port receives an MCS, it signals all other ports of the fact that an MCS was received. Each port can optionally transmit an MCS when it is notified that an MCS has been received by another port. A port is forwarded an MCS when the MCS_EN bit is set in the "RapidIO Serial Port x Control CSR" on page 273.

Multicast-Event control symbols (MCS) are explained in detail in the *RapidIO Interconnect Specification (Revision 1.3)*.

2.5.1 MCS Reception

When a RapidIO port receives a MCS, it does the following:

- Raises an interrupt that can be masked.
  - Interrupts are masked when the port's MCS_INT_EN bit is set in "RapidIO Port x Mode CSR" on page 302
- Forwards the symbol to all the other RapidIO ports
  - Each port then forwards the control symbol if its MCS_EN field in the "RapidIO Serial Port x Control CSR" on page 273 is set to 1.

Per-port interrupt status appears in the MCS field in the "RapidIO Port x Multicast-Event Control Symbol and Reset Control Symbol Interrupt CSR" on page 304. Additionally, the logical OR of all per-port Multicast Event interrupt status is available in both the MCS field in the "RapidIO Port x Multicast-Event Control Symbol and Reset Control Symbol Interrupt CSR" on page 304 and the MCS field in the "Global Interrupt Status Register" on page 377.

Interrupts can be cleared, either per-port or for all ports, by writing 1 to the MCS field in the "RapidIO Port x Multicast-Event Control Symbol and Reset Control Symbol Interrupt CSR" on page 304.

Additionally, the MCES pin in the Tsi574 can output an edge when an MCS is received. The MCES pin toggles to signal an MCS is received; that is, the first MCS causes the pin to go low and the second MCS causes it to go back high. The ports that are used as the source for toggling this MCES pin is selectable using the MCS_INT_EN bit in the "RapidIO Port x Mode CSR" on page 302. To select the MCES pin as an output, set the MCES_CTRL to 10 in the "MCES Pin Control Register" on page 382.

Due to the finite time it takes to translate an MCS to a signal on the MCES pin, the minimum time between any two MCS received in the Tsi574 is 500 ns. The second MCS can be lost if this condition is not met.

2.5.2 Generating an MCS

The Tsi574 supports the generation of an MCS in two ways. The first method is called the *software usage model* which use of a maintenance write transaction in a port (see "RapidIO Port x Send Multicast-Event Control Symbol Register" on page 314). The write operation to this port does not complete — that is, no response is sent — until the MCS is enqueued for transmission. Subsequent writes to the register are ignored until the MCS is transmitted. A register write can also be performed from both JTAG and I^2C.

The Tsi574 also supports a *hardware usage model*, which generates an MCS using the MCES pin as an input. When enabled, a transition on the MCES pin signals all ports that a request to transmit an MCS is received. All ports enabled to forward multicast control symbols then transmit an MCS (see "RapidIO Serial Port x Control CSR" on page 273). The minimum time between two transitions on the MCES pin is 1µs. For example, when the host needs to create a "*heartbeat*" for the entire system at 125kHz, it should use a 62.5kHz clock to generate the pulse driving the MCES pin.

MCES_CTRL setting should be completed before traffic starts. Changing the MCES_CTRL setting during operation can result in the transmission of spurious MCES.

2.5.3 Restrictions

Only one port on the Tsi574 should be assigned to receive Multicast-Event control symbols.

If multiple ports receive Multicast-Event control symbols closely spaced in time, or if a single port receives multicast control symbols spaced closely in time, only one control symbol is forwarded correctly. The other control symbols are discarded. The minimum separation between MCS is the time, on the port with the lowest possible aggregate baud rate, to send at least 64 code groups. The 64 code groups is taken from the lowest clock speed (port rate) in the system.

2.6 Reset Control Symbol Processing

One of the functions that can be performed by control symbols is requesting that the link partner reset itself. The Tsi574 can generate link-request/reset control symbols using the standard RapidIO registers defined for the purpose. The Tsi574 generates four link-request/reset control symbols with only one register access to the “RapidIO Serial Port x Link Maintenance Request CSR” on page 265. For more information on reset control symbol handling, see“Resets” on page 207.

2.7 Data Integrity Checking

Data integrity checking is performed on both control symbols and packets.

2.7.1 Packet Data Integrity Checking

Packets have two locations where CRC can occur. The first location is 80 bytes into the packet while the second location is at the end of the packet. This means that packets 80 bytes or smaller in size have only one CRC, while packets larger than 80 bytes have two 16 bit CRC codes. With the exception of maintenance packets, the Tsi574 does not (re)compute CRC codes for packets. The CRC code is forwarded with the packet across the ISF, and the packet is transmitted with the same CRC code it was received with. This ensures that packet corruption within the Tsi574 is detected.

The exception to the rule for CRC codes is the handling of maintenance packets. Maintenance packets have a hop count field, covered by CRC, which must be changed by the Tsi574 if the packet is to be forwarded. So, CRC is recomputed for maintenance packets for each link they traverse.

2.7.2 Control Symbol Data Integrity Checking

Control symbols have 24 bits, five of which are devoted to a CRC code. The CRC code is verified to ensure that the control symbol was not corrupted in transmission. Additional checks are performed on a control symbol’s fields to ensure that they are valid. If the CRC check or the control symbols fields are invalid, the control symbol is discarded.

2.8 Error Management

The Tsi574 supports the Software Assisted Error Recovery registers as defined by the *RapidIO Interconnect Specification (Revision 1.3)*. Refer to “RapidIO Physical Layer Registers” on page 261 for the complete list of registers supported for Software Assisted Error Recovery.

2.8.1 Software Assisted Error Recovery

The software-assisted error recovery process is described in terms of the ackIDs of a Tsi574 port connected to a link partner that becomes mismatched. A system host[1], which can be local or remote to the Tsi574 switch, has access to the device through another port. The system host can be any processor in a system that is tasked with error management responsibility. In a large system, multiple processors may have this responsibility. The link partner is assumed to be register compliant to the *RapidIO Interconnect Specification (Revision 1.3)*. All transactions between the system host and the Tsi574 switch are maintenance transactions.

Before, during, and at the conclusion of the process, monitor and clear any error bits that were set in the “RapidIO Port x Error and Status CSR” on page 270.

If an ackID mismatch occurs between a Tsi574 switch port and an endpoint, the system host manipulates registers in the Tsi574 switch port connected to the endpoint to perform error recovery. If this occurs, the following software-assisted error recovery process can be used:

1. The system host sets the PORT_LOCKOUT bit in the Tsi574’s “RapidIO Serial Port x Control CSR” on page 273 in order to flush the port’s ingress and egress buffers. The PORT_LOCKOUT must be asserted for 50 microseconds to guarantee that all packets are flushed.
2. The system host writes and clears the PORT_LOCKOUT bit on in order to perform a maintenance transaction to the link partner.
3. The system host reads the Tsi574’s “RapidIO Serial Port x Local ackID Status CSR” on page 268 and makes note of the inbound, outbound, and outstanding ACK_IDs.
4. The system host instructs the Tsi574 to generate a link request to its link partner using the “RapidIO Serial Port x Link Maintenance Request CSR” on page 265.
5. The system host reads the link partner's response in the Tsi574's “RapidIO Serial Port x Link Maintenance Response CSR” on page 267.
6. The system host sets the switch's outbound ACK_ID value to match the value in the ACK_ID_STAT field of the“RapidIO Serial Port x Link Maintenance Response CSR” on page 267. The ACK_ID_STAT indicates the link partner's next expected ACK_ID.
7. The system host sends a maintenance write with a priority 2 to the link partner. The maintenance write updates the link partner’s ACK_ID status register with a new OUTBOUND value that matches the Tsi574’s INBOUND value, and an INBOUND value which is incremented by 1 compared to the value returned in step 5. The values must be updated before the link partner sends its maintenance response so the response has the correct ACK_ID.
   - If the link partner's implementation is such that the ackID is not updated before the maintenance response is issued, the SEMP must wait until the transaction times out (through the TVAL timer), re-issue the link request and compare again the Tsi574 port's “RapidIO Serial Port x Local ackID Status CSR” on page 268 values to those in the “RapidIO Serial Port x Link Maintenance Response CSR” on page 267.
   - The SEMP should send another link request from the Tsi574 to verify that the ACK_IDs are the same.

1. This type of system host is sometimes referred to as a System Error Management Processor (SEMP).

2.9 Hot Insertion and Hot Extraction

Hot insertion and hot extraction functionality enables reliable systems to safely add, remove, and replace components while the system continues to operate. The system host can use the Tsi574's capability to restrict the access of a newly inserted component to prevent a faulty component from negatively affecting the system.

The following bit fields in "RapidIO Serial Port x Control CSR" control access to the system:

- PORT_LOCKOUT: When this bit is set, only link request/response control symbols can be exchanged. When the PORT_LOCKOUT bit is cleared, access is controlled by OUTPUT_EN and INPUT_EN.
  When a PORT_LOCKOUT bit is set and the link is initialized, a port write can be sent periodically to notify the system host that a new component has been added to the system.
- OUTPUT_EN: Controls whether packets other than maintenance requests/responses may be sent by the Tsi574.
- INPUT_EN: Controls whether packets other than maintenance requests/responses may be received by the Tsi574.

In "RapidIO Port x Interrupt Status Register":

- LINK_INIT_NOTIFICATION: This is an interrupt bit. When the PORT_LOCKOUT bit is set, this bit indicates that the link has been successfully initialized. This is an interrupt bit and to disable the generation of an interrupt, set LINK_INIT_NOTIFICATION_EN to 0 in "RapidIO Port x Control Independent Register". A port write can be sent if the link is initialized.

The LINK_INIT_NOTIFICATION_GEN can force a LINK_INIT_NOTIFICATION interrupt to be generated through the "RapidIO Port x Interrupt Generate Register",. This is useful in software testing and integration.

In "RapidIO Port x Error and Status CSR":

- PORT_OK: Indicates when a port is functioning and can carry traffic.
- PORT_UNINIT: When the port is not initialized, this bit is set.

The lookup tables (LUTs), although not necessary, can also be used to ensure that no traffic is being routed to the component being inserted/removed. For more information on lookup table functionality, see "Lookup Tables".

2.9.1 Hot Insertion

When Hot Insertion occurs at Port#N, the following steps should be completed:

1. Power up the Port#N in Tsi574.
2. Lock out Port#N by writing 1 to PORT_LOCKOUT in "RapidIO Serial Port x Control CSR".
3. Insert the card.

   Re-initialization occurs and a port-write is received once both sides are synchronized.
4. Clear Input Error-Stop state errors in "RapidIO Port x Error and Status CSR".

   Only if extraction happens on the same Port#N
5. Send Link Request to clear Input Error-Stop states to Link Partner.

   Only if extraction happens on the same Port#N
6. Re-synchronize the inbound and outbound ackIDs.

   The system host inquires about the link partner's Inbound/Outbound ackIDs and re-programs the Tsi574's ackIDs accordingly (see "RapidIO Serial Port x Local ackID Status CSR").

Tsi574 ports on which a component insertion event can occur can be configured to notify the system host when this event occurs. The PORT_LOCKOUT bit must be set to allow the LINK_INIT_NOTIFICATION bit in the "RapidIO Port x Interrupt Status Register" to be set. To determine that a component insertion event has occurred, the system host has the option of polling the "RapidIO Port x Interrupt Status Register", or of setting the LINK_INIT_NOTIFICATION_GEN bit in the "RapidIO Port x Control Independent Register" to assert an interrupt or send port write transactions (see "RapidIO Port x Control Independent Register").

Once the system host is notified that a new component is inserted, the LINK_INIT_NOTIFICATION bit should be cleared in the "RapidIO Port x Interrupt Status Register" to stop the assertion of interrupts.

If multiple ports become active simultaneously, only one port write is generated. For more information, see "Port-write Notifications" on page 131.

The PORT_LOCKOUT bit must be cleared to allow the system host to access the new component and to allow the new component to access the remainder of the system. The OUTPUT_EN and INPUT_EN bits must be set according to the amount of access the system designer requires to allow the new component to be brought into the system safely. Error notification for the link should also be enabled, if required by the system designer.

Before any packets can be exchanged, the OUTBOUND field in "RapidIO Serial Port x Local ackID Status CSR" must be programmed to match the INBOUND value of the other side of the link. The link partner's next expected inbound ackID value is determined by issuing a link request to the link partner, and examining the ackID field of the link response that the link partner returned. Similarly, the OUTBOUND value for the component that was just inserted must be programmed to match the INBOUND value of the Tsi574's port, contained in "RapidIO Serial Port x Local ackID Status CSR".

The next expected inbound and next outbound ackIDs of the link partner are determined through the use of link request/response control symbols.

As with a controlled reset of a link partner (see "Generating a RapidIO Reset Request to a Peer Device"), the writes of the two OUTBOUND values must occur in the order given (for example, the Tsi574 followed by the link partner).

If the requests are performed in the reverse order, or if other packets are transmitted before the OUTBOUND values are programmed, the link experiences a fatal error due to an ackID mismatch.

2.9.1.1 Link Partner and Unsupported Error Recovery

In the event that the link partner does not support the software-assisted error recovery registers, "RapidIO Serial Port x Local ackID Status CSR" will not exist in the link partner. Since it is impossible to set the link partner's OUTBOUND value in this case, the Tsi574 INBOUND value must become zero.

2.9.2 Hot Extraction

Tsi574 ports where a hot extraction event occurs should not have any transactions flowing through them in preparation for the extraction. The PORT_LOCKOUT bit must be set on the port where the hot extraction event occurs in order to drop all packets arriving from the ISF for transmission, to flush any existing packets in the transmit and receive queues of the port, and to prevent new packets from being received from the device about to be extracted. At this point, the component can be safely extracted.

The LUT entries for all ports in the Tsi574 can be configured to not route any packets to the port on which the hot extraction occurs.

When hot extracting a port (Port#N) originally connected to the Tsi574, the following steps should be completed:

1. Lock out Port#N by writing 1 to PORT_LOCKOUT in "RapidIO Serial Port x Control CSR".
2. Extract the card.

   This causes Port #N to lose synchronization. After the Lane Sync Timer has expired, a PORT_ERR status bit may be asserted in the "RapidIO Port x Error and Status CSR". PORT_UNINIT is set and PORT_OK is de-asserted in the same register. INPUT_ERR_STOP and OUTPUT_ERR_STOP are also set.

If Port #N is an odd port, a PORT_ERR condition may occur on the even port of the Tsi574 MAC.

2.9.3 Hot Extraction System Notification

System designers may require confirmation of when a component is extracted. The following sections describe the confirmation methods supported by the Tsi574.

2.9.3.1 Polling

The system can poll the PORT_OK and PORT_UNINIT bits in the "RapidIO Port x Error and Status CSR" for indication that the link partner is no longer present.

2.9.3.2 Interrupts and Port Writes

Interrupts and/or port writes can be implemented as part of the hot insertion and hot extraction process. For example, while the PORT_LOCKOUT bit in the "RapidIO Serial Port x Control CSR" and the LINK_INIT_NOTIFICATION bit in the "RapidIO Port x Interrupt Status Register" are set, interrupts are asserted (if LINK_INIT_NOTIFICATION_EN is set in the "RapidIO Port x Control Independent Register") until the component is extracted. A port write can also be sent once whenever a link is initialized and the port is locked out or when the port is locked out, and the link re-acquires initialization.

2.9.3.3 Link Errors

Another notification implementation makes use of the link errors that occur when a component is extracted. In this design, the PORT_DISABLE bit in"RapidIO Serial Port x Control CSR" is set to 1 by software and the LINK_INIT_NOTIFICATION bit in"RapidIO Port x Interrupt Status Register" should be cleared to 0. Error notification by the LINK_INIT_NOTIFICATION_EN in the "RapidIO Port x Control Independent Register" continues to be enabled. When the component is removed, lane synchronization and/or lane alignment is lost. The errors detected cause a port write and/or interrupt to be sent to the system host, indicating that a component may have been extracted.

2.10 Loss of Lane Synchronization

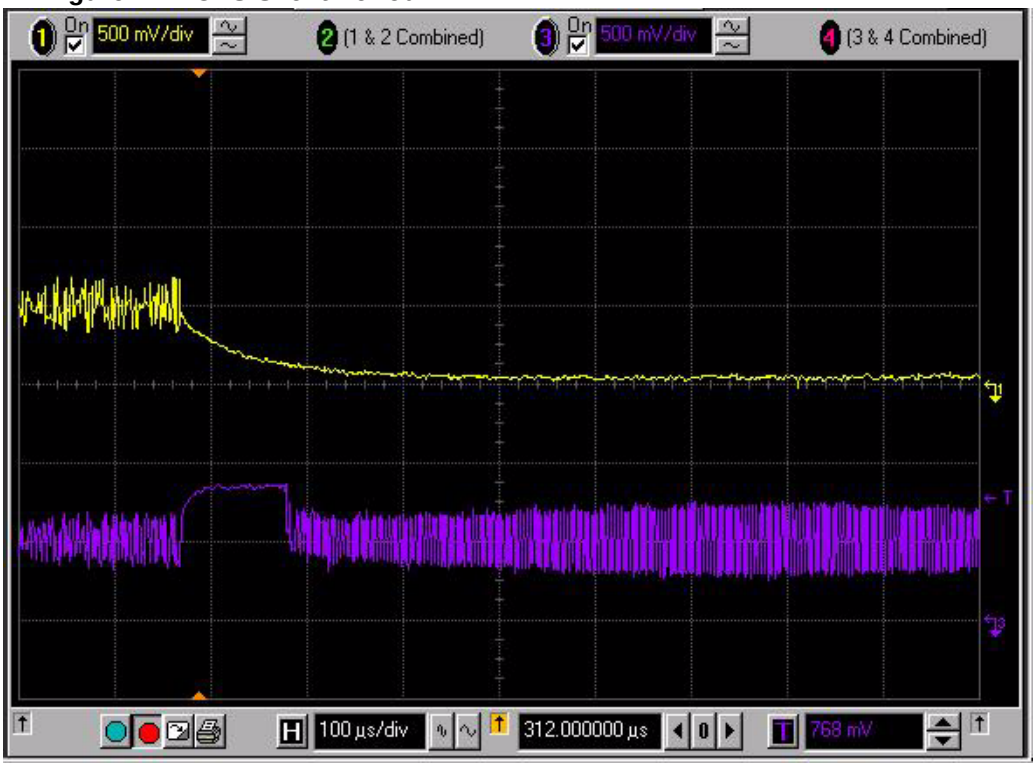
A loss of lane synchronization (LOLS) can occur due to high error rates on a link, reset of a link partner, or hot extraction of a link partner. This section discusses with LOLS recovery related to high error rates on a link. For an explanation of LOLS handling due to reset of a link partner, see "Generating a RapidIO Reset Request to a Peer Device" on page 209. For a discussion of LOLS handling due to hot insertion or hot extraction of a link partner, see "Hot Insertion and Hot Extraction" on page 59.

When the Tsi574 detects a LOLS, it attempts to regain synchronization and recover so that no packets are lost, duplicated, or unnecessarily retransmitted. This is in compliance with the *RapidIO Interconnect Specification (Revision 1.3)*. To guarantee that no packets are lost, ensure that the duration of the packet time-to-live timer is programmed to be greater than the duration of the port's silence timer.

When a Tsi574 port detects LOLS, it restarts its synchronization state machine and stops its Timeout Interval Value (TVAL) timer for expected packet and control-symbol acknowledgements. The Tsi574 port is in input-error stopped state due to errors seen on the link. For the duration of the timer, see the TVAL field in the "RapidIO Switch Port Link Timeout Control CSR" on page 263. The packet time-to-live timer is not stopped.

Figure 11 shows the Tsi574 entering the silence period when it experiences the loss of signal from its link partner.

Figure 11: LOLS Silent Period



Once synchronization is re-acquired, the Tsi574 transmitter resumes all timers and resumes sending packets from the next un-sent packet in its transmit queue, using the next available ackID. The transmitter handles the LOLS event as a temporary interruption that is completely ignored from the perspective of packet transfers and control symbol transfers; the actual duration of the LOLS condition has no impact on the process once the link is re-acquired.

Any packets transmitted to the Tsi574 are not acknowledged because the port is in input-error stopped state. The link partner times out waiting for a packet acknowledge control symbol, and enters the output-error stopped state. To recover, the link partner sends a link-request/input-status control symbol to the Tsi574 port. This clears the input-error stopped state on the Tsi574.

The Tsi574 responds to its link partner's link-request/input-status control symbol with a link-response/status control symbol. The link partner accepts the symbol and exits the output-error stopped state. The packet associated with the next expected ackID contained in the link-response/status control symbol (if any) is then retransmitted and accepted by the Tsi574.

2.10.1 Dead Link Timer

When a LOLS event occurs, the loss of communication can continue for an extended length of time. For example, there may be an uncontrolled extraction of the link partner, and a hardware fault on the link partner. Packets continue to be directed to the non-functional (dead) link, but are not able to make forward progress. As a result, this may eventually block every traffic path in the system.

To enable systems to robustly deal with dead links, the Tsi574 has a “dead link timer” feature. This is a proprietary function that is outside of the RapidIO specification. The DLT_EN and DLT_THRESH fields in the “SRIO MAC x Digital Loopback and Clock Selection Register” enable/disable the dead link timer and specify the duration of the dead link timer. There is one DLT for each pair of ports (Ports N and N+1). The dead link timer can be disabled by setting DLT_EN to 0.

When the dead link timer is enabled (in the DLT_EN bit in the “SRIO MAC x Digital Loopback and Clock Selection Register”) and a link failure causes the timer to expire, it is reported in the PORT_ERR bit in the “RapidIO Port x Error and Status CSR”. When the PORT_ERR bit is set, and port-writes are enabled, a port-write is generated.

If the dead link timer expires, which the link is no longer able to transmit or receive, then the port starts removing the impact of the dead link partner from the system. The port drops all packets in its transmit buffers. Any new packets that are transferred to the port from the ISF are accepted and dropped. Packets received by the port from its link partner can still be forwarded to the ISF.

The dead link timer register fields affect the RapidIO ports (Ports N and N+1) sharing the Tsi574 MAC.

2.10.2 Lane Sync Timer

Supplementary to the Dead Link Timer is the Lane Sync Timer (LST). There is one LST for each lane of the Tsi574 MAC. The LST for a lane starts when lane sync is lost after a link has successfully initialized. When the LST expires for any lane on a port, the PORT_ERR bit is set in the “RapidIO Port x Error and Status CSR”.

When the Tsi574 MAC is operating in two 1x mode and the LST expires on the odd port, the even port will detect a spurious PORT_ERR.

The LST is a constant 0xFFFFF symbol periods for a link. The timeouts for different lane speeds are given in the following table:

Lane Speed	Timeout (nsec)
1.25	8,388,600
2.5	4,194,300
3.125	3,355,440

Note that only the PORT_ERR status bit indicates that an LST has expired.

3. Serial RapidIO Electrical Interface

This chapter describes the IDT-specific electrical layer features of the Tsi574 Serial RapidIO Electrical Interface. See the "Serial RapidIO Interface" for a description of the standards-defined RapidIO features common to all RapidIO ports.

This chapter includes the following information:

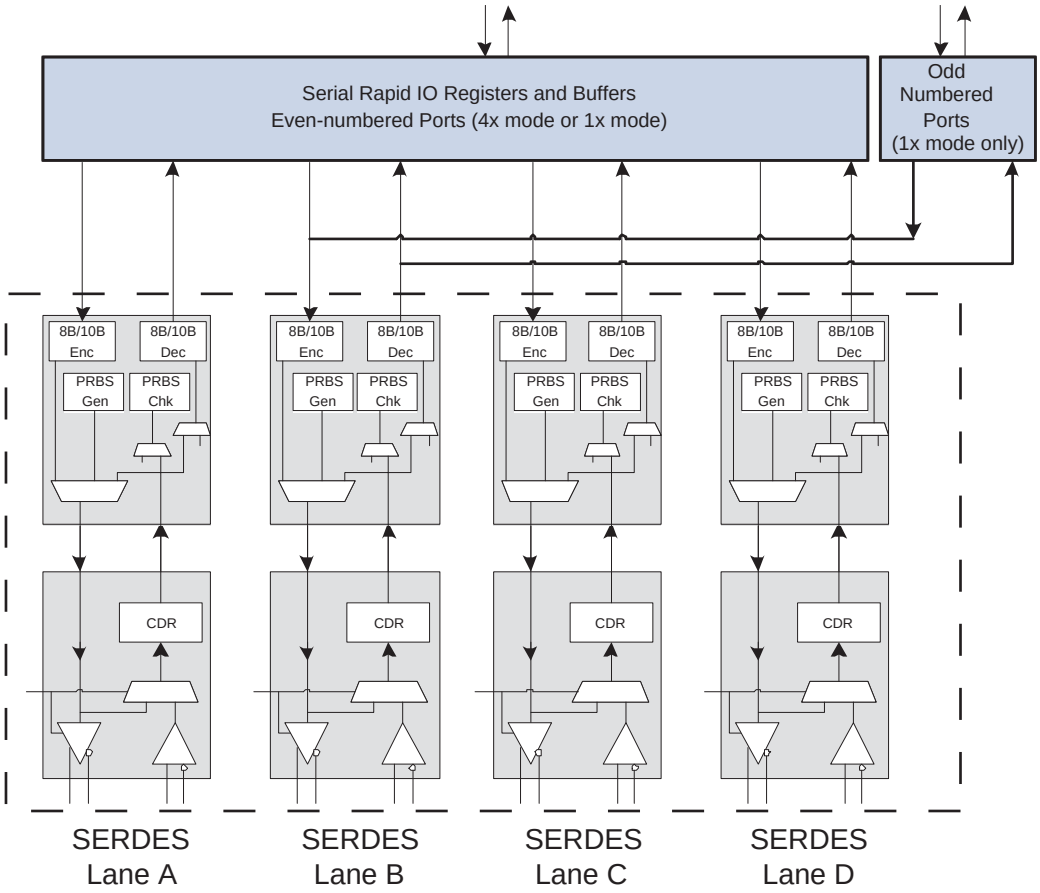
3.1 Overview

The Tsi574 has four Media Access Controllers (MAC) comprising the 8 Serial RapidIO ports. The 8 ports are grouped into pairs consisting of one even numbered port and one odd numbered port. Each port has flexible testing features including multiple loopback modes and bit error rate testing. Each pair of ports share four differential transmit lanes and four differential receive lanes.

Even and odd number ports have different capabilities. Even numbered ports can operate in either 4x or 1x mode, while odd numbered ports can only operate in 1x mode. When the even numbered port is operating in 4x mode, it has control over all four differential pairs (designated Lanes A, B, C and D). In 4x mode, the default state of the odd numbered port is powered on. All registers in the even and odd numbered port are accessible but the odd numbered port does not have access to the PHY. In order to decrease the power dissipation of the port, the odd numbered port can be powered down in this configuration. When the even numbered port is operating in 1x mode it uses only Lane A and the odd numbered port is permitted to operate in 1x mode using Lane B.

The Tsi574 MAC and SerDes interconnect block diagram is shown in the following figure.

Figure 12: Tsi574 MAC Block Diagram



Each serial RapidIO MAC includes the following features:

- One port in 4x Serial mode
- Two ports in 1x Serial mode (each 4x port can be configured as two 1x ports)
- RapidIO standard operating baud rate per data lane: 1.25 Gbit/s, 2.5 Gbit/s, or 3.125 Gbit/s
  - 12.5 Gbit/s inbound and 12.5 Gbit/s outbound bandwidth at 3.125 Gbps for a port configured for 4x mode
  - 3.125 Gbit/s inbound and 3.125 Gbit/s outbound bandwidth at 3.125 Gbps for a port configured for 1x mode
- Adjustable receive equalization that is programmable per lane
- Serial loopback with a built-in testability
- Bit error rate testing (BERT)
- Scope function of eye signals
- Hot-insertion capable I/Os and hardware support

3.2 Port Numbering

The RapidIO ports on the Tsi574 are numbered sequentially from 0 to 7. The following table shows the mapping between port numbers and the physical ports. These port numbers are used within the destination lookup tables for ingress RapidIO ports and in numerous register configuration fields .

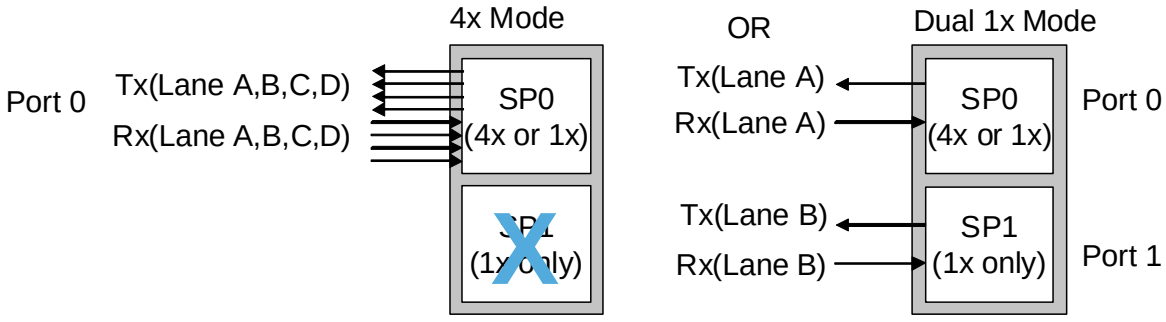
Table 4: Tsi574 Port Numbering

Port Number	RapidIO Port	Mode
0	Serial Port 0 (SP0)	1x or 4x
1	Serial Port 1 (SP1)	1x
2	Serial Port 2 (SP2)	1x or 4x
3	Serial Port 3 (SP3)	1x
4	Serial Port 4 (SP4)	1x or 4x
5	Serial Port 5 (SP5)	1x
6	Serial Port 6 (SP6)	1x or 4x
7	Serial Port 7 (SP7)	1x

3.2.1 Port Configuration

Ports that operate in either 4x or 1x mode can be configured as either one 4x mode port or dual 1x mode ports. For example, SP0 can be configured as either one 4x mode port, or Port 0 and Port 1 can be dual 1x mode ports.

Figure 13: Port Configuration



3.3 Port Aggregation: 1x and 4x Modes

The RapidIO ports on the Tsi574 are grouped into pairs that share the same MAC. The MAC provides the PMA/PCS encoding/decoding layers, as well as the RapidIO physical, transport and logical layer functionality required of a RapidIO switch device.

The MACs are numbered by even numbers and support the ports in the following manner: serial ports 0 and 1 use MAC 0, ports 2 and 3 use MAC 2, etc. Ports are grouped into pairs of N and N+1, where N is even.

Two configurations are possible on each 4x mode-capable port:

- Both port N and port N+1 can operate in 1x mode (the 1x + 1x configuration)
- Port N can operate in 4x while port N+1 is unused and can be powered down (the 4x + 0x configuration)

1x mode means that one physical SerDes lane is used between link partners, and *4x mode* means that four physical lanes are used between link partners. 4x mode offers four times the bandwidth as 1x mode at the same baud rate.

Each Tsi574 MAC has an external pin called SPx_MODESEL. This pin can be pulled high to configure the MAC for either 1x + 1x mode or pulled low for 4x + 0x mode (see “Signals” on page 213). These pins are sampled after reset is de-asserted. To ensure that the pins are sampled correctly, the pins must be stable at the release of reset, and held at a stable level for 10 clock cycles after reset is de-asserted. The sampled state of the pins is reflected in the PORT_WIDTH field in the “RapidIO Serial Port x Control CSR” on page 273.

After reset, the configuration mode can be re-programmed by changing the MAC_MODE field in the “SRIO MAC x Digital Loopback and Clock Selection Register” on page 369 and the programmed value overrides the pin setting of SPx_MODESEL. Changes to the MAC_MODE field that are different than that set by the SPx_MODESEL pin must be programmed after a hardware or software reset with a register write in order to restore the desired condition.

The PWRDNx4 and PWRDNx1 bits must both be asserted prior to changing the state of the MAC_MODE bit. Therefore, changing the MAC operation from x4 to two x1 or from two x1 to x4 operation requires that the ports both be powered down using the PWRDNx4 and PWRDNx1 bits, and then powered back up with the new setting of the MAC_MODE bit.

A port’s operation is not affected if the SPx_MODESEL signal values are changed after they have been sampled at reset release.

The port width in use can be different from the pin-selected width; the pin indicates what the port was set to operate at, while the registers show what it is actually operating at. An even port with the capability to function in either 1x or 4x mode port can be downgraded to a 1x mode port when faults on lanes prevent operation in 4x mode. Additionally, the port width can be overridden through register programming and changed into operating at a different port mode. Refer to “RapidIO Serial Port x Control CSR” on page 273 for status and control fields for port width and “4x Configuration” on page 69 for downgraded port configuration.

3.3.1 1x + 1x Configuration

When the 4x mode-capable port in a Tsi574 MAC is configured to operate in 1x mode, the odd-numbered port in a MAC can also be used in 1x mode. In this configuration, the even-numbered port always uses SerDes lane A and the odd-numbered port always uses SerDes lane B.

The two ports that share the same MAC also share the same transmit clock, which means the two ports must have the same bit rate. To select the bit rate, write the IO_SPEED field (see "SRIO MAC x Digital Loopback and Clock Selection Register" on page 369), as described in "Clocking" on page 69. The initial clock rate is selected by the global power-up option for all ports.

3.3.2 4x Configuration

When the even-numbered port in a Tsi574 MAC is configured to operate in 4x mode (for example port 0), the odd-numbered port in a MAC (for example port 1) cannot be used and the register values for the odd-numbered port should be ignored. To save power, the odd-numbered port can be powered down (see "Port Power Down" on page 71).

The unusable, odd-numbered port is still a part of the Tsi574's memory map. However, system software must be aware that the port is not usable and that its per-port registers should not be accessed. If the port is accessed the Tsi574's behavior is undefined. Refer to "Port Power Down" on page 71 for more details on register behavior under power down conditions.

The even-numbered port configured for 4x mode follows the link-width negotiation rules outlined in the *RapidIO Interconnect Specification (Revision 1.3)*. Depending on the configuration or capabilities of the link partner, or on the quality of the connection, it is possible that a port configured for 4x mode actually operates in 1x mode on either SerDes lane A or C. Under this scenario, the degraded port can not be configured to an 1x + 1x mode.

System software can force a downgrade in port mode by writing the OVER_PWIDTH field on either the Tsi574 or in its link partner (see "RapidIO Serial Port x Control CSR" on page 273). The current operating link width is available in the INIT_PWIDTH field. Software may need to manage ackID recovery for the link partner when changing port usage between lanes A and C.

It is necessary to know if the link partner can continue to communicate when changing the port width between Lanes A and C. Refer to the "RapidIO Serial Port x Control CSR" in the link partner to determine the capability of the link partner.

3.3.2.1 Degraded Link Mode

When a 4x port has degraded to a 1x mode, software may attempt to recover to 4x mode by using the FORCE_REINIT bit in the "RapidIO Port x Control Independent Register" on page 311.

Connecting four 1x links to a 4x port is not supported. Doing so results in the port failing to achieve lane alignment.

3.4 Clocking

Serial RapidIO ports use source clocked transmission; the clock is embedded in the data stream using 8B/10B encoding. The Tsi574 recovers the embedded clock in the received data stream and generates a separate clock (based on S_CLK) to transmit its own data.

The Tsi574 uses only one external differential clock source (S_CLK_P/N) as the reference to generate all internal clocks for processing the data. When the frequency of the reference clock is set at 156.25 MHz, Tsi574 can support three different RapidIO standard signaling rates (3.125 Gbps, 2.5 Gbps, and 1.25 Gbps). Table 5 shows the port speeds and bandwidths supported by the Tsi574. For more information on clocking refer to "Clocks, Resets and Power-up Options" on page 203 and "Clocking" on page 475.

Table 5: Reference Clock Frequency and Supported Serial RapidIO Data Rates

Reference Clock Frequency (S_CLK_p/n)	Supported Data Rate	SP_IO_SPEED[1:0] Setting	Default Speed for all Ports	User Bandwidth (1x mode)	User Bandwidth (4x mode)
156.25MHz[a]	1.25 Gbit/s	00	1.25 Gbit/s	1.0 Gbit/s	4.0 Gbit/s
	2.50 Gbit/s	01	2.50 Gbit/s	2.0 Gbit/s	8.0 Gbit/s
	3.125 Gbit/s	10	3.125 Gbit/s	2.5 Gbit/s	10 Gbit/s
	N/A	11 (Illegal)	Undefined	Undefined	Undefined

a. For information about 125 MHz S_CLK refer to "Clocking" on page 475.

When ports in the same MAC are both operating in 1x mode, both ports operate at the same rate.

The data rate of all the ports in Tsi574 at power-up is determined by the setting of the SP_IO_SPEED[1:0] pins (see "Signal Descriptions"). There is only one pair of SP_IO_SPEED pins for the entire device, which means all RapidIO ports default to the same speed at power-up. After reset, the individual port speeds can be configured through registers (IO_SPEED in "SRIO MAC x Digital Loopback and Clock Selection Register" on page 369) or through the I^2C configuration EEPROM.

The settings of SP_IO_SPEED[1:0] pins and the reference clock used have a strict relationship. Entering an illegal setting causes unpredictable behavior of the device.

The *RapidIO Interconnect Specification (Revision 1.3)* requires the receive and transmit signals must operate at the same baud rate. This means a port must transmit at the same clock rate that it receives within +/-100 ppm.

3.4.1 Changing the Clock Speed

The following procedure changes the signaling rate of a port:

1. Set PWDN_X4 in the "SRIO MAC x Digital Loopback and Clock Selection Register" on page 369 to 1
2. Select the new clock speed using IO_SPEED in the SMACx_DLOOP_CLK_SEL register
3. Set PWDN_X4 in the SMACx_DLOOP_CLK_SEL to 0

For more information about powering down ports and special requirements for powering down port 0, see “Port Power Down” on page 71.

3.4.2 Changing the Clock Speed Through I2C

The Tsi574 can be configured to power up with ports at different link speeds by setting the “SRIO MAC x Digital Loopback and Clock Selection Register” on page 369 by an external I2C master access or by an EEPROM boot load.

Care must be taken writing this register by an I2C master because the port is initialized before the I2C load is completed and therefore must follow the same rules as outlined in “Changing the Clock Speed” on page 70.

Initializing the port speed using an I2C EEPROM boot load must also follow the rules outlined in “Changing the Clock Speed” on page 70 because the port is loaded with the power-up option selection on reset release and, although the port is prevented from initializing during the EEPROM boot load, the PLL is running.

The most effective way to configure the port link speed through the I2C register load is to leave the port powered down at boot time through the SP{n}_PWRDN configuration pin (see “Signal Descriptions”) and have entries in the I2C EEPROM to load the appropriate contents of the SMACn_DLOOP_CLK_SEL to power up the port and set the correct port speed.

If port 0 requires a different speed from the default speed, two I2C EEPROM entries are necessary because port 0 does not have a power down configuration pin. In this case, the first I2C EEPROM entry for SMAC0_DLOOP_CLK_SEL must power down the port (SMAC0_DLOOP_CLK_SEL = 0xXXXXXXXC). The second I2C EEPROM entry can power up the port and set IO_SPEED field in the SMAC0_DLOOP_CLK_SEL register to the correct value and power the port back up again.

3.5 Port Power Down

All of the Tsi574 RapidIO ports can be powered down to minimize power consumption when the port is not required. However, port 0 has special power-down requirements that must be followed (see “Special Conditions for Port 0 Power Down” on page 72).

When a port is powered down, some registers return 0 and all writes to these registers are ignored. These values indicate that the RapidIO port is uninitialized. The following register types are read only and return zero when a port is powered-down:

- RapidIO Physical Layer Registers (see “RapidIO Physical Layer Registers”)
- RapidIO Error Management Extension Registers (see “RapidIO Error Management Extension Registers”)
- IDT-Specific RapidIO Registers (see “IDT-Specific RapidIO Registers”)

Both the “RapidIO Port x Error and Status CSR” on page 270 and “RapidIO Serial Port x Control CSR” on page 273 registers return 0x00000001 when read instead of 0s.

The following register types can be read and written to when a port is powered-down:

- “Serial Port Electrical Layer Registers” on page 353

- "Internal Switching Fabric (ISF) Registers"
- "I2C Registers" on page 403
- "Utility Unit Registers" on page 377

3.5.1 Default Configurations on Power Down

When a port is powered down, the port loses configuration information that is stored for that specific port. For example, multicast settings and port write settings return to their default reset settings after a port reset (port is powered down and back up). After port reset, there is no way to determine that the configuration for a particular port is correct. The registers listed in "Global Registers to Program after Port Power Down" are imaged in every port but are accessed through one register address. Therefore, the powering down of a port removes the image of that register in that port.

For example, if a port is shut down and then restored, the port write destinationID in that port is reset to the default value. The port write destinationID must be re-written for the whole device after a port is shut down and restored. Similarly, multicast settings for the entire device must be re-written when a port is shut down and restored.

3.5.1.1 Global Registers to Program after Port Power Down

The following global registers must be re-programmed with the appropriate configuration values when a port is reset:

- "RapidIO Component Tag CSR"
- "RapidIO Route LUT Attributes (Default Port) CSR"
- "RapidIO Switch Port Link Timeout Control CSR"
- "RapidIO Switch Port General Control CSR"
- "RapidIO Logical and Transport Layer Error Enable CSR"
- "RapidIO Port-Write Target Device ID CSR"
- "RapidIO Multicast Write ID x Register"

3.5.2 Special Conditions for Port 0 Power Down

Port 0 can only be powered down through register accesses and not through the SPn_PWRDN pin. Port 0 has the has following special conditions after it is powered down:

- The list of registers in "Default Configurations on Power Down" on page 72 can not be read while port 0 is in reset
- After reset the listed registers must be re-written (like any other port that has been powered down)

3.5.3 Power-Down Options

The following power-down options are available on a port:

- A port's main logic can be powered down at boot up through the SP{n}_PWRDN pins.
- The default configuration provided by the pins can be changed using the PWDN_X4 and PWDN_X1 bits in the "SRIO MAC x Digital Loopback and Clock Selection Register". This can occur during a boot load using an EEPROM on the I^2C bus, or during normal operation through a register write.

3.5.4 Configuration and Operation Through Power-down

The transceivers for the individual bit lanes can be powered down when they are not used. All valid power-down scenarios are shown in Table 6.

Table 6: Serial Port Power-down Procedure

Mode for Serial Port *n*	Mode for Serial Port *n+1*	Required Power Down Configuration
4x	N/A	• De-assert the SP*n*_PWRDN pin and/or set the PWDN_X4 bit to 0 in the "SRIO MAC x Digital Loopback and Clock Selection Register" on page 369. • To save power, assert the SP*n+1*_PWRDN pin and/or set the PWDN_X1 bit to 1 in the "SRIO MAC x Digital Loopback and Clock Selection Register" on page 369. If this bit is not set, Port n+1 consumes unnecessary power.
1x	1x	• De-assert the SP*n*_PWRDN pin and/or set the PWDN_X4 bit to 0 in the "SRIO MAC x Digital Loopback and Clock Selection Register" on page 369. • De-assert the SP*n+1*_PWRDN pin and/or set the PWDN_X1 bit to 0 in the "SRIO MAC x Digital Loopback and Clock Selection Register" on page 369.
1x	Port Not Used	• De-assert the SP*n*_PWRDN pin and/or set the PWDN_X4 bit to 0 in the "SRIO MAC x Digital Loopback and Clock Selection Register" on page 369. • To conserve power, assert the SP*n+1*_PWRDN pin and/or set the PWDN_X1 bit to 1 in the "SRIO MAC x Digital Loopback and Clock Selection Register" on page 369. Otherwise, Port n+1 consumes power unnecessarily.
Port Not Used	1x	• Not supported
Port Not Used	Port Not Used	• To save power, assert the SP*n*_PWRDN pin and/or set the PWDN_X4 bit to 1 in the "SRIO MAC x Digital Loopback and Clock Selection Register" on page 369. • To conserve power, assert the SP*n+1*_PWRDN pin and/or set the PWDN_X1 bit to 1 in the "SRIO MAC x Digital Loopback and Clock Selection Register" on page 369.

3.5.4.1 Signals Sampled After Reset

After a hardware reset is de-asserted, the Tsi574 samples the state of the SP{n}_PWRDN pins and only powers up the ports that are enabled. Each RapidIO port has a unique pin, SPn_PWRDN.

Port 0 is the default port and can only be powered down through a direct register write.

The sampled state of the pins is available in the "SRIO MAC x Digital Loopback and Clock Selection Register" on page 369 register. This register can be overwritten at any time — during boot-up through the I^2C interface, JTAG, or during normal operation through the RapidIO interfaces — allowing the system software to override the pin-based configuration.

3.5.4.2 4x Mode and Odd Ports

When a pair of ports sharing the same MAC are configured in 4x mode, only the even-numbered port is used. The odd numbered port should be powered down by software or configuration pins to minimize power consumption.

3.6 Port Lanes

For ports that support both 1x and 4x mode functionality, even and odd number ports have different capabilities. Even numbered ports can operate in either 4x or 1x mode, while odd numbered ports can only operate in 1x mode. When the even numbered port is operating in 4x mode, it has control over all four differential pairs (designated Lanes A, B, C and D).

In 4x mode, the default state of the odd numbered port is powered on. All registers in the even and odd numbered port are accessible but the odd numbered port does not have access to the PHY. In order to decrease the power dissipation of the port, the odd numbered port can be powered down in this configuration. When the even numbered port is operating in 1x mode it uses only Lane A and the odd numbered port is permitted to operate in 1x mode using Lane B.

For more information on lanes, refer to "Lanes and Channels" on page 75.

3.6.1 Lane Synchronization and Alignment

When coming out of reset, the transmit side of the port must continuously send out /K28.5/ code groups on each lane to assist the receive side of its link partner to synchronize. Once a /K28.5/ code group is detected by the receive port, another 127 /K28.5/ code groups must be received error free before the receive port can declare that it is synchronized. No other useful information is communicated between the link partners until the ports are synchronized.

For a 4x port, after lane synchronization is complete, lane alignment starts. The port transmits /A/'s (||A||) on all four lanes, according to the *RapidIO Interconnect Specification (Revision 1.3)* idle sequence generation rules. Reception of four ||A||'s without the intervening reception of a misaligned column is the condition for achieving lane alignment. A misaligned column (that is a column with at least one ||A|| but not all ||A||s in a row) causes the alignment process to restart. Bit errors, or receptions of rows without all /A/'s, result in sampling/buffering adjustments in an attempt to achieve alignment.

For more information, see the *RapidIO Interconnect Specification (Revision 1.3)*.

3.6.2 Lane Swapping

Lane swap is the ability to reverse the order of the transmit and receive pins. The Tsi574 allows the order of the transmit and/or receive pins of each 4x port to be reversed in order to simplify board layout issues. Lane swap is only supported when the MAC is operating in 4x mode.

Lane swapping for 1x mode is not supported. If the Tsi574 port to a connector is lane swapped as 4x mode and a 1x mode device in inserted into the connector, the 4x mode port will fail-down (that is, become a 1x mode connection) and lane A is initialized. However, if the Tsi574 port to a connector is lane swapped as 4x mode and then re-configured to operate as a 1x mode, the swapping is canceled and the port will be unable to connect to a 1x mode device.

Table 7 shows the lane sequence for Tsi574's swapped 4x mode lanes, the connector, and the sequence for non-swapped 1x mode lanes.

Table 7: Lane Sequence

Swapped 4x Mode	Mated Connector	Unswapped 1x Mode
Lane Sequence	Lane Sequence	Lane Sequence
D	D	A
C	C	B
B	B	C
A	A	D

3.6.2.1 Configuration

On reset, the lane swap setting for the entire device is controlled by two configuration pins, SP_RX_SWAP and SP_TX_SWAP (see"Signals" on page 213).

Register fields SWAP_TX and SWAP_RX fields in the "SRIO MAC x Digital Loopback and Clock Selection Register" on page 369 can also be written at reset time by I^2C initialization or by software override to set lane swap on a per MAC basis. The reset value of these fields indicates the sampled values of the SP_RX_SWAP and SP_TX_SWAP configuration pins. When a different value is written to either the SWAP_TX or SWAP_RX fields, the MAC has to be reset in order to ensure that the links retrain and communication is re-established with the changed lane configuration.

When changing the lane swap setting for a MAC, it is necessary to reset the port through the SOFT_RST_X4 fields in the "SRIO MAC x Digital Loopback and Clock Selection Register" on page 369.

3.6.2.2 Lanes and Channels

The terms lanes and channels identify input and output signals. Lanes are enumerated using alphabetic characters (A, B, C, D). The pin associated with a lane changes depending on the lane swap settings.

Channels are numbered 0 through 3. Channels are never reordered. When lanes are not swapped, the following mapping between channels and lanes is used:

- Channel 0 maps to Lane A
- Channel 1 maps to Lane B
- Channel 2 maps to Lane C
- Channel 3 maps to Lane D

When lanes are swapped, the following mapping between channels and lanes is used:

- Channel 0 maps to Lane D
- Channel 1 maps to Lane C
- Channel 2 maps to Lane B
- Channel 3 maps to Lane A

Even numbered ports in 4x mode are associated with lanes A-D. When an even numbered port is in 1x mode, it is associated with lane A. Odd numbered ports are always associated with lane B.

The *RapidIO Interconnect Specification (Revision 1.3)* uses Lane 0, 1, 2, 3 terminology instead of the IDT terminology of Lane A, B, C, D.

3.6.2.3 Tx and Rx Swapping

The operations in the SMACx_DLOOP_CLK_SEL register are associated with lanes. When lanes are swapped, the channels associated with these operations must change. The user can independently swap only the Tx or Rx lanes.

Operations on channels, as supported by the MAC Channel Configuration registers always operate on the specific channels regardless of the lane swap settings for a MAC (see “SRIO MAC x SerDes Configuration Channel 0” on page 355). If lane swap functionality is enabled in the system, the proper channels must also be configured. The channel number of a transmit lane and a receive lane differs when Tx lanes are swapped and Rx lanes are not, or vice versa.

3.7 Programmable Transmit and Receive Equalization

The Tsi574 has programmable drive strengths and de-emphasis of a transmit lane. The Tsi574 also has the ability to internally boost the received signal. This functionality is described in the following sections.

3.7.1 Transmit Drive Level and Equalization

The Tsi574 has programmable drive strengths and de-emphasis of a transmit lane. This ability adjusts for the electrical characteristics that can degrade the signal quality of a link which connects a device to the Tsi574. Decreasing the drive strength of signals also provides the ability to reduce the power consumption of a port.

The drive strength current of each lane can be controlled through the TX_LVL field in the "SRIO MAC x SerDes Configuration Global" on page 364, and the TX_BOOST field in the "SRIO MAC x SerDes Configuration Channel 0" on page 355 (see Figure 14).

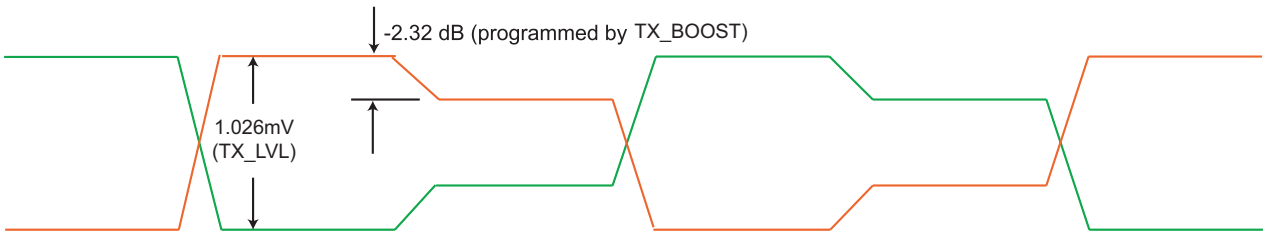
The de-emphasis functionality can be programmed by the TX_BOOST field in the "SRIO MAC x SerDes Configuration Channel 0" on page 355. The TX_BOOST field controls the drive level of subsequent non-transitional bits with respect to the transitional ones. The amount of de-emphasis is specified as a ratio of the de-emphasis drive strength to the TX_LVL drive strength, in steps of ~0.37dB.

The Nominal Drive Level is 1.0 V +/-10%. Refer to the *Tsi574 Hardware Manual* for the more information.

The formula for calculating the TX_BOOST is shown in "SRIO MAC x SerDes Configuration Channel 0" on page 355.

TheTX_LVL[4:0] register affects the Tx signal swing. For normal operation, the TX_LVL should be set at a minimum of 1 V, and for long reach compliance, TX_LVL can be programmed up to 1.26 V.

Figure 14: Drive Strength and Equalization Waveform



3.7.2 Receive Equalization

The received signal can be internally boosted in each receiver by controlling the register RX_EQ_VAL field in the "SRIO MAC x SerDes Configuration Channel 0" on page 355. The equation involving the 3-bit values of the register field are described by:

- Receiver boost = (RX_EQ_VAL + 1)*0.5 dB

For example, setting RX_EQ_VAL[2:0] = 3'b100 results in a 2.5dB boost of the received signal. This boost is internal to the device and is useful in improving the signal at the slicer when the signal arriving at the pins are degraded.

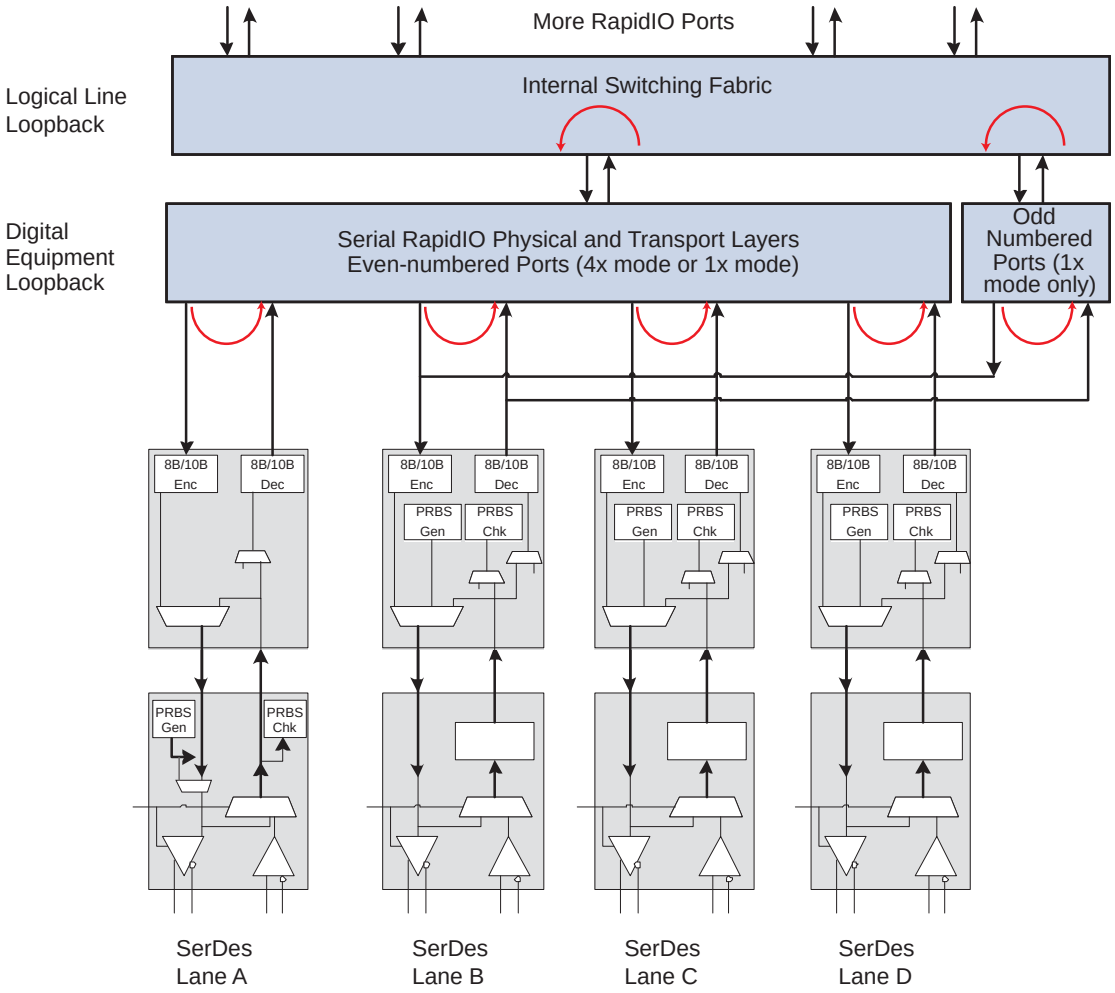
3.8 Port Loopback Testing

The Tsi574's serial RapidIO ports support the following kinds of loopback:

- Digital equipment loopback
- Logical line loopback

Figure 15 shows where each loopback is implemented in the Tsi574.

Figure 15: Tsi574 Loopbacks



3.8.1 Digital Equipment Loopback

Digital equipment loopback is enabled on a per-port basis through the "SRIO MAC x Digital Loopback and Clock Selection Register" on page 369.

When this form of loopback is enabled, the serial port transmit logic is connected to the receive logic just before the 8B/10B encoder and transmitter. Digital equipment loopback requires the use of packets and a correctly configured lookup table. The Bit Error Rate Tester patterns cannot be used when in Digital Equipment Loopback mode. The SerDes does not have to be trained or operational for this loopback to function since the SerDes PHY is not included in the data path.

All incoming data for the port on its external link is ignored when digital equipment loopback is enabled.

3.8.2 Logical Line Loopback

Logical line loopback causes a packet sent into the Tsi574's internal switching fabric to be directed back to the originating port. To cause packets to loop back in this fashion, configure the lookup tables (LUTs) so the destination IDs are destined for the incoming port.

For more information on LUT programming, refer to "Lookup Tables".

3.9 Bit Error Rate Testing (BERT)

The RapidIO ports on the Tsi574 have a built-in bit error rate test (BERT). This test is based either on fixed symbols or on a pseudo-random bit sequence (PRBS). Each lane within a port has a pair of Pattern Generators and Pattern Matchers.

BERT patterns are not framed RapidIO packets and, therefore, when running BERT testing in Tsi574 the word alignment has to be turned off. This can be completed by de-asserting RX_ALIGN_EN bit for the corresponding lane (see "SRIO MAC x SerDes Configuration Channel 0" on page 355).

3.9.1 BERT Pattern Generator

The BERT Pattern Generator can generate different patterns when the link is put into test mode. Table 8 shows what patterns are supported by programming the MODE bit in the "SerDes Lane 0 Pattern Generator Control Register" on page 391.

Table 8: Patterns Supported by Generator

MODE Setting	Description
0	Pattern Generator is disabled
1	15^{th} order linear feedback shift register (LFSR) polynomial: $x^{15} + x^{14} + 1$
2	7^{th} order LFSR polynomial: $x^7 + x^6 + 1$
3	Fixed 10-bit pattern from bottom of PAT0 field

Table 8: Patterns Supported by Generator

MODE Setting	Description
4	2 byte DC balanced pattern constructed as {PAT0, ~PAT0}
5	4 byte DC balanced pattern constructed as: {0x000, PAT0, 0x3FF, ~PAT0}
6:7	Reserved

BERT testing is enabled on a per-bit lane basis, and normal traffic flow on the bit lane ceases when BERT testing is enabled. To enable BERT testing, program the "SerDes Lane 0 Pattern Generator Control Register" on page 391 to enable either normal operation, PRBS-based BERT, or fixed-pattern-based BERT.

BERT testing must be performed across a link from one Tsi574 MAC to another Tsi574 MAC or between the Tsi574 and a device that supports the same polynomial equation.

When testing a link on the Tsi574 with the BERT feature, the link partner device must support PRBS testing with at least one of the two polynomials shown in Table 8, or it must support fixed-pattern tests. Alternatively, the link partner must support some form of loopback to the Tsi574. Consult the appropriate documentation for other devices to determine if they support these features, and to determine how to configure them.

Other PRBS test sequences may be unsuitable for testing in an AC coupled system. The PRBS pattern must ensure that it does not introduce baseline wander and cause an unrealistically high bit error rate. The PRBS patterns generated by the Tsi574 are DC balanced.

3.9.1.1 Disable SerDes Framing

Depending on the type of testing required in the system, the SerDes framing function might need to be disabled in the Tsi574. For example, framing must be disabled if a BERT test is performed.

To disable the framer, write to the RX_ALIGN_EN bit in the SMACx_CFG_CHy register (see "SRIO MAC x SerDes Configuration Channel 0"). Disabling this feature makes sure that data passes through the loopback path without being re-aligned to 10 bit codeword boundaries.

3.9.2 BERT Pattern Matcher and Error Counter

The pattern matcher is capable of synchronizing to and detecting erroneous bytes in the two LFSR patterns mentioned in Table 9. Erroneous bytes are counted in the error counter in the "SerDes Lane 0 Pattern Matcher Control Register" on page 395.

Table 9: Patterns Supported by Matcher

MODE Settings	Description
0	Pattern Matcher and Error Counter are disabled
1	Expect 15th order lfsr polynomial: $x^{15} + x^{14} + 1$
2	Expect 7th order lfsr polynomial: $x^7 + x^6 + 1$
3	Expect d[n]=d[n-10]
4	Expect d[n]=!d[n-10]
5:7	reserved

The Pattern Generator and Matcher are independently controllable within the same lane. They do not need to be enabled, or programmed, the same way. For example, the Tsi574 can transmit a different PRBS pattern than the pattern it is receiving.

When the MODE field in the "SerDes Lane 0 Pattern Matcher Control Register" on page 395 is set to 3'b001 or 3'b010, the pattern matcher operates by generating the expected pattern and synchronizing to the incoming pattern.

The Error Count (COUNT) field in the "SerDes Lane 0 Pattern Matcher Control Register" on page 395 is a 15-bit value. Together with the OV14 bit, a total of $2^{22} - 2^7 - 1$ errors can be reported. When the OV14 bit is set, the count value should be read as count*128.

3.9.3 Fixed Pattern-based BERT

Fixed pattern-based BERT uses data in software-configurable registers to send an alternating pattern of 10-bit 8B10B code groups. Fixed pattern-based BERT does not produce error count results.

Fixed patterns are programmed in the PAT0 field and selected by setting the appropriate MODE field in the "SerDes Lane 0 Pattern Generator Control Register" on page 391.

The following three patterns are useful for BERT testing:

- pat0 = 1010101010 creates a high-frequency pattern, with SMACx_PG_CTL.mode=3'b011
- pat0 = 0011111000, ~pat0 = 1100000111 creates a low-frequency pattern, with SMACx_PG_CTL.mode=3'b100

3.9.3.1 Fixed Pattern-based BERT — Transmitter Configuration

To configure a Tsi574 transmitter for fixed-pattern BERT operation:

- Write the bit stream to be transmitted into the PAT0 field in the "SerDes Lane 0 Pattern Generator Control Register" on page 391.
- Set MODE to the desired fixed pattern mode (MODE=011:100).
- Setting this field causes the software defined pattern to transmit.

3.9.3.2 Fixed Pattern-based BERT — Receiver Configuration

The Pattern Matcher can only match fixed-pattern mode of {PAT0,PAT0} and {PAT0, ~PAT0}. The error counting method is the same as described in "BERT Pattern Matcher and Error Counter".

- Tell the transmitter to stop sending PRBS pattern.
- Re-enable the receiver's framer by writing to the RX_ALIGN_EN bit in the SMACx_CFG_CH{0..3} register.

3.9.4 Using PRBS Scripts for the Transmitters and Receivers

IDT provides PRBS scripts in "PRBS Scripts". All of the PRBS scripts affect all of the ports, therefore editing the files to comment out the respective transmitting and receiving ports where testing is not desired is required.

The following sequence must be followed when using the PRBS scripts:

- Turn on the desired PRBS transmitter with Tsi574_start_prbs_all.txt.
- In the receiving port, turn off the framer using the Tsi574_framer_disable.txt script.
- In the receiving port sync the pattern matcher with the incoming PRBS stream using the Tsi574_sync_prbs_all.txt script
- Read the error count registers. These registers have the following characteristics:
  - Two reads are required in order to obtain the count because the registers are pipelined.
  - The registers must be cleared before use. The registers must be cleared because errors that may have occurred on the port are counted and the registers can contain non-zero values at the start of PRBS testing.

4. Internal Switching Fabric

This chapter describes the main features and functions of the Tsi574's Internal Switching Fabric (ISF). It includes the following information:

4.1 Overview

The Internal Switching Fabric (ISF) is the crossbar switching matrix at the core of the Tsi574. It transfers packets from ingress ports to egress ports and prioritizes traffic based on the RapidIO priority associated with a packet and port congestion.

The ISF has the following features:

- Full-duplex, non-blocking, crossbar-based switch fabric
- 10 Gbits/s fabric ports allow up to 10x internal speedup
- Manages head-of-line blocking on each port
- Cut-through and store-and-forward switching of variable-length packets

4.2 Functional Behavior

The ISF is responsible for transporting packets from an ingress port to an egress port and to and from the multicast engine. When RapidIO packets arrive at the ingress ports, the Tsi574 performs several tests to ensure the packet is valid. If a packet passes these tests, the ingress port consults its destination ID lookup table (LUT) to determine the egress port for the packet. The ISF transfers entire packets without interruption in store-and-forward mode (for more information, see "Transfer Modes" on page 85).

Tip Refer to the "Serial RapidIO Interface" on page 35 for more information how RapidIO packets are tested as valid.

The ISF is a crossbar switch, which means that an ingress port can only send one packet at a time to the ISF, and an egress port can only receive one packet at a time from the ISF. However, the ISF can simultaneously transport packets from multiple independent ingress ports and egress port pairs simultaneously. This architecture has no shared memory area that holds packets.

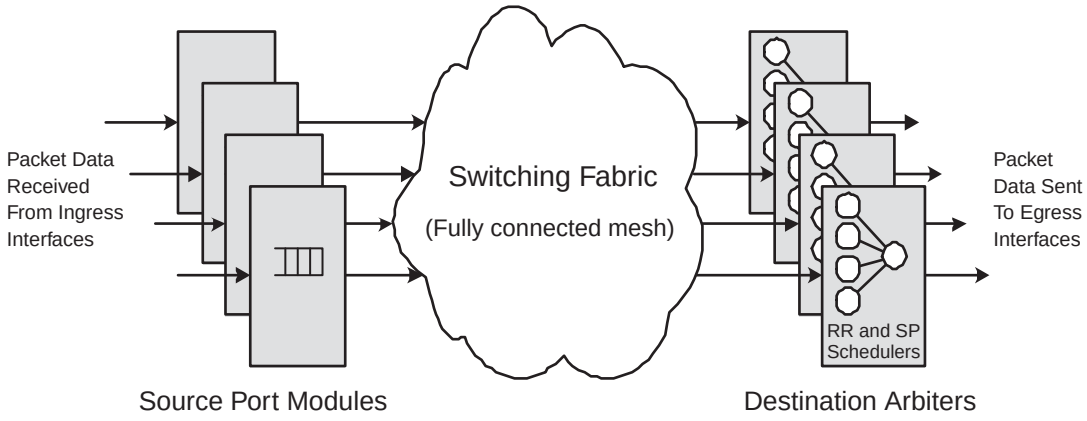
Since many ingress ports can attempt to send a packet to the same egress port, queuing is required at the ingress ports. Special arbitration algorithms at both the ingress and egress sides of the fabric ensure that head-of-line blocking is avoided in these queues.

Queuing is also required at the egress ports. Packets can accumulate when an egress port has to re-transmit a packet (for example, due to a CRC error), or when a high-bandwidth ingress port sends traffic to a lower-bandwidth egress port.

Queuing is also required to support multicast functionality. The ISF supports dedicated connections between each ingress port and the multicast work queue and a dedicated connection between the work queue and the broadcast buffers. This allows packets to be replicated in parallel. For more information on multicast, refer to "Multicast" on page 101.

Figure 16 illustrates a conceptual block diagram, showing the relationship of the components within the ISF.

Figure 16: ISF Block Diagram



4.2.1 Transfer Modes

The ISF supports both cut-through and store-and-forward transfer modes. These modes are selectable on a per-port basis. By default, all ports are configured for store-and-forward mode. To change the configuration, write the TRANS_MODE field in the "RapidIO Port x Control Independent Register" on page 311 when traffic is not flowing through the port.

4.2.1.1 Store-and-Forward Mode

When a port is configured for store-and-forward mode, the port must receive the entire packet before the ISF delivers the packet to an egress port. This increases the latency of all packets received on the port. The increase in latency is directly proportional to the packet size and bit rate of the port.

In store and forward mode, the incoming packet is not sent to the ISF until the whole packet is received.

4.2.1.2 Cut-through Mode

When a port is configured for cut-through mode, the port is permitted to start sending the packet before the packet has fully arrived at the Tsi574. This is possible because the RapidIO destination ID (routing information) appears near the front of a RapidIO packet.

In cut-through mode, the incoming packet is forwarded through the switch as soon as the routing information is received.

Congestion

Configuring a port for cut-through mode does not guarantee that the packet is sent to the ISF immediately after the destination ID arrives for the packet. Congestion in the ISF can mean that some or all of the packet is received before the switching operation begins.

Cut-through mode, generally, provides better system performance. However, in cases where there is a mix of high-speed and low-speed ports, a packet sent from a low-speed port to a high-speed port in cut-through mode prevents the high-speed port from maximizing its output bandwidth. If other ports are also sending to the same destination, the high speed ingress ports could suffer a drop in throughput.

Congestion Example

In this congestion example the following parameters are true:

- Port 0 is currently sending Packet #1 to port 2
- Packet #2, also destined for port 2, starts to arrive on port 1

Packet #2 must wait for the Packet #1 to finish before it has access to port 2. Some or all of Packet #2 must be buffered.

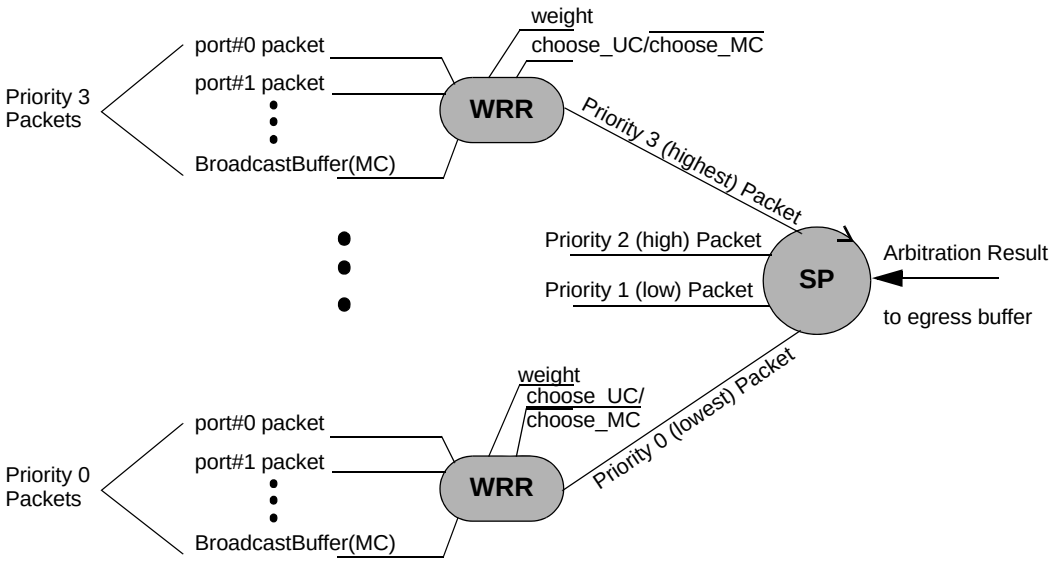
4.3 Arbitration for Egress Port

When multiple ingress ports need to send a packet to the same egress port at the same time, the egress port must make an arbitration decision about which packet to accept.

An output arbiter exists for each egress port. The output arbiters work in conjunction with the input arbiters to avoid head-of-line (HOL) blocking and maximize throughput. When the multicast engine is used, the output arbiter allows system designers to control the maximum number of sequential multicast or non-multicast (unicast) packets that are accepted for a given priority.

The egress port arbiter abides by the RapidIO buffer control rules. It allows the buffer controls to be configured to improve throughput. Two arbitration schemes are employed to handle traffic from the ingress and multicast ports, namely Strict Priority and Weighted Round Robin (see Figure 17). Only HOL packets at the ingress queues or broadcast buffers are considered for arbitration.

Figure 17: Egress Arbitration: Weighted Round Robin and Strict Priority



4.3.1 Strict Priority Arbitration

The ISF always considers packet priority with a strict priority (SP) service algorithm. The output arbiters ensure that all traffic with RapidIO priority *N* is sent before any traffic with RapidIO priority *N-1*.

Refer to the *RapidIO Interconnect Specification (Revision 1.3)* for more information on packet arbitration.

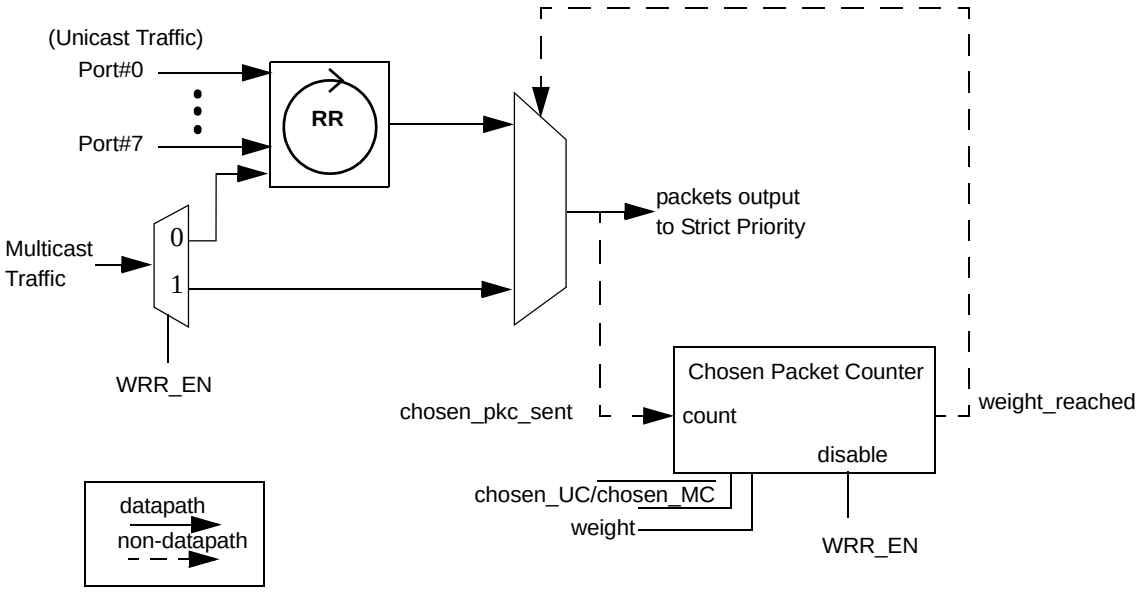
The SP arbiter gives preference to the highest priority packets among the egress ports. As long as priority 3 packets are being presented for arbitration by any port, those packets are accepted ahead of any priority 2 packets. Similar behavior holds for priority 2 packets being chosen over priority 1; and priority 1 over priority 0 packets.

Priority 3 packets from a given port are always transmitted when the port has its turn. However, priority 2 or lower priority packets may not be sent to an egress port when the number of free buffer associated with that particular port is equal or smaller than the watermark for that particular priority. For more information, see “Egress Watermark” on page 90.

4.3.2 Weighted Round Robin (WRR) Arbitration

Within the same priority group, the Weighted Round Robin (WRR) arbiter at each egress port decides which ingress port it receives packets from. This WRR arbiter is a modified Round Robin arbiter with the option to assign different weights on Unicast or Multicast traffic. There are four WRR arbiters per egress port, one for each priority.

Figure 18: Weighted Round Robin Arbiter per Priority Group



The conceptual block diagram of the WRR arbiter is shown in Figure 18. The same arbiter exists for each Priority Group. Depending on the setting of WRR_EN, the Multicast Traffic can participate in the Round Robin arbiter. The WRR arbiter consists of a Round Robin arbiter which services its inputs sequentially, starting at Port#0 on reset. The Chosen Packet Counter is only used for weighted operation between multicast and unicast transactions. Otherwise, the RR arbiter outputs are used.

If the system has no preferred traffic, WRR_EN is de-asserted (see “Port x Prefer Unicast and Multicast Packet Prio 0 Register” on page 386), the multicast packets are treated as unicast packets. Packets from each port, including the multicast ones (if available) are allowed to proceed in order, one after the other, to the strict priority arbitration. The average probability of multicast packets being serviced, with equal traffic load among multicast and unicast ports, is one out of thirteen when operating in 1x mode with eight ports (~5.89%) which is the same as a unicast port.

When weighted operation is required, WRR_EN is asserted (WRR_EN=1). Then the type and quantity of preferred traffic is selected by programming the CHOOSE_UC bit and the minimum number of packets allocated for the chosen traffic on the egress port using the WEIGHT field in the "Port x Prefer Unicast and Multicast Packet Prio 0 Register" on page 386. These two register values set the parameters of operation for the Chosen Packet Counter inside the WRR arbiter. The CHOOSE_UC ($\overline{\text{CHOOSE\_MC}}$) value determines which type of traffic is selected to be favored (0= Multicast, 1 = Unicast). The WEIGHT value determines the number of the packets of the *chosen type* are to be sent in between non-chosen ones.

Every time a chosen packet (either multicast or unicast) is sent, the Chosen Packet Counter is notified. The chosen packets are selected for transmission as long as the WEIGHT value is not reached. Once the WEIGHT value is reached, a non-chosen packet is selected instead and the Chose Packet Counter is reset.

In the case when no chosen packet is available when its opportunity arises, the WRR arbiter automatically selects the non-chosen packets. Similar behaviour applies to non-chosen packet. When the opportunity to transmit non-chosen packets arises, and there is none available, a packet of the chosen type is sent. However, this does not consume the original opportunity allocated.

4.3.2.1 Examples of WRR Arbitration

A few examples of register settings for the WRR arbiter is shown in Table 10.

Table 10: Sample Register settings for WRR in a given priority group (WRR_EN=1)[a]

$\underline{\text{CHOOSE\_UC}}$/ CHOOSE_MC	WEIGHT	% of Multicast Packets Sent to SP Arbiter	% of Unicast Packets Sent to SP Arbiter	Packet Sequence (M = Multicast, U = Unicast)
0	0	0	100	...UUUUUUUU...
0	1	50	50	...UMUMUMUM...
0	15	93.75	6.25	...MUMMMMMMMMMMMMMUM...
1	0	100	0	...MMMMMMMM...

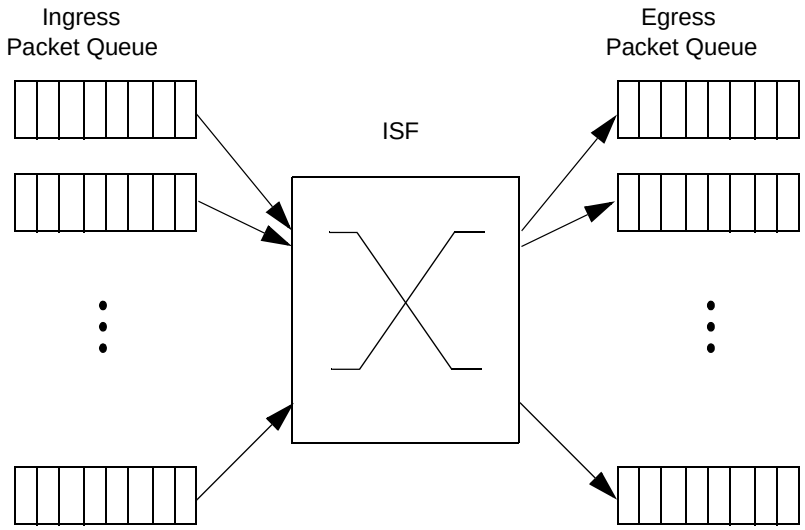
a. The percent values in the table assumes all opportunity for transmission is filled by either the selected or un-selected types.

When there is 100% utilization of either unicast or multicast, lack of transfer of the other type of packet can be encountered in the system.

4.4 Packet Queuing

The Tsi574 has a queuing system on both the ingress and egress ports.

Figure 19: Ingress and Egress Packet Queues in Tsi574



4.4.1 Output Queuing on the Egress Port

Each egress port has a queue that holds up to eight packets. This buffer is required because packets may need re-transmitting. The buffer is also necessary to store the incoming packets when the egress port has a slower baud rate than the ingress port. The depth of the buffer queue dictates the switch fabric flow control. This flow control determines how many packets of a certain priority an egress port can receive. In the event that the output queue is full, the ingress port is notified and must begin queuing packets. If the ingress port runs out of buffers for packets as well, RapidIO link level flow control (packet retries) is activated on the ingress RapidIO port.

For more information on packet retries, refer to *RapidIO Interconnect Specification (Revision 1.3)*.

4.4.1.1 Egress Watermark

The ISF egress arbiter generates flow control for a given priority of traffic based on *watermarks*. Watermarks are defined for priority 0, 1, and 2 packets (no watermark is defined for priority 3 packets because they are always accepted whenever there are free buffers). The watermark values are programmable and are located in “RapidIO Port x ISF Watermarks” on page 385.

Rules for Programming Watermarks

The following rules are applied to Tsi574 watermarks:

- No watermark is associated with Priority 3 packets
- A priority *x* packet is accepted in the buffer if the number of free buffers is greater than the programmed watermark of the associated priority. For example, when the PRIO1WM field is programmed to three, a priority 1 packet is accepted only when there are four or more free buffers.
- The three programmed watermarks (PRIO0WM, PRIO1WM, and PRIO2WM) must contain values where PRIO0WM > PRIO1WM > PRIO2WM > 0 at all times.
- The watermarks for the three priorities must allow for the following minimum levels:
  - PRIO2WM >= 1
  - PRIO1WM >= 2
  - PRIO0WM >= 3

Violating any one of the watermark rules creates a deadlock situation in the system.

The hierarchy of watermarks ensures that packets of lower priority can never consume all buffers and prevent packets of higher priority from being transmitted. With the correct setting of the watermarks, there is at least one priority 3 packet in a full buffer (that is, if all buffers are filled, then at least one of the buffers must be occupied by a priority 3 packet).

Two examples are given in Table 11. The first example describes the default setting of the three watermarks. This maximizes the number of buffers that can accept lower priority packets, which maximizes the throughput of these priorities. The second example describes a customized setting which favors the priority 3 and 2 traffic at the expense of the throughput of priority 1 and 0 packets.

Table 11: Examples of Use of Watermarks

Packet Buffers Available	Example One: PRIO2WM = 1 PRIO1WM = 2 PRIO0WM = 3	Example Two: PRIO2WM = 2 PRIO1WM = 4 PRIO0WM = 5
	Packet Priority that can be Accepted	Packet Priority that can be Accepted
8	0, 1, 2, 3	0, 1, 2, 3
7	0, 1, 2, 3	0, 1, 2, 3
6	0, 1, 2, 3	0, 1, 2, 3
5	0, 1, 2, 3	1, 2, 3
4	0, 1, 2, 3	2, 3
3	1, 2, 3	2, 3
2	2, 3	3
1	3	3
0	none	none

In some systems, it is necessary to guarantee maximum throughput for a burst (continuous sequence) of packets at the same priority. In a congested system, it is possible that only one buffer is available for these packets. This can restrict throughput on the egress port, since while one packet in the burst is being transmitted and is awaiting acknowledgment, another packet in the burst cannot be accepted or transmitted. Watermarks can be used to guarantee that two buffers are available for these packets. When two buffers are available, while one packet is transmitted and awaits acknowledgement another packet can be accepted. This leads to an increase in throughput for packets in the burst.

The packet offered for selection by the output port is subject to the input queuing arbitration. For information on how the ingress port selects which packet to offer for transmission, see “Input Arbitration” on page 93.

4.4.1.2 Transmitting Packets from the Egress Port to the Link Partner

Packets in the output queue are transmitted on the RapidIO link in first-come, first-served (FCFS) order (except during re-transmission). Retransmission represents an opportunity for reordering operations as described in input arbitration (see "Input Queue for the ISF Port" on page 92). Whenever a packet is retried by the link partner, the oldest of those packets with the highest priority in the egress buffer is selected for transmission.

When a port cannot transmit packets, its output queue becomes full. Since the depth of this queue is used to manage flow control across the internal fabric, a mechanism is required to ensure that the fabric does not eventually suffer performance degradation when a port is unable to retire its packets. Inability to retire packets can be caused by a powered down port or the port being in a error state. These states cause the following issues:

- When a port is powered down, it flushes its buffer and continues to accept packets from the ISF. Packets accepted by a powered down port are silently discarded.
- When a port does not enter a normal operating mode with its link partner, this can be detected and the impact to the rest of the system is limited. For more information on detection and recovery from non-operative links, refer to "Loss of Lane Synchronization" on page 62.

4.4.2 Input Queue for the ISF Port

Each ingress port has a queue that holds up to eight packets. Buffering is required to deal with any congestion in the ISF. Since the ISF is a crossbar switch, each egress port can receive one packet from the ISF at a time. If multiple ingress ports need to send to the same egress port, all but one of the ingress ports must buffer its packet and try to transfer it at a later time.

4.4.2.1 Ingress Watermarks

Similar to the egress port, the ingress port generates flow control for a given priority of traffic based on a programmable number of free buffers using watermark. Watermarks can be programmed for priority 0, 1 and 2 packets. Priority 3 packets are always accepted whenever there are free buffers. The "RapidIO Port x RapidIO Watermarks" on page 305 programs watermarks for the ingress port. The rules for programming the Ingress Watermarks are the same as in the egress. These rules and examples can be found in "Egress Watermark" on page 90

This hierarchy of watermarks ensures that packets of lower priority cannot consume all buffers and prevent packets of higher priority from passing them. For example, if all buffers are filled, then at least one of the buffers must be occupied by a packet of priority 3. Since priority 3 is the highest priority in the system, the priority 3 packet should be given the first opportunity to make forward progress.

The default watermark values are 1 for priority 2, 2 for priority 1, and 3 for priority 0. This maximizes the number of buffers that can accept lower priority packets, which maximizes the throughput of these packets.

In some systems, it is necessary to guarantee maximum throughput for a burst (continuous sequence) of packets at the same priority. In a congested system, it is possible that only one buffer is available for these packets. This can restrict throughput on the egress port, since while one packet in the burst is being transmitted and is awaiting acknowledgment, another packet in the burst cannot be accepted or transmitted. Watermarks can be used to guarantee that two buffers are available for these packets. When two buffers are available, while one packet is transmitted and awaits acknowledgement another packet can be accepted. This leads to an increase in throughput for packets in the burst.

If a packet cannot be admitted by the ingress buffer, the packet is dropped and a RETRY is sent to the link partner. The RETRY control symbol begin transmission within 12 SYS_CLK cycles of the reception of the first 4 bytes of the packet. This allows the link partner to select another packet for transmission that has a higher probability of being accepted by the link partner.

The Tsi574 provides performance registers that system software can use to determine the extent of input congestion on the switch (refer to "IDT-Specific Performance Registers" on page 323). Table 11 on page 91 shows which priorities of packets can be accepted given the number of free buffers.

4.4.3 Input Arbitration

When packets are placed in a single input queue, head-of-line (HOL) blocking can result. HOL occurs when the packet at the head of a queue is blocked, and the packets must remain in the same order. This means that no packet in the queue can be sent across the ISF, even if all the packets, save the first, have an uncongested path to their respective destinations.

The ISF manages HOL blocking by reordering packets in a manner compliant with the *RapidIO Interconnect Specification (Revision 1.3)*. This technique may allow another packet to proceed if the packet at the head of a queue is blocked, depending on the arbitration mode selected. In other words the packets are reordered in the queue, but this reordering never violates the RapidIO packet ordering rules.

Three modes are supported and can be configured with the IN_ARB_MODE field in the "Fabric Control Register" on page 372:

- First come, first served (default)
- Strict Priority #1
- Strict Priority #2

Each time the internal switching fabric reorders a packet within its queues[1], the Tsi574 increments a 16-bit counter field (CTR) in the "RapidIO Port x Reordering Counter Register" on page 352 on the affected port. This value can be monitored as an indication of the level of switching congestion. The register also contains a threshold. When the counter is incremented and its new value equals the threshold, the Tsi574 raises the maskable INB_RDR interrupt. This interrupt is masked with the "RapidIO Port x Interrupt Status Register" on page 318.

The number of times a packet is reordered is configurable (see "Reorder Limiting" on page 95).

1. Counting the number of times a packet is reordered within a queue is different from counting the number of times packets are actually sent out of order. The switching fabric might reorder the queue several times before finding one packet to send.

4.4.3.1 First Come, First Served Mode

In this mode, packets flow through the ingress queues in order unless reordering is required to manage head-of-line blocking. The packet closest to the head of the queue that can make progress is selected to make progress regardless of its priority.

Reordering of packets only occurs if the packet at the head of the queue is blocked and there is at least one packet that can make progress. Reordering of packets does not occur if there is no other packets in the buffer.

The reordering limit impacts which packet can be chosen (see “Reorder Limiting” on page 95).

This input arbitration mode can produce the best throughput when prioritization of traffic is not important.

4.4.3.2 Strict Priority One

In this mode, higher priority packets are served ahead of lower priority packets if the higher priority packets are not blocked. Write the IN_ARB_MODE field in the “Fabric Control Register” on page 372 to select this mode.

In this mode, reordering operations only occur when the head of the queue is blocked and there is a packet request with a higher precedence (according to the rules in “Reorder Limiting” on page 95) which exists in the queue.

The reordering limit impacts which packet can be chosen (see “Reorder Limiting” on page 95).

The arbiter selects a packet to compete in egress arbitration based on the following rules:

- Select the priority 3 packet that can be accepted by its destination fabric port and is closest to the head of the queue;
- Else if there are no such packets, Select the priority 2 packet that can be accepted by its destination fabric port and is closest to the head of the queue;
- Else if there are no such packets, Select the priority 1 packet that can be accepted by its destination fabric port and is closest to the head of the queue;
- Else if there are no such packets, Select the priority 0 packet that can be accepted by its destination fabric port and is closest to the head of the queue;
- Else if there are no such packets, Select the priority 3 packet closest to the head of the queue; Note that this packet cannot make progress.
- Else if there are no such packets, Select the priority 2 packet closest to the head of the queue; Note that this packet cannot make progress.
- Else if there are no such packets, Select the priority 1 packet closest to the head of the queue; Note that this packet cannot make progress.
- Else if there are no such packets, Select the priority 0 packet closest to the head of the queue.Note that this packet cannot make progress.

4.4.3.3 Strict Priority Two

In this mode, higher priority packets are served ahead of lower priority packets, even when the high priority packets are blocked. This mode has decreased throughput, but does have the lowest latency on high priority packets.

In this mode, reorder operations only occur when the head of the queue is blocked and there is a packet request with a higher precedence (according to the rules in “Reorder Limiting” on page 95) which exists in the queue.

The arbiter selects a packet to compete in egress arbitration based on the following rules:

- Select the priority 3 packet that can be accepted by its destination fabric port and is closest to the head of the queue;
- Else if there are no such packets, Select the priority 3 packet closest to the head of the queue; Note that this packet cannot make progress. This means all priority 3 packets have to be out of queue before looking at other levels, even if priority 3 packet cannot make progress.
- Else if there are no such packets, Select the priority 2 packet that can be accepted by its destination fabric port and is closest to the head of the queue.
- Else if there are no such packets, Select the priority 2 packet closest to the head of the queue; Note that this packet cannot make progress.
- Else if there are no such packets, Select the priority 1 packet that can be accepted by its destination fabric port and is closest to the head of the queue;
- Else if there are no such packets, Select the priority 1 packet closest to the head of the queue; Note that this packet cannot make progress.
- Else if there are no such packets, Select the priority 0 packet that can be accepted by its destination fabric port and is closest to the head of the queue;
- Else if there are no such packets, Select the priority 0 packet closest to the head of the queue. Note that this packet cannot make progress.

4.4.3.4 Reorder Limiting

When packets leave an input queue in other than first-come first-served order, a packet is said to have been *reordered*. Reordering occurs as described in the previous sections on input arbitration algorithms: “First Come, First Served Mode” on page 94, “Strict Priority One” on page 94, and “Strict Priority Two” on page 95.

If a packet is reordered, that packet is sent earlier than it would otherwise be sent, some packets are sent later than would otherwise be sent, and others are sent in the same relative order. For example, if the fifth packet to arrive at an ingress port is sent first, the first, second, third, and fourth packets are delayed while the sixth, seventh, and eighth packets are not affected.

The Tsi574 never violates the RapidIO protocol when it selects the switching order for packets.

Reorder limiting prevents excessive delays of a packet by packets of lower or equal priority. Reorder limiting does not prevent delays of a packet by packets of higher priority. When reorder limiting is enabled, each time a packet X is delayed in the queue because a lower or same priority packet was sent earlier, the fabric decrements the reorder counter. When the packet reordered ahead of packet X has a higher priority than packet X, the reorder counter of packet X is not decremented. Refer to "Fabric Control Register" on page 372 for more information.

When the reorder counter for packet X reaches 0, no packets of lower or same priority are permitted to be reordered ahead of packet X. When the reorder counter of packet X is 0, packet X must be transmitted ahead of all other packets of lower or equal priority that are positioned after packet X in the queue.

Higher priority packets which appear after packet X in the queue can cause the continued delay of packet X.

Higher priority packets can always be reordered ahead of packet X, whatever the value of the reorder counter of packet X.

One of the properties of reorder limiting is that when the reorder counter of a packet X of given priority Y reaches 0, the reorder counters of all packets with a priority equal or greater than Y that appear ahead of packet X in the queue must also be 0.

Reorder limiting is disabled by default and can be enabled by setting the RDR_LIMIT_EN bit to 1 in the "Fabric Control Register" on page 372. Enabling this feature is recommended. Note that reorder limiting applies to all ports and all packets in the Tsi574.

The number of times a packet is permitted to be delayed by a lower or same priority packet is configurable through the RDR_LIMIT register field in the "Fabric Control Register" on page 372.

Note that reorder limiting can change the packet chosen by FCFS and Strict Priority 1 arbitration. For example, assume three packets, X1, X2 and X3, are held in the ingress queue in that order. Packets X1 and X2 have the same priority, and packet X3 has a higher priority. Packet X1 and X3 cannot make progress. Using Strict Priority 1 arbitration without reorder limiting results in packet X2 being reordered to the head of the queue. However, if the reorder limiting is used, and packet X1's reorder limit counter has reached 0, then the Strict Priority 1 arbitration algorithm cannot select packet X2. Packet X3 is chosen in this case.

4.4.3.5 Transaction Error Acknowledge (TEA)

A Transaction Error Acknowledge signal is implemented in the ISF request queue to control the time a packet can be at the head of the request queue. When an ingress packet at the head of the request queue sends a request to the ISF, a timer is started to keep track of the request time. If the timer value reaches a customer programmable threshold due to congestion at the destination port (defined in TEA_OUT bit in the "Fabric Control Register" on page 372), the TEA interrupt is asserted (maskable by TEA_INT_EN in "Fabric Control Register" on page 372 and "Fabric Interrupt Status Register" on page 374). The packet is removed from the request queue to free up space for ingress traffic and the transaction is deemed incomplete. The TEA error can also be reported back to the host by a port-write.

Port-Write due to TEA can be disabled by either writing 1 in the PW_DIS bit in the "RapidIO Port x Mode CSR" on page 302 (which also turns off other port-writes), or by disabling the TEA interrupt generation by writing a 0 to the TEA_EN bit in the "Fabric Control Register" on page 372.

When there is re-ordering in the request queue, the TEA timer is reset and counting starts with the new head-of-queue packet.

4.4.4 Input Queuing Model for the Multicast Work Queue

The multicast work queue accepts packets from all of the ingress ports. The multicast work queue does not accept packets from the broadcast buffers.

The multicast work queue accepts packets based on strict priority, as described in "Arbitration for Multicast Engine Ingress Port" on page 115. Any priority N packet is accepted before packets of priority N-1. Within each priority, the multicast work queue uses the round robin algorithm.

The multicast work queue operates in strict First-In, First-Out (FIFO) order and has no watermarks associated with it.

The multicast work queue always allows packets to cut-through to the broadcast buffer. Refer to "Cut-through Mode" on page 85 for more information.

4.4.4.1 Multicast Work Queue Ingress Flow Control

Unlike the ingress port and egress port queues, the multicast work queue operates as a bounded buffer. Packets can begin at any point within the bounded buffer. For more information on the operation of the multicast work queue, refer to the Multicast chapter (see Section 5 on page 101).

The multicast work queue has space for up to seven maximum sized (276 byte) packets, or 245 minimum sized (8 byte) packets.

The multicast work queue ingress arbitration operates on the following two rules:

- If there is not sufficient space for a maximum sized (276 byte) packet to be received, all ingress ports are signalled that no packets can be accepted by the multicast work queue.
- If there is sufficient space for at least one maximum sized (276 byte) packet to be received, all ingress ports are signalled that packets of any priority can be accepted by the multicast work queue.

4.4.5 Input Queuing Model for the Broadcast Buffer

The broadcast buffer receives data from only one source - the multicast work queue.

The broadcast buffer operates in strict First-In, First-Out (FIFO) order. The broadcast buffer does not use watermarks.

The broadcast buffer always operates in store-and-forward mode. For more information, refer to “Store-and-Forward Mode” on page 85.

4.4.5.1 Broadcast Buffer Ingress Flow Control

Like the multicast work queue, the broadcast buffer operates as a bounded buffer. Packets can begin at any point within the bounded buffer. For more information on the operation of the broadcast buffer, refer to “Multicast” on page 101.

The broadcast buffer has space for 1 maximum sized (276 byte) packet, or up to 8 smaller packets. Up to 8 packets can be accepted, provided that their individual sizes, rounded up to the nearest multiple of 8 bytes, sum to less than 280 bytes.

The broadcast buffer ingress arbitration operates on the following two rules:

- If there is not sufficient space for 8 more bytes of data to be received, the multicast work queue is signalled that no more packet data can be accepted by the broadcast buffer.
- If there is sufficient space for 8 more bytes of data to be received, the multicast work queue is signalled that more packet data can be accepted by the broadcast buffer.

4.4.6 Output Queuing Model for Multicast

Both the multicast work queue and the broadcast buffer operate in First-In First-Out order. No packet reordering is performed.

The multicast work queue always allows packets to cut-through to the broadcast buffers. This reduces the latency of multicast operations.

Broadcast buffers always operate in store-and-forward mode. This ensures that packet transmission to the egress port is never delayed by packet replication.

IDT recommends that multicast packets within a system all have the same priority.

4.4.7 ISF Bandwidth

The ISF delivers full 10 Gbits/s of bandwidth in both transmit and receive directions. This is sufficient to handle a 4x Serial RapidIO port operation at 3.125 Gbits/s. ISF packets can be sent back-to-back, without interruption.

Delays due to arbitration only occur for the first packet to be sent after a period when no packets were available for transmission. After the first packet has been sent, following packets can be sent back to back.

Bandwidth can be wasted during transfers in cut through mode. If the ingress port operates at a slower rate than the egress port, the egress port receives idles whenever the ingress port has not yet received data for transmission. However, during the transfer, the egress port cannot receive information from ports other than the egress port. Therefore, when transferring data between ports of different bandwidths, it is recommended that the slower port not operate in cut-through mode. Refer to the TRANS_MODE bit in “RapidIO Port x Control Independent Register” on page 311 for more information on how to control cut-through mode.

5. Multicast

This chapter describes the multicast features of the Tsi574. It includes the following information:

5.1 Overview

The Tsi574 multicast functionality is compliant to the *RapidIO Version 1.3 Part 11 Multicast Specification.*

5.1.1 Multicast Operation

In a multicast operation, packets are received at the speed of any ingress port (up to 10 Gbits/s) and broadcast at the speed of the egress ports (up to 10 Gbits/s for a 4x mode port operating at 3.125 Gbits/s) to multiple ports capable of accepting packets for transmission.

Packets are routed to the multicast engine based on their destinationID and Transaction Type (TT) field value. If no match is found for the destinationID and TT field, then the ingress lookup tables are used to route the packet. A maximum of eight different DestID/TT field combinations can be routed to the multicast engine. Each destinationID/TT set can be multicast to a different set of egress ports. A set of egress ports that packets are multicast to is called a multicast group and is represented by the multicast mask in the group table. A multicast packet is never sent out on the port that it was received on. Any number of ports can share the same multicast group.

Multicast packets are accepted by egress ports based on priority. In the event that multicast and unicast traffic are competing for resources in the egress port, multicast specific egress arbitration can be used to favour multicast or unicast traffic. This allows a group of endpoints that need to multicast to each other to share the same multicast mask.

5.1.2 Features

The Tsi574 supports multicast packet replication in accordance with *RapidIO Specification Version 1.3, Part 11 Multicast*.

The Tsi574 includes the following features:

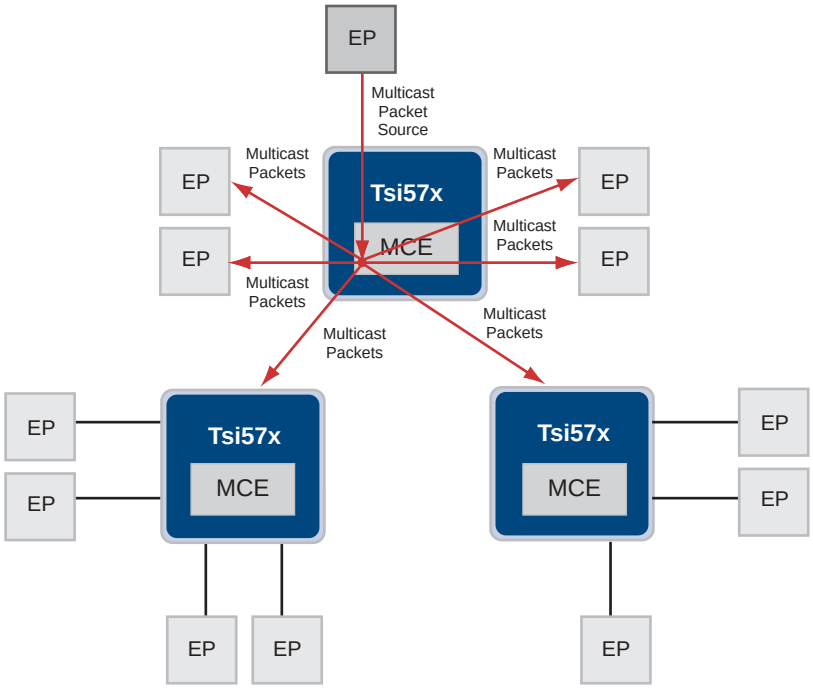
- One multicast engine provides dedicated multicast resources without impacting throughput on the ports
- Eight multicast groups
- Sustained multicast output bandwidth, up to 10 Gbit/s per egress port
- 10 Gbit/s of instantaneous multicast input bandwidth[1]
- Packets are replicated to each egress port in parallel
- The multicast engine can accept bursts of traffic with different packet sizes
- Arbitration at the egress port to allow management of resource contention between multicast or unicast traffic

System behavior for the multicasting of packets which require responses is not defined in the *RapidIO Interconnect Specification (Revision 1.3) - Part 11 Multicast Specification*.

5.1.3 Multicast Operation with Multiple Tsi57x Switches

Tsi57x multicast support is designed to allow information from a single source to be multicast efficiently over a tree topology within a RapidIO fabric (see Figure 20). When two or more Tsi57x switches are connected to each other, however, multicasting packets in both directions between the switches may create a dependency loop that causes a deadlock.

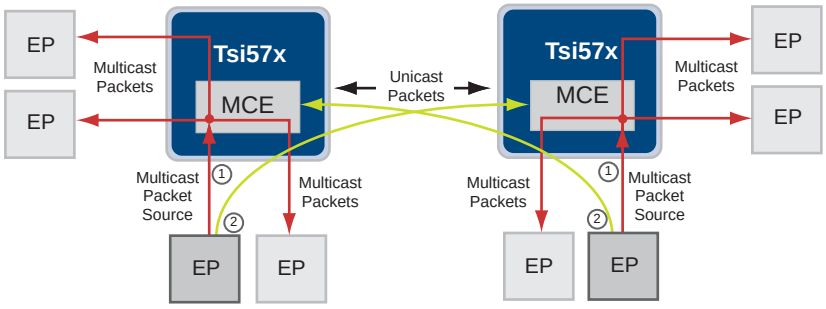
Figure 20: Multicast Operation – Option 1



1. All bandwidths assume the internal switching fabric is clocked at 156.25 MHz.

For systems that use multiple Tsi57x switches, the multicast engine (MCE) in each switch can be used provided that multicast packets are not sent between the switches. This can be accomplished by having the multicast source program the MCE in the switch it is directly connected to (see step 1 in Figure 21), and then initiate an additional unicast transaction to the MCE in the other switch (see step 2).

Figure 21: Multicast Operation – Option 2



5.1.4 Multicast Terminology

The following table contains the terms used to describe the multicast functionality.

Table 12: Multicast Terminology

Term	Definition
Multicast Group	A multicast group is a set of ports that must all receive a copy of a packet. A system can support multiple multicast groups, each of which is completely independent of the other (a group can have all, some, or no ports in common with another group). A multicast group is associated with a unique destinationID and TT of a packet. Note: A packet is never multicast back out of the port that it is received on, regardless of whether or not this port is included in the multicast group.
Multicast Mask	The set of egress ports in a multicast group.
Multicast Vector	The set of ports in a multicast group that will receive the multicast packets. The vector is used to control which broadcast buffers are loaded with the packet to be multicast.
Original Packet	A single multicast packet that arrives at a switch and gets replicated and sent to multiple egress ports according to the multicast mask.
Packet Copy	A copy of an original packet. The copies are sent out on the egress ports.
Multicast Traffic	Packets which are sent to the multicast engine from ingress ports, and packets which are received from the multicast engine by egress ports.
Unicast Traffic	All packets which are not sent to or received from the multicast engine.

5.1.5 Multicast Behavior Overview

The multicast operation involves the following blocks:

- Multicast Engine
- Multicast Group Table
- Multicast Work Queue
- Broadcast Buffer for each egress port
- ISF ingress arbitration algorithms for the Multicast Port
- ISF egress arbitration algorithms for each egress port

The operation of the Work Queue and Broadcast Buffer is described in this section. For more information ingress/egress arbitration algorithms, refer to "Input Queuing Model for the Multicast Work Queue" on page 97, "Input Queuing Model for the Broadcast Buffer" on page 98, and "Output Queuing Model for Multicast" on page 98.

Multicast packets received by an ingress port are routed to the Multicast Engine port based on the destination ID and transaction type (TT) field of the packet. A packet arriving at the Tsi574 is directed to the Multicast Engine (MCE) by the *multicast group table* on the packet's ingress port. The multicast work queue selects a multicast group for the packet, again based on the packet's destination ID and TT field. If the ingress port for the multicast packet is a part of the multicast group, the port is removed from the multicast vector.

Once the multicast vector has been computed, the Work Queue transmits the original packet to the Broadcast Buffers associated with the ports in the vector. Transmission between the multicast work queue and the Broadcast Buffers uses a dedicated ISF path that is separate from those used to route unicast traffic. Once the packet copies have been completely received by all of the broadcast buffers in the multicast vector, each broadcast buffer arbitrates with its associated egress ports to accept the packet copy. The broadcast buffer begins to transmit the packet copy to the egress port's buffer when the egress port signals that it is able to accept a packet.

Packets can cut-through from the ingress port to the multicast work queue and from the multicast work queue to the broadcast buffers. However, a complete packet copy must be received by a broadcast buffer before it attempts to forward the packet copy to the egress port.

Multicast packets are forwarded to the egress ports exactly as they were received. That is, the source and destination IDs are not altered by the Multicast engine. This means that the replicated packets emerge from the egress port with the same destination ID as the original packets.

System designers must be aware of this characteristic of multicast packets because of the necessity that multiple endpoints own the same device ID in order to be able to accept multicast packets.

The routing of unicast packets to a system-wide unique device ID is supported in a multicast application because endpoints are expected to support two or more device IDs in order to differentiate unicast from multicast packets.

5.1.6 Multicast Work Queue

The multicast work queue accepts packets from ingress ports and forwards them to the broadcast buffers according to the multicast group table. The multicast work queue can store a maximum of 2208[1] bytes of packet data or seven maximum sized packets inside its buffer. Once the packet buffer has stored 1936 bytes of data, it forces the ISF to stop further packet transmission.

For information on ISF arbitration for the multicast work queue, refer to “Input Queuing Model for the Multicast Work Queue” on page 97 and “Output Queuing Model for Multicast” on page 98.

Once the first 8 bytes of a packet have been received by the multicast work queue, the destination ID and TT fields of the packet are examined and used by the multicast work queue to determine which multicast group's vector to process in order to replicate the packets. The multicast group table computes a multicast mask which indicates which ports the packet copies should be sent to. The ingress port where the original packet was received is always removed from the multicast mask to form the multicast vector. If the broadcast buffer for a port has detected a maximum latency violation, a system designer can optionally automatically remove this port from the multicast mask. For more information, refer to “Multicast Maximum Latency Timer” on page 117.

When the multicast work queue has computed the multicast vector, it arbitrates to transmit packet copies to the broadcast buffers accordingly. The work queue always operates in a cut-through fashion.

A packet is not dropped if it is STOMPed when it is received in the multicast work queue. The STOMPed packet is replicated to the broadcast buffers. Similarly, if a packet exceeds the time-to-live counter value in the ingress queue, it is replicated to the broadcast buffers by the multicast work queue. For more information on error scenarios, refer to “Error Management of Multicast Packets” on page 116.

5.1.7 Broadcast Buffers

Each egress port has a dedicated broadcast buffer associated with it. The broadcast buffers accept packet copies from the multicast work queue, and forward the packet copies to the egress port. A broadcast buffer can accept one maximum sized packet (276 bytes) or up to eight smaller packets. Eight packets can only be accepted if the sizes of the packets, individually rounded up to the nearest multiple of eight, sum to less than 280 bytes.

For more information on ISF arbitration for the broadcast buffers, refer to “Input Queuing Model for the Broadcast Buffer” on page 98 and “Output Queuing Model for Multicast” on page 98.

The broadcast buffers wait until a packet has been completely received before starting arbitration with the egress port.

1. Packets are stored in an 8-byte boundary. Packets with length of non-multiple of 8-bytes is rounded up to the nearest multiple of 8 for storage in the buffer. The packets are not altered and are transmitted exactly as they are received.

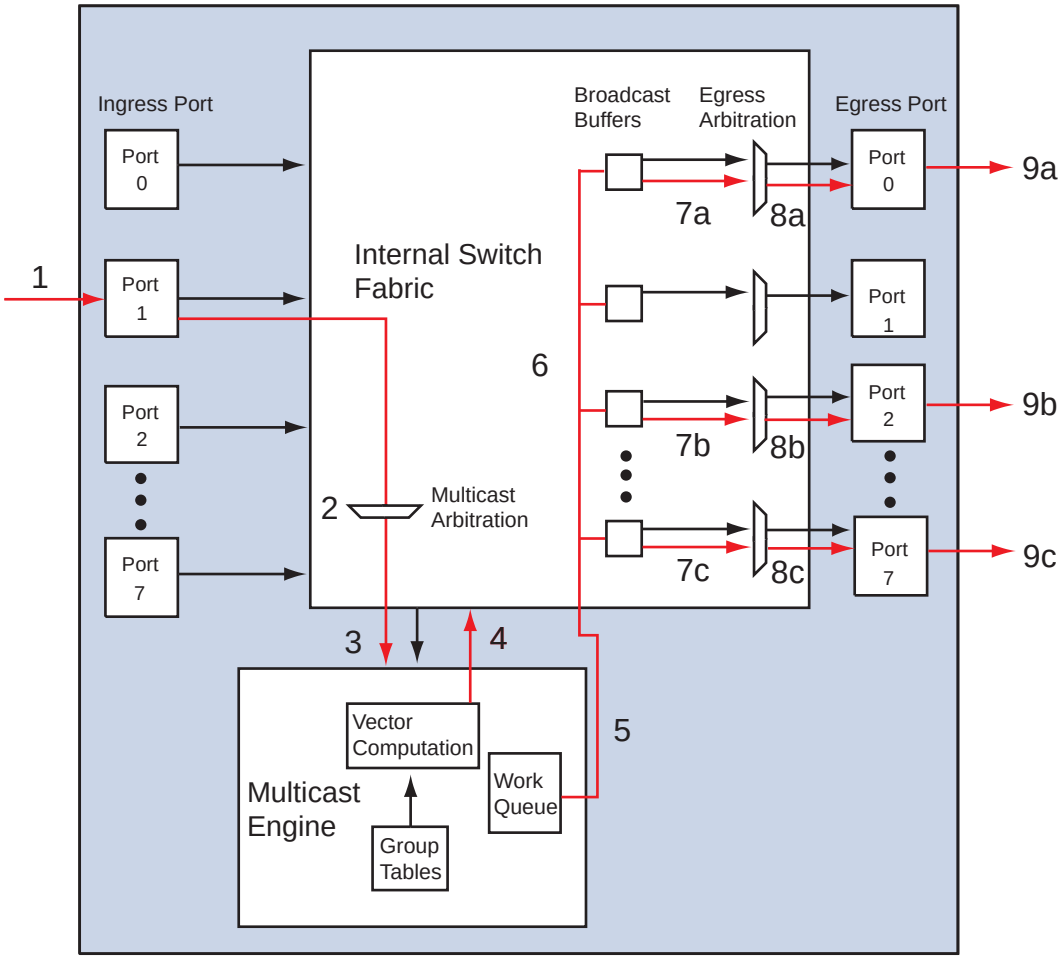
Once the egress port acknowledges the broadcast buffer's request, the broadcast buffer transmits datums to the egress port at sustained rates of up to 10 Gbits/s[1]. The egress port receives the broadcast buffers data, and can start to transmit that data as soon as the first datum is received.

In RapidIO technology, a datum means a word of data sent in a single clock cycle. When the Tsi574 is in 4x mode, a datum is 32 bits and in 1x mode, a datum is 8 bits.

If a packet is STOMPed when it is received at the broadcast buffer, it is not dropped. The STOMPed packet is transmitted to the egress port. The broadcast buffer always sets the time-to-live counter value for a packet copy to the maximum value of seven. For more information, refer to "Error Management of Multicast Packets" on page 116.

The following figure shows a step-by-step multicast operation through the Tsi574 with numbered descriptions of the events below the diagram.

Figure 22: Multicast Packet Flow in the Tsi574



1. When the ISF clock speed is set to 156.25 MHz.

Description of events in the figure:

1. Port 1 receives a packet and consults its multicast group table. It determines that the packet is a multicast packet by examining the packet's destination ID and TT fields. The packet ackID field is overwritten with the ingress port identifier.
   - A packet request is issued by Port 1 to the switch fabric requesting packet transmission to the ISF (MCE port).
   - The multicast packet is placed in the port's ingress buffer. Multicast packets residing in the ingress buffer are subject to TEA as described in "Packet TEA" on page 116. Multicast packets residing in the ingress buffer are also subject to packet reordering.
2. The packet request competes for access to the Multicast Engine (MCE) and is granted transmission permission by the MCE specific arbiter.
3. Having won arbitration to the MCE, Port 1 sends the multicast packet through ISF to the MCE. The transfer occurs at 10 Gb/s[1] and starts after the first 8 or 16 bytes of the multicast packet has been received in the ingress buffer (cut-through mode).
4. The multicast packet is buffered by the MCE in the multicast work queue. The multicast group is determined from the destinationID and TT fields of the packet and the vector computed for the multicast work queue. Multicast packets residing in the multicast work queue are not subject to reordering and are strictly processed in a FIFO manner. The multicast work queue operates in cut-through mode.
5. By consulting the multicast group table, Ports 0, 1, 2, and 7 are identified as members of the vector and as the receiving ports.

   If a destinationID in the multicast group table is disassociated while the packet with that destinationID is still in multicast work queue, the packet is silently dropped.

   - Because Port 1 is the ingress port which originates the multicast packet, as shown in the ackID, it is removed from the Multicast Mask. The resulting Multicast Vector indicates that the packet should be transmitted to Ports 0, 2, and 5.
6. The MCE transmits the multicast vector to the ISF. The ISF is responsible for broadcasting the packets to the broadcast buffers according to the multicast vector. The replication of the multicast packets can occur at a rate of up to 30 Gbps in 4x mode or 37.5 Gbps in 1x mode. Transmission of the packet copies to the selected Broadcast Buffers is a concurrent and parallel operation.
7. The individual broadcast buffers must fully buffer the packet copies before presenting the corresponding packet request to the destination arbiter (store and forward mode). Packet copies residing in the broadcast buffers are subject to multicast latency timeout as described in "Multicast Maximum Latency Timer" on page 117. However, they are not subject to packet reordering and are processed in a strict FIFO manner.
8. When the individual broadcast buffer is granted arbitration, the replicated multicast packet is transmitted to the egress buffer at a rate of up to 10 Gbps. Although represented by the same multicast vector, each individual broadcast buffer operates independently.

1. When the ISF clock speed is set to 156.25 MHz.

9. Once in the egress buffer, the packet copies are subjected to STOMP ("Multicast Packet Stomping" on page 116), and packet reordering. The packet copies at each egress ports are transmitted out from the egress buffer independent of each port.

When packets are being transferred in cut-through mode, it is possible for the packet to have an error detected in it (that is, CRC), or for the packet to be STOMPed by the RapidIO link partner. In these cases, the packet is still accepted and replicated by the multicast work queue and stored in the broadcast buffers. The last datum in the packet is marked with a STOMP bit in the multicast work queue, the broadcast buffer, and the egress port buffer so that the packet is stomped when it is sent out.

System behavior which requires responses from multicast packets is not defined in the *RapidIO Interconnect Specification (Revision 1.3)*.

5.2 Multicast Group Tables

Each ingress RapidIO port on the Tsi574 contains a multi-stage lookup table. The Tsi574 compares the incoming packet's destination ID and TT field to the entries in the lookup table to determine the correct egress port. The destination ID and TT field in the packet uniquely identify a multicast group. The destination ID/TT number space for multicast groups is shared with the number space for unicast destinations.

The first stage of the lookup table contains eight entries used only for multicast packets, and is called the *multicast group table*. This group table resides in both the ingress ports and the multicast engine. An entry is configured either to match a 16-bit destination ID (TT=0) or an 8-bit destination ID (TT=1). For example, to match both the 8-bit destination ID of five and the 16-bit destination ID of five requires two entries in the multicast group table. If the destination ID contained in an incoming packet matches any of the eight entries, the ingress port sends the packet to the multicast engine for replication.

The matching table entry contains a list of ports (multicast mask) to which the multicast engine sends a copy of the packet. However, a packet copy is never sent out from the port that the original packet was received on, regardless of the contents of the port list.

At the ingress port, if none of the multicast destination IDs in the group table match the packet's destination ID, the Tsi574 assumes the packet is a unicast packet and consults the unicast lookup table.

The eight entries in the multicast group table are configured in a multi-step process:

1. Add a set of ports to one of eight multicast masks through repeated writes to the Multicast Mask Configuration Register. The Tsi574 silently ignores attempts to configure masks greater than the mask number. The mask number is defined with the MAX_MASKS field of the "RapidIO Switch Multicast Information CAR" on page 250.

The Tsi574 silently ignores attempts to add or remove non-existent port numbers to/from multicast masks. A non-existent port number is a port number greater than that which exists on the device. For the Tsi574, port numbers greater than 7 are ignored. It is possible to add and remove powered down or otherwise disabled ports to/from the multicast masks.

2. Write the destination ID that identifies the multicast group, and the multicast mask number from the previous step to the "RapidIO Multicast DestID Configuration Register" on page 258.

3. Write the "RapidIO Multicast DestID Association Register" on page 259, setting the LARGE field to indicate whether the destination ID is an 8-bit or a 16-bit ID; and setting the CMD field to 11. This associates the destination ID to the list of ports that must receive copies of the packet. Note that there must be a 1:1 association between destination IDs and multicast masks.

In alignment with the RapidIO multicast specification, if multiple destination ID association operations occur for a multicast mask, the last association operation executed determines which destination ID and TT value is associated with a multicast mask.

Ports can be removed from a multicast mask by writing the Multicast Mask Configuration Register, even when the mask is associated with a destination ID.

If a port that is powered down or detected as faulty is a part of a multicast mask, packets are still replicated and sent to that port. However, the port silently drops the packets.

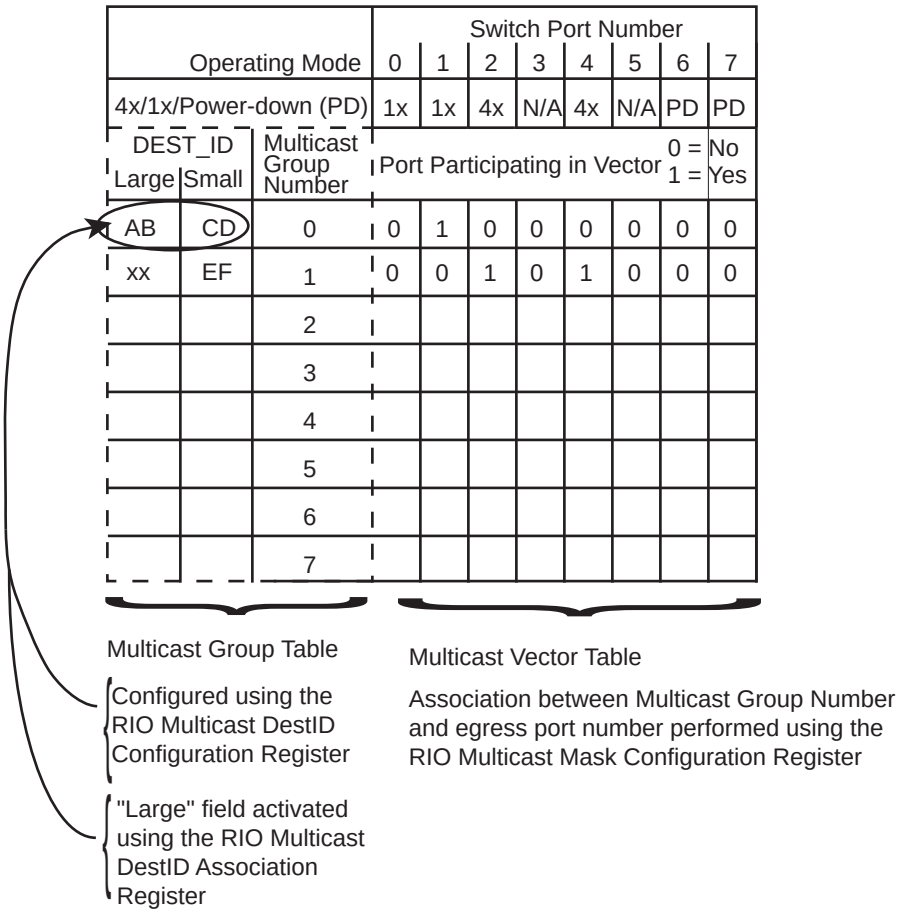
Multicast groups can be deleted by breaking the association between destination IDs and multicast mask numbers through the "RapidIO Multicast DestID Configuration Register" on page 258 and "RapidIO Multicast DestID Association Register" on page 259.

To execute either of the previous two operations (port removal or group deletion), the system software must remember what port is associated to which multicast masks and to which multicast mask number the destination ID is bound. If the software designer selects not to maintain a state table, it is possible to determine what multicast mask a destination ID/TT value is associated with through the use of Write-to-Verify commands.

It is possible that a multicast mask has no ports selected, or the only port selected is the ingress port the original packet was received on. In this case, the multicast engine silently discards the packet.

The following figure is a representation of the relationship between the destination ID, multicast group number, multicast vector, and egress port.

Figure 23: Relationship Representation



5.2.1 Configuring Basic Associations

It is necessary to associate a destination ID with each multicast mask.

For this example, assume the following system requirements:

- The 16-bit destination ID 0x1234 needs to be associated with multicast mask 0.
- The 8-bit destination ID 0x44 needs to be associated with multicast mask 1.
- The 16-bit destination ID 0xFEED needs to be associated with multicast mask 2.

In order to create the multicast mask associations, the following register accesses are required:

The individual association operations can be performed in any order.

1. Set up the operation to associate destination ID 0x1234 with multicast mask 0
   - Write the value 0x1234_0000 to the “RapidIO Multicast DestID Configuration Register” on page 258

2. Associate destination ID 0x1234 with multicast mask 0
   - Write the value 0x0000_00E0 to the "RapidIO Multicast DestID Association Register" on page 259
3. Set up the operation to associate destination ID 0x44 with multicast mask 1
   - Write the value 0x0044_0001 to the RIO Multicast DestID Configuration Register
4. Associate destination ID 0x44 with multicast mask 1
   - Write the value 0x0000_0060 to the RIO Multicast DestID Association Register
5. Set up the operation to associate destination ID 0xFEED with multicast mask 2
   - Write the value 0xFEED_0002 to the RIO Multicast DestID Configuration Register
6. Associate destination ID 0xFEED with multicast mask 2
   - Write the value 0x0000_00E0 to the RIO Multicast DestID Association Register

5.2.2 Configuring Multicast Masks

This section discusses assigning an egress port list to a multicast mask.

5.2.2.1 Clearing Multicast Masks

In this example, the state of the multicast masks is unknown, and therefore the masks must be cleared before being configured. In order to clear the masks the following register accesses are made:

The accesses to the "RapidIO Multicast Mask Configuration Register" on page 256 can be performed in any order.

1. Remove all egress ports from multicast mask 0
   - Write the value 0x0000_0040 to the Multicast Mask Configuration Register
2. Remove all ports from multicast mask 1
   - Write the value 0x0001_0040 to the Multicast Mask Configuration Register
3. Remove all ports from multicast mask 2
   - Write the value 0x0002_0040 to the Multicast Mask Configuration Register

5.2.2.2 Assigning Ports to Multicast Masks

To configure mask 0 to multicast to ports 6 and 7, mask 1 to multicast to ports 3, 4 and 5, and mask 2 to multicast to every port, requires the following series of register accesses:

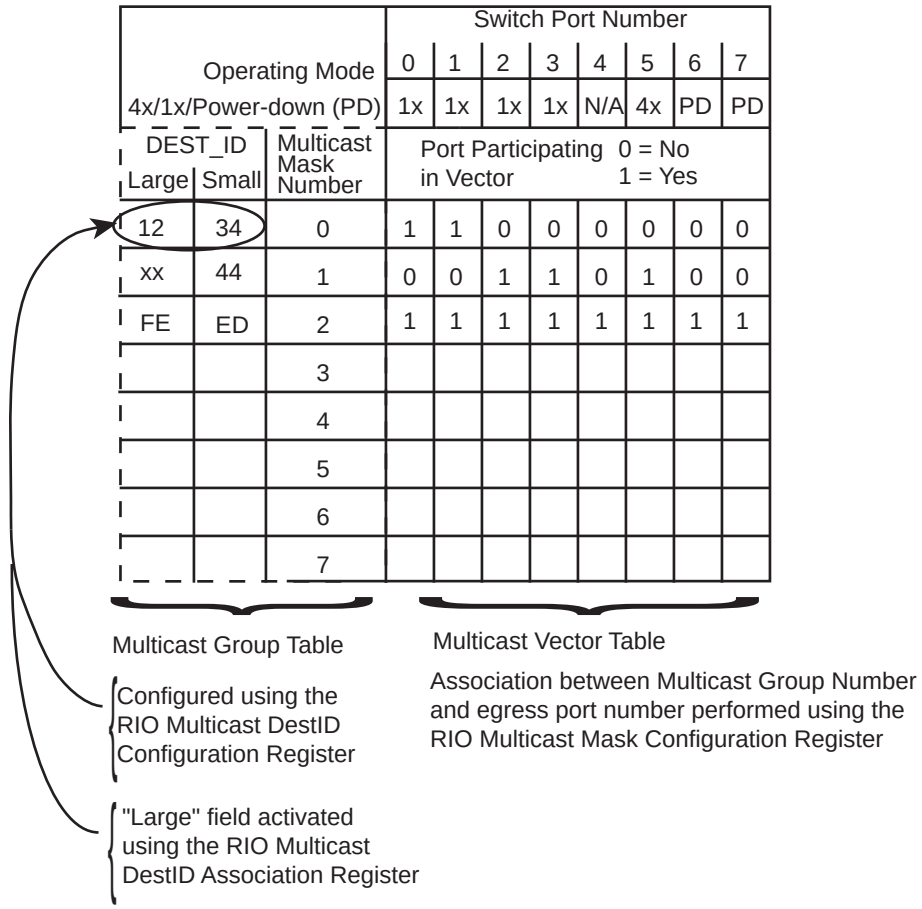
The accesses to the "RapidIO Multicast Mask Configuration Register" on page 256 can be performed in any order.

1. Add port 6 to multicast mask 0
   - Write the value 0x0000_0610 to the Multicast Mask Configuration Register
2. Add port 7 to multicast mask 0
   - Write the value 0x0000_0710 to the Multicast Mask Configuration Register

3. Add port 3 to multicast mask 1
   - Write the value 0x0001_0310 to the Multicast Mask Configuration Register
4. Add port 4 to multicast mask 1
   - Write the value 0x0001_0410 to the Multicast Mask Configuration Register
5. Add port 5 to multicast mask 1
   - Write the value 0x0001_0510 to the Multicast Mask Configuration Register
6. Add all ports to multicast mask 2
   - Write the value 0x0002_0050 to the Multicast Mask Configuration Register

The following figure shows the completed configuration.

Figure 24: Completed Tables at the End of Configuration



5.2.2.3 Removing a Port from a Multicast Mask

In this example, the device attached to port four must be removed from the system. The following register accesses are used to modify multicast masks one and two to stop port four from being a multicast destination:

The accesses to the "RapidIO Multicast Mask Configuration Register" on page 256 can be performed in any order.

1. Remove port 4 from multicast mask 1
   - Write the value 0x0001_0420 to the Multicast Mask Configuration Register
2. Remove port 4 from multicast mask 2
   - Write the value 0x0002_0420 to the Multicast Mask Configuration Register

5.2.2.4 Querying a Multicast Mask

In this example, a system designer needs to determine which of the ports are included in multicast mask 2. The following accesses are to be performed to provide this information:

In each case, the write operation setting up the 'Write to Verify' operation must be performed before the subsequent read to check the Port Present bit status. The individual multicast masks can be queried in any order.

1. Verify that port 0 is included in mask 2
   - Write the value 0x0002_0000 to the Multicast Mask Configuration Register
   - Read the value 0x0002_0001 from the Multicast Mask Configuration Register
2. Verify that port 1 is included in mask 2
   - Write the value 0x0002_0100 to the Multicast Mask Configuration Register
   - Read the value 0x0002_0101 from the Multicast Mask Configuration Register
3. Verify that port 2 is included in mask 2
   - Write the value 0x0002_0200 to the Multicast Mask Configuration Register
   - Read the value 0x0002_0201 from the Multicast Mask Configuration Register
4. Verify that port 3 is included in mask 2
   - Write the value 0x0002_0300 to the Multicast Mask Configuration Register
   - Read the value 0x0002_0301 from the Multicast Mask Configuration Register
5. Verify that port 4 is not included in mask 2
   - Write the value 0x0002_0400 to the Multicast Mask Configuration Register
   - Read the value 0x0002_0400 from the Multicast Mask Configuration Register
6. Verify that port 5 is included in mask 2
   - Write the value 0x0002_0500 to the Multicast Mask Configuration Register
   - Read the value 0x0002_0501 from the Multicast Mask Configuration Register

7. Verify that port 6 is included in mask 2
   - Write the value 0x0002_0600 to the Multicast Mask Configuration Register
   - Read the value 0x0002_0601 from the Multicast Mask Configuration Register
8. Verify that port 7 is included in mask 2
   - Write the value 0x0002_0700 to the Multicast Mask Configuration Register
   - Read the value 0x0002_0701 from the Multicast Mask Configuration Register

5.2.2.5 Removing a Destination ID to Multicast Mask Association

In this example, assume packets to destination ID 0xFF02 on port 4 should no longer be allowed to multicast to all nodes (multicast mask 2). To remove destination ID 0xFF02 from being associated with multicast mask 2, the following register accesses need to be performed in order.

1. Set up the operation to remove the association between destination ID 0xFF02 and multicast mask 2.
   — Write the value 0xFF02_0002 to the "RapidIO Multicast DestID Configuration Register"
2. Remove the association between destination ID 0xFF02 and multicast mask 2.
   — Write the value 0x0000_00C0 to the "RapidIO Multicast DestID Association Register"

When CMD value 0b10 is written (which indicates Remove Associations), then the ASSOC_PRESENT and INGRESS_PORT fields are ignored.

5.2.3 Configuring Multicast Masks Using the IDT Specific Registers

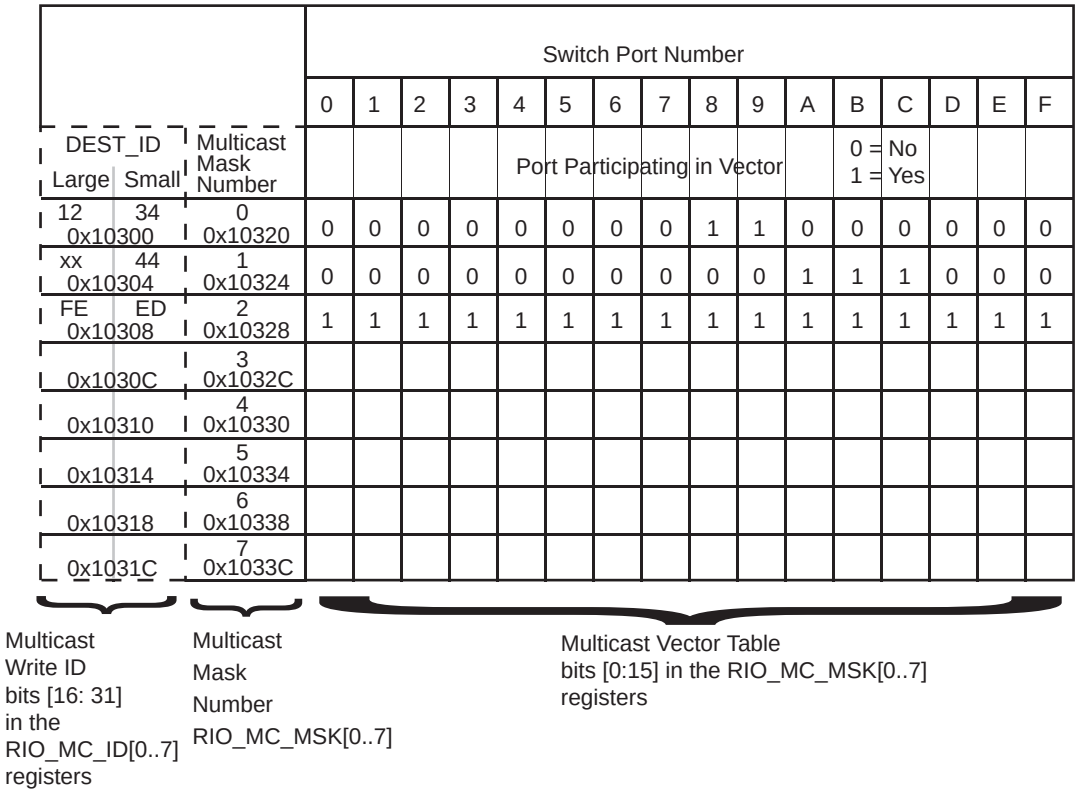
The Tsi574 also has a device specific implementation to configuring the multicast masks. This implementation allows the direct writing of configuration information into the multicast group and vector tables a through the "RapidIO Multicast Write Mask x Register" on page 310 and the "RapidIO Multicast Write ID x Register" on page 309. The use of these two registers permits the direct writing of configuration information into the multicast group and vector tables.

The method described in this section is a Tsi574-specific implementation. The implementation described in "Configuring Multicast Masks" on page 111 conforms to the *RapidIO Interconnect Specification (Revision 1.3).*

Eight registers contain the association between a destinationID and the Multicast Mask number, and eight registers contain the association between the Multicast egress port vector table and the Multicast Mask.

In a closed architecture embedded system, IDT recommends the use of the IDT-specific implementation. However, in an open architecture system the use of the RapidIO compliant register set is recommended. The RapidIO compliant register allows the re-use of switch device independent drivers.

Figure 25: IDT-specific Multicast Mask Configuration



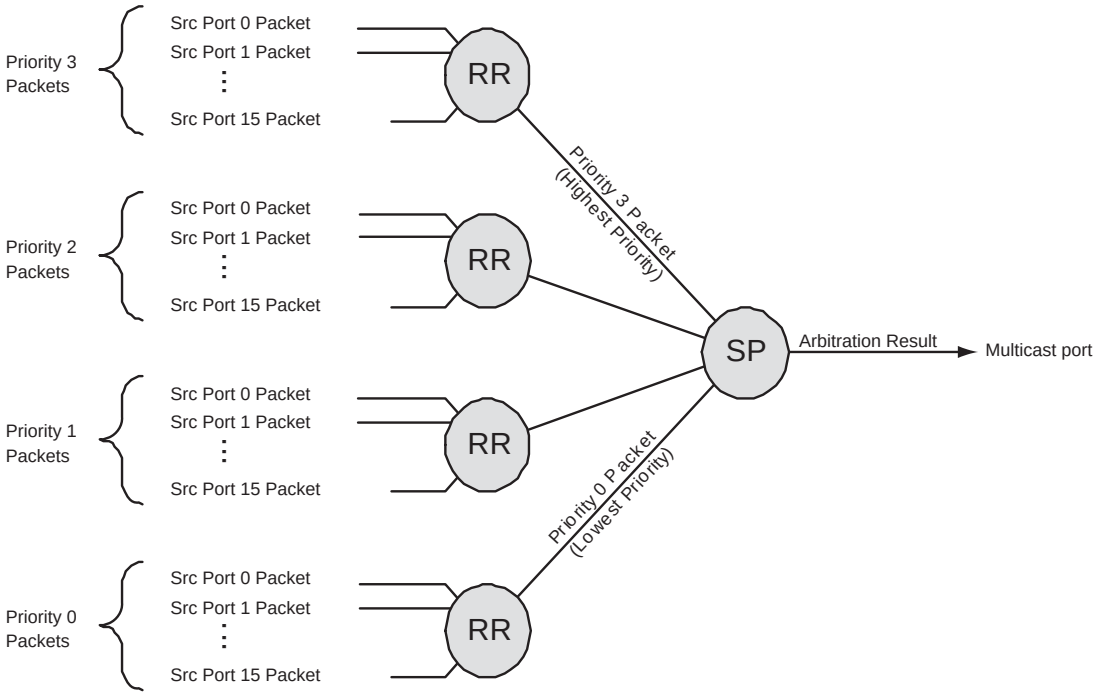
5.3 Arbitration for Multicast Engine Ingress Port

The arbitration scheme used to handle multiple ingress accesses to the Multicast engine ingress port is simple Round Robin (RR) arbitration followed by the Strict Priority (SP) arbitration (see Figure 26). Each ingress port goes through RR arbitration based on their priority group. The outputs of the RR arbitration are handled with SP arbitration where higher priority packets are sent before lower priority ones.

The RR arbiter looks through the ports sequentially starting from Port 0, one after the other, to accept packets when available. No one port can monopolize the RR arbiter. When a port skips an opportunity to transmit because it carries no packet at the moment, the RR arbiter does not compensate for the lost chance and moves to the next port, in sequence, for an available packet.

The packets from the RR arbiters are selected by the SP arbiter based solely on their priority.

Figure 26: Arbitration Algorithm for Multicast Port



5.4 Error Management of Multicast Packets

Multicast packets have four sources of error - packet TEA, packet STOMPing, exceeding the maximum latency time, and exceeding the time-to-live timeout.

5.4.1 Packet TEA

A multicast packet at the head of the ingress queue is subject to TEA. The TEA function does not differentiate between multicast and unicast packets. The assertion of TEA for a packet at the head of the ingress queue causes the packet to be immediately dropped from the queue. The packet is not forwarded to the destination port (see “Multicast Work Queue” on page 105).

5.4.2 Multicast Packet Stomping

Transfers from the ingress port through the multicast work queue into the broadcast buffers operate in cut-through mode. It is possible for the ingress port’s link partner to stomp a packet, or terminate it due to some other error condition. It is also possible for the ingress port to detect a CRC or some other error with the packet. To handle these situations, the Tsi574 supports stomping of a packet while it is being transferred from ingress port to broadcast buffer.

When a packet arrives at the Tsi574 and a stomp control symbol is received part way though the packet, the packet is still multicast to the egress ports. However, the egress ports stomp the packet when they transmit the packet.

5.4.3 Multicast Maximum Latency Timer

It is expected that the time-outs required for multicast packets in many systems are shorter than that required to enforce time-to-live time-outs for unicast packets, so a separate timer known as the maximum latency timer is used. The maximum latency value is similar in concept to the timer, but separate from it. This allows systems designers to enforce latency rules for multicast traffic that are different from that of unicast traffic in the system.

Each broadcast buffer has a separate maximum latency timer to enforce latency limits for a packet and has the capacity to hold a single maximum size packets or up to eight smaller packets (totaling up to, but not more than, 280 bytes). The maximum latency timer starts counting when the packet is completely received by the broadcast buffer. The multicast latency timer has a maximum period of 0xFFFFFF*6.4ns. The multicast latency timer is programmed using the “RapidIO Multicast Maximum Latency Counter CSR” on page 384.

The maximum latency timer is associated only with the packet at the head of the broadcast buffer. The multicast latency timer is held in reset when a packet is being transferred from the broadcast buffer to the egress port.

Once a packet is completely received in the broadcast buffer, the multicast latency timer starts counting. If the multicast latency timer expires while the packet is in the broadcast buffer, an interrupt is raised (see “RapidIO Broadcast Buffer Maximum Latency Expired Error Register” on page 375) and a port-write can be sent (if enabled). All packets in the broadcast buffer at the time the multicast latency timer expires are discarded. A packet copy being transferred from the multicast work queue to the broadcast buffer when the multicast latency timer expires is also discarded. Depending on the system latency restrictions on multicast and the frequency with which the maximum latency timer expires, discarding packets within the broadcast buffer can be sufficient to allow the system to continue to operate. Note that the broadcast buffer is purged only once when the multicast latency timer expires. The next packet copy coming from the multicast work queue is stored in the broadcast buffer as usual. The multicast latency timer is reset and when the new packet reaches the head of the buffer, the timer starts counting again.

In systems where exceeding the maximum latency timer is an indication of the failure of a port, system designers can set the AUTODEAD bit in “RapidIO Multicast Maximum Latency Counter CSR” on page 384 to 1. If the AUTODEAD bit is 1 when the multicast latency timer expires, in addition to packets being purged, the port is removed from multicast operation. By clearing the multicast latency timer error for that failed port, the traffic from the MCE to the broadcast buffer is restored and new packet copies can be received by the port. The AUTODEAD bit should only be set to 1 if the expiry of the maximum multicast latency timer means that an error has been detected, and the continued operation of the system requires removal of the offending port. Refer to “RapidIO Broadcast Buffer Maximum Latency Expired Error Register” on page 375 and to “Global Interrupt Status Register” on page 377 for the register bits related to notification and handling of a multicast latency timeout error.

The “RapidIO Broadcast Buffer Maximum Latency Expired Override” on page 376 can be used to verify the operation of software associated with “RapidIO Broadcast Buffer Maximum Latency Expired Error Register” on page 375.

5.4.4 Silent Discard of Packets

It is possible for the multicast engine to silently discard packets. The following are examples of situations where a multicast packet is dropped:

- A multicast group has no egress ports selected. The multicast mask and multicast vector for a packet using this multicast group is empty.
- If the multicast group has only one port selected which corresponds to the ingress port on which the packet was received, the multicast vector is empty.
- The multicast group contains a port with which the AUTODEAD bit is set in the “RapidIO Multicast Maximum Latency Counter CSR” on page 384

Only the copy of the packet is dropped.

In these cases, no interrupt is issued. No other information is latched regarding the packet(s) that were dropped.

5.4.5 Port-writes and Multicast

Port-writes can be multicast to multiple output links, depending on the destinationID of the port-write. Using the multicast feature improves the likelihood of delivery of port-writes for link failures.

If a blocked or failed port becomes unblocked, port-writes may be delivered late.

5.5 Port Reset

When a port is powered down, the port looses configuration information that is stored for that particular port. For example, multicast settings and port write settings return to their default power up settings after a port reset. After port reset, there is no way to determine that the configuration for a particular port is correct.

When one of the active ports is reset, the multicast mask is required to be re-bound to that port. Refer to “Port Power Down” on page 71 for more information

6. Event Notification

This chapter describes the system of error and event notification in the Tsi574. It includes the following information:

6.1 Overview

The Tsi574 has the following ways to notify external devices about events occurring within the switch:

1. Generate a RapidIO Port-write maintenance message when enabled (as described in the *RapidIO Interconnect Specification (Revision 1.3)*).
2. Assert the INT_b interrupt pin when an enabled interrupt is generated

Most events can generate both types of notification, however some events only generate interrupts.

There is no priority or precedence between events in the error notification scheme of the Tsi574.

6.2 Event Summary

Table 13 describes all the events that can be raised within the Tsi574 and whether these events generate a interrupt, a port-write, or both.

Table 13: Tsi574 Events

Event Name (Status Bit)	Type	Description	Interface Where Event Occurs	Can Generate Interrupt	Can Generate Port-write
Max Retry Occurred	Error	This event occurs when a port's retry counter reaches the configured retry counter threshold. The same retry counter is incremented for retries of all packets (it is not a per-packet retry counter). The status of this event is contained in the IMP_SPEC_ERR bit in the "RapidIO Port x Error Detect CSR" and the MAX_RETRY bit in the "RapidIO Port x Interrupt Status Register". The Retry Counter is reset in the following situations: • Counter reaches threshold • A packet acceptance is received • A packet non-acceptance is received	RapidIO	Yes	Yes
Illegal Transaction	Error	This event occurs when an inbound port receives a transaction with one of the following errors: • Unmapped entry in LUT • Reserved TT field value for data packet or maintenance packet with a hop count not equal to zero The status of this event is contained in the IMP_SPEC_ERR bit in the "RapidIO Port x Error Detect CSR" and the ILL_TRANS_ERR bit in the "RapidIO Port x Interrupt Status Register".	RapidIO	Yes	Yes
LUT Parity Error	Error	This event occurs when a parity error is detected when a port is performing a destination ID lookup (refer to "RapidIO Port x LUT Parity Error Info CSR"). The status of this event is contained in the IMP_SPEC_ERR bit in the "RapidIO Port x Error Detect CSR" on page 286 and the LUT_PAR_ERR bit in the "RapidIO Port x Interrupt Status Register" on page 318.	RapidIO	Yes	Yes
Error Rate Failed Threshold Reached	Error	This event occurs when the error rate counter in the "RapidIO Port x Error Rate CSR" on page 296 reaches the selected Link-Failed threshold found in the "RapidIO Port x Error Rate Threshold CSR" on page 298. For more detail about the Error Rate reporting mechanism, see "Error Rate Thresholds" on page 124. The status of this event is contained in the OUTPUT_FAIL bit in the "RapidIO Port x Error and Status CSR".	RapidIO	Yes	Yes

Table 13: Tsi574 Events *(Continued)*

Event Name (Status Bit)	Type	Description	Interface Where Event Occurs	Can Generate Interrupt	Can Generate Port-write
Error Rate Degraded Threshold Reached	Error	This event occurs when the error rate counter in the "RapidIO Port x Error Rate CSR" on page 296 register reaches the selected Link-Degraded threshold value found in the "RapidIO Port x Error Rate Threshold CSR" on page 298. For more detail about the Error Rate reporting mechanism, see "RapidIO Port x Error Rate Threshold CSR". The status of this event is contained in the OUTPUT_DEG bit in the "RapidIO Port x Error and Status CSR".	RapidIO	Yes	Yes
Reset Request Received	Status	This event occurs when four consecutive reset requests are received by an inbound port. This event is described in detail in "RapidIO Reset Requests" on page 208. The status of this event is contained in the RCS bit in the "RapidIO Port x Multicast-Event Control Symbol and Reset Control Symbol Interrupt CSR" on page 304, as well as the RCS bit in the "Global Interrupt Status Register".	RapidIO	Yes	No
Multicast Symbol Received	Status	This event occurs when an inbound port receives the multicast control symbol. The status of this event is contained in the MCS bit in the "RapidIO Port x Multicast-Event Control Symbol and Reset Control Symbol Interrupt CSR" on page 304, as well as the MCS bit in the "Global Interrupt Status Register".	RapidIO	Yes	No
Outbound Queue Threshold Reached	Status	This event is raised when the "outbound queue threshold exceeded counter" reaches the counter threshold configured for that port. This is used to detect congestion on outbound queues. The status of this event is contained in the OUTB_DEPTH bit of the "RapidIO Port x Interrupt Status Register".	RapidIO	Yes	Yes
Inbound Queue Threshold Reached	Status	This event is raised when the "inbound queue threshold exceeded counter" reaches the counter threshold configured for that port. This is used to detect congestion on inbound queues. The status of this event is contained in the INB_DEPTH bit of the "RapidIO Port x Interrupt Status Register".	RapidIO	Yes	Yes
Inbound Reorder Count Threshold Reached	Status	This event is raised when the reorder count for a particular port reaches the configured reorder counter threshold for that port. The status of this event is contained in the INB_RDR bit of the "RapidIO Port x Interrupt Status Register" on page 318. Note: The "reorder counter threshold" is set to 0xFFFF by default.	RapidIO	Yes	Yes

Table 13: Tsi574 Events *(Continued)*

Event Name (Status Bit)	Type	Description	Interface Where Event Occurs	Can Generate Interrupt	Can Generate Port-write
TEA in Fabric (Output Drop)	Error	This event is raised when a fabric transmission request times out and a packet is dropped. The status of this event is contained in the TEA bit of the "RapidIO Port x Interrupt Status Register" on page 318, the OUTPUT_DROP bit in the "RapidIO Port x Error and Status CSR" on page 270, and the bits of the "Fabric Interrupt Status Register" on page 374, and the TEA bit of the "Global Interrupt Status Register" on page 377.	ISF	Yes	Yes
Multicast TEA	Error	This event is raised when the Multicast Engine fails to deliver a packet to the Broadcast Buffer before a time-out. The status of this event is contained in the MC_TEA bit of the "RapidIO Port x Interrupt Status Register" on page 318.	ISF	Yes	Yes
Fatal Port Error	Error	Inbound or Outbound port has encountered an error from which the hardware was unable to recover (fatal error). The following fatal errors are included: • Four link-request tries with link-response, but no outstanding ackID • Four link-request tries with time-out error for link-response • The "Dead Link Timer" for a port has expired. • The "Lane Sync Timer" for at least one lane of the port has expired. The status of this event is contained in the PORT_ERR bit of the "RapidIO Port x Error and Status CSR". Link-request retries with timeout errors also cause the LINK_TO bit in the"RapidIO Port x Error Detect CSR" to be asserted. Link-responses with no outstanding ackID cause the LR_ACKID_ILL bit in the "RapidIO Port x Error Detect CSR" to be asserted.	RapidIO	Yes	Yes
Port Available Event	Status	This event is raised when a RapidIO port completes its automatic interface initialization after it detects a peer on the interface. This event is typically used to detect hot-swap events. The status of this event is contained in the LINK_INIT_NOTIFICATION bit of the "RapidIO Port x Interrupt Status Register" on page 318.	RapidIO	Yes	Yes
Illegal AckID in Control Symbol	Error	This event is raised when a RapidIO port receives a control symbol (packet-accepted, packet-not-accepted, or retry) with an ackID that is not in use. The status of this event is contained in the CS_ILL_ACKID bit of the "RapidIO Port x Error Detect CSR" on page 286	RapidIO	Yes	Yes[a]

Table 13: Tsi574 Events *(Continued)*

Event Name (Status Bit)	Type	Description	Interface Where Event Occurs	Can Generate Interrupt	Can Generate Port-write
Illegal AckID in Packet	Error	This event is raised when a RapidIO port receives a packet with an ackID that is not in sequence. The status of this event is contained in the PKT_ILL_ACKID bit of the “RapidIO Port x Error Detect CSR” on page 286	RapidIO	Yes	Yes[1]
Illegal Packet Size	Error	This event is raised when a RapidIO port receives a packet that is larger than 276 bytes. The status of this event is contained in the PKT_ILL_SIZE bit of the “RapidIO Port x Error Detect CSR” on page 286	RapidIO	Yes	Yes[1]
Illegal AckID in Link Response	Error	This event is raised when a RapidIO port receives an unused ackID in a Link-Response control symbol. Link-Response control symbols are used to clear retry and error conditions on a link. The status of this event is contained in the LR_ACKID_ILL bit of the “RapidIO Port x Error Detect CSR” on page 286	RapidIO	Yes	Yes[1]
Protocol Error	Error	This event is raised when a RapidIO port receives an unexpected, but otherwise correctly composed, control symbol. An example would be receiving a link-response control symbol when no link-request is outstanding. The status of this event is contained in the PROT_ERR bit of the “RapidIO Port x Error Detect CSR” on page 286	RapidIO	Yes	Yes[1]
Delineation Error	Error	This event is raised when a RapidIO port receives an unaligned /SC/ or /PD/ symbol, or an undefined code group. The status of this event is contained in the DELIN_ERR bit of the “RapidIO Port x Error Detect CSR” on page 286	RapidIO	Yes	Yes[1]
Unexpected Acknowledge	Error	This event is raised when a RapidIO port receives an unexpected packet-accepted control symbol. The status of this event is contained in the CS_ACK_ILL bit of the “RapidIO Port x Error Detect CSR” on page 286	RapidIO	Yes	Yes[1]
Link Timeout	Error	This event is raised when a RapidIO port does not receive an acknowledgement (either a packet-accepted control symbol or a link-response control symbol) in time. The status of this event is contained in the LINK_TO bit of the “RapidIO Port x Error Detect CSR” on page 286	RapidIO	Yes	Yes[1]
Control Symbol CRC Error	Error	This event is raised when a RapidIO port receives a Control Symbol packet with a CRC error. The status of this event is contained in the CS_CRC_ERR bit of the “RapidIO Port x Error Detect CSR” on page 286	RapidIO	Yes	Yes[1]

Table 13: Tsi574 Events *(Continued)*

Event Name (Status Bit)	Type	Description	Interface Where Event Occurs	Can Generate Interrupt	Can Generate Port-write
Control Symbol Not Accepted	Error	This event is raised when a RapidIO port receives a "Packet-Not-Accepted" Control Symbol. The status of this event is contained in the CS_NOT_ACC bit of the "RapidIO Port x Error Detect CSR" on page 286	RapidIO	Yes	Yes[1]
Packet CRC Error	Error	This event is raised when a RapidIO port receives a packet with a CRC error. The status of this event is contained in the PKT_CRC_ERR bit of the "RapidIO Port x Error Detect CSR" on page 286	RapidIO	Yes	Yes[1]
I^2C Event	Status and Error	This event is raised when the I^2C block has an internal interrupt. The status of this event is contained in the I2C bit in the "Global Interrupt Status Register" on page 377, which is a logical OR of all bits in "I^2C Interrupt Status Register" when the corresponding bits in "I^2C Interrupt Enable Register" are enabled.	I^2C	Yes	No
Multicast Latency Exceeded	Error	This event occurs when a multicast request for a particular port cannot be transmitted in the time specified by the RIO Multicast Latency Timer. This results in the MC_LAT being set in the Global Interrupt Status Register. The port does not receive multicast requests unless or until the appropriate error bit in the "RapidIO Multicast Maximum Latency Counter CSR" on page 384 is cleared.	Multicast	Yes	Yes
For error events related to Maintenance Packets with hop count =0, see Table 3					

a.Part of Error Rate Failed/ Degraded Threshold counter

6.3 Error Rate Thresholds

There are two event thresholds in the Tsi574: the Error Rate Failed Threshold Reached and the Error Rate Degraded Threshold Reached. These events notify the system that errors are occurring on a RapidIO interface at a rate that requires special attention.

The error rate detection threshold function works as follows:

- Each RapidIO port maintains a single error counter that is incremented each time one of the RapidIO errors is encountered (see Table 13)
  - The port's hardware can be configured to automatically decrement this counter. The rate at which the counter is decremented is configurable through the "RapidIO Port x Error Rate CSR" on page 296.
- There are two thresholds configured per port on the Tsi574: Error Rate Failed Threshold Reached and the Error Rate Degraded Threshold Reached. These thresholds specify the counter level at which the port is considered either degraded or failed.

— When the degraded threshold is hit, the Error Rate Degraded Threshold Reached event is raised. The port can be configured to raise an interrupt or issue a Port-write (or both). Another degraded event is not raised until the counter falls below the threshold and then reaches it again, due to subsequent errors.

— When the failed threshold is hit, the Error Rate Failed Threshold Reached event is raised. The port can be configured to raise an interrupt or issue a port-write (or both). Another failed event is not be raised until the counter falls below the threshold and then reaches it again, due to subsequent errors. It is also possible to configure the port to drop packets when the Error Rate Failed Threshold Reached event occurs.

- "RapidIO Port x Error Rate CSR" on page 296: This register is used in conjunction with the "RapidIO Port x Error Rate Threshold CSR" on page 298. This register contains the following fields:

  — Error Rate Bias (ERR_RB): This field contains the count used to set the tick timer rate of the Error Rate Bias Timer. At each tick, the Congestion Counter and the Error Rate Counter are decremented.

  — Error Rate Count (ERR_RATE_CNT): This counter maintains a running total of the transmission errors that the port has encountered. It is decremented by 1 on every tick of the Error Rate Bias Timer. The counts do not decrement below 0. This field only tracks transmission errors that have been unmasked in the Port x Error Rate Enable CSR. This counter does not monitor queue depths.

  — Error Rate recovery (ERR_RR): This field allows the user to define how far above the Error Rate Threshold Trigger the Error Rate Counter is allowed to count.

  — PEAK: This field maintains the peak value attained by the Error Rate Counter. It can only be decremented or cleared by a register write.

6.3.1 Maintaining Packet Flow

To maintain packet flow through the switch, the switch can be programmed to selectively discard packets. The following conditions, and the related register bit settings, are required to discard packets:

- Inbound Buffer

  — If an inbound buffer attempts to forward a packet to an outbound buffer, and the TEA_EN bit is set, when the TEA timer expires the OUTPUT_DROP bit is set in the "RapidIO Port x Error and Status CSR" on page 270 of the ingress port where that packet was received. The appropriate port's IRQ error bit is also asserted in the "Fabric Interrupt Status Register" on page 374. The packet which was at the head of the queue to be forwarded through the fabric and caused the TEA to assert is then discarded.

- Outbound Buffer

  — If the outbound buffer attempts to transmit a packet to a link partner, and the DROP_EN bit is set, when the Error Rate Failed Threshold is reached the packet is discarded and the OUTPUT_DROP, OUTPUT_FAIL bits are asserted in the port "RapidIO Port x Error and Status CSR" on page 270 where the link failure occurred.

- Multicast
  - If the multicast engine attempts to transfer a packet copy to a broadcast buffer but the buffer is full and unable to accept the packet, the MC_TEA is asserted in the congested port's "RapidIO Port x Interrupt Status Register" on page 318 and the timed-out packet is discarded.
  - If the port specific RIO Multicast Maximum Latency timer expires per the setting in the "RapidIO Multicast Maximum Latency Counter CSR" on page 384, the bit in the "RapidIO Broadcast Buffer Maximum Latency Expired Error Register" on page 375 is asserted indicating which outbound buffer was unable to accept packets and caused the timer to expire. All packets in the broadcast buffer which were unable to make forward progress are discarded.

6.4 Error Stopped State Recovery

This section describes how to clear physical layer error conditions and status. The standard RapidIO error recovery mechanism is sufficient in normal operation to recover from errors, however in the following special cases software intervention is required for software recovery:

- Link partner reset error
- Hot Swap errors
- Power-up sequence errors

When a link enters a error stopped state, there are multiple ways to clear the error conditions. However, the method described in this section uses IDT specific functionality and control symbols to clear the errors by forcing a hardware recovery situation through software.

Using IDT specific functionality and control symbols to clear errors means that sending of maintenance transactions across the link, and dealing with the resulting time-outs, is not required.

6.4.1 Error Stopped States

An Input Error-stopped state is entered when a RapidIO receiver detects a protocol error. When in an Input Error-stopped state, a port processes control symbols but discards packets. An output Error-stopped state is entered when a RapidIO transmitter is notified that one of the RapidIO receivers on the link has detected a protocol error.

The standard error recovery mechanism for resuming communication is for the port in output error-stopped state to transmit a link-request/input-status control symbol to its link partner. The link partner, which should be in input error-stopped state, sends a link-response control symbol. If the link request/response sequence is repeated four times unsuccessfully, then the port requires software intervention to recover.

6.4.1.1 Input Error-stopped State

In an Input Error Stopped State, all received packets are discarded. Control symbols are still processed when the link is in an Input Error Stopped State. An Input Error Stopped State is entered when an input port detects an invalid character or any valid character other then A, K, or R in an idle sequence or an Stype1 control symbol protocol error.

The following examples are Stype 1 errors:

- Packet with an unexpected ackID value
- Packet with an incorrect CRC value
- Packet containing invalid characters or valid non-data characters
- Packet that overruns some defined boundary such as the maximum data payload.

Tip Refer to error section (Part 6, Chapter 5, section 5.11) of the *RapidIO Interconnect Specification (Revision 1.3)* and "RapidIO Error Management Extension Registers" on page 277 for more information on triggers and specific errors for entering input and output error-stopped states.

6.4.1.2 Output Error Stop State

An Output Error Stop State prevents packets from being transmitted. The error recovery protocol is the only activity on the link.

The reception of a control symbol with no detected corruption, but that violates the Stype 0 control symbol field link protocol, causes the receiving port to immediately enter the Output Error-stopped state.

Link protocol violations include the following:

- Unexpected packet accepted or packet retry control symbol
- Packet-not-accepted control symbol
- Packet accepted control symbol with an unexpected packet_ackID value
- Link time-out while waiting for an acknowledgment control symbol

Tip Refer to error section (Part 6, Chapter 5, section 5.11) of the *RapidIO Interconnect Specification (Revision 1.3)* and "RapidIO Error Management Extension Registers" on page 277 for more information on triggers and specific errors for entering input and output error-stopped states.

6.4.2 Link Error Clearing and Recovery

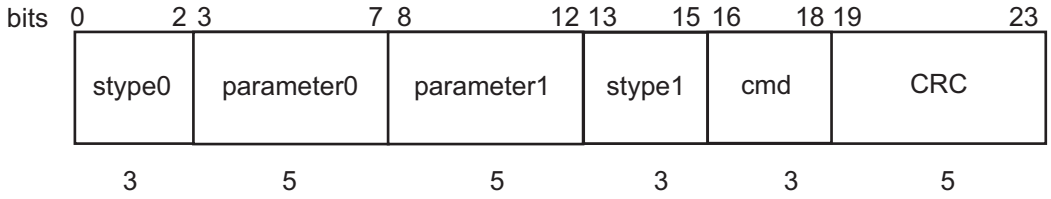
One technique to clear error conditions is to create a situation that forces the hardware recovery process to run again. This situation is created by sending a control symbol to a link partner that the partner can respond to. The control symbol means for software to start the recovery process on the near end of the link, and simultaneously start the recovery process on the far end of the link.

6.4.2.1 Control Symbol Example

When link partners are in an error condition, control symbols can be used to clear the error. There are multiple steps required from both link partners to clear output and input error stopped states. The standard RapidIO mechanism for recovering from errors uses Link Request/Input Status and Link Response control symbols.

Figure 27 shows the RapidIO standard packet-retry control symbol format.

Figure 27: Control Symbol Format



The IDT-specific functionality and control symbols process of using multiple control symbols in error clearing and recovery uses the following control symbols:

- Packet-not-accepted (Stype0) + General Error (Parameter0) + Link Request (Stype1) + Input Status (CMD)

This method triggers hardware recovery at both ends of the link. A Link Request control symbol is sent from the near end, and as part of the same control symbol, a Packet-Not-Accepted control symbol is also sent which notifies the far end of the link of a transmission error and triggers error recovery from that end.

The following steps show IDT specific functionality and control symbols that are used for clearing stop states:

1. The near-end link partner sends a Packet-not-accepted + General Error + Link Request + Input Status control symbol

   To cause the transmission of the required control symbol write the value 0x40FC8000 to the "RapidIO Port x Control Symbol Transmit" register.

2. The far-end link partner (the link partner at the far end of the link) responds with a Link Request/Input Status control symbol

   The far-end link partner also responds with a Link Response/Input Status control symbol in response to the Link Request/Input Status portion of the received Packet-not-accepted + General Error + Link Request + Input Status control symbol.

3. The near-end link partner receives the Link Request/Input Status control symbol and the assertion of the Input Error Stopped State error condition is cleared.
4. The far-end link partner's receipt of a Link Response clears the Output Error Stopped State error condition on the transmit side of the receiving port.
5. Clear any remaining sticky bits in the "RapidIO Port x Error and Status CSR" on page 270 and the "RapidIO Port x Error Detect CSR" on page 286.

6.5 Event Capture

When a notification-enabled RapidIO error occurs, the port where the error occurred also logs information about the packet that caused the event. This information is stored within the RapidIO Packet Error Capture registers (see “RapidIO Error Management Extension Registers” on page 277).

When a packet is logged in these registers, the Valid Capture (VAL_CAPT) bit is set in the “RapidIO Port x Error Capture Attributes CSR and Debug 0” on page 291. While the VAL_CAPT bit is set, further errors do not capture any packet information in order to preserve the first packet information that caused the enabled error. When the capture information has been retrieved, the VAL_CAPT bit must be written to zero in order to clear it and allow subsequent error packets to be captured.

Table 14 lists the errors that cause the error counter to be incremented. All of these errors except the Implementation Specific Logical Error are defined in the *RapidIO Interconnect Specification (Revision 1.2)*. The Implementation Specific Logical Error is set when any of the illegal transaction, maximum retry, lookup table parity error, or time-to-live events occur regardless of whether they are enabled or not.

When most of the errors listed in Table 14 occur, they are logged in the “RapidIO Port x Error Detect CSR” on page 286. If the error is enabled in the RIO Port x Error Rate Enable CSR, the error counter is incremented and information about the packet causing the error is logged in the error capture registers (as long as the VAL_CAPT field is not already set in the “RapidIO Port x Error Capture Attributes CSR and Debug 0” on page 291).

When a port generates a port-write packet, the port-write is routed to the multicast engine if the destinationID of the port-write packet (contained in the RIO Port-Write Target Device ID CSR) is also contained in a multicast group. The multicast engine multicasts the port-write packet to ports according to the mask associated with destinationID of a multicast group. However, in accordance with the *RapidIO Interconnect Specification (Revision 1.2)*, if the originating port is contained in the multicast mask, the multicast engine removes the source port from the mask list which prevents the port-write packet from being transmitted out of the port which originated the port-write packet.

The multicast engine does not distinguish between port-write packets and other types of packets.

For more detail on these events, see the *RapidIO Interconnect Specification (Revision 1.3)* — Error Management.

Table 14: Error Rate Error Events

RapidIO Error	Description	Capture Registers
Implementation Specific	The Tsi574 Switch uses the implementation specific error to combine with other error events, so that they can be included within the Error Rate reporting function. These events are: • Reserved Transport Type detected (tt field = 10 or 11 for all but maintenance packets with hop count =0) • Max Retry Occurred Error • Unmapped DestID Error • Parity Error in Lookup Table • ISF TEA Error • Multicast TEA Error • Port Error When any of these events occur, the Implementation Specific Error event is considered to have occurred. The status of this error is contained in the IMP_SPEC_ERR bit in the "RapidIO Port x Error Detect CSR" on page 286. Caution: The Error Capture register information is only valid for Reserved Transport Type Detected errors and Unmapped DestID errors. For the Max Retry errors the information latched is the last packet received, not the packet that was retried.	Yes
Received corrupt control symbol	Received a control symbol with a bad CRC value. The status of this error is contained in the CS_CRC_ERR bit in the "RapidIO Port x Error Detect CSR" on page 286.	Yes
Received acknowledge control symbol with unexpected ackID	Received an acknowledge control symbol with an unexpected ackID (packet-accepted or packet_retry) The status of this error is contained in the CS_ILL_ACKID bit in the "RapidIO Port x Error Detect CSR" on page 286.	No
Received packet-not-accepted control symbol	Received packet-not-accepted acknowledge control symbol. The status of this error is contained in the CS_NOT_ACC bit in the "RapidIO Port x Error Detect CSR" on page 286.	Yes
Receive packet with unexpected ackID	Received packet with unexpected ackID value - out-of-sequence ackID. The status of this error is contained in the PKT_ILL_ACKID bit in the "RapidIO Port x Error Detect CSR" on page 286.	Yes
Received packet with bad CRC	Received packet with a bad CRC value. The status of this error is contained in the PKT_CRC_ERR bit in the "RapidIO Port x Error Detect CSR" on page 286.	Yes
Received packet exceeds 276 Bytes	Received packet which exceeds the maximum allowed size. The status of this error is contained in the PKT_ILL_SIZE bit in the "RapidIO Port x Error Detect CSR" on page 286.	Yes

Table 14: Error Rate Error Events *(Continued)*

RapidIO Error	Description	Capture Registers
Non-outstanding ackID	Link_response received with an ackID that is not outstanding The status of this error is contained in the LR_ILL_ACKID bit in the "RapidIO Port x Error Detect CSR" on page 286.	No
Protocol error	An unexpected packet or control symbol was received. The status of this error is contained in the PROT_ERR bit in the "RapidIO Port x Error Detect CSR" on page 286.	Yes
Delineation error	Received unaligned /SC/ or /PD/ or undefined code-group. The status of this error is contained in the DELIN_ERR bit in the "RapidIO Port x Error Detect CSR" on page 286.	No
Unsolicited acknowledge control symbol	An unexpected acknowledge control symbol was received. The status of this error is contained in the CS_ACK_ILL bit in the "RapidIO Port x Error Detect CSR" on page 286.	Yes
Link time-out	An acknowledge-response or Link-response is not received within the specified time-out interval, see the RIO_SW_LT_CTL register. The status of this error is contained in the LINK_TO bit in the "RapidIO Port x Error Detect CSR" on page 286.	No

6.6 Port-write Notifications

In the Tsi574, all RapidIO ports can generate port-write messages based on interrupt events. The system is notified of most events that occur in the Tsi574 RapidIO interfaces through the RapidIO port-write message. The port-write function is enabled by default, but can be disabled through the PW_DIS field in the "RapidIO Port x Mode CSR" on page 302. Table 13 on page 120 indicates which events cause RapidIO port-write messages.

When the port-write function is enabled, the occurrence of an enabled port-write capable event causes a port-write message to be sent to the destination ID specified in the "RapidIO Port-Write Target Device ID CSR" on page 285. If the event occurs but the interrupt capability is disabled (through the appropriate interrupt enable register) no port-write message is generated. The port-write message is generated for each event regardless of whether there is already a pending interrupt bit for the event set in the interrupt status register. However, when a new event occurs before the previous port-write has been sent, no port-write is sent for the new event. The second port-write is only sent when the first one is cleared. The outstanding port-writes, if any, are indicated in register "RapidIO Port Write Outstanding Request Register" on page 381.

The port-write packet does not have a guaranteed delivery and does not have an associated response (see *RapidIO Interconnect Specification (Revision 1.3)*). Depending on system design, a port write can be sent repeatedly until cleared. A programmable timeout counter controls the frequency the port-write packets are transmitted (defined in the PW_TIMER field of the "RapidIO Port-Write Timeout Control Register" on page 380). When this timer expires, and the port write has not yet been cleared, another port-write is sent and the timer begins counting again.

6.6.1 Destination ID

There is only one port-write destination ID programmed for the entire device; a port-write event that occurs at any RapidIO port is sent to the same destination ID. The specified destination ID must be mapped within the port's lookup table. Refer to "RapidIO Port-Write Target Device ID CSR" on page 285 for more information.

An image of the "RapidIO Port-Write Target Device ID CSR" and the "RapidIO Logical and Transport Layer Error Enable CSR" are kept in each port. Therefore, if a port is powered down, these three registers must be re-programmed when the port is powered back up.

6.6.2 Payload

The 16 byte data payload of the maintenance port-write packet contains the contents of several CSRs, the port that encountered the error condition, and implementation specific information. The layout of the port-write packet is shown in the Table 15 on page 133.

The payload of the maintenance port-write packet is defined by the *RapidIO Interconnect Specification (Revision 1.3) RapidIO Error Management Extensions*.

Port-writes are sent at the priority defined in the PW_PRIORITY field in the "RapidIO Port x Discovery Timer". The default value is priority 3. Port-write packets are transmitted with a sourceID of 0x00. Port-writes are issued with a hop count of 0xFF.

Table 15 shows the port write packet data payload for error reporting.

Table 15: Port Write Packet Data Payload — Error Reporting

Data Payload Byte Offset	Word 0		Word 1
0x0	"RapidIO Component Tag CSR" on page 252		"RapidIO Port x Error Detect CSR" on page 286 for Port ID
0x8	• Implementation specific bits ("RapidIO Port x Interrupt Status Register" on page 318): — Bit 12: MC_TEA — Bit 13: LINK_INIT_NOTIFICATION — Bit 14: LUT_PAR_ERR bit — Bit 15: OUTPUT_DEG bit — Bit 16: OUTPUT_FAIL bit — Bit 17: INB_RDR bit — Bit 18: INB_DEPTH bit — Bit 19: OUTB_DEPTH — Bit 20: PORT_ERR — Bit 21: ILL_TRANS_ERR — Bit 22: OUTPUT_DROP — Bit 23: MAX_RETRY	Port ID (8 bits) (bits 24 to 31)	"RapidIO Logical and Transport Layer Error Detect CSR" on page 280

6.6.3 Servicing Port-writes

The Tsi574 supports a programming model for servicing port-writes. The first algorithm described minimizes the number of port writes generated by the Tsi574, and therefore the number of specific interrupt events that must be handled by a system host.

When a system host receives a port-write because of an event on Port N, the host follows these steps:

- Determine what error caused the port-write to be generated by going through "RapidIO Logical and Transport Layer Error Detect CSR" on page 280 and the following registers of Port N:
  - "RapidIO Port x Error and Status CSR" on page 270 (bits 6, 7, and 29)
  - "RapidIO Port x Interrupt Status Register" on page 318
- Correct the error conditions and clear the error sources.
- Clear the PORT_W_PEND bit in the "RapidIO Port x Error and Status CSR" on page 270.

In the case when there are other errors from other ports that have generated a port-write, bits in the register "RapidIO Port Write Outstanding Request Register" on page 381 are set.

6.6.4 Port-writes and Hot Insertion/Hot Extraction Notification

Port-write requests are used to support hot insertion/extraction notification. For more information, refer to Hot Insertion and Hot Extraction (see “Hot Insertion and Hot Extraction” on page 59).

The sending device sets the PORT_W_PEND status bit in the “RapidIO Port x Error and Status CSR” on page 270. Software indicates that it has seen the port-write operation by clearing the PORT_W_PEND bit. In order to clear the PORT_W_PEND bit, software must first clear the “RapidIO Port x Error Detect CSR” on page 286.

6.6.5 Port-writes and Multicast

Port-writes can be multicast to multiple output links, depending on the destinationID of the port-write. Using the multicast feature improves the likelihood of delivery of port-writes for link failures.

If a blocked or failed port becomes unblocked port-writes may be delivered late.

If a port generates a port-write packet, and the destinationID of the port-write packet that is contained in the “RapidIO Port-Write Target Device ID CSR” is also contained in a multicast group, the port-write will be routed to the multicast engine. The multicast engine does not distinguish between port-write packets and other types of packets. The multicast engine will multicast the port-write packet to ports according to the mask associated with destinationID of a multicast group. However, in respecting the RapidIO Multicast specification, if the originating port is contained in the multicast mask, the multicast engine will remove the source port from the mask list preventing the port-write packet from being transmitted out of the port that originated the port-write packet.

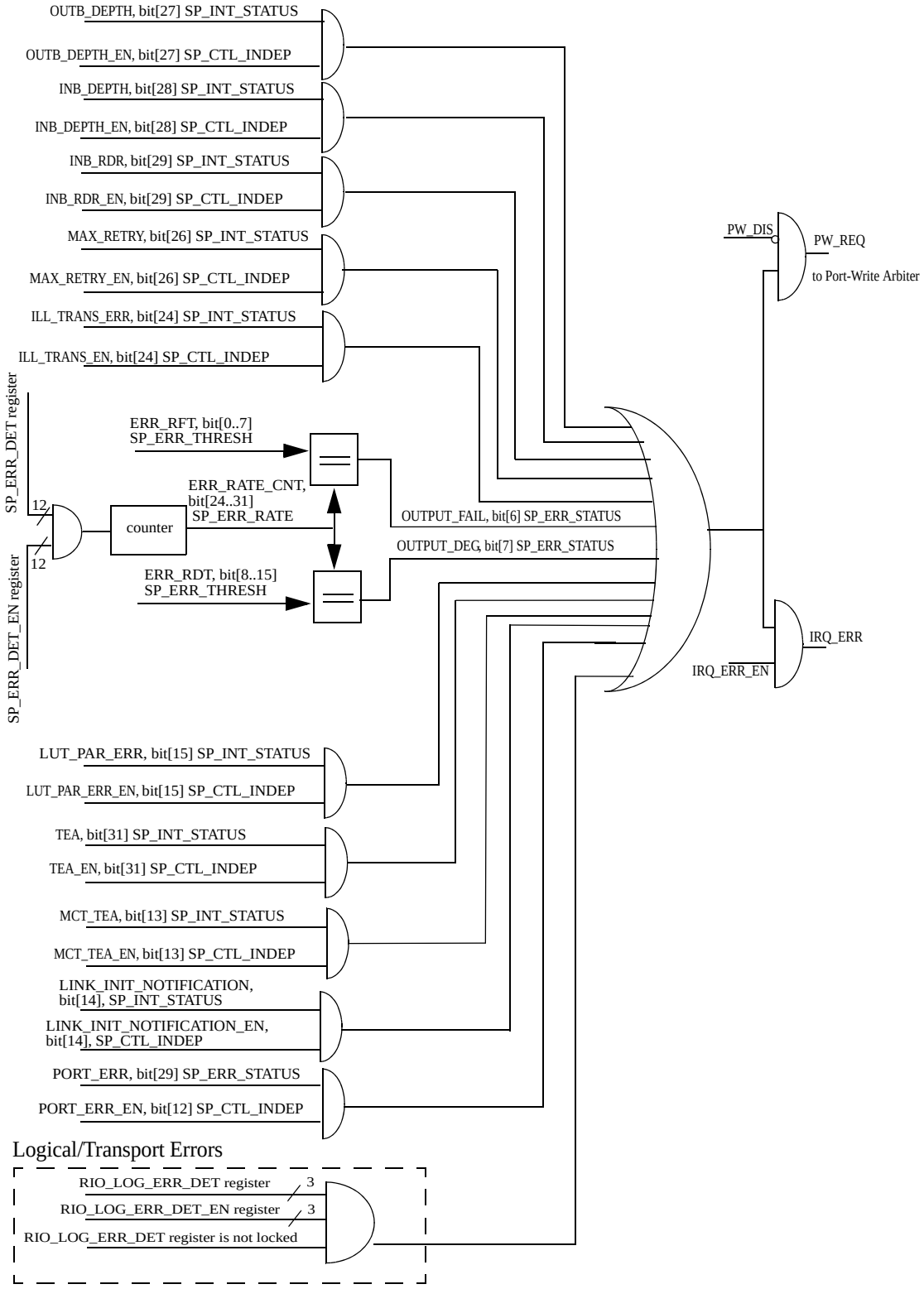
6.7 Interrupt Notifications

In the Tsi574 interrupts are hierarchical, which allows software to determine the cause of the interrupt with minimum register access.

System designers must decide upon a maximum rate of interrupt notifications, and set the error thresholds appropriately.

Figure 28 illustrates the interrupt hierarchy within the Tsi574 RapidIO ports.

Figure 28: RapidIO Block Interrupt and Port Write Hierarchy



6.7.1 INT_b Signal

At the top level of the interrupt hierarchy is the external interrupt signal INT_b. This active low signal is asserted when any fully enabled interrupt occurs. The INT_b signal remains asserted until all interrupts are cleared within the device.

The INT_b signal is driven by the "Global Interrupt Status Register" on page 377. The INT_b signal is asserted when any bit within the Global Interrupt Status register is set and its corresponding enable bit in the"Global Interrupt Enable Register" on page 379 is also set. If an interrupt in the Global Interrupt Enable register is not enabled, the bit in Global Interrupt Status register is still set when an interrupt occurs, but INT_b is not asserted.

Interrupts can be cleared by either writing the interrupt status register bit or by disabling that interrupt. When a previously asserted interrupt is disabled, the interrupt bit remains set in the interrupt status register, but the interrupt is no longer propagated up the interrupt hierarchy.

6.7.2 Global Interrupt Status Register and Interrupt Handling

The "Global Interrupt Status Register" on page 377 must be read to determine why the interrupt was raised. Interrupt causes in the Global Interrupt Status register allow the interrupt service routine to decide which port raised an interrupt. The I^2C controller has a separate indicator bit, as it is not associated with any port.

Two functions that are port specific have separate indicator bits to allow for faster handling. These functions are Multicast Event Control Symbol reception, and reception of a valid reset control symbol sequence. Both Multicast Event Control Symbol and Reset Control Symbol interrupts can be cleared with one register write to the status bit in the broadcast address of the "RapidIO Port x Mode CSR" on page 302.

After the software has read the Global Interrupt Status register and determined which port has an interrupt pending, the port's interrupt status registers must be accessed to determine the exact cause. Each port contains an interrupt status register (see "RapidIO Port x Interrupt Status Register" on page 318) and an associated interrupt enable register (see "RapidIO Port x Control Independent Register" on page 311). When an interrupt occurs within a port, the associated bit for that interrupt is set within the interrupt status register regardless of the setting of the interrupt enable register. The port only notifies the "Global Interrupt Status Register" on page 377 if that interrupt is enabled.

The RapidIO defined status registers are discussed in "Other Interrupts Types and Interrupt Handling" on page 136.

6.7.2.1 Other Interrupts Types and Interrupt Handling

TEA events have a separate register which allows an interrupt handler to quickly determine on which port the TEA occurred. Similarly, multicast latency errors have a separate register to indicate which port is unable to receive multicast packets.

Because there is only one logical layer error per device, the LOG_ERR bit is also in the"Global Interrupt Status Register" on page 377.

For those port specific interrupt causes which are not visible in the "Global Interrupt Status Register" on page 377 register, the interrupt handler must access the port's registers to determine the cause of an interrupt. There are two RapidIO standard registers which must be accessed - the "RapidIO Port x Error and Status CSR" on page 270, and the "RapidIO Port x Error Detect CSR" on page 286.

The Implementation Specific Error (IMP_SPEC_ERR) bit in the "RapidIO Port x Error Detect CSR" on page 286 leads to a number of other IDT specific error and performance related interrupts. These interrupts are found in one other register, the "RapidIO Port x Interrupt Status Register" on page 318. The Tsi574 also provides the implementation specific option of sending an interrupt for some of the bits found in the"RapidIO Port x Error Detect CSR" on page 286.

All interrupt sources and their associated data can be configured by register writes in order to facilitate the testing of software.

Table 16: Port x Error and Status Register Status

Status Bit	Further Information	Interrupt Enable	Interrupt Clearing
OUTPUT_DROP	RIO Serial Port x Control CSR RIO Port x Error Rate CSR RIO Port x Error Rate Threshold CSR RIO Port x Packet Time to Live CSR RIO Port x Interrupt Status CSR	RIO Port x Control Independent CSR for the TEA interrupts. There is no specific interrupt status bit for OUTPUT_DROP of packets when OUTPUT_FAIL is asserted. This must be inferred by the fact that OUTPUT_DROP is set and no TEA interrupts are asserted.	Write 1 to OUTPUT_DROP to clear the RIO Port x Interrupt Status CSR TEA interrupt bits for the port. Writing 1 to this bit will not clear the Port x Error Rate CSR ERR_RATE_CNT, so the next time a packet is sent to this port it may be dropped again.
OUTPUT_FAIL	RIO Port x Error Rate CSR RIO Port x Error Rate Threshold CSR	Port x Error Rate Threshold CSR	Write 1 to OUTPUT_FAIL to clear this interrupt. Writing 1 to this bit will not clear the Port x Error Rate CSR ERR_RATE_CNT, so this bit may become set again immediately.
OUTPUT_DEG	RIO Port x Error Rate CSR RIO Port x Error Rate Threshold CSR	Port x Error Rate Threshold CSR	Write 1 to OUTPUT_DEG to clear this interrupt. Writing 1 to this bit will not clear the Port x Error Rate CSR ERR_RATE_CNT, so this bit may become set again immediately.

6.7.3 Interrupt Notification and Port-writes

In the Tsi574, all RapidIO ports can also generate port-write messages based on interrupt events. Because of this architecture, the RapidIO interrupt enables also control whether a port-write message is issued for each interrupt. In each port the register bit IRQ_EN in “RapidIO Port x Control Independent Register” on page 311 controls whether any enabled interrupts are propagated to the “Global Interrupt Status Register” on page 377 to generate an interrupt. If the IRQ_EN bit is disabled, no interrupt propagates to the “Global Interrupt Status Register” on page 377 register from this port.

The IRQ_EN does not control port-write generation. Port-write generation is controlled by the PW_DIS bit in the “RapidIO Port x Mode CSR” on page 302.

6.7.4 Reset Control Symbol and Interrupt Handling

Reception of a valid reset control symbol sequence is a port specific feature that has separate indicator bits to allow for faster interrupt handling.

7. I²C Interface

Topics discussed include the following:

7.1 Overview

The I²C Interface provides a master and slave serial interface that can be used for the following purposes:

- Initializing device registers from an EEPROM after reset
- Reading and writing external devices on the I²C bus
- Reading and writing Tsi574's internal registers for management purposes by an external I²C master

The I²C Interface has the following features:

- Operates as a master or slave on the I²C bus
  - — Multi-master support
    - – Arbitrates among multiple masters for ownership of the I²C bus
    - – Automatically retries accesses if arbitration is lost
    - – Provides timeout indication if the Tsi574 is unable to arbitrate for the I²C bus

— I2C Interface: Master interface
  – Supports 7-bit device addressing
  – Supports 0, 1, or 2-byte peripheral addressing
  – Supports 0, 1, 2, 3, or 4-byte data transfers
  – Reverts to slave mode if arbitration is lost
  – Supports clock stretching by an external slave to limit bus speed to less than 100 kHz
  – Handles timeouts and reports them through interrupts

— I2C Interface: Slave interface
  – Slave address can be loaded from three sources: power-up signals, boot load from EEPROM, or by software configuration
  – Provides read and write accesses that are 32 bits in size to all Tsi574 registers
  – Ignores General-Call accesses
  – Ignores Start-Byte protocol
  – Provides a status register for determination of Tsi574's health
  – Slave operation enabled/disabled through power-up signal, boot load from EEPROM, or by software configuration
  – Provides mailbox registers for communicating between maintenance software operating on RapidIO based processors and external I2C masters

- Supports I2C operations up to 100 kHz
- Provides boot-time register initialization
  — Supports 1- and 2-byte addressing of the EEPROM selected by power-up signal
  — Verifies the number of registers to be loaded is legal before loading registers
  — Supports up to 2K byte address space and up to 255 address/data pairs for register configuration in 1-byte addressing mode, or up to 65 Kbyte address space and up to 8 K-1 address/data pairs in 2-byte addressing mode.
  — Supports chaining to a different EEPROM and/or EEPROM address during initialization.

The I2C Interface does not support the following features:

- START Byte protocol
  — Tsi574 does not provide a START Byte in transactions it masters
  — Tsi574 does not respond to START Bytes in transactions initiated by other devices. The Tsi574 will respond to the repeated start following the start byte provided the 7-bit address provided matches the Tsi574 device address.

- CBUS compatibility
  - Tsi574 does not provide the DLEN signal
  - Tsi574 does not respond as a CBUS device when addressed with the CBUS address. The Tsi574 will interpret the CBUS address like any other 7-bit address and compare it to its device address without consideration for any other meaning.
- Fast Mode or High-Speed Mode (HS-MODE)
- Reserved 7-bit addresses should not be used as the Tsi574's 7-bit address. If a reserved address is programmed, the Tsi574 will respond to that address as though it were any other 7-bit address with no consideration of any other meaning.
- 10-bit addressing
  - Tsi574 must not have its device address programmed to the 10-bit address selection (11110XXb) in systems that use 10-bit addressing. The Tsi574 will interpret this address like any other 7-bit address and compare it to its device address without consideration for any other meaning.
- General Call. The general call address will be NACK'd and the remainder of the transaction ignored up to a subsequent Restart or Stop.

7.2 Protocol Overview

The I^2C protocol is a two-wire serial interface that consists of a bidirectional, open-drain clock bus (I2C_SCLK), and a bidirectional open-drain data bus (I2C_SD). Multiple master and/or slave devices can be connected to an I^2C bus. I^2C data is transmitted from one device to another across the I2C_SD bus with timing referenced to the I2C_SCLK bus. With some exceptions, each bus can be driven low (to a logic 0) by any device, but is pulled high (to a logic 1) by an external resistor tied to VDD. This creates a "wired-and" configuration, where any single device can drive a bus to a logic 0, but a bus rises to a logic 1 only if no devices are driving to a logic 0, allowing the pull-up resistor to bring the bus to a logic 1 voltage.

I^2C requires one device to assume the role of master during a transfer. The master generates the clock on the I2C_SCLK bus and controls the overall transfer protocol, as defined by the *I^2C Specification*. One or more devices assume the role of slaves during the transfer and respond to the master by either accepting data from the I2C_SD bus, or providing data to the I2C_SD bus. The selection of a specific device to act as a slave results from a master transmitting a unique slave address as part of the I^2C protocol. Only one device is normally configured with the specific slave address and is the only device to respond to the master. Other parts of the I^2C protocol provide for arbitration between multiple master devices, allowing more than one master device to share the bus on a one-at-a-time basis.

This document refers to I^2C signals, serial clock and serial data, by names that are defined by the device package as opposed to the *I^2C Specification*. For example:

- Serial clock – Package name is I2C_SCLK, specification name is SCL.
- Serial data – Package name is I2C_SD, specification name is SDA.

7.3 Block Diagram

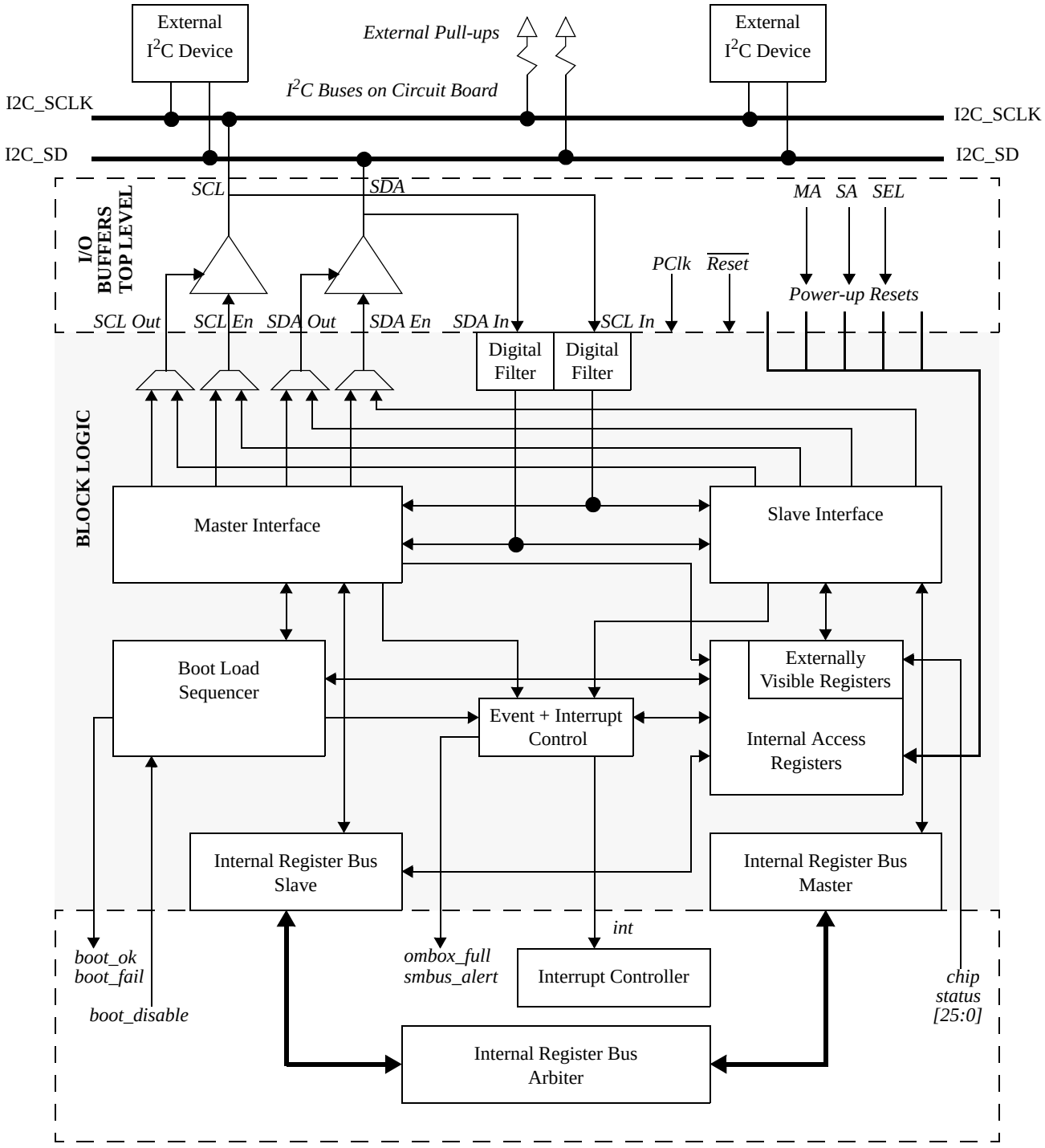
Figure 30 shows an overview of the I2C Interface. The shaded area is the block logic. The master and slave interfaces mux between control of the I2C_SD and I2C_SCLK I/O buffers, and connect through the package to the buses on the board. The reference clock (P_CLK) and active-low hard reset are inputs to the block. The power-up reset values are either static signals from outside the block, or connect to package pins for board-level configuration. The I2C_MA pin is a power-up configuration pin that is latched during reset.

On the core side, the I2C block connects to the internal device register bus as a slave and master:

- As a slave, it enables access to the I2C block registers by a host or processor.
- As a master, it enables access to other device registers (for example, during the I2C load at power-up).

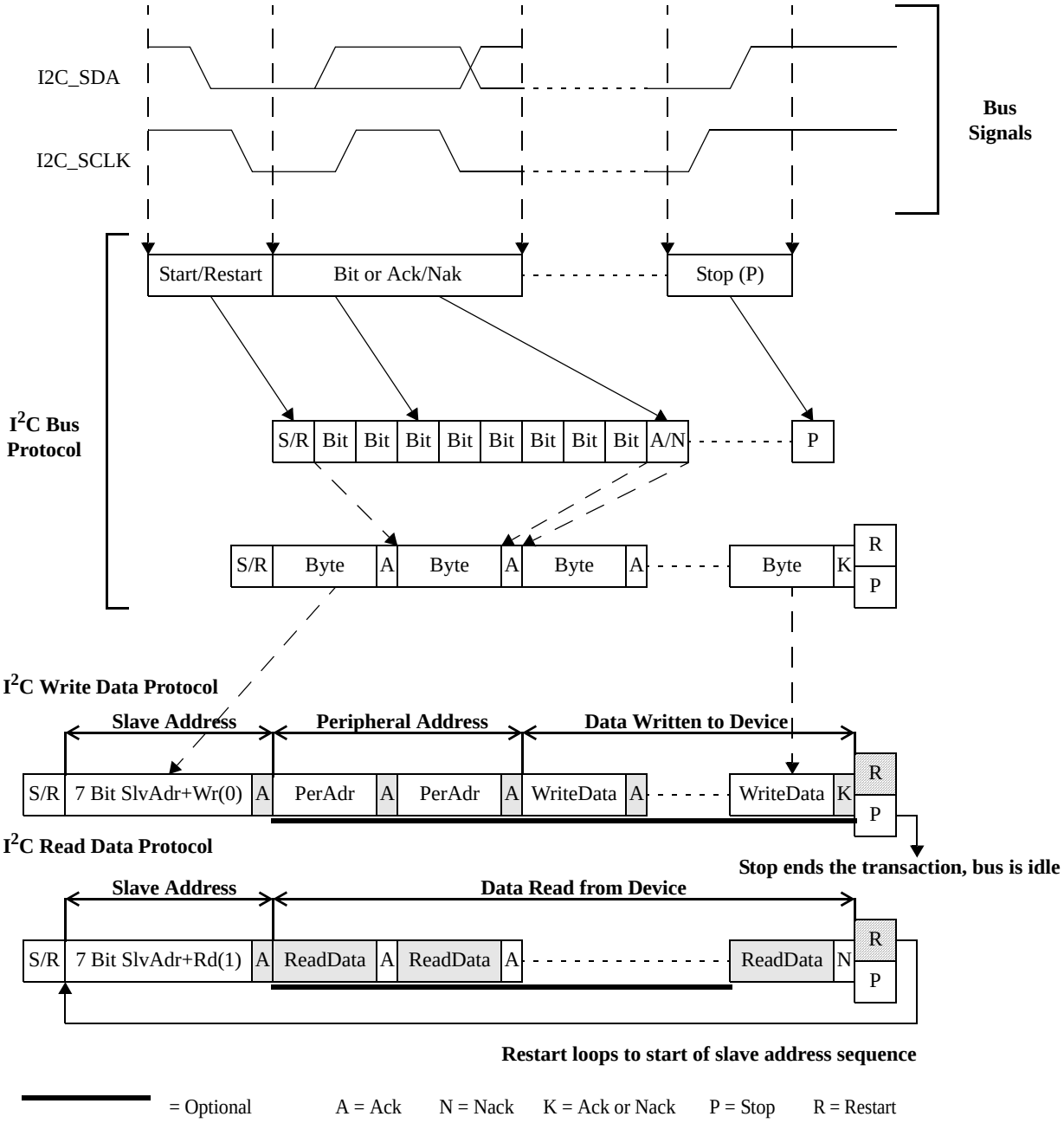
In addition, various signals relate to the boot load sequencer, an interrupt signal connects to the Tsi574 Interrupt Controller, and device-level status connects to the EXI2C_STAT register in the externally visible registers.

Figure 29: I2C Block Diagram



The reference diagram in Figure 30 shows the I2C bus and data protocol. Three basic signal relationships are defined by the bus timing: Start/Restart, Bit, and Stop.

Figure 30: I2C Reference Diagram



Note: The I2C read data protocol section of this figure implies that the peripheral addressing phase has already occurred. The I2C Interface will remember where it left off such that new reads to the same device do not require the peripheral addressing phase; however, an initial read of an I2C device will require a peripheral addressing phase.

The *start/restart* and *stop* conditions delineate a transaction – a master issues a *start* to claim ownership of the bus and a *stop* to release ownership. A *restart* is a repeated start condition between the first *start* and the terminating *stop*, and is used by a master to start a new transaction without giving up bus ownership. Between the start and stop, data is transferred one bit at a time, with the basic protocol calling for full bytes of data, this being 8 consecutive bits from master to slave or slave to master, followed by 1 more bit driven by the device receiving the data to acknowledge the receipt of the byte. The first stream of bytes following a start/restart is the *device connection sequence*, where the master places a slave address on the bus to make a connection to the device with which it needs to communicate. It is also during this period that primary arbitration for bus ownership is completed for the bus between multiple masters. If more than one master attempts to address a slave, a well-defined procedure results in all but one master backing off and waiting for a stop condition, when the bus ownership is again released.

Once a connection is made between a master and a slave, data can be transferred to or from the slave in the *data read/write sequence* phase of the transaction. This sequence is device-dependent, but a common protocol used by memory-oriented devices such as EEPROMs involves the master sending one or more bytes of *memory address* to the slave to position the slave's memory address (or peripheral address), then the master writes/reads data to/from the slave. Eventually the master ends the transaction with a *stop condition*, at which point the bus is free for other masters to start transactions.

These I2C master and slave operations are explained in the following sections.

7.4 Tsi574 as I2C Master

When the Tsi574 is an I2C master, it addresses an external slave device, generates the I2C_SCLK clock, and controls the overall transfer protocol. There are two instances where the Tsi574 is master: boot loading (see "Boot Load Sequence"), and transactions initiated by setting the START bit in the "I2C Master Control Register" (see the following section).

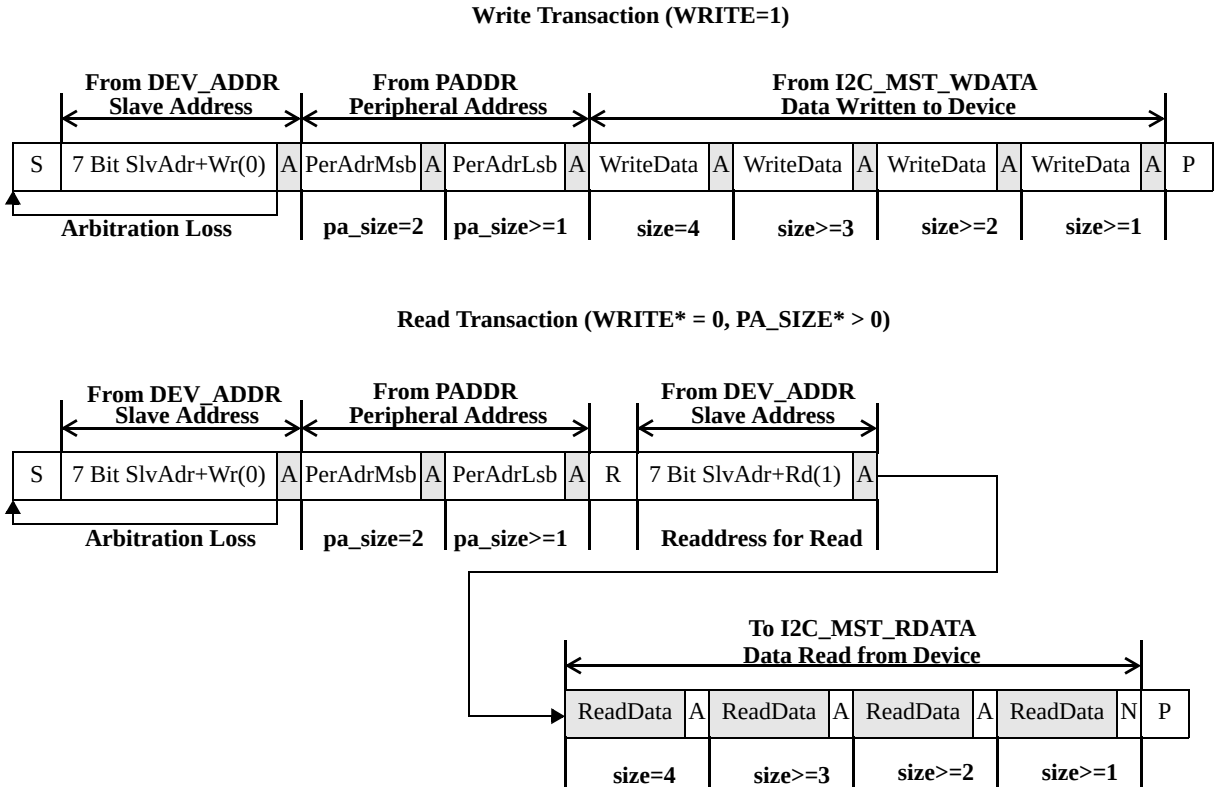
Because EEPROM devices do not have reset pins, if the Tsi574 is reset the EEPROM is unaffected and can continue to drive data at the previous state. For more information on how this issue may affect your application, and possible work around options, see "Masterless bus busy" in the *Tsi574 Design Notes*.

Software can instruct the Tsi574 to read or write to an external slave device using the following registers:

- "I2C Master Configuration Register" to configure external device parameters
- "I2C Master Control Register" to select and start the transaction
- "I2C Master Receive Data Register" for data to be read (received)
- "I2C Master Transmit Data Register" for data to be written (transmitted)
- "I2C Access Status Register" for status

Figure 31 depicts the sequences on the I2C bus when the Tsi574 is mastering a read or write transaction:

Figure 31: Software-initiated Master Transactions



Note: WRITE resides in the "I2C Master Control Register", while PA_SIZE resides in the "I2C Master Configuration Register".

The overall procedure is to configure the device through the "I2C Master Configuration Register", load the data to be written in the "I2C Master Transmit Data Register" (only needed for write operations), then load the "I2C Master Control Register" with the specific transaction information and have the START bit in that register set to 1. This initiates the master transaction. Software usually then waits for a master-related interrupt to know when the transaction has completed, or the status can be polled using the "I2C Access Status Register". If the operation was a read, the data can then be retrieved by reading the "I2C Master Receive Data Register".

Once a master operation is started, it cannot be aborted by software except when the I2C Interface is reset using the "I2C Reset Register". Reset using the "I2C Reset Register" is not recommended, however, since it can leave the I2C bus in a state that makes it difficult to ensure that all devices are back to an idle state.

The internal delay required by an EEPROM device to complete a write operation to its memory array must be managed by software. The I2C Interface only tracks the status of the operation by the bus signals.

7.4.1 Example EEPROM Read and Write

This section shows example pseudocode for writing and reading the EEPROM. This example is configured in the context of register writes that must be made during a boot load of the EEPROM.

7.4.1.1 Write Example

```
//Write 8 bytes to the EEPROM
w 1d114 0x0042FFFF // load the write data register with the write contents
w 1d10c 0xc4000000 // load the destination address in EEPROM and go
w 1d114 0xFFFFFFFF // load the next 4 bytes into the write data register
w 1d10c 0xc4000004 // load the destination address and go
```

7.4.1.2 Read Example

```
//Read the locations programmed above
w 1d10c 0x84000008 // set the start bit and address
r 1d110 [receive data] // get the received data
w 1d10c 0x8400000c // same again - set start with the address
w 1d10c [receive data] // get the received data
```

7.4.2 Master Clock Generation

The I2C clock (I2C_SCLK) for master operation is generated by dividing down the reference clock (P_CLK). The reset value for the I2C_SCLK generates a nominal 100-kHz operating speed. The bus speed is affected by external devices stretching the clock.

For master operations, the clock frequency can be changed by modifying the timing parameter registers (see "Bus Timing"). Operation above 100 kHz is possible but the Tsi574 does not implement all the standards requirements for fast mode.

7.4.3 Master Bus Arbitration

Because the Tsi574 can operate in a multi-master I2C system, it arbitrates for the I2C bus as required by the *I2C Specification*. During the Start and Slave Address phase, any unexpected state on the bus causes the Tsi574 to back off, release the bus, and wait for a Stop before retrying the transaction. If the arbitration timer configured in the "I2C_SCLK Low and Arbitration Timeout Register" expires before the device can get through the slave address phase without collision, then the transaction is aborted with the MA_ATMO status in the "I2C Interrupt Status Register". An optional interrupt can also be sent to the Interrupt Controller if enabled in the "I2C Interrupt Enable Register".

If the Tsi574 loses the arbitration for the I2C bus, and the winning master selects the Tsi574 as the target of its access, the Tsi574 responds as the slave.

7.4.4 Master External Device Addressing

A master transaction starts with a Start condition followed by the 7-bit slave address from the DEV_ADDR field of the "I2C Master Configuration Register", followed by the R/W bit. Assuming the 8 bits did not collide with another master, an ACK/NACK response bit is expected. If an ACK is received, the slave device exists and the transaction proceeds. If a NACK is received, the slave device is presumed to not exist and the transaction is aborted with a Stop and an MA_NACK status in the I2C_INT_STAT register, and an optional MA_NACK interrupt to the Interrupt Controller if enabled in MA_NACK of "I2C Interrupt Enable Register". In this case, the transaction is not automatically retried and it is up to software to retry if needed.

If the master transaction addresses the slave address of the Tsi574, the slave logic responds correctly as the target device.

7.4.5 Master Peripheral Addressing

Some devices, such as EEPROMs, require a peripheral address to be specified to set a starting position in their memory or address space for the read or write. The Tsi574 supports transactions with 0, 1, or 2 bytes of peripheral address. Because this is device dependent, the correct setting for the target device must be set in PA_SIZE of the "I2C Master Configuration Register"; otherwise, the transaction protocol is not correct.

The peripheral address is also set in the "I2C Master Configuration Register" when the transaction is started. If the transaction is a read, the Tsi574 must switch the I2C bus protocol to read mode following the peripheral address (sending the peripheral address requires write mode). To do this, a Restart condition is generated, followed by a repeat of the slave address to readdress the device in read mode. Because the bus was not released by the Restart, this phase is not subject to the arbitration timer, therefore any mismatch on the bus aborts the transaction with a MA_COL interrupt. This restart is not necessary if there is no peripheral address status in the "I2C Interrupt Status Register". An optional interrupt can also be sent to the Interrupt Controller if enabled in MA_COL of the "I2C Interrupt Enable Register".

7.4.6 Master Data Transactions

After the peripheral address phase, if any, 0 to 4 bytes of data are read or written, followed by the Stop condition. The number of bytes to be transferred is set in the SIZE field of the "I2C Master Control Register" when the operation is started, the type of transaction (read or write) is set in the WRITE field of the same register, and the order of byte transfer is set in the DORDER field of the "I2C Master Configuration Register".

For a write transaction, bytes are taken from the "I2C Master Transmit Data Register" based on DORDER and sent to the target device. Each byte must be ACKed by the device. If a NACK is received, the transaction is aborted with a Stop, an MA_NACK interrupt status in the "I2C Interrupt Status Register". An optional interrupt can also be sent to the Interrupt Controller if enabled in MA_NACK of the "I2C Interrupt Enable Register".

For a read transaction, bytes are received from the target device and placed in the "I2C Master Receive Data Register" based on DORDER. Each byte is ACKed by the Tsi574, except for the final byte that is NACKed to tell the target device to stop sending data, followed by a Stop condition to idle the bus.

Upon successful completion of a transaction, the MA_OK interrupt status is updated in the "I2C Interrupt Status Register". An optional interrupt can also be sent to the Interrupt Controller if enabled in MA_OK of the "I2C Interrupt Enable Register".

If the Tsi574 experiences a chip reset while it is writing to an EEPROM, the write does not complete and the data at the target EEPROM address may be corrupted.

7.5 Tsi574 as I2C Slave

The Tsi574 can operate as a slave device on the I2C bus. An external master device places a transaction on the bus with a device address that matches that programmed in the SLV_ADDR field of the "I2C Slave Configuration Register", or matches the fixed SMBus Alert Response address. The external master can then read or write to the Tsi574 through a small block of 256 addresses called the *Tsi574 peripheral address space*, and do the following:

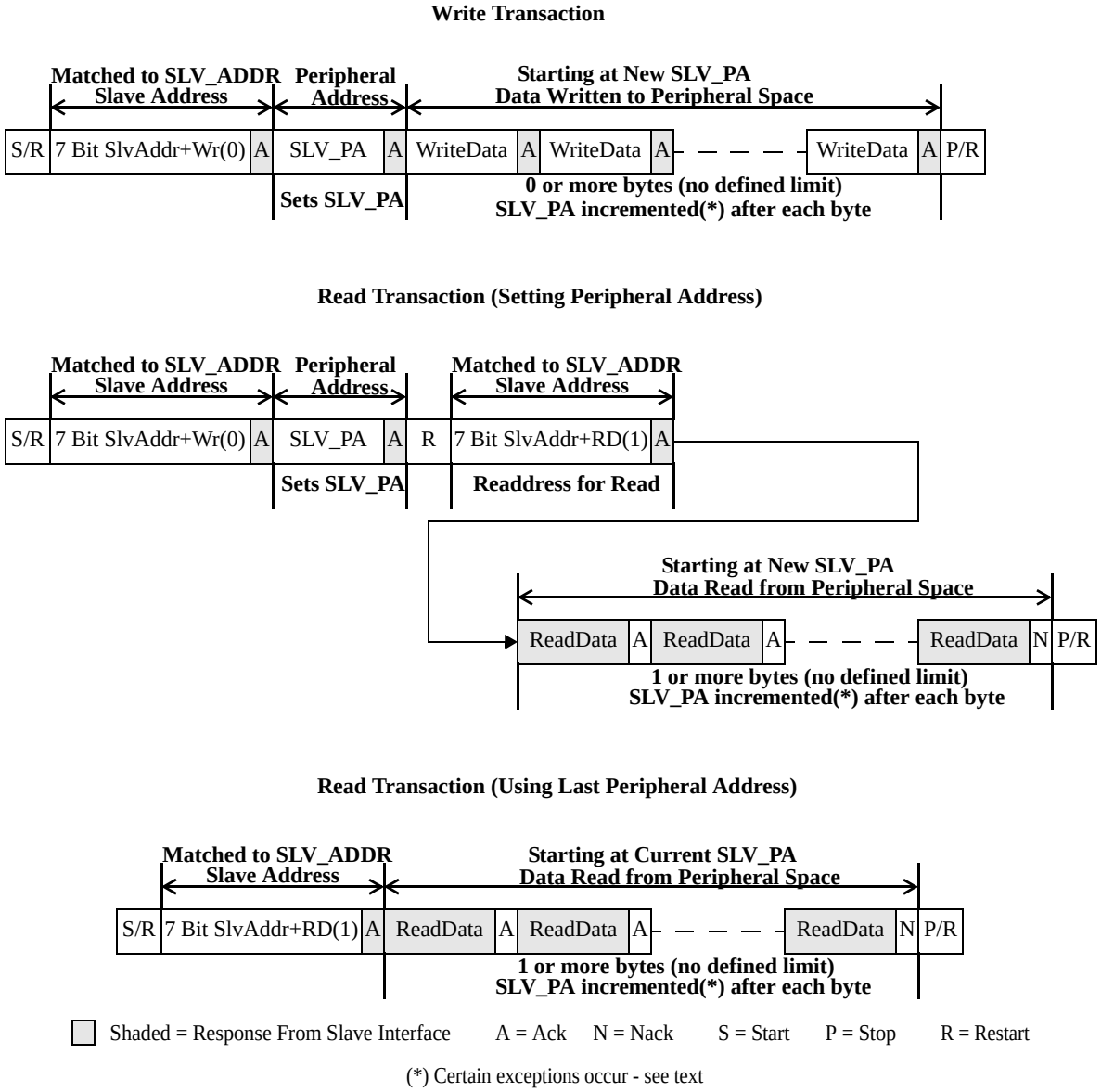
- Directly access limited status, read and write mailboxes
- Configure some options related to the slave access
- Indirectly read or write any other register in the Tsi574 that is accessible through the register bus

Figure 32 shows the bus protocols for accessing the Tsi574 as a slave device. The general procedure requires the external master to address the Tsi574, set the peripheral address to a position within the Tsi574 peripheral address space, then write or read some number of bytes. A write is terminated with a Stop or Restart, and a read is ended when the master responds to a byte with a NACK. There is no limit to the number of bytes that can be read or written in one transaction. The Tsi574 increments the peripheral address pointer after each byte, and wraps within the 256 space (see "Slave Peripheral Addressing"). Read and write transactions can be mixed by the external master issuing a Restart instead of a Stop, then sending a new transaction that addresses the Tsi574 (all writes must include the peripheral address byte).

At the completion of any slave transaction, either the SA_OK or SA_FAIL interrupt status is updated in the "I2C Interrupt Status Register". An optional interrupt can be sent to the Interrupt Controller, if enabled in SA_OK or SA_FAIL of the "I2C Interrupt Enable Register", to alert the processor/host that an external device has accessed the peripheral address space.

In addition, either the SA_WRITE or SA_READ interrupt status is updated in the "I2C Interrupt Status Register" if a read or write to the internal register space was triggered by the access. An optional interrupt can also be sent to the Interrupt Controller if enabled in SA_WRITE or SA_READ of the "I2C Interrupt Enable Register" The SA_FAIL interrupt indicates the slave transaction encountered an error. An SA_FAIL includes the slave logic detecting a collision when it was responding with data or an ACK/NACK, or one of the time-outs triggering and aborting the transaction (see "Timeouts"). If no error condition occurred, then the SA_OK interrupt is triggered.

Figure 32: Transaction Protocols for Tsi574 as Slave



7.5.1 Slave Clock Stretching

When the Tsi574 is a slave, the external master generates the I2C clock (I2C_SCLK). If the Tsi574 must access the internal register bus, I2C_SCLK is held low until data is available on a register read, or until a register write completes.

7.5.2 Slave Device Addressing

The Tsi574 supports 7-bit device addressing. The device address of the Tsi574 is set in the SLV_ADDR field of the "I2C Slave Configuration Register". For the Tsi574 to respond to an external master, the slave address on the bus must match either the address in the SLV_ADDR field, or the SMBus alert response address (see "SMBus Alert Response Protocol Support". However, an address of all zeros (0000000) never triggers a response because this address is used for the START byte and General Call protocol. Neither the START byte or General Call protocol are supported by the Tsi574.

The SLV_ADDR field can be changed at any time, either using the boot load sequence or by the processor/host. At power-up, the lower 2 bits of the 7-bit device address (defined by SLV_ADDR) are loaded from the PWRUP_I2C pin. This allows the user to support up to four separate Tsi574 devices on the same I2C bus.Changing the SLV_ADDR field does not change the value of the lower 2 bits unless those bits are unlocked by first setting the SLV_UNLK field in the "I2C Slave Configuration Register".

If the Tsi574 masters an I2C transaction and the device address matches the 7-bit address programmed by SLV_ADDR, the Tsi574 will respond to its own transaction. This is the only method that allows software to write to any of the externally visible registers that are read-only from the internal register bus.

The Tsi574 only responds as a slave if the SLV_EN bit in the "I2C Slave Configuration Register" is set to 1.

7.5.3 Slave Peripheral Addressing

The Tsi574 peripheral space comprises an addressable range of 256 bytes (from 0x00 to 0xFF) that can be directly read and written by an external I2C master device. When an external master sets the peripheral address, this sets a pointer (viewable in the SLV_PA field of the "I2C Access Status Register") maintained in the Tsi574 that determines where bytes read and written by the external master are within the peripheral address space.

This 256-byte space is mapped to the Externally Visible Registers within the I2C Interface; these registers all start with EXI2C_ (see "Externally Visible I2C Internal Write Address Register"). These registers can be accessed either directly by an external master using the addresses in the peripheral address space, or by the processor/host using the internal register bus addresses. Depending on how the registers are accessed also defines their properties: for example, some registers are read-only through the peripheral address space but read/write through the internal register bus, and vice-versa.

The next section discusses the mapping between the peripheral address space and the externally visible registers.

7.5.4 External I2C Register Map

Table 17 lists the register map that is visible to external I2C devices. The lowest peripheral address maps to the LSB of the register, while the highest peripheral address maps to the MSB of the register.

The external master can set the peripheral address to any location in the 256-byte range. If that byte maps to a register, and the byte is read or written, then the specific byte within the register is read or written. These reads and writes are not through the internal address bus but are instead local to the I2C Interface. If the peripheral address does not map to a register, then a read returns 0 and a write has no effect except to increment the peripheral address.

When a byte is read or written, the peripheral address is automatically incremented to the next value, with three exceptions listed in the table (addresses 0x07, 0x17, and 0xFF).

Table 17: Externally Visible I2C Register Map

Tsi574 Peripheral Address Range	Mapped Register	Description
0x00–0x03 R/W	EXI2C_REG_WADDR	Specifies the 4-byte aligned internal register address for the register bus write access performed when EXI2C_REG_WDATA is written.
0x04–0x07 R/W	EXI2C_REG_WDATA	Specifies the data to write to the internal register address held in EXI2C_REG_WADDR. Side effects: When address 0x07 (the MSB) is written, the data in this register is written to the internal register address held in EXI2C_REG_WADDR, and the peripheral address is returned to 0x04 (the LSB). This allows consecutive internal registers to be written in one transaction without resetting the peripheral address. Note: If 0x07 is read, the peripheral address increments to 0x08; if written, the peripheral address does not increment.
0x08–0x0F Read-Only	Reserved	This range does not map to any registers.
0x10–0x13 R/W	EXI2C_REG_RADDR	Specifies the 4-byte aligned internal register address for the register bus read access performed when EXI2C_REG_RDATA is read.
0x14–0x17 Read-Only	EXI2C_REG_RDATA	Returns the data read from the internal register address held in EXI2C_REG_RADDR. Side effects: When address 0x14 (the LSB) is read, the data in this register is loaded from the internal register address held in EXI2C_REG_RADDR, before the data is returned on the I2C bus. When peripheral address 0x17 (the MSB) is read, the peripheral address is returned to 0x14 (the LSB). This allows consecutive internal registers to be read in one transaction without resetting the peripheral address. Note: If 0x17 is written, the peripheral address increments to 0x18; if written, the peripheral address does not increment.
0x18–0x1F Read-Only	Reserved	This range does not map to any registers.

Table 17: Externally Visible I2C Register Map *(Continued)*

Tsi574 Peripheral Address Range	Mapped Register	Description
0x20–0x23 Read-Only	EXI2C_ACC_STAT	Returns status information on accesses performed by external devices, on the incoming/outgoing mailboxes and on the state of the alert response flag.
0x24–0x27 R/W	EXI2C_ACC_CNTRL	Provides control information on how the Tsi574 handles internal register accesses through the EXI2C_REG_RDATA and EXI2C_REG_WDATA registers.
0x28–0x7F Read-Only	Reserved	This range does not map to any registers.
0x80–0x83 Read-Only	EXI2C_STAT	Returns a summary of the internal status of the Tsi574.
0x84–0x87 R/W	EXI2C_STAT_ ENABLE	Enables the bits in the status summary (EXI2C_STAT) to set the alert flag in the EXI2C_ACC_STAT register.
0x88–0x8B Read-Only	Reserved	This range does not map to any registers.
0x90–0x93 Read-Only	EXI2C_MBOX_OUT	This register allows a RapidIO-enabled processor to transfer a 32-bit message to an external I2C master.
0x94–0x97 R/W	EXI2C_MBOX_IN	This register allows an external I2C master to transfer a 32-bit message to a RapidIO-enabled processor.
0x98–0xFF Read-Only	Reserved	This range does not map to any registers. Side effects: When peripheral address 0xFF is read or written, the peripheral address wraps back to 0x00.

7.5.5 Slave Write Data Transactions

An external master must set the peripheral address as part of a write transaction before transferring any data. The effect of the written data depends on the register in which the peripheral address maps (see Table 17). The peripheral address is usually incremented after each byte such that consecutive bytes are written into increasing addresses within the peripheral address space. Certain exceptions exist, however, as indicated in Table 17. In addition, a write that hits the most significant byte of the "Externally Visible I2C Internal Write Address Register" (peripheral address 0x07) has the side-effect of triggering a write to a register on the Tsi574 internal register bus.

7.5.6 Slave Read Data Transactions

An external master is not required to set the peripheral address as part of a read transaction, but can do so by first writing the peripheral address and then issuing a Restart before writing any data. If not set, the read data starts wherever the peripheral address pointer was left by the previous transaction. Because it is possible that another master changed the pointer, it is recommended that the peripheral address be set at the start of a transaction. Data is returned to the external master from consecutive bytes within the peripheral address space, with certain exceptions indicated in Table 17.

There also can be side effects to reading some bytes (see Table 17 and the EXI2C register descriptions). For example, a read that hits the LSB of the "Externally Visible I2C Internal Read Address Register" (peripheral address 0x14), also triggers a read from a register on the Tsi574 internal register bus.

7.5.7 Slave Internal Register Accesses

The Tsi574 allows external masters to access all of its internal registers through the externally visible I2C registers.

The address in the register definitions refers to an offset only. The offset must be prefixed with the block address.

The "Externally Visible I2C Internal Read Data Register" is a special register in that a read operation to the first (least significant) byte of this register through the peripheral address space (0x14) triggers the Tsi574 to perform an internal register read operation using the address in the "Externally Visible I2C Internal Read Address Register". When the internal register read operation is completed, the data is first loaded into the "Externally Visible I2C Internal Read Data Register", then returned to the external I2C master byte by byte. The "Externally Visible I2C Internal Access Control Register" controls the internal register read operations and allows the user to specify when and how the register read is performed.

Likewise, the "Externally Visible I2C Internal Write Data Register" triggers the Tsi574 to perform an internal register write operation using the address in the "Externally Visible I2C Internal Write Address Register". The external device writes data to the "Externally Visible I2C Internal Write Data Register" through the peripheral space, and when the last (most significant) byte is written (0x07), the register contents is written through the internal register bus to the address in the "Externally Visible I2C Internal Write Address Register". The "Externally Visible I2C Internal Access Control Register" controls the internal register write operation and allows the user to specify when and how the register write is performed.

Internal register accesses can be prohibited by the processor/host through the RD_EN and WR_EN fields in the "I2C Slave Configuration Register".

At the completion of a slave transaction that includes a successful read or write to an internal register, a SA_WRITE or SA_READ interrupt status is updated in the I2C_INT_STAT register. An optional interrupt can also be sent to the Interrupt Controller if enabled in SA_WRITE and SA_READ of "I2C Interrupt Enable Register". This allows the processor/host to be aware that an external device is accessing the internal registers of the Tsi574.

7.5.8 Slave Access Examples

This section shows a slave internal register access by an external master. The following abbreviations are used:

<S> Start condition
<R> Restart condition
<SLVA> The 7-bit Tsi574 slave address (that matches SLV_ADDR)
<PA=#> 8-bit peripheral address (current setting viewable in SLV_PA)
<A> Acknowledge (ACK)
<N> Not acknowledge (NACK)
<P> Stop condition
<W> Write
<$\overline{W}$> Read (not write)
<WD=#> 8-bit write data (from master to Tsi574)
<RD=#> 8-bit read data (from Tsi574 to master)

Also, registers and register fields are referenced by name.

All examples assume that the transactions occur in the order given; only one master is accessing (such that nothing is changed by another master between transactions); and nothing is changed by the processor/host during the transactions.

The following conditions pre-exist: ALERT_FLAG is set in the "Externally Visible I2C Slave Access Status Register".

1. External device reads "Externally Visible I2C Slave Access Status Register" (LSB only). The returned value of 0x01 is the ALERT_FLAG. External device must NACK after the first read byte to stop the transfer.

   I2C Sequence: <S><SLVA><W><PA=0x20><A><R><SLVA><$\overline{W}$><RD=0x01><N><P>

   Following the transaction, SLV_PA is 0x21 and interrupt status SA_OK asserts. An optional interrupt can also be sent to the Interrupt Controller if enabled in SA_OK of "I2C Interrupt Enable Register".

2. External device reads "Externally Visible I2C Status Register" (all 4 bytes). Note that the data from this register is returned LSB to MSB. External device must NACK the fourth byte to stop the transfer.

   I2C Sequence:

   <S><SLVA><W><PA=0x80><A><R><SLVA><$\overline{W}$>

   <RD=0x56><A><RD=0x34><A><RD=0x12><A><RD=0x80><N><P>

   Following the transaction, SLV_PA is 0x84 and interrupt status SA_OK asserts. An optional interrupt can also be sent to the Interrupt Controller if enabled in SA_OK of "I2C Interrupt Enable Register".

3. External device does an Alert Response request. Because ALERT_FLAG is asserted, the alert response address is ACK'd and the Tsi574 slave address is returned.

   I2C Sequence: <S><0001100><$\overline{W}$><A><RD=SLVA+0><N><P>

   Following the transaction, SLV_PA is 0x84 (unchanged from previous transaction) and interrupt status SA_OK asserts. An optional interrupt can also be sent to the Interrupt Controller if enabled in SA_OK of "I2C Interrupt Enable Register". The ALERT_FLAG in "Externally Visible I2C Slave Access Status Register" is cleared because of the successful response.

4. External device writes all 4 bytes of the "Externally Visible I2C Enable Register" to 0, reads "Externally Visible I2C Slave Access Status Register", then does another Alert Response request. The ALERT_FLAG is zero (all enables were cleared), so the alert response address is NACKed.

   I2C Sequence:

   <S><SLVA><W><PA=0x84><A>

   <WD=0x00><A><WD=0x00><A><WD=0x00><A><WD=0x00><A>

   <R><SLVA><W><PA=0x20><A><R><SLVA><$\overline{W}$><RD=0x00><N>

   <R><0001100><$\overline{W}$><N><P>

   Following the transaction, SLV_PA is 0x21, "Externally Visible I2C Enable Register" is 0x00000000 and interrupt status SA_OK asserts. An optional interrupt can also be sent to the Interrupt Controller if enabled in SA_OK of "I2C Interrupt Enable Register".

5. External device writes "Externally Visible I2C Internal Access Control Register" to enable internal register auto-incrementing.

   I2C Sequence: <S><SLVA><W><PA=0x24><A><WD=0xAC><A><P>

   Following the transaction, SLV_PA is 0x25, "Externally Visible I2C Internal Access Control Register" is 0x000000AC and interrupt status SA_OK asserts. An optional interrupt can also be sent to the Interrupt Controller if enabled in SA_OK of "I2C Interrupt Enable Register".

6. External device sets up "I2C_SCLK Low and Arbitration Timeout Register" address (0x354) in EXI2C_REG_WADDR, then writes three registers back-to-back with 0x11223344, 0x55667788, and 0x99AABBCC. Because of the register auto-increment, and because the PA auto-wraps from 0x07 to 0x04, the writes can be completed in a stream. Note that data is written from LSB to MSB.

   I2C Sequence:

   <S><SLVA><W><PA=0x00><A>

   <WD=0x54><A><WD=0x03><A><WD=0x00><A><WD=0x00><A><R>

   <0x44><A><0x33><A><0x22><A><0x11><A>

   <0x88><A><0x77><A><0x66><A><0x55><A>

   <0xCC><A><0xBB><A><0xAA><A><0x99><A><P>

   Following the transaction, SLV_PA is 0x04, "Externally Visible I2C Internal Write Address Register" is 0x0001_D360, "I2C_SCLK Low and Arbitration Timeout Register" is 0x11223344, "I2C Byte/Transaction Timeout Register" is 0x55667788, and "I2C Boot and Diagnostic Timer" I2C_BOOT_DIAG_TIMER is 0x8000BBCC (reserved fields stay zero) and interrupt status SA_OK and SA_WRITE assert. An optional interrupt can also be sent to the Interrupt Controller if enabled in the "I2C Interrupt Enable Register".

7. External device sets up "I2C_SCLK Low and Arbitration Timeout Register" address (0x1D354) in "Externally Visible I2C Internal Read Address Register", then reads 3 registers back-to-back. Because of the register auto-increment, and because the PA auto-wraps from 0x17 to 0x14, the reads can be completed in a stream. Note that data is read from LSB to MSB.

   I2C Sequence:

   <S><SLVA><W><PA=0x10><A>

   <WD=0x54><A><WD=0xD3><A><WD=0x01><A><WD=0x00><A><R>

   <S><SLAV><$\overline{W}$><A>

   <RD=0x44><A><RD=0x33><A><RD=0x22><A><RD=0x11><A>

   <RD=0x88><A><RD=0x77><A><RD=0x66><A><RD=0x55><A>

   <RD=0xCC><A><RD=0xBB><A><RD=0x00><A><RD=0x80><N><P>

   Following the transaction, SLV_PA is 0x14, "Externally Visible I2C Internal Read Address Register" is 0x0001_D360, and interrupt status SA_OK and SA_READ assert. An optional interrupt can also be sent to the Interrupt Controller if enabled in SA_OK and SA_READ of the "I2C Interrupt Enable Register".

8. External device writes "Externally Visible I2C Internal Access Control Register" to disable internal register auto-incrementing.

   I2C Sequence: <S><SLVA><W><PA=0x24><A><WD=0xA0><A><P>

   Following the transaction, SLV_PA is 0x25, "Externally Visible I2C Internal Access Control Register" is 0x000000A0, and interrupt status SA_OK asserts. An optional interrupt can also be sent to the Interrupt Controller if enabled in SA_OK of "I2C Interrupt Enable Register".

9. External device sets up "I2C_SCLK Low and Arbitration Timeout Register" address (0x1D354) in "Externally Visible I2C Internal Read Address Register", then reads same register 3 times. The register address no longer auto-increments, but the PA still auto-wraps from 0x17 to 0x14, so the reads can be completed in a stream. Note that data is read from LSB to MSB.

   I2C Sequence:

   <S><SLVA><W><PA=0x10><A>

   <WD=0x54><A><WD=0xD3><A><WD=0x01><A><WD=0x00><A><R>

   <S><SLAV><$\overline{W}$><A>

   <RD=0x44><A><RD=0x33><A><RD=0x22><A><RD=0x11><A>

   <RD=0x44><A><RD=0x33><A><RD=0x22><A><RD=0x11><A>

   <RD=0x44><A><RD=0x33><A><RD=0x22><A><RD=0x11><N><P>

   Following the transaction, SLV_PA is 0x14, "Externally Visible I2C Internal Read Address Register" is0x001D354, and interrupt status SA_OK and SA_READ assert. An optional interrupt can also be sent to the Interrupt Controller if enabled in SA_OK and SA_READ of "I2C Interrupt Enable Register".

7.5.9 Resetting the I2C Slave Interface

The I2C slave interface is reset by two conditions: chip reset or the detection of a START condition. When a chip reset is applied, the I2C slave interface immediately returns to the idle state. Any active transfer, to or from the Tsi574 when the reset is asserted, is interrupted. All registers are initialized by a full-chip reset.

As required by the *I2C Specification*, the Tsi574 resets its bus interface logic on receipt of a START or repeated START condition such that it anticipates receiving a device address phase, even if the START condition is not positioned according to the proper format. The I2C registers, however, are not reset.

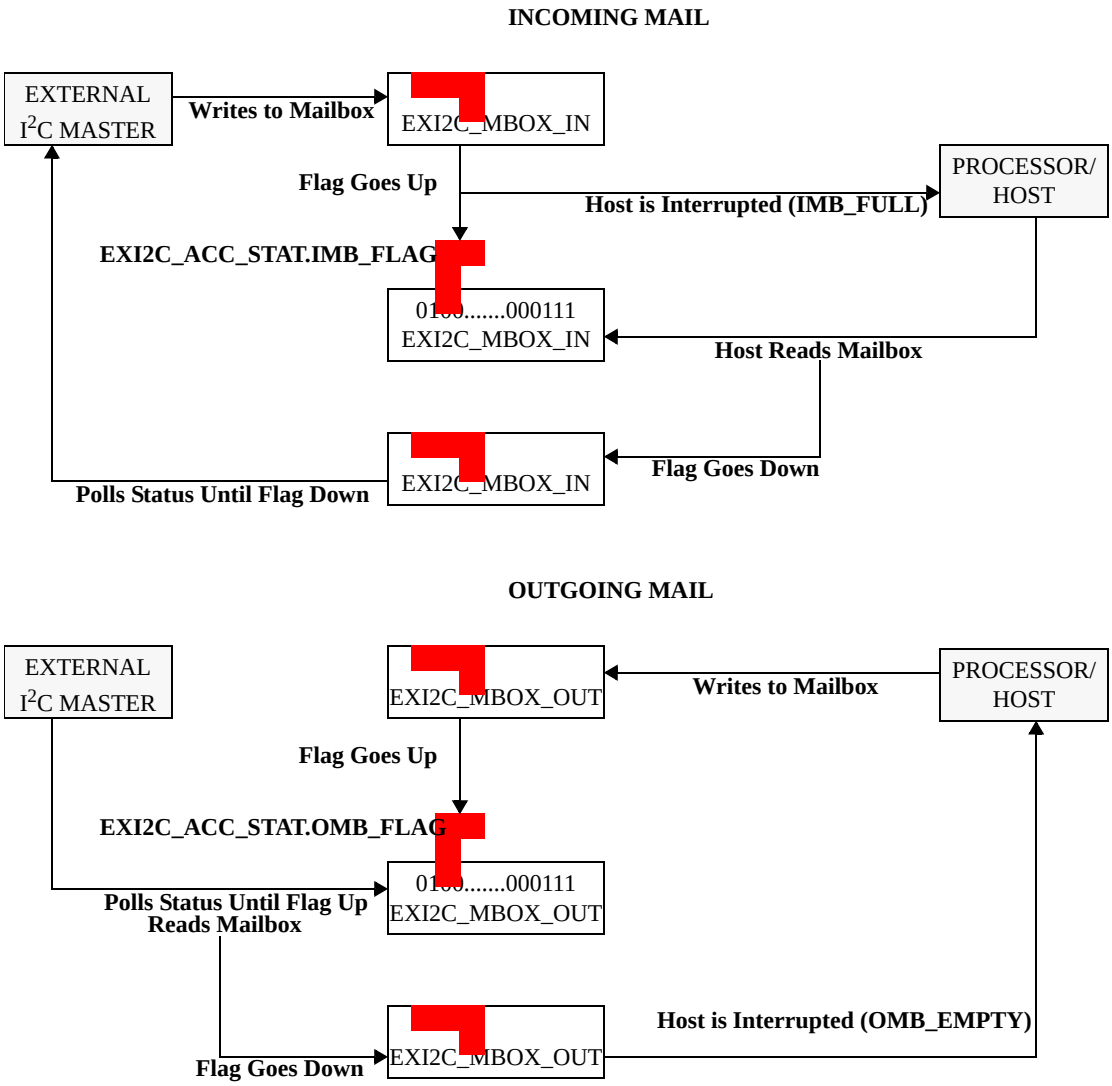
7.6 Mailboxes

As part of the peripheral address space on the Tsi574, the following registers act as I2C mailboxes for communicating information between an external I2C master and a processor or host:

- "Externally Visible I2C Incoming Mailbox Register" for data incoming from an external I2C master to the processor or host.
- "Externally Visible I2C Outgoing Mailbox Register" for data outgoing from the processor or host to an external I2C master.

In addition, flags in the "Externally Visible I2C Slave Access Status Register" are accessible by an external host to examine the mailbox status. Figure 33 shows the use of I2C mailboxes.

Figure 33: I2C Mailbox Operation



The incoming and outgoing I2C mailbox registers are discussed further in the following sections.

7.6.1 Incoming Mailbox

To send data to the processor/host, an external I2C master writes data to the incoming mailbox register, EXI2C_MBOX_IN, through the Tsi574 slave interface. When the Stop condition is seen (indicating the external master is completed writing to the mailbox), the slave interface sets the IMB_FLAG in the "Externally Visible I2C Slave Access Status Register", and the processor/host is interrupted to let it know the mailbox is full by setting IMB_FULL status in the "I2C Interrupt Status Register". An optional interrupt can be sent to the Interrupt Controller if enabled in IMB_FULL of the "I2C Interrupt Enable Register".

In response to the interrupt, the processor/host reads the incoming mailbox to retrieve the data. This process of reading the register clears the IMB_FLAG in the status register. The external I2C master can poll the status register through the slave interface, and when it sees the flag go to 0, it knows the processor/host has read the data and it is safe to write more. If the external I2C master writes more data before earlier data is read, the old data is overwritten. In this case, depending on timing, the processor/host may read a mixture of old and new data.

7.6.2 Outgoing Mailbox

To send data to an external I2C master, the processor/host writes data to the outgoing mailbox register, EXI2C_MBOX_OUT. This sets the OMB_FLAG in the "Externally Visible I2C Slave Access Status Register", which the external I2C master can poll through the slave interface. When the flag goes up (1), the external I2C master reads the outgoing mailbox register through the slave interface.

Once the Stop condition is seen (indicating the external master has completed reading the mailbox), the slave interface clears the OMB_FLAG in the status register, EXI2C_ACC_STAT, and interrupts the processor/host with an OMB_EMPTY in the "I2C Interrupt Status Register" to let it know the mailbox has been read and it is safe to write more data. An optional interrupt can be sent to the Interrupt Controller if enabled in OMB_EMPTY of "I2C Interrupt Enable Register". If the processor/host writes more data to the mailbox before the data is read, the old data is overwritten. In this case, depending on timing, the external I2C master may read a mixture of old and new data.

7.7 SMBus Support

The I2C Interface provides limited functionality for SMBus applications. The Tsi574 can act as an SMBus Host and communicate to other SMBus slave devices through a subset of the SMBus protocols (see "SMBus Protocol Support").

As a host, the Tsi574 cannot effectively receive a SMBus Host Notify message sent by another non-host SMBus device acting as a master. In addition, the Tsi574 cannot effectively act as a non-host SMBus device and receive commands from an external SMHost. Although the Tsi574 responds as a slave to the SMBus protocols, they are processed relative to the slave interface functionality. The SMBus command code is assumed to be a peripheral address, and data written to or read through the slave interface will depend on the peripheral address selected.

7.7.1 Unsupported SMBus Features

The Tsi574 does not support the following SMBus features:

- Non-host response to external SMBus host protocols, except for Alert Response Protocol
- Address Resolution Protocol (ARP) or any related commands
- SMBSUS# (suspend mode signal pin)
- Packet Error Code (PEC). The Tsi574 does not generate, check, or expect PECs
- Process Call command
- Block write command with more than 4 data bytes
- Block read command
- Block write-block read process call command
- SMBus host notify protocol as a SMBus host device in slave mode

7.7.2 SMBus Protocol Support

The Tsi574 master interface functionality supports a subset of the SMBus Protocols (see Figure 34). In all cases, the Tsi574 masters a transaction to another SMBus device. All register and register field references are to the following I2C master interface registers:

- "I2C Master Configuration Register"
- "I2C Master Control Register"
- "I2C Master Receive Data Register"
- "I2C Master Transmit Data Register"

Use of the Quick Command with Read requires the target device to support the command. Under normal I2C protocol, the slave device returns data on the bus following the ACK, so the master could not always generate the required Stop condition. The target device must release the bus following the ACK if it is responding to this command.

Figure 34: SMBus Protocol Support

SMBus Quick Command with Write, PA_SIZE*=0, SIZE*=0, WRITE*=1

S	SlaveAdr	Wr	A	P

SMBus Quick Command with Read, PA_SIZE=0, SIZE=0, WRITE=0

S	SlaveAdr	Rd	A	P

SMBus Send Byte, PA_SIZE=1, SIZE=0, WRITE=1

S	SlaveAdr	Wr	A	PA0=Data	A	P

SMBus Receive Byte, PA_SIZE=0, SIZE=1, DORDER*=1, WRITE=0

S	SlaveAdr	Rd	A	RD0=Data	N	P

SMBus Write Byte, PA_SIZE=1, SIZE=1, DORDER=1, WRITE=1

S	SlaveAdr	Wr	A	PA0=Cmd	A	TD0	A	P

SMBus Write Word, PA_SIZE=1, SIZE=2, DORDER=1, WRITE=1

S	SlaveAdr	Wr	A	PA0=Cmd	A	TD0	A	TD1	A	P

SMBus Read Byte, PA_SIZE=1, SIZE=1, DORDER=1, WRITE=0

S	SlaveAdr	Wr	A	PA0=Cmd	A	R	SlaveAdr	Rd	A	RD0	N	P

SMBus Read Word, PA_SIZE=1, SIZE=2, DORDER=1, WRITE=0

S	SlaveAdr	Wr	A	PA0=Cmd	A	R	SlaveAdr	Rd	A	RD0	A	RD1	N	P

SMBus Write Block (NB = 1-4 bytes), PA_SIZE=2, SIZE=NB, DORDER=1, WRITE=1

S	SlaveAdr	Wr	A	PA1=Cmd	A	PA0=NB	A	TD0	A	- - - - -	TD(NB-1)	A	P

SMBus Host Notify Protocol, PA_SIZE=1, SIZE=2, DORDER=1, WRITE=1

S	SlaveAdr	Wr	A	PA0=DevAdr	A	TD0	A	TD1	A	P

SlaveAdr = SMBus device address set in DEV_ADDR
PA0 = LSB of PADDR
PA1 = MSB of PADDR
TD0 = LSB of I2C_MST_WDATA
TD3 = MSB of I2C_MST_TDATA
RD0 = LSB of I2C_MST_RDATA (data returned here)
RD3 = MSB of I2C_MST_RDATA (data returned here)

S = Start Condition
P = Stop Condition
R = Restart Condition
A = ACK
N = NACK
Unshaded = Master is driving the bus
Shaded = Slave is driving the bus

Wr = Write mode bit (0)
Rd = Read mode bit (1)

* For information about SIZE and WRITE, see "I2C Master Control Register". For information about PA_SIZE and DORDER, see "I2C Master Configuration Register".

7.7.3 SMBus Alert Response Protocol Support

The Tsi574 supports the SMBus Alert Response Protocol as either master or slave. As a master, an external device can be polled using a master read operation. As a slave, the Tsi574 slave interface responds to the Alert Response Address with the Tsi574's slave device address based on the value of ALERT_FLAG in the "Externally Visible I2C Slave Access Status Register", if enabled in ALRT_EN of "I2C Slave Configuration Register". Once the alert response is given and the Tsi574's slave device address is returned, the ALERT_FLAG is de-asserted. For the register fields indicated in Figure 34, reference the master interface registers I2C_MST_CFG, I2C_MST_CNTRL, and I2C_MST_RDATA, as well as the slave configuration register I2C_SLV_CFG.

Figure 35: SMBus Alert Response Protocol

SMBus Alert Response (master interface), DEV_ADDR = 0001100, PA_SIZE=0, SIZE=1, DORDER=1, WRITE=0

S	ARA	Rd	A	RD0=DevAdr	N	P

A device returns their address, loaded into read data register

S	ARA	Rd	N	P

No device responds to alert poll, operation fails with MA_NACK interrupt

SMBus Alert Response (slave interface), ALRT_EN=1, ALERT_FLAG=1

S	ARA	Rd	A	SLV_ADDR	N	P

Alert asserted from Tsi574, slave address is returned

SMBus Alert Response (slave interface), ALRT_EN=0 or ALERT_FLAG=0

S	ARA	Rd	N	P

No alert asserted from Tsi574, poll is NACK'd

ARA = SMBus Alert Response Address (0001100)
DevAdr = Slave address of external device asserting alert
SLV_ADDR = Slave address of Tsi574 from I2C_SLV_CFG
RD0 = LSB of I2C_MST_RDATA (data returned here)

S = Start Condition
P = Stop Condition
A = ACK
N = NACK
Shaded = Slave is driving the bus
Unshaded = Master is driving the bus

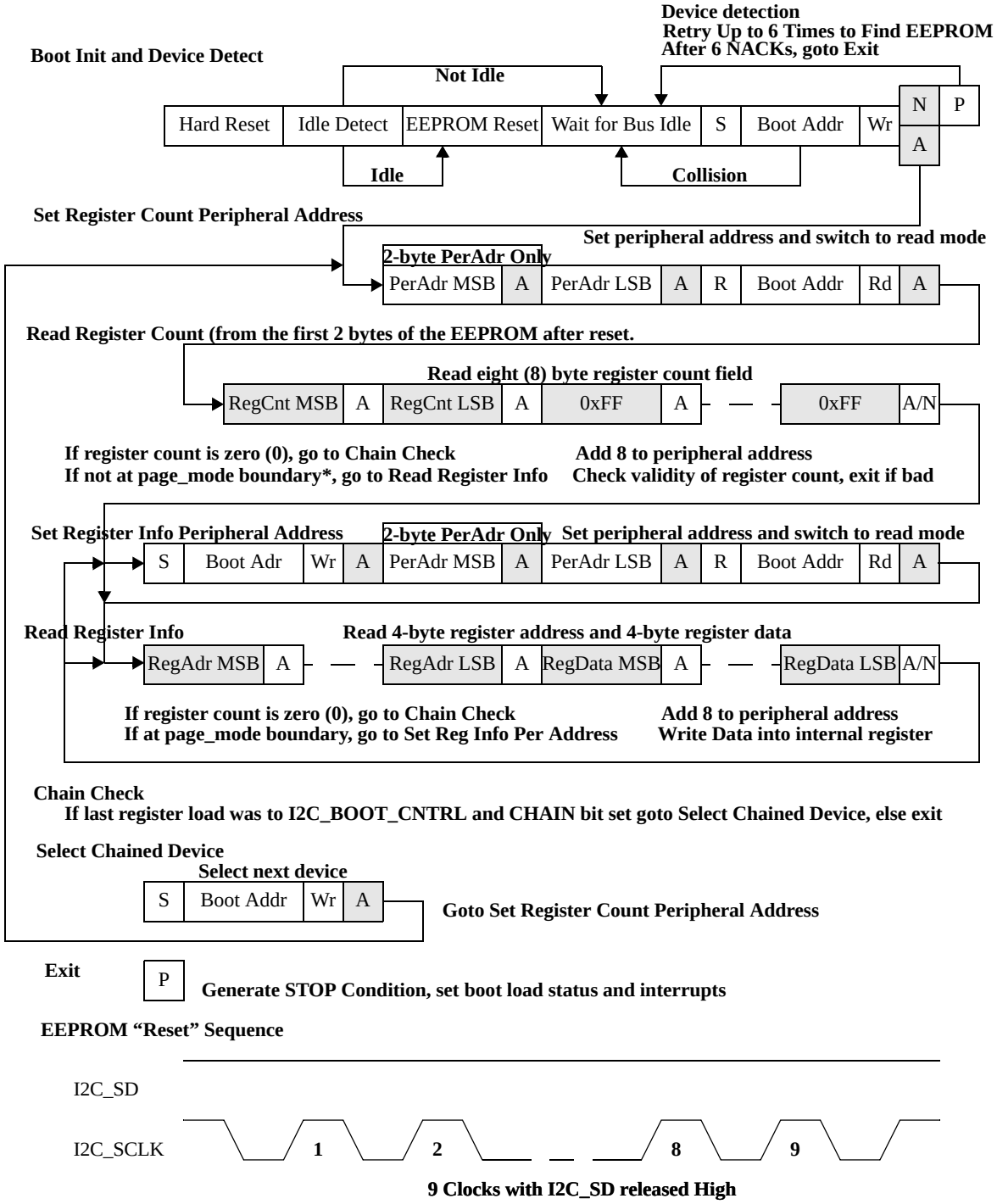
Rd = Read mode bit (1)

* For information about DEV_ADDR, DORDER, and PA_SIZE, see "I2C Master Configuration Register". For more information about SIZE and WRITE, see "I2C Master Control Register". For more information about ALRT_EN, see "I2C Slave Configuration Register".

7.8 Boot Load Sequence

Unless the I2C_DISABLE pin is held high, the Tsi574 will perform a post-hard reset register initialization sequence where it reads data from one or more external EEPROM devices (for more information about I2C_DISABLE, see "Power-up Options" in this document). This data initializes the Tsi574's internal registers. The boot load sequence occurs only after a full chip reset, and follows the steps shown in Figure 36. The boot load sequence is controlled by the contents of the I2C_BOOT_CNTRL register.

Figure 36: Boot Load Sequence



* For information about page mode boundary, see PAGE_MODE in "I2C Boot Control Register". For information about PA_SIZE, see "I2C Master Configuration Register".

7.8.1 Idle Detect

Upon exit from reset, it is unknown if another master is active. The Idle Detect period determines if the I2C_SCLK signal remains high long enough (roughly 50 microseconds) that it is unlikely another master is active. If I2C_SCLK is seen low during this period, it is assumed another master is active, the EEPROM Reset phase is skipped, and boot sequence proceeds to the Wait for Bus Idle phase. This detection is performed whether or not the boot sequence is disabled using the I2C_DISABLED pin. If the boot sequence is disabled, the BL_OK interrupt status is asserted immediately in the I2C_INT_STAT register, and an optional interrupt can be sent to the Interrupt Controller if enabled using BL_OK in the "I2C Interrupt Enable Register". If a master transaction is initiated before the idle detect completes, the transaction is started once the idle detect completes.

Upon exit from reset, if I2C_SDA is seen low the device assumes the bus is busy and does not attempt to reset the bus. Therefore the boot sequence will not take place.

7.8.2 EEPROM Reset Sequence

The EEPROM reset sequence is intended to cover the condition where a hard reset occurs while a transaction is active on the I2C bus. In this case, because the Tsi574 I2C master may have been reset and stopped generating the I2C_SCLK clock, one or more slave devices may be in a hung state where they are expecting a read or write to complete, and may be holding the I2C_SD signal low, preventing the generation of a STOP or START condition.

To try to force these devices out of their hung state, the Tsi574 allows the I2C_SD signal to stay high and generate 9 clock pulses on the I2C_SCLK signal. If no device was hung, this should not cause any problems because all devices are looking for a START condition. If a device was in the middle of a receiving a byte, the remainder of the byte will appear to have all 1s, and the device can generate an ACK or NACK. It is possible it may look to the device as if part of another byte is being sent, but because this is the master transmitting part of the protocol, the device will have released its control on the I2C_SD signal, so the master can force a START or STOP condition, even in the middle of the byte. If a device was in the middle of sending a byte, the clocks pulses will allow it to finish the transmission. The I2C_SD left high by the master (the Tsi574) will appear as a NACK to the device and it will not try to transmit another byte, but will leave the I2C_SD signal free so that another master can force a START or STOP condition.

This sequence is sent only once after a hard reset, and only if the Idle Detect phase was successful, and the Tsi574 believes it is not interfering with another master.

7.8.3 Wait for Bus Idle

Before attempting to access an EEPROM device, the boot loader waits for the bus to be idle. This is either the result of a successful Idle Detect phase, or, if the Idle Detect phase failed, once a STOP condition is seen on the bus, indicating another master has released control. In addition, if the I2C_SCLK and I2C_SD signals are both high for longer than the idle detect period while waiting for a STOP condition, the bus is assumed idle and the boot load process proceeds.

7.8.4 EEPROM Device Detection

Once the bus is available, the Tsi574 tries to connect to the EEPROM. A START condition is generated followed by BOOT_ADDR from the "I2C Boot Control Register". The upper 5 bits of that field reset to 0b10100 and the lower 2 bits are sampled from the I2C_SA[1:0] pins on exit from hard reset. This allows up to four unique Tsi574 devices to boot from different EEPROMs on the same I2C bus. If an ACK is received, which indicates the device is present, the sequence proceeds to the next phase. If there is a bus collision, the loader returns to the Wait for Bus Idle phase because another master has the bus. If a NACK is received, the process is retried from the Wait for Bus Idle phase up to 6 times in case the device was busy. If six NACKs are received, the boot load is aborted and the BL_FAIL interrupt status is asserted. An interrupt can also be sent to the Interrupt Controller if enabled using BL_FAIL in the "I2C Interrupt Enable Register".

7.8.5 Loading Register Data from EEPROM

Once the EEPROM is successfully addressed, the Tsi574 does not release the bus until the boot load is complete. First, the peripheral address is set. The address resets to 0, so the first EEPROM accessed must be loaded from address 0. The peripheral address is either 1 or 2 bytes depending on the state of the I2C_MA pin, which much be set appropriately depending on the type of EEPROM connected.

The boot loader then switches to read mode and reads the first 8 bytes, expecting to find a count of the number of registers to be initialized in the first 2 bytes, followed by 6 bytes of 0xFF. A validity check is completed on this field — if the number of registers exceeds the maximum (see "EEPROM Data Format"), or if any of the last 6 bytes are not 0xFF, it is assumed the EEPROM does not contain boot load data, the boot load is aborted and the BL_FAIL interrupt status is updated in the "I2C Interrupt Status Register". On these boot load status bits, the optional interrupt can be forwarded to the Interrupt Controller if enabled in the "I2C Interrupt Enable Register". If the register count was 0, the boot load is ended successfully and the BL_OK interrupt status is updated. An optional interrupt can also be forwarded to the Interrupt Controller if enabled in the "I2C Interrupt Enable Register". For information on the expected EEPROM data format used for boot loading, see "EEPROM Data Format".

The boot loader continues by reading eight bytes of data for each register to be loaded, and increments the peripheral address by 8. Depending on the PAGE_MODE field in the "I2C Boot Control Register", the peripheral address is periodically reset by issuing a Restart, re-selecting the boot device, and sending the updated peripheral address. On reset, the PAGE_MODE resets to a boundary of 8 such that initially the peripheral address is updated to the device after every register is loaded (see "Accelerating Boot Load"). In addition, for 1-byte peripheral addresses, if the BINC bit is 1, then when the peripheral address crosses a 256-byte boundary (that is, when the 1-byte address rolls over to 0x00), the LSB 3 bits of the BOOT_ADDR are incremented and the device is re-addressed. This supports those EEPROMs that use the lower 3 bits of their address as a 256-byte page indicator.

For each block of 8 bytes loaded, the first 4 bytes are the register address on the internal Tsi574 register bus, and the next 4 bytes are the 32-bit data value to be written to the register. No checking is completed for register address or data validity. As soon as all 8 bytes are read, the data is written to the internal address, the peripheral address count is updated, and the register count is decremented. Once the register count reaches 0 the boot load from the current EEPROM is complete, and, unless chaining is invoked, the boot load sequence is complete, a STOP condition is issued to release the bus, and the BL_OK interrupt status is updated. An optional interrupt can also be forwarded to the Interrupt Controller if enabled in the "I2C Interrupt Enable Register".

7.8.6 Chaining

The boot loader provides for booting from multiple EEPROMs, or from multiple sections within a single EEPROM (or any combination of both). This process is called *chaining*. Chaining is invoked during the boot load sequence when three conditions occur together:

- All the registers indicated by the register count are loaded
- The final register loaded was the "I2C Boot Control Register"
- The value loaded into the I2C_BOOT_CNTRL register had the CHAIN bit set

If these conditions are met, then the boot load sequence continues using the updated information in the I2C_BOOT_CNTRL register. This allows all aspects of the boot load to be changed – the device address, the peripheral address, and so forth. When a chain occurs, the boot load sequence addresses the new device and reads a new register count from the peripheral address. This address could be non-zero, so on a chain it is possible to start loading from other than address 0 in an EEPROM.

On a chain, it is important to set the peripheral address size (PSIZE), boot address increment (BINC) and page mode (PAGE_MODE) fields so they are valid for the new EEPROM; otherwise, the boot load process may be corrupted (for information about these bits, see "I2C Boot Control Register").

It may also be necessary to use the BOOT_UNLK field to change the lower 2 bits of the EEPROM address. By default, the BOOT_UNLK field is not set, so if the BOOT_ADDR field is changed, the lower 2 bits remain at their previous value. This way the power-up reset value is not inadvertently lost. If as part of the chaining process it is necessary to change those bits (such as if the boot load is being switched to a common EEPROM), then a two-step process is needed. The I2C_BOOT_CNTRL register should be written once with the BOOT_UNLK field set to 1, then written a second time with the correct information. The lower 2 bits of the BOOT_ADDR field are only allowed to change if the BOOT_UNLK field was a 1 before the register load.

7.8.7 EEPROM Data Format

Table 18 shows the EEPROM data format for boot loading. The first 8 bytes of the EEPROM contain the number of registers to be loaded during the boot procedure. This count is the 16-bit value in EEPROM location 0 (MSB) and location 1 (LSB). The I2C Interface is limited to 255 register loads in 1-byte address mode, and limited to 8 KB-1 register loads in 2-byte address mode. The remaining 6 bytes (memory locations 2 through 7) must be set to 0xFF or the register count validity check will fail and the boot load will be aborted.

When 1-byte address mode is selected, any number of registers greater than 255 (0x00FF) aborts the boot load from the EEPROM.

When 2-byte address mode is selected, any number of registers greater than 8191 (8 KB-1 = 0x1FFF) aborts the boot load from the EEPROM.

The register load data consists of 8-byte fields aligned to 8-byte peripheral address boundaries. The first 4 bytes are the internal register address and the second 4 bytes are the register data. Note that the address and data are ordered from MSB to LSB within increasing peripheral byte addresses.

Table 18: Format for Boot Loadable EEPROM

PerAdr	PerAdr+0	PerAdr+1	PerAdr+2	PerAdr+3
0x0	RegCnt(MSB)	RegCnt(LSB)	0xFF	0xFF
0x4	0xFF	0xFF	0xFF	0xFF
0x8	RegAdr(MSB)	RegAdr	RegAdr	RegAdr(LSB)
0xC	RegData(MSB)	RegData	RegData	RegData(LSB)
0x10	RegAdr(MSB)	RegAdr	RegAdr	RegAdr(LSB)
0x14	RegData(MSB)	RegData	RegData	RegData(LSB)
...	...	...	...	...

As an example, the following shows an EEPROM configured to load two registers and then complete – first the "I2C Master Configuration Register" at internal address 0x1D108, loaded with data value 0x0102_0304; then the "I2C Master Transmit Data Register" at internal address 0x1D114, loaded with data value 0x0506_0708.

Table 19: Sample EEPROM Loading Two Registers

PerAdr	PerAdr+0	PerAdr+1	PerAdr+2	PerAdr+3	Description
0x0	0x00	0x02	0xFF	0xFF	RegCnt = 2, must have 0xFFFF at end
0x4	0xFF	0xFF	0xFF	0xFF	Must be 0xFFFF_FFFF
0x8	0x00	0x01	0xD1	0x08	RegAdr = 0x1D108 I2C_MST_CFG
0xC	0x01	0x02	0x03	0x04	RegData = 0x0102_0304
0x10	0x00	0x01	0xD1	0x14	RegAdr = 0x1D114 I2C_MST_TDATA
0x14	0x05	0x06	0x07	0x08	RegData = 0x0506_0708
>= 0x18	xx	xx	xx	xx	Unused by Boot

As a second example, the following shows an EEPROM configured to first load the I2C_MST_CFG register then chain to address 0x80 in the same EEPROM and load the I2C_MST_TDATA register. Note that the chain requires loading the I2C_BOOT_CNTRL register. The new peripheral address is 0x80 >> 3 = 0x10, because the 3 LSBs must be zero and are not part of the PADDR field.

Table 20: Sample EEPROM With Chaining

PerAdr	PerAdr+0	PerAdr+1	PerAdr+2	PerAdr+3	Description
0x0	0x00	0x02	0xFF	0xFF	RegCnt = 2, must have 0xFFFF at end
0x4	0xFF	0xFF	0xFF	0xFF	Must be 0xFFFF_FFFF
0x8	0x00	0x01	0xD1	0x08	RegAdr = 0x1D108 I2C_MST_CFG
0xC	0x01	0x02	0x03	0x04	RegData = 0x0102_0304
0x10	0x00	0x01	0xD1	0x40	RegAdr = 0x1D140 I2C_BOOT_CNTRL
0x14	0x80	0x50	0x00	0x10	RegData = 0x8050_0010 CHAIN = 1 BOOT_ADDR = 1010000 PADDR = 0x10
0x18 - 0x7F	xx	xx	xx	xx	Unused by Boot
0x80	0x00	0x01	0xFF	0xFF	RegCnt = 1, must have 0xFFFF at end
0x84	0xFF	0xFF	0xFF	0xFF	Must be 0xFFFF_FFFF
0x88	0x00	0x01	0xD1	0x14	RegAdr = 0x1D114 I2C_MST_TDATA
0x8C	0x05	0x06	0x07	0x08	RegData = 0x0506_0708
>= 0x90	xx	xx	xx	xx	Unused by Boot

7.8.8 I2C Boot Time

The time required to perform an I2C boot depends on the following:

- The number of registers that require configuration
- The number of devices contending for EEPROM or I2C bus access
- The number of chaining operations
- The clocking speeds of the master devices

Because many of these parameters are outside the control of the Tsi574, the boot time cannot be predicted with complete accuracy.

If there are no other devices contending for bus access, a 1-byte peripheral address is used, no boot acceleration techniques are used, and no retries are necessary for device detect, then boot time can be estimated as follows:

Boot_Time =

50 *u*s idle detect time +

(9 * ClkPer) EEPROM reset time +

(102 * (RegisterCount + 1) * ClkPer) register load time +

(1 * ClkPer) STOP time

Where:

ClkPer = clock period (resets to 10 *u*s for a 100 kHz clock)

RegisterCount is the sum of number of registers from the Register Count fields in the EEPROM (only one count field unless chaining is involved).

If a 2-byte peripheral address is used, then the “102” constant increases to “111”. The 102 constant comes from the sum of Start + (9-bit boot address) + (9-bit peripheral address) + Restart + (9-bit boot address) + (9-bit data byte * 8 bytes per register = 72 bits) = 101 clocks, but the Start and Restart take an extra 1/2 clock each, so an extra clock cycle is consumed.

For example, if 255 registers are read the boot time is:

Boot_Time = 50*u*s + (90*u*s EEPROM reset) + (10*u*s * 102 * 256 register load) + 10*u*s Stop

Boot_Time = 261,270 *u*s = slightly over 1/4 second

7.8.9 Accelerating Boot Load

If boot load time is a design concern, the following techniques may accelerate the boot load sequence:

1. If the EEPROM supports reading of a large block of data sequentially, change PAGE_MODE in “I2C Boot Control Register” as the first register load. Depending on the page size, this reduces the number of times the boot load re-addresses the device and resets the peripheral address. At the limit, if the “infinite” setting was chosen and the device did not wrap on any page boundaries, the 102 constant in the boot time formula in “I2C Boot Time” would be reduced to 72 cycles per register, with only one address phase initially or per chain operation.
2. If the EEPROM supports reading at higher than 100-kHz clock speeds, the timing parameters can be changed during boot load. The success of this depends on the bus properties because the Tsi574 does not contain the Schmitt Triggers or slope controlled outputs needed to guarantee conformance to the 400-kHz high-speed mode. However, it is possible that many configurations will be interoperable at higher speeds (for information on changing timing parameters, see “Bus Timing”). Timing parameters are reloaded upon a chain operation, so the technique is to program the timing parameters for the higher speed, set up the digital filters if required, and then invoke a chain operation using the same EEPROM but the next peripheral address. Everything from the chain onwards will be mastered at the higher speed.

7.9 Error Handling

The Tsi574 handles a number of I2C errors and reports them with status bits, as summarized in Table 21.

Table 21: I2C Error Handling

Error Cause	Access Type	Tsi574 Response	Interrupt Status Bit (Events) [a]
Master Access Errors			
Master arbitration timeout expired. Tsi574 could not successfully arbitrate for the I2C bus; Arbitration lost during device addressing phase.	Master read or write initiated using I2C_MST_CNTRL register	The I2C transaction is aborted.	MA_ATMO
Tsi574 determined that it lost arbitration for the I2C bus after the device addressing phase	Read or Write	The I2C transaction is aborted.	MA_COL
No device ACK'd the slave address, or target device NACK'd a peripheral address or write data byte.	Any read or write access during slave address phase or peripheral address phase, or any write access during the data phase.	Access aborted, STOP generated. The I2C_ACC_STAT register indicates where transaction was on error.	MA_NACK
Timeout expired (I2C_SCLK Low, Byte or Transaction). Target device was too slow, or some device was interfering with the I2C_SCLK signal.	Any transfer to or from the Tsi574	Access aborted. The I2C_ACC_STAT register indicates where transaction was on error. For Byte or Transaction, master issues STOP at first legal opportunity. For I2C_SCLK Low, bus is hung, software must recover.	MA_TMO (MSCLTO, MBTTO or MTRTO)
Slave Access Errors			
Peripheral Address selects reserved external address space	Read operation	Peripheral Address byte is acknowledged, 0x00 is returned as data.	SA_OK
	Write operation	Peripheral Address byte is acknowledged. Write data is ignored.	
Peripheral Address selects a defined register, but data burst continues into reserved address	Read operation	0x00 is returned as data.	SA_OK
	Write operation	Write data is ignored	

Table 21: I2C Error Handling *(Continued)*

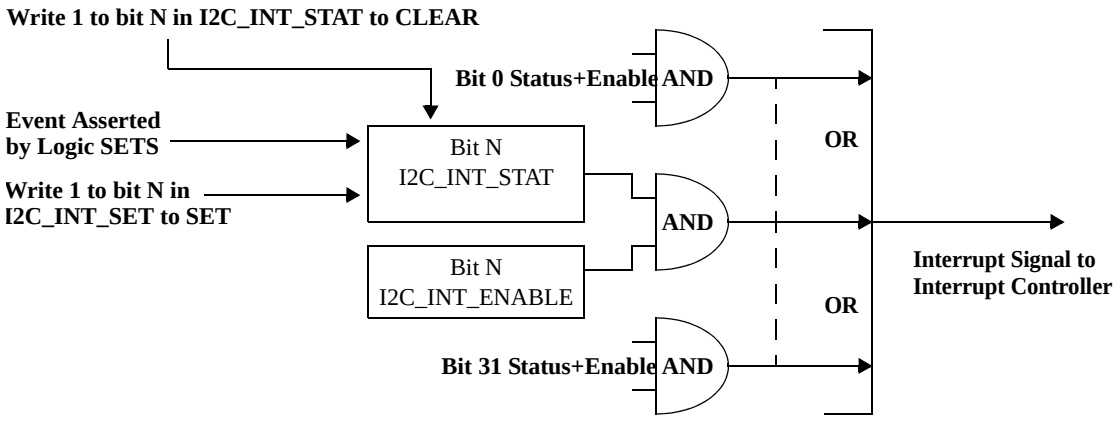
Error Cause	Access Type	Tsi574 Response	Interrupt Status Bit (Events) [a]
Programmed register address accesses a non-existent internal register block	Read operation	Read returns 0x00	SA_OK
	Write operation	Write data discarded	SA_OK
Internal register access when disabled	External master read to the EXI2C_REG_RDATA register, or write to the EXI2C_REG_WDATA register	Operation completes, returns existing RDATA or updates WDATA, but no internal register access generated.	SA_OK No SDW/SDR
Timeout expired (I2C_SCLK Low, Byte or Transaction). Target device was too slow, or some device was interfering with the I2C_SCLK signal.	Any transfer to or from the Tsi574	Slave releases I2C_SD and I2C_SCLK, goes into wait state.	SA_FAIL (SSCLTO, SBTTO or STRTO)
Protocol violation (collision detected)	Read data or Ack/Nack, when slave puts a 1 on the I2C_SD signal and another device holds the signal to 0.	Slave releases I2C_SD and I2C_SCLK, goes into wait state.	SA_FAIL (SCOL)
Register Initialization Loader Errors			
Failed to find EEPROM	Initialization read	Read operation retried up to 6 times before aborting. If not Ack'ed by the 6th try, status bits set	BL_FAIL (BLNOD)
Size field specifies more than 255 registers to load in 1-byte addressing mode, or 8 KB-1 registers in 2-byte addressing mode.	Initialization read	Initialization load aborted	BL_FAIL (BLSZ)
Register address selects non-existent register.	Register initialization write	Data discarded	None
Failed to arbitrate for I2C bus during boot load, boot load timer expired.	Initialization read	Initialization load aborted	BL_FAIL (BLTO)
Protocol error during boot load, including bytes 2-7 of a register count not containing 0xFF.	Initialization read	Initialization load aborted	BL_FAIL (BLERR)

a. To determine the setting of the interrupt status bits, see "I2C Interrupt Status Register".

7.10 Interrupt Handling

I2C interrupts are generated as shown in Figure 37. An I2C event detected by the I2C Interface sets a bit in the "I2C Interrupt Status Register" to a 1 to assert the interrupt. This bit is then anded with the corresponding bit in the "I2C Interrupt Enable Register" to determine if that interrupt is enabled. Any enabled interrupt status bit asserts the interrupt output signal to the Interrupt Controller. This signal stays asserted until all enabled bits in the interrupt status register are cleared.

Figure 37: I2C Interrupt Generation



The interrupt status bits are cleared by a write-one-to-clear operation to the Interrupt Status Register, provided the interrupt status register has first been read. For test purposes, bits in the Interrupt Status Register can also be set by a write-one-to-set operation to the "I2C Interrupt Set Register".

A bit that is set in the Interrupt Status Register is cleared by a write-1-to-clear operation only after the register has first been read, and then providing another event that would result the interrupt condition has not occurred since the read of the register (see "Events versus Interrupts".

7.11 Events versus Interrupts

Interrupts are generated by I2C events. Figure 38 shows the design of the event and interrupt logic. A single interrupt status bit may be derived from one or more events. The event registers provide control over the individual events that in turn produce the interrupt status. In the diagram, the shaded boxes represent virtual registers. These registers behave correctly when read or written, but can be constructed from combinational logic as opposed to flip-flops. Whether a register is virtual or not is inconsequential to their behavior from a software perspective. The distinction is shown only for exactness.

A new event is set in the "I2C New Event Register" when an event is asserted in the logic, or when a 1 is written to the register (or to the related interrupt bit in the "I2C Interrupt Set Register"). New events are ored with the I2C_SNAP_EVENT register to create the virtual I2C_EVENT register. A snapshot operation occurs when the "I2C Interrupt Status Register" is read. As a result of the snapshot, the new event register is "copied" to the snapshot register by oring the new events into the current snapshot state, then clearing the new event register. Each event is anded with the corresponding enable bit in the "I2C Enable Event Register", and then ored with any other enabled events that are related to a single interrupt status bit. The combined event state becomes the interrupt status bit in the Interrupt Status Register, and is then anded with the corresponding enable in the "I2C Interrupt Enable Register". All the enabled interrupt status bits are then ored together to become the single interrupt signal to the Interrupt Controller.

The new event and snapshot registers separate events that occurred prior to a read of the interrupt status register from those that occur during or after the read. When a 1 is written to the interrupt status register to clear an interrupt, all related events that are enabled are cleared in the snapshot register. Since events are copied to the snapshot register only when the interrupt status register is read, the read must be completed first for the write 1 to clear to have effect. If no new events have occurred, this write-1-to-clear de-asserts the interrupt status. If a new event has occurred, the event remains set in the new event register, so the interrupt status remains set.

For control purposes, software can read and clear the snapshot event bits directly, allowing individual events to be cleared while leaving any new events intact. Software can also select to read or clear events using the new event register. Reading the event register shows the "or" of the new and snapshot, and thus shows whether an event is asserting. Writing a 1 to an event bit clears both the snapshot and new events register bits, thus clearing out the event entirely, unless that event happens to be asserting again on the same cycle the clear is completed, thus setting it again.

As long as all event enables are set (the reset state), then the behavior is logical (see "Interrupt Handling").

Figure 38: I2C Event and Interrupt Logic

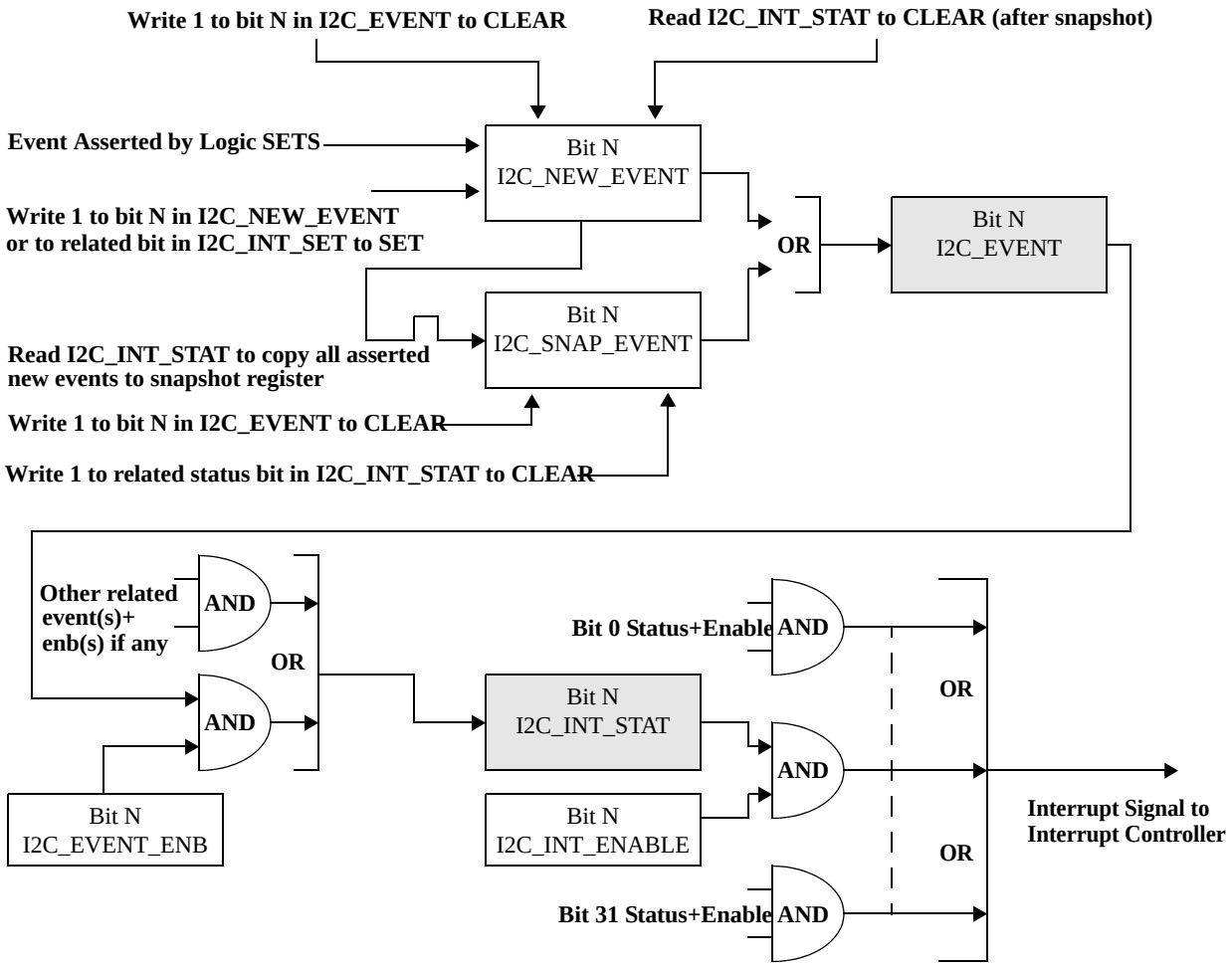


Table 22 shows the mapping of interrupts in the "I2C Interrupt Status Register" to the events in the "I2C Event and Event Snapshot Registers". Any asserted and enabled event sets the corresponding interrupt status, and clearing an asserted interrupt status bit clears all the related and enabled events.

Table 22: I2C Interrupt to Events Mapping

Interrupt Status Bit	Events Related to Interrupt
OMB_EMPTY (Outgoing Mailbox Empty)	OMBR (Outgoing Mailbox Read Event)
IMB_FULL (Incoming Mailbox Full)	IMBW (Incoming Mailbox Write Event)
BL_FAIL (Boot Load Fail)	BLTO (Boot Load Timeout Error) BLERR (Boot Load Error Event) BLSZ (Boot Load Size Error Event) BLNOD (Boot Load No Device Event)

Table 22: I2C Interrupt to Events Mapping *(Continued)*

Interrupt Status Bit	Events Related to Interrupt
BL_OK (Boot Load OK)	BLOK (Boot Load OK Event)
SA_FAIL (Slave Access Failed)	SCOL (Slave Collision Detect Event) STRTO (Slave Transaction Timeout Event) SBTTO (Slave Byte Timeout Event) SSCLTO (Slave I2C_SCLK Low Timeout Event)
SA_WRITE (Slave Access Write)	SDW (Slave Internal Register Write Done Event)
SA_READ (Slave Access Read)	SDR (Slave Internal Register Read Done Event)
SA_OK (Slave Access OK)	SD (Slave Transaction Done Event)
MA_DIAG (Master Diagnostic Event)	DTIMER (Diagnostic Timer Expired Event) DHIST (Diagnostic History Filling Event) DCMDD (Diagnostic Command Done Event)
MA_COL (Master Collusion)	MCOL (Master Collision Detect Event)
MA_TMO (Master Timeout)	MTRTO (Master Transaction Timeout Event) MBTTO (Master Byte Timeout Event) MSCLTO (Master I2C_SCLK Low Timeout Event)
MA_NACK (Master NACK)	MNACK (Master NACK Received Event)
MA_ATMO (Master Arbitration Timeout)	MARBTO (Master Arbitration Timeout Event)
MA_OK (Master Transaction OK)	MTD (Master Transaction Done Event)

7.12 Timeouts

The I2C Interface supports a number of timeout periods to detect a set of error conditions related to I2C operation. These timeouts, and the registers that configure them, include the following:

- I2C_SCLK low timeout (see "I2C_SCLK Low and Arbitration Timeout Register") – This timeout detects a situation where a device on the bus is stuck holding the clock low. Because the clock is stuck low, no progress can be made. If enabled, this timeout expiring will set either the SSCLTO or MSCLTO events and result in a SA_FAIL or MA_TMO interrupt status being updated in the I2C_INT_STAT register (depending on whether a master or slave operation was in progress). An optional interrupt can be sent to the Interrupt Controller if SA_FAIL or MA_TMO is enabled in the "I2C Interrupt Enable Register". This is an extreme failure. With I2C_SCLK held low, no Stop condition can be generated. Any operation is aborted, both I2C_SCLK and I2C_SD are released, and both master and slave revert to their monitor-for-bus-idle phase. It is up to software to decide how to handle this error. Because any operation was aborted without correct termination (no Stop), it is possible that the external device is left in an invalid state.

- Arbitration timeout (see "I2C_SCLK Low and Arbitration Timeout Register") – This timeout applies only to master transactions initiated by setting the START bit in the "I2C Master Control Register". Its purpose is to limit the length of time the master controller tries to gain ownership of the bus. The arbitration timer is disabled once the <Start><Slave Address><Read/Write> are successfully transmitted without detecting another master attempting a different transaction. If the Tsi574 I2C master subsequently loses ownership of the bus after this phase of the transaction, the transaction is aborted. If the Tsi574 I2C master detects another master corrupting the <Start><Slave Address><Read/Write> bits it has transmitted, the Tsi574 I2C master reverts to waiting for bus idle then tries again. The arbitration timeout continues to run in this case. If the arbitration timer expires before ownership is gained and the master is waiting for bus idle, then it aborts the operation and sets the MARBTO event, which causes a MA_ATMO interrupt status to be updated in the "I2C Interrupt Status Register". An optional interrupt can also be sent to the Interrupt Controller if the MA_ATMO is enabled in the "I2C Interrupt Enable Register".

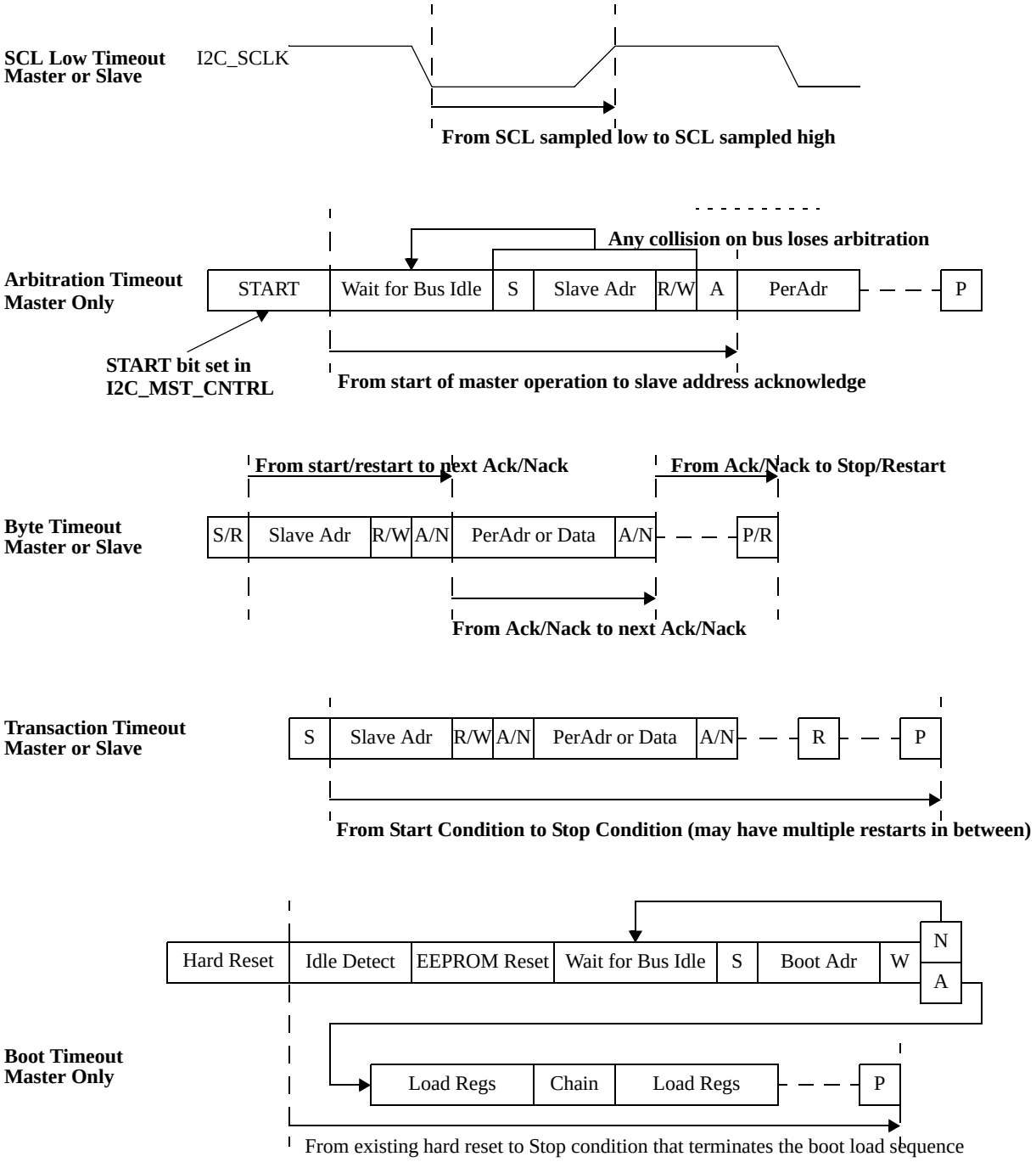
  If the Tsi574 I2C master was in the midst of transmitting the <Slave Address> when the timeout expires, it allows the <Slave Address> to complete. If an ACK or NACK is successfully received, the master continues as if the timeout had not expired. If another I2C master collides with <Slave Address>, the timeout immediately takes effect following the <Slave Address> bit where the collision took place.

- Byte timeout (see "I2C Byte/Transaction Timeout Register") – This timeout is disabled on reset. It detects a situation where one or more devices are stretching the clock enough to slow the transfer speed on the bus beyond some limit. This timeout is available primarily to detect a violation of the SMBus TLOW:MEXT time. The response to this timeout expiring depends on the phase of the transfer and whether it is detected by the master or slave interface. For a master transaction, the master continues to generate clocks until the next bit time where it would have control of the bus; that is, writing data or generating an Ack/Nack in response to a read byte. At that time, the master generates a Stop condition, aborts the operation and sets the MBTTO event, which causes an MA_TMO interrupt status to get updated in the "I2C Interrupt Status Register". An optional interrupt can also be sent to the Interrupt Controller if the MA_TMO bit is enabled in the "I2C Interrupt Enable Register". For a slave transaction, the slave waits for the start of the next bit time, releases the I2C_SD and I2C_SCLK signals and sets the SBTTO event, which causes an SA_FAIL interrupt status to get updated in the I2C_INT_STAT register. An optional interrupt can be sent to the Interrupt Controller if the SA_FAIL bit is enabled in the "I2C Interrupt Enable Register". The slave then reverts to looking for the next Start/Restart/Stop.

- Transaction timeout (see "I2C Byte/Transaction Timeout Register") – This timeout is disabled on reset. It detects a situation where a master is keeping the bus for an extended period of time, as measured from the Start to Stop condition. This timeout is available primarily to detect a violation of the SMBus TLOW:SEXT time. The response to this timeout expiring is identical to a Byte timeout, with the exception that the events are MTRTO or STRTO for the master or slave respectively.
- Boot timeout (see "I2C Boot and Diagnostic Timer") – This timeout detects a situation where the boot load sequence has not completed in a reasonable time. This could occur if the EEPROM was improperly programmed with an infinite chaining loop, the bus ownership is held by some other device, or some other anomalous situation resulting in any of the time-outs above. If the boot timeout expires before the normal end of the boot load sequence, the master interface reads until the next data byte and drives a Stop condition on the bus. It then sets the BLTO event, which causes a BL_FAIL interrupt status to get updated in the "I2C Interrupt Status Register". An optional interrupt can also be sent to the Interrupt Controller if the BL_FAIL bit is enabled in the "I2C Interrupt Enable Register". The boot_complete signal is asserted when the boot load timeout expires. If the boot timeout is not desired, then the EEPROM programming should immediately write the I2C_BOOT_DIAG_TIMER.COUNT to 0 to disable the timeout.

Figure 39 shows the relationship of the I2C time-outs to I2C operations.

Figure 39: I2C Timeout Periods



7.13 Bus Timing

Figure 40 shows the relationship of the bus timing parameters to the generation of the I2C_SCLK and I2C_SD signals on the I2C bus. These parameters are configured in the following registers:

- "I2C Start Condition Setup/Hold Timing Register"
- "I2C Stop/Idle Timing Register"
- "I2C_SD Setup and Hold Timing Register"
- "I2C Stop/Idle Timing Register"
- "I2C_SCLK High and Low Timing Register"
- "I2C_SCLK Minimum High and Low Timing Register"

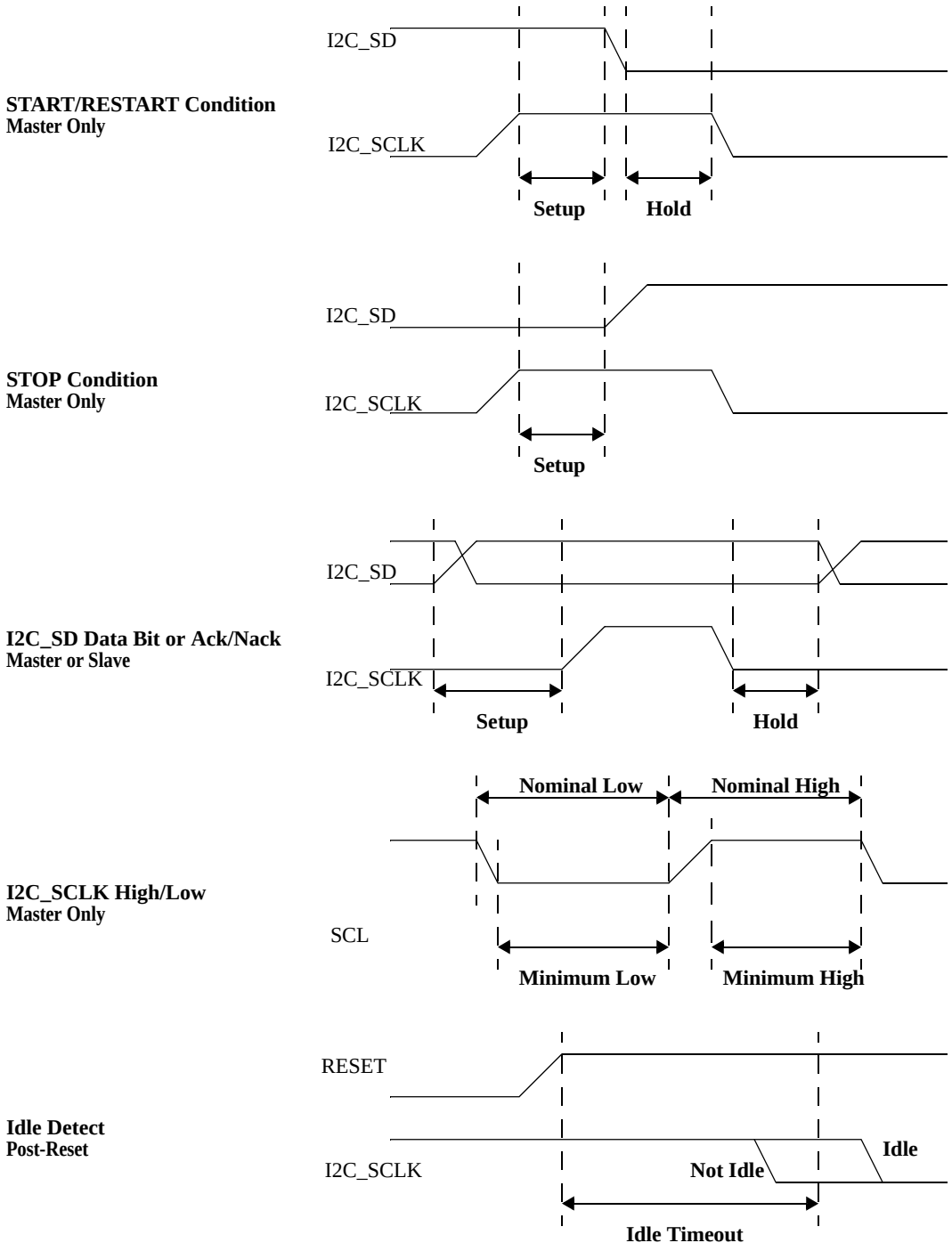
The bus timing resets to 100-kHz operation. By reprogramming these registers, other bus speeds can be configured. Speeds above 100 kHz are not guaranteed to conform to the *I2C Specification* because of the absence of Schmitt triggers on the input of the I2C_SD and I2C_SCLK signals, and the absence of slope controlled outputs for the I2C_SD and I2C_SCLK signals. It is up to the board or system designer to decide on the applicability of operation at speeds above 100 kHz.

Bus timing does not normally change during a transaction, even if these registers are changed. The timing registers are sampled at certain times to prevent this from occurring. The following are the times when timing adjustments take effect:

- On hard reset (times are reset to 100 kHz)
- When reset using the "I2C Reset Register"
- At the start of a master transaction through the "I2C Master Control Register", when the START condition is generated
- Upon a chain operation during boot load

Timing parameters are discussed further in the following sections.

Figure 40: I2C Bus Timing Diagrams



7.13.1 Start/Restart Condition Setup and Hold

The Start/Restart Condition is generated by a master. As shown in Figure 40, the Start Setup time defines the minimum period both the I2C_SD and I2C_SCLK signals must be seen high (1) before the I2C_SD signal is pulled low (0) to trigger the Start. The I2C_SD signal must also have fulfilled the I2C_SD Setup time prior to the rising edge of I2C_SCLK. Once the I2C_SD signal is seen low (0), the Start Hold time is the minimum period the I2C_SCLK signal must continue to remain high (1) before it is pulled low (0). These parameters are used by the Tsi574 as a master when generating the Start condition. These times may be violated by an external master or slave pulling the I2C_SD or I2C_SCLK signals low before the setup/hold periods are expired, which may result in an arbitration loss or collision.

7.13.2 Stop Condition Setup

The Stop Condition is generated by a master. As shown in Figure 40, the Stop Setup time defines the minimum period the I2C_SD must be seen low (0) and the I2C_SCLK signal must be seen high (1) before the I2C_SD signal is released high (1) to trigger the Start. The I2C_SD signal must also have fulfilled the I2C_SD Setup time prior to the rising edge of I2C_SCLK. There is no separate Stop Hold parameter, as the only valid condition following a Stop would be a Start; therefore, the Start Setup fulfills the same use as a Stop Hold or Stop-to-Start buffer time. This parameter is used by the Tsi574 as a master when generating the Stop condition. If the I2C_SCLK signal was prematurely pulled low (0) by an external master or slave, this would be seen as a collision event.

7.13.3 I2C_SD Setup and Hold

Either a master or a slave can be in control of the I2C_SD signal, depending on the phase of the data transfer protocol. As shown in Figure 40, the I2C_SD Setup time defines the minimum period the I2C_SD signal must set to the desired state while I2C_SCLK is low (0) before the I2C_SCLK signal is release high (1) to generate the high period of the clock. The I2C_SD Hold time defines the minimum period the I2C_SD signal is left unchanged after the falling edge of I2C_SCLK (I2C_SCLK seen low). The I2C_SD hold time may be violated by another device pulling I2C_SD low, but this is not an error, as it normally indicates another device with a different design.

The I2C_SD setup time is not as defined in the *I2C Specification*. The setup time parameter encompasses both the maximum rise/fall time of the I2C_SD signal plus the output hold time and must be set accordingly. There is no feedback check that the I2C_SD signal goes to the desired state, as this could result in I2C_SCLK being held low erroneously. If another device is also controlling I2C_SD, the likely result is an arbitration loss or collision.

7.13.4 I2C_SCLK Nominal and Minimum Periods

These parameters are used by the Tsi574 as a master to generate the I2C_SCLK clock. The master must obey the minimum times to conform to the *I2C Specification*, and must also attempt to regulate the overall I2C_SCLK frequency to a defined period. From Figure 40, it can be seen that the logic measures the minimum periods high/low from the detected rising/falling edges of the I2C_SCLK signal to the point where I2C_SCLK is driven low or released high to generate the opposing edge. In conjunction, a separate nominal period timer measures from driven low to released high, and released high to driven low. Both timers must expire if unaffected by external devices. If another device pulls the I2C_SCLK signal low prematurely in the high period, the high period timers are expired and the lower period timers restart for the low period, so the actual low period may be stretched by the nominal timer. If another device holds the I2C_SCLK signal low longer in the low period than the nominal low period, the high period nominal timer will likely expire early and the minimum high period timer will control the high period when the clock is finally released.

7.13.5 Idle Detect Period

This is a master-only parameter that is used in two cases. First, upon exit from reset it is unknown if another master is active. The Idle Detect timeout determines if the I2C_SCLK signal remains high long enough (roughly 50 microseconds) that it is unlikely another master is active. If I2C_SCLK is seen low during this period, it is assumed another master is active, and the master enters the Wait for Bus Idle phase. If the idle detect period expires without I2C_SCLK seen low, then it is assumed the bus is idle and the master is free to generate a Start Condition if needed.

Second, during the Wait for Bus Idle phase, it is possible that an external master that has claimed the bus ceases activity without issuing a STOP condition. When a master operation is started but the bus is currently seen busy, the idle detect timer monitors the I2C_SCLK and I2C_SD signals. If the I2C_SCLK and I2C_SD signals both remain high longer than the idle detect period, the bus is then assumed idle even though a STOP had not been seen, and the master logic will attempt the requested transaction.

8. Performance

This chapter is a detailed description of the packet switching performance characteristics of the Tsi574. It consists of the following general topics:

8.1 Overview

Performance for packet switching is characterized by three measurements: throughput, latency, and latency variation.

Performance is specified for error free transmission and reception of packets. Performance is specified for end-to-end transfers through the Tsi574. No performance specifications are made for the different stages of transfers through the Tsi574.

Performance is specified for a single switch. Performance for larger systems can be computed from this data.

8.1.1 Throughput

Throughput for packets is a measurement of the amount of packet data that can be transferred in a given amount of time. It can be presented in different forms:

- Percentage of a link's bandwidth (for example, 56% of a 1x @ 3.125 Gbaud)
- Number of packets of a given size per unit time (for example, 3000 44 byte packets every second)
- Bit transfer rate (for example, 300 Mbits/s)

Throughput measurements include only successfully transferred packets. Measured throughput does not include control symbols, retried packets, or other non-packet data transmitted/received on a link (/K/ and /R/ characters).

8.1.2 Latency

Latency is the amount of time between packet reception and packet transmission. However, the specific time at which packet reception and packet transmission are deemed to have started must still be defined. Throughout this document, latency is measured as the time interval between the first bit of the Start-of-Packet arriving at the ingress of the Tsi574 and that same bit leaving the device.

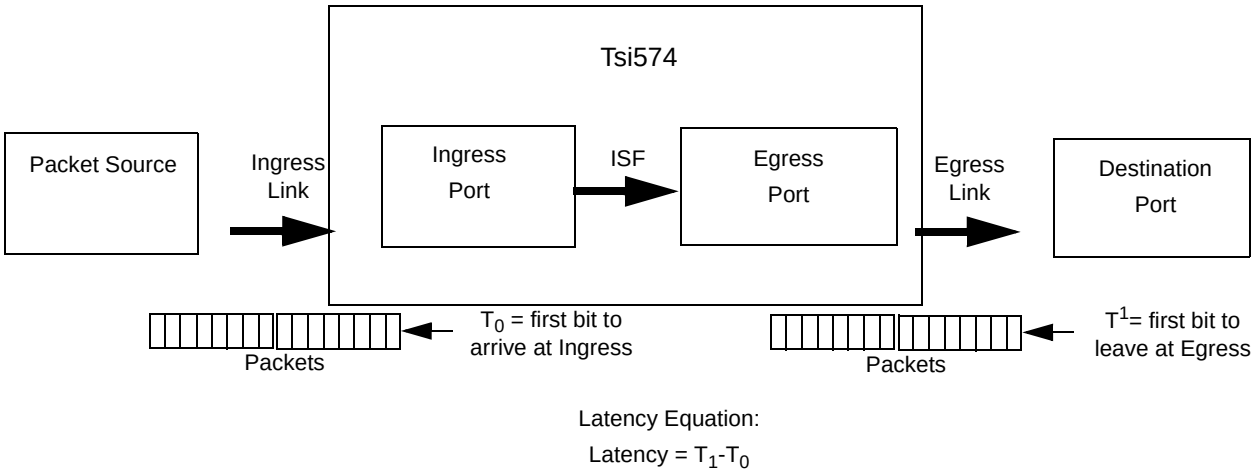
Figure 41 illustrates the path a packet flows through a Tsi574. For Tsi574 latency performance, packet reception time begins with the time the first bit of a packet is seen on the input pins. Packet transmission begins when the first bit of a packet has been transmitted on the output pins.

As part of the resolution of resource contention, higher priority packets can be allowed to pass packets of lower priority. Latencies should therefore decrease as the priority of a packet increases.

A specific time for packet latency can only be specified when there are no conditions that create resource contention between packets. For example, if a single stream of packets passing from one ingress port to a single egress port is the only traffic handled by the Tsi574, it is possible to specify the latency for the packets in this stream.

A complex traffic pattern is defined to be one which has resource contention. Complex traffic patterns make specifying the exact latency figure that each packet experiences difficult, because the amount of contention that a packet experiences can vary widely. As such, these scenarios are not covered in this manual.

Figure 41: Latency Illustration



In the Tsi574, packets experience packet latency variations caused by the asynchronous ability of the device. Packets can experience an extra one or two clock cycles of delay over the minimum latency when crossing from one clock domain to another clock domain. These factors should be taken into account when creating a system timing budget (refer to Table 24 on page 192).

8.2 Performance Monitoring

The main purpose of the performance monitoring functionality is to observe the data traffic on the RapidIO interface. The RapidIO traffic can come from different sources (different processing endpoints) and can cause data congestion in one of the destination interfaces. This congestion can have a negative impact on overall system performance. Performance monitoring can be used to identify and help prevent situations that negatively impact system performance.

Performance monitoring decisions can be made by system software in real-time. The system software can be programmed to routinely read the performance monitoring registers, analyze the traffic flow patterns, and re-route accordingly to avoid congestion.

Each Serial RapidIO port in the device has a copy of the performance monitoring registers.

Table 23 lists the statistic parameters that are available from the Outbound and Inbound registers, as part of each port's performance monitoring capabilities.

Table 23: Performance Monitoring Parameters

Parameters	Registers	Description
Number of 32-bit words	• "RapidIO Port x Performance Statistics Counter 0 and 1 Control Register" on page 324 • "RapidIO Port x Performance Statistics Counter 0 Register" on page 336	Any of the performance statistics counter registers can be configured to count the number of 32 bit words sent or received by a RapidIO link.
Number of transactions	• "RapidIO Port x Performance Statistics Counter 0 and 1 Control Register" on page 324 • "RapidIO Port x Performance Statistics Counter 0 Register" on page 336	Any of the performance statistics counter registers can be configured to count the number of packets sent or received by a RapidIO link.
Number of packets for each priority (0, 1, 2, and 3)	• "RapidIO Port x Performance Statistics Counter 0 and 1 Control Register" on page 324 • "RapidIO Port x Performance Statistics Counter 0 Register" on page 336	Any of the performance statistics counter registers can be configured to count the number of packets sent or received by a RapidIO link with a particular priority.
Queue depth for inbound and outbound buffer	• "RapidIO Port x Transmitter Output Queue Depth Threshold Register" on page 342 • "RapidIO Port x Transmitter Output Queue Congestion Status Register" on page 344 • "RapidIO Port x Transmitter Output Queue Congestion Period Register" on page 346 • "RapidIO Port x Receiver Input Queue Depth Threshold Register" on page 347 • "RapidIO Port x Receiver Input Queue Congestion Status Register" on page 349 • "RapidIO Port x Receiver Input Queue Congestion Period Register" on page 351	Performance Statistics for monitoring the congestion situation in both Tx and Rx directions are supported.

The following sub-sections describe the use of these parameters for monitoring the performance of the serial RapidIO ports in Tsi574.

8.2.1 Traffic Efficiency

To characterize the efficiency of system traffic, the following parameters are used:

1. Packet rate (number of packets / time)
   - Packet rate is calculated using the number of packets computed from a counter register configured to count the number of packets.
2. Average packet size (number of 32-bit words / number of packets)
   - Average packet size is calculated using a counter configured to count the number of 32-bit words (call it COUNTER A), and a counter configured to count the number of packets (call it COUNTER B). The average packet size is COUNTER A divided by the value in COUNTER B.
3. Utilization ((packet rate * packet size) / max capacity)
   - Utilization is calculated using parameter 1 and parameter 2, above.

These values are derived from the number of packets and the number of 32-bit words on each interface. The calculations of the packet rate, packet size, and utilization are completed externally.

8.2.2 Throughput

The count of packets per priority in each interface can be a very important parameter when debugging RapidIO systems. This information can also be valuable when used by system software to dynamically re-route traffic around congested interfaces. The following parameters are used to monitor the throughput on each RapidIO interface:

- Number of packets for each priority level (0, 1, 2 and 3)
  - Each performance counter register (for example, “RapidIO Port x Performance Statistics Counter 0 Register” on page 336) can be configured to count the number of packets selected based on priority.

Retried packets are not counted.

8.2.3 Bottleneck Detection

Monitoring the queue depth of the inbound and outbound modules can detect bottleneck traffic in the RapidIO interfaces. It can also be used to determine the period of time that packets of a given priority and below cannot be accepted. Both the inbound and outbound directions have the ability to program a queue depth watermark. The number of times that the queue depth watermark is exceeded is counted. As well, the amount of time that the queue depth watermark is exceeded is also counted to a programmable degree of accuracy. A port-write and/or an interrupt can be asserted if the queue depth watermark value exceeds a programmable number.

The registers in the outbound direction that contain the values and counters described above are:

- “RapidIO Port x Transmitter Output Queue Depth Threshold Register” on page 342
- “RapidIO Port x Transmitter Output Queue Congestion Status Register” on page 344
- “RapidIO Port x Transmitter Output Queue Congestion Period Register” on page 346

The registers in the inbound direction are:

- “RapidIO Port x Receiver Input Queue Depth Threshold Register” on page 347
- “RapidIO Port x Receiver Input Queue Congestion Status Register” on page 349
- “RapidIO Port x Receiver Input Queue Congestion Period Register” on page 351

8.2.4 Congestion Detection

A packet is reordered when it cannot make forward progress through the internal switching fabric. Packet reordering can be a sign of congestion in a RapidIO interface. A count of the number of times packets are reordered in each interface is stored in the “RapidIO Port x Reordering Counter Register” on page 352. After the value in a programmable threshold is reached, an interrupt is triggered.

For example, if the traffic is time-critical control data, a very low threshold is programmed so that it is not congested for long. The interrupt handler is invoked. The system host can then take action to help ease the congestion.

8.2.5 Resetting Performance Registers

The Inbound and Outbound performance registers are both read and writable. These registers are cleared after every read and saturate at the maximum counter values.

8.3 Configuring the Tsi574 for Performance Measurements

Performance measurements for complex traffic patterns can be specified for two different configurations of performance settings.

The first configuration is for lightly loaded systems, where the likelihood of resource contention is low. This is known as the ‘fair share’ performance configuration.

The second configuration is for congested systems which optimize the throughput and latency of the highest priority packets at the expense of lower priority packets. This is known as the ‘high priority’ performance configuration.

It is expected that configurations different from the two described will have performance figures between the two values specified.

There are many controls in the Tsi574 that allow a system designer to optimize their system interconnect performance. These controls can be categorized as clock speeds, ISF arbitration settings, RapidIO packet scheduling and buffer management settings.

8.3.1 Clock Speeds

Port speeds directly affect throughput, latency and latency variation. Generally, the slower the port, the lower the throughput, the higher the average latency and the greater the spread between minimum and maximum latency.

- For ports operating in 1x mode, performance measurements are specified for operation at 3.125 Gbaud.
- For ports operating in 4x mode, performance measurements are specified for operation at 3.125 Gbaud per lane.
- All performance measurements assume that the ISF is operating at its maximum frequency of 156.25 MHz.

Performance changes linearly with port and ISF speed.

8.3.2 Tsi574 ISF Arbitration Settings

The ISF has three possible settings for its egress arbitration: First Come, First Served, Strict Priority 1, and Strict Priority 2.

The First Come, First Served algorithm is used in the *fair share* performance configuration.

For *high priority* systems which require the absolute lowest possible latency for the highest priority packets, and are willing to tolerate the additional latency and latency variation induced on the lower priority packets, the ISF Strict Priority 2 arbitration algorithm should be used. ISF Strict Priority 1 can be used in verification to ensure that ISF Strict Priority 2 does deliver optimal performance.

8.3.3 Tsi574 RapidIO Transmission Scheduler Settings

The First Come, First Served packet scheduling algorithm is used in fair share systems. In this algorithm, the oldest packet is transmitted. If this packet is retried, then the oldest, highest priority packet is transmitted. The oldest packet is the transmitted again. This leads to increased latency and decreased throughput for higher priority packets, since their forward progress is dependent upon the speed with which a lower priority packet can be retried.

8.3.4 Tsi574 RapidIO Buffer Watermark Selection Settings

Buffer watermarks are used to restrict the transmission of lower priority packets, to the advantage of higher priority packets. Watermark settings directly affect throughput and indirectly latency and latency variation. For more information on watermarks, refer to “Egress Watermark” on page 90.

The default watermark settings should be used for the *fair share* configuration for both RapidIO ingress buffer management and RapidIO egress buffer management.

For ‘high priority’ configurations, watermark settings should be used which deliver maximal throughput for the highest priority packets. For ingress and egress ports, a maximum of 6 priority 2 packets can be accepted, a maximum of 4 priority 1 packets can be accepted, and a maximum of 2 priority 0 packets are accepted.

8.4 Port-to-Port Performance Characteristics

The most intuitively obvious performance measurements of the Tsi574’s use port-to-port traffic models to characterize the maximum possible throughput and minimum latency performance of the Tsi574.

In this case, all traffic is of uniform size and the same priority. Due to the simple type of traffic, the throughput and latency performance numbers do not change with the priority of the packets.

8.4.1 Port-to-Port Packet Latency Performance

Table 24 on page 192 shows the 4x and 1x mode latency numbers under no congestion with default ISF arbitration and watermark settings. The numbers are based on the same ingress and egress port widths and baud rates. The minimum latency is the minimum time an ingress packet takes to appear at the egress. Due to the multi-clock domain system, the device operates in, the minimum latency can vary by one 312.5 MHz clock period and one reference clock (S_CLK) period.

Cut-through mode is assumed.

Table 24: 4x/1x Latency Numbers Under No Congestion

Reference Clock	Ingress and Egress Port Width	Ingress and Egress Baud Rate	Minimum Latency (ns)[a]
156.25MHz	4x mode	3.125	112
		2.5	128.8
		1.25	212.8
	1x mode	3.125	131.2
		2.5	152.8
		1.25	260.8

a. Due to the asynchronous ability of the clock frequencies within the device, the latency numbers can vary as much by one 312.5 MHz clock period and one reference clock (S_CLK) period.

The Tsi574 is designed to allow high priority traffic to bypass low priority traffic in periods of contention, as allowed in the RapidIO protocol specification.

8.4.2 Packet Throughput Performance

Packet throughput varies from the packet type (for example, NWRITE packets do not require a logical layer response), availability of resources within the device, ability for source and destination of traffic to generate or receive packets, retries of packet, and actual data rates.

A *bubble* is a control symbol inserted by an egress port to maintain the baud rate of the port. The appearance of a bubble indicates that the egress port is under-utilized.

A bubble packet is not the Idle Sequence inserted to maintain link synchronization, as required by the *RapidIO Interconnect Specification (Revision 1.3)*.

8.4.2.1 One Port-to-One Port Throughput Performance

Under a non-congested port-to-port packet traffic situation, when the ingress and egress have the same line rate (1.25, 2.5, or 3.125 Gbaud), the ingress and egress always maintain the line rates. This means there is no retry of packets at the ingress ports and no bubbles will appear in the egress packet stream except for the idle sequence insertion every 5000 code-groups as required by the *RapidIO Interconnect Specification (Revision 1.3)*. This is true for any payload size and different priorities.

When the ingress line rate exceeds that of the egress port, a retry occurs at the ingress port when the buffer is filled to the capacity permitted by the priority of the packets. The egress port still maintains its maximum packet rate with no bubble. This is true for any payload size and priorities.

8.4.2.2 Many Ports-to-One Port Throughput Performance

Under a non-congested, many ports-to-one port packet traffic scenario, when all of the total ingress line rates are the same as the egress line rate (for example, four 1x mode, 3.125 Gbaud ingress ports all going to one 4x mode, 3.125 Gbaud egress port), the ingress port and egress port will always maintain line rates. This means there will be no retry of packets at the ingress and no bubble occurring in the egress packet streams except for the idle sequence insertion every 5000 code-groups required by the *RapidIO Interconnect Specification (Revision 1.3)*. This is true for any payload size and different priorities. The arbitration scheme within the device allocates sufficient bandwidth for each ingress port.

When the total of the ingress line rates exceed that of egress port, retries occur at one or more of the ingress ports if the packet density exceeds the capacity of the egress port. The egress port still maintains its maximum packet rate with no bubble. This is true for any payload size and priorities.

8.4.2.3 One Port-to-Many Port Throughput Performance

Under a non-congested one port-to-many ports packet traffic scenario, when the ingress line rate is the same as the total egress line rates (for example one 4x mode, 3.125 Gbaud ingress port splitting to four 1x mode, 3.125 Gbaud egress port), the ingress and egress always maintain line rates. This means there is no retry of packets in ingress and no bubble-packet in the egress packet streams except for the idle sequence insertion every 5000 code-groups required by the RapidIO specification. This is true for any payload size and different priorities. The arbitration scheme within the device divides the traffic according to the egress port bandwidths.

When the ingress line rate exceeds that of the total of the egress ports, retries occur at the ingress port when the packet density exceeds the buffer capacity. The egress ports still maintains their maximum packet rates with no bubble. This is true for any payload size and priorities.

8.4.3 Multicast Performance

8.4.3.1 Multicast Latency

Since multicast involves more than one egress port and each egress port can have independent traffic conditions, a multicast packet can appear at the destination egress ports at different times. A minimum multicast latency is defined as the shortest time from the arrival of the first bit of a packet at an ingress port that will be multicast, to the appearance of the first bit of the multicast packet at an egress port under no resource contention.

Table 25: 4x/1x Multicast Latency Numbers Under No Congestion

Reference Clock	Ingress and Egress Port Width	Ingress and Egress Baud Rate (Gbaud)	Minimum Latency (ns)[a]
156.25MHz	4x	3.125	163.2
		2.5	178.4
		1.25	254.4
	1x	3.125	188.8
		2.5	210.4
		1.25	318.4

a. Due to the asynchronous ability of the clock frequencies within the device, the latency numbers can vary as much as 6.4 ns.

8.4.3.2 Multicast Throughput

The maximum input payload bandwidth of the multicast engine is 10 Gbit/s. This corresponds to a line rate of 4x mode, 3.125 Gbaud at the ingress port. The maximum input bandwidth of the multicast engine can be sourced from one ingress port or multiple ingress ports.

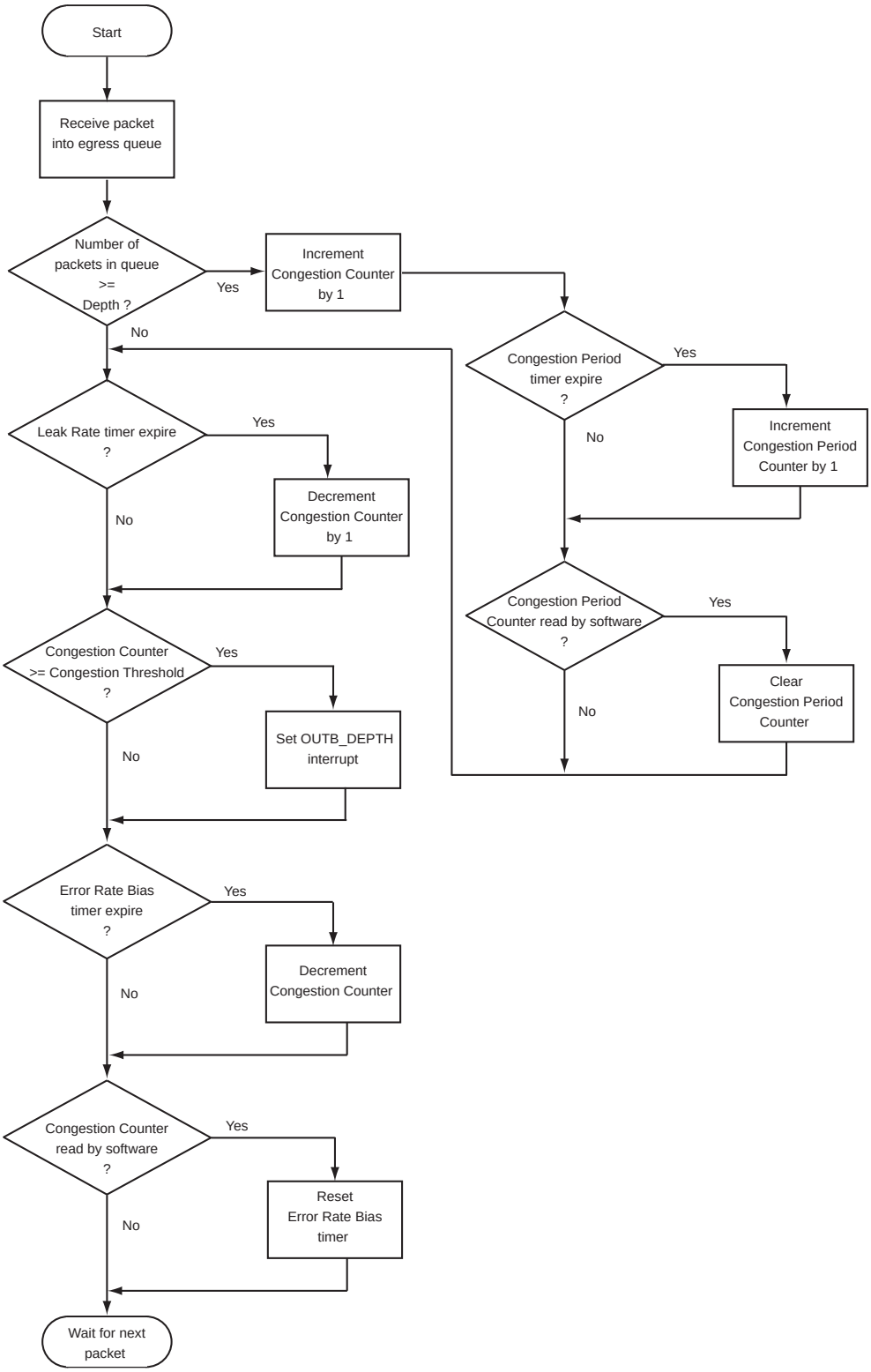
When there is no congestion, and when all destination egress ports have a line rate of 4x mode (3.125 Gbaud) the egress port always maintains the line rate. There is no bubble-packet in the egress packet streams except for the idle sequence insertion every 5000 code-groups required by the *RapidIO Interconnect Specification (Revision 1.3)*. There is also no retry at the ingress port because the ingress aggregation is handled by the multicast arbitration. This is true for any payload size and different priorities.

When any of the egress port has a line rate lower than the input bandwidth of the multicast engine, retries occur at the ingress port. In this situation, the egress port maintains its line rate.For example, when an egress port is set to 4x mode, 2.5 Gbaud while the multicast engine is receiving a single or aggregated input data at maximum 10Gbit/s, retries happen at the ingress port(s). However, the egress port still maintains its line rate with no bubble inserted in that packet stream.

8.5 Congestion Detection and Management

The congestion detection and management functionality enables the system management host to monitor the system through a series of registers. The system host can monitor the ingress and egress queue levels and the frequency at which the queues are above the threshold defined by the DEPTH parameter. The behavior and effects of the various tick timers and counters is described in the flow chart shown in Figure 42.

Figure 42: Congestion and Detection Flowchart



8.5.1 Congestion Registers

The Tsi574 contains registers in every port that can be used for the detection and monitoring of ingress and egress queue levels. The registers and their descriptions are as follows:

- "RapidIO Port x Transmitter Output Queue Depth Threshold Register" on page 342: This register contains tick timer values for the Congestion Period timer and the Leak Rate timer. This register also contains the Depth threshold which is compared against the number of packets in present in the egress queue.
  - Congestion Period (CONG_PERIOD): This value sets the tick interval for the Congestion Period Timer. At the timer expiry, the Congestion Counter is incremented by 1 if the counter value is greater than zero. The Congestion Period Counter indicates the number of tick intervals that the number of packets in the egress queue has exceeded the preset value in the DEPTH field of the register between the current and previous register reads of the Congestion Period Counter.
  - DEPTH: This field sets the threshold which exceeded by the number of packets in the egress buffer cause the Congestion Counter to be incremented.
  - Leak Rate (LEAK_RT): This register field is the count for the Leak Rate tick timer. At every tick of the Leak Rate tick timer, the Congestion Counter is decremented.
- "RapidIO Port x Transmitter Output Queue Congestion Period Register" on page 346: This register contains only one field, CONG_PERIOD_CTR. This is the Congestion Period Counter. This counter is incremented at every tick whose period is set by the CONG_PERIOD field when the Congestion Counter is greater than zero. This register is keeps a running count, and is only cleared with a register read.
- "RapidIO Port x Transmitter Output Queue Congestion Status Register" on page 344: This register contains two fields, the Congestion Counter and the Congestion Threshold.
  - Congestion Counter (CONG_CTR): This field keeps a running count of the number of times that the number of packets in the egress queue exceed the DEPTH field setting. The test to increment this counter is performed when a packet has arrived in the egress buffer in its entirety. The test is not synchronous to any clocks.
    The Congestion Counter is decremented on Leak Rate timer ticks and on Error Rate Bias timer ticks, and is cleared by writing 1 to the OUTB_DEPTH bit in the RIO Port x Interrupt Status Register.
  - Congestion Threshold (CONG_THRESH): This is the threshold value which if exceeded by the Congestion Counter, sets the OUTB_DEPTH interrupt bit in the RIO Port x Interrupt Status Register.
- "RapidIO Port x Reordering Counter Register" on page 352: This register contains two fields which are used to track when a packet in an ingress buffer is unable to make forward progress to an egress buffer because the egress buffer cannot accept any more packets of the priority of the received packet. Packet reordering takes place when a packet of higher priority is blocked by a packet of lower priority.
  - Counter (CTR): This is a counter which is incremented each time the switch fabric selects a packet in the ingress queue that is not at the head of the queue for transmission to an egress buffer.

— Threshold (THRESH): This field sets the threshold of how many times the ISF can re-order packets before the INB_RDR bit in the RIO Port x Interrupt Status register is set.

The receiver versions of the registers contain the same fields as the registers related to the transmitters however the receiver registers pertain to the ingress buffer queue status.

- “RapidIO Port x Receiver Input Queue Depth Threshold Register” on page 347
- “RapidIO Port x Receiver Input Queue Congestion Period Register” on page 351
- “RapidIO Port x Receiver Input Queue Congestion Status Register” on page 349

8.5.1.1 Interrupts

Each port's congestion management register set has two status interrupts, one for the ingress queue depth status and one for the egress queue status. Both interrupts are located in the “RapidIO Port x Interrupt Status Register” on page 318 for the port.

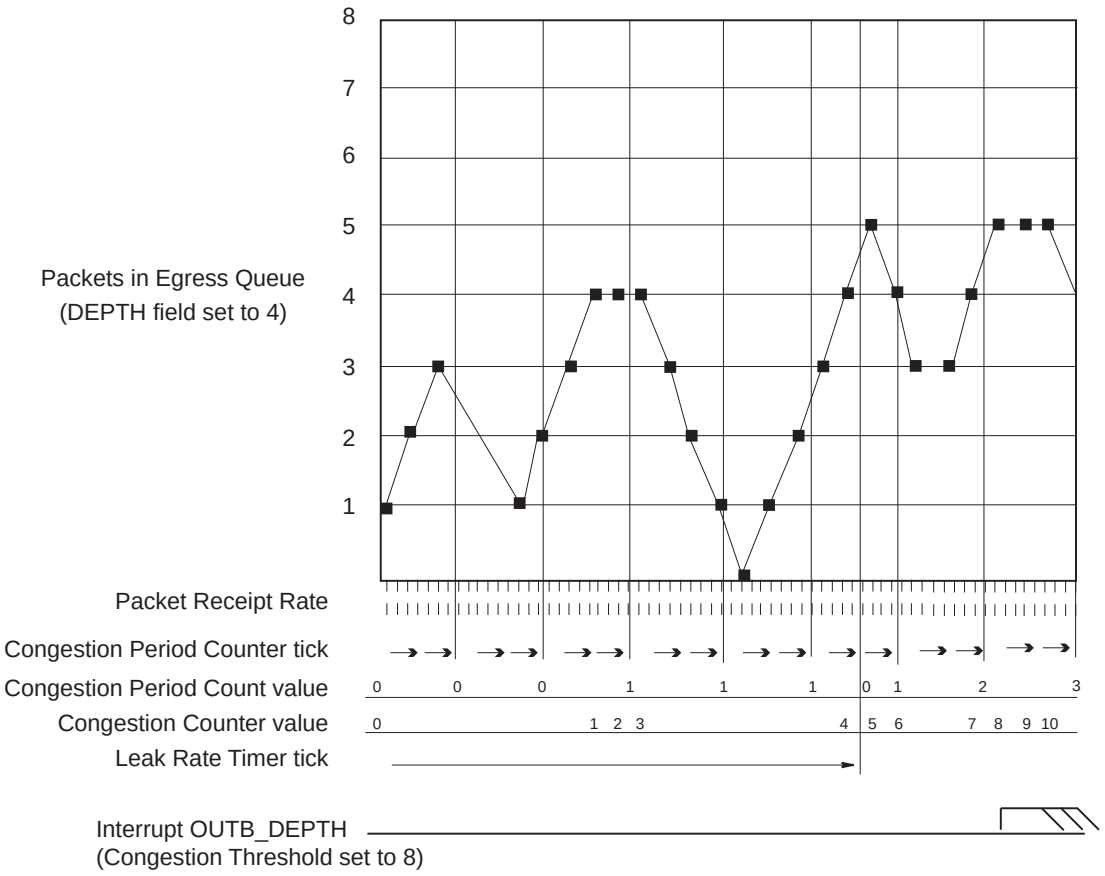
The Congestion Counter and Congestion Period Counter fields must be polled in order to determine the current congestion trend if waiting for the interrupts to occur is insufficient warning that the buffers in the switch have become congested. Also, by tracking the Congestion Counter value and the Congestion Period Counter value, it is possible to determine the traffic trend for the buffer.

8.5.1.2 Example of Congestion Register Behavior

In this example, the number of packets in the egress queue are monitored. The bursting ability of the queue is shown with the frequent increase and decrease of packets. When the queue appears flat it indicates either condition where the number of packets entering the queue is the same as the number of packets leaving the queue appear in this manner or that queue was stalled during that period.

Figure 43 on page 198 is an example of what the congestion registers may contain at various times during port operation.

Figure 43: Congestion Example



The Packet Receipt Rate in the chart indicates how quickly packets can enter and leave the queue. Essentially, this represents the packet line rate. A system with smaller packets increases this receipt rate while a system with predominantly large packages decreases this rate.

The chart also shows the Congestion Period Counter tick and the Leak Rate Timer tick. These ticks occur based on their programmed values (see “RapidIO Port x Transmitter Output Queue Depth Threshold Register” on page 342). In this example, the DEPTH bit is programmed to four in the “RapidIO Port x Transmitter Output Queue Depth Threshold Register” on page 342.

Also shown is the Congestion Period Count value (see “RapidIO Port x Transmitter Output Queue Depth Threshold Register” on page 342) and the Congestion Counter value (see “RapidIO Port x Transmitter Output Queue Congestion Status Register” on page 344).

In this example, the value of the Congestion Threshold has been set to eight (see “RapidIO Port x Transmitter Output Queue Congestion Status Register” on page 344). When the Congestion Counter equals the value in the Congestion Threshold, the OUTB_DEPTH interrupt is asserted (see “RapidIO Port x Interrupt Status Register” on page 318). When the OUTB_DEPTH interrupt is asserted, a port-write packet can be generated which causes an in-band notification of the condition that can be routed to any host in the system.

9. JTAG Interface

This chapter describes the main features of the JTAG interface. It includes the following information:

9.1 Overview

The JTAG interface in Tsi574 is fully compliant with IEEE 1149.6 B*oundary Scan Testing of Advanced Digital Networks* as well as IEEE 1149.1 *Standard Test Access Port and Boundary Scan Architecture* standards. There are five standard pins associated with the interface (TMS, TCK, TDI, TDO and TRST_b) which allow full control of the internal TAP (Test Access Port) controller.

The JTAG Interface has the following features:

- Contains a 5-pin Test Access Port (TAP) controller, with support for the following registers:
  - Instruction register (IR)
  - Boundary scan register
  - Bypass register
  - Device ID register
  - User test data register (DR)
- IDT-specific pin (BCE) which allows full 1149.6 compliant boundary-scan tests. This pin should be held high on the board.
- Supports debug access of Tsi574's configuration registers
- Supports the following instruction opcodes:
  - Sample/Preload
  - Extest
  - EXTEST_PULSE *(1149.6)*
  - EXTEST_TRAIN *(1149.6)*
  - Bypass
  - Hi-Z
  - IDCODE
  - Clamp
  - User data select

9.2 JTAG Device Identification Number

The JTAG device ID number for the Tsi574 is 0x80571167.

9.3 JTAG Register Access Details

The Tsi574 has the capability to read and write registers through the JTAG interface.

Prior to using the IEEE Register Access Command feature, the part must be reset by driving TRST_b low.

The JTAG Interface has the ability to access registers in order to debug issues that can affect RapidIO register accesses. Register access through the JTAG Interface can also be used in normal mode to do extensive read and write accesses on the performance registers without slowing down the normal traffic in the device or during initialization.

A user defined command is used to enable the read and write capabilities of the interface. The command is in the IEEE 1149.1 Instruction Register (IR) in the Tsi574.

- IEEE Register Access Command (IRAC)

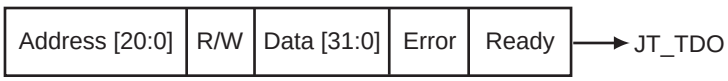
There must be IEEE 1149.1 capability on the board to use the IEEE 1149.1 register access feature.

9.3.1 Format

The format used to access the registers is shown in Figure 44 and Figure 45. The address shown in the figure is the RapidIO offset.

Figure 44: Register Access From JTAG - Serial Data In

Figure 45: Register Access From JTAG - Serial Data Out



9.3.2 Write Access to Registers from the JTAG Interface

The following steps are required in order to write to a register through the JTAG interface:

1. Move to the Tap controller “Shift-IR” state and program the instruction register with IRAC instruction by writing into Instruction Register bits [2:0] with 3’b101.
2. Move to the “Shift-DR” state and shift the data[31:0], R/W = 1 and the address[20:0] serially in the TDI pin. To prevent corruption of un-used bits, the full DR bits have to be written. The following values must be written:
   - DR[119:99] = ADDR[20:0]
   - DR[98] = R/W
   - DR[97:66] = DATA[31:0]
   - DR[65:64] = 0b0
   - DR[63:0] = 0b0.
3. Move to the “Run-test idle” state and loop in this state for a minimum of 20 TCK cycles.
4. Move to the “shift-DR” state again and shift-in 120 zero bits to DR[119:0], while at the same time verify the Ready and Error bits that are being shifted-out as the first two bits shifted-out.
5. Go back to step two to perform another write.

9.3.3 Read Access to Registers from the JTAG Interface

The following steps are required in order to read a register through the JTAG interface:

1. Move to the Tap controller “Shift-IR” state and program the instruction register with IRAC instruction.
   — This step is optional if the instruction register is already programmed during the write cycle.
2. Move to the “Shift-DR” state and shift the R/W = 0 and the address[20:0] serially in the TDI pin. To prevent corruption of un-used bits, the full DR bits have to be written. The following values must be written:
   - DR[119:99] = ADDR[20:0]
   - DR[98] = R/W
   - DR[97:66] = DATA[31:0]
   - DR[65:64] = 0b0
   - DR[63:0] = 0b0
3. Move to the “Run-test idle” state and loop in this state for a minimum of 20 TCK cycles.
4. Move to the “Shift-DR” state and shift in 120 bits of 0. The first two bits in data shifted out are the Error and Ready bits. The next 32 bits are data. The rest of the shifted out data can be discarded.
5. Verify that the Error bit is at logic low and the Ready bit is at logic high.
6. Go back to step two to perform another read.

10. Clocks, Resets and Power-up Options

This chapter describes the clock and reset of the Tsi574. It includes the following information:

10.1 Clocks

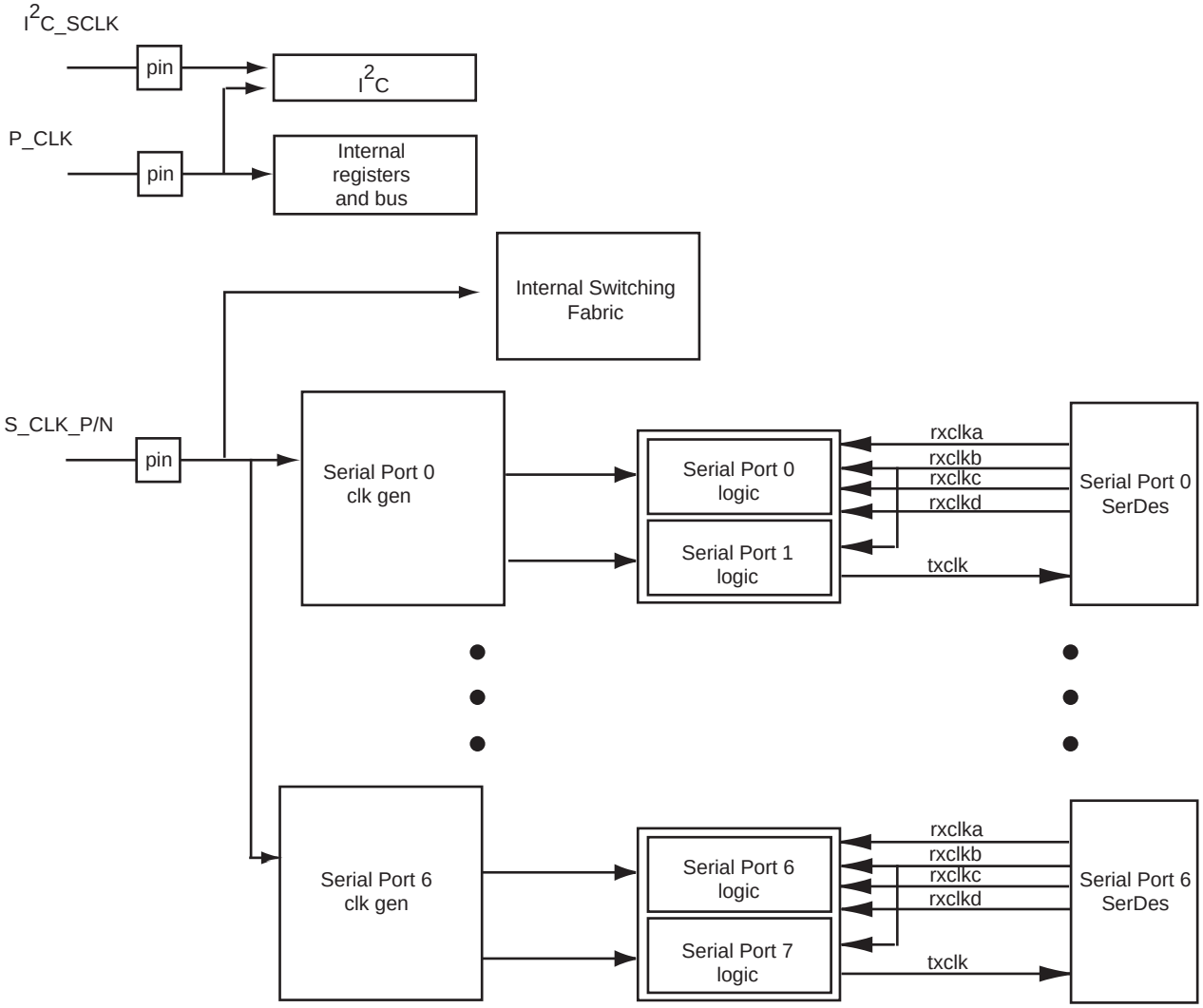
The Tsi574 has three input clocks (S_CLK_p/n, P_CLK and I2C_SCLK) that are used to produce the Tsi574’s internal clock domains.

In addition to these reference clocks, each RapidIO ingress port contains independent receive clock domains, one for each lane. The receive clock is extracted from the 8B/10B encoding on each lane.

10.1.1 Clocking Architecture

The Tsi574 device relies on the reference clock (S_CLK_p/n) to generate most clocks inside device. S_CLK_p/n is fed into each SerDes. On the receive side, each SerDes recovers clocks from the data stream. In 4x mode, four different synchronous clocks are extracted (RXCLKA..D). In 1x mode, either one (RXCLKA) or two (RXCLKA..B) clocks are recovered. On the transmit side, the clock (TX_CLK) is derived from the SerDes. An extra clock (SYS_CLK) is also sourced from the SerDes to the MAC. The S_CLK_p/n signal is also an input to the Switch Fabric and internal registers.

Figure 46: Tsi574 Clocking Architecture



10.1.2 SerDes Clocks

All SerDes in Tsi574 use the same external reference clock (S_CLK_p/n). Depending on the pin or register setup, the SerDes generates the appropriate clocks to serialize/deserialize the data as well as the clocks for the internal logic. On the Receive side, each lane of the SerDes recovers their own clocks. These clocks can be powered down by register controls ("SRIO MAC x SerDes Configuration Global" on page 364).

10.1.3 Reference clocks

The two reference clocks are described in Table 26.

For information on configuring the clock rate of RapidIO ports, refer to "Clocking" on page 69.

Table 26: Tsi574 Input Reference Clocks

Clock Input Pin	Type	Frequency[a]	Clock Domains
S_CLK_p/n	Differential	156.25MHz[b]	Serial Transmit Domain (Maximum 156.25 MHz) Internal Switching Fabric (ISF) Domain
P_CLK[c]	Single-ended	100 MHz	Internal Register Domain I^2C Domain

a. For more electrical characteristics of the clocks, please refer to the *Tsi574 Hardware Manual*.

b. For more information about operation at alternative S_CLK frequencies, refer to "Line Rate Support" on page 475.

c. For more information on programming additional frequencies for the P_CLK, refer to "P_CLK Programming" on page 479.

10.1.4 Clock Domains

The Tsi574 contains a number of clock domains that are generated from the two input reference clocks. These domains are detailed in Table 27. For more information about special line rate support see "Clocking" on page 475.

Table 27: Tsi574 Clock Domains

Clock Domain	Clock Source	Frequency[a]	Description
Internal Register Domain (P_CLK domain)	P_CLK[b]	100 MHz	This clock domain includes all internal registers within each of the internal blocks, as well as the bus that performs the register accesses. The domain uses the input P_CLK directly.
Internal Switching Fabric (ISF) Domain	S_CLK_p/n	156.25 MHz	This clock domain includes the switching matrix of the ISF, the Multicast blocks, and the portion of each block that communicates with the ISF. The domain uses the S_CLK_P/N.
Serial Transmit Domain	S_CLK_p/n	156.25 MHz	This clock domain is used to clock all the Serial RapidIO transmit ports. The S_CLK_P/N input is used directly to clock the transmit logic. This clock is used to generate the high-speed clock that is used to output the serial data on output pins SP{0..}_T{A..D}_P/N. The maximum data rate for this domain is 3.125 Gb/s per lane.
I^2C Domain	P_CLK divided by 1024	97.7 kHz	This clock domain is responsible for driving the I^2C output clock pin I2C_SCLK. This clock domain is generated by dividing the P_CLK input by a programmable value. The majority of the I^2C logic runs in the Internal Register Domain (P_CLK domain).

a. For more electrical characteristics of the clocks, please refer to the *Tsi574 Hardware Manual*.

b. For more information on programming additional frequencies for the P_CLK, refer to "P_CLK Programming" on page 479.

10.1.5 Clock Gating

When a RapidIO port is powered down using the PWDN_X1/X4 bits in the "SRIO MAC x Digital Loopback and Clock Selection Register" on page 369, the clock to that RapidIO port is gated to prevent the port from consuming power.

10.2 Resets

Internal logic is responsible for automatically sequencing the removal of reset in all internal blocks to meet their requirements; no additional software programming is required.

10.2.1 Device Reset

The Tsi574 can be reset the following ways:

1. Assertion of the HARD_RST_b input pin
2. Receiving four RapidIO Link Request/Reset-Device Control Symbols in a row (without any other intervening packets or control symbols, except status control symbols) from any of the RapidIO ports.
   - The SELF_RST bit in the "RapidIO Port x Mode CSR" on page 302 must be set to 1 (self-reset)

In both cases, when the Tsi574 is reset it goes through its full reset and power-up sequence. All state machines and the configuration registers are reset to the original power on states.

Lookup tables are left in an undefined state after reset. It is recommended that lookup tables be completely initialized after a reset to ensure deterministic operation.

10.2.1.1 I^2C Boot

When all blocks have been taken out of reset, the I^2C Interface is responsible for performing automatic reads from an externally attached EEPROM device in order to load the initial configuration of the device. For more details refer to "I^2C Interface" on page 139.

External I^2C devices are not reset by the Tsi574, so the I^2C bus could be left in an undefined state if the Tsi574 is reset during initial configuration. It is recommended that resets of the Tsi574 occur at a rate that ensures that register loading from I^2C device has completed before another reset is issued.

10.2.1.2 HARD_RST_b Reset

The HARD_RST_b signal is an external system reset input signal and causes a general reset of the Tsi574; all blocks are reset within the device. HARD_RST_b is an active low signal with asynchronous assertion and de-assertion. The internal reset synchronizers are responsible for assuring that reset is de-asserted internally at the correct time for each of the clock domains.

When HARD_RST_b is asserted, SW_RST_b is de-asserted. SW_RST_b remains de-asserted after HARD_RST_b is released.

Timing of HARD_RST_b

The Tsi574 requires the following timing for the HARD_RST_b signal:

- HARD_RST_b must be asserted for a minimum of 1 millisecond (ms).
- Tsi574 comes out of reset within 1 ms after HARD_RST_b is de-asserted after assertion.
  - A boot load from the I2C EEPROM will delay when the device becomes ready for RapidIO traffic on the ports. See "Boot Load Sequence" on page 164 for more information on boot loading and the time implications.

10.2.1.3 RapidIO Reset Requests

The Tsi574 responds to Reset Request Control Symbols as defined by the *RapidIO Interconnect Specification (Revision 1.3)*.

Self Reset

When a reset request occurs, the Tsi574's response is controlled by the "RapidIO Port x Mode CSR" on page 302. By default, the Tsi574 resets itself. A self-reset occurs when four link-request/reset-device control symbols are received in a row (without any other intervening packets or control symbols, except status control symbols) and the SELF_RST field in the "RapidIO Port x Mode CSR" on page 302 is set. When a self-reset is performed, it is not necessary to drive the HARD_RST_b input signal. The SW_RST_b signal remains asserted for the duration of the self reset, which is at least four P_CLK clock cycles.

If the SELF_RST field is not set an interrupt signal is asserted (if RCS_INT_EN is also set in the "RapidIO Port x Mode CSR" on page 302).

System Control of Resets

Self-reset of the Tsi574 may not be sufficient in systems which require other components to be reset at the same time as the Tsi574. The Tsi574 supports system control of resets in two ways. First, the Tsi574 can assert the INT_b interrupt so that a local processor can trigger a reset through the Tsi574's HARD_RST_b pin. For this design to work, reset interrupts must be enabled in the "RapidIO Port x Mode CSR" on page 302 and the "Global Interrupt Enable Register" on page 379.

Secondly, if interrupts are not suitable for reset functionality in a system, it is possible to use the SW_RST_b output pin. When the Tsi574 has received a reset request, the SW_RST_b output pin is asserted until the reset request status is cleared in the port that received it. The SW_RST_b output pin can be used as an input to a reset controller to trigger the start of a system reset. If self-reset is not enabled, SW_RST_b remains asserted until the device is reset through the input reset pin HARD_RST_b or until the interrupt bit is cleared for the port that received the reset message. If self-reset is enabled, SW_RST_b is asserted for the duration of the reset, which is at least four P_CLK clock cycles.

When the Tsi574 is in reset, the INT_b pin is not asserted.

SW_RST_b is the only method to determine that a reset request has been received and should be handled as an interrupt. Port-writes cannot be sent for notification of reset request reception.

10.2.2 Per-Port Reset

In order to reset an individual RapidIO port it must be powered down and back up again using the following procedure:

1. Power down the port using the procedure in "Port Power Down".
2. Wait 50 microseconds to guarantee that all packets are flushed.
3. Power the port back up.
4. Reconfigure the registers that were impacted by the power down (see "Default Configurations on Power Down").

After a port has been reset (powered down and back up), the following is applicable:

- The port's configuration registers revert to their default values.
- The port's SerDes related registers and LUTs must be re-configured.
- The port's register values are not loaded from I2C on a port reset.

The SOFT_RST_X1 and SOFT_RST_X4 bits in the "SRIO MAC x Digital Loopback and Clock Selection Register" only reset the MAC – they do not reset the SerDes.

10.2.3 Generating a RapidIO Reset Request to a Peer Device

The following steps can be used by software to reset a peer device:

1. Determine which RapidIO port is connected to the peer to be reset.
2. Alter the LUT contents to ensure that no packets are being routed to the link partner that is to be reset.
3. Lockout the port using the PORT_LOCKOUT field in the "RapidIO Serial Port x Control CSR" on page 273. This ensures that any traffic received from the peer device is dropped, and any traffic still in flight to the peer device is dropped.
4. Use the "RapidIO Serial Port x Link Maintenance Request CSR" on page 265 to transmit four reset control symbols in a row.
5. Write 0 to the OUTBOUND field of the"RapidIO Serial Port x Local ackID Status CSR" on page 268.

10.2.4 JTAG Reset

The JTAG TAP controller's reset is independent of the Tsi574 functional resets. For boundary scan operation, the TAP controller can be reset with either the external pin TRST_b or by holding the pin TMS asserted for more than five TCK cycles.

To ensure predictable operation of the Tsi574, for power-up reset, HARD_RST_b and TRST_b must be asserted prior to operation. After power-up, the TAP controller can be reset at any time and this does not affect the Tsi574 operation.

Normal functional reset is still required to reset the device's internal registers. Reset of the Tsi574 does not reset the TAP.

The TAP controller must be reset on power-up, whether or not it is going to be used, to ensure predictable operation of the Tsi574.

10.3 Power-up Options

The Tsi574 has the following types of power-up option pins: default port speed (SP_IO_SPEED[1:0]), port power-up and power-down (SPn_PWRDN), mode selection (SPn_MODESEL), lane swap (SP_RX_SWAP and SP_TX_SWAP), and I^2C pins (I2c_DISABLE, I2C_MA, I2C_SA[1:0], I2C_SEL).

The power-up option pins must be stable for 10 P_CLK cycles after HARD_RST_b is de-asserted.

10.3.1 Power-up Option Signals

Power up options are latched at reset for initializing the Tsi574. The power-up option pins are listed in Table 28. All power-up option pins have to remain stable for 10 P_CLK cycles after HW_RST_b is de-asserted in order to be sampled correctly. These signals are ignored after reset and software is able to override the settings.

The power-up signals do have internal PU or PD, however external resistors are recommended on these signals.

Table 28: Power-Up Options Signals

Pin Name	Description
SP{n}_MODESEL	Selects the serial port operating mode for ports n and n+1 0 = Port n operating in 4x mode (Port n+1 not available) 1 = Ports n and n+1 operating in 1x mode Note: The MAC_MODE in the "SRIO MAC x Digital Loopback and Clock Selection Register" on page 369 overrides and determine the operating mode for the corresponding ports.
SP_IO_SPEED[1:0]	Serial Port Transmit and Receive operating frequency select. SP_IO_SPEED[1:0], these pin select the power-up serial port frequency for *all* ports. 00 = 1.25Gbit/s 01 = 2.5Gbit/s 10 = 3.125Gbit/s (default) 11 = illegal Note; The SP_IO_SPEED[1:0] setting is equal to the IO_SPEED field in the "SRIO MAC x Digital Loopback and Clock Selection Register" on page 369. Output capability of this pin is only used in test mode.
SP{n}_PWRDN	Port n Transmit and Receive Power Down Control This signal controls the state of Port n *and Port n+1* The PWRDN controls the state of all four lanes (A/B/C/D) of SerDes Macro. 0 = Port n Powered Up. Port n+1 controlled by SP{n+1}_PWRDN. 1 = Port n Powered Down. Port n+1 Powered Down. Override SP{n}_PWRDN using PWDN_X4 field in the "SRIO MAC x Digital Loopback and Clock Selection Register" on page 369.
SP{n+1}_PWRDN	Port n+1 Transmit and Receive Power Down Control This signal controls the state of Port n+1. Note that Port n+1 is never used when 4x mode is selected for a Serial Rapid I/O MAC, and it must be powered down. 0 = Port n+1 Powered Up 1 = Port n+1 Powered Down Override SP{n+1}_PWRDN using PWDN_X1 field in the "SRIO MAC x Digital Loopback and Clock Selection Register" on page 369.
SP_RX_SWAP	Configures the order of 4x receive/transmit lanes on serial ports [0,2,4,6,] 0 = A, B, C, D 1 = D, C, B, A Override SP_RX(TX)_SWAP using RX(TX)_SWAP field in the "SRIO MAC x Digital Loopback and Clock Selection Register" on page 369.
SP_TX_SWAP	

Table 28: Power-Up Options Signals

Pin Name	Description
I2C_MA	I^2C Multibyte Address When driven high, I^2C module expects multi-byte peripheral addressing; otherwise, when driven low, single-byte peripheral address is assumed.
I2C_SA[1,0]	I^2C Slave Address pins. The values on these two pins represent the values for the lower 2 bits of the 7-bit address of Tsi574 when acting as an I^2C slave.
I2C_SEL	I^2C Pin Select. Together with the I2C_SA[1,0] pins, Tsi574 determines the lower 2 bits of the 7-bit address of the EEPROM address it boots from. When asserted, the I2C_SA[1,0] values are also used as the lower 2 bits of the EEPROM address. When de-asserted, the I2C_SA[1,0] pins are ignored and the lower 2 bits of the EEPROM address are default to 00.
I2C_DISABLE	Disable I^2C register loading after reset. When asserted, the Tsi574 does not attempt to load register values from I^2C. 0 = Enable I^2C register loading 1 = Disable I^2C register loading Must remain stable for 10 P_CLK cycles after HARD_RST_b is de-asserted in order to be sampled correctly. For alternative P_CLK values, refer to “Clocking” on page 475. This signal is ignored after reset.

10.3.2 Default Port Speed

When the SP_IO_SPEED[1:0] pins are left unconnected in the board, the device’s internal pull-ups configure the Tsi574 to 3.125 Gbit/s (default). The speed can be overridden by the IO_SPEED field in the “SRIO MAC x Digital Loopback and Clock Selection Register” on page 369.

It is strongly recommended to drive the SP_IO_SPEED[1:0] with known values instead of relying on the internal default values in order to set the default speeds of the device.

10.3.3 Port Power-up and Power-down

The power-up and power-down is overridden by the PWDN_X1 and PWDN_X4 fields in the “SRIO MAC x Digital Loopback and Clock Selection Register” on page 369.

10.3.4 Port Width Override

Initial port width of the port is set by SPx_MODESEL pins at power-up. After power-up the SPx_MODESEL signals are ignored and the port width setting can be overridden by the OVER_PWIDTH field in the “RapidIO Serial Port x Control CSR” on page 273.

11. Signals

This chapter describes the signals and pinout of the Tsi574. It includes the following information:

11.1 Overview

The following conventions are used in the signal description table:

- Signals with the suffix "_p" are the positive half of a differential pair.
- Signals with the suffix "_n" are the negative half of a differential pair.
- Signals with the suffix "_b" are active low.

Signals are classified according to the types defined in Table 29.

Table 29: Signal Types

Pin Type	Definition
I	Input
O	Output
I/O	Input/Output
OD	Open Drain
SRIO	Differential driver/receiver defined by *RapidIO Interconnect Specification (Revision 1.3)*
CML	Current Mode Logic Defined by *RapidIO Interconnect Specification (Revision 1.3)*
PU	Pulled up internal to the Tsi574
PD	Pulled down internal to the Tsi574
LVTTL	CMOS I/O with LVTTL thresholds
Hyst	Hysteresis

Table 29: Signal Types *(Continued)*

Pin Type	Definition
Core Power	Core supply
Core Ground	Ground for core logic
I/O Power	I/O supply
N/C	No connect These signals must be left unconnected.

11.2 Endian Ordering

This document follows the bit-numbering convention adopted by *RapidIO Interconnect Specification (Revision 1.3)*, where [0:7] is used to represent an 8 bit bus with bit 0 as the most-significant bit.

11.3 Port Numbering

The following table shows the mapping between port numbers and the physical ports. These port numbers are used within the destination ID lookup tables for ingress RapidIO ports and in numerous register configuration fields

Table 30: Tsi574 Port Numbering

Port Number	RapidIO Port	Mode
0	Serial Port 0 (SP0)	1x or 4x
1	Serial Port 1 (SP1)	1x
2	Serial Port 2 (SP2)	1x or 4x
3	Serial Port 3 (SP3)	1x
4	Serial Port 4 (SP4)	1x or 4x
5	Serial Port 5 (SP5)	1x
6	Serial Port 6 (SP6)	1x or 4x
7	Serial Port 7 (SP7)	1x

11.4 Signal Groupings

Table 31 describes the Tsi574 signals.

Table 31: Tsi574 Signal Descriptions

Pin Name	Type	Description	Recommended Termination[a]
Signal Port Numbering **PORT n - 1x/4x Mode Serial RapidIO** **PORT (n+1) - 1x Mode Serial RapidIO** **Port {n} where {n} = 0, 2, 4, 6**			
Serial Port Transmit			
SP{n}_TA_p	O, SRIO	Port n Lane A Differential Non-inverting Transmit Data output (4x mode) Port n Lane A Differential Non-inverting Transmit Data output (1x mode)	No termination required.
SP{n}_TA_n	O, SRIO	Port n Lane A Differential Inverting Transmit Data output (4x mode) Port n Lane A Differential Inverting Transmit Data output (1x mode)	No termination required.
SP{n}_TB_p	O, SRIO	Port n Lane B Differential Non-inverting Transmit Data output (4x mode) Port n+1 Lane B Differential Non-inverting Transmit Data output (1x mode)	No termination required.
SP{n}_TB_n	O, SRIO	Port n Lane B Differential Inverting Transmit Data output (4x mode) Port n+1 Lane B Differential Inverting Transmit Data output (1x mode)	No termination required.
SP{n}_TC_p	O, SRIO	Port n Lane C Differential Non-inverting Transmit Data output (4x mode)	No termination required.
SP{n}_TC_n	O, SRIO	Port n Lane C Differential Inverting Transmit Data output (4x mode)	No termination required.
SP{n}_TD_p	O, SRIO	Port n Lane D Differential Non-inverting Transmit Data output (4x mode)	No termination required.
SP{n}_TD_n	O, SRIO	Port n Lane D Differential Inverting Transmit Data output (4x mode)	No termination required.
Serial Port Receive			
SP{n}_RA_p	I, SRIO	Port n Lane A Differential Non-inverting Receive Data input (4x mode) Port n Lane A Differential Non-inverting Receive Data input (1x mode)	DC blocking capacitor of 0.1uF in series

Table 31: Tsi574 Signal Descriptions *(Continued)*

Pin Name	Type	Description	Recommended Termination[a]
SP{n}_RA_n	I, SRIO	Port n Lane A Differential Inverting Receive Data input (4x node) Port n Lane A Differential Inverting Receive Data input (1x mode)	DC blocking capacitor of 0.1uF in series
SP{n}_RB_p	I, SRIO	Port n Lane B Differential Non-inverting Receive Data input (4x mode) Port n+1 Lane B Differential Non-inverting Receive Data input (1x mode)	DC blocking capacitor of 0.1uF in series
SP{n}_RB_n	I, SRIO	Port n Lane B Differential Inverting Receive Data input (4x mode) Port n+1 Lane B Differential Inverting Receive Data input (1x mode)	DC blocking capacitor of 0.1uF in series
SP{n}_RC_p	I, SRIO	Port n Lane C Differential Non-inverting Receive Data input (4x mode)	DC blocking capacitor of 0.1uF in series
SP{n}_RC_n	I, SRIO	Port n Lane C Differential Inverting Receive Data input (4x mode)	DC blocking capacitor of 0.1uF in series
SP{n}_RD_p	I, SRIO	Port n Lane D Differential Non-inverting Receive Data input (4x mode)	DC blocking capacitor of 0.1uF in series
SP{n}_RD_n	I, SRIO	Port n Lane D Differential Inverting Receive Data input (4x mode)	DC blocking capacitor of 0.1uF in series
Serial Port Configuration			
SP{n}_REXT	Analog	Used to connect a resistor to VSS to provide a reference current for the driver and equalization circuits.	Must be connected to VSS with a 191-ohm (1%) resistor.
SP{n}_MODESEL	I/O, LVTTL, PD	Selects the serial port operating mode for ports n and n+1 0 = Port n operating in 4x mode (Port n+1 not available) 1 = Ports n and n+1 operating in 1x mode Output capability of this pin is only used in test mode. Must remain stable for 10 P_CLK cycles after HARD_RST_b is de-asserted in order to be sampled correctly. For alternative P_CLK values, refer to “Clocking” on page 475. This signal is ignored after reset.	Pin must be tied off according to the required configuration. Either a 10K pull up to VDD_IO or a 10K pull-down to VSS_IO. Internal pull-down can be used for logic 0.

Table 31: Tsi574 Signal Descriptions *(Continued)*

Pin Name	Type	Description	Recommended Termination[a]
SP{n}_PWRDN	I/O, LVTTL, PU	Port n Transmit and Receive Power Down Control This signal controls the state of Port n *and Port n+1.* The PWRDN controls the state of all four lanes (A/B/C/D) of SerDes Macro. 0 = Port n Powered Up. Port n+1 controlled by SP{n+1}_PWRDN. 1 = Port n Powered Down. Port n+1 Powered Down. Override SP{n}_PWRDN using PWDN_X4 field in the "SRIO MAC x Digital Loopback and Clock Selection Register" on page 369. Output capability of this pin is only used in test mode. This signal must remain stable for 10 P_CLK cycles after HARD_RST_b is de-asserted in order to be sampled correctly. This signal is ignored after reset.	Pin must be tied off according to the required configuration. Either a 10K pull up to VDD_IO or a 10K pull-down to VSS_IO. Internal pull-up can be used for logic 1.
SP{n+1}_PWRDN	I/O, LVTTL, PU	Port n+1 Transmit and Receive Power Down Control This signal controls the state of Port n+1. Note that Port n+1 is never used when 4x mode is selected for a Serial Rapid I/O MAC, and it must be powered down. 0 = Port n+1 Powered Up 1 = Port n+1 Powered Down Override SP{n+1}_PWRDN using PWDN_X1 field in "SRIO MAC x Digital Loopback and Clock Selection Register" on page 369. Output capability of this pin is only used in test mode. This signal must remain stable for 10 P_CLK cycles after HARD_RST_b is de-asserted in order to be sampled correctly. This signal is ignored after reset.	Pin must be tied off according to the required configuration. Either a 10K pull up to VDD_IO or a 10K pull-down to VSS_IO. Internal pull-up can be used for logic 1.

Table 31: Tsi574 Signal Descriptions *(Continued)*

Pin Name	Type	Description	Recommended Termination[a]
Serial Port Speed Select			
SP_IO_SPEED[1]	I/O, LVTTL, PU	Serial Port Transmit and Receive operating frequency select, bit 1. When combined with SP_IO_SPEED[0], this pin selects the default serial port frequency for all ports. 00 = 1.25 Gbps 01 = 2.5 Gbps 10 =3.125 Gbps (default) 11 = Illegal Selects the speed at which the ports operates when reset is removed. This could be at either HARD_RST_b being de-asserted or by the completion of a self-reset. These signals must remain stable for 10 P_CLK cycles after HARD_RST_b is de-asserted in order to be sampled correctly. These signals are ignored after reset and software is able to over-ride the port frequency setting in the "SRIO MAC x Digital Loopback and Clock Selection Register" on page 369. The SP_IO_SPEED[1:0] setting is equal to the IO_SPEED field in the SRIO MAC x Clock Selection Register. Output capability of this pin is only used in test mode.	Pin must be tied off according to the required configuration. Either a 10K pull-up to VDD_IO or a 10K pull-down to VSS_IO. Internal pull-down may be used for logic 0.
SP_IO_SPEED[0]	I/O, LVTTL, PD	See SP_IO_SPEED[1]	Pin must be tied off according to the required configuration. Either a 10K pull-up to VDD_IO or a 10K pull-down to VSS_IO. Internal pull-up may be used for logic 1.

Table 31: Tsi574 Signal Descriptions *(Continued)*

Pin Name	Type	Description	Recommended Termination[a]
Serial Port Lane Ordering Select			
SP_RX_SWAP	I, LVTTL, PD	Configures the order of 4x receive lanes on serial ports[. 0 = A, B, C, D 1 = D, C, B, A Must remain stable for 10 P_CLK cycles after HARD_RST_b is de-asserted in order to be sampled correctly. This signal is ignored after reset. Note: Ports that require the use of lane swapping for ease of routing will only function as 4x mode ports. The re-configuration of a swapped port to dual 1x mode operation results in the inability to connect to a 1x mode link partner.	No termination required. Internal pull-down can be used for logic 0. Pull up7 to VDD_IO through 10K if external pull-up is desired. Pull down to VSS_IO through a 10K resistor if an external pull-down is desired.
SP_TX_SWAP	I, LVTTL, PD	Configures the order of 4x transmit lanes on serial ports [. 0 = A, B, C, D 1 = D, C, B, A Must remain stable for 10 P_CLK cycles after HARD_RST_b is de-asserted in order to be sampled correctly. This signal is ignored after reset. Note: Ports that require the use of lane swapping for ease of routing only function as 4x mode ports. The re-configuration of a swapped port to dual 1x mode operation results in the inability to connect to a 1x mode link partner.	No termination required. Internal pull-down can be used for logic 0. Pull up to VDD_IO through 10K if external pull-up is desired. Pull down to VSS_IO through a 10K resistor if an external pull-down is desired.
Clock and Reset			
P_CLK	I, LVTTL	This clock is used for the register bus clock. The nominal frequency of this input clock is 100 MHz. For more information on programming the P_CLK operating frequency, refer to “P_CLK Programming” on page 479.	No termination required.

Table 31: Tsi574 Signal Descriptions *(Continued)*

Pin Name	Type	Description	Recommended Termination[a]
S_CLK_p	I, CML	Differential reference clock at 156.25MHz. The clock is used as a reference clock for the SerDes and as a source for the internal logic clocks. These signals are internally terminated. The S_CLK inputs should be ac-coupled.	AC coupling capacitor of 0.1uF required. If this clock input is not used, pull this signal up to SP_VDD with a 10K resistor.
S_CLK_n	I, CML		AC coupling capacitor of 0.1uF required. If this clock input is not used, pull this signal down to VSS with a 10K resistor.
HARD_RST_b	I, LVTTL, Hyst, PU	Schmidt-triggered hard reset. Asynchronous active low reset for the entire device.	Connect to a power-up reset source. Refer to the *Tsi574 Hardware Manual* for more information.
Interrupts			
INT_b	O, OD, LVTTL	Interrupt signal (open drain output)	External pull-up required. Pull up to VDD_IO through a 10K resistor.
SW_RST_b	O, OD, LVTTL	Software reset (open drain output): This signal is asserted when a RapidIO port receives a valid reset request on a RapidIO link. If self-reset is not selected, this pin remains asserted until the reset request is cleared from the status registers. If self-reset is selected, this pin remains asserted until the self reset is complete. If the Tsi574 is reset from the HARD_RST_b pin, this pin is de-asserted and remains de-asserted after HARD_RST_b is released. For more information, see "Resets". Note: SW_RST_b is the only external indicator that a reset request has been received, and should be handled as an interrupt. Port-writes cannot be sent for notification of reset request reception.	External pull-up required. Pull up to VDD_IO through a 10K resistor.
Multicast			
MCES	I/O, LVTTL, PD	Multicast Event Control Symbol As an input, an edge (rising or falling) triggers a Multicast Event Control Symbol to be sent to all enabled ports. As an output, this pin toggles its value every time an Multicast Event Control Symbol is received by any port which is enabled for Multicast Event Control Symbols. This signal must remain stable for 10 P_CLK cycles before and after a transition.	No termination required. This pin must not be driven by an external source until all power supply rails are stable.

Table 31: Tsi574 Signal Descriptions *(Continued)*

Pin Name	Type	Description	Recommended Termination[a]
I²C			
I2C_SCLK	I/O, OD, LVTTL, PU	I²C input/output clock, up to 100 kHz. If an EEPROM is present on the I²C bus, this clock signal must be connected to the clock input of the serial EEPROM on the I²C bus. If an EEPROM is not present, the recommended termination should be used.	No termination required. Internal pull-up can be used for logic 1. Pull up to VDD_IO through a minimum 470 ohms resistor if higher edge rate required.
I2C_SD	I/O, OD, LVTTL, PU	I²C input and output data bus (bidirectional open drain)	No termination required. Internal pull-up can be used for logic 1. Pull up to VDD_IO through a minimum 470 ohms resistor if higher edge rate required.
I2C_DISABLE	I, LVTTL, PD	Disable I²C register loading after reset. When asserted, the Tsi574 does not attempt to load register values from I²C. 0 = Enable I²C register loading 1 = Disable I²C register loading Must remain stable for 10 P_CLK cycles after HARD_RST_b is de-asserted in order to be sampled correctly. Note: This signal does not control the slave accessibility of the interface. This signal is ignored after reset.	No termination required. Pull up to VDD_IO through a 10K resistor if I²C loading is not required.
I2C_MA	I, LVTTL, PU	I²C Multibyte Address. When driven high, I²C module expects multi-byte peripheral addressing; otherwise, when driven low, single-byte peripheral address is assumed. Must remain stable for 10 P_CLK cycles after HARD_RST_b is de-asserted in order to be sampled correctly. This signal is ignored after reset.	No termination required. Internal pull-up can be used for logic 1. Pull up to VDD_IO through 10K resistor if external pull-up is desired. Pull down to VSS_IO to change the logic state.
I2C_SA[1,0]	I, LVTTL, PU	I²C Slave Address pins. The values on these two pins represent the values for the lower 2 bits of the 7-bit address of Tsi574 when acting as an I²C slave (see "I²C Slave Configuration Register" on page 425). The values at these pins can be overridden by software after reset.	No termination required. Internal pull-up can be used for logic 1. Pull up to VDD_IO through 10K resistor if external pull-up is desired. Pull down to VSS_IO to change the logic state.

Table 31: Tsi574 Signal Descriptions *(Continued)*

Pin Name	Type	Description	Recommended Termination[a]
I2C_SEL	I, LVTTL, PU	I^2C Pin Select. Together with the I2C_SA[1,0] pins, the Tsi574 determines the lower 2 bits of the 7-bit address of the EEPROM address it boots from. When asserted, the I2C_SA[1,0] values are also be used as the lower two bits of the EEPROM address. When de-asserted, the I2C_SA[1,0] pins are ignored and the lower two bits of the EEPROM address are default to 00. The values of the lower two bits of the EEPROM address can be over-ridden by software after reset.	No termination required. Internal pull-up can be used for logic 1. Pull up to VDD_IO through a 10K resistor if external pull-up is desired. Pull down to VSS_IO to change the logic state.
JTAG / TAP Controller			
TCK	I, LVTTL, PD	IEEE 1149.1/1149.6 Test Access Port Clock input	Pull up to VDD_IO through a 10K resistor if not used.
TDI	I, LVTTL, PU	IEEE 1149.1/1149.6 Test Access Port Serial Data Input	Pull up to VDD_IO through 10K if not used or if higher edge rate is required.
TDO	O, LVTTL	IEEE 1149.1/1149.6 Test Access Port Serial Data Output	No connect if JTAG is not used. Pull up to VDD_IO through a 10K resistor if used.
TMS	I, LVTTL, PU	IEEE 1149.1/1149.6 Test Access Port Test Mode Select	Pull up to VDD_IO through a 10K resistor if not used.
TRST_b	I, LVTTL, PU	IEEE 1149.1/1149.6 Test Access Port TAP Reset Input This input must be asserted during the assertion of HARD-RST_b. Afterwards, it can be left in either state. Combine the HARD_RST_b and TRST_b signals with an AND gate and use the output to drive the TRST_b pin.	Tie to VSS_IO through a 10K resistor if not used.
BCE	I, LVTTL, PU	Boundary Scan compatibility enabled pin. This input is used to aid 1149.6 testing. This signal also enables system level diagnostic capability using features built into the SerDes. For more information on this functionality, refer to the Serial RapidIO Signal Analyzer documentation available on the IDT website. This signal must be tied to VDD_IO during normal operation of the device, and during JTAG accesses of the device registers	This signal should have the capability to be pulled-up or pulled-low. • The default setting is to be pulled-up. • Pulling the signal low enables the signal analyzer functionality on the SerDes • A 10K resistor to VDD_IO should be used.

Table 31: Tsi574 Signal Descriptions *(Continued)*

Pin Name	Type	Description	Recommended Termination[a]
Power Supplies			
SP_AVDD	-	Port n and n+1: 3.3V supply for bias generator circuitry. This is required to be a low-noise supply.	Refer to decoupling recommendations in the *Tsi574 Hardware Manual* for more information
REF_AVDD	-	Analog 1.2V for Reference Clock (S_CLK_P/N). Clock distribution network power supply.	Refer to decoupling recommendations in the *Tsi574 Hardware Manual* for more information
Common Supply			
VDD_IO	-	Common 3.3V supply for LVTTL I/O	Refer to decoupling recommendations in the *Tsi574 Hardware Manual* for more information
VSS	-	Common ground returns for digital logic	Refer to decoupling recommendations in the *Tsi574 Hardware Manual* for more information
VDD	-	Common 1.2V supply for digital logic	Refer to decoupling recommendations in the *Tsi574 Hardware Manual* for more information
SP_VDD	-	1.2V supply for CDR, Tx/Rx, and digital logic for all RapidIO ports	For more information on decoupling recommendations, refer to the *Tsi574 Hardware Manual*

a. Signals for unused serial ports do not require termination and can be left as N/Cs.

11.5 Pinlist and Ballmap

For more information, see the following documents:

- *Tsi574 Pinlist*
- *Tsi574 Ballmap*

12. Serial RapidIO Registers

This chapter describes the Tsi574 registers. The following topics are discussed:

12.1 Overview

The application defined Tsi574 registers receive initial values during power on initialization through the I^2C Interface and external serial EEPROM; all undefined registers read 0 and a write is ignored.

The Tsi574 registers use direct addressing of 32-bit registers. The *RapidIO Interconnect Specification (Revision 1.3)*, uses 64-bit addressing of registers. Table 32 shows the rules used to associate the register offsets in both specifications.

Table 32: Address Rules

Tsi574 Address — Register Offset	RapidIO Specification Address — Register Offset
0x*XXXX*0	0x*XXXX*0, Word 0
0x*XXXX*4	0x*XXXX*0, Word 1
0x*XXXX*8	0x*XXXX*8, Word 0
0x*XXXX*C	0x*XXXX*8, Word 1

12.1.1 Reserved Register Addresses and Fields

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

For the RapidIO Standard Registers (Section 12.5 to 12.7), the Reserved fields should always be written as 0.

For the IDT implementation-specific registers (Section 12.8 to 13.2), a read modify write operation must be performed for register reserved fields that have an undefined reset value. Other reserved fields should always be written as 0 unless otherwise noted.

Table 33 shows the defined register access types.

Table 33: Register Access Types

Abbreviation	Description
R	Read Only Note: R registers are not read/write during the I^2C boot unless it specified in the register description
R/W	Read or Write Note: All R/W registers are read/write during the I^2C boot.
R/W1C	Readable Write 1 to Clear
R/W0C	Readable Write 0 to Clear.
R/W1S	Readable Write 1 to Set.Writing a 1 triggers an event, bit reads as 0.
RC	Read, then automatically Clear These fields are writable for test purposes.

12.2 Port Numbering

The following table shows the mapping between port numbers and the physical ports. These port numbers are used within the destination ID lookup tables for ingress RapidIO ports and in numerous register configuration fields.

The odd ports are unavailable when the even port is in 4x mode.

Table 34: Port Numbering

Port Number	RapidIO Port	Mode
0	Serial Port 0 (SP0)	1x or 4x
1	Serial Port 1 (SP1)	1x
2	Serial Port 2 (SP2)	1x or 4x
3	Serial Port 3 (SP3)	1x
4	Serial Port 4 (SP4)	1x or 4x
5	Serial Port 5 (SP5)	1x
6	Serial Port 6 (SP6)	1x or 4x
7	Serial Port 7 (SP7)	1x

12.3 Conventions

In many instances, there are multiple instances of a register, for example, one instance per RapidIO port. Two notations are used to refer to these registers.

- In the first notation, a lower-case letter such as “x” is used as a wildcard character. For example, Sx_DESTID refers to S0_DESTID, S1_DESTID, S2_DESTID, and so on.
- In the second notation, the names of the instances are explicitly listed. For example, S{BC,0..2}_DESTID refers to registers SBC_DESTID, S0_DESTID, S1_DESTID, and S2_DESTID.

Generally, the instance number refers to a RapidIO port number. The special instance “BC” (broadcast) refers to a register that when written simultaneously affects all powered-up ports, and that when read returns a value from port number 0.

Port 0 should not be powered-down. Refer to “Special Conditions for Port 0 Power Down” on page 72 for more information

12.4 Register Map

Table 35 gives an overview of the Tsi574 register map.

Table 35: Register map overview

Register Group	Start Address	End Address
"RapidIO Logical Layer and Transport Layer Registers" on page 238	0x00000	0x000FC
"RapidIO Physical Layer Registers" on page 261	0x00100	0x0033C
Reserved	0x00340	0x00FFC
"RapidIO Error Management Extension Registers" on page 277	0x01000	0x0143C
Reserved	0x014C0	0x0FFFF
"IDT-Specific RapidIO Registers" on page 299 and Serial Port Electrical Layer Registers	0x10000	0x14FFC
Reserved	0x15000	0x1A9FC
"Internal Switching Fabric (ISF) Registers" on page 372	0x1AA00	0x1AAFC
Reserved	0x1AB00	0x1ABFC
"Utility Unit Registers" on page 377	0x1AC00	0x1ACFC
Reserved	0x1AD00	0x1AFFC
Fabric Arbitration Registers	0x1B000	0x1BFFC
Reserved	0x1C000	0x1CFFC
Tsi574 I^2C Registers These registers are not described in this chapter, they are described in the *I^2C Register Chapter.*	0x1D000	0x1DFFC
"SerDes Per Lane Register" on page 390	0x1E000	0x1EFFC
Reserved	0x1F000	0x1FFFC

Table 36 shows the Tsi574 register map.

Table 36: Register Map

Offset	Register Name	See
RapidIO Logical Layer and Transport Layer Registers		
00000	RIO_DEV_ID	“RapidIO Device Identity CAR” on page 239
00004	RIO_DEV_INFO	“RapidIO Device Information CAR” on page 240
00008	RIO_ASBLY_ID	“RapidIO Assembly Identity CAR” on page 241
0000C	RIO_ASBLY_INFO	“RapidIO Assembly Information CAR” on page 242
00010	RIO_PE_FEAT	“RapidIO Processing Element Features CAR” on page 243
00014	RIO_SW_PORT	“RapidIO Switch Port Information CAR” on page 245
00018	RIO_SRC_OP	“RapidIO Source Operation CAR” on page 246
00020 - 0002C	Reserved	
00030	RIO_PE_MC_FEAT	“RapidIO Switch Multicast Support CAR” on page 248
00034	RIO_LUT_SIZE	“RapidIO Route LUT Size CAR” on page 249
00038	RIO_SW_MC_INFO	“RapidIO Switch Multicast Information CAR” on page 250
0003C - 00064	Reserved	
00068	RIO_HOST_BASE_ID_LOCK	“RapidIO Host Base Device ID Lock CSR” on page 251
0006C	RIO_COMP_TAG	“RapidIO Component Tag CSR” on page 252
00070	RIO_ROUTE_CFG_DESTID	“RapidIO Route Configuration DestID CSR” on page 253
00074	RIO_ROUTE_CFG_PORT	“RapidIO Route Configuration Output Port CSR” on page 254
00078	RIO_LUT_ATTR	“RapidIO Route LUT Attributes (Default Port) CSR” on page 255
0007C	Reserved	
00080	RIO_MC_MASK_CFG	“RapidIO Multicast Mask Configuration Register” on page 256
00084	RIO_MC_DESTID_CFG	“RapidIO Multicast DestID Configuration Register” on page 258
00088	RIO_MC_DESTID_ASSOC	“RapidIO Multicast DestID Association Register” on page 259
0008C - 000FC	Reserved	
RapidIO Physical Layer Registers (using extended features block ID = 0x0009)		
General Physical Layer Registers		

Table 36: Register Map *(Continued)*

Offset	Register Name	See
00100	RIO_SP_MB_HEAD	“RapidIO 1x or 4x Switch Port Maintenance Block Header” on page 262
00104 - 0011C	Reserved	
00120	RIO_SW_LT_CTL	“RapidIO Switch Port Link Timeout Control CSR” on page 263
00124 - 00138	Reserved	
0013C	RIO_SW_GEN_CTL	“RapidIO Switch Port General Control CSR” on page 264
Serial Port 0 Registers (Offset 0x140 - 0x15C)		
00140	SP0_LM_REQ	“RapidIO Serial Port x Link Maintenance Request CSR” on page 265
00144	SP0_LM_RESP	“RapidIO Serial Port x Link Maintenance Response CSR” on page 267
00148	SP0_ACKID_STAT	“RapidIO Serial Port x Local ackID Status CSR” on page 268
0014C - 00154	Reserved	
00158	SP0_ERR_STATUS	“RapidIO Port x Error and Status CSR” on page 270
0015C	SP0_CTL	“RapidIO Serial Port x Control CSR” on page 273
160 - 17C	Serial Port 1	Same set of registers as Serial Port 0, offset 140 - 15C.
180 - 19C	Serial Port 2	
1A0 - 1BC	Serial Port 3	
1C0 - 1DC	Serial Port 4	
1E0 - 1FC	Serial Port 5	
200 - 21C	Serial Port 6	
220 - 23C	Serial Port 7	
240 - FFC	Tsi574 Reserved	
RapidIO Error Management Extensions		
General Error Management Registers		
01000	RIO_ERR_RPT_BH	“RapidIO Error Reporting Block Header” on page 279
01004	Reserved	
01008	RIO_LOG_ERR_DET	“RapidIO Logical and Transport Layer Error Detect CSR” on page 280

Table 36: Register Map *(Continued)*

Offset	Register Name	See
0100C	RIO_LOG_ERR_DET_EN	“RapidIO Logical and Transport Layer Error Enable CSR” on page 281
01014	RIO_LOG_ERR_ADDR	“RapidIO Logical and Transport Layer Address Capture CSR” on page 282
01018	RIO_LOG_ERR_DEVID	“RapidIO Logical and Transport Layer Control Capture CSR” on page 284
0101C	RIO_LOG_ERR_CTRL_INFO	“RapidIO Logical and Transport Layer Device ID Capture CSR” on page 283
01020 - 01024	Reserved	
01028	RIO_PW_DESTID	“RapidIO Port-Write Target Device ID CSR” on page 285
0102C - 0103C	Reserved	
Per Port Error Management Registers		
01040	SP0_ERR_DET	“RapidIO Port x Error Detect CSR” on page 286
01044	SP0_RATE_EN	“RapidIO Port x Error Rate Enable CSR” on page 289
01048	SP0_ERR_ATTR_CAPT_DBG0	“RapidIO Port x Error Capture Attributes CSR and Debug 0” on page 291
0104C	SP0_ERR_ATTR_CAPT_0_DBG1	“RapidIO Port x Packet and Control Symbol Error Capture CSR 0 and Debug 1” on page 293
01050	SP0_ERR_ATTR_CAPT_1_DBG2	“RapidIO Port x Packet Error Capture CSR 1 and Debug 2” on page 294
01054	SP0_ERR_ATTR_CAPT_2_DBG3	“RapidIO Port x Packet Error Capture CSR 2 and Debug 3” on page 294
01058	SP0_ERR_ATTR_CAPT_3_DBG4	“RapidIO Port x Packet Error Capture CSR 3 and Debug 4” on page 295
0105C - 1064	Reserved	
01068	SP0_ERR_RATE	“RapidIO Port x Error Rate CSR” on page 296
0106C	SP0_ERR_THRESH	“RapidIO Port x Error Rate Threshold CSR” on page 298
01070 - 0107C	Reserved	

Table 36: Register Map *(Continued)*

Offset	Register Name	See
01080 - 010BC	Serial Port 1	Same set of registers as for SP0, offsets 0x01040 - 0x0107C.
010C0 - 010FC	Serial Port 2	
01100 - 0113C	Serial Port 3	
01140 - 0117C	Serial Port 4	
01180 - 011BC	Serial Port 5	
011C0 - 011FC	Serial Port 6	
01200 - 0123C	Serial Port 7	
01240 - 0FFFC	Tsi574 Reserved	
IDT-Specific RapidIO Registers		
Broadcast Registers (Offset 10000 - 10FFC) - Writing these registers affects all ports. Read data comes from port SP0.		
10000	SPBC_DISCOVERY_ TIMER	“RapidIO Port x Discovery Timer” on page 301
10004	SPBC_MODE	“RapidIO Port x Mode CSR” on page 302
10008	SPBC_CS_INT_STATUS	“RapidIO Port x Multicast-Event Control Symbol and Reset Control Symbol Interrupt CSR” on page 304
1000C	SPBC_RIO_WM	“RapidIO Port x RapidIO Watermarks” on page 305
10010 - 1006C	Reserved	
10070	SPBC_ROUTE_CFG_DESTID	“RapidIO Port x Route Config DestID CSR” on page 306
10074	SPBC_ROUTE_CFG_ PORT	“RapidIO Port x Route Config Output Port CSR” on page 307
10078	SPBC_ROUTE_BASE	“RapidIO Port x Local Routing LUT Base CSR” on page 308
1007C - 102FC	Reserved	
10300, 10304, 10308, 1030C, 10310, 10314, 10318, 1031C	RIO_MC_ID{0..7}	“RIO Multicast Write ID x” on page 245
10320, 10324, 10328, 1032C, 10330, 10334, 10338, 1033C	RIO_MC_MASK{0..7}	“RIO Multicast Write Mask x” on page 246

Table 36: Register Map *(Continued)*

Offset	Register Name	See
Per Port Instances of the Broadcast Registers (Offset 11000 - 110FC) Writing/reading these registers is a port specific operation.		
11000 - 110FC	Serial Port 0	Same set of registers as Broadcast Registers, offset 10000 - 100FC.
11100 - 111FC	Serial Port 1	
11200 - 112FC	Serial Port 2	
11300 - 113FC	Serial Port 3	
11400 - 114FC	Serial Port 4	
11500 - 115FC	Serial Port 5	
11600 - 116FC	Serial Port 6	
11700 - 117FC	Serial Port 7	
11800 - 12FFC	Tsi574 Reserved	
Non-Broadcast Per Port Registers		
13000	Reserved	
13004	SP0_CTL_INDEP	“RapidIO Port x Control Independent Register” on page 311
13008	Reserved	
1300C	SP0_SEND_MCS	“RapidIO Port x Send Multicast-Event Control Symbol Register” on page 314
13010	SP0_LUT_PAR_ERR_INFO	“RapidIO Port x LUT Parity Error Info CSR” on page 315
13014	SP0_CS_TX	“RapidIO Port x Control Symbol Transmit” on page 317
13018	SP0_INT_STATUS	“RapidIO Port x Interrupt Status Register” on page 318
1301C	SP0_INT_GEN	“RapidIO Port x Interrupt Generate Register” on page 321
13020	SP0_PSC0n1_CTRL	“RapidIO Port x Performance Statistics Counter 0 and 1 Control Register” on page 324
13024	SP0_PSC2n3_CTRL	“RapidIO Port x Performance Statistics Counter 2 and 3 Control Register” on page 328
13028	SP0_PSC4n5_CTRL	“RapidIO Port x Performance Statistics Counter 4 and 5 Control Register” on page 332
1302C - 1303C	Reserved	
13040	SP0_PSC0	“RapidIO Port x Performance Statistics Counter 0 Register” on page 336

Table 36: Register Map *(Continued)*

Offset	Register Name	See
13044	SP0_PSC1	“RapidIO Port x Performance Statistics Counter 1 Register” on page 337
13048	SP0_PSC2	“RapidIO Port x Performance Statistics Counter 2 Register” on page 338
1304C	SP0_PSC3	“RapidIO Port x Performance Statistics Counter 3 Register” on page 339
13050	SP0_PSC4	“RapidIO Port x Performance Statistics Counter 4 Register” on page 340
13054	SP0_PSC5	“RapidIO Port x Performance Statistics Counter 5 Register” on page 341
13058 - 1307C	Reserved	
13080	SP0_TX_Q_D_THRESH	“RapidIO Port x Transmitter Output Queue Depth Threshold Register” on page 342
13084	SP0_TX_Q_STATUS	“RapidIO Port x Transmitter Output Queue Congestion Status Register” on page 344
13088	SP0_TX_Q_PERIOD	“RapidIO Port x Transmitter Output Queue Congestion Period Register” on page 346
1308C	Reserved	
13090	SP0_RX_Q_D_THRESH	“RapidIO Port x Receiver Input Queue Depth Threshold Register” on page 347
13094	SP0_RX_Q_STATUS	“RapidIO Port x Receiver Input Queue Congestion Status Register” on page 349
13098	SP0_RX_Q_PERIOD	“RapidIO Port x Receiver Input Queue Congestion Period Register” on page 351
1309C	Reserved	
130A0	SP0_REORDER_CTR	“RapidIO Port x Reordering Counter Register” on page 352
130A4-130AC	Reserved	
130B0	SMAC0_CFG_CH0	“SRIO MAC x SerDes Configuration Channel 0” on page 355
130B4	SMAC0_CFG_CH1	“SRIO MAC x SerDes Configuration Channel 1” on page 358
130B8	SMAC0_CFG_CH2	“SRIO MAC x SerDes Configuration Channel 2” on page 360
130BC	SMAC0_CFG_CH3	“SRIO MAC x SerDes Configuration Channel 3” on page 362
130C0	SMAC0_CFG_GBL	“SRIO MAC x SerDes Configuration Global” on page 364

Table 36: Register Map *(Continued)*

Offset	Register Name	See
130C8	SMAC0_DLOOP_CLK_SEL	“SRIO MAC x Digital Loopback and Clock Selection Register” on page 369
130CC	Reserved	
130D0	MCES_PIN_CTRL	“MCES Pin Control Register” on page 382
130D4-130FC	Reserved	
13100 - 131AC	Serial Port 1	Same set of registers as for SP0, offsets 0x13000 - 0x130AC. The registers at offsets 0x130B0 - 0x130FC are excluded.
13200 - 132FC	Serial Port 2	All registers as for SP0, offsets 0x13000 - 0x130FC.
13300 - 133AC	Serial Port 3	Same set of registers as for SP0, offsets 0x13000 - 0x130AC. The registers at offsets 0x130B0 - 0x130FC are excluded.
13400 - 134FC	Serial Port 4	All registers as for SP0, offsets 0x13000 - 0x130FC.
13500 - 135AC	Serial Port 5	Same set of registers as for SP0, offsets 0x13000 - 0x130AC. The registers at offsets 0x130B0 - 0x130FC are excluded.
13600 - 136FC	Serial Port 6	All registers as for SP0, offsets 0x13000 - 0x130FC.
13700 - 137AC	Serial Port 7	Same set of registers as for SP0, offsets 0x13000 - 0x130AC. The registers at offsets 0x130B0 - 0x130FC are excluded.
13800 - 13FAC	Tsi574 Reserved	
Fabric Global Interrupt Registers		
1AA00	FAB_CTL	“Fabric Control Register” on page 372
1AA04	FAB_INT_STAT	“Fabric Interrupt Status Register” on page 374
1AA08	RIO_MC_LAT_ERR	“RapidIO Broadcast Buffer Maximum Latency Expired Error Register” on page 375
1AA0C	RIO_MC_LAT_ERR_SET	“RapidIO Broadcast Buffer Maximum Latency Expired Override” on page 376
1AA10 - 1ABFC	Reserved	
Utility Unit Registers		
1AC00	GLOB_INT_STATUS	“Global Interrupt Status Register” on page 377
1AC04	GLOB_INT_ENABLE	“Global Interrupt Enable Register” on page 379
1AC08 - 1AC10	Reserved	
1AC14	RIO_PW_TIMEOUT	“” on page 379

Table 36: Register Map *(Continued)*

Offset	Register Name	See
1AC18	RIO_PW_OREQ_STATUS	"RapidIO Port Write Outstanding Request Register" on page 381
1AFFC	Reserved	
Multicast Registers		
1B000	RIO0_MC_REG_VER	"RapidIO Multicast Register Version CSR" on page 383
1B004	RIO0_MC_LAT_LIMIT	"RapidIO Multicast Maximum Latency Counter CSR" on page 384
1B008	SP0_ISF_WM	"RapidIO Port x ISF Watermarks" on page 385
1B00C	Reserved	
1B010	SP0_WRR_0	"Port x Prefer Unicast and Multicast Packet Prio 0 Register" on page 386
1B014	SP0_WRR_1	"Port x Prefer Unicast and Multicast Packet Prio 1 Register" on page 387
1B018	SP0_WRR_2	"Port x Prefer Unicast and Multicast Packet Prio 2 Register" on page 388
1B01C	SP0_WRR_3	"Port x Prefer Unicast and Multicast Packet Prio 3 Register" on page 389
1B020 - 1B0FC	Reserved	
1B100 - 1B1FC	Serial Port 1	Same set of registers as Serial Port 0, offset 1B000 - 1B0FC
1B200 - 1B2FC	Serial Port 2	
1B300 - 1B3FC	Serial Port 3	
1B400 - 1B4FC	Serial Port 4	
1B500 - 1B5FC	Serial Port 5	
1B600 - 1B6FC	Serial Port 6	
1B700 - 1B7FC	Serial Port 7	
1B800 - 1CFFC	Tsi574 Reserved	
I^2C Registers		
1D000-1DFFC	See the *I^2C Register Chapter* for a full description of the I^2C registers.	
SerDes Per Lane Registers		
1D000 - 1DFFC	Documented in the *I^2C Register Chapter*	
1E000-1E01C	Reserved	

Table 36: Register Map *(Continued)*

Offset	Register Name	See
1E020	SMAC{0,2,4,6}_PG_CTL_0	"SerDes Lane 0 Pattern Generator Control Register" on page 391
1E024-1E02C	Reserved	
1E030	SMAC{0,2,4,6}_PM_CTL_0	"SerDes Lane 0 Pattern Matcher Control Register" on page 395
1E034	SMAC{0,2,4,6}_FP_VAL_0	"SerDes Lane 0 Frequency and Phase Value Register" on page 399
1E038-1E03C	Reserved	
1E060	SMAC{0,2,4,6}_PG_CTL_1	"SerDes Lane 1 Pattern Generator Control Register" on page 392
1E064-1E06C	Reserved	
1E070	SMAC{0,2,4,6}_PM_CTL_1	"SerDes Lane 1 Pattern Matcher Control Register" on page 396
1E074	SMAC{0,2,4,6}_FP_VAL_1	"SerDes Lane 1 Frequency and Phase Value Register" on page 400
1E078-1E07C	Reserved	
1E0A0	SMAC{0,2,4,6}_PG_CTL_2	"SerDes Lane 2 Pattern Generator Control Register" on page 393
1E0A4-1E0AC	Reserved	
1E0B0	SMAC{0,2,4,6}_PM_CTL_2	"SerDes Lane 2 Pattern Matcher Control Register" on page 397
1E0B4	SMAC{0,2,4,6}_FP_VAL_2	"SerDes Lane 2 Frequency and Phase Value Register" on page 401
1E0B8-1E0BC	Reserved	
1E0E0	SMAC{0,2,4,6}_PG_CTL_3	"SerDes Lane 3 Pattern Generator Control Register" on page 394
1E0E4-1E0EC	Reserved	
1E0F0	SMAC{0,2,4,6}_PM_CTL_3	"SerDes Lane 3 Pattern Matcher Control Register" on page 398
1E0F4	SMAC{0,2,4,6}_FP_VAL_3	"SerDes Lane 3 Frequency and Phase Value Register" on page 402
1E0F8-1E0FC	Reserved	

12.5 RapidIO Logical Layer and Transport Layer Registers

Every processing element contains a set of capability registers (CARs) that allows another processing element to determine its capabilities through maintenance read operations. All registers are 32 bits wide and are organized and accessed in 32-bit quantities. CARs are read-only and are big-endian — bit 0 is the most significant bit.

A processing element contains a set of command and status registers (CSRs) that allows another processing element to control and determine the status of its internal hardware. All registers are organized and accessed in the same way as the CARs.

All of the registers in this section are defined in the *RapidIO Interconnect Specification (Revision 1.3)*.

When an individual port is powered down, the RapidIO Logical Layer and Transport Layer Registers for that port are read only and return 0.

These registers are reset by the HARD_RST_b reset input signal, as well as when the Tsi574 performs a self-reset. The registers within a port are also reset by a "Port Reset". For more information on Tsi574 reset implementation and behavior, see "Clocks, Resets and Power-up Options" on page 203.

It is possible to override reset values of writable fields, and some read-only fields, using the I^2C register loading capability on boot. Refer to "I^2C Interface" on page 139 for more information on the use of I^2C register loading capability.

The I^2C register loading capability is only used on a power-on reset (that is, HARD_RST_b)

12.5.1 RapidIO Device Identity CAR

This register identifies the device and vendor information for the Tsi574.

Register name: RIO_DEV_ID Reset value: 0x0574_000D	Register offset: 00000

Bits	0	1	2	3	4	5	6	7
00:7	DEV_ID							
07:15	DEV_ID							
16:23	DEV_VEN_ID							
24:31	DEV_VEN_ID							

Bits	Name	Description	Type	Reset Value
0:15	DEV_ID	Device Identifier This field contains the IDT-assigned part number of the device.	R	0x0574
16:31	DEV_VEN_ID	Device Vendor Identifier This field identifies IDT as the vendor that manufactured the device. This value is assigned by the RapidIO Trade Association.	R	0x000D

12.5.2 RapidIO Device Information CAR

The SILICON_REV and METAL_REV fields in this register identify the device.

Register name: RIO_DEV_INFO **Reset value: 0x0000_0020**	**Register offset: 00004**

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

Bits	**Name**	**Description**	**Type**	**Reset Value**
0:23	Reserved	N/A	R	0
24:27	SILICON_REV	Indicates the version of silicon used in the device. This value may change with different silicon revisions of the device.	R	0b0010
28:31	METAL_REV	Indicates the version of the metal layers for the given silicon version. This value may change with different metal revisions of the device.	R	0b0000

12.5.3 RapidIO Assembly Identity CAR

This register contains assembly identification information about the Tsi574.

Register name: RIO_ASBLY_ID Reset value: 0x0001_000D	Register offset: 00008

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

Bits	Name	Description	Type	Reset Value
0:15	ASBLY_ID	Assembly ID. Identifies the type of assembly from the vendor specified by the ASBLY_VEN_ID field. I^2C load from EEPROM	R	0x0001
16:31	ASBLY_VEN_ID	Assembly Vendor ID Identifies the vendor that manufactured the assembly or subsystem that contains the device. I^2C load from EEPROM	R	0x000D

12.5.4 RapidIO Assembly Information CAR

This register contains additional information about the assembly.

Register name: RIO_ASBLY_INFO Reset value: 0x0000_0100	Register offset: 0000C

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

Bits	Name	Description	Type	Reset Value
0:15	ASBLY_REV	Assembly Revision Level I^2C load from EEPROM.	R	0x0000
16:31	EXT_FEAT_PTR	Extended Features Pointer This is the pointer to the first entry in the extended features list. In the Tsi574 it points to the Serial Physical Layer (see “RapidIO Physical Layer Registers” on page 261).	R	0x0100

12.5.5 RapidIO Processing Element Features CAR

This register identifies the major functionality provided by the processing element.

Register name: RIO_PE_FEAT Reset value: 0x1000_051F	Register offset: 00010

Bits	0	1	2	3	4	5	6	7
00:07	BRDG	MEM	PROC	SW	Reserved			
08:15	Reserved							
16:23	Reserved					MC	Reserved	SBR
24:31	Reserved			CTLS	EXT_FEA	EXT_AS		

Bits	Name	Description	Type	Reset Value
0	BRDG	Bridge 0 = Processing element is not a bridge 1 = Processing element can bridge to another interface	R	0
1	MEM	Endpoint 0 = Not a RapidIO endpoint addressable for reads and writes 1 = The processing element has physically addressable local address space and can be accessed as an endpoint through non-maintenance (that is, NREAD and NWRITE) transactions.	R	0
2	PROC	Processor 0 = Not a processor 1 = Physically contains a local processor or similar device that executes code. A device that bridges to an interface that connects to a processor does not count (see bit 0).	R	0
3	SW	Switching Capabilities PE can bridge to another external RapidIO Interface. For example, a device with two RapidIO ports and a local endpoint is a two port switch, not a three port switch, regardless of the internal architecture. 0 = Processing element is not a switch 1 = Processing element is a switch. Ftype 8 packets with hop count equal to 0 are routed to the register bus.	R	1
4:20	Reserved	N/A	R	0
21	MC	Multicast 0 = Does not support the multicast extensions 1 = Supports the multicast extensions	R	1

(Continued)

Bits	Name	Description	Type	Reset Value
22	Reserved	N/A	R	0
23	SBR	System bringup register extension 0 = System bringup register extension is not supported 1 = System bringup register extension is supported	R	1
24:26	Reserved	N/A	R	0
27	CTLS	For the Tsi574, packets are forwarded according to the configuration of the ingress port's lookup table. This bit is not used in the control of any functionality in the Tsi574. 0 = Device supports 8-bit destination IDs only 1 = Device supports 8-bit and 16-bit destination IDs	R	1
28	EXT_FEA	Extended Features Pointer is valid Pointer to the first entry in the extended features list. In the Tsi574 this pointer points to the "RapidIO Physical Layer Registers" on page 261	R	1
29:31	EXT_AS	Extended Addressing Support. 001 = 34-bit addresses 011 = 50- and 34-bit addresses 101 = 66- and 34-bit addresses 111 = 66-, 50-, and 34-bit addresses	R	0b111

12.5.6 RapidIO Switch Port Information CAR

This register defines the switching capabilities of a processing element.

Register name: RIO_SW_PORT Reset value: Undefined	Register offset: 00014

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

Bits	Name	Description	Type	Reset Value
0:15	Reserved	N/A	R	0x0000
16:23	PORT_TOTAL	Port Total The total number of RapidIO ports on the device. Note that the number of ports reported in this register assumes that all RapidIO ports are in 1x mode. For example, when a port is configured for 4x mode it consumes two ports from this reported number.	R	0x08
24:31	PORT_NUM	Port Number The port number that received the maintenance read packet that caused this register to be read. This value is undefined if the register is read through JTAG.	R	Undefined

12.5.7 RapidIO Source Operation CAR

This register defines the set of RapidIO I/O logical operations that can be issued by the Tsi574. The device can generate I/O logical maintenance read and write requests if it is required to access CARs and CSRs in other processing elements. The Tsi574 can route any packet.

Register name: RIO_SRC_OP **Reset value: 0x0000_0004**	**Register offset: 00018**

Bits	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
00:07	Reserved							
08:15	Reserved						IMPLEMENT_DEF	
16:23	READ	WRITE	STRM_WR	WR_RES	D_MSG	DBELL	Reserved	A_TSWAP
24:31	A_INC	A_DEC	A_SET	A_CLEAR	Reserved	PORT_WR	Reserved	

Bits	**Name**	**Description**	**Type**	**Reset Value**
0:13	Reserved	N/A	R	0
14:15	IMPLEMENT_DEF	Implementation defined	R	0
16	READ	Read operation supported	R	0
17	WRITE	Write operation supported	R	0
18	STRM_WR	Streaming write operation supported	R	0
19	WR_RES	Write-with-response operation supported	R	0
20	D_MSG	Data messaging	R	0
21	DBELL	Doorbell	R	0
22	Reserved	N/A	R	0
23	A_TSWAP	Atomic (test-and-swap) operation supported	R	0
24	A_INC	Atomic (increment) operation supported	R	0
25	A_DEC	Atomic (decrement) operation supported	R	0
26	A_SET	Atomic (set) operation supported	R	0
27	A_CLEAR	Atomic (clear) operation supported	R	0
28	Reserved	N/A	R	0

(Continued)

Bits	Name	Description	Type	Reset Value
29	PORT_WR	Port-write operation The RapidIO ports support port-write generation to report errors.	R	1
30:31	Reserved	Implementation defined	R	0

12.5.8 RapidIO Switch Multicast Support CAR

This register identifies the multicast programming model supported by a switch. The Tsi574 does not support the simple programming model (for more information, see the “RapidIO Multicast Mask Configuration Register” on page 256).

Register name: RIO_PE_MC_FEAT **Reset value: 0x0000_0000**	**Register offset: 00030**

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

Bits	Name	Description	Type	Reset Value
0	SIMP	Simple Programming 0 = Does not support simple block programming model 1 = Supports simple block programming model (not implemented)	R	0
1:31	Reserved	N/A	R	0

12.5.9 RapidIO Route LUT Size CAR

This register tells host software that the Tsi574 supports 512 destination IDs in its lookup table (LUT). When the LUT_512 bit in the “RapidIO Port x Mode CSR” on page 302 is set to 0, the corresponding switch port supports 64K destination IDs with limited capabilities

Register name: RIO_LUT_SIZE **Reset value: 0x0000_01FF**	**Register offset: 0034**

Bits	0	1	2	3	4	5	6	7
00:07	Reserved							
08:15	Reserved							
16:23	LUT_SIZE[0:7]							
24:31	LUT_SIZE[8:15]							

Bits	Name	Description	Type	Reset Value
0:15	Reserved	Reserved	R	0
16:31	LUT_SIZE	Lookup Table Size Identifies the destination IDs that can be used to route packets through the switch. Destination IDs 0x0000 to LUT_SIZE are valid. Note: When the LUT_512 bit in “RapidIO Port x Mode CSR” on page 302 is set to 0, destination IDs 0x0000 to 0xFFFF are valid on the corresponding port, regardless of the LUT_SIZE value.	R	0x01FF

12.5.10 RapidIO Switch Multicast Information CAR

This RapidIO standard register gives information about the multicast programming model, the number of multicast destination IDs supported, and the number of multicast masks supported.

Register name: RIO_SW_MC_INFO **Reset value: 0x0000_0008**	**Register offset: 00038**

Bits	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
00:07	ASSOC_ MODE	ASSOC_ SCOPE	MAX_DESTID_ASSOC					
08:15	MAX_DESTID_ASSOC							
16:23	MAX_MASKS							
24:31	MAX_MASKS							

Bits	**Name**	**Description**	**Type**	**Reset Value**
0	ASSOC_MODE	Defines the capabilities of a switch to associate destination IDs with multicast masks. 0 = Single associate. One “associate” write is required per association between a destinationID and a multicast mask. 1 = Block association is supported (Not implemented in the Tsi574)	R	0
1	ASSOC_SCOPE	Defines the capabilities of a switch to associate a destination ID with a multicast mask on a per-inbound-port basis. 0 = A destination ID, when associated with a multicast mask, associates with the mask regardless of which switch inbound port received the packet. 1 = Per-port association is supported (Not implemented in the Tsi574)	R	0
2:15	MAX_DESTID_ ASSOC	This is the maximum number of destinationIDs that can be associated with any multicast mask. The values are one less than the maximum number of associations. For example, 0x0000 - Maximum of one destinationID per multicast mask.	R	0
16:31	MAX_MASKS	Defines the number of multicast masks available. Multicast masks are sequentially numbered, with the first multicast mask number being 0x0000. Additional multicast masks are sequentially numbered.	R	8

12.5.11 RapidIO Host Base Device ID Lock CSR

The host base device ID lock CSR contains the base device ID value for the processing element in the system that is responsible for initializing this processing element.

The HOST_BASE_ID field is a write-once/reset field. Once the HOST_BASE_ID field is written, all subsequent writes to the field are ignored, except when the value written matches the value in the field. In this case, the register is re-initialized to 0xFFFF. Note that writing 0xFFFF to this register does not result in a lock being obtained.

After writing the HOST_BASE_ID field, a processing element must read the Host Base Device ID Lock CSR to verify that it owns the lock before attempting to initialize this processing element.

Register name: RIO_HOST_BASE_ID_LOCK **Reset value: 0x0000_FFFF**	**Register offset: 00068**

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

Bits	**Name**	**Description**	**Type**	**Reset Value**
0:15	Reserved	Reserved	R	0x0000
16:31	HOST_BASE_ID	Host Base ID Base Device ID for the processing element that is initializing this processing element.	R/W	0xFFFF

The HOST_BASE_ID set in this register does not enforce exclusive access to the device. It coordinates device identification during initialization and discovery.

12.5.12 RapidIO Component Tag CSR

This register is written by software. It is used for labeling and identifying the port-write transactions to the host.

Register name: RIO_COMP_TAG Reset value: 0x0000_0000	Register offset: 0006C

Bits	0	1	2	3	4	5	6	7
00:07	CTAG[0:7]							
08:15	CTAG[8:15]							
16:23	CTAG[16:23]							
24:31	CTAG[24:31]							

Bits	Name	Description	Type	Reset Value
00:31	CTAG	Component Tag	R/W	0

12.5.13 RapidIO Route Configuration DestID CSR

This register and “RapidIO Route Configuration Output Port CSR” on page 254 operate together to provide indirect read and write access to the destination ID lookup tables (LUTs).

Writes to the LUTs through these registers affect the LUTs of all ports on the device. Reads from these registers always return the data from Port 0.

This register set is identical to “RapidIO Port x Route Config DestID CSR” on page 306 (Offset 10070) and “RapidIO Port x Route Config Output Port CSR” on page 307 (Offset 10074), except that “RapidIO Port x Route Config Output Port CSR” on page 307 are per-port configuration registers and they include an auto-increment bit to increment the contents of SPx_ROUTE_CFG_DESTID after a read or write operation.

For details on how to configure the LUTs using this register, refer to “Lookup Tables” on page 37.

Register name: RIO_ROUTE_CFG_DESTID **Reset value: 0x0000_0000**	**Register offset: 0070**

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

Bits	**Name**	**Description**	**Type**	**Reset Value**
0:15	Reserved	Reserved	R	0
16:23	LRG_CFG_DEST_ID	Large Configuration Destination ID This field specifies the most significant byte of the destination ID used to select an entry in the LUT, when the “RapidIO Port x Route Config Output Port CSR” on page 307 is read or written.	R/W	0x00
24:31	CFG_DEST_ID	Configuration Destination ID This field specifies the destination ID used to select an entry in the LUT when the “RapidIO Port x Route Config Output Port CSR” on page 307 register is read or written.	R/W	0x00

12.5.14 RapidIO Route Configuration Output Port CSR

This register and “RapidIO Route Configuration DestID CSR” on page 253 operate together to provide indirect read and write access to the LUTs.

Writes to the LUTs through these registers affect the LUTs of all ports on the device. Reads from these registers always return the data from Port 0.

This register set is identical to “RapidIO Port x Route Config DestID CSR” on page 306 (Offset 10070) and “RapidIO Port x Route Config Output Port CSR” on page 307 (Offset 10074), except that “RapidIO Port x Route Config Output Port CSR” on page 307 are per-port configuration registers and they include an auto-increment bit to increment the contents of SPx_ROUTE_CFG_DESTID after a read or write operation.

For details on how to configure the LUTs using this register, refer to “Lookup Tables” on page 37.

Register name: RIO_ROUTE_CFG_PORT **Reset value: Undefined**	**Register offset: 0074**

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

Bits	**Name**	**Description**	**Type**	**Reset Value**
0:23	Reserved	Reserved	R	0
24:31	PORT	Port This is the RapidIO output port through which all reads and writes meant for the CFG_DEST_ID field in the “RapidIO Route Configuration DestID CSR” on page 253 are sent. Writing a value greater to this field than the PORT_TOTAL field in the “RapidIO Switch Port Information CAR” on page 245 sets the LUT entry to an unmapped state. For future compatibility, write the value 0xFF to indicate an unmapped destination ID. When reading an unmapped value from the LUT, this field is set to 0xFF.	R/W	Undefined

12.5.15 RapidIO Route LUT Attributes (Default Port) CSR

This register provides a default route for packets that do not match a valid entry in the destination ID lookup table (LUT). By default, the default route is unmapped and packets that attempt to use the default route are discarded.

Register name: RIO_LUT_ATTR **Reset value: 0x0000_00FF**	**Register offset: 0078**

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

Bits	**Name**	**Description**	**Type**	**Reset Value**
0:23	Reserved	Reserved	R	0
24:31	DEFAULT_ PORT	Default Output Port All transactions with destination IDs not defined in the LUT are routed to this pre-defined default output port. DEFAULT_PORT can be set to “unmapped” with a value greater than PORT_TOTAL in the RIO_SW_PORT register. For compatibility with future devices, it is recommended that the value 0xFF be used to indicate unmapped. If a packet needs to consult the default route and the default route is unmapped, the packet is discarded. See “Port Numbering” on page 67 for a mapping of port numbers to physical ports.	R/W	0xFF

12.5.16 RapidIO Multicast Mask Configuration Register

This register is used to add and remove egress port numbers to multicast masks. This can be completed either before or after a mask is bound to a destination ID and placed in the multicast group table. This register can also be used to retrieve the current configuration of a multimask mask.

Register name: RIO_MC_MASK_CFG **Reset value: 0x0000_0000**	**Register offset: 00080**

Bits	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
00:07	MC_MASK_NUM[15:8]							
08:15	MC_MASK_NUM[7:0]							
16:23	EG_PORT_NUM[7:0]							
24:31	Reserved	MASK_CMD			Reserved			PORT_ PRESENT

Bits	**Name**	**Description**	**Type**	**Reset Value**
0:15	MC_MASK_NUM	Specifies the multicast mask [0:7] which is to be modified when this register is written with the MASK_CMD field set to “Add” or “Delete“(0x0000 to 0x0007). Specifies the Multicast mask which is to be queried for the presence of a port (by a subsequent read of this register) when this register is written with a “Write_to_Verify“command. When the register is read, this field returns the contents that were previously written. Note: Tsi574 supports four Multicast Masks. The MSB should always read 0.	R/W	0
16:23	EG_PORT_NUM	When this register is written, it specifies the port number to be added or deleted from the Multicast_Mask. This field is ignored when the MASK_CMD field indicates “Delete All Ports” or “Add All Ports”. When the register is read, this field returns the contents that were previously written. Note: Tsi574 only supports 8 ports. The MSB should always read 0	R/W	0
24	Reserved	N/A	R	0

(Continued)

Bits	Name	Description	Type	Reset Value
25:27	MASK_CMD	Specifies the mask action on a write. Contains the last value written for read operations. • 000 = Write_to_Verify. This write is only to set up a Multicast_Mask and Egress_Port_Number for a subsequent read of this register to check the Port_Present bit. No bits are changed in any multicast mask. • 001 = Add the given Egress_Port_Number to the specified Multicast Mask. • 010 = Delete the given Egress_Port_Number from the specified Multicast Mask. • 011 = Reserved • 100 = Delete all egress ports from the specified Multicast Mask. • 101 = Add all egress ports to the specified Multicast Mask. • 110 = Reserved • 111 = Reserved When the register is read, this field returns the contents that were previously written.	R/W	0
28:30	Reserved	N/A	R	0
31	PORT_ PRESENT	Port Present 0 = Port is not enabled as an outbound port in the specified multicast mask. 1 = Port is enabled as an outbound port in the specified multicast mask. The Multicast_Mask and Egress_Port_Number were specified in a prior write to this register using the “Write_to_Verify” command.	R	0

12.5.17 RapidIO Multicast DestID Configuration Register

This register is used to configure the multicast group table. It contains the association between a destination ID and a multicast mask number. The association is formed or removed only when the "RapidIO Multicast DestID Association Register" on page 259 register is written.

This register associates only one destination ID to one mask. Associating ranges of destination IDs to ranges of masks is not supported.

Register name: RIO_MC_DESTID_CFG **Reset value: 0x0000_0000**	**Register offset: 00084**

Bits	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
00:07	DESTID_BASE_LT[7:0]							
08:15	DESTID_BASE[7:0]							
16:23	MASK_NUM_BASE[15:8]							
24:31	MASK_NUM_BASE[7:0]							

Bits	**Name**	**Description**	**Type**	**Reset Value**
0:7	DESTID_BASE_LT	The most significant 8 bits of a 16-bit destination ID to be associated with the mask MASK_NUM_BASE. This field is ignored when an 8-bit destination ID is being associated with a multicast mask.	R/W	0
8:15	DESTID_BASE	The 8-bit destination ID or the least significant 8 bits of a 16-bit destination ID to be associated with the mask MASK_NUM_BASE.	R/W	0
16:31	MASK_NUM_BASE	The multicast mask number [0:7] to be associated with the destination ID configured above (0x0000 to 0x0007). Note: Tsi574 only supports four Multicast Masks. The MSB should always read 0.	R/W	0

12.5.18 RapidIO Multicast DestID Association Register

This register populates and depopulates the multicast group table. When this register is written, the device consults the value in the "RapidIO Multicast DestID Configuration Register" on page 258 register to determine which destination ID is associated with which multicast mask (or which association must be removed).

Register name: RIO_MC_DESTID_ASSOC **Reset value: 0x0000_0000**	**Register offset: 00088**

Bits	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
00:07	ASSOC_BLK_SIZE							
08:15	ASSOC_BLK_SIZE							
16:23	INGRESS_PORT							
24:31	LARGE	CMD		RESERVED				ASSOC_ PRESENT

Bits	**Name**	**Description**	**Type**	**Reset Value**
0:15	ASSOC_BLK_ SIZE	This field is ignored. Set this field to 0 when writing the register for future software compatibility.	R	0
16:23	INGRESS_ PORT	This field is ignored.	R	0
24	LARGE	0 = The association is for a small transport destination IDs 1 = The association is for a large transport destination IDs This field returns the previously written value when this register is read.	R/W	0

(Continued)

Bits	Name	Description	Type	Reset Value
25:26	CMD	Command • 00 = Verify Association This read checks the association status, for a given transport type, between a given destination ID and a multicast mask number. The transport type is specified by the LARGE field. The multicast mask number and destination ID are specified by a prior write to the "RapidIO Multicast DestID Configuration Register". • 01 = Reserved • 10 = Remove Associations This register write removes the associations between a destination ID and multicast mask number. The multicast mask number and destination ID are specified by a prior write to the "RapidIO Multicast DestID Configuration Register". • 11 = Add Associations This register write adds associations between a destination ID and multicast mask number. The multicast mask number and destination ID are specified by a prior write to the "RapidIO Multicast DestID Configuration Register". • This field returns the previously written value when this register is read.	R/W	0
27:30	Reserved	Reserved	R	0000
31	ASSOC_ PRESENT	Association Present When read, this bit reflects the association status between a destination ID, given a transport type, and a multicast mask number. The destination ID and multicast mask number are specified by a previous write to the "RapidIO Multicast DestID Configuration Register". The transport type is specified by a previous write to this register with a command of "Write_To_Verify" with the LARGE field set accordingly. 0 = No association present 1 = Association present This bit is reserved on a write to this register.	R/W	0

12.6 RapidIO Physical Layer Registers

This section describes the Command and Status Register (CSR) set. All registers in the set are 32-bits long and aligned to a 32-bit boundary. These registers allow an external processing element to determine the capabilities, configuration, and status of a processing element using the Serial physical layer. The registers can be accessed using the maintenance operations defined in the *RapidIO Interconnect Specification (Revision 1.3)*.

When an individual port is powered down, the RapidIO Physical Layer Registers for that port are read only and return undetermined values.

These registers are reset by the HARD_RST_b reset input signal, as well as when the Tsi574 performs a self-reset. The registers within a port are also reset by a "Port Reset". For more information on Tsi574 reset implementation and behavior, see "Clocks, Resets and Power-up Options" on page 203. It is possible to override reset values of writable fields, and some read-only fields, using the I^2C register loading capability on boot. Refer to "I^2C Interface" on page 139 for more information on the use of I^2C controller register loading capability.

The I^2C register loading capability is only used on a power-on reset (that is, HARD_RST_b)

Reads to reserved register addresses return 0, writes to reserved register addresses complete without error.

The following table shows the register offsets of the physical layer of the Tsi574. When the even-numbered port in a Tsi574 is configured to operate in 4x mode, the odd-numbered port cannot be used and the register values for the odd-numbered port should be ignored

Table 37: Physical Interface Register Offsets

RapidIO Port x Registers		
Port	**Offset**	**Description**
All	0x0100	These registers affect the operation of all ports.
0	0x0140	1x/4x serial port
1	0x0160	1x serial port
2	0x0180	1x/4x serial port
3	0x01A0	1x serial port
4	0x01C0	1x/4x serial port
5	0x01E0	1x serial port
6	0x0200	1x/4x serial port
7	0x0220	1x serial port

12.6.1 RapidIO 1x or 4x Switch Port Maintenance Block Header

This register contains the block header information.

Register name: RIO_SW_MB_HEAD Reset value: 0x1000_0009	Register offset: 100

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

Bits	Name	Description	Type	Reset Value
0:15	EF_PTR	Extended Features Pointer Hard-wired pointer to the next block in the features data structure.	R	0x1000
16:31	EF_ID	Hard-wired Extended Features ID 0x0009 = Switch with software recovery capability	R	0x0009

12.6.2 RapidIO Switch Port Link Timeout Control CSR

This register contains the timeout timer value for all ports on a device. This timeout is for link events such as sending a packet and receiving the corresponding acknowledge, or sending a link-request and receiving the corresponding link-response. The reset value is the maximum timeout interval, and is 5.4 seconds with a P_CLK of 100 MHz. When Link Time Out is expired the port enters the Output-Error state, as outlined in the *RapidIO Interconnect Specification (Revision 1.3)*.

Register name: RIO_SW_LT_CTL **Reset value: 0xFFFF_FF00**	**Register offset: 120**

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

Bits	**Name**	**Description**	**Type**	**Reset Value**
0:23	TVAL	Timeout Interval Value Timeout = (32/F) * TVAL, where F is the register bus frequency, P_CLK (recommended operation is 100 MHz). When F = 100 MHz, the default value of this register gives a time out of 5.4 seconds. When TVAL is 0, the timer is disabled. Note: Refer to “P_CLK Programming” on page 479 for information on programming options for the P_CLK frequency.	R/W	0xFFFFFF
24:31	Reserved	N/A	R	0

12.6.3 RapidIO Switch Port General Control CSR

This register applies to all ports on the device. A device has only one copy of the bits in this register. These bits are also accessible through the Port General Control CSR of any other physical layer implemented on a device.

Register name: RIO_SW_GEN_CTL **Reset value: 0x0000_0000**	**Register offset: 13C**

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

Bits	**Name**	**Description**	**Type**	**Reset Value**
0:1	Reserved	N/A	R	0
2	DISC	Discovered This device has been located by the processing element responsible for system configuration. 1 = Device discovered by system host 0 = Device not discovered	R/W	0
3:31	Reserved	N/A	R	0

12.6.4 RapidIO Serial Port x Link Maintenance Request CSR

According to the *RapidIO Interconnect Specification (Revision 1.3)* only one link maintenance request can be outstanding at a time. However, the Tsi574 can be instructed to produce four consecutive link maintenance requests in order to quickly re-establish a link.

Multiple link maintenance request symbols are generated by the CMD field in the RapidIO Serial Port x Link Maintenance Request CSR. An external device can write to this register and generate a link-request control symbol on the corresponding RapidIO port. A read to this register returns the last value written.

The control symbols generated by this register are encoded as: Status+ackID_status+buf_status + link_req+CMD.

If the Tsi574 sends its own link maintenance request, and if that request is outstanding and the CMD field is written to, then the register write is ignored. If this register is written twice in rapid succession, it could cause a protocol violation.

If the RapidIO Serial Port x Link Maintenance Response CSR does not indicate that the link-request is complete, software must ensure that a period of time equal to the Port Link Timeout period, controlled by the RapidIO Switch Port Link Time Out Control CSR, has passed before attempting another link maintenance request to avoid protocol violations.

Register name: SP{0..7}_LM_REQ **Reset value: 0x0000_0000**	**Register offset: 140, 160, 180, 1A0, 1C0, 1E0, 200, 220**

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

Bits	**Name**	**Description**	**Type**	**Reset Value**
0:28	Reserved	N/A	R	0
29:31	CMD	Command Command to be sent in the link-request control symbol. If read, this field returns the last written value. 011 = Reset. Writing this value causes the device to send four consecutive reset control symbols. 100 = Input-status Other values are reserved, but the Tsi574 sends a control symbol with the requested value.	R/W	0

Writing to this register on a port in normal operation affects traffic on that port. This register should only be used on ports in an error state.

12.6.5 RapidIO Serial Port x Link Maintenance Response CSR

This register is accessed by an external RapidIO device. A read of this register returns the status from the last link-response received from the link-partner due to a link-request/input-status issued using the RapidIO Serial Port x Link Maintenance Request CSR CSR or the RapidIO Port x Control Symbol Transmit register.

Register name: SP{0..7}_LM_RESP **Reset value: 0x0000_0000**	**Register offset: 144, 164, 184, 1A4, 1C4, 1E4, 204, 224**

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

Bits	**Name**	**Description**	**Type**	**Reset Value**
0	RESP_VLD	Response Valid 0 = No link-response control symbol received or no link-request/reset transmitted 1 = If the link-request was a link-request/input-status, this bit indicates that the link-response control symbol has been received. The LINK_STAT field contains information pertaining to that link response. If link-request was a link-request/reset, this bit indicates that the link-request/reset has been transmitted. Note: For link-response control symbols, this bit certifies the availability of data, it does not certify the correctness of the data.	RC	0
1:21	Reserved	N/A	R	0
22:26	ACK_ID_STAT	AckID Status AckID status field from the link-response control symbol. The value of the next ackID expected by the receiver.	R	0
27:31	LINK_STAT	Link Status Link status field from the link-response control symbol. Other values are reserved. 0b00010 = Error 0b00100 = Retry-stopped 0b00101 = Error-stopped 0b10000 = OK	R	0

12.6.6 RapidIO Serial Port x Local ackID Status CSR

A read to this register returns the local ackID for both the inbound and outbound port of the device.

Register name: SP{0..7}_ACKID_STAT Reset value: 0x0000_0000	Register offset: 148, 168, 188, 1A8, 1C8, 1E8, 208, 228

Bits	0	1	2	3	4	5	6	7
00:07	Reserved			INBOUND				
08:15	Reserved							
16:23	Reserved			OUTSTANDING				
24:31	Reserved			OUTBOUND				

Bits	Name	Description	Type	Reset Value
0:2	Reserved	N/A	R/W	0
3:7	INBOUND	Inbound Acknowledge ID Next expected ackID value for the receive side of the port. Software can write this field to force re-transmission of outstanding unacknowledged packets, in order to manually implement error recovery. Note: The INBOUND value can be initialized through the I^2C Interface. Initializing the INBOUND value from I^2C is required for test purposes only. Unless the INBOUND value is initialized to 0, the device state is not consistent with the state required by the *RapidIO Specification*. It is not possible to exchange packets after a reset if the INBOUND value is other than 0. The INBOUND field may only be written when there are no packets outstanding in the transmit queue, and no packets are being exchanged with a link partner. Caution: Changing the INBOUND field when there are packets being exchanged with a link partner results in non-deterministic ackID values. It is likely that a fatal error due to ackID mismatch will result.	R/W	0
8:18	Reserved	N/A	R	0
19:23	OUTSTANDING	Outstanding Acknowledge IDs The first unacknowledged ackID. Note: When software writes to OUTBOUND field, the OUTSTANDING is also updated with the value in OUTBOUND.	R	0
24:26	Reserved	N/A	R	0

(Continued)

Bits	Name	Description	Type	Reset Value
27:31	OUTBOUND	Outbound Acknowledge ID Next ackID to be transmitted by the port. Software can write this field to force re-transmission of outstanding unacknowledged packets, in order to manually implement error recovery. If an error or retry occurs on the inbound while the write to this register is being processed, the OUTBOUND value is not used for the next packet transmitted. The new OUTBOUND value is always used when the port is operating normally, and when the port is in an error state. The OUTBOUND field may only be written when there are no packets outstanding in the transmit queue, and no packets are being exchanged with a link partner. Caution: Changing the OUTBOUND field when there are packets being exchanged with a link partner results in non-deterministic ackID values. It is likely that a fatal error due to ackID mismatch will result.	R/W	0

12.6.7 RapidIO Port x Error and Status CSR

This register contains the port error and status information. This register returns 0x0000001 if it is read when the port is powered down.

Register name: SP{0..7}_ERR_STATUS Reset value: 0x0000_0001	Register offset: 158, 178, 198, 1B8, 1D8, 1F8, 218, 238

Bits	0	1	2	3	4	5	6	7
00:07	Reserved					OUTPUT_DROP	OUTPUT_FAIL	OUTPUT_DEG
08:15	Reserved			OUTPUT_RE	OUTPUT_R	OUTPUT_RS	OUTPUT_ERR	OUTPUT_ERR_STOP
16:23	Reserved					INPUT_RS	INPUT_ERR	INPUT_ERR_STOP
24:31	Reserved			PORT_W_PEND	Reserved	PORT_ERR	PORT_OK	PORT_UNINIT

Bits	Name	Description	Type	Reset Value
0:4	Reserved	N/A	R	0
5	OUTPUT_DROP	Output port has discarded a packet. The packet is dropped when the Error Rate Threshold is reached, when the time-to-live counter has expired, or when a TEA or MAC_TEA error has occurred.	R/W1C	0
6	OUTPUT_FAIL	Output Failed Encountered Output port has encountered a failed condition, meaning that the failed port error threshold has been reached in the “RapidIO Port x Error Rate Threshold CSR” on page 298.	R/W1C	0
7	OUTPUT_DEG	Output Degraded Encountered Output port has encountered a degraded condition, meaning that the degraded port error threshold has been reached in “RapidIO Port x Error Rate CSR” on page 296.	R/W1C	0
8:10	Reserved	N/A	R	0
11	OUTPUT_RE	Output Retry-encountered Outbound port has encountered a retry condition.This bit is set when the Output Retry-stopped bit (bit 13) is set.	R/W1C	0

(Continued)

Bits	Name	Description	Type	Reset Value
12	OUTPUT_R	Output Retried Outbound port has received a packet-retry control symbol and cannot make forward progress. This bit is set when the Output Retry-stopped bit (bit 13) is set. This bit is cleared after receiving a packet-accepted or packet-not-accepted control symbol.	R	0
13	OUTPUT_RS	Output Retry-stopped Outbound port has received a packet-retry control symbol and is in the output retry-stopped state.	R	0
14	OUTPUT_ERR	Output Error-encountered Outbound port has encountered (and possibly recovered from) a transmission error. This bit is set when the Output Error-stopped bit (bit 15) is set.	R/W1C	0
15	OUTPUT_ERR_STOP	Output Error-stopped Outbound port is in the output error-stopped state.	R	0
16:20	Reserved	N/A	R	0
21	INPUT_RS	Input Retry-stopped Inbound port is in the input retry-stopped state.	R	0
22	INPUT_ERR	Input Error-encountered Inbound port has encountered (and possibly recovered from) a transmission error. This bit is set when bit 23, Input Error-stopped, is set.	R/W1C	0
23	INPUT_ERR_STOP	Input Error-stopped Inbound port is in the input error-stopped state.	R	0
24:26	Reserved	N/A	R	0
27	PORT_W_PEND	Port-write Pending Port has encountered a condition which required it to issue an I/O logical port-write maintenance request.	R/W1C	0
28	Reserved	N/A	R	0

(Continued)

Bits	Name	Description	Type	Reset Value
29	PORT_ERR	Port Error PORT_ERR on the even port is composed of a logical OR of PORT_ERR signal produced by the individual lanes independent of whether the MAC is configured in 4x mode or 1x mode. Therefore, when the MAC is in a x1 configuration, and an error occurs on the odd port, the PORT_ERR bit will be asserted in the Error and Status CSR of both the odd and even ports. The assertion of PORT_ERR in the even port's Error and Status CSR occurs independently of the state of the PORT_OK bit of the even port. Inbound or Outbound port has encountered an error from which the hardware was unable to recover (fatal error). The following fatal errors cause a PORT_ERR: • Four link-request tries with link-response, but no outstanding ackID • Four link-request tries with time-out error for link-response • Dead link timer is enabled (in the DLT_EN bit in the "SRIO MAC x Digital Loopback and Clock Selection Register" on page 369) and the timer expires. Refer to "Dead Link Timer" on page 64 for more information on this feature. • The Lane Sync Timer (LST) expires for at least one lane of a port (for more information, see "Lane Sync Timer").	R/W1C	0
30	PORT_OK	Port OK Inbound and Outbound ports are initialized and can communicate with the adjacent device. This bit and bit 31, Port Un-initialized, are mutually exclusive.	R	0
31	PORT_UNINIT	Port Un-initialized Inbound and Outbound ports are not initialized. This bit and bit 30, PORT_OK, are mutually exclusive. This bit is set to a 1 after reset.	R	1

12.6.8 RapidIO Serial Port x Control CSR

This register returns a default value when read in power down mode. This register returns 0x0000001 if it is read when the port is powered down.

Register name: SP{0..7}_CTL **Reset value: Undefined**	**Register offset: 15C, 17C, 19C, 1BC, 1DC, 1FC, 21C, 23C**

Bits	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
00:07	PORT_WIDTH		INIT_PWIDTH			OVER_PWIDTH		
08:15	PORT_DIS	OUTPUT_EN	INPUT_EN	ERR_DIS	MCS_EN	Reserved	ENUM_B	Reserved
16:23	Reserved							
24:31	Reserved				STOP_FAIL_EN	DROP_EN	PORT_LOCKOUT	PORT_TYPE

Bits	**Name**	**Description**	**Type**	**Reset Value**
0:1	PORT_WIDTH	Port Width This field displays the port mode after reset. • 00 = Single-lane port (the port is 1x mode only) • 01 = Four-lane port (the port has 1x/4x mode and can operate in 1x or 4x mode) PORT_WIDTH is defined by the SPx_MODESEL pin as follows: • If the SPx_MODESEL pin is high, PORT_WIDTH for SP*x* is 0 and PORT_WIDTH for SP*x+1* is 0. • If the SPx_MODESEL pin is low, PORT_WIDTH for SPx is 1 and PORT_WIDTH for SP*x+1* is 0. However, that port SP*x+1* cannot be used.	R	Undefined
2:4	INIT_PWIDTH	Initialized Port Width Current operating mode of the port. This bit is set by hardware when the initialization process is complete, and whenever the operating width of the port changes (that is, a 4x port degrades to a 1x port). • 000 = 1x port, lane 0 • 001 = 1x port, lane 2 • 010 = 4x port	R	0

(Continued)

Bits	Name	Description	Type	Reset Value
5:7	OVER_PWIDTH	Override Port Width Software port configuration that overrides the hardware size. This field is valid only if the PORT_WIDTH field is set to 01 and the port is operating in 4x mode. If a port that only has 1x mode functionality is placed in 4x mode, the odd numbered port becomes disabled. Can be configured as a power-up option (I^2C) or forced during normal mode of operation (forced re-initialization). • 000 = No override (stay in current operation mode, either 1x or 4x) • 001 = Reserved • 010 = Force single lane, lane 0 • 011 = Force single lane, lane 2 Other values are reserved. Re-initialization occurs on the port if the value of this register is changed. Re-initialization can be forced by the FORCE_REINIT field of the “RapidIO Port x Control Independent Register” on page 311 register. Note: Initial port width of the port is set by SPx_MODESEL pins at power-up. After reset release, the SPx_MODESEL pins are ignored and configuration is controlled by this register.	R/W	0
8	PORT_DIS	Port Disable 0 = Port receivers/drivers are enabled. 1 = Port receivers/drivers are disabled and are unable to receive/transmit to any packets or control symbols. When the port is disabled, there is no data flow to the output drivers. Transmit drivers of a disabled port transmits all zeros. Any data sent to this port sits in the Output Queue.	R/W	0
9	OUTPUT_EN	Output Port Transmit Enable 0 = Port is stopped and is not able to issue any packets. It can only route and respond to maintenance packets. 1 = Port is enabled to issue any packets.	R/W	1
10	INPUT_EN	Inbound Port Enable 0 = Inbound port is stopped and only routes or responds to maintenance requests. Other packets generate packet-not-accepted control symbols to force an error condition on the sending device. 1 = Inbound port responds to any packet.	R/W	1

(Continued)

Bits	Name	Description	Type	Reset Value
11	ERR_DIS	Error Checking Disable - physical layer CRC error only 0 = Enable error checking and recovery 1 = Disable error checking and recovery. Retransmission is suppressed for all packets. Caution: If error checking is disabled, corrupt maintenance packets may be accepted by the Tsi574. Even when error checking is disabled, a corrupt maintenance write request is ignored by the registers. If error checking is enabled, corrupt maintenance packets are not accepted.	R/W	0
12	MCS_EN	Multicast-event Participant 0 = Do not forward incoming Multicast Event control symbols out this port. 1 = Forward incoming Multicast Event control symbols out this port	R/W	0
13	Reserved	N/A	R	0
14	ENUM_B	Enumeration boundary bit, used in system discovery algorithms. This bit does not control any functionality within the Tsi574. The reset value of this bit is 1 for port 2. For all other ports, the reset value of this bit is 0.	R/W	Undefined
15:27	Reserved	N/A	R	0
28	STOP_FAIL_EN	Stop on Port Failed Encountered Enable This bit is used the DROP_EN bit to force the port to stop attempting to send packets to the connected device when the Error Rate Failed Threshold has been met or exceeded. For more information, see Table 39 on page 278.	R/W	0
29	DROP_EN	Drop Packet Enable This bit is used with the STOP_FAIL_EN bit to force the output port to drop packets that are acknowledged with a packet-not-accepted control symbol when the Error Rate Failed Threshold has been met or exceeded. For more information, see Table 39 on page 278.	R/W	0

(Continued)

Bits	Name	Description	Type	Reset Value
30	PORT_LOCKOUT	Port Lockout 0 = The packets that may be received and issued are controlled by the state of the Output Port Enable and Input Port Enable bits. 1= This port is stopped and is not enabled to issue or receive any packets. The input port can still send and respond to link-requests. All received packets return packet-not-accepted control symbols to force the sending device to signal an error condition. The receipt of a packet by the locked-out port causes the assertion of the INPUT_ERR and INPUT_ERR_STOP bits in its RapidIO Port x Error and Status CSR. The link partner indicates packet rejection by asserting its OUTPUT_ERR and OUTPUT_ERR_STOP bits. Setting the PORT_LOCKOUT bit also causes the port to drop all packets arriving from the ISF for transmission, and to flush any existing packets in the transmit and receive queues of the port. Packet ackIDs are not modified in the RapidIO Serial Port x Local ackID Status CSR.	R/W	0
31	PORT_TYPE	Port Type Indicates the port type 1 = Serial port	R	1

12.7 RapidIO Error Management Extension Registers

This section describes the registers in the Extended Features block (EF_ID = 0x0007), which is defined in Part VIII of the RapidIO specification. These registers enable an external processing element to manage the error status and reporting for a processing element.

Not all Error Management Extension registers are supported in the Tsi574.

These registers are reset by the HARD_RST_b reset input signal, as well as when the Tsi574 performs a self-reset. The registers within a port are also reset by a "Port Reset". For more information on Tsi574 reset implementation and behavior, see "Clocks, Resets and Power-up Options" on page 203. It is possible to override reset values of writable fields using the I^2C register loading capability on boot. Refer to "I^2C Interface" on page 139 for more information on the use of I^2C controller register loading capability.

When an individual port is powered down, the RapidIO Error Management Extension Registers are read only and return 0.

The Logical/Transport Error Detect registers are not required for a switch. However, a switch's register bus access errors and transport errors are reported per port in bit 0 of the "RapidIO Port x Error Detect CSR" on page 286. The port's capture registers contain error information.

Software must not write to reserved addresses, and reserved bits in the RapidIO Error Management Extension registers should be written with zero.

All registers are 32-bits and aligned to a 32-bit boundary

Table 38: Error Management Registers

Port	Offset	Description
All	0x1000	General Error Management capability registers
SP0	0x1040	1x/4x Serial port
SP1	0x1080	1x Serial port
SP2	0x10C0	1x/4x Serial port
SP3	0x1100	1x Serial port
SP4	0x1140	1x/4x Serial port
SP5	0x1180	1x Serial port
SP6	0x11C0	1x/4x Serial port
SP7	0x1200	1x Serial port

12.7.1 Port Behavior When Error Rate Failed Threshold is Reached

When the Error Rate Counter (ERR_RATE_CNT field) reaches the enabled Error Rate Failed Threshold (ERR_RFT field), the behavior of the port depends upon the value of the STOP_FAIL_EN bit and the DROP_EN bit in the “RapidIO Port x Error and Status CSR” on page 270. The required behavior is defined in Table 39.

Table 39: STOP_FAIL_EN and DROP_EN Setting

Bit Setting		Port Behavior
STOP_FAIL_EN **Stop on Port Failed Encountered Enable**	**DROP_EN** **Drop Packet Enable**	
0	0	The port continues to attempt to transmit packets to the connected device if the Output Failed Encountered bit is set and/or if the Error Rate Failed threshold has been met or exceeded.
0	1	The port discards packets that receive a Packet-not-accepted control symbol when the Error Rate Failed Threshold has been met or exceeded. Upon discarding a packet, the port sets the Output Packet-dropped bit in the Port *x* Error and Status CSR. If the output port “heals”, the Error Rate Counter falls below the Error Rate Failed Threshold and the output port continues to forward all packets.
1	0	The port stops attempting to send packets to the connected device when the Output Failed Encountered bit is set. The output port becomes congested.
1	1	The port discards all output packets without attempting to send when the port’s Output Failed Encountered bit is set. Upon discarding a packet, the port sets the Output Packet-dropped bit in the “RapidIO Port x Error and Status CSR” on page 270. Caution: When both bits are set, only the stored packets in the queue are discarded. Any packets sent after are forwarded.

12.7.2 RapidIO Error Reporting Block Header

The error reporting block header indicates the start of the Error Management Extensions registers in the Tsi574.

Register name: RIO_ERR_RPT_BH Reset value: 0x0000_0007	Register offset: 1000

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

Bits	Name	Description	Type	Reset Value
0:15	EF_PTR	Extended Features Pointer Hard wired pointer to the next block in the data structure. 0000 = Last extended feature block	R	0x0000
16:31	EF_ID	Hard-wired Extended Features ID 0x0007 = EF ID for error management capability	R	0x0007

12.7.3 RapidIO Logical and Transport Layer Error Detect CSR

This register indicates the error that was detected by the Logical or Transport logic layer. Multiple bits can be set in the register if simultaneous errors are detected during the same clock cycle errors are logged.

Note that for switches the errors detected are limited to maintenance packets (maintenance requests, maintenance responses, and port writes) with a hop count of 0. No other packets reach the logical layer of a switch

Register name: RIO_LOG_ERR_DET **Reset value: 0x0000_0000**	**Register offset:1008**

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

Bits	**Name**	**Description**	**Type**	**Reset Value**
0:3	Reserved	N/A	R	0
4	L_ILL_TRANS	Illegal Transaction Bit is set when a terminating maintenance (Type 8, hopcount = 0) request transaction was received with one or more of the following conditions: • TTYPE = 0b0101 - 0b1111 • TT code = 0b11/0b10	R/W0C	0
5:7	Reserved	N/A	R	0
8	L_ILL_RESP	Illegal Response A maintenance response was received with a hop count of 0.	R/W0C	0
9	L_UNSUP_TRANS	Unsupported Transaction A port-write transaction was received with a hop count of 0.	R/W0C	0
10:31	Reserved	N/A	R	0

12.7.4 RapidIO Logical and Transport Layer Error Enable CSR

This register contains the bits that control if an error condition locks the Logical/Transport Layer Error Detect and Capture registers, and is reported to the system host through an interrupt and/or a port-write.

For switches, the errors detected are limited to maintenance packets (maintenance requests, maintenance responses, and port writes) with a hop count of 0. Once enabled, port-writes and interrupts can be generated for these sources. No other packets reach the logical layer of a switch.

Register name: RIO_LOG_ERR_DET_EN **Reset value: 0x0000_0000**	**Register offset:100C**

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

Bits	**Name**	**Description**[a]	**Type**	**Reset Value**
0:3	Reserved	N/A	R	0
4	ILL_TRANS_EN	Illegal Transaction Decode Enable 0 = disable L_ILL_TRANS 1 = enable L_ILL_TRANS	R/W	0
5:7	Reserved	N/A	R	0
8	ILL_RESP_EN	Illegal Response Enable 0 = disable L_ILL_RESP 1 = enable L_ILL_RESP	R/W	0
9	UNSUP_TRANS_EN	Unsupported Transaction Enable 0 = disable L_UNSUP_TRANS 1 = enable L_UNSUP_TRANS	R/W	0
10:31	Reserved	N/A	R	0

a. All bits in this register enable bits in “RapidIO Logical and Transport Layer Error Detect CSR” on page 280.

12.7.5 RapidIO Logical and Transport Layer Address Capture CSR

This register contains error information. It is locked when a Logical/Transport error is detected and the corresponding enable bit is set.

For switches, the errors detected are limited to maintenance packets (maintenance requests, maintenance responses, and port writes) with a hop count of 0. No other packets reach the logical layer of a switch. Therefore, the only time this register contains valid information is for maintenance requests which have incorrect field values. Only invalid data can be captured in this register for erroneous port-writes and maintenance responses, as these transactions reserve the address field. If the TT code for an erroneous maintenance request is invalid, this register captures the address of the invalid data.

Refer to "RapidIO Logical and Transport Layer Device ID Capture CSR" on page 283 for the source ID where the error originated.

Note that this register is not updated when a correctly formatted maintenance request fails.

Register name: RIO_LOG_ERR_ADDR **Reset value: 0x0000_0000**	**Register offset:1014**

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

Bits	**Name**	**Description**	**Type**	**Reset Value**
0:7	Reserved	N/A	R	0
8:28	ADDRESS	Address of the illegal maintenance request received.	R/W	0
29	Reserved	N/A	R	0
30	WDPTR	Word pointer from the illegal maintenance request received	R/W	0
31	Reserved	N/A	R	0

12.7.6 RapidIO Logical and Transport Layer Device ID Capture CSR

This register contains error information, specifically the device ID field values for failed transactions. It is locked when a Logical/Transport error is detected and the corresponding enable bit is set. When the TT field of the erroneous message is not a defined value, the contents of this register are bytes 3 and 4 of the packet received.

For switches, the errors detected are limited to maintenance packets (maintenance requests, maintenance responses, and port writes) with a hop count of 0. No other packets can reach the logical layer of a switch.

Note that this register is not updated when a correctly formatted maintenance request fails.

Register name: RIO_LOG_ERR_DEVID **Reset value: 0x0000_0000**	**Register offset:1018**

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

Bits	**Name**	**Description**	**Type**	**Reset Value**
0:15	Reserved	N/A	R	0
16:23	SRCID_MSB	Source ID Most Significant Byte Most significant byte of the source ID associated with the error (large transport systems only)	R/W	0
24:31	SRCID	Source ID The sourceID associated with the error.	R/W	0

12.7.7 RapidIO Logical and Transport Layer Control Capture CSR

This register contains error information, specifically the message type and subtype field values for failed transactions. It is locked when a Logical/Transport error is detected and the corresponding enable bit is set.

The Ftype value should always be eight for a maintenance packet. The TTYPE value always reflects the four bits following the FTYPE field in the packet. Note that for switches, the errors detected are limited to maintenance packets (maintenance requests, maintenance responses, and port writes) with a hop count of 0. No other packets reach the logical layer of a switch.

Note that this register is not updated when a correctly formatted maintenance request fails.

Register name: RIO_LOG_ERR_CTRL_INFO **Reset value: 0x0000_0000**	**Register offset:101C**

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

Bits	**Name**	**Description**	**Type**	**Reset Value**
0:3	FTYPE	Format type associated with the error	R/W	0
4:7	TTYPE	Transaction type associated with the error	R/W	0
8:31	Reserved	N/A	R	0

A value of 0x00000000 must be written to this register to clear it.

12.7.8 RapidIO Port-Write Target Device ID CSR

This register contains the target device ID to be used when the switch generates a maintenance port-write operation to report errors to a system host. Port-write packets are routed to the output port defined by the routing LUT of the switch.

Register name: RIO_PW_DESTID **Reset value: 0x0000_0000**	**Register offset: 1028**

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

Bits	**Name**	**Description**	**Type**	**Reset Value**
0:7	DESTID_MSB	Most Significant Byte of port-write Target Device ID. This field is used only when LARGE_DESTID is 1.	R/W	0
8:15	DESTID_LSB	If LARGE_DESTID is 0, the DESTID_LSB field is the 8-bit DESTID used in locally-generated port-write requests. If LARGE_DESTID is 1, the DESTID_LSB field forms the least significant bits of a 16-bit DestID used in locally-generated port-write requests.	R/W	0
16	LARGE_DESTID	0 = Port-write transactions are generated with an 8-bit destination ID. 1 = Port-write transactions are generated with a 16-bit destination ID.	R/W	0
17:31	Reserved	N/A	R	0

12.7.9 RapidIO Port x Error Detect CSR

This register indicates transmission errors that are detected by the hardware.

Each write of a non-zero value to the Port x Error Detect CSR causes the Error Rate Counter to increment, if the corresponding error bit is enabled in the “RapidIO Port x Error Rate Enable CSR” on page 289. When the threshold is reached, hardware informs the system software of the error using its standard error reporting function. After the error has been reported, the system software can read and clear registers as necessary to complete its error handling protocol testing.

Register name: SP{0..7}_ERR_DET **Reset value: 0x0000_0000**	**Register offset: 1040, 1080, 10C0, 1100, 1140, 1180, 11C0, 1200**

Bits	0	1	2	3	4	5	6	7
00:07	IMP_SPEC_ERR	Reserved						
08:15	Reserved	CS_CRC_ERR	CS_ILL_ID	CS_NOT_ACC	PKT_ILL_ACKID	PKT_CRC_ERR	PKT_ILL_SIZE	Reserved
16:23	Reserved							
24:31	Reserved		LR_ACKID_ILL	PROT_ERR	Reserved	DELIN_ERR	CS_ACK_ILL	LINK_TO

Bits	Name	Description	Type	Reset Value
0	IMP_SPEC_ERR	Implementation Specific Error Detected Logical/Transport error per port. This bit indicates one or more of the following illegal field errors: • Reserved transport type (TT) detected (TT field = 10 or 11 for all but maintenance packets with hop count = 0) • Maximum retry threshold exceeded • Unmapped destination ID error • Parity Error in Lookup Table • ISF TEA error • Multicast TEA error • Port error Note: Clearing all the bits in this register also clears the following interrupt status bits in the “RapidIO Port x Interrupt Status Register” on page 318: • MAX_RETRY • TEA • MC_TEA • LUT_PAR_ERR • ILL_TRANS_ERR Caution: The Error Capture register information is only valid for Reserved Transport Type Detected errors and Unmapped DestID errors. For the Max Retry errors the information latched is the last packet received, not the packet that was retried. For more information on error capture, see “RapidIO Port x Error Capture Attributes CSR and Debug 0” on page 291.	R/W0C	0
1:8	Reserved	N/A	R	0
9	CS_CRC_ERR	Received a control symbol with a CRC error.	R/W0C	0
10	CS_ILL_ID	Received an acknowledge control symbol with an unexpected ackID (packet-accepted or packet_retry). The Capture register does not have valid information during this error detection.	R/W0C	0
11	CS_NOT_ACC	Received packet-not-accepted control symbol.	R/W0C	0
12	PKT_ILL_ACKID	Received packet with unexpected ackID.	R/W0C	0
13	PKT_CRC_ERR	Received a packet with a CRC error.	R/W0C	0
14	PKT_ILL_SIZE	Received packet exceeds 276 bytes.	R/W0C	0
15:25	Reserved	N/A	R	0

(Continued)

Bits	Name	Description	Type	Reset Value
26	LR_ACKID_ILL	Link response received with an ackID that is not outstanding. The Capture register does not have valid information during this error detection. During a recovery attempt by the Tsi574, it issued a link request control symbol to its link partner in order to attempt to clear the outstanding port error states. The link response it got back is supposed to contain the ACK_ID that the link partner is expecting in the next new packet it receives. The Tsi574 is indicating that it has already sent a packet with that ACK_ID and has received a packet accepted control symbol for it, and has moved to the next ACK_ID value. This value can be found in the "RapidIO Serial Port x Local ackID Status CSR" on page 268. Therefore, an ACK_ID mismatch has occurred and until the ACK_IDs are re-aligned, no packet transfers take place.	R/W0C	0
27	PROT_ERR	Protocol Error Received control symbol is unexpected.	R/W0C	0
28	Reserved	N/A	R	0
29	DELIN_ERR	Delineation Error Received unaligned /SC/ or /PD/, or undefined code-group. The Capture register does not capture information for this error.	R/W0C	0
30	CS_ACK_ILL	Received an unexpected acknowledge control symbol	R/W0C	0
31	LINK_TO	An acknowledge or Link-response is not received within the specified timeout interval (see "RapidIO Switch Port Link Timeout Control CSR" on page 263 register). The Capture register does not capture information for this error.	R/W0C	0

12.7.10 RapidIO Port x Error Rate Enable CSR

This register contains the bits that control when an error condition is allowed to increment the error rate counter. and be captured in the error capture register.

Each write of a non-zero value to the "RapidIO Port x Error Detect CSR" on page 286 causes the Error Rate Counter to increment, if the corresponding error bit is enabled in the Port x Error Rate Enable CSR. When the threshold is reached, hardware informs the system software of the error using its standard error reporting function. After the error has been reported, the system software can read and clear registers as necessary to complete its error handling protocol testing.

Register name: SP{0..7}_RATE_EN **Reset value: 0x0000_0000**	**Register offset: 1044, 1084, 10C4, 1104, 1144, 1184, 11C4, 1204**

Bits	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
00:07	IMP_SPEC_ERR	Reserved						
08:15	Reserved	CS_CRC_ERR_EN	CS_ILL_ID_EN	CS_NOT_ACC_EN	PKT_ILL_ACKID_EN	PKT_CRC_ERR_EN	PKT_ILL_SIZE_EN	Reserved
16:23	Reserved							
24:31	Reserved		LR_ACKID_ILL_EN	PROT_ERR_EN	Reserved	DELIN_ERR_EN	CS_ACK_ILL_EN	LINK_TO_EN

Bits	**Name**	**Description**	**Type**	**Reset Value**
0	IMP_SPEC_ERR	Logical /Transport Error Enable Enable error rate counting of implementation specific errors. See the "RapidIO Port x Error Detect CSR" on page 286 for a description of the errors applicable to this field. Caution: The Error Capture register information is only valid for Reserved Transport Type Detected errors and Unmapped DestID errors. For the Max Retry errors the information latched is the last packet received, not the packet that was retried. For more information on error capture, see "RapidIO Port x Error Capture Attributes CSR and Debug 0" on page 291.	R/W	0
1:8	Reserved	N/A	R	0
9	CS_CRC_ERR_EN	Enable error rate counting. Received Control Symbol with a CRC error.	R/W	0
10	CS_ILL_ID_EN	Enable error rate counting. Received an acknowledge control symbol with an unexpected ackID (packet-accepted or packet_retry).	R/W	0

(Continued)

Bits	Name	Description	Type	Reset Value
11	CS_NOT_ACC_EN	Enable error rate counting. Received packet-not-accepted control symbol.	R/W	0
12	PKT_ILL_ACKID_EN	Enable error rate counting. Received packet with unexpected ackID.	R/W	0
13	PKT_CRC_ERR_EN	Enable error rate counting. Received packet with a CRC error.	R/W	0
14	PKT_ILL_SIZE_EN	Enable error rate counting. Received packet exceeds 276 bytes.	R/W	0
15:25	Reserved	N/A	R	0
26	LR_ACKID_ILL_EN	Enable error rate counting. A received Link Response control symbol contains an ackID that is not outstanding.	R/W	0
27	PROT_ERR_EN	Enable error rate counting. Protocol Error. Received Control Symbol is unexpected.	R/W	0
28	Reserved	N/A	R	0
29	DELIN_ERR_EN	Enable error rate counting Delineation Error Received unaligned /SC/or/PD/ or undefined code-group.	R/W	0
30	CS_ACK_ILL_EN	Enable error rate counting An unexpected acknowledge control symbol was received.	R/W	0
31	LINK_TO_EN	Enable error rate counting An acknowledge or Link-response is not received within the specified time-out interval.	R/W	0

12.7.11 RapidIO Port x Error Capture Attributes CSR and Debug 0

This register indicates the type of information contained in the Port *x* Error Capture registers. In the case of multiple detected errors during the same clock cycle, only one of the errors must be reflected in the error type (ERR_TYPE) field.

When VAL_CAPT is set, the fields (except VAL_CAPT) are read-only. In debug mode this register is unlocked and all its fields are used for writing the content of the debug packet.

Register name: SP{0..7}_ERR_ATTR_CAPT_DBG0 **Reset value: 0x0000_0000**	**Register offset: 1048, 1088, 10C8, 1108, 1148, 1188, 11C8, 1208**

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

Bits	Name	Description	Type	Reset Value
0:1	INFO_TYPE	Type of information logged. • 00 = Packet • 01 = Control Symbol and unaligned /SC/or/PD/ or undefined code-group • 10 = Implementation specific (capture register contents are implementation specific to report implementation specific errors) • 11 = Reserved for serial port	R/W	0
2	Reserved	N/A	R	0
3:7	ERR_TYPE	Error Type Encoded 5-bit value of captured error bit in the “RapidIO Port x Error Detect CSR” on page 286. For detailed information on the ERR_TYPE values, refer to Table 40 on page 292. When an ERR_TYPE is reported, only specific errors cause the capture registers to capture valid data.	R/W	0
8:30	Reserved	N/A	R	0
31	VAL_CAPT	Capture Valid Information This bit is set by hardware to indicate that the packet/control symbol capture registers contain valid information. For control symbols, only capture register 0 contains meaningful information. Software writes 0 to clear this bit and unlock all capture registers of port *x*.	R/W0C	0

Table 40: ERR_TYPE Values

ERR_TYPE Value	Name	Error Description	Valid Data Capture
00000	Bit 0 = IMP_SPEC_ERR	Reserved Transport Type	Yes
		Max Retry Error Occurred	Yes
		Unmapped DestID Error	Yes
		Parity Error in Lookup Table	Yes
		ISF TEA Error	No
		Multicast TEA Error	No
		Port Error	No
00001-to-00111	Bit 1:8 = Reserved	-	-
01000	Bit 9 = CS_CRC_ERR	Control Symbol CRC Error	Yes
01001	Bit 10 = CS_ILL_ID	Control Symbol Illegal ID	Yes
01011	Bit 11 = CS_NOT_ACC	Control Symbol Not Accepted	Yes
01011	Bit 12 = PKT_ILL_ACKID	Packet Illegal AckID	Yes
01011	Bit 13 = PKT_ILL_SIZE	Packet Illegal Size	Yes
0110-to-11001	Bit 14:25 = Reserved	-	-
11010	Bit 26 = LR_ACKID_IL	Link Response Received with an Illegal AckID	No
11011	Bit 27 = PROT_ERR	Protocol Error	Yes
11100	Bit 28 = Reserved	-	-
11101	Bit 29 = DELIN_ERR	Delineation Error	No
11110	Bit 30 = CS_ACKID_ILL	Control Symbol Illegal AckID	Yes
11111	Bit 31 = LINK_TO	Link Timeout	No

12.7.12 RapidIO Port x Packet and Control Symbol Error Capture CSR 0 and Debug 1

In debug mode this register is unlocked. It contains bytes 4 to 7 of the debug packet being composed.

During normal operation, this register captures bytes 0 to 3 of the packet, or the entire control symbol, that was detected to be in error.

To assist in software testing and debug of the system error recovery and threshold function, the "RapidIO Port x Error Detect CSR" on page 286 and the Port x Error Capture registers are also writable. Software must clear the Capture Valid Info (VAL_CAPT) bit in the "RapidIO Port x Error Capture Attributes CSR and Debug 0" on page 291, then write the packet/control symbol information to the other capture registers.

Register name: SP{0..7}_ERR_CAPT_0_DBG1 **Reset value: 0x0000_0000**	**Register offset: 104C, 108C, 10CC, 110C, 114C, 118C, 11CC, 120C**

Bits	0	1	2	3	4	5	6	7
00:7	CAPT_0[0:7]							
8:15	CAPT_0[8:15]							
16:23	CAPT_0[16:23]							
24:31	CAPT_0[24:31]							

Bits	Name	Description	Type	Reset Value
0:31	CAPT_0	Character and control symbol or bytes 0 to 3 of packet header.	R/W	0

12.7.13 RapidIO Port x Packet Error Capture CSR 1 and Debug 2

Register name: SP{0..7}_ERR_CAPT_1_DBG2 Reset value: 0x0000_0000	Register offset: 1050, 1090, 10D0, 1110, 1150, 1190, 11D0, 1210

Bits	0	1	2	3	4	5	6	7
00:7	CAPT_1[0:7]							
8:15	CAPT_1[8:15]							
16:23	CAPT_1[16:23]							
24:31	CAPT_1[24:31]							

Bits	Name	Description	Type	Reset Value
0:31	CAPT_1	Bytes 4 to 7 of the packet	R/W	0

12.7.14 RapidIO Port x Packet Error Capture CSR 2 and Debug 3

Register name: SP{0..7}_ERR_CAPT_2_DBG3 Reset value: 0x0000_0000	Register offset: 1054, 1094, 10D4, 1114, 1154, 1194, 11D4, 1214

Bits	0	1	2	3	4	5	6	7
0:7	CAPT_2[0:7]							
8:15	CAPT_2[8:15]							
16:23	CAPT_2[16:23]							
24:31	CAPT_2[24:31]							

Bits	Name	Description	Type	Reset Value
0:31	CAPT_2	Byte 8 to 11 of the packet	R/W	0

12.7.15 RapidIO Port x Packet Error Capture CSR 3 and Debug 4

Register name: SP{0..7}_ERR_CAPT_3_DBG4 Reset value: 0x0000_0000	Register offset: 1058, 1098, 10D8, 1118, 1158, 1198, 11D8, 1218

Bits	0	1	2	3	4	5	6	7
0:7	CAPT_3[0:7]							
8:15	CAPT_3[8:15]							
16:23	CAPT_3[16:23]							
24:31	CAPT_3[24:31]							

Bits	Name	Description	Type	Reset Value
0:31	CAPT_3	Byte 12 to 15 of the packet	R/W	0

12.7.16 RapidIO Port x Error Rate CSR

This register and the "RapidIO Port x Error Rate Threshold CSR" on page 298 are used to monitor and control the reporting of transmission errors.

Register name: SP{0..7}_ERR_RATE Reset value: 0x8000_0000	Register offset: 1068, 10A8, 10E8, 1128, 1168, 11A8, 11E8, 1228

Bits	0	1	2	3	4	5	6	7
00:07	ERR_RB							
08:15	Reserved							ERR_RR
16:23	PEAK							
24:31	ERR_RATE_CNT							

Bits	Name	Description	Type	Reset Value
0:7	ERR_RB	The Error Rate Bias value. Register bus frequency - 100 MHz • 00 = Do not decrement error rate counter • 01 = Decrement every 1.31ms • 02 = Decrement every 10.48ms • 04 = Decrement every 83.88ms • 08 = Decrement every 1.342s • 10 = Decrement every 10.74s • 20 = Decrement every 86s • 40 = Decrement every 1374s • 80 = Decrement every 10995s • FF - Decrement every 1.28us (Debug only) Other values are reserved.	R/W	0x80
8:13	Reserved	N/A	R	0
14:15	ERR_RR	Error Rate Recovery This field defines how far above the Error Rate Failed Threshold Trigger the Error Rate Counter is allowed to count. • 00 = 2 errors above • 01 = 4 errors above • 10 = 16 errors above • 11 = No limit	R/W	0
16:23	PEAK	The maximum value attained by the error rate counter. This value increments with ERR_RATE_CNT, but does not decrement except through a host controlled register write.	R/W	0

(Continued)

Bits	Name	Description	Type	Reset Value
24:31	ERR_ RATE_ CNT	Error Rate Counter These bits maintain a count of the number of transmission errors that have been detected by the port. This number is decremented by the Error Rate Bias function. The counter cannot over or underflow and continue to increment or decrement as defined, even if thresholds are met. Software can reset this counter. If the value of the counter equals the error rate threshold trigger register, an error is reported. For more information see the *RapidIO Interconnect Specification (Revision 1.3), Part 8: Error Management Extensions Specification*.	R/W	0

12.7.17 RapidIO Port x Error Rate Threshold CSR

This register and the "RapidIO Port x Error Rate CSR" on page 296 are used to monitor and control the reporting of transmission errors.

Register name: SP{0..7}_ERR_THRESH Reset value: 0xFFFF_0000	Register offset: 106C, 10AC, 10EC, 112C, 116C, 11AC, 11EC, 122C

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

Bits	Name	Description	Type	Reset Value
0:7	ERR_RFT	Error Rate Failed Threshold These bits provide the threshold value for reporting an error condition due to a possibly broken link. • 00 = Disable the error rate failed register • 01 = Set the error reporting threshold to 1 • 02 = Set the error reporting threshold to 2 • ... • FF - Set the error reporting threshold to 255	R/W	0xFF
8:15	ERR_RDT	Error Rate Degraded Threshold These bits provide the threshold value for reporting an error condition due to a degrading link. • 00 = Disable the error rate degraded register • 01 = Set the error reporting threshold to 1 • 02 = Set the error reporting threshold to 2 • ... • FF - Set the error reporting threshold to 255	R/W	0xFF
16:31	Reserved	N/A	R	0

12.8 IDT-Specific RapidIO Registers

The registers in this section are specific to IDT's switching products. The following table shows IDT-specific RapidIO Registers that are not defined in the *RapidIO Interconnect Specification (Revision 1.3)*.

When a individual port is powered down, the IDT-Specific RapidIO Registers are read only and return 0 with the exception of "RapidIO Port x Error and Status CSR" on page 270 and "RapidIO Serial Port x Control CSR" on page 273, both of which return 0x00000001 when read.

These registers are reset by the HARD_RST_b reset input signal, as well as when the Tsi574 performs a self-reset. The registers within a port are also reset by a "Port Reset". For more information on Tsi574 reset implementation and behavior, see "Clocks, Resets and Power-up Options" on page 203. It is possible to override reset values of writable fields, and some read-only fields, using the I^2C register loading capability on boot. Refer to "I^2C Interface" on page 139 for more information on the use of I^2C controller register loading capability.

Table 41: IDT-Specific Broadcast RapidIO Registers

Port	Register Offset	Description
BC	10000	Broadcast addresses, which affect register copies in all the ports.
SP0	11000	1x/4x mode serial port
SP1	11100	1x mode serial port
SP2	11200	1x/4x mode serial port
SP3	11300	1x mode serial port
SP4	11400	1x/4x mode serial port
SP5	11500	1x mode serial port
SP6	11600	1x/4x mode serial port
SP7	11700	1x mode serial port

Non-Broadcast Per-Port Registers

The following table shows the IDT-specific per-port registers not defined by the *RapidIO Interconnect Specification (Revision 1.3)*. It is not possible to broadcast to these registers.

Table 42: IDT-Specific Per-Port Performance Registers

Port	Register Offset	Description
SP0	13000	1x/4x mode serial port
SP1	13100	1x mode serial port
SP2	13200	1x/4x mode serial port
SP3	13300	1x mode serial port
SP4	13400	1x/4x mode serial port
SP5	13500	1x mode serial port
SP6	13600	1x/4x mode serial port
SP7	13700	1x mode serial port

12.8.1 RapidIO Port x Discovery Timer

This register defines discovery-timer value for the serial ports in 4x mode..

Register name: SP{BC,0..7}_DISCOVERY_TIMER Reset value: 0x90C0_0000	Register offset: 10000, 11000, 11100, 11200, 11300, 11400, 11500, 11600, 11700

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

Bits	Name	Description	Type	Reset Value
0:3	DISCOVERY_TIMER	Discovery Timer This field is used by serial ports configured to operate in 4x mode. The discovery-timer allows time for the link partner to enter its discovery state, and if the link partner supports 4x mode, for all four lanes to be aligned. The discovery timer has a value of 11.79 ms, with the P_CLK set to 100 MHz • 0: 320 • {1:15}: DISCOVERY_TIMER (decimal) * 1179639 * 1/ P_CLK = actual time Note: Refer to “P_CLK Programming” on page 479 for more information on programming the P_CLK operating frequency.	R/W	9
4:7	Reserved	N/A	R	0
8:9	PW_PRIORITY	Port-write packet priority This field sets the priority of a port-write packet. The priority can be set from 0 to 3 00 = priority 0 01 = priority 1 10 = priority 2 11 = priority 3	R/W	11
10:31	Reserved	N/A	R	0

12.8.2 RapidIO Port x Mode CSR

This register defines the mode of operation for the ports, and contains the interrupt enables for the Multicast-Event control symbol and Reset control symbol.

Register name: SP{BC,0..7}_MODE **Reset value: 0x0300_0000**	**Register offset: 10004, 11004, 11104, 11204, 11304, 11404, 11504, 11604, 11704**

Bits	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
00:07	Reserved		IDLE_ERR_DIS	Reserved	PW_DIS	Reserved	SELF_RST	LUT_512
08:15	Reserved							
16:23	Reserved							
24:31	Reserved						MCS_INT_EN	RCS_INT_EN

Bits	**Name**	**Description**	**Type**	**Reset Value**
0:1	Reserved	N/A	R	0
2	IDLE_ERR_DIS	Idle Error Checking Disable 0 = Error checking is enabled by default if one or more data characters are sent (Dx.y characters not delimited with start of packet/end of packet control symbols) in an idle sequence, the device enters the Input Error stopped state. 1 = Ignore all not idle or invalid characters in the idle sequence.	R/W	0
3	Reserved	N/A	R	0
4	PW_DIS	Port_Write Disable 0 = Port-write Error reporting is enabled (default) 1 = Port-write is disabled	R/W	0
5	Reserved	N/A	R	0
6	SELF_RST	Self Reset Enable After four link-request reset control symbols are accepted, the device either resets itself or raises an interrupt, according to the value in this register field. 0 = Disabled. Interrupt signal is asserted (if RCS_INT_EN is also asserted) 1 = Enabled. Device is reset	R/W	1

(Continued)

Bits	Name	Description	Type	Reset Value
7	LUT_512	LUT_512 Sets the mode of the destination ID lookup table 0 = Global LUT (64K destination IDs, assigned with resolution of 256 destination IDs) 1 = One 512-entry local LUT	R/W	1
8:29	Reserved	N/A	R	0
30	MCS_INT_EN	Multicast-Event Control Symbol Interrupt Enable 0 = Disabled 1 = Enabled. The interrupt signal is high when the multicast-event control symbol is received.	R/W	0
31	RCS_INT_EN	Reset Control Symbol Interrupt Enable 0 = Disabled 1 = Enabled. The interrupt signal is High when four reset control symbols are received in a sequence.	R/W	0

12.8.3 RapidIO Port x Multicast-Event Control Symbol and Reset Control Symbol Interrupt CSR

This register contains the interrupt status for Multicast-Event control symbols and Reset control symbols.

Register name: SP{BC,0..7}_CS_INT_STATUS Reset value: 0x0000_0000	Register offset: 10008, 11008, 11108, 11208, 11308, 11408, 11508, 11608, 11708

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

Bits	Name	Description	Type	Reset Value
0:29	Reserved	N/A	R	0
30	MCS	Multicast Event Control Symbol Interrupt Status Indicates whether a multicast event control symbol has been received on the port. Reading the MCS field using the BC offset gives the value of Port 0. All MCS interrupts from ports are ORed together. The “Global Interrupt Status Register” on page 377 shows the status of the combined MCS interrupts from all ports. Write 1 to clear this bit. Writing 1 to this bit clears the interrupt on all the ports when using the BC offset.	R/W1C	0
31	RCS	Reset Control Symbol Received Interrupt Status Indicates that four consecutive Reset control symbols have been received on the port. Reading the RCS field using the BC offset gives the value of Port 0. All RST interrupts from ports are ORed together. The “Global Interrupt Status Register” on page 377 register shows the status of the combined RCS from all ports. Write 1 to clear this bit. Writing 1 to this bit clears the interrupt on all the ports.	R/W1C	0

12.8.4 RapidIO Port x RapidIO Watermarks

This register controls ingress buffer allocation for reception of packets for each port (see "Egress Watermark" on page 95).

Register name: SP{BC,0..7}_RIO_WM Reset value: 0x0001_0203	Register offset: 1000C, 1100C, 1110C, 1120C, 1130C, 1140C, 1150C, 1160C, 1170C

Bits	0	1	2	3	4	5	6	7
00:7	Reserved							
8:15	Reserved					PRIO2WM		
16:23	Reserved					PRIO1WM		
24:31	Reserved					PRIO0WM		

Bits	Name	Description	Type	Reset Value
0:12	Reserved	N/A	R	0
13:15	PRIO2WM	Priority 2 packets are accepted if the number of free buffer is greater than this value. This value must be smaller than PRIO1WM. Note: It is a programming error for this value to be either greater than or equal to PRIO1WM or PRIO0WM, or greater than 7.	R/W	1
16:20	Reserved	N/A	R	0
21:23	PRIO1WM	Priority 1 packets are accepted if the number of free buffer is greater than this value. This value must be smaller than PRIO0WM. Note: It is a programming error for this value to be either greater than or equal to PRIO0WM, or greater than 7.	R/W	2
24:28	Reserved	N/A	R	0
29:31	PRIO0WM	Priority 0 packets are accepted if the number of free buffer is greater than this value. Note: It is a programming error for this value to be greater than 7.	R/W	3

This register must be programmed after reset and not when transactions are in progress.

12.8.5 RapidIO Port x Route Config DestID CSR

This register and SPx_ROUTE_CFG_PORT operate together to provide indirect read and write access to the LUTs. The registers are identical to RIO_ROUTE_CFG_DESTID and RIO_ROUTE_CFG_PORT, except the "RapidIO Port x Route Config Output Port CSR" on page 307 are per-port registers and they include an auto-increment bit to increment the contents of the destination ID register after a read or write operation.

Register name: SP{BC,0..7}_ROUTE_CFG_DESTID **Reset value: 0x0000_0000**	**Register offset: 10070, 11070, 11170, 11270, 11370, 11470, 11570, 11670, 11770**

Bits	0	1	2	3	4	5	6	7
00:07	AUTO_INC	PAR_INVERT	Reserved					
08:15	Reserved							
16:23	LRG_CFG_DEST_ID[0:7]							
24:31	CFG_DEST_ID[8:15]							

Bits	Name	Description	Type	Reset Value
0	AUTO_INC	Automatically post-increment the destination ID when the destination ID is used to perform either a read or a write, through the "RapidIO Port x Route Config Output Port CSR" on page 307.	R/W	0
1	PAR_INVERT	Parity Invert This bit is for testing of interrupt and/or demerit software systems. 0 = Normal operation 1 = Invert the parity bit for each LUT entry written (but not read). This causes a parity error when the LUT entry is used to route a packet.	R/W	0
2:15	Reserved	N/A	R	0
16:23	LRG_CFG_DEST_ID	This field specifies the most significant byte of the destination ID used to select an entry in the LUT, when the "RapidIO Port x Route Config Output Port CSR" on page 307 is read or written.	R/W	0x00
24:31	CFG_DEST_ID	Specifies the destination ID used to select an entry in the LUT when the RIO_ROUTE_CFG_PORT register is read or written. This value increments by one for every write to the "RapidIO Port x Route Config Output Port CSR" on page 307 when the AUTO_INC bit is set	R/W	0x00

12.8.6 RapidIO Port x Route Config Output Port CSR

This register and SPx_ROUTE_CFG_DESTID operate together to provide indirect read and write access to the LUTs. The registers are identical to RIO_ROUTE_CFG_DESTID and RIO_ROUTE_CFG_PORT, except the RIO_ROUTE_CFG_PORT are per-port registers and they include an auto-increment bit to increment the contents of the destination ID register after a read or write operation.

Register name: SP{BC,0..7}_ROUTE_CFG_PORT **Reset value: Undefined**	**Register offset: 10074, 11074, 11174, 11274, 11374, 11474, 11574, 11674, 11774**

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

Bits	**Name**	**Description**	**Type**	**Reset Value**
0:23	Reserved	N/A	R	0
24:31	PORT	This is the RapidIO output port through which all transactions meant for CFG_DEST_ID are sent. Writing a value greater or equal to PORT_TOTAL field in the “RapidIO Switch Port Information CAR” on page 245 sets the LUT entry to an unmapped state. For future compatibility, write the value 0xFF to indicate an unmapped destination ID. When reading an unmapped value from the LUT, this field is set to 0xFF.	R/W	Undefined

12.8.7 RapidIO Port x Local Routing LUT Base CSR

This register is required for switch devices that operate in a large system. For small systems, this register is ignored.

The serial port supports local and global routing LUT pages. The number of entries is defined by the "RapidIO Route LUT Size CAR" on page 249

Register name: SP{BC,0..7}_ROUTE_BASE **Reset value: 0x0000_0000**	**Register offset: 10078, 11078, 11178, 11278, 11378, 11478, 11578, 11678, 11778**

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

Bits	Name	Description	Type	Reset Value
0:7	BASE	This value represents the most significant byte of a destination ID. If the most significant upper 8 bits of an incoming 16-bit destination ID match this field, the least significant bits of the destination ID is used to index the local LUT. Otherwise, the most significant 8 bits of the destination ID is used to index the global LUT. Refer to "Lookup Tables" on page 37 for more information.	R/W	0
8:31	Reserved	N/A	R	0

12.8.8 RapidIO Multicast Write ID x Register

This register contains the Multicast ID, which is associated to the multicast mask registers. The switch supports eight multicast groups and the Multicast ID registers for each multicast group must contain unique values.

These registers are implemented globally, and are visible from every ingress port.

Register name: RIO_MC_ID{0..7} **Reset value: 0x0000_0000**	**Register offset: 10300, 10304, 10308, 1030C, 10310, 10314, 10318, 1031C**

Bits	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
00:7	MC_EN	LARGE_ SYS	Reserved					
8:15	Reserved							
16:23	MC_ID[15:8]							
24:31	MC_ID[7:0]							

Bits	**Name**	**Description**	**Type**	**Reset Value**
0	MC_EN	Multicast can be disabled by setting this bit. 0 = Disabled 1 = Enabled	R/W	0
1	LARGE_ SYS	This field defines multicast destination ID (MC_ID) in the Large or Small system. The MC_ID of Small system is not a subset of MC_ID of Large system, but both systems can co-exist together. 0 = Small system 1 = Large system	R/W	0
2:15	Reserved	N/A	R	0
16:31	MC_ID	This field defines the multicast destination ID for which the associated multicast mask is activated for this extended features block.	R/W	0x0000

When using these registers, it is important that no multiple, identical entries exist because an addition of an association of destination ID does not delete the association of a duplicate destination ID to a different mask.

12.8.9 RapidIO Multicast Write Mask x Register

This register contains the set of egress ports where a multicast packet is sent when it matches the destination ID associated with the mask. These bits form the multicast vector used by the broadcast buffer to determine which egress ports the packet is copied to.

The bit descriptions apply to all packets received on a port whose destination ID field maps to the multicast ID register value. A multicast packet received on an input port is sent to all egress ports whose multicast select bit is set to 1. However, the multicast packet is not sent to the port from which it was received, regardless of the setting of that port's multicast select bit.

This registers is only located in the multicast engine.

Register name: RIO_MC_MSK{0..7} **Reset value: 0x0000_0000**	**Register offset: 10320, 10324, 10328, 1032C, 10330, 10334, 10338, 1033C**

Bits	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
00:07	MC_MSK[15:8]							
08:15	MC_MSK[7:0]							
16:23	Reserved							
24:31	Reserved							

Bits	**Name**	**Description**	**Type**	**Reset Value**
00:15	MC_MSK	Port x Multicast Select Where x refers to ports 0 through 7. (Other values are reserved.) 0 = Do not Multicast the packet to output port x 1 = Multicast the packet to output port x An output port is specified by the bit position: Bit 0 = mask of output port 0 Bit 1 = mask of output port 1 ...	R/W	0
16:31	Reserved	N/A	R	0

12.8.10 RapidIO Port x Control Independent Register

This register is used for error recovery.

Register name: SP{0..7}_CTL_INDEP Reset value: 0x0100_0000	Register offset: 13004, 13104, 13204, 13304, 13404, 13504, 13604, 13704

Bits	0	1	2	3	4	5	6	7
00:07	Reserved		SCRATCH	Reserved		FORCE_REINIT	Reserved	TRANS_MODE
08:15	DEBUG_MODE	SEND_DBG_PKT	Reserved		PORT_ERR_EN	MC_TEA_EN	LINK_INIT_NOTIFICATION_EN	LUT_PAR_ERR_EN
16:23	MAX_RETRY_THRESHOLD							
24:31	ILL_TRANS_ERR	IRQ_EN	MAX_RETRY_EN	OUTB_DEPTH_EN	INB_DEPTH_EN	INB_RDR_EN	Reserved	TEA_EN

Bits	Name	Description	Type	Reset Value
0:1	Reserved	N/A	R	0
2	SCRATCH	Scratch Pad This bit controls no functionality. It is a read/write *scratch pad* bit for software use.	R/W	0
3:4	Reserved	N/A	R	0
5	FORCE_REINIT	Force Link Re-initialization Process This bit is active on write and automatically returns to 0.	R/W1S	0
6	Reserved	N/A	R	0
7	TRANS_MODE	Transfer mode for each port 0 = Cut-through mode. In cut-through mode, the incoming packet is forwarded through the switch as soon as the routing information is received. 1 = Store-and-forward mode (default). In store-and-forward mode, the incoming packet is not sent to the switch fabric until the whole packet is received. Note: If ports are operating at different speeds, cut-through mode may impact the overall performance of the switch. This is because in cut-through mode, a slower port can use the internal switching fabric for a long time relative to a faster port, incurring additional latency and potentially throughput loss on the faster port.	R/W	1

(Continued)

Bits	Name	Description	Type	Reset Value
8	DEBUG_MODE	Mode of operation 0 = Normal 1 = Debug mode Debug mode unlocks the capture registers for writing and enables the debug packet generator feature.	R/W	0
9	SEND_DBG_PKT	Send Debug Packet 0 = Normal 1 = Send debug packet This bit is set by software and is cleared by hardware after the debug packet is sent. Writes when the bit is already set are ignored. Debug mode only.	R/W	0
10:11	Reserved	N/A	R	0
12	PORT_ERR_EN	Port Error Enable An interrupt and/or port-write is generated if there is a Port Error.	R/W	0
13	MC_TEA_EN	Multicast TEA Enable An interrupt and/or port write is generated when the multicast engine has timed out before it could deliver a packet to the broadcast buffer.	R/W	0
14	LINK_INIT_NOTIFICATION_EN	Enables interrupts and port writes for LINK_INIT_NOTIFICATION events. 0 = Interrupt and/or port write disabled 1 = Interrupt and/or port write enabled	R/W	0
15	LUT_PAR_ERR_EN	Enables interrupts for parity errors in the lookup table. 0 = Interrupt disabled 1 = Interrupt enabled	R/W	0
16:23	MAX_RETRY_THRESHOLD	Maximum Retry Threshold These bits provide the threshold value for reporting congestion at an outbound switch buffer caused by congestion at the link partner. When the number of consecutive retries reaches this threshold, the switch generates a port-write and sends the IMP_SPEC_ERR bit in the RapidIO Port x Error Detect CSR. 00 = Disable the RETRY_ERROR reporting 01 = Set the MAX_RETRY_THRESHOLD to 1 02 = Set the MAX_RETRY_THRESHOLD to 2 ... FF = Set the MAX_RETRY_THRESHOLD to 255	R/W	0x00
24	ILL_TRANS_ERR	Illegal Transfer Error Reporting Enable If enabled, the port-write and/or interrupt report an error when the ILL_TRANS_ERR bit is set.	R/W	0

(Continued)

Bits	Name	Description	Type	Reset Value
25	IRQ_EN	Interrupt Error Report Enable If enabled, the interrupt signal is high when the IRQ_ERR bit is set to 1.	R/W	0
26	MAX_RETRY_EN	Maximum Retry Report Enable If enabled, the port-write and interrupt report an error when the MAX_RETRY_THRESHOLD is exceeded and the MAX_RETRY bit is set in the “RapidIO Port x Interrupt Status Register” on page 318.	R/W	0
27	OUTB_DEPTH_EN	Output Queue Depth Interrupt Enable An interrupt is generated when the OUTB_DEPTH bit is set in the “RapidIO Port x Interrupt Status Register” on page 318.	R/W	0
28	INB_DEPTH_EN	Input Queue Depth Interrupt Enable An interrupt is generated when the INB_DEPTH bit is set in the “RapidIO Port x Interrupt Status Register” on page 318.	R/W	0
29	INB_RDR_EN	Inbound Reorder Interrupt Enable An interrupt is generated when the INB_RDR bit is set in the “RapidIO Port x Interrupt Status Register” on page 318.	R/W	0
30	Reserved	Reserved	R/W	0
31	TEA_EN	Transfer Error Acknowledge Enable An interrupt is generated if the internal switching fabric times out trying to send the packet to its egress port.	R/W	0

12.8.11 RapidIO Port x Send Multicast-Event Control Symbol Register

When this register is written, it causes a Multicast-Event control symbol to be sent on the corresponding RapidIO output port. The port must be enabled for multicast control symbol forwarding through the MCS_EN bit in the SP{0..7}_CTL register.

A write to this register is not considered complete until the multicast-event control symbol is queued to the outbound flow. There can be only one outstanding request at a time. Subsequent requests are ignored until the multicast control symbol is sent.

Register name: SP{0..7}_SEND_MCS **Reset value: 0x0000_0002**	**Register offset: 1300C, 1310C, 1320C, 1330C, 1340C, 1350C, 1360C, 1370C**

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

Bits	**Name**	**Description**	**Type**	**Reset Value**
0:29	Reserved	N/A	R	0
30	DONE	0= The Tsi574 sets this field to 0 when system software sets SEND to 1. 1 = The Tsi574 sets this field to 1 once it has sent the Multicast-Event control symbol. It indicates that the Tsi574 is ready to send another Multicast Event control symbol.	R	1
31	SEND	Write 1 to send a multicast-event control symbol when the DONE bit is set to 1.	R/W1S	0

12.8.12 RapidIO Port x LUT Parity Error Info CSR

The RapidIO Port x LUT Parity Error Info CSR contains information about the look up operation that caused the parity error, as well as the LUT information associated with the parity error.

The contents of this register are frozen when a LUT parity error is indicated in the "RapidIO Port x Interrupt Status Register" on page 318 register. Writes to this register have no affect when a LUT parity error is not indicated.

Register name: SP{0..7}_LUT_PAR_ERR_INFO **Reset value: 0x0000_0000**	**Register offset: 13010, 13110, 13210, 13310, 13410, 13510, 13610, 13710**

Bits	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
00:07	DESTID_MSB							
08:15	DESTID_LSB							
16:23	LG_DESTID	Reserved						
24:31	PTY_BIT	LUT_VLD	Reserved		PORT_NUM			

Bits	**Name**	**Description**	**Type**	**Reset Value**
0:7	DESTID_MSB	Most significant byte of a 16 bit destination ID used in the lookup operation which caused the error. Only valid if the LG_DESTID field value is 1 and a LUT parity error is signalled in the Port x Interrupt Status CSR. When the parity error is cleared in "RapidIO Port x Interrupt Status Register" on page 318, the information in these bits become meaningless. Note: In small systems, this field holds the 8-bit Dest_ID.	R/W	0
8:15	DESTID_LSB	Least significant byte of a 16 bit destination ID used in the lookup operation which caused the error. Only valid if a LUT parity error is signalled in the Port x Interrupt Status CSR. When the parity error is cleared in "RapidIO Port x Interrupt Status Register" on page 318, the information in these bits become meaningless. Note: This field is not used in small systems.	R/W	0
16	LG_DESTID	1 = This bit is set if the TT code of the packet which caused the error is anything other than 0.	R/W	0
17:23	Reserved	N/A	R	0
24	PTY_BIT	The parity bit read from the LUT memory array.	R/W	0

(Continued)

Bits	Name	Description	Type	Reset Value
25	LUT_VLD	0 = Indicates that the LUT entry is unmapped. The PORT_NUM field value should be 0xF in this case. 1 = Indicates the LUT entry is mapped. The PORT_NUM field value is the port to which the packet could be routed (0x0 to 0xF). Caution: The value of this bit is unpredictable when there is a parity error in the LUT. For more information, see “Lookup Table Parity” on page 49.	R/W	0
26:27	Reserved	N/A	R	0
28:31	PORT_NUM	The Tsi574 port number where packets are routed. If the port is unmapped (LUT_VLD = 0), then the field reads 0xF.	R/W	0

12.8.13 RapidIO Port x Control Symbol Transmit

Writing to this register transmits a single control symbol to RapidIO. This register is only used for debug purposes.

All control symbol fields are defined according to the *RapidIO Interconnect Specification (Revision 1.3)*. The control symbol's CRC field is generated by hardware.

Register name: SP{0..7}_CS_TX **Reset value: 0x0000_0000**	**Register offset: 13014, 13114, 13214, 13314, 13414, 13514, 13614, 13714**

Bits	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
00:07	STYPE_0			PAR_0				
08:15	PAR_1					STYPE_1		
16:23	CMD			CS_EMB	Reserved			
24:31	Reserved							

Bits	**Name**	**Description**	**Type**	**Reset Value**
0:2	STYPE_0	Encoding for control symbol This field uses the parameters PAR_0 and PAR_1.	R/W	0
3:7	PAR_0	Used in conjunction with stype0 encoding.	R/W	0
8:12	PAR_1	Used in conjunction with stype0 encoding.	R/W	0
13:15	STYPE_1	Encoding for the control symbol that uses the CMD parameter.	R/W	0
16:18	CMD	Used in conjunction with stype1 encoding to define the link maintenance commands.	R/W	0
19	CS_EMB	Embed the control symbol into a data stream 0 = Control symbol is sent out immediately 1 = Control symbol is sent immediately if there is data transferring on the output port, or is inserted until after the first 32 bits of data of the next packet sent if there is currently no data transferring on the output port.	R/W	0
20:31	Reserved	N/A	R	0

Writing to this register causes control symbols to be generated that can interfere with the operation of the port. This can cause the port or its link partner to enter the input error stopped state due to the reception of an unexpected control symbol.

12.8.14 RapidIO Port x Interrupt Status Register

Register name: SP{0..7}_INT_STATUS Reset value: 0x0000_0000	Register offset: 13018, 13118, 13218, 13318, 13418, 13518, 13618, 13718

Bits	0	1	2	3	4	5	6	7
00:07	Reserved							
08:15	Reserved					MC_TEA	LINK_INIT_NOTIFICATION	LUT_PAR_ERR
16:23	Reserved							
24:31	ILL_TRANS_ERR	IRQ_ERR	MAX_RETRY	OUTB_DEPTH	INB_DEPTH	INB_RDR	Reserved	TEA

Bits	Name	Description	Type	Reset Value
0:12	Reserved	N/A	R	0
13	MC_TEA	This interrupt is raised when the Multicast Engine has timed out before it could deliver a packet to the broadcast buffer. This bit is cleared by writing a 1 to it, or by clearing all bits in the "RapidIO Port x Error Detect CSR" on page 286.	R/W1C	0
14	LINK_INIT_NOTIFICATION	Link Initialization Notification Once set, the LINK_INIT_NOTIFICATION bit is cleared by writing 1 to it. When the PORT_LOCKOUT bit is set in "RapidIO Serial Port x Control CSR" on page 273, and a link has initialized according to the PORT_OK bit in "RapidIO Port x Error and Status CSR" on page 270, the LINK_INIT_NOTIFICATION is set to 1. To stop the LINK_INIT_NOTIFICATION bit from being set, PORT_LOCKOUT must be set to 0 and/or the link must no longer be in an initialized state.	R/W1C	0
15	LUT_PAR_ERR	Lookup Table Parity Error 1= a packet looks up its destination ID in the lookup table, and the selected lookup table entry has a parity error. This bit is cleared by writing a 1 to it, or by clearing all the bits in the "RapidIO Port x Error Detect CSR" on page 286.	R/W1C	0
16:23	Reserved	N/A	R	0

(Continued)

Bits	Name	Description	Type	Reset Value
24	ILL_TRANS_ERR	Illegal Transfer Error This bit is set to 1 when the following occurs: • Received transaction has reserved *tt* field for all but maintenance packets with hop count = 0 • DestinationID is unmapped (not defined in lookup table) Once set, the bit remains set until written with logic 1 to clear. The setting of this bit generates a port-write and interrupt if a the bit ILL_TRANS_EN bit is set in the "RapidIO Port x Control Independent Register" on page 311. This bit duplicates a function of bit 0 of the "RapidIO Port x Error Detect CSR" on page 286, but a port-write is sent immediately when the error is detected without exceeding the "RapidIO Port x Error Rate Threshold CSR" on page 298. This error also reported in "RapidIO Port x Error Detect CSR" on page 286.	R/W1C	0
25	IRQ_ERR	Interrupt Error Status Set to one if an error occurs (refer to the "Error Management" on page 57). Once set, the bit remains unchanged until all the error sources are cleared. The setting of this bit generates an interrupt if the IRQ_EN bit in"RapidIO Port x Control Independent Register" on page 311 is set.	R	0
26	MAX_ RETRY	Maximum Retry Error Set when number of retries has reached MAX_RETRY_THRESHOLD. An interrupt is generated if MAX_RETRY_EN is set. A port-write request can also be generated if enabled. This bit is ignored if MAX_RETRY_THRESHOLD is 0x00. This bit is cleared by writing a 1 to it, or by clearing all bits in the "RapidIO Port x Error Detect CSR" on page 286.	R/W1C	0
27	OUTB_DEPTH	Outbound Depth Interrupt This value is set when Output Queue Depth Count reaches the maximum number defined in the Output Queue Depth Threshold field in the "RapidIO Port x Transmitter Output Queue Depth Threshold Register" on page 342. To get an interrupt in this status register, the Outbound Depth Interrupt Enable bit in the "RapidIO Port x Control Independent Register" on page 311 is set to 1. Writing a 1 to this bit clears the interrupt and clears CONG_CNTR in the "RapidIO Port x Transmitter Output Queue Congestion Status Register" on page 344.	R/W1C	0

(Continued)

Bits	Name	Description	Type	Reset Value
28	INB_DEPTH	Inbound Depth Interrupt This value is set when Input Queue Depth Count reaches the maximum number defined in the Input Queue Depth Threshold field in "RapidIO Port x Receiver Input Queue Depth Threshold Register" on page 347. To get an interrupt in this status register, the Inbound Depth Interrupt Enable bit in the"RapidIO Port x Control Independent Register" on page 311 is set to 1. Writing a 1 to this bit clears the interrupt and clears INB_QD_CNT in the "RapidIO Port x Receiver Input Queue Congestion Status Register" on page 349.	R/W1C	0
29	INB_RDR	Inbound Interrupt: Reordering This value is set when Reordering Count reaches the maximum number defined in the Inbound Reordering Threshold field in the "RapidIO Port x Reordering Counter Register" on page 352. To get an interrupt in this status register, the Inbound Interrupt Reordering Enable bit in the "RapidIO Port x Control Independent Register" on page 311 has to be set to 1. Writing a 1 to this bit clears the interrupt and clears INB_RDR_CNT in the "RapidIO Port x Reordering Counter Register" on page 352.	R/W1C	0
30	Reserved	Reserved	R/W1C	0
31	TEA	This interrupt is raised when the internal switching fabric has timed out before it could deliver a packet to an egress port. This bit is cleared by writing a 1 to it, or by clearing all bits in the "RapidIO Port x Error Detect CSR" on page 286.	R/W1C	0

Writing 0 to any bit in the "RapidIO Port x Interrupt Generate Register" on page 321 clears the corresponding bit in this register.

12.8.15 RapidIO Port x Interrupt Generate Register

This register can be used to generate the corresponding error in the "RapidIO Port x Interrupt Status Register" on page 318. When bits in the register are set, behavior associated with the error (port writes, interrupts) occur.

Register name: SP{0..7}_INT_GEN **Reset value: 0x0000_0000**	**Register offset: 1301C, 1311C, 1321C, 1331C, 1341C, 1351C, 1361C, 1371C**

Bits	0	1	2	3	4	5	6	7
00:07	Reserved							
08:15	Reserved					MC_TEA_GEN	LINK_INIT_NOTIFICATION_GEN	LUT_PAR_ERR_GEN
16:23	Reserved							
24:31	ILL_TRANS_GEN	Reserved	MAX_RETRY_GEN	OUTB_DEPTH_GEN	INB_DEPTH_GEN	INB_RDR_GEN	Reserved	TEA_GEN

Bits	Name	Description[a]	Type	Reset Value
0:12	Reserved	N/A	R	0
13	MC_TEA_GEN	Forces the MC_TEA bit to be set. Bit always reads as zero.	R/W1S	0
14	LINK_INIT_NOTIFICATION_GEN	Forces the LINK_INIT_NOTIFICATION bit to be set. This bit always reads as zero.	R/W1S	0
15	LUT_PAR_ERR_GEN	Force the LUT_PAR_ERR bit to be set. This bit always reads as zero.	R/W1S	0
16:23	Reserved	N/A	R	0
24	ILL_TRANS_GEN	Forces the ILL_TRANS_ERR bit to be set. Bit always reads as zero.	R/W1S	0
25	Reserved	N/A	R	0
26	MAX_RETRY_GEN	Forces the MAX_RETRY bit to be set to 1. This bit always reads as zero.	R/W1S	0
27	OUTB_DEPTH_GEN	Forces the OUTB_DEPTH bit to be set to 1. This bit always reads as zero.	R/W1S	0
28	INB_DEPTH_GEN	Forces the INB_DEPTH bit to be set to 1. This bit always reads as zero.	R/W1S	0

(Continued)

Bits	Name	Description[a]	Type	Reset Value
29	INB_RDR_GEN	Forces the INB_RDR bit to be set to 1. This bit always reads as zero.	R/W1S	0
30	Reserved	Reserved	R/W1S	0
31	TEA_GEN	Forces the TEA bit to be set to 1. This bit always reads as zero.	R/W1S	0

a. All bits in this register set/clear bits in the "RapidIO Port x Interrupt Status Register" on page 318.

Writing 0 to any bit in this register clears the corresponding bit in the "RapidIO Port x Interrupt Status Register" on page 318.

12.9 IDT-Specific Performance Registers

The registers in this section are specific to IDT's switching products. The following table shows the IDT-specific per-port registers not defined by the *RapidIO Interconnect Specification (Revision 1.3)*. It is not possible to broadcast to these registers

Table 43: IDT-Specific Per-Port Performance Registers

Port	Register Offset	Description
SP0	13000	1x/4x Serial port
SP1	13100	1x Serial port
SP2	13200	1x/4x Serial port
SP3	13300	1x Serial port
SP4	13400	1x/4x Serial port
SP5	13500	1x Serial port
SP6	13600	1x/4x Serial port
SP7	13700	1x Serial port

.

12.9.1 RapidIO Port x Performance Statistics Counter 0 and 1 Control Register

This register is used to control the performance statistics counters PS0 and PS1 registers. For every performance statistics register SPx_PSCy (where y refers to the Performance Statistics counter PS0 and PS1), the following configurations (direction, type, and priority) are selected through the SP{0..7}_PSC0n1_CTRL register:

- The PSy_DIR field determines the performance statistics receiver versus transmitter direction application.
- The PSy_TYPE field assigns the type of statistics collection (packet, control symbol, multicast, etc..) to be accumulated for a given SPx_PSCy.
- The PSy_PRIO[0..3] fields determine the priority of the packets for performance statistics collection through the SPx_PSCy register (see "RapidIO Port x Performance Statistics Counter 0 Register" on page 336 and "RapidIO Port x Performance Statistics Counter 1 Register" on page 337). The SPx_PSCy can be disabled by selecting PSy_PRIO[0..3] to be set to 0. Setting PSy_PRIO[0..3] to all ones, allows for collecting performance statistics through the SPx_PSCy for all priority packets.

Register name: SP{0..7}_PSC0n1_CTRL **Reset value: 0x0000_0000**	**Register offset: 13020, 13120, 13220, 13320, 13420, 13520, 13620, 13720**

Bits	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
00:7	PS0_PRIO 3	PS0_PRIO 2	PS0_PRIO 1	PS0_PRIO 0	Reserved			PS0_DIR
8:15	Reserved					PS0_TYPE		
16:23	PS1_PRIO 3	PS1_PRIO 2	PS1_PRIO 1	PS1_PRIO 0	Reserved			PS1_DIR
24:31	Reserved					PS1_TYPE		

Bits	**Name**	**Description**	**Type**	**Reset Value**
0	PS0_PRIO3	Performance Stats Reg PS0 Priority 3 Selection This value represents the packet priority 3 is selected for which performance statistics are accumulated for in the "RapidIO Port x Performance Statistics Counter 0 Register" on page 336. 0 = If all PS0_PRIO[0..3] are set to zero, the "RapidIO Port x Performance Statistics Counter 0 Register" on page 336 is disabled. 1 = Count priority 3 packets.	R/W	0

(Continued)

Bits	Name	Description	Type	Reset Value
1	PS0_PRIO2	Performance Stats Reg PS0 Priority 2 Selection This value represents the packet priority 2 is selected for which performance stats are accumulated for in the "RapidIO Port x Performance Statistics Counter 0 Register" on page 336. 0 = If all PS0_PRIO[0..3] are set to zero, the "RapidIO Port x Performance Statistics Counter 0 Register" on page 336 is disabled. 1 = Count priority 2 packets.	R/W	0
2	PS0_PRIO1	Performance Stats Reg PS0 Priority 1 Selection This value represents the packet priority 1 is selected for which performance stats are accumulated for in the "RapidIO Port x Performance Statistics Counter 0 Register" on page 336. 0 = If all PS0_PRIO[0..3] are set to zero, the "RapidIO Port x Performance Statistics Counter 0 Register" on page 336 is disabled. 1 = Count priority 1 packets.	R/W	0
3	PS0_PRIO0	Performance Stats Reg PS0 Priority 0 Selection This value represents the packet priority 0 is selected for which performance stats are accumulated for in the "RapidIO Port x Performance Statistics Counter 0 Register" on page 336. 0 = If all PS0_PRIO[0..3] are set to zero, the "RapidIO Port x Performance Statistics Counter 0 Register" on page 336is disabled. 1 = Count priority 0 packets.	R/W	0
4:6	Reserved	N/A	R	0
7	PS0_DIR	Performance Stats Reg PS0 Direction Selection This value selects the direction (receiver vs. transmitter) for the "RapidIO Port x Performance Statistics Counter 0 Register" on page 336. 0 = Receiver Stats Counter Register 1 = Transmitter Stats Counter Register	R/W	0
8:12	Reserved	N/A	R	0

(Continued)

Bits	Name	Description	Type	Reset Value
13:15	PS0_TYPE	Performance Stats Reg PS0 Type Selection This value determines the type of performance statistics that is collected in the "RapidIO Port x Performance Statistics Counter 0 Register" on page 336. Retries are not counted as part of the data. 000 = Count all unicast request packets only. The response packets, maintenance packets, and maintenance packets with hop count of 0 are excluded from this counter. 001 = Count all unicast packet types. This counter includes all request, response, maintenance packets (including the maintenance packets with hop count 0). 010 = Count all retry control symbols only. 011 = Count all control symbols (excluding retry control symbols). 100 = Count every 32-bits of unicast data. This counter counts all accepted unicast packets data (including header). 101 = Count all multicast packets only. 110 = Count all multicast control symbols. 111 = Count every 32-bits of multicast data. This counter includes counting the entire multicast packet (including header).	R/W	0
16	PS1_PRIO3	Performance Stats Reg PS1 Priority 3 Selection This value represents the packet priority 3 is selected for which performance stats are accumulated for in the "RapidIO Port x Performance Statistics Counter 1 Register" on page 337. 0 = If all PS1_PRIO[0..3] are set to zero, the "RapidIO Port x Performance Statistics Counter 1 Register" on page 337 is disabled. 1 = Count priority 3 packets.	R/W	0
17	PS1_PRIO2	Performance Stats Reg PS1 Priority 2 Selection This value represents the packet priority 2 is selected for which performance stats are accumulated for in the "RapidIO Port x Performance Statistics Counter 1 Register" on page 337. 0 = If all PS1_PRIO[0..3] are set to zero, the "RapidIO Port x Performance Statistics Counter 1 Register" on page 337 is disabled. 1 = Count priority 2 packets.	R/W	0
18	PS1_PRIO1	Performance Stats Reg PS1 Priority 1 Selection This value represents the packet priority 1 is selected for which performance stats are accumulated for in the "RapidIO Port x Performance Statistics Counter 1 Register" on page 337. 0 = If all PS1_PRIO[0..3] are set to zero, the "RapidIO Port x Performance Statistics Counter 1 Register" on page 337 is disabled. 1 = Count priority 1 packets.	R/W	0

(Continued)

Bits	Name	Description	Type	Reset Value
19	PS1_PRIO0	Performance Stats Reg PS1 Priority 0 Selection This value represents the packet priority 0 is selected for which performance stats are accumulated for in the "RapidIO Port x Performance Statistics Counter 1 Register" on page 337. 0 = If all PS1_PRIO[0..3] are set to zero, the "RapidIO Port x Performance Statistics Counter 1 Register" on page 337 is disabled. 1 = Count priority 0 packets.	R/W	0
20:22	Reserved	N/A	R	0
23	PS1_DIR	Performance Stats Reg PS1 Direction Selection This value selects the direction (receiver vs. transmitter) for the "RapidIO Port x Performance Statistics Counter 1 Register" on page 337. 0 = Receiver Stats Counter Register 1 =Transmitter Stats Counter Register	R/W	0
24:28	Reserved	N/A	R	0
29:31	PS1_TYPE	Performance Stats Reg PS1 Type Selection This value determines the type of performance statistics that is collected in the "RapidIO Port x Performance Statistics Counter 1 Register" on page 337. Retries are not counted as part of the data. 000 = Count all unicast request packets only. The response packets, maintenance packets, and maintenance packets with hop count of 0 are excluded from this counter. 001 = Count all unicast packet types. This counter includes all request, response, maintenance packets (including the maintenance packets with hop count 0). 101 = Count all retry control symbols only. 011 = Count all control symbols (excluding retry control symbols). 100 = Count every 32-bits of unicast data. This counter counts all accepted unicast packets data (including header). 101 = Count all multicast packets only. 110 = Count all multicast control symbols. 111 = Count every 32-bits of multicast data. This counter includes counting the entire multicast packet (including header).	R/W	0

12.9.2 RapidIO Port x Performance Statistics Counter 2 and 3 Control Register

This register is used to control the performance statistics counters PS2 and PS3 registers. For every performance stats register SPx_PSCy (where y refers to the Performance Statistics counter PS2 and PS3), the following configurations (direction, type, and priority) are selected through the SP{0..7}_PSC2n3_CTRL register:

- The PSy_DIR field determines the performance stats receiver versus transmitter direction application.
- The PSy_TYPE field assigns the type of stats collection (for example, packet, control symbol, and multicast) to be accumulated for a given SPx_PSCy.
- The PSy_PRIO[0..3] fields determine the priority of the packets for performance statistics collection through the “RapidIO Port x Performance Statistics Counter 2 Register” on page 338 and “RapidIO Port x Performance Statistics Counter 3 Register” on page 339. The SPx_PSCy can be disabled by selecting PSy_PRIO[0..3] to be set to 0. Setting PSy_PRIO[0..3] to all ones, allows for collecting performance statistics through the SPx_PSCy for all priority packets.

Register name: SP{0..7}_PSC2n3_CTRL **Reset value: 0x0000_0000**	**Register offset: 13024, 13124, 13224, 13324, 13424, 13524, 13624, 13724**

Bits	0	1	2	3	4	5	6	7
00:7	PS2_PRIO 3	PS2_PRIO 2	PS2_PRIO 1	PS2_PRIO 0	Reserved			PS2_DIR
8:15	Reserved					PS2_TYPE		
16:23	PS3_PRIO 3	PS3_PRIO 2	PS3_PRIO 1	PS3_PRIO 0	Reserved			PS3_DIR
24:31	Reserved					PS3_TYPE		

Bits	Name	Description	Type	Reset Value
0	PS2_PRIO3	Performance Stats Reg PS2 Priority 3 Selection This value represents the packet priority 3 is selected for which performance stats are accumulated for in the “RapidIO Port x Performance Statistics Counter 4 and 5 Control Register” on page 332. 0 = If all PS2_PRIO[0..3] are set to zero, the “RapidIO Port x Performance Statistics Counter 4 and 5 Control Register” on page 332 is disabled. 1 = Count priority 3 packets.	R/W	0

(Continued)

Bits	Name	Description	Type	Reset Value
1	PS2_PRIO2	Performance Stats Reg PS2 Priority 2 Selection This value represents the packet priority 2 is selected for which performance stats are accumulated for in the "RapidIO Port x Performance Statistics Counter 4 and 5 Control Register" on page 332. 0 = If all PS2_PRIO[0..3] are set to zero, the "RapidIO Port x Performance Statistics Counter 4 and 5 Control Register" on page 332 is disabled. 1 = Count priority 2 packets.	R/W	0
2	PS2_PRIO1	Performance Stats Reg PS2 Priority 1 Selection This value represents the packet priority 1 is selected for which performance stats are accumulated for in the "RapidIO Port x Performance Statistics Counter 4 and 5 Control Register" on page 332. 0 = If all PS2_PRIO[0..3] are set to zero, the "RapidIO Port x Performance Statistics Counter 4 and 5 Control Register" on page 332 is disabled. 1 = Count priority 1 packets.	R/W	0
3	PS2_PRIO0	Performance Stats Reg PS2 Priority 0 Selection This value represents the packet priority 0 is selected for which performance stats are accumulated for in the "RapidIO Port x Performance Statistics Counter 4 and 5 Control Register" on page 332. 0 = If all PS2_PRIO[0..3] are set to zero, the "RapidIO Port x Performance Statistics Counter 4 and 5 Control Register" on page 332 is disabled. 1 = Count priority 0 packets.	R/W	0
4:6	Reserved	N/A	R	0
7	PS2_DIR	Performance Stats Reg PS2 Direction Selection This value selects the direction (receiver vs. transmitter) for the "RapidIO Port x Performance Statistics Counter 4 and 5 Control Register" on page 332. 0 = Receiver Stats Counter Register 1 =Transmitter Stats Counter Register	R/W	0
8:12	Reserved	N/A	R	0

(Continued)

Bits	Name	Description	Type	Reset Value
13:15	PS2_TYPE	Performance Stats Reg PS2 Type Selection This value determines the type of performance statistics that is collected in the "RapidIO Port x Performance Statistics Counter 4 and 5 Control Register" on page 332. Retries are not counted as part of the data. 000 = Count all unicast request packets only. The response packets, maintenance packets, and maintenance packets with hop count of 0 are excluded from this counter. 001 = Count all unicast packet types. This counter includes all request, response, maintenance packets (including the maintenance packets with hop count 0). 010 = Count all retry control symbols only 011 = Count all control symbols (excluding retry control symbols). 100 = Count every 32-bits of unicast data. This counter counts all accepted unicast packets data (including header). 101 = Count all multicast packets only 110 = Count all multicast control symbols 111 = Count every 32-bits of multicast data. This counter includes counting the entire multicast packet (including header).	R/W	0
16	PS3_PRIO3	Performance Stats Reg PS3 Priority 3 Selection This value represents the packet priority 3 is selected for which performance stats are accumulated for in the "RapidIO Port x Performance Statistics Counter 3 Register" on page 339. 0 = If all PS3_PRIO[0..3] are set to zero, the "RapidIO Port x Performance Statistics Counter 3 Register" on page 339 is disabled. 1 = Count priority 3 packets	R/W	0
17	PS3_PRIO2	Performance Stats Reg PS3 Priority 2 Selection This value represents the packet priority 2 is selected for which performance stats are accumulated for in the "RapidIO Port x Performance Statistics Counter 3 Register" on page 339. 0 = If all PS3_PRIO[0..3] are set to zero, the "RapidIO Port x Performance Statistics Counter 3 Register" on page 339 is disabled. 1 = Count priority 2 packets	R/W	0
18	PS3_PRIO1	Performance Stats Reg PS3 Priority 1 Selection This value represents the packet priority 1 is selected for which performance stats are accumulated for in the "RapidIO Port x Performance Statistics Counter 3 Register" on page 339. 0 = If all PS3_PRIO[0..3] are set to zero, the "RapidIO Port x Performance Statistics Counter 3 Register" on page 339 is disabled. 1 = Count priority 1 packets	R/W	0

(Continued)

Bits	Name	Description	Type	Reset Value
19	PS3_PRIO0	Performance Stats Reg PS3 Priority 0 Selection This value represents the packet priority 0 is selected for which performance stats are accumulated for in the "RapidIO Port x Performance Statistics Counter 3 Register" on page 339. 0 = If all PS3_PRIO[0..3] are set to zero, the "RapidIO Port x Performance Statistics Counter 3 Register" on page 339 is disabled. 1 = Count priority 0 packets.	R/W	0
20:22	Reserved	N/A	R	0
23	PS3_DIR	Performance Stats Reg PS3 Direction Selection This value selects the direction (receiver vs. transmitter) for the "RapidIO Port x Performance Statistics Counter 3 Register" on page 339. 0 = Receiver Stats Counter Register 1 =Transmitter Stats Counter Register	R/W	0
24:28	Reserved	N/A	R	0
29:31	PS3_TYPE	Performance Stats Reg PS3 Type Selection This value determines the type of performance statistics that is collected in the "RapidIO Port x Performance Statistics Counter 3 Register" on page 339. Retries are not counted as part of the data. 000 = Count all unicast request packets only. The response packets, maintenance packets, and maintenance packets with hop count of 0 are excluded from this counter. 001 = Count all unicast packet types. This counter includes all request, response, maintenance packets (including the maintenance packets with hop count 0). 010 = Count all retry control symbols only. 011 = Count all control symbols (excluding retry control symbols). 100 = Count every 32-bits of unicast data. This counter counts all accepted unicast packets data (including header). 101 = Count all multicast packets only. 110 = Count all multicast control symbols. 111 = Count every 32-bits of multicast data. This counter includes counting the entire multicast packet (including header).	R/W	0

12.9.3 RapidIO Port x Performance Statistics Counter 4 and 5 Control Register

This register is used to control the performance statistics counters PS4 and PS5 registers. For every performance stats register SPx_PSCy (where y refers to the Performance Statistics counter PS4 to PS5), the following configurations (direction, type, and priority) are selected through the SP{0..7}_PSC4n5_CTRL register:

- The PSy_DIR field determines the performance stats receiver versus transmitter direction application.
- The PSy_TYPE field assigns the type of stats collection (for example, packet, control symbol, and multicast) to be accumulated for a given SPx_PSy_CTR.
- The PSy_PRIO[0..3] fields determine the priority of the packets for performance statistics collection through the "RapidIO Port x Performance Statistics Counter 4 Register" on page 340 and "RapidIO Port x Performance Statistics Counter 5 Register" on page 341. The SPx_PSCy can be disabled by selecting PSy_PRIO[0..3] to be set to 0. Setting PSy_PRIO[0..3] to all ones, allows for collecting performance statistics through the SPx_PSCy for all priority packets.

Register name: SP{0..7}_PSC4n5_CTRL **Reset value: 0x0000_0000**	**Register offset: 13028, 13128, 13228, 13328, 13428, 13528, 13628, 13728**

Bits	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
00:7	PS4_PRIO 3	PS4_PRIO 2	PS4_PRIO 1	PS4_PRIO 0	Reserved			PS4_DIR
8:15	Reserved					PS4_TYPE		
16:23	PS5_PRIO 3	PS5_PRIO 2	PS5_PRIO 1	PS5_PRIO 0	Reserved			PS5_DIR
24:31	Reserved					PS5_TYPE		

Bits	**Name**	**Description**	**Type**	**Reset Value**
0	PS4_PRIO3	Performance Stats Reg PS4 Priority 3 Selection This value represents the packet priority 3 is selected for which performance stats are accumulated for in the "RapidIO Port x Performance Statistics Counter 4 Register" on page 340. 0 = If all PS4_PRIO[0..3] are set to zero, the "RapidIO Port x Performance Statistics Counter 4 Register" on page 340 is disabled. 1 = Count priority 3 packets.	R/W	0

(Continued)

Bits	Name	Description	Type	Reset Value
1	PS4_PRIO2	Performance Stats Reg PS4 Priority 2 Selection This value represents the packet priority 2 is selected for which performance stats are accumulated for in the "RapidIO Port x Performance Statistics Counter 4 Register" on page 340. 0 = If all PS4_PRIO[0..3] are set to zero, the "RapidIO Port x Performance Statistics Counter 4 Register" on page 340 is disabled. 1 = Count priority 2 packets.	R/W	0
2	PS4_PRIO1	Performance Stats Reg PS4 Priority 1 Selection This value represents the packet priority 1 is selected for which performance stats are accumulated for in the "RapidIO Port x Performance Statistics Counter 4 Register" on page 340. 0 = If all PS4_PRIO[0..3] are set to zero, the "RapidIO Port x Performance Statistics Counter 4 Register" on page 340 is disabled. 1 = Count priority 1 packets.	R/W	0
3	PS4_PRIO0	Performance Stats Reg PS4 Priority 0 Selection This value represents the packet priority 0 is selected for which performance stats are accumulated for in the "RapidIO Port x Performance Statistics Counter 4 Register" on page 340. 0 = If all PS4_PRIO[0..3] are set to zero, the "RapidIO Port x Performance Statistics Counter 4 Register" on page 340 is disabled. 1 = Count priority 0 packets.	R/W	0
4:6	Reserved	N/A	R	0
7	PS4_DIR	Performance Stats Reg PS4 Direction Selection This value selects the direction (receiver vs. transmitter) for the "RapidIO Port x Performance Statistics Counter 4 Register" on page 340. 0 = Receiver Stats Counter Register 1 = Transmitter Stats Counter Register	R/W	0
8:12	Reserved	N/A	R	0

(Continued)

Bits	Name	Description	Type	Reset Value
13:15	PS4_TYPE	Performance Stats Reg PS4 Type Selection This value determines the type of performance statistics that is collected in the "RapidIO Port x Performance Statistics Counter 4 Register" on page 340. Retries are not counted as part of the data. 000 = Count all unicast request packets only. The response packets, maintenance packets, and maintenance packets with hop count of 0 are excluded from this counter. 001 = Count all unicast packet types. This counter includes all request, response, maintenance packets (including the maintenance packets with hop count 0). 010 = Count all retry control symbols only[a] 011 = Count all control symbols (excluding retry control symbols)[1]. 100 = Count every 32-bits of unicast data. This counter counts all accepted unicast packets data (including header). 101 = Count all multicast packets only. 110 = Count all multicast control symbols. 111 = Count every 32-bits of multicast data. This counter includes counting the entire multicast packet (including header).	R/W	0
16	PS5_PRIO3	Performance Stats Reg PS5 Priority 3 Selection This value represents the packet priority 3 is selected for which performance stats are accumulated for in the "RapidIO Port x Performance Statistics Counter 5 Register" on page 341. 0 = If all PS5_PRIO[0..3] are set to zero, the "RapidIO Port x Performance Statistics Counter 5 Register" on page 341 is disabled. 1 = Count priority 3 packets.	R/W	0
17	PS5_PRIO2	Performance Stats Reg PS5 Priority 2 Selection This value represents the packet priority 2 is selected for which performance stats are accumulated for in the "RapidIO Port x Performance Statistics Counter 5 Register" on page 341. 0 = If all PS5_PRIO[0..3] are set to zero, the "RapidIO Port x Performance Statistics Counter 5 Register" on page 341 is disabled. 1 = Count priority 2 packets.	R/W	0
18	PS5_PRIO1	Performance Stats Reg PS5 Priority 1 Selection This value represents the packet priority 1 is selected for which performance stats are accumulated for in the "RapidIO Port x Performance Statistics Counter 5 Register" on page 341. 0 = If all PS5_PRIO[0..3] are set to zero, the "RapidIO Port x Performance Statistics Counter 5 Register" on page 341 is disabled. 1 = Count priority 1 packets.	R/W	0

(Continued)

Bits	Name	Description	Type	Reset Value
19	PS5_PRIO0	Performance Stats Reg PS5 Priority 0 Selection This value represents the packet priority 0 is selected for which performance stats are accumulated for in the "RapidIO Port x Performance Statistics Counter 5 Register" on page 341. 0 = If all PS5_PRIO[0..3] are set to zero, the "RapidIO Port x Performance Statistics Counter 5 Register" on page 341 is disabled. 1 = Count priority 0 packets.	R/W	0
20:22	Reserved	N/A	R	0
23	PS5_DIR	Performance Stats Reg PS5 Direction Selection This value selects the direction (receiver vs. transmitter) for the "RapidIO Port x Performance Statistics Counter 5 Register" on page 341. 0 = Receiver Stats Counter Register 1 =Transmitter Stats Counter Register	R/W	0
24:28	Reserved	N/A	R	0
29:31	PS5_TYPE	Performance Stats Reg PS5 Type Selection This value determines the type of performance statistics that is collected in the "RapidIO Port x Performance Statistics Counter 5 Register" on page 341. Retries are not counted as part of the data. 000 = Count all unicast request packets only. The response packets, maintenance packets, and maintenance packets with hop count of 0 are excluded from this counter. 001 = Count all unicast packet types. This counter includes all request, response, maintenance packets (including the maintenance packets with hop count 0). 010 = Count all retry control symbols only. 011 = Count all control symbols (excluding retry control symbols). 100 = Count every 32-bits of unicast data. This counter counts all accepted unicast packets data (including header). 101 = Count all multicast packets only. 110 = Count all multicast control symbols. 111 = Count every 32-bits of multicast data. This counter includes counting the entire multicast packet (including header).	R/W	0

a. The Control Symbol has no priority. In this case, any non-zero setting in bit 0-3 increments the counter.

12.9.4 RapidIO Port x Performance Statistics Counter 0 Register

This register is used to collect performance statistics. These counters provide the means of accumulating statistics for the purposes of performance monitoring measurements: throughput and latency.

The PS0_CTR counter collects performance statistics information based on the configuration fields specified in the “RapidIO Port x Performance Statistics Counter 0 and 1 Control Register” on page 324.

The PS0_CTR counter value is writable for testing purposes.This counter saturates when it reaches its maximum value 0xFFFFFFFF and is cleared on a read. The PS0_CTR is enabled, when PS0_PRIO[0..3] value in the “RapidIO Port x Performance Statistics Counter 0 and 1 Control Register” on page 324 is configured to a value other than 0.

Register name: SP{0..7}_PSC0 **Reset value: 0x0000_0000**	**Register offset: 13040, 13140, 13240, 13340, 13440, 13540, 13640, 13740**

Bits	0	1	2	3	4	5	6	7
00:7	PS0_CTR							
8:15	PS0_CTR							
16:23	PS0_CTR							
24:31	PS0_CTR							

Bits	Name	Description	Type	Reset Value
0:31	PS0_CTR	This counter is used to collect performance statistics based on the configurations specified through the “RapidIO Port x Performance Statistics Counter 0 and 1 Control Register” on page 324 A read clears this register.	R/W	0

12.9.5 RapidIO Port x Performance Statistics Counter 1 Register

This register is used to collect performance statistics. These counters provide the means of accumulating statistics for the purposes of performance monitoring measurements: throughput and latency.

The PS1_CTR counter collects performance statistics information based on the configuration fields specified in the “RapidIO Port x Performance Statistics Counter 0 and 1 Control Register” on page 324.

The PS1_CTR counter value is writable for testing purposes. This counter saturates when it reaches its maximum value 0xFFFFFFFF and is cleared on a read. The PS1_CTR is enabled, when PS1_PRIO[0..3] value in the “RapidIO Port x Performance Statistics Counter 0 and 1 Control Register” on page 324 is configured to a value other than 0.

Register name: SP{0..7}_PSC1 **Reset value: 0x0000_0000**	**Register offset: 13044, 13144, 13244, 13344, 13444, 13544, 13644, 13744**

Bits	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
00:7	PS1_CTR							
8:15	PS1_CTR							
16:23	PS1_CTR							
24:31	PS1_CTR							

Bits	**Name**	**Description**	**Type**	**Reset Value**
0:31	PS1_CTR	This counter is used to collect performance statistics based on the configurations specified through the “RapidIO Port x Performance Statistics Counter 0 and 1 Control Register” on page 324. A read clears this register.	R/W	0

12.9.6 RapidIO Port x Performance Statistics Counter 2 Register

This register is used to collect performance statistics. These counters provide the means of accumulating statistics for the purposes of performance monitoring measurements: throughput and latency.

The PS2_CTR counter collects performance statistics information based on the configuration fields specified in the “RapidIO Port x Performance Statistics Counter 2 and 3 Control Register” on page 328 register.

The PS2_CTR counter value is writable for testing purposes. This counter saturates when it reaches its maximum value 0xFFFFFFFF and is cleared on a read. The PS2_CTR is enabled, when PS2_PRIO[0..3] value in the“RapidIO Port x Performance Statistics Counter 2 and 3 Control Register” on page 328 is configured to a value other than 0.

Register name: SP{0..7}_PSC2 **Reset value: 0x0000_0000**	**Register offset: 13048, 13148, 13248, 13348, 13448, 13548, 13648, 13748**

Bits	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
00:7	PS2_CTR							
8:15	PS2_CTR							
16:23	PS2_CTR							
24:31	PS2_CTR							

Bits	**Name**	**Description**	**Type**	**Reset Value**
0:31	PS2_CTR	This counter is used to collect performance statistics based on the configurations specified through the “RapidIO Port x Performance Statistics Counter 2 and 3 Control Register” register. A read clears this register.	R/W	0

12.9.7 RapidIO Port x Performance Statistics Counter 3 Register

This register is used to collect performance statistics. These counters provide the means of accumulating statistics for the purposes of performance monitoring measurements: throughput and latency.

The PS3_CTR counter collects performance statistics information based on the configuration fields specified in the “RapidIO Port x Performance Statistics Counter 2 and 3 Control Register” on page 328.

The PS3_CTR counter value is writable for testing purposes. This counter saturates when it reaches its maximum value 0xFFFFFFFF and is cleared on a read. The PS3_CTR is enabled, when PS3_PRIO[0..3] value in the “RapidIO Port x Performance Statistics Counter 2 and 3 Control Register” on page 328 is configured to a value other than 0.

Register name: SP{0..7}_PSC3 **Reset value: 0x0000_0000**	**Register offset: 1304C, 1314C, 1324C, 1334C, 1344C, 1354C, 1364C, 1374C**

Bits	0	1	2	3	4	5	6	7
00:7	PS3_CTR							
8:15	PS3_CTR							
16:23	PS3_CTR							
24:31	PS3_CTR							

Bits	Name	Description	Type	Reset Value
0:31	PS3_CTR	This counter is used to collect performance statistics based on the configurations specified through the “RapidIO Port x Performance Statistics Counter 2 and 3 Control Register” on page 328. A read clears this register.	R/W	0

12.9.8 RapidIO Port x Performance Statistics Counter 4 Register

This register is used to collect performance statistics. These counters provide the means of accumulating statistics for the purposes of performance monitoring measurements: throughput and latency.

The PS4_CTR counter collects performance statistics information based on the configuration fields specified in the “RapidIO Port x Performance Statistics Counter 4 and 5 Control Register” on page 332.

The PS4_CTR counter value is writable for testing purposes.This counter saturates when it reaches its maximum value 0xFFFFFFFF and is cleared on a read. The PS4_CTR is enabled, when PS4_PRIO[0..3] value in the “RapidIO Port x Performance Statistics Counter 4 and 5 Control Register” on page 332) register is configured to a value other than 0.

Register name: SP{0..7}_PSC4 **Reset value: 0x0000_0000**	**Register offset: 13050, 13150, 13250, 13350, 13450, 13550, 13650, 13750**

Bits	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
00:7	PS4_CTR							
8:15	PS4_CTR							
16:23	PS4_CTR							
24:31	PS4_CTR							

Bits	**Name**	**Description**	**Type**	**Reset Value**
0:31	PS4_CTR	This counter is used to collect performance statistics based on the configurations specified through the “RapidIO Port x Performance Statistics Counter 4 and 5 Control Register” on page 332. A read clears this register.	R/W	0

12.9.9 RapidIO Port x Performance Statistics Counter 5 Register

This register is used to collect performance statistics. These counters provide the means of accumulating statistics for the purposes of performance monitoring measurements: throughput and latency.

The PS5_CTR counter collects performance statistics information based on the configuration fields specified in the SPx_PSy_CTRL1 (“RapidIO Port x Performance Statistics Counter 4 and 5 Control Register”) register.

The PS5_CTR counter value is writable for testing purposes. This counter saturates when it reaches its maximum value 0xFFFFFFFF and is cleared on a read. The PS5_CTR is enabled, when PS5_PRIO[0..3] value in the “RapidIO Port x Performance Statistics Counter 4 and 5 Control Register” is configured to a value other than 0.

Register name: SP{0..7}_PSC5 **Reset value: 0x0000_0000**	**Register offset: 13054, 13154, 13254, 13354, 13454, 13554, 13654, 13754**

Bits	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
00:7	PS5_CTR							
8:15	PS5_CTR							
16:23	PS5_CTR							
24:31	PS5_CTR							

Bits	**Name**	**Description**	**Type**	**Reset Value**
0:31	PS5_CTR	This counter is used to collect performance statistics based on the configurations specified through the “RapidIO Port x Performance Statistics Counter 4 and 5 Control Register” on page 332. A read clears this register.	R/W	0

12.9.10 RapidIO Port x Transmitter Output Queue Depth Threshold Register

Queue depth registers are designed to allow for the rapid detection and notification of congestion.

This register sets the Transmitter Queue Depth threshold, which is used in conjunction with "RapidIO Port x Transmitter Output Queue Congestion Status Register" to monitor congestion on the output buffers.

This register also sets the CONG_PERIOD, which is used in conjunction with the "RapidIO Port x Transmitter Output Queue Congestion Period Register" to determine how long the output buffers have been in a congestion state.

Register name: SP{0..7}_TX_Q_D_THRESH **Reset value: 0x0000_0000**	**Register offset: 13080, 13180, 13280, 13380, 13480, 13580, 13680, 13780**

Bits	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
00:7	CONG_PERIOD							
8:15	CONG_PERIOD							
16:23	DEPTH				Reserved	LEAK_RT		
24:31	LEAK_RT							

Bits	**Name**	**Description**	**Type**	**Reset Value**
0:15	CONG_PERIOD	This value is programmed by software to indicate the maximum number of clock cycles that the output buffer can be in a continuous congestion state. The congestion state is determined based on the DEPTH and "RapidIO Port x Transmitter Output Queue Congestion Status Register" on page 344. The programmed CONG_PERIOD value is then used as follows: 0000 = CONG_PERIOD_CTR (in "RapidIO Port x Transmitter Output Queue Congestion Period Register" on page 346) is disabled. 0001= For every clock cycle that the output buffer is in continuous congestion state, increment the CONG_PERIOD_CTR by 1. For example, for a 4x port operating at 3.125 Gbaud/s, this is 3.2 ns. FFFF = For every 64K clock cycles that the output buffer is in continuous congestion state, increment the CONG_PERIOD_CTR by 1. For example, for a 4x port operating at 3.125 Gbaud/s, this is 209.7 us	R/W	0x0000

(Continued)

Bits	Name	Description	Type	Reset Value
16:19	DEPTH	This number is used to decide the congestion state of the output buffers. If the number of packets in the output queue meets or exceeds this number, the congestion counter is incremented. 0 = Disables congestion monitoring 1 = The congestion counter increments when the buffer fill reaches or exceeds 2 2 = The congestion counter increments when the buffer fill reaches or exceeds 3 …… 7 = the congestion counter increments when the buffer fill reaches 8 8 = Reserved	R/W	0x0
20	Reserved	N/A	R	0
21:31	LEAK_RT	This value is the leak rate for both the receiver and transmitter congestion counters. Whenever this time period expires, the CONG_CTR values for both transmitter (“RapidIO Port x Transmitter Output Queue Congestion Status Register” on page 344) and receiver (“RapidIO Port x Receiver Input Queue Congestion Status Register” on page 349) are decremented by 1. 0 = Leak rate is disabled 1 = Decrement CONG_CTR every 1.28µs 2 = Decrement CONG_CTR every 2*1.28µs = 2.56µs ... 2047 = Decrement CONG_CTR every 2047*1.28µs = 2.62ms.	R/W	0x0

12.9.11 RapidIO Port x Transmitter Output Queue Congestion Status Register

This register is used to monitor data congestion in the output buffer. New packets accumulate in the output buffers, destined for the switching fabric. When the number of buffers in use equals or exceeds the threshold set in DEPTH field of the "RapidIO Port x Transmitter Output Queue Depth Threshold Register", the CONG_CTR field in this register is incremented.

The CONG_CTR counter value is writable for testing purposes. This counter stops counting when it reaches its maximum value. Writing 1 into the OUTB_DEPTH field in the "RapidIO Port x Interrupt Status Register" on page 318 interrupt status bit causes this counter to be reset to 0. The CONG_CTR is enabled, when CONG_THRESH value is configured to a value other than 0. The CONG_CTR value is decremented by 1 if it is not read within the Error Rate Bias frequency as specified by the ERR_RB field in the "RapidIO Port x Error Rate CSR" on page 296

If the CONG_CTR equals or exceeds the threshold CONG_THRESH, the maskable OUTB_DEPTH interrupt is generated.

Register name: SP{0..7}_TX_Q_STATUS **Reset value: 0x0000_0000**	**Register offset: 13084, 13184, 13284, 13384, 13484, 13584, 13684, 13784**

Bits	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
00:7	CONG_CTR							
8:15	CONG_CTR							
16:23	CONG_THRESH							
24:31	CONG_THRESH							

Bits	**Name**	**Description**	**Type**	**Reset Value**
0:15	CONG_CTR	Output Queue Depth Count The number of times that the output queue exceeds the threshold DEPTH field of the "RapidIO Port x Transmitter Output Queue Depth Threshold Register" on page 342. The count is incremented by 1 when a packet is received. This counter counts up to 0xFFFFand remains at 0xFFFF until reset. The counter is reset when 1 is written to the OUTB_DEPTH (see "RapidIO Port x Interrupt Status Register" on page 318) status bit. The counter is enabled if CONG_THRESH is set to a value other than 0.The CONG_CTR value is decremented by 1 whenever the LEAK_RT [see "RapidIO Port x Transmitter Output Queue Depth Threshold Register" on page 342] time period expires. The CONG_CTR value never goes below 0.	R/W	0x0000

(Continued)

Bits	Name	Description	Type	Reset Value
16:31	CONG_THRESH	Output Queue Depth Threshold If the CONG_CTR count is equal to the value in this field, an interrupt is reported to the system through the OUTB_DEPTH status bit in the “RapidIO Port x Interrupt Status Register” on page 318. Setting the CONG_THRES to zero, disables the CONG_CTR.	R/W	0x0000

12.9.12 RapidIO Port x Transmitter Output Queue Congestion Period Register

This register is used to monitor the duration of time that the output buffer is in congestion state.

The CONG_PERIOD_CTR counter value is incremented for every N clock cycles specified by the CONG_PERIOD field in the "RapidIO Port x Transmitter Output Queue Depth Threshold Register", while the output buffer is under congestion state. This counter represents the amount of time that the output buffer is under congestion state.

The CONG_PERIOD_CTR counter value is writable for testing purposes. This counter stops counting when it reach its maximum value. Reading the CONG_PERIOD_CTR clears the counter value. The CONG_PERIOD_CTR can be disabled when the CONG_PERIOD field in the "RapidIO Port x Transmitter Output Queue Depth Threshold Register" is set to 0.

Register name: SP{0..7}_TX_Q_PERIOD **Reset value: 0x0000_0000**	**Register offset: 13088, 13188, 13288, 13388, 13488, 13588, 13688, 13788**

Bits	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
00:7	CONG_PERIOD_CTR							
8:15	CONG_PERIOD_CTR							
16:23	CONG_PERIOD_CTR							
24:31	CONG_PERIOD_CTR							

Bits	**Name**	**Description**	**Type**	**Reset Value**
0:31	CONG_PERIOD_CTR	Output Queue Congestion Period Count Each time the output buffer enters a congestion state, this counter is incremented for every N clock cycles (as specified in CONG_PERIOD field of the "RapidIO Port x Transmitter Output Queue Depth Threshold Register". This counter counts up to 0xFFFFand remains at 0xFFFF until reset. Note: The counter is reset when read. The counter is enabled if CONG_PERIOD is set to a value other than 0.	R/W	0x0000

12.9.13 RapidIO Port x Receiver Input Queue Depth Threshold Register

Queue depth registers are designed to allow for the rapid detection and notification of congestion.

This register sets the Receiver Queue Depth threshold, which is used in conjunction with "RapidIO Port x Receiver Input Queue Congestion Status Register" to monitor congestion on the input buffers.

This register also sets the CONG_PERIOD, which is used in conjunction with the "RapidIO Port x Receiver Input Queue Congestion Period Register" to determine how long the input buffers have been in a congestion state.

Register name: SP{0..7}_RX_Q_D_THRESH **Reset value: 0x0000_0000**	**Register offset: 13090, 13190, 13290, 13390, 13490, 13590, 13690, 13790**

Bits	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
00:7	CONG_PERIOD							
8:15	CONG_PERIOD							
16:23	DEPTH				Reserved			
24:31	Reserved							

Bits	**Name**	**Description**	**Type**	**Reset Value**
0:15	CONG_PERIOD	This value is programmed by SW to indicate the maximum number of clock cycles that the output buffer can be in a continuous congestion state. The congestion state is determined based on the DEPTH and SPx_RX_Q_STATUS. The programmed CONG_PERIOD value is then used as follows: 0000 = CONG_PERIOD_CTR is disabled. 0001= For every clock cycle (for a 4x port operating at 3.125 Gbaud, this is 3.2 ns) that the output buffer is in continuous congestion state, increment the CONG_PERIOD_CTR by 1. FFFF = For every 64K clock cycles (for a 4x port operating at 3.125 Gbaud, this is 209.7 usec) that the output buffer is in continuous congestion state, increment the CONG_PERIOD_CTR by 1.	R/W	0x0000

(Continued)

Bits	Name	Description	Type	Reset Value
16:19	DEPTH	This number is used to decide the congestion state of the input buffers. If the number of packets in the input queue meets or exceeds this number, the congestion counter is incremented. 0 = Disables congestion monitoring 1 = The congestion counter increments when the buffer fill reaches or exceeds 2 2 = The congestion counter increments when the buffer fill reaches or exceeds 3 …… 7 = the congestion counter increments when the buffer fill reaches 8 8 = Reserved	R/W	0
20:31	Reserved	N/A	R	0

12.9.14 RapidIO Port x Receiver Input Queue Congestion Status Register

This register is used to monitor data congestion in the input buffer.

New packets accumulate in the input buffers, destined for the switching fabric. When the number of buffers in use equals or exceeds the threshold set in DEPTH field of the "RapidIO Port x Receiver Input Queue Depth Threshold Register", the CONG_CTR field in SPx_R_Q_STATUS is incremented.

The CONG_CTR counter value is writable for testing purposes. This counter stops counting when it reaches its maximum value. Writing 1 into the INB_DEPTH (see "RapidIO Port x Interrupt Status Register" on page 318) interrupt status bit causes this counter to be reset to 0. The CONG_CTR is enabled, when CONG_THRESH value is configured to a value other than 0. The CONG_CTR value is decremented by 1 if it is not read within the Error Rate Bias frequency as specified by the ERR_RB field in the "RapidIO Port x Error Rate CSR" on page 296.

If the CONG_CTR equals or exceeds the threshold CONG_THRESH, the maskable INB_DEPTH interrupt is generated.

Register name: SP{0..7}_RX_Q_STATUS **Reset value: 0x0000_0000**	**Register offset: 13094, 13194, 13294, 13394, 13494, 13594, 13694, 13794**

Bits	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
00:7	CONG_CTR							
8:15	CONG_CTR							
16:23	CONG_THRESH							
24:31	CONG_THRESH							

Bits	**Name**	**Description**	**Type**	**Reset Value**
0:15	CONG_CTR	Input Queue Depth Count The number of times that the input queue meets or exceeds the threshold DEPTH field of the "RapidIO Port x Receiver Input Queue Depth Threshold Register" on page 347. The count is incremented by 1 when a packet is received. This counter counts up to 0xFFFFand remains at 0xFFFF until reset. The counter is reset when 1 is written to the INB_DEPTH status bit (see "RapidIO Port x Interrupt Status Register" on page 318). The counter is enabled if CONG_THRES is set to a value other than 0. The CONG_CTR value is decremented by 1 if it is not read within the Error Rate Bias frequency as specified by the ERR_RB field in the "RapidIO Port x Error Rate CSR" on page 296.	R/W	0x0000

(Continued)

Bits	Name	Description	Type	Reset Value
16:31	CONG_THRESH	Input Queue Depth Threshold If the CONG_CTR count is equal to the value in this field, an interrupt is reported to the system through the INB_DEPTH status bit in the “RapidIO Port x Interrupt Status Register” on page 318. Setting the CONG_THRESH to zero, disables the CONG_CTR.	R/W	0x0000

12.9.15 RapidIO Port x Receiver Input Queue Congestion Period Register

This register is used to monitor the duration of time that the input buffer is in congestion state.

The CONG_PERIOD_CTR counter value is incremented for every N clock cycles specified by the CONG_PERIOD field in the "RapidIO Port x Receiver Input Queue Depth Threshold Register", while the input buffer is under congestion state. This counter represents the amount of time that the input buffer is under congestion state.

The CONG_PERIOD_CTR counter value is writable for testing purposes. This counter stops counting when it reach its maximum value. Reading the CONG_PRIOD_CTR clears the register. The CONG_PERIOD_CTR can be disabled when the CONG_PERIOD field in the "RapidIO Port x Receiver Input Queue Depth Threshold Register" is set to 0.

Register name: SP{0..7}_RX_Q_PERIOD **Reset value: 0x0000_0000**	**Register offset: 13098, 13198, 13298, 13398, 13498, 13598, 13698, 13798**

Bits	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
00:7	CONG_PERIOD_CTR							
8:15	CONG_PERIOD_CTR							
16:23	CONG_PERIOD_CTR							
24:31	CONG_PERIOD_CTR							

Bits	**Name**	**Description**	**Type**	**Reset Value**
0:31	CONG_PERIOD_CTR	Input Queue Congestion Period Count Each time the input buffer enters a congestion state, this counter is incremented for every N clock cycles (as specified in CONG_PERIOD field of the "RapidIO Port x Transmitter Output Queue Depth Threshold Register"). This counter counts up to 0xFFFFand remains at 0xFFFF until reset. The counter is reset when read. The counter is enabled if CONG_PERIOD is set to a value other than 0. Note: Reading clears this register	R/W	0

12.9.16 RapidIO Port x Reordering Counter Register

When a packet cannot make forward progress due to internal switching congestion, the internal switching fabric selects packets in an order different from the order in which the packets were received. Each time this happens, it is counted as a "reorder" event in this register.

The Input Reordering Threshold (THRESH) defines the number of times the Input Reordering Count is incremented before an interrupt is generated, if enabled (see "RapidIO Port x Interrupt Status Register" on page 318).

Register name: SP{0..7}_REORDER_CTR **Reset value: 0x0000_FFFF**	**Register offset: 130A0, 131A0, 132A0, 133A0, 134A0, 135A0, 136A0, 137A0**

Bits	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
00:7	CTR							
8:15	CTR							
16:23	THRESH							
24:31	THRESH							

Bits	**Name**	**Description**	**Type**	**Reset Value**
0:15	CTR	Reorder Counter This counter is updated every time the input queue is reordered. This counter counts up to 0xFFFFand remains at 0xFFFF until reset. The counter is reset when 1 is written to the INB_RDR status bit in the "RapidIO Port x Interrupt Status Register" on page 318. The counter is enabled if the THRESH is configured to a value other than 0.	R/W	0x0000
16:31	THRESH	Input Reordering Threshold When CTR equals THRESH, the maskable interrupt "INB_RDR" in the "RapidIO Port x Interrupt Status Register" on page 318 is generated. Setting the THRESH value to 0 disables the CTR.	R/W	0xFFFF

12.10 Serial Port Electrical Layer Registers

The Serial Port Electrical Layer Registers are not defined in the *RapidIO Interconnect Specification (Revision 1.3)*. They are specific to IDT's switching products.

These registers are reset by the HARD_RST_b reset input signal, as well as when the Tsi574 performs a self-reset. The registers within a port are also reset by a "Port Reset". For more information on Tsi574 reset implementation and behavior, see "Clocks, Resets and Power-up Options" on page 203.

It is possible to override reset values of writable fields, and some read-only fields, using the I^2C register loading capability on boot. Refer to "I^2C Interface" on page 139 for more information on the use of I^2C controller register loading capability.

Software must not access reserved addresses or bits, because this can affect device operation in non-deterministic ways.

Table 44: IDT-Specific RapidIO Registers

Port	Register Offset	Description
BC	10000	Broadcast addresses. These registers affect all the ports.
SP0	11000	1x/4x serial port
SP1	11100	1x serial port
SP2	11200	1x/4x serial port
SP3	11300	1x serial port
SP4	11400	1x/4x serial port
SP5	11500	1x serial port
SP6	11600	1x/4x serial port
SP7	11700	1x serial port

The registers in the following table are accessible even when the serial RapidIO ports are in reset or powered down.

Table 45: Serial Port Electrical Layer Registers

MAC	Register Offset	Description
MAC0	130B0	Ports 0 and 1
MAC2	132B0	Ports 2 and 3

Table 45: Serial Port Electrical Layer Registers

MAC	Register Offset	Description
MAC4	134B0	Ports 4 and 5
MAC6	136B0	Ports 6 and 7

12.10.1 BYPASS_INIT Functionality

The traffic affecting SerDes controls are locked by the BYPASS_INIT bit "SRIO MAC x SerDes Configuration Global" on page 364. The following bits and fields are unlocked when the BYPASS_INIT bit is set to 1:

- MPLL_CK_OFF in the SMACx_CFG_GBL register
- SERDES_RESET in the SMACx_CFG_GBL register
- MPLL_PWRON in the SMACx_CFG_GBL register
- TX_EN in the SMACx_CFG_CH3 register
- TX_EN in the SMACx_CFG_CH2 register
- TX_EN in the SMACx_CFG_CH1 register
- TX_EN in the SMACx_CFG_CH0 register
- RX_PLL_PWRON in the SMACx_CFG_CH0 register
- RX_PLL_PWRON in the SMACx_CFG_CH1 register
- RX_PLL_PWRON in the SMACx_CFG_CH2 register
- RX_PLL_PWRON in the SMACx_CFG_CH3 register
- RX_EN in the SMACx_CFG_CH0 register
- RX_EN in the SMACx_CFG_CH1 register
- RX_EN in the SMACx_CFG_CH2 register
- RX_EN in the SMACx_CFG_CH3 register

12.10.2 SRIO MAC x SerDes Configuration Channel 0

This register is used to control serial port SerDes channel 0. For more details on port configuration after power down, see “Port Power Down” on page 71.

Register name: SMAC{0,2,4,6}_CFG_CH0 **Reset value: Undefined**	**Register offset: 130B0, 132B0, 134B0, 136B0**

Bits	0	1	2	3	4	5	6	7
00:07	HALF_RATE	TX_CALC	Unused					
08:15	Reserved	TX_EN[2:0]			TX_BOOST[3:0]			
16:23	RX_PLL_PWRON	RX_EN	DPLL_RESET	Reserved		RX_EQ_VAL[2:0]		
24:31	RX_DPLL_MODE[2:0]			TX_CKO_EN	LOS_CTL[1:0]		RX_ALIGN_EN	Reserved

Bits	Name	Description	Type	Reset Value
0	HALF_RATE	Baud Rate Control 0 = Running at 2.5Gbps and 3.125Gbps 1 = Running at 1.25Gbps This bit corresponds to the SerDes PLL divider setting selected by the IO_SPEED field in the “SRIO MAC x Digital Loopback and Clock Selection Register” on page 369. Caution: This field should not be independently modified. Changing it can lead to unpredictable behavior.	R/W	Undefined
1	TX_CALC	Transmitter Calculation A rising edge causes a recalculation of the transmitter attenuation and boost configuration. The rising edge can be generated by writing a 0 and then a 1 to the register bit.	R/W	0
2:7	Unused	N/A	R/W	0x4
8	Reserved	N/A	R	0
9:11	TX_EN[2:0]	Transmitter enable 000 = off, no clocks running 011 = transmitter fully enabled, all clocks running Other values are Reserved. This bit is read only unless BYPASS_INIT bit “SRIO MAC x SerDes Configuration Global” on page 364 is set to 1.	R/W	0x3

(Continued)

Bits	Name	Description	Type	Reset Value
12:15	TX_BOOST[3:0]	Transmit Boost control Programmed boost value (ratio of drive level of transition bit to non-transition bit) is: boost = -20*log(1-(tx_boost[3:0]+0.5)/32)dB, except that setting tx_boost to 0 produces 0dB of boost. This produces results up to 5.75dB in steps of ~0.37dB	R/W	0xC
16	RX_PLL_PWRON	Power up/reset the receive PLL 0 = Rx PLL off 1 = Rx PLL on This bit is read only unless BYPASS_INIT bit "SRIO MAC x SerDes Configuration Global" on page 364 is set to 1.	R/W	0x1
17	RX_EN	Enable receive clock and data outputs RX_EN can only be asserted when both RX_PLL_PWRON and RX_PLL_STATE are both asserted. Set to 1 to enable Receive Data; 0 to disable Receive Data This bit is read only unless BYPASS_INIT bit "SRIO MAC x SerDes Configuration Global" on page 364 is set to 1.	R/W	0x1
18	DPLL_RESET	A rising edge resets the frequency register of the DPLL. The rising edge may be generated by writing a 0 and then a 1 to the register bit.	R/W	0x1
19:20	Reserved	N/A	R/W	0
21:23	RX_EQ_VAL[2:0]	Receive Equalization control Internal linear equalizer boost is approximately = (rx_eq_val+1)*0.5dB Example: 3'b100 = 2.5dB boost	R/W	0x5
24:26	RX_DPLL_MODE[2:0]	DPLL Mode Selection When RX_EN is not asserted, this can change any time. This should not change when RX_EN is asserted,	R/W	0x0
27	TX_CKO_EN	0 = Port is not in used. 1 = Power-up clock for Transmit Domain. Must be set 1 even when only one lane is in use.	R/W	1
28:29	LOS_CTL[1:0]	Enable LOS detector and/of filtering of raw LOS output 00 = LOS detector disabled 11 = Heavy filtering for generic LOS signaling Other values are reserved.	R/W	0x0
30	RX_ALIGN_EN	Enable Word Alignment 0 = Alignment (framer) disabled 1 = Alignment enabled Note: Must be disabled during PRBS test (see "Bit Error Rate Testing (BERT)" on page 79)	R/W	1

(Continued)

Bits	Name	Description	Type	Reset Value
31	Reserved	N/A Note: Only write 1 to this reserved field.	R/W	1

12.10.3 SRIO MAC x SerDes Configuration Channel 1

This register is used to control serial port SerDes channel 1. For more details on port configuration after power down, refer to “Port Power Down” on page 71.

Register name: SMAC{0,2,4,6}_CFG_CH1 **Reset value: Undefined**	**Register offset: 130B4, 132B4, 134B4, 136B4**

Bits	0	1	2	3	4	5	6	7
00:07	HALF_ RATE	TX_CALC	Unused					
08:15	Reserved	TX_EN[2:0]			TX_BOOST[3:0]			
16:23	RX_PLL_P WRON	RX_EN	DPLL_RES ET	Reserved		RX_EQ_VAL[2:0]		
24:31	RX_DPLL_MODE[2:0]			TX_CKO_ EN	LOS_CTL[1:0]		RX_ALIGN _EN	Reserved

Bits	Name	Description	Type	Reset Value
0	HALF_RATE	Baud rate control. 0 = Running at 2.5 Gbps and 3.125 Gbps 1 = Running at 1.25 Gbps This bit corresponds to the SerDes PLL divider setting selected by the IO_SPEED field in the “SRIO MAC x Digital Loopback and Clock Selection Register” on page 369. Caution: This field should not be independently modified. Changing it can lead to unpredictable behavior.	R/W	Undefined
1	TX_CALC	A rising edge causes a recalculation of the transmitter attenuation and boost configuration. The rising edge can be generated by writing a 0 and then a 1 to the register bit.	R/W	0
2:7	Unused	N/A	R/W	0x4
8	Reserved	N/A	R	0
9:11	TX_EN[2:0]	Transmitter enable 000 = Off, no clocks running 011 = Transmitter fully enabled, all clocks running Other values are Reserved. This bit is read only unless BYPASS_INIT bit “SRIO MAC x SerDes Configuration Global” on page 364 is set to 1.	R/W	0x3

(Continued)

Bits	Name	Description	Type	Reset Value
12:15	TX_BOOST[3:0]	Transmit Boost control. Programmed boost value (ratio of drive level of transition bit to non-transition bit) is: boost = -20*log(1-(tx_boost[3:0]+0.5)/32)dB, except that setting tx_boost to 0 produces 0dB of boost. This produces results up to 5.75dB in steps of ~0.37dB.	R/W	0xC
16	RX_PLL_PWRON	Power up/reset the receive PLL 0 = Rx PLL off 1 = Rx PLL on This bit is read only unless BYPASS_INIT bit "SRIO MAC x SerDes Configuration Global" on page 364 is set to 1.	R/W	0x1
17	RX_EN	Enable receive clock and data outputs. RX_EN can only be asserted when both RX_PLL_PWRON and RX_PLL_STATE are both asserted. Set to 1 to enable Receive Data; 0 to disable Receive Data. This bit is read only unless BYPASS_INIT bit "SRIO MAC x SerDes Configuration Global" on page 364 is set to 1.	R/W	0x1
18	DPLL_RESET	A rising edge resets the frequency register of the DPLL. The rising edge can be generated by writing a 0 and then a 1 to the register bit.	R/W	0x1
19:20	Reserved	N/A	R/W	0
21:23	RX_EQ_VAL[2:0]	Receive Equalization control. Internal linear equalizer boost is approximately = (rx_eq_val+1)*0.5dB For example, 3'b100 = 2.5dB boost	R/W	0x5
24:26	RX_DPLL_MODE[2:0]	DPLL Mode selection When RX_EN is not asserted, this can change any time. This should NOT change when RX_EN is asserted,	R/W	0x0
27	TX_CKO_EN	0= Port is not in use 1 = Power-up clock for Transmit Domain. Must be set to1 even when only 1 lane is in use.	R/W	1
28:29	LOS_CTL[1:0]	Enable LOS detector and/of filtering of raw LOS output 00 = LOS detector disabled 11 = Heavy filtering for generic LOS signaling Other values are reserved.	R/W	0x0
30	RX_ALIGN_EN	Enable Word Alignment Must be disabled turned off during PRBS test ("Bit Error Rate Testing (BERT)" on page 79)	R/W	1
31	Reserved	N/A Note: Only write 1 to this reserved field.	R/W	1

12.10.4 SRIO MAC x SerDes Configuration Channel 2

This register is used to control serial port SerDes channel 2. For more details on port configuration after power down, refer to “Port Power Down” on page 71.

Register name: SMAC{0,2,4,6}_CFG_CH2 **Reset value: Undefined**	**Register offset: 130B8, 132B8, 134B8, 136B8**

Bits	0	1	2	3	4	5	6	7
00:07	HALF_RATE	TX_CALC	Unused					
08:15	Reserved	TX_EN[2:0]			TX_BOOST[3:0]			
16:23	RX_PLL_PWRON	RX_EN	DPLL_RESET	Reserved		RX_EQ_VAL[2:0]		
24:31	RX_DPLL_MODE[2:0]			TX_CKO_EN	LOS_CTL[1:0]		RX_ALIGN_EN	Reserved

Bits	Name	Description	Type	Reset Value
0	HALF_RATE	Baud rate control. 0 = Running at 2.5 Gbps and 3.125 Gbps 1 = Running at 1.25 Gbps This bit corresponds to the SerDes PLL divider setting selected by the IO_SPEED field in the “SRIO MAC x Digital Loopback and Clock Selection Register” on page 369. Caution: This field should not be independently modified. Changing it can lead to unpredictable behavior.	R/W	Undefined
1	TX_CALC	A rising edge causes a recalculation of the transmitter attenuation and boost configuration. The rising edge can be generated by writing a 0 and then a 1 to the register bit.	R/W	0
2:7	Unused	N/A	R/W	0x4
8	Reserved	N/A	R	0
9:11	TX_EN[2:0]	Transmitter enable 000 = Off, no clocks running 011 = Transmitter fully enabled, all clocks running Other values are Reserved. This bit is read only unless SMACx_CFG_GBL[BYPASS_INIT] is set to 1.	R/W	0x3

(Continued)

Bits	Name	Description	Type	Reset Value
12:15	TX_BOOST[3:0]	Transmit Boost control. Programmed boost value (ratio of drive level of transition bit to non-transition bit) is: boost = -20*log(1-(tx_boost[3:0]+0.5)/32)dB, except that setting tx_boost to 0 produces 0dB of boost. This produces results up to 5.75dB in steps of ~0.37dB.	R/W	0xC
16	RX_PLL_PWRON	Power up/reset the receive PLL 0 = Rx PLL off 1 = Rx PLL on This bit is read only unless SMACx_CFG_GBL[BYPASS_INIT] is set to 1.	R/W	0x1
17	RX_EN	Enable receive clock and data outputs. RX_EN can only be asserted when both rx_pll_pwron and rx_pll_state are both asserted. Set to 1 to enable Receive Data; 0 to disable Receive Data. This bit is read only unless SMACx_CFG_GBL[BYPASS_INIT] is set to 1.	R/W	0x1
18	DPLL_RESET	A rising edge resets the frequency register of the DPLL. The rising edge can be generated by writing a 0 and then a 1 to the register bit.	R/W	0x1
19:20	Reserved	N/A	R/W	0
21:23	RX_EQ_VAL[2:0]	Receive Equalization control. Internal linear equalizer boost is approximately = (rx_eq_val+1)*0.5dB For example, 100 = 2.5dB boost	R/W	0x5
24:26	RX_DPLL_MODE[2:0]	DPLL Mode selection When RX_EN is not asserted, this can change any time. This should NOT change when RX_EN is asserted,	R/W	0x0
27	TX_CKO_EN	Set to “1” to power-up clock for Transmit Domain. Set to “0” when port is not in used. Have to be set to “1” even when only 1 lane is in used.	R/W	1
28:29	LOS_CTL[1:0]	Enable LOS detector and/of filtering of raw LOS output 00 = LOS detector disabled 11 = Heavy filtering for generic LOS signaling Other values are reserved.	R/W	0x0
30	RX_ALIGN_EN	Enable Word Alignment Has to be turned off during PRBS test (“Bit Error Rate Testing (BERT)” on page 79)	R/W	1
31	Reserved	N/A Note: Only write 1 to this reserved field.	R/W	1

12.10.5 SRIO MAC x SerDes Configuration Channel 3

This register is used to control serial port SerDes channel 3. For more details on port configuration after power down, refer to “Port Power Down” on page 71.

Register name: SMAC{0,2,4,6}_CFG_CH3 Reset value: Undefined	Register offset: 130BC, 132BC, 134BC, 136BC

Bits	0	1	2	3	4	5	6	7
00:07	HALF_ RATE	TX_CALC	Unused					
08:15	Reserved	TX_EN[2:0]			TX_BOOST[3:0]			
16:23	RX_PLL_P WRON	RX_EN	DPLL_RES ET	Reserved		RX_EQ_VAL[2:0]		
24:31	RX_DPLL_MODE[2:0]			TX_CKO_ EN	LOS_CTL[1:0]		RX_ALIGN _EN	Reserved

Bits	Name	Description	Type	Reset Value
0	HALF_RATE	Baud rate control. 0 = Running at 2.5Gbps and 3.125Gbps 1 = Running at 1.25Gbps This bit corresponds to the SerDes PLL divider setting selected by the IO_SPEED field in the “SRIO MAC x Digital Loopback and Clock Selection Register” on page 369. Caution: This field should not be independently modified. Changing it can lead to unpredictable behavior.	R/W	Undefined
1	TX_CALC	A rising edge causes a recalculation of the transmitter attenuation and boost configuration. The rising edge can be generated by writing a 0 and then a 1 to the register bit.	R/W	0
2:7	Unused	N/A	R/W	0x4
8	Reserved	N/A	R	0
9:11	TX_EN[2:0]	Transmitter enable 000 = off, no clocks running 011 = transmitter fully enabled, all clocks running Other values are Reserved. This bit is read only unless BYPASS_INIT bit “SRIO MAC x SerDes Configuration Global” on page 364 is set to 1.	R/W	0x3

(Continued)

Bits	Name	Description	Type	Reset Value
12:15	TX_BOOST[3:0]	Transmit Boost control. Programmed boost value (ratio of drive level of transition bit to non-transition bit) is: boost = -20*log(1-(tx_boost[3:0]+0.5)/32)dB, except that setting tx_boost to 0 produces 0dB of boost. This produces results up to 5.75dB in steps of ~0.37dB.	R/W	0xC
16	RX_PLL_PWRON	Power up/reset the receive PLL 0 = Rx PLL off 1 = Rx PLL on This bit is read only unless BYPASS_INIT bit “SRIO MAC x SerDes Configuration Global” on page 364 is set to 1.	R/W	0x1
17	RX_EN	Enable receive clock and data outputs. rx_en can only be asserted when both rx_pll_pwron and rx_pll_state are both asserted. The bit is set to 1 to enable Receive Data and 0 to disable Receive Data. This bit is read only unless BYPASS_INIT bit “SRIO MAC x SerDes Configuration Global” on page 364 is set to 1.	R/W	0x1
18	DPLL_RESET	A rising edge resets the frequency register of the DPLL. The rising edge can be generated by writing a 0 and then a 1 to the register bit.	R/W	0x1
19:20	Reserved	N/A	R/W	0
21:23	RX_EQ_VAL[2:0]	Receive Equalization control. Internal linear equalizer boost is approximately = (rx_eq_val+1)*0.5dB For example, 3’b100 = 2.5dB boost	R/W	0x5
24:26	RX_DPLL_MODE[2:0]	DPLL Mode selection When RX_EN is not asserted, this can change any time. This should not change when RX_EN is asserted,	R/W	0x0
27	TX_CKO_EN	0 =Port is not in used 1 = Power-up clock for Transmit Domain. Have to be set to 1 even when only 1 lane is in used.	R/W	1
28:29	LOS_CTL[1:0]	Enable LOS detector and/of filtering of raw LOS output 00 = LOS detector disabled 11 = Heavy filtering for generic LOS signaling Other values are reserved.	R/W	0x0
30	RX_ALIGN_EN	Enable Word Alignment This bit must be disabled during PRBS test (“Bit Error Rate Testing (BERT)” on page 79)	R/W	1
31	Reserved	N/A Note: Only write 1 to this reserved field.	R/W	1

12.10.6 SRIO MAC x SerDes Configuration Global

This register configures the SerDes of all four lanes of each port.

Register name: SMAC{0,2,4,6}_CFG_GBL Reset value: undefined	Register offset: 130C0, 132C0, 134C0, 136C0

Bits	0	1	2	3	4	5	6	7
00:07	SERDES_RSTN	BYPASS_INIT	Reserved	TX_LVL[4:0]				
08:15	Reserved			ACJT_LVL[4:0]				
16:23	Reserved			LOS_LVL[4:0]				
24:31	MPLL_PWR_ON	MPLL_CK_OFF	Reserved					

Bits	Name	Description	Type	Reset Value
0	SERDES_RSTN	Active low reset signal to the SerDes This bit is read only unless BYPASS_INIT is set to 1.	R/W	1
1	BYPASS_INIT	Control bit to bypass initialization logic 0 = (default) SerDes initialization is determined by SP_IO_SPEED[1,0] 1 = Bypass initialization logic set by the SP_IO_SPEED[1,0] pins and allow direct control to SerDes. See "BYPASS_INIT Functionality" on page 354 for more information on this bit.	R/W	0
2	Reserved	N/A	R	0
3:7	TX_LVL[4:0]	Fine Resolution setting of Tx signal level. Equation: Pk-Pk output level (without attenuation) = 1230 x (48 + tx_lvl/2)/63.5 mV Vdiff-pp Note: TX_LVL should be set to >= 0b01010 (which results in an output of 1Vp-p). Refer to Table 46 on page 365 for more information on available settings.	R/W	0x0A
8:10	Reserved	N/A	R/W	0
11:15	ACJT_LVL[4:0]	ACJT Receiver Comparator Level This sets the hysterisis level for AC JTAG (Table 47 on page 366). Refer to IEEE 1149.6 for setting the correct voltage levels.	R/W	0x06
16:18	Reserved	N/A	R	0x0

(Continued)

Bits	Name	Description	Type	Reset Value
19:23	LOS_LVL[4:0]	Control the LOS detection threshold. Level at which LOS is asserted falls between the programmed threshold + 2mV and programmed threshold + 55mV. Programmed threshold = ((LOS_LVL+1)/(32*16))*1.21 Vpk	R/W	0x0
24	MPLL_PWR_ON	0 = power down 1 = normal operation This bit is read only unless BYPASS_INIT is set to 1.	R/W	1
25	MPLL_CK_OFF	0 = Turns on the MPLL clock 1 = Stops the reference clock This bit is read only unless BYPASS_INIT in is set to 1.	R/W	0
26:31	Reserved	N/A	R/W	Undefined

The reserved bits in this register are connected to signals that are configuration dependent.

Table 46: TX_LVL Values

TX_LVL	Value	TX_LVL[0:4]	Vdiff-pp (mV)
0	0x00	5'b00000	929.8
1	0x01	5'b00001	939.4
2	0x02	5'b00010	949.1
3	0x03	5'b00011	958.8
4	0x04	5'b00100	968.5
5	0x05	5'b00101	978.2
6	0x06	5'b00110	987.9
7	0x07	5'b00111	997.6
8	0x08	5'b01000	1007.2
9	0x09	5'b01001	1016.9
10	0xA	5'b01010	1026.6
11	0xB	5'b01011	1036.3
12	0xC	5'b01100	1046.0
13	0xD	5'b01101	1055.7

Table 46: TX_LVL Values

TX_LVL	Value	TX_LVL[0:4]	Vdiff-pp (mV)
14	0xE	5'b01110	1065.4
15	0xF	5'b01111	1075.0
16	0x10	5'b10000	1084.7
17	0x11	5'b10001	1094.4
18	0x12	5'b10010	1104.1
19	0x13	5'b10011	1113.8
20	0x14	5'b10100	1123.5
21	0x15	5'b10101	1133.1
22	0x16	5'b10110	1142.8
23	0x17	5'b10111	1152.5
24	0x18	5'b11000	1162.2
25	0x19	5'b11001	1171.9
26	0x1A	5'b11010	1181.6
27	0x1B	5'b11011	1191.3
28	0x1C	5'b11100	1200.9
29	0x1D	5'b11101	1210.6
30	0x1E	5'b11110	1220.3
31	0x1F	5'b11111	1230.0

Table 47: AC JTAG level programmed by ACJT_LVL[4:0]

ACJT_LVL[4:0]	Vmin level peak-to-peak differential (mV)	Vmin level peak single-ended (mV)
5'h02	310	77
5'h03	353	80
5'h04	395	100
5'h05	437	111
5'h06	478	121

Table 47: AC JTAG level programmed by ACJT_LVL[4:0]

ACJT_LVL[4:0]	Vmin level peak-to-peak differential (mV)	Vmin level peak single-ended (mV)
5’h07	521	133
5’h08	563	144
5’h09	605	155
5’h0A	648	165
5’h0B	692	176
5’h14	605	100
5’h15	670	111
5’h16	735	121
5’h17	800	133
5’h18	865	144
5’h19	932	155
5’h1A	997	165
5’h1B	1065	3176

12.10.7 SRIO MAC x SerDes Configuration GlobalB

This register configures the SerDes of all four ports.

Register name: SMAC{0,2,4,6}_CFG_GBLB Reset value: 0x0023_014F	Register offset: 130C4, 132C4, 134C4, 136C4

Bits	0	1	2	3	4	5	6	7
00:07	Reserved							
08:15	Reserved		MPLL_PRESCALE[1:0]		Unused			
16:23	Unused							
24:31	Unused		Reserved					

Bits	Name	Description	Type	Reset Value
0:9	Reserved	Reserved	R	0
10:11	MPLL_PRESCALE[1:0]	Controls the MPLL's REF_CLK prescaler. Should be set to 2'b10 in Tsi574. Mapping: 00 = Reserved 01 = Reserved 10 = Divide REF_CLK by 2 11 = Unused Transition only during RESET or when the MPLL Is disabled.	R/W	0x2
12:25	Unused	N/A	R/W	0xC05
26:31	Reserved	N/A	R	0x0F

12.10.8 SRIO MAC x Digital Loopback and Clock Selection Register

This register consists of controls for Dead Link Timer and Digital Equipment Loopback (TX -> RX) as well as clock selection on a per port basis.

Register name: SMAC{0,2,4,6}_DLOOP_CLK_SEL Reset value: Undefined	Register offset:130C8, 132C8, 134C8, 136C8

Bits	0	1	2	3	4	5	6	7
00:07	DLT_EN	DLT_THRESH						
08:15	DLT_THRESH							
16:23	LINE_LB[3:0]				Reserved	MAC_MODE	DLB_ODD_EN	DLB_EVEN_EN
24:31	SWAP_TX	SWAP_RX	SOFT_RST_X1	SOFT_RST_X4	PWDN_X1	PWDN_X4	IO_SPEED	

Bits	Name	Description	Type	Reset Value
0	DLT_EN	Dead Link Timer Enable 0 = Disabled (default) 1 = Enabled This timer is used to determine when a link is powered-up and enabled, but dead (that is, there is no link partner responding). When a link is declared dead, the transmitting port on the Tsi574 removes all packets from its transmit queue and ensure that all new packets sent to port are dropped rather than placed in the transmit queue. This feature affects both RapidIO ports sharing the MAC. This feature is not limited to lane 0 of the SerDes.	R/W	0
1:15	DLT_THRESH	Dead Link Timer Threshold Each time a silence is detected on a link, the counter is reloaded from this register and starts to count down. When the count reaches 0, the link is declared dead, which means that all packets are flushed from the transmit queue and no new packets are admitted to the queue until the link comes up. The duration of the dead link timer is computed as: 2^^13 * (DLT_THRESH +1) * P_CLK period. If DLT_THRESH = 0, even when DLT_EN = 1, the counter is still disabled. duration = 8192*(DLT_TRESH + 1) * 10nS default time = 2.68 sec If DLT_THRESH = 1, time = 164uS.	R/W	0x7FFF

(Continued)

Bits	Name	Description	Type	Reset Value
16:19	LINE_LB	Line Loopback 0 = Disabled 1 = Enabled Line Loopback LINE_LB[3..0] = lane[D, C, B, A] Caution: This function is available but its use is not recommended as CDR and elastic buffering from receive to transmit is not available.	R/W	0
20	Reserved	N/A	R/W	0
21	MAC_MODE	After the Tsi574 is reset, this field reflects the configuration of the SPx_MODESEL. Writing to this register overrides the pin settings of SPx_MODESEL. 0 = MAC supports a single 1x/4x port. 1 = MAC supports two independent 1x ports.	R/W	0
22	DLB_ODD_EN	Digital Equipment Loopback Mode Odd-numbered Port Digital equipment loopback mode connects Tx data flow to Rx data flow before the 8B10B encoder/decoder. 0 = Normal operation 1 = Loopback enabled for the odd numbered port served by this MAC. Note: The loopback path does not include the 8b/10B encoder/decoder. For more information, refer to “Port Loopback Testing” on page 78.	R/W	0
23	DLB_EVEN_EN	Digital Equipment Loopback Mode Even-numbered Port Digital equipment loopback mode connects Tx data flow to Rx data flow before the 8B10b encoder/decoder. 0 = Normal operation 1 = Loopback enabled for the even-numbered port served by this MAC. The loopback path does not include the 8b/10B encoder/decoder. For more information, refer to “Port Loopback Testing” on page 78.	R/W	0
24	SWAP_TX	Software control for transmitter lane swap functionality for this MAC. Initially, this field reflects the sampled value of the SP_TX_SWAP pin. 0 = A, B, C, D 1 = D, C, B, A	R/W	Undefined
25	SWAP_RX	Software control for receiver lane swap functionality for this MAC. Initially, this field reflects the sampled value of the SP_RX_SWAP pin. 0 = A, B, C, D 1 = D, C, B, A	R/W	Undefined
26	SOFT_RST_X1	Software reset control for the odd-numbered port. 0 = Normal mode of operation 1 = Odd-numbered port held in reset Note: This bit only affects the port logic and per-port registers; it does not reset the SerDes. In order to perform a per-port reset for an odd-numbered port, the PWDN_X1 bit must be used.	R/W	0

(Continued)

Bits	Name	Description	Type	Reset Value
27	SOFT_RST_X4	Software reset control for the even-numbered port. 0 = Normal mode of operation 1 = Even-numbered port held in reset Note: This bit only affects the port logic and per-port registers; it does not reset the SerDes. In order to perform a per-port reset for an even-numbered port, the PWDN_X4 bit must be used.	R/W	0
28	PWDN_X1	Power down control for the odd-numbered port using this MAC. Initially, this field reflects the sampled value of the SPx_PWDN pin, where “x” is 1, 3, 5,..., 7. Writing to this register overrides the configuration provided by the pin. 0 = Normal mode of operation 1 = Port powered down	R/W	Undefined
29	PWDN_X4	Power down control for even-numbered ports using this MAC. Initially, this field reflects the sampled value of the SPx_PWDN pin, where “x” is 0, 2, 4,..., 6. Writing to this register overrides the configuration provided by the odd numbered pins connected to this MAC. 0 = Normal mode of operation 1 = Even numbered port is powered down, odd numbered port is reset	R/W	Undefined
30:31	IO_SPEED	This field determines the lane speed for the serial port: 00 = 1.25Gbps 01 = 2.5Gbps 10 = 3.125Gbps 11 = Reserved Note: This field reflects the value on SP_IO_SPEED after reset. Writing to this register overrides a power up value of SP_IO_SPEED speed selection.	R/W	Undefined

12.11 Internal Switching Fabric (ISF) Registers

These registers provide control and status information concerning time-out errors in data crossing the internal switching fabric.

12.11.1 Fabric Control Register

The TEA signal is asserted when a timeout is detected on the ISF due to the requested destination being blocked. When this signal is asserted, it indicates to the source of the transaction that the requested transaction could not be completed and is removed from the request queue.

The TEA error is reported through a port-write and/or an interrupt.

Register name: FAB_CTL **Reset value: 0x0F01_0200**	**Register offset: 1AA00**

Bits	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
00:07	Reserved				RDR_LIMIT			
08:15	RDR_LIMIT_EN	Reserved	IN_ARB_MODE		Reserved		TEA_INT_EN	TEA_EN
16:23	TEA_OUT[15:8]							
24:31	TEA_OUT[7:0]							

Bits	**Name**	**Description**	**Type**	**Reset Value**
0:3	Reserved	N/A	R	0x0
4:7	RDR_LIMIT	Reorder Limit When packets arrive at an ingress port they are sent to the fabric in order. The fabric can change the order due to packet priority (if enabled through IN_ARB_MODE), and the fabric can change the order to avoid head-of-line blocking. For the latter case, a limit can be placed on the number of times a packet allows a lower or same priority packet to be placed ahead of it. This can be used to provide an upper bound on packet latency. If RDR_LIMIT_EN is set to 1, then the value in RDR_LIMIT is the maximum number of times a packet with lower or same priority can be moved ahead of a packet.	R/W	0xF
8	RDR_LIMIT_EN	Reorder Limit Enable. 0 = No limit 1 = Reordering of lower or same priority packets is limited by the value in RDR_LIMIT (recommended)	R/W	0
9	Reserved	N/A	R	0

Bits	Name	Description	Type	Reset Value
10:11	IN_ARB_MODE	Input Arbitration Mode. This field selects the arbitration scheme used by the fabric's ingress arbiters. 0 = First-come, first-served 1 = Strict Priority 1 2 = Reserved 3 = Strict Priority 2	R/W	0
12:13	Reserved	N/A	R	0
14	TEA_INT_EN	Interrupt Enable for TEA 0 = Disabled 1 = An interrupt is produced when a TEA event occurs.	R/W	0
15	TEA_EN	TEA Enable. 0 = TEA timer is disabled, similar to writing all 0s to the TEA_OUT field. 1 = TEA timer is enabled.	R/W	1
16:31	TEA_OUT[15:0]	TEA Period This value is multiplied by 2^15 to determine the number of ISF clock cycles a request waits for an acknowledge before a transaction error acknowledge (TEA) occurs. For example, assume the ISF clock is operating at maximum frequency of 156.25 MHz, and TEA_OUT is at its default value of 0x0200. The TEA timeout period is: (0x0200) * 2^15 * 6.4 ns = 107.4 ms. A value of 0x0000 disables the TEA timer.	R/W	0x0200

12.11.2 Fabric Interrupt Status Register

This register contains a status bit for every port on the fabric. The status bits indicate on which port(s) a Transaction Error Acknowledge (TEA) has occurred. Writing 1 to a bit clears it. The status bits are "ORed" together to produce the IRQ signal.

Register name: FAB_INT_STAT **Reset value: 0x0000_0000**	**Register offset: 1AA04**

Bits	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
00:07	Reserved							
08:15	Reserved							
16:23	Reserved							
24:31	PORT7_IRQ	PORT6_IRQ	PORT5_IRQ	PORT4_IRQ	PORT3_IRQ	PORT2_IRQ	PORT1_IRQ	PORT0_IRQ

Bits	**Name**	**Description**	**Type**	**Reset Value**
0:23	Reserved	Reserved	R	0
24	PORT7_IRQ	Serial port 7 IRQ Indicates that a TEA has occurred on this port. Writing a 1 to this bit clears it and causes the IRQ signal to be de-asserted.	R/W1C	0
25	PORT6_IRQ	Serial port 6 IRQ	R/W1C	0
26	PORT5_IRQ	Serial port 5 IRQ	R/W1C	0
27	PORT4_IRQ	Serial port 4 IRQ	R/W1C	0
28	PORT3_IRQ	Serial port 3 IRQ	R/W1C	0
29	PORT2_IRQ	Serial port 2 IRQ	R/W1C	0
30	PORT1_IRQ	Serial port 1 IRQ	R/W1C	0
31	PORT0_IRQ	Serial port 0 IRQ	R/W1C	0

12.11.3 RapidIO Broadcast Buffer Maximum Latency Expired Error Register

This register is a bit vector of ports which have had their maximum latency timer expire. If the AUTODEAD bit is set in the "RapidIO Multicast Maximum Latency Counter CSR" on page 384, these ports do not receive multicast packets from the broadcast buffers. Setting any of these bits cause an exception in the Global Interrupt Mask Status. For more information, refer to "RapidIO Multicast Maximum Latency Counter CSR" on page 384.

Register name: RIO_MC_LAT_ERR **Reset value: 0x0000_0000**	**Register offset: 1AA08**

Bits	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
00:07	Reserved							
08:15	Reserved							
16:23	Reserved							
24:31	P7_ERR	P6_ERR	P5_ERR	P4_ERR	P3_ERR	P2_ERR	P1_ERR	P0_ERR

Bits	**Name**	**Description**	**Type**	**Reset Value**
0:23	Reserved	N/A	R	0
24	P7_ERR	Port 7 violated the maximum multicast latency time, and will not be multicast to.	R/W1C	0
25	P6_ERR	Port 6 violated the maximum multicast latency time, and will not be multicast to.	R/W1C	0
26	P5_ERR	Port 5 violated the maximum multicast latency time, and will not be multicast to.	R/W1C	0
27	P4_ERR	Port 4 violated the maximum multicast latency time, and will not be multicast to.	R/W1C	0
28	P3_ERR	Port 3 violated the maximum multicast latency time, and will not be multicast to.	R/W1C	0
29	P2_ERR	Port 2 violated the maximum multicast latency time, and will not be multicast to.	R/W1C	0
30	P1_ERR	Port 1 violated the maximum multicast latency time, and will not be multicast to.	R/W1C	0
31	P0_ERR	Port 0 violated the maximum multicast latency time, and will not be multicast to.	R/W1C	0

12.11.4 RapidIO Broadcast Buffer Maximum Latency Expired Override

Writing to this register causes the corresponding bits in the "RapidIO Broadcast Buffer Maximum Latency Expired Error Register" on page 375 to be set. This bit causes the corresponding broadcast buffer to be purged of all data currently held in the broadcast buffer. Any packet in the process of being transferred to the broadcast buffer is purged as well. The packet being sent out from the broadcast buffer, however, finishes transmission and is not purged. If the AUTODEAD bit is set in the "RapidIO Multicast Maximum Latency Counter CSR" on page 384, the ports are prevented from receiving multicast packets.

Register name: RIO_MC_LAT_ERR_SET **Reset value: 0x0000_0000**	**Register offset: 1AA0C**

Bits	0	1	2	3	4	5	6	7
00:07	Reserved							
08:15	Reserved							
16:23	Reserved							
24:31	P7_SET	P6_SET	P5_SET	P4_SET	P3_SET	P2_SET	P1_SET	P0_SET

Bits	Name	Description	Type	Reset Value
0:23	Reserved	N/A	R	0
24	P7_SET	Port 7 multicast mask will be overridden once every time this bit is written as a 1.	R/W1S	0
25	P6_SET	Port 6 multicast mask will be overridden once every time this bit is written as a 1.	R/W1S	0
26	P5_SET	Port 5 multicast mask will be overridden once every time this bit is written as a 1.	R/W1S	0
27	P4_SET	Port 4 multicast mask will be overridden once every time this bit is written as a 1.	R/W1S	0
28	P3_SET	Port 3 multicast mask will be overridden once every time this bit is written as a 1.	R/W1S	0
29	P2_SET	Port 2 multicast mask will be overridden once every time this bit is written as a 1.	R/W1S	0
30	P1_SET	Port 1 multicast mask will be overridden once every time this bit is written as a 1.	R/W1S	0
31	P0_SET	Port 0 multicast mask will be overridden once every time this bit is written as a 1.	R/W1S	0

12.12 Utility Unit Registers

The utility block contains global registers for interrupts and clocking.

12.12.1 Global Interrupt Status Register

This register indicates which block within the Tsi574 has generated an interrupt. The interrupt requests from a given block are “ORed” together and the value of the output is reflected in this register.

Register name: GLOB_INT_STATUS **Reset value: 0x0000_0000**	**Register offset: 1AC00**

Bits	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
00:07	Reserved				RCS	MCS	I2C	TEA
08:15	Reserved						MC_LAT	Reserved
16:23	Reserved							
24:31	PORT7	PORT6	PORT5	PORT4	PORT3	PORT2	PORT1	PORT0

Bits	**Name**	**Description**	**Type**	**Reset Value**
0:3	Reserved	N/A Note: These upper 4 bits are reserved in order to be compatible with the I2C Interface	R	0
4	RCS	Combined four Reset Control Symbols interrupt status from all ports	R	0
5	MCS	Combined Multicast-Event Control Symbol interrupt status from all ports	R	0
6	I2C	I2C Interrupt Port	R	0
7	TEA	TEA occurred in fabric. To determine what port(s) experienced the TEA, see “Fabric Interrupt Status Register” on page 374).	R	0
8:13	Reserved	N/A	R	0
14	MC_LAT	At least one broadcast buffer has exceeded its maximum multicast latency timeout. To find out which ports have experienced this error, refer to “RapidIO Broadcast Buffer Maximum Latency Expired Error Register” on page 375.	R	0
15:23	Reserved	N/A	R	0
24	PORT7	Port 7 Interrupt	R	0
25	PORT6	Port 6 Interrupt	R	0
26	PORT5	Port 5 Interrupt	R	0

(Continued)

Bits	Name	Description	Type	Reset Value
27	PORT4	Port 4 Interrupt	R	0
28	PORT3	Port 3 Interrupt	R	0
29	PORT2	Port 2 Interrupt	R	0
30	PORT1	Port 1 Interrupt	R	0
31	PORT0	Port 0 Interrupt	R	0

12.12.2 Global Interrupt Enable Register

This register allows an internal interrupt request to signal an external interrupt through the INT_b pin.

Register name: GLOB_INT_ENABLE Reset value: 0x0000_0000	Register offset: 1AC04

Bits	0	1	2	3	4	5	6	7
00:07	Reserved				RCS_EN	MCS_EN	I2C_EN	TEA_EN
08:15	Reserved						MC_LAT_EN	Reserved
16:23	Reserved							
24:31	PORT7_EN	PORT6_EN	PORT5_EN	PORT4_EN	PORT3_EN	PORT2_EN	PORT1_EN	PORT0_EN

Bits	Name	Description	Type	Reset Value
0:3	Reserved	N/A	R	0
4	RCS_EN	Four Reset Control Symbols Interrupt Enable	R/W	0
5	MCS_EN	Multicast Event Control Symbol Interrupt Enable	R/W	0
6	I2C_EN	I2C Interrupt Port Enable	R/W	0
7	TEA_EN	TEA interrupt Enable	R/W	0
8:13	Reserved	N/A	R	0
14	MC_LAT_EN	Multicast Latency Interrupt Enable	R/W	0
15:23	Reserved	N/A	R	0
24	PORT7_EN	Port 7 Interrupt Enable	R/W	0
25	PORT6_EN	Port 6 Interrupt Enable	R/W	0
26	PORT5_EN	Port 5 Interrupt Enable	R/W	0
27	PORT4_EN	Port 4 Interrupt Enable	R/W	0
28	PORT3_EN	Port 3 Interrupt Enable	R/W	0
29	PORT2_EN	Port 2 Interrupt Enable	R/W	0
30	PORT1_EN	Port 1 Interrupt Enable	R/W	0
31	PORT0_EN	Port 0 Interrupt Enable	R/W	0

12.12.3 RapidIO Port-Write Timeout Control Register

This register defines port-write time-out value. Whenever a port-write is pending, this timer begins counting. When this timer expires and the port write has not yet been cleared, another port-write is sent and the timer begins counting again.

Register name: RIO_PW_TIMEOUT Reset value: 0x0000_0000	Register offset: 1AC14

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

Bits	Name	Description	Type	Reset Value
0:3	PW_TIMER	Port-write Timer This field defines the time period to repeat sending an error reporting port-write request for software assistance. The timer is stopped by software writing to the error detect registers. The timeout value is computed by = {[167772160 ns x pw_timer_value (in decimal)] + 2x(10 ns)}, where 10 ns = clock cycle period of the AMBA clk Register Bus frequency - 100 MHz 0000 = Disabled and a port-write is sent once per event. 0001 = 167.7ms 0010 = 335.5ms 0100 = 671.1ms 1000 = 1.34s 1111 = 1.3us (Debug only) Other values are reserved.	R/W	0
4:31	Reserved	N/A	R	0

12.12.4 RapidIO Port Write Outstanding Request Register

This register displays the port number that has an outstanding port-write still in the port-write arbiter. After a port-write is sent, any remaining port-write requests from any port sets a bit in the register.

Register name: RIO_PW_OREQ_STATUS **Reset value: 0x0000_0000**	**Register offset: 1AC18**

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

Bits	Name	Description	Type	Reset Value
0:23	Reserved	N/A	R	0
24:31	PORTX_OREG	Port X Port Write Outstanding Request When a bit is set, it indicates that an outstanding port-write still exists in the port-write arbiter. Bit 31: Port 0	R	0

12.12.5 MCES Pin Control Register

This register controls the operation of the MCES pin.

Register name: MCES_PIN_CTRL Reset value: 0x0000_0000	Register offset: 130D0

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

Bits	Name	Description	Type	Reset Value
0:1	Reserved	N/A	R	0
2:3	MCES_CTRL	MCES Pin Control 00 = Disabled. The MCES pin does not affect generation or receipt of Multicast Event Control Symbol 01 = MCES pin is an input (see “Generating an MCS” on page 56) 10 = MCES pin is an output (see “MCS Reception” on page 55) 11 = Reserved	R/W	0
4:31	Reserved	N/A	R	0

Changing MCES_CTRL setting during operation may result in spurious Multicast Event Control Symbols being sent.

12.13 Multicast Registers

12.13.1 RapidIO Multicast Register Version CSR

This register identifies the multicast register interface version of the IDT specific registers that is supported by this device.

Register name: RIO{0..7}_MC_REG_VER Reset value: 0x0000_0001	Register offset: 1B000,1B100, 1B200, 1B300, 1B400, 1B500, 1B600, 1B700

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

Bits	Name	Description	Type	Reset Value
0:23	Reserved	N/A	R	0
24:31	REG_VER	IDT MC Register Interface Version supported by this device	R	0x01

12.13.2 RapidIO Multicast Maximum Latency Counter CSR

This register identifies the maximum time a packet copy can wait at the head of a broadcast buffer. If this time limit is exceeded the multicast packet and packet copies in flight to the broadcast buffer are dropped, and an interrupt is raised/port-write packet sent. Optionally, the port is removed from future multicast operations until software clears the error.

Register name: RIO{0..7}_MC_LAT_LIMIT **Reset value: 0x0000_0000**	**Register offset: 1B004,1B104, 1B204, 1B304, 1B404, 1B504, 1B604, 1B704**

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

Bits	**Name**	**Description**	**Type**	**Reset Value**
0	AUTODEAD	Remove Port from Multicast if Latency Timer Expires Enable 0 = Do not remove port from multicast operations if the multicast maximum latency timer expires for this port. 1 = Remove this port from future multicast operations if the multicast maximum latency timer expires for this port.	R/W	0
1:7	Reserved	N/A	R	0
8:31	MAX_MC_LAT	The time period after which the oldest packet copy residing in the broadcast buffer is deemed to have expired. If MAX_MC_LAT == 0x0000, the multicast maximum latency feature is disabled. The timeout period is: 6.4ns * MAX_MC_LAT When the multicast maximum latency counter expires, all packet copies in the broadcast buffer are discarded. Packet copies that have been partially transferred to the broadcast buffer are also discarded. A port-write packet or interrupt may be issued to report an error. Optionally, the port may be removed from future multicast operations until software recovers the port. Ports whose maximum latency timer have expired are marked in the "RapidIO Broadcast Buffer Maximum Latency Expired Error Register" on page 375. Note: If MAX_MC_LAT is set to its maximum value, it is equivalent to the maximum time-to-live timeout value for packets.	R/W	0

12.13.3 RapidIO Port x ISF Watermarks

This register controls egress buffer allocation for reception of packets from ISF for each port.

Register name: SP{0..7}_ISF_WM Reset value: 0x0001_0203	Register offset: 1B008,1B108, 1B208, 1B308, 1B408, 1B508, 1B608, 1B708

Bits	0	1	2	3	4	5	6	7
00:7	Reserved							
8:15	Reserved					PRIO2WM		
16:23	Reserved					PRIO1WM		
24:31	Reserved					PRIO0WM		

Bits	Name	Description	Type	Reset Value
0:12	Reserved	N/A	R	0
13:15	PRIO2WM	Priority 2 packets are accepted if the number of free buffer is greater than this value. This value must be smaller than PRIO1WM. Note: It is a programming error for this value to be either greater than or equal to PRIO1WM or PRIO0WM, or greater than 7.	R/W	0x1
16:20	Reserved	N/A	R	0
21:23	PRIO1WM	Priority 1 packets are accepted if the number of free buffer is greater than this value. This value must be smaller than PRIO0WM. Note: It is a programming error for this value to be either greater than or equal to PRIO0WM, or greater than 7.	R/W	0x2
24:28	Reserved	N/A	R	0
29:31	PRIO0WM	Priority 0 packets are accepted if the number of free buffer is greater than this value. Note: It is a programming error for this value to be greater than 7.	R/W	0x3

This register must only be programmed after reset and not while traffic is flowing.

12.13.4 Port x Prefer Unicast and Multicast Packet Prio 0 Register

This register is used by the egress arbitration to control desired percentage of packets of either multicast or unicast within the same priority group. For more information, refer to “Arbitration for Egress Port” on page 92.

Register name: SP{0..7}_WRR_0 **Reset value: 0x0000_0000**	**Register offset: 1B010,1B110, 1B210, 1B310, 1B410, 1B510, 1B610, 1B710**

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

Bits	**Name**	**Description**	**Type**	**Reset Value**
0:5	Reserved	N/A	R	0
6	WRR_EN	Weighted Round Robin Enable 0 = Weighted Round Robin is disabled and no preference is given to multicast nor unicast packets. The registers WEIGHT and CHOOSE_UC have no effect. 1= Weight Round Robin is enabled and the WEIGHT field is applied to the preferred traffic chosen by CHOOSE_UC.	R/W	0
7	CHOOSE_UC	Set the preferred traffic type within the same priority group. 0 = Multicast 1 = Unicast	R/W	0
8:27	Reserved	N/A	R	0
28:31	WEIGHT	This sets the number of packets of the chosen type to be sent between non-chosen type.	R/W	0

12.13.5 Port x Prefer Unicast and Multicast Packet Prio 1 Register

This register is used by the egress arbitration to control desired percentage of packets of either multicast or unicast within the same priority group. For more information, refer to “Arbitration for Egress Port” on page 92.

Register name: SP{0..7}_WRR_1 **Reset value: 0x0000_0000**	**Register offset: 1B014,1B114, 1B214, 1B314, 1B414, 1B514, 1B614, 1B714**

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

Bits	**Name**	**Description**	**Type**	**Reset Value**
0:5	Reserved	N/A	R	0
6	WRR_EN	Weighted Round Robin Enable 0 = Weighted Round Robin is disabled and no preference is given to multicast nor unicast packets. The registers WEIGHT and CHOOSE_UC have no effect. 1 = Weight Round Robin is enabled and the WEIGHT field is applied to the preferred traffic chosen by CHOOSE_UC.	R/W	0
7	CHOOSE_UC	Choose Traffic Type Set the preferred traffic type within the same priority group. 0 = Multicast 1 = Unicast	R/W	0
8:27	Reserved	N/A	R	0
28:31	WEIGHT	Weight This sets the number of packets of the chosen type to be sent between non-chosen type.	R/W	0

12.13.6 Port x Prefer Unicast and Multicast Packet Prio 2 Register

This register is used by the egress arbitration to control desired percentage of packets of either multicast or unicast within the same priority group. For more information, refer to "Arbitration for Egress Port" on page 92.

Register name: SP{0..7}_WRR_2 Reset value: 0x0000_0000	Register offset: 1B018,1B118, 1B218, 1B318, 1B418, 1B518, 1B618, 1B718

Bits	0	1	2	3	4	5	6	7
00:07	Reserved						WRR_EN	CHOOSE_UC
08:15	Reserved							
16:23	Reserved							
24:31	Reserved				WEIGHT			

Bits	Name	Description	Type	Reset Value
0:5	Reserved	N/A	R	0
6	WRR_EN	Weighted Round Robin Enable 0 = Weighted Round Robin is disabled and no preference is given to multicast nor unicast packets. The registers WEIGHT and CHOOSE_UC have no effect. 1 = Weight Round Robin is enabled and the WEIGHT field is applied to the preferred traffic chosen by CHOOSE_UC.	R/W	0
7	CHOOSE_UC	Choose Traffic Type Set the preferred traffic type within the same priority group. 0 = Multicast 1 = Unicast	R/W	0
8:27	Reserved	N/A	R	0
28:31	WEIGHT	Weight This sets the number of packets of the chosen type to be sent between non-chosen type.	R/W	0

12.13.7 Port x Prefer Unicast and Multicast Packet Prio 3 Register

This register is used by the egress arbitration to control desired percentage of packets of either multicast or unicast within the same priority group. For more information, refer to “Arbitration for Egress Port” on page 92.

Register name: SP{0..7}_WRR_3 Reset value: 0x0000_0000	Register offset: 1B01C,1B11C, 1B21C, 1B31C, 1B41C, 1B51C, 1B61C, 1B71C

Bits	0	1	2	3	4	5	6	7
00:07	Reserved						WRR_EN	CHOOSE_UC
08:15	Reserved							
16:23	Reserved							
24:31	Reserved				WEIGHT			

Bits	Name	Description	Type	Reset Value
0:5	Reserved	N/A	R	0
6	WRR_EN	Weighted Round Robin Enable 0 = Weighted Round Robin is disabled and no preference is given to multicast nor unicast packets. The registers WEIGHT and CHOOSE_UC have no effect. 1 = Weight Round Robin is enabled and the WEIGHT field is applied to the preferred traffic chosen by CHOOSE_UC.	R/W	0
7	CHOOSE_UC	Choose Traffic Type Set the preferred traffic type within the same priority group. 0 = Multicast 1 = Unicast	R/W	0
8:27	Reserved	N/A	R	0
28:31	WEIGHT	Weight This sets the number of packets of the chosen type to be sent between non-chosen type.	R/W	0

12.14 SerDes Per Lane Register

This section details the access registers that control the functionality of the SerDes in Tsi574.

The SerDes register offsets in this section are based on lane 0. In order to define lanes 1, 2, and 3 the offset is incremented by 0x40 for each lane. For example, 0x1E000 represents lane 0 of SerDes 0, 0x1E040 represents lane 1 of SerDes 0, 0x1E080 represents lane 2 of SerDes 0, and 0x1E0C0 represents lane 3 of SerDes 0.

The reset values of the registers listed in this section are only valid when the SerDes are fully initialized. Any read operations to these registers before the SerDes is initialized returns meaningless values. The SerDes is fully initialized when MPLL_PWR_ON is equal to 1 (see "SRIO MAC x SerDes Configuration Global" on page 364).

When software has powered-down a port, 10 us must pass before the port is powered-up again.

Table 48: SerDes Register Map

Channel	Register Offset	Notes
0	1E000 - 1E03F	SerDes Per Lane Register
1	1E040 - 1E07F	
2	1E080 - 1E0BF	
3	1E0C0 - 1E0FF	

12.14.1 SerDes Lane 0 Pattern Generator Control Register

This register controls the Pattern Generator in each lane.

Register name: SMAC{0,2,4,6}_PG_CTL_0 Reset value: 0x0000_0000	Register offset: 1E020, 1E220, 1E420, 1E620

Bits	0	1	2	3	4	5	6	7
00:07	Reserved							
08:15	Reserved							
16:23	Reserved		PAT0					
24:31	PAT0				TRIGGER_ERR	MODE		

Bits	Name	Description	Type	Reset Value
0:17	Reserved	NA	R	0x0
18:27	PAT0	Pattern for MODE setting 3-5. Program the desired pattern in these 10bits when using modes 3-5. Note: This field returns to its reset value on reset	R/W	0x0
28	TRIGGER_ERR	Insert a single error into a LSB Note: This field returns to its reset value on reset	R/W	0x0
29:31	MODE	Pattern to Generate 0 = Disabled 1 = lfsr15 ($x^{15}+x^{14}+1$) 2 = lfsr7 ($x^{7}+x^{6}+1$) 3 = Fixed word (pat0) 4 = DC balanced word (pat0, ~pat0) 5 = Fixed pattern: (000, pat0, 3FF, ~pat0) 6:7 = Reserved Note: This field returns to its reset value on reset	R/W	0x0

12.14.2 SerDes Lane 1 Pattern Generator Control Register

This register controls the Pattern Generator in each lane.

Register name: SMAC{0,2,4,6}_PG_CTL_1 Reset value: 0x0000_0000	Register offset: 1E060, 1E260, 1E460, 1E660

Bits	0	1	2	3	4	5	6	7
00:07	Reserved							
08:15	Reserved							
16:23	Reserved			PAT0				
24:31	PAT0				TRIGGER_ERR	MODE		

Bits	Name	Description	Type	Reset Value
0:17	Reserved	NA	R	0x0
18:27	PAT0	Pattern for MODE setting 3-5. Program the desired pattern in these 10bits when using modes 3-5. Note: This field returns to its reset value on reset	R/W	0x0
28	TRIGGER_ERR	Insert a single error into a LSB Note: This field returns to its reset value on reset	R/W	0x0
29:31	MODE	Pattern to Generate 0 = Disabled 1 = lfsr15 ($x^{15}+x^{14}+1$) 2 = lfsr7 ($x^{7}+x^{6}+1$) 3 = Fixed word (pat0) 4 = DC balanced word (pat0, ~pat0) 5 = Fixed pattern: (000, pat0, 3FF, ~pat0) 6:7 = Reserved Note: This field returns to its reset value on reset	R/W	0x0

12.14.3 SerDes Lane 2 Pattern Generator Control Register

This register controls the Pattern Generator in each lane.

Register name: SMAC{0,2,4,6}_PG_CTL_2 Reset value: 0x0000_0000	Register offset: 1E0A0, 1E2A0, 1E4A0, 1E6A0

Bits	0	1	2	3	4	5	6	7
00:07	Reserved							
08:15	Reserved							
16:23	Reserved		PAT0					
24:31	PAT0				TRIGGER_ERR	MODE		

Bits	Name	Description	Type	Reset Value
0:17	Reserved	NA	R	0x0
18:27	PAT0	Pattern for MODE setting 3-5. Program the desired pattern in these 10bits when using modes 3-5. Note: This field returns to its reset value on reset	R/W	0x0
28	TRIGGER_ERR	Insert a single error into a LSB Note: This field returns to its reset value on reset	R/W	0x0
29:31	MODE	Pattern to Generate 0 = Disabled 1 = lfsr15 ($x^{15}+x^{14}+1$) 2 = lfsr7 ($x^{7}+x^{6}+1$) 3 = Fixed word (pat0) 4 = DC balanced word (pat0, ~pat0) 5 = Fixed pattern: (000, pat0, 3FF, ~pat0) 6:7 = Reserved Note: This field returns to its reset value on reset	R/W	0x0

12.14.4 SerDes Lane 3 Pattern Generator Control Register

This register controls the Pattern Generator in each lane.

Register name: SMAC{0,2,4,6}_PG_CTL_3 Reset value: 0x0000_0000	Register offset: 1E0E0, 1E2E0, 1E4E0, 1E6E0

Bits	0	1	2	3	4	5	6	7
00:07	Reserved							
08:15	Reserved							
16:23	Reserved			PAT0				
24:31	PAT0				TRIGGER_ERR	MODE		

Bits	Name	Description	Type	Reset Value
0:17	Reserved	NA	R	0x0
18:27	PAT0	Pattern for MODE setting 3-5. Program the desired pattern in these 10bits when using modes 3-5. Note: This field returns to its reset value on reset	R/W	0x0
28	TRIGGER_ERR	Insert a single error into a LSB Note: This field returns to its reset value on reset	R/W	0x0
29:31	MODE	Pattern to Generate 0 = Disabled 1 = lfsr15 ($x^{15}+x^{14}+1$) 2 = lfsr7 ($x^{7}+x^{6}+1$) 3 = Fixed word (pat0) 4 = DC balanced word (pat0, ~pat0) 5 = Fixed pattern: (000, pat0, 3FF, ~pat0) 6:7 = Reserved Note: This field returns to its reset value on reset	R/W	0x0

12.14.5 SerDes Lane 0 Pattern Matcher Control Register

This register contains the controls the Pattern Matcher and the error counters associated with the corresponding matcher in each lane.

Register name: SMAC{0,2,4,6}_PM_CTL_0 Reset value: 0x0000_0000	Register offset: 1E030, 1E230, 1E430, 1E630

Bits	0	1	2	3	4	5	6	7
00:07	OV14	COUNT						
08:15	COUNT							
16:23	Reserved							
24:31	Reserved				SYNC	MODE		

Bits	Name	Description	Type	Reset Value
0	OV14	1= multiply COUNT by 128. When OV14=1 and count = 2^{15}-1, signal overflows Note: Read operations on this register is pipelined. Two reads needed to get current value. The values are volatile and the value may change at any time	R/W	0x0
1:15	COUNT	Current error count If OV14 field is active, multiply count by 128. Note: Read operation on this register is pipelined. Two reads needed to get “current” value. The values are volatile (that is, value may change at any time).The second read resets the counter.	R/W	0x0
16:27	Reserved	NA	R	0x0
28	SYNC	Synchronize pattern matcher LFSR with incoming data. Must be turned on then off to enable checking. RX_ALIGN_EN must be disabled when checking PRBS patterns Note: This bit returns to its reset value on reset	R/W	0x0
29:31	MODE	Pattern to match 0 = Disabled 1 = lfsr15 2 = lfsr7 3 = d[n] = d[n-10] 4 = d[n] = !d[n-10] 5:7 = Reserved Note: This field returns to its reset value on reset	R/W	0x0

12.14.6 SerDes Lane 1 Pattern Matcher Control Register

This register contains the controls the Pattern Matcher and the error counters associated with the corresponding matcher in each lane.

Register name: SMAC{0,2,4,6}_PM_CTL_1 **Reset value: 0x0000_0000**	**Register offset: 1E070, 1E270, 1E470, 1E670**

Bits	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
00:07	OV14	COUNT						
08:15	COUNT							
16:23	Reserved							
24:31	Reserved				SYNC	MODE		

Bits	**Name**	**Description**	**Type**	**Reset Value**
0	OV14	1= multiply COUNT by 128. When OV14=1 and count = 2^{15}-1, signal overflows Note: Read operations on this register is pipelined. Two reads needed to get current value. The values are volatile and the value may change at any time	R/W	0x0
1:15	COUNT	Current error count If OV14 field is active, multiply count by 128. Note: Read operation on this register is pipelined. Two reads needed to get “current” value. The values are volatile (that is, value may change at any time)	R/W	0x0
16:27	Reserved	NA	R	0x0
28	SYNC	Synchronize pattern matcher LFSR with incoming data. Must be turned on then off to enable checking. RX_ALIGN_EN must be disabled when checking PRBS patterns Note: This bit returns to its reset value on reset	R/W	0x0
29:31	MODE	Pattern to match 0 = Disabled 1 = lfsr15 2 = lfsr7 3 = d[n] = d[n-10] 4 = d[n] = !d[n-10] 5:7 = Reserved Note: This field returns to its reset value on reset	R/W	0x0

12.14.7 SerDes Lane 2 Pattern Matcher Control Register

This register contains the controls the Pattern Matcher and the error counters associated with the corresponding matcher in each lane.

Register name: SMAC{0,2,4,6}_PM_CTL_2 Reset value: 0x0000_0000	Register offset: 1E0B0, 1E2B0, 1E4B0, 1E6B0

Bits	0	1	2	3	4	5	6	7
00:07	OV14	COUNT						
08:15	COUNT							
16:23	Reserved							
24:31	Reserved				SYNC	MODE		

Bits	Name	Description	Type	Reset Value
0	OV14	1= multiply COUNT by 128. When OV14=1 and count = 2^{15}-1, signal overflows Note: Read operations on this register is pipelined. Two reads needed to get current value. The values are volatile and the value may change at any time	R/W	0x0
1:15	COUNT	Current error count If OV14 field is active, multiply count by 128. Note: Read operation on this register is pipelined. Two reads needed to get “current” value. The values are volatile (that is, value may change at any time)	R/W	0x0
16:27	Reserved	NA	R	0x0
28	SYNC	Synchronize pattern matcher LFSR with incoming data. Must be turned on then off to enable checking. RX_ALIGN_EN must be disabled when checking PRBS patterns Note: This bit returns to its reset value on reset	R/W	0x0
29:31	MODE	Pattern to match 0 = Disabled 1 = lfsr15 2 = lfsr7 3 = d[n] = d[n-10] 4 = d[n] =!d[n-10] 5:7 = Reserved Note: This field returns to its reset value on reset	R/W	0x0

12.14.8 SerDes Lane 3 Pattern Matcher Control Register

This register contains the controls the Pattern Matcher and the error counters associated with the corresponding matcher in each lane.

Register name: SMAC{0,2,4,6}_PM_CTL_3 Reset value: 0x0000_0000	Register offset: 1E0F0, 1E2F0, 1E4F0, 1E6F0

Bits	0	1	2	3	4	5	6	7
00:07	OV14	COUNT						
08:15	COUNT							
16:23	Reserved							
24:31	Reserved				SYNC	MODE		

Bits	Name	Description	Type	Reset Value
0	OV14	1= multiply COUNT by 128. When OV14=1 and count = 2^{15}-1, signal overflows Note: Read operations on this register is pipelined. Two reads needed to get current value. The values are volatile and the value may change at any time	R/W	0x0
1:15	COUNT	Current error count If OV14 field is active, multiply count by 128. Note: Read operation on this register is pipelined. Two reads needed to get “current” value. The values are volatile (that is, value may change at any time)	R/W	0x0
16:27	Reserved	NA	R	0x0
28	SYNC	Synchronize pattern matcher LFSR with incoming data. Must be turned on then off to enable checking. RX_ALIGN_EN must be disabled when checking PRBS patterns Note: This bit returns to its reset value on reset	R/W	0x0
29:31	MODE	Pattern to match 0 = Disabled 1 = lfsr15 2 = lfsr7 3 = d[n] = d[n-10] 4 = d[n] =!d[n-10] 5:7 = Reserved Note: This field returns to its reset value on reset	R/W	0x0

12.14.9 SerDes Lane 0 Frequency and Phase Value Register

This register contains the frequency and phase of the incoming eyes on the SerDes.

Register name: SMAC{0,2,4,6}_FP_VAL_0 Reset value: 0x0000_0000	Register offset: 1E034, 1E234, 1E434, 1E634

Bits	0	1	2	3	4	5	6	7
00:07	Reserved		FVAL					
08:15	FVAL							DTHR_0
16:23	Reserved					PVAL		
24:31	PVAL							DTHR_1

Bits	Name	Description	Type	Reset Value
0:1	Reserved	NA	R	0x0
2:14	FVAL	Frequency is 1.526*FVAL ppm from the reference. Note: Read operations on this register is pipelined. Two reads needed to get current value. The values are volatile and the value may change at any time	R/W	0x0
15	DTHR_0	Bits below the useful resolution Note: Read operations on this register is pipelined. Two reads needed to get current value. The values are volatile and the value may change at any time	R/W	0
16:20	Reserved	NA	R	0x0
21:30	PVAL	Phase is 0.78125*pval ps from zero reference Note: Read operations on this register is pipelined. Two reads needed to get current value. The values are volatile and the value may change at any time	R/W	0x0
31	DTHR_1	Bits below the useful resolution Note: Read operations on this register is pipelined. Two reads needed to get current value. The values are volatile and the value may change at any time	R/W	0x0

12.14.10 SerDes Lane 1 Frequency and Phase Value Register

This register contains the frequency and phase of the incoming eyes on the SerDes.

Register name: SMAC{0,2,4,6}_FP_VAL_1 Reset value: 0x0000_0000	Register offset: 1E074, 1E274, 1E474, 1E674

Bits	0	1	2	3	4	5	6	7
00:07	Reserved		FVAL					
08:15	FVAL							DTHR_0
16:23	Reserved					PVAL		
24:31	PVAL							DTHR_1

Bits	Name	Description	Type	Reset Value
0:1	Reserved	NA	R	0x0
2:14	FVAL	Frequency is 1.526*FVAL ppm from the reference. Note: Read operations on this register is pipelined. Two reads needed to get current value. The values are volatile and the value may change at any time	R/W	0x0
15	DTHR_0	Bits below the useful resolution Note: Read operations on this register is pipelined. Two reads needed to get current value. The values are volatile and the value may change at any time	R/W	0
16:20	Reserved	NA	R	0x0
21:30	PVAL	Phase is 0.78125*pval ps from zero reference Note: Read operations on this register is pipelined. Two reads needed to get current value. The values are volatile and the value may change at any time	R/W	0x0
31	DTHR_1	Bits below the useful resolution Note: Read operations on this register is pipelined. Two reads needed to get current value. The values are volatile and the value may change at any time	R/W	0x0

12.14.11 SerDes Lane 2 Frequency and Phase Value Register

This register contains the frequency and phase of the incoming eyes on the SerDes.

Register name: SMAC{0,2,4,6}_FP_VAL_2 Reset value: 0x0000_0000	Register offset: 1E0B4, 1E2B4, 1E4B4, 1E6B4

Bits	0	1	2	3	4	5	6	7
00:07	Reserved		FVAL					
08:15	FVAL							DTHR_0
16:23	Reserved					PVAL		
24:31	PVAL							DTHR_1

Bits	Name	Description	Type	Reset Value
0:1	Reserved	NA	R	0x0
2:14	FVAL	Frequency is 1.526*FVAL ppm from the reference. Note: Read operations on this register is pipelined. Two reads needed to get current value. The values are volatile and the value may change at any time	R/W	0x0
15	DTHR_0	Bits below the useful resolution Note: Read operations on this register is pipelined. Two reads needed to get current value. The values are volatile and the value may change at any time	R/W	0
16:20	Reserved	NA	R	0x0
21:30	PVAL	Phase is 0.78125*pval ps from zero reference Note: Read operations on this register is pipelined. Two reads needed to get current value. The values are volatile and the value may change at any time	R/W	0x0
31	DTHR_1	Bits below the useful resolution Note: Read operations on this register is pipelined. Two reads needed to get current value. The values are volatile and the value may change at any time	R/W	0x0

12.14.12 SerDes Lane 3 Frequency and Phase Value Register

This register contains the frequency and phase of the incoming eyes on the SerDes.

Register name: SMAC{0,2,4,6}_FP_VAL_3 Reset value: 0x0000_0000	Register offset: 1E0F4, 1E2F4, 1E4F4, 1E6F4

Bits	0	1	2	3	4	5	6	7
00:07	Reserved		FVAL					
08:15	FVAL							DTHR_0
16:23	Reserved					PVAL		
24:31	PVAL							DTHR_1

Bits	Name	Description	Type	Reset Value
0:1	Reserved	NA	R	0x0
2:14	FVAL	Frequency is 1.526*FVAL ppm from the reference. Note: Read operations on this register is pipelined. Two reads needed to get current value. The values are volatile and the value may change at any time	R/W	0x0
15	DTHR_0	Bits below the useful resolution Note: Read operations on this register is pipelined. Two reads needed to get current value. The values are volatile and the value may change at any time	R/W	0
16:20	Reserved	NA	R	0x0
21:30	PVAL	Phase is 0.78125*pval ps from zero reference Note: Read operations on this register is pipelined. Two reads needed to get current value. The values are volatile and the value may change at any time	R/W	0x0
31	DTHR_1	Bits below the useful resolution Note: Read operations on this register is pipelined. Two reads needed to get current value. The values are volatile and the value may change at any time	R/W	0x0

13. I²C Registers

Topics discussed include the following:

- "Register Map"
- "Register Descriptions"

13.1 Register Map

The following table lists the register map for the I²C registers.

All registers can be accessed through the internal register bus. A portion of the register space is visible to an external I²C master through the slave interface. The registers in this portion have a *peripheral address* in addition to their internal register bus address, and are called the *externally visible* I²C registers, meaning they can be accessed by the external I²C master using I²C reads and writes (see Table 49). The peripheral address equates to a 4-byte range within a consecutive 256-byte address space. For peripheral addresses, the lowest address maps to the least significant byte of the internal register (LSB), while the highest address maps to the most significant byte of the internal register (MSB).

The internal address for the I²C registers is 0x1D000 to 0x1DFFC.

Table 49: I²C Register Map

Internal Address	Peripheral Address	Register Name	See
0x1D000–0x1D0FC	n/a	Reserved	
0x1D100	n/a	I2C_DEVID	"I²C Device ID Register"
0x1D104	n/a	I2C_RESET	"I²C Reset Register"
0x1D108	n/a	I2C_MST_CFG	"I²C Master Configuration Register"
0x1D10C	n/a	I2C_MST_CNTRL	"I²C Master Control Register"
0x1D110	n/a	I2C_MST_RDATA	"I²C Master Receive Data Register"
0x1D114	n/a	I2C_MST_TDATA	"I²C Master Transmit Data Register"
0x1D118	n/a	I2C_ACC_STAT	"I²C Access Status Register"
0x1D11C	n/a	I2C_INT_STAT	"I²C Interrupt Status Register"
0x1D120	n/a	I2C_INT_ENABLE	"I²C Interrupt Enable Register"

Table 49: I²C Register Map *(Continued)*

Internal Address	Peripheral Address	Register Name	See
0x1D124	n/a	I2C_INT_SET	"I²C Interrupt Set Register"
0x1D12C	n/a	I2C_SLV_CFG	"I²C Slave Configuration Register"
0x11D130–0x1D13C	n/a	Reserved	
0x1D140	n/a	I2C_BOOT_CNTRL	"I²C Boot Control Register"
0x1D144–0x1D1FC	n/a	Reserved	
0x1D200	0x00–0x03	EXI2C_REG_WADDR	"Externally Visible I²C Internal Write Address Register"
0x1D204	0x04–0x07	EXI2C_REG_WDATA	"Externally Visible I²C Internal Write Data Register"
0x1D208–0x1D20C	0x08–0x0F	Reserved	
0x1D210	0x10–0x13	EXI2C_REG_RADDR	"Externally Visible I²C Internal Read Address Register"
0x1D214	0x14–0x17	EXI2C_REG_RDATA	"Externally Visible I²C Internal Read Data Register"
0x1D218–0x1D21C	0x18–0x1F	Reserved	
0x1D220	0x20–0x23	EXI2C_ACC_STAT	"Externally Visible I²C Slave Access Status Register"
0x1D224	0x24–0x27	EXI2C_ACC_CNTRL	"Externally Visible I²C Internal Access Control Register"
0x1D228–0x1D27C	0x28–0x7F	Reserved	
0x1D280	0x80–0x83	EXI2C_STAT	"Externally Visible I²C Status Register"
0x1D284	0x84–0x87	EXI2C_STAT_ENABLE	"Externally Visible I²C Enable Register"
0x1D288–0x1D28C	0x88–0x8F	Reserved	
0x1D290	0x90–0x93	EXI2C_MBOX_OUT	"Externally Visible I²C Outgoing Mailbox Register"
0x1D294	0x94–0x97	EXI2C_MBOX_IN	"Externally Visible I²C Incoming Mailbox Register"
0x1D298–0x1D2FC	0x98–0xFF	Reserved	
0x1D300	n/a	I2C_EVENT	"I²C Event and Event Snapshot Registers"
0x1D304	n/a	I2C_SNAP_EVENT	"I²C Event and Event Snapshot Registers"

Table 49: I²C Register Map *(Continued)*

Internal Address	Peripheral Address	Register Name	See
0x1D308	n/a	I2C_NEW_EVENT	"I²C New Event Register"
0x1D30C	n/a	I2C_EVENT_ENB	"I²C Enable Event Register"
0x1D310–0x1D31C	n/a	Reserved	
0x1D320	n/a	I2C_DIVIDER	"I²C Time Period Divider Register"
0x1D324–0x1D33C	n/a	Reserved	
0x1D340	n/a	I2C_START_SETUP_HOLD	"I²C Start Condition Setup/Hold Timing Register"
0x1D344	n/a	I2C_STOP_IDLE	"I²C Stop/Idle Timing Register"
0x1D348	n/a	I2C_SDA_SETUP_HOLD	"I2C_SD Setup and Hold Timing Register"
0x1D34C	n/a	I2C_SCL_PERIOD	"I2C_SCLK High and Low Timing Register"
0x1D350	n/a	I2C_SCL_MIN_PERIOD	"I2C_SCLK Minimum High and Low Timing Register"
0x1D354	n/a	I2C_SCL_ARB_TIMEOUT	"I2C_SCLK Low and Arbitration Timeout Register"
0x1D358	n/a	I2C_BYTE_TRAN_TIMEOUT	"I²C Byte/Transaction Timeout Register"
0x1D35C	n/a	I2C_BOOT_DIAG_TIMER	"I²C Boot and Diagnostic Timer"
0x1D360–0x1D3B4	n/a	Reserved	
0x1D3B8	n/a	I2C_BOOT_DIAG_PROGRESS	"I²C Boot Load Diagnostic Progress Register"
0x1D3BC	n/a	I2C_BOOT_DIAG_CFG	"I²C Boot Load Diagnostic Configuration Register"
0x1D3C0–0x1D3FC	n/a	Reserved	

13.2 Register Descriptions

This section describes the I^2C registers. These registers are reset by a chip reset.

13.2.1 I^2C Device ID Register

This register identifies the version of the IDT I^2C block in this device.

Register name: I2C_DEVID **Reset value: 0x0000_0001**	**Register offset: 0x1D100**

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

Bits	**Name**	**Description**	**Type**	**Reset Value**
00:27	Reserved	Reserved	R	0x000_0000
28:31	REV	Indicates the revision ID for the I^2C block.	R	0x1

13.2.2 I²C Reset Register

This register completes a reset of the I²C block. This reset returns the logic to its idle, non-transacting state while retaining all configuration registers, such that the block does not have to be reprogrammed. This is provided for exceptional conditions. A reset while the block is involved in a transaction as a master or slave may leave the bus in an unexpected state relative to any external I²C masters or slaves, and thus should be used with caution and only as a last solution if the block seems unresponsive.

I²C registers that are affected by a this reset are indicated in the description of that register. All other registers should be assumed to be unaffected by this reset.

Register name: I2C_RESET **Reset value: 0x0000_0000**	**Register offset: 0x1D104**

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

Bits	**Name**	**Description**	**Type**	**Reset Value**
00	SRESET	Reset under Software Control Setting this bit resets the I²C block. The R/W fields of configuration and control registers are not affected (nor is this register affected). While in reset neither the Master nor slave interface will be operational: the I2C_SCLK and I2C_SD signals will be undriven so as to not obstruct the bus. This bit must be written to 0 to bring the block out of reset. Any active bus transactions while reset occurs are aborted. All status and events are returned to the reset state. The boot load sequence will not be invoked upon exit from reset, although the bus idle detect sequence will be invoked. Power-up latch values will not be re-latched, and the fields will remain at their pre-soft-reset value.	R/W	0
01:31	Reserved	Reserved	R	0

13.2.3 I²C Master Configuration Register

This register contains options that apply to master operations initiated through the "I²C Master Control Register". The configuration specifies the properties of the external slave device to which a read or write transaction will be directed.

Register name: I2C_MST_CFG Reset value: Undefined	Register offset: 0x1D108

Bits	0	1	2	3	4	5	6	7
00:07	Reserved							
08:15	DORDER	Reserved					PA_SIZE	
16:23	Reserved							
24:31	Reserved	DEV_ADDR						

Bits	Name	Description	Type	Reset Value
00:07	Reserved	Reserved	R	0x00
08	DORDER	Data Order 0 = Data from/to data registers is ordered (processed) from MSB to LSB within an I²C transaction. 1 = Data from/to data registers is ordered (processed) from LSB to MSB within an I²C transaction. Data registers are "I²C Master Transmit Data Register" and "I²C Master Receive Data Register".	R/W	0
09:13	Reserved	Reserved	R	0x00
14:15	PA_SIZE	Peripheral Address Size 00 = No peripheral address used 01 = 8-bit peripheral device addressing using LSB of PADDR 10 = 16-bit peripheral device addressing using MSB and LSB of PADDR (in that order) 11 = Reserved (handled as 00) This field selects the number of bytes in the peripheral address for master transactions. The peripheral address itself is specified in the "I²C Master Control Register".	R/W	Undefined
16:24	Reserved	Reserved	R	0x000
25:31	DEV_ADDR	Device Address Specifies the 7-bit device address to select the I²C device for a read or write transaction initiated through the "I²C Master Control Register".	R/W	Undefined

Do not change this register while a master operation is active. The effect on the transaction cannot be determined.

13.2.4 I2C Master Control Register

This register sets the peripheral address and to start an I2C transaction. The transaction is directed to the device defined in the "I2C Master Configuration Register".

Note: Software must not set the peripheral address and the SIZE parameters such that unintended page wrap-arounds occur in the target device. The Tsi574 does not force repeated start conditions within a single software initiated access.

Register name: I2C_MST_CNTRL **Reset value: 0x0000_0000**	**Register offset: 0x1D10C**

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

Bits	**Name**	**Description**	**Type**	**Reset Value**
00	START	Start Operation 0 = I2C operation is not in progress (self clears) 1 = Start of an I2C operation. Clears itself back to zero when the initiated operation has completed. This bit cannot be cleared by software once set, except through a reset under software control. Note: This bit is cleared by a reset initiated by the "I2C Reset Register".	R/W1S	0
01	WRITE	I2C Read or Write 0 = Read from I2C memory 1 = Write to I2C memory For a read, data is returned to the "I2C Master Receive Data Register". For a write, data is taken from the "I2C Master Transmit Data Register".	R/W	0
02:04	Reserved	Reserved	R	0

(Continued)

Bits	Name	Description	Type	Reset Value
05:07	SIZE	Number of bytes in an I^2C operation (read or write) 000 = 0 bytes 001 = 1 bytes 010 = 2 bytes 011 = 3 bytes 100 = 4 bytes 101 = Reserved (equivalent of 0 bytes) 110 = Reserved (equivalent of 0 bytes) 111 = Reserved (equivalent of 0 bytes) A value of 000 should not be normally used for a Read operation, as any device put into read mode assumes at least one byte will be read. An exception would be the SMBus Quick Command protocol to a device that is known to not hold the bus following the slave address phase.	R/W	000
08:15	Reserved	Reserved	R	0
16:31	PADDR	Peripheral address for master operation If PA_SIZE in the "I^2C Master Configuration Register" is 10, then all 16 bits are used, with the most significant 8 bits placed on I^2C bus first, followed by the least significant 8 bits. If PA_SIZE is 01, then only least significant 8 bits are used. If PA_SIZE is 00, this field is not used for the transaction.	R/W	0x0000

A master operation is initiated by writing this register and setting the START bit to 1. The same write should set the WRITE, SIZE, and PADDR fields as required by the transaction. Other information for the transaction must have been pre-loaded into the "I^2C Master Configuration Register".

Do not change this register while a master operation is active. The effect on the transaction cannot be determined.

The following is the sequence triggered by setting the START bit:

Table 50: Master Operation Sequence

Phase	Description	Outcome
1. Begin transaction	Start arbitration timer.	-
2. Address slave	Detect bus idle. Send START. Send I2C_MST_CFG.[DEV_ADDR]. Send Read or Write (normally Write unless WRITE=0 and PA_SIZE=0). Wait for ACK/NACK. Disable arbitration timer.	Any loss of arbitration repeats this phase. Phase completes with ACK. NACK terminates operation with a MNACK event (issues STOP). Expiration of arbitration timer aborts operation with MARBTO.
3. Peripheral Address	If PA_SIZE = 10, send PADDR[15:8] and wait for ACK. If PA_SIZE = 01 or 10, send PADDR[7:0] and wait for ACK.	If PA_SIZE is 00 or 11, this phase is skipped. Any loss of arbitration will abort with MCOL. Any NACK will abort with MNACK.
4. Send Data (WRITE = 1)	If SIZE > 0, send SIZE bytes from I2C_MST_TDATA based on DORDER. Wait for ACK from each.	This phase is skipped if WRITE=1. Any loss of arbitration will abort with MCOL. Any NACK will abort with MNACK.
5. Read Data Setup (WRITE = 0)	Send RESTART. Repeat the “Address Slave” process with last bit a Read. Wait for ACK/NACK.	This phase is skipped if WRITE=1 or SIZE=0. NACK aborts with MNACK. Loss of arbitration aborts with MCOL (because bus was never released).
6. Read Data (WRITE = 0)	Read SIZE bytes and place in I2C_MST_RDATA based on DORDER. Respond with ACK to each, except for final byte respond with a NACK.	This phase is skipped if WRITE=1 or SIZE=0. Any loss of arbitration aborts with MCOL.
7. Complete	Issue STOP. Clear I2C_MST_CNTRL[START]. Set MSD event.	Except for arbitration loss, master always tries to force a STOP condition.

13.2.5 I²C Master Receive Data Register

This register contains the data read from an external slave device following a read operation initiated using the "I²C Master Control Register".

As bytes are read from the I²C bus, they are placed in this register depending on DORDER in the "I²C Master Configuration Register". If DORDER is 0, bytes are loaded from MSB to LSB, in order: RBYTE3, RBYTE2, RBYTE1, RBYTE0. If DORDER is 1, bytes are loaded from LSB to MSB, in order: RBYTE0, RBYTE1, RBYTE2, RBYTE3. If the transaction size is less than four (4) bytes (that is, SIZE in the "I²C Master Control Register" < 4) then any remaining bytes in the register are left unchanged (that is, they retain the values they had from the prior read operation).

Register name: I2C_MST_RDATA **Reset value: 0x0000_0000**	**Register offset: 0x1D110**

Bits	0	1	2	3	4	5	6	7
00:07	RBYTE3							
08:15	RBYTE2							
16:23	RBYTE1							
24:31	RBYTE0							

Bits	Name	Description	Type	Reset value
00:07	RBYTE3	Received I²C data — Byte 3 (most significant)	R	0x00
08:15	RBYTE2	Received I²C data — Byte 2	R	0x00
16:23	RBYTE1	Received I²C data — Byte 1	R	0x00
24:31	RBYTE0	Received I²C data — Byte 0 (least significant)	R	0x00

13.2.6 I²C Master Transmit Data Register

This register contains the data to be written (transmitted) to an external slave when a write operation is initiated using the "I²C Master Control Register". This register should be written with data to be sent prior to setting the START bit in that register.

As bytes are written to the I²C bus, they are taken from this register depending on DORDER in the "I²C Master Configuration Register". If DORDER is 0, bytes are taken from MSB to LSB, in order: TBYTE3, TBYTE2, TBYTE1, TBYTE0. If DORDER is 1, bytes are taken from LSB to MSB, in order: TBYTE0, TBYTE1, TBYTE2, TBYTE3. If the transaction size is less than 4 bytes (that is, SIZE in the "I²C Master Control Register" <4) then any remaining bytes in the register are unused. The contents of this register are not affected by the transaction.

Register name: I2C_MST_TDATA **Reset value: 0x0000_0000**	**Register offset: 0x1D114**

Bits	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
00:07	TBYTE3							
08:15	TBYTE2							
16:23	TBYTE1							
24:31	TBYTE0							

Bits	**Name**	**Description**	**Type**	**Reset value**
00:07	TBYTE3	Transmitted I²C data — Byte 3 (most significant)	R/W	0x00
08:15	TBYTE2	Transmitted I²C data — Byte 2	R/W	0x00
16:23	TBYTE1	Transmitted I²C data — Byte 1	R/W	0x00
24:31	TBYTE0	Transmitted I²C data — Byte 0 (least significant)	R/W	0x00

Do not change this register while a master operation is active. The effect on the transaction cannot be determined.

13.2.7 I2C Access Status Register

This register indicates the status of the I2C block. Fields in this register change dynamically as operations are initiated or progress.

Register name: I2C_ACC_STAT Reset value: 0x0000_0000	Register offset: 0x1D118

Bits	0	1	2	3	4	5	6	7
00:07	SLV_ ACTIVE	BUS_ ACTIVE	Reserved		SLV_WAIT	SLV_PHASE		SLV_AN
08:15	SLV_PA							
16:23	MST_ ACTIVE	Reserved			MST_PHASE			MST_AN
24:31	Reserved				MST_NBYTES			

Bits	Name	Description	Type	Reset Value
00	SLV_ACTIVE	Slave Active 0 = Slave is not addressed 1 = Slave is addressed by external master and a read or write is active on the bus This bit is set following the slave address phase if the address matched the SLV_ADDR or Alert Response Address and the slave interface was enabled. Note: This bit is zeroed on a reset controlled by the "I2C Reset Register".	R	0
01	BUS_ACTIVE	Bus Active 0 = I2C bus is not active 1 = I2C bus is active: a START bit is seen (and no subsequent STOP) Note: This bit is zeroed on a reset controlled by the "I2C Reset Register", and is not set to 1 until a START condition is seen after reset is de-asserted in the "I2C Reset Register".	R	0
02:03	Reserved	Reserved	R	00

(Continued)

Bits	Name	Description	Type	Reset Value
04	SLV_WAIT	Slave Wait 0 = Slave is not waiting for a STOP or RESTART 1 = Slave is waiting for a STOP or RESTART This bit is clear if the bus is not active or the slave address is being received or the slave is active. This bit is set if the bus is active but the slave is not active and the slave address is not being received. Note: This bit is zeroed on a reset controlled by the "I2C Reset Register".	R	0
05:06	SLV_PHASE	Slave Phase 00 = Slave address being received (even if slave interface is disabled using SLV_EN). 01 = Peripheral address being received 10 = Data incoming (write from external master) 11 = Data outgoing (read by external master) At the end of a slave operation, this field will hold its value until the next START/RESTART. If a slave operation aborts, this field will qualify where in the transaction the error occurred.	R	0x0
07	SLV_AN	Slave Ack/Nack 0 = Slave transaction is not in the ACK/NACK bit of a byte 1 = Slave transaction is in the ACK/NACK bit of a byte This qualifies the SLV_PHASE field.	R	0
08:15	SLV_PA	Slave Peripheral Address This field indicates the current peripheral address that is used when the Tsi574 is accessed by an external master.	R	0x00
16	MST_ACTIVE	Master Active 0 = No master operation in progress 1 = Master operation is in progress This status is the same as the START bit in the "I2C Master Control Register". Note: This bit is zeroed on a reset controlled by the "I2C Reset Register".	R	0
17:19	Reserved	Reserved	R	0

(Continued)

Bits	Name	Description	Type	Reset Value
20:22	MST_ PHASE	Master Phase 000 = START condition being sent 001 = External slave address being transmitted 010 = Peripheral address being transmitted 011 = RESTART condition being sent 100 = Data incoming (read operation) 101 = Data outgoing (write operation) 110 = STOP condition being sent 111 = Reserved At the end of a master operation, this field will hold its value until the next master operation is started. If a master operation aborts, this field will qualify where in the transaction the error occurred.	R	000
23	MST_AN	Master Ack/Nack 0 = Master transaction is not in the ACK/NACK bit of a byte 1 = Master transaction is in the ACK/NACK bit of a byte This qualifies the MST_PHASE field.	R	0
24:27	Reserved	Reserved	R	0x0
28:31	MST_ NBYTES	Master Number of Bytes This is the running count of the number of data bytes transferred in the current master operation (read or write). At the end of an operation, if successfully completed, the field will equal the SIZE field from the "I2C Master Control Register". If an operation aborts prematurely, this field will indicate the number of bytes transferred before the error occurred.	R	0x0

13.2.8 I²C Interrupt Status Register

This register indicates the status of the I²C interrupts. When an interrupt status bit is set, an interrupt is generated to the Interrupt Controller if the corresponding bit is enabled in the "I²C Interrupt Enable Register". If the corresponding enable is not set, the interrupt status bit will still assert but will not result in assertion of an interrupt to the Interrupt Controller. This register can only be accessed through the register bus.

Note: This register is affected by a reset controlled by the "I²C Reset Register". All interrupts will be cleared and no interrupt will assert until an event occurs after SRESET bit in the "I²C Reset Register" is de-asserted.

Register name: I2C_INT_STAT **Reset value: 0x0000_0000**	**Register offset: 0x1D11C**

Bits	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
00:07	Reserved						OMB_ EMPTY	IMB_FULL
08:15	Reserved						BL_FAIL	BL_OK
16:23	Reserved				SA_ FAIL	SA_ WRITE	SA_READ	SA_OK
24:31	MA_DIAG	Reserved		MA_COL	MA_TMO	MA_NACK	MA_ATMO	MA_OK

Bits	**Name**	**Description**	**Type**	**Reset Value**
0:5	Reserved	Reserved	R	0x00
6	OMB_EMPTY	Outgoing Mailbox Empty 0 = Interrupt status not asserted 1 = Outgoing mailbox is empty Set when an external I²C master reads data from the mailbox (see "Externally Visible I²C Outgoing Mailbox Register"), if data had been previously been written to the mailbox by software.	R/W1C	0
7	IMB_FULL	Incoming Mailbox Full 0 = Interrupt status not asserted 1 = Incoming mailbox is full Set when an external I²C master writes data into the "Externally Visible I²C Incoming Mailbox Register".	R/W1C	0
8:13	Reserved	Reserved	R	0x00
14	BL_FAIL	Boot Load Failed 0 = Interrupt status not asserted 1 = Boot load sequence failed to complete	R/W1C	0

(Continued)

Bits	Name	Description	Type	Reset Value
15	BL_OK	Boot Load OK 0 = Interrupt status not asserted 1 = Boot load sequence completed successfully This will also be set if the boot loading was disabled at reset.	R/W1C	0
16:19	Reserved	Reserved	R	0x0
20	SA_FAIL	Slave Access Failed 0 = Interrupt status not asserted 1 = Error detected during slave access transaction A slave transaction addressed to the Tsi574 aborted.	R/W1C	0
21	SA_WRITE	Slave Write 0 = Interrupt status not asserted 1 = Internal register write performed The "Externally Visible I2C Internal Write Data Register" was written by an external master, invoking a write to an internal register. This will not assert if slave writes are disabled (WR_EN in the "I2C Slave Configuration Register" = 0).	R/W1C	0
22	SA_READ	Slave Read 0 = Interrupt status not asserted 1 = Internal register read performed The "Externally Visible I2C Internal Read Data Register" was read by an external master, invoking a read to an internal register. This will not assert if slave reads are disabled (RD_EN = 0 in the "I2C Slave Configuration Register").	R/W1C	0
23	SA_OK	Slave Access OK 0 = Interrupt status not asserted 1 = Access completed successfully The Tsi574 was addressed as a slave device and the transaction completed without error.	R/W1C	0
24	MA_DIAG	Master Diagnostic Event 0 = Interrupt status not asserted 1 = Diagnostic event	R/W1C	0
25:26	Reserved	Reserved	R	00

(Continued)

Bits	Name	Description	Type	Reset Value
27	MA_COL	Master Collision 0 = Interrupt status not asserted 1 = Collision (arbitration loss) occurred following the device address phase A transaction initiated using the "I2C Master Control Register" aborted due to loss of arbitration after the slave device address phase. This indicates multiple masters tried to access the same slave. This can also be set at the end of boot load due to BL_FAIL.	R/W1C	0
28	MA_TMO	Master Timeout 0 = Interrupt status not asserted 1 = Transaction aborted due to timeout expiration A transaction initiated using the "I2C Master Control Register" aborted due to expiration of the clock low, byte, or transaction time-outs. This can also be set at the end of boot load due to BL_FAIL.	R/W1C	0
29	MA_NACK	Master NACK 0 = Interrupt status not asserted 1 = NACK received during transaction A transaction initiated through the "I2C Master Control Register" aborted due to receipt of a NACK in response to slave address, peripheral address, or a written byte. This can also be set at the end of boot load due to BL_FAIL.	R/W1C	0
30	MA_ATMO	Master Arbitration Timeout 0 = Interrupt status not asserted 1 = Bus arbitration timeout expired A transaction initiated through the "I2C Master Control Register" aborted due to expiration of the arbitration timeout (see ARB_TO in "I2C_SCLK Low and Arbitration Timeout Register"). This indicates the bus is in use by other masters.	R/W1C	0
31	MA_OK	Master Transaction OK 0 = Interrupt status not asserted 1 = Access completed and successful A transaction initiated through the "I2C Master Control Register" completed without error.	R/W1C	0

The write-1-to-clear (W1C) operation requires that this register first be read to create an event snapshot.

13.2.9 I2C Interrupt Enable Register

This register controls which of the interrupt status bits in the "I2C Interrupt Status Register" will result in an interrupt asserted to the Interrupt Controller. It can only be accessed from the register bus.

Register name: I2C_INT_ENABLE Reset value: 0x0000_0000	Register offset: 0x1D120

Bits	0	1	2	3	4	5	6	7
00:07	Reserved						OMB_ EMPTY	IMB_FULL
08:15	Reserved						BL_FAIL	BL_OK
16:23	Reserved				SA_ FAIL	SA_ WRITE	SA_READ	SA_OK
24:31	MA_DIAG	Reserved		MA_COL	MA_TMO	MA_NACK	MA_ATMO	MA_OK

Bits	Name	Description	Type	Reset Value
0:5	Reserved	Reserved	R	0x00
6	OMB_ EMPTY	Enable OMB_EMPTY Interrupt 0 = Interrupt is disabled 1 = Interrupt is enabled	R/W	0
7	IMB_FULL	Enable IMB_FULL Interrupt 0 = Interrupt is disabled 1 = Interrupt is enabled	R/W	0
8:13	Reserved	Reserved	R	0x00
14	BL_FAIL	Enable BL_FAIL Interrupt 0 = Interrupt is disabled 1 = Interrupt is enabled	R/W	0
15	BL_OK	Enable BL_OK Interrupt 0 = Interrupt is disabled 1 = Interrupt is enabled	R/W	0
16:19	Reserved	Reserved	R	0x0
20	SA_FAIL	Enable SA_FAIL Interrupt 0 = Interrupt is disabled 1 = Interrupt is enabled	R/W	0

(Continued)

Bits	Name	Description	Type	Reset Value
21	SA_WRITE	Enable SA_WRITE Interrupt 0 = Interrupt is disabled 1 = Interrupt is enabled	R/W	0
22	SA_READ	Enable SA_READ Interrupt 0 = Interrupt is disabled 1 = Interrupt is enabled	R/W	0
23	SA_OK	Enable SA_OK Interrupt 0 = Interrupt is disabled 1 = Interrupt is enabled	R/W	0
24	MA_DIAG	Enable MA_DIAG Interrupt 0 = Interrupt is disabled 1 = Interrupt is enabled	R/W	0
25:26	Reserved	Reserved	R	00
27	MA_COL	Enable MA_COL Interrupt 0 = Interrupt is disabled 1 = Interrupt is enabled	R/W	0
28	MA_TMO	Enable MA_TMO Interrupt 0 = Interrupt is disabled 1 = Interrupt is enabled	R/W	0
29	MA_NACK	Enable MA_NACK Interrupt 0 = Interrupt is disabled 1 = Interrupt is enabled	R/W	0
30	MA_ATMO	Enable MA_ATMO Interrupt 0 = Interrupt is disabled 1 = Interrupt is enabled	R/W	0
31	MA_OK	Enable MA_OK Interrupt 0 = Interrupt is disabled 1 = Interrupt is enabled	R/W	0

13.2.10 I²C Interrupt Set Register

This register sets the status of the I²C blocks interrupts. It can only be accessed from the register bus.

Note: Setting an interrupt sets all related underlying events in the "I²C New Event Register". This is significant in that if all underlying events are disabled for a specific interrupt bit, this register will not appear to work for that bit.

Register name: I2C_INT_SET **Reset value: 0x0000_0000**	**Register offset: 0x1D124**

Bits	0	1	2	3	4	5	6	7
00:07	Reserved						OMB_ EMPTY	IMB_ FULL
08:15	Reserved						BL_FAIL	BL_OK
16:23	Reserved				SA_FAIL	SA_ WRITE	SA_READ	SA_OK
24:31	MA_DIAG	Reserved		MA_COL	MA_TMO	MA_NACK	MA_ATMO	MA_OK

Bits	Name	Description	Type	Reset Value
0:5	Reserved	Reserved	R	0x00
6	OMB_EMPTY	Set OMB_EMPTY Interrupt 0 = No effect 1 = Interrupt is set	R/W1S	0
7	IMB_FULL	Set IMB_FULL Interrupt 0 = No effect 1 = Interrupt is set	R/W1S	0
8:13	Reserved	Reserved	R	0x00
14	BL_FAIL	Set BL_FAIL Interrupt 0 = No effect 1 = Interrupt is set	R/W1S	0
15	BL_OK	Set BL_OK Interrupt 0 = No effect 1 = Interrupt is set	R/W1S	0
16:19	Reserved	Reserved	R	0x0
20	SA_FAIL	Set SA_FAIL Interrupt 0 = No effect 1 = Interrupt is set	R/W1S	0

(Continued)

Bits	Name	Description	Type	Reset Value
21	SA_WRITE	Set SA_WRITE Interrupt 0 = No effect 1 = Interrupt is set	R/W1S	0
22	SA_READ	Set SA_READ Interrupt 0 = No effect 1 = Interrupt is set	R/W1S	0
23	SA_OK	Set SA_OK Interrupt 0 = No effect 1 = Interrupt is set	R/W1S	0
24	MA_DIAG	Set MA_DIAG Interrupt 0 = No effect 1 = Interrupt is set	R/W1S	0
25:26	Reserved	Reserved	R	00
27	MA_COL	Set MA_COL Interrupt 0 = No effect 1 = Interrupt is set	R/W1S	0
28	MA_TMO	Set MA_TMO Interrupt 0 = No effect 1 = Interrupt is set	R/W1S	0
29	MA_NACK	Set MA_NACK Interrupt 0 = No effect 1 = Interrupt is set	R/W1S	0
30	MA_ATMO	Set MA_ATMO Interrupt 0 = No effect 1 = Interrupt is set	R/W1S	0
31	MA_OK	Set MA_OK Interrupt 0 = No effect 1 = Interrupt is set	R/W1S	0

13.2.11 I2C Slave Configuration Register

This register configures the slave interface portion of the I2C block. The slave interface is the logic that responds to transactions from an external master on the I2C bus.

Register name: I2C_SLV_CFG Reset value: 0xD0000030	Register offset: 0x1D12C

Bits	0	1	2	3	4	5	6	7
00:07	RD_EN	WR_EN	ALRT_EN	SLV_EN	Reserved			SLV_UNLK
08:15	Reserved							
16:23	Reserved							
24:31	Reserved	SLV_ADDR						

Bits	Name	Description	Type	Reset Value
0	RD_EN	Register Bus Read Enable This bit controls whether external masters can read registers internal to the Tsi574. The SLV_EN bit must also be set for this option to have any effect. 0 = Transactions that read the "Externally Visible I2C Internal Read Data Register" on page 435 will not invoke reads of the internal registers. 1 = Transactions that read the "Externally Visible I2C Internal Read Data Register" on page 435 will trigger reads of the internal register whose address is in the "Externally Visible I2C Internal Read Address Register" on page 434.	R/W	1
1	WR_EN	Register Bus Write Enable This bit controls whether external masters can write to Tsi574's internal registers. The SLV_EN bit must also be set for this option to have any effect.0 = Transactions that write the "Externally Visible I2C Internal Write Data Register" on page 433 will not invoke writes of the internal registers. 1 = Transactions that write the "Externally Visible I2C Internal Write Data Register" on page 433 will trigger writes of the internal register whose address is in the "Externally Visible I2C Internal Write Address Register" on page 432.	R/W	1

(Continued)

Bits	Name	Description	Type	Reset Value
2	ALRT_EN	Alert Address Enable 0 = Do not respond to read of the Alert Response Address of 0001100 1 = Respond to read of the Alert Response Address of 0001100 if any bits are set in the "Externally Visible I2C Status Register" on page 440. If enabled, the slave interface will respond to the Alert Response Address for a read transaction if the ALERT_FLAG is set in the "Externally Visible I2C Slave Access Status Register" on page 436. The response is to return the SLV_ADDR field followed by a 0 to the external master, then clear the ALERT_FLAG. If ALRT_EN is 0, then the Alert Response address may be used as a SLV_ADDR. If ALRT_EN is 1 and SLV_ADDR is also set to the alert response address, then the alert response behavior will take precedence.	R/W	0
3	SLV_EN	Slave Enable 0 = Slave is not enabled; SLV_ADDR is ignored. 1 = Slave interface is enabled; SLV_ADDR is responded to when transaction started by external master. When enabled, the slave interface will acknowledge transactions to the SLV_ADDR from an external master. If not enabled, then all transactions are NACK'd, except the Alert Response Address read, if ALRT_EN is 1. This bit controls access to the peripheral address space of the Tsi574. Access to the internal register space is also controlled by the RD_EN and WR_EN bits. If SLV_EN is 0 then internal register access is also disabled.	R/W	1
4:6	Reserved	Reserved	R	000
7	SLV_UNLK	Slave Address Unlock 0 = The LSB 2 bits of the SLV_ADDR are locked for writing (a write will leave those bits unchanged). 1 = The LSB 2 bits of the SLV_ADDR are unlocked, and can be changed by a write. This bit controls a write-protect on the 2 LSBs of the SLV_ADDR field. When 0 those bits are not writeable, which protects the power-up latch value of those bits. To change the bits, this SLV_UNLK bit must be written to 1 on the write performed to this register that is changing the SLV_ADDR[30:31] bits.	R/W	0
8:24	Reserved	Reserved	R	0x0000

(Continued)

Bits	Name	Description	Type	Reset Value
25:31	SLV_ADDR	Slave Address This is the device address for the Tsi574 as an I^2C slave. An external master uses this address to access the Tsi574 peripheral register space. For the slave interface to respond to this address, the SLV_EN bit must be set. The LSB two bits [30:31] of this field are latched at power-up from the state of input pins. This allows board configuration of up to four unique Tsi574 devices on the I2C bus. These two bits are then locked for writing until the SLV_UNLK bit is set to 1 during a write. This feature allows the boot load process to write a slave address value to this register without changing the power-up latch field. A SLV_ADDR of 0x00 is never valid, as that value is used for the I2C START_BYTE and General Call functions. The General Call functions are not supported by the Tsi574 and are ignored and NACK'd.	R/W	0x30

13.2.12 I²C Boot Control Register

This register controls the boot load sequence that is initiated following a chip reset of the Tsi574. The initial boot load operation is controlled by the reset state of this register. Some of the fields are also latched from device pins at power-up.

Once boot loading is in progress, the data read from the EEPROM can modify the contents of this register and redirect the loading to another EEPROM, or to another address within the same EEPROM. This process is called “chaining.” The progress of a boot load operation can be monitored using the “I²C Boot Load Diagnostic Progress Register” and “I²C Boot Load Diagnostic Configuration Register”.

This register can be read and written after boot loading is complete, but has no further effect on block operation.

Register name: I2C_BOOT_CNTRL **Reset value: Undefined**	**Register offset: 0x1D140**

Bits	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
00:07	CHAIN	PSIZE	BINC	BUNLK	Reserved			
08:15	Reserved	BOOT_ADDR						
16:23	PAGE_MODE			PADDR				
24:31	PADDR							

Bits	**Name**	**Description**	**Type**	**Reset Value**
00	CHAIN	Chain During Boot 0 = No chain 1 = Chain to new device This bit is set to invoke a chain operation during boot load. In order to chain, this bit must be set and the register load count must be at zero; that is, the write to this register must be the last one in the boot sequence within this EEPROM if chaining were not continuing the boot. Except for the BUNLK and PAGE_MODE fields, modifications to the remaining fields in this register have no effect unless this bit is set. The fields will change value, but they will not affect the boot load sequence. Once boot load is complete, this register has no further effect on block operation.	R/W	0

(Continued)

Bits	Name	Description	Type	Reset Value
01	PSIZE	Peripheral Address Size 0 = Use 1 byte for peripheral address 1 = Use 2 bytes for peripheral address This selects the number of bytes in the peripheral address. If 0 then only the least significant 5 bits of PADDR are used (+ 3 LSBs of 000). If 1 then all 13 bits of PADDR are used (+ 3 LSBs of 000). For 2-byte addressing, the MSB of the address is transmitted first on the I2C bus (see the PADDR field for an example). This field can be changed during the boot load, in conjunction with setting the CHAIN bit, in order to jump the boot load to a new boot device with different address size.	R/W	Undefined
02	BINC	Boot Address Increment 0 = Do not increment boot address when peripheral address overflows 1 = Increment the least significant 3 bits of the internal boot address when peripheral address overflows, then re-address device This option is valid only when PSIZE is 0, and is used to access devices that use the least significant 3 bits of their device address as a 256-byte page select (typically 2K EEPROMs). When enabled, and the 1-byte peripheral address wraps back to zero, the least significant 3 bits of the device address is incremented, followed by a Restart and a new device address cycle. The device address starts as the value of the BOOT_ADDR field, and is copied internally at boot start or upon a chain operation. It is the internal value that is incremented to simulate addressing a 2K EEPROM. This field can be changed during the boot load, in conjunction with setting the CHAIN bit, in order to jump the boot load to a new boot device with new page properties.	R/W	1
03	BUNLK	Boot Address Unlock 0 = The the least significant 2 bits of the BOOT_ADDR are locked for writing (a write will leave those bits unchanged). 1 = The the least significant 2 bits of the BOOT_ADDR are unlocked, and can be changed by a write. This bit controls a write-protect on the two least significant bits of the BOOT_ADDR field. When 0, those bits are not writeable, which protects the power-up latch value of those bits. To change the bits, this BUNLK bit must be written as 1 on the write performed to this register that is changing the BOOT_ADDR[14:15] bits.	R/W	0
04:08	Reserved	Reserved	R	0x00

(Continued)

Bits	Name	Description	Type	Reset Value
09:15	BOOT_ADDR	Boot Device Address This is the device address that the Tsi574 will access during the boot load sequence. The least significant two bits [14:15] of this field are latched at power-up from the state of the I2C_SA[1:0] pins. This allows board configuration of up to four unique Tsi574 devices on the I2C bus, each of which access a different EEPROM during boot load. These two bits are then locked for writing until the BUNLK bit is set to 1 during a write. This feature allows the boot load process to change the boot address value to this register without changing the power-up latch field. This field can be changed during the boot load in conjunction with setting the CHAIN bit, in order to jump the boot load to a new boot device. This starting address is copied internally at boot start or boot load, and it is the internal value that is incremented, as explained in the BINC field.	R/W	Undefined
16:18	PAGE_MODE	Page Mode 000 = 8 bytes 001 = 32 bytes 010 = 64 bytes 011 = 128 bytes 100 = 256 bytes 101 = 512 bytes 110 = 1024 bytes 111 = Infinite This field modifies the boot load process to adjust the boundary at which the boot device is re-addressed. In the default case, the boot load sequence reads 8 bytes, then does a Restart followed by the device and peripheral address phases. By changing this field, the device will be re-addressed only when the peripheral address crosses the indicated boundary, thus saving a considerable number of clock cycles during the boot. It is up to the programmer of the EEPROM to ensure that the addressed device can support consecutive reads up to the selected boundary. Some devices wrap at certain page boundaries, and this setting must be consistent with such limitations. A setting of 111 causes the entire boot load to be read sequentially, with two exceptions. No matter what the setting of this field, if the boot address is incremented due to the BINC mode being enabled, or if chaining occurs, then the device is readdressed. Changing this field during boot load will immediately affect the boot sequence.	R/W	000

(Continued)

Bits	Name	Description	Type	Reset Value
19:31	PADDR	Peripheral Address This is the most significant 5 or 13 bits of the peripheral address (depending on PSIZE setting). The least significant 3 bits are not programmable and are assumed 000; that is, the peripheral address must be aligned to a multiple of 8 address in the EEPROM. To form the peripheral address, this field is shifted left by 3 and then copied internally upon boot start or a chain operation. The internal address is then incremented as the boot load progresses. For 2-byte addressing, the MSB of the peripheral address is sent first. For example, setting this field to 0x0127 gives a peripheral address of (0x0127 << 3) = 0x0938. The first byte sent to the external device is 0x09 and the second byte is 0x38. This field can be changed during the boot load, in conjunction with setting the CHAIN bit, in order to jump the boot load to a new peripheral address.	R/W	0x0000

13.2.13 Externally Visible I2C Internal Write Address Register

This register contains the internal register address set by an external I2C master to be used for internal register writes when the "Externally Visible I2C Internal Write Data Register" is written. The address is forced to be 4-byte aligned (the 2 lowest bits are read-only).

This register is read-only from the register bus, and R/W from the I2C bus through the slave interface. This register corresponds to the I2C peripheral address 0x00 through 0x03.

Note: This register is also used during the boot load process to accumulate the address read from the EEPROM for each address/data pair. Therefore, at the end of the boot load process, this register will contain the last register address read from the EEPROM, or the first four bytes of the register count.

Register name: EXI2C_REG_WADDR **Reset value: 0x0000_0000**	**Register offset: 0x1D200**

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

Bits	**Name**	**Description**	**Type**	**Reset Value**
0:29	ADDR	Internal Register Write Address Register address to be used when a write to the "Externally Visible I2C Internal Write Data Register" invokes an internal register write. This address is 4-byte aligned. The specific byte accessed is controlled by the peripheral address within the data register. Only the least significant 24 bits are significant to the Tsi574. Bits [00:07] can be written but will not have any effect. This address auto-increments by 4 if WINC in the "Externally Visible I2C Internal Access Control Register" is set, and the MSB of the data (peripheral address 0x07) is written.	R	0x0000_0000
30:31	Reserved	Reserved	R	00

13.2.14 Externally Visible I2C Internal Write Data Register

This register contains the internal register data last written by an external I2C master through the slave interface. The register is read-only from the register bus, and R/W from the I2C bus through the slave interface.

This register corresponds to the I2C peripheral addresses 0x04 through 0x07.

Note: This register is also used during the boot load process to accumulate the data read from the EEPROM for each address/data pair. Therefore, at the end of the boot load process, this register will contain the last register data read from the EEPROM, or the last four bytes of the register count.

Register name: EXI2C_REG_WDATA **Reset value: 0x0000_0000**	**Register offset: 0x1D204**

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

Bits	**Name**	**Description**	**Type**	**Reset Value**
0:31	WDATA	Internal Register Write Data Data written by the external I2C master to be used for an internal register write. When WSIZE is configured for 4-byte access in the "Externally Visible I2C Internal Access Control Register" on page 438, the contents of this register are written to an internal register when the MSB is written by an external I2C master (peripheral address 0x07 = WDATA[00:07]). The register bus address is taken from the "Externally Visible I2C Internal Write Address Register" on page 432. Writes of other bytes (peripheral addresses 0x04-06) do not invoke internal accesses. Note: When the MSB of this register is written (peripheral address 0x07), the slave peripheral address wraps to 0x04 (the LSB of this register) instead of incrementing to 0x08. This allows an external master to write a block of internal registers without having to change the slave peripheral address, assuming WINC in the "Externally Visible I2C Internal Access Control Register" is set to auto-increment the WADDR. When 0x07 is read, the peripheral address increments to 0x08.	R	0

13.2.15 Externally Visible I2C Internal Read Address Register

This register contains the internal register address set by an external I2C master to be used for internal register reads when the "Externally Visible I2C Internal Read Data Register" is read. The address is forced to be 4-byte aligned (the 2 lowest bits are read-only).

This register is read-only from the register bus, and R/W from the I2C bus through the slave interface. This register corresponds to the I2C peripheral address 0x10 through 0x13.

Register name: EXI2C_REG_RADDR **Reset value: 0x0000_0000**	**Register offset: 0x1D210**

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

Bits	**Name**	**Description**	**Type**	**Reset Value**
0:29	ADDR	Internal Register Read Address Register address to be used when a read to the "Externally Visible I2C Internal Read Data Register" invokes an internal register read. This address is 4-byte aligned. The specific byte accessed is controlled by the peripheral address within the data register. Only the least significant 24 bits are significant to the Tsi574. Bits [00:07] can be written, but will not have any effect. This address auto-increments by 4 if RINC in the "Externally Visible I2C Internal Access Control Register" is set, and the LSB of the data (peripheral address 0x14) is read.	R	0x0000_0000
30:31	Reserved	Reserved	R	00

13.2.16 Externally Visible I2C Internal Read Data Register

This register contains the internal register data last read by an external I2C master through the slave interface. The register is read-only from both the register bus and the I2C bus through the slave interface.

This register corresponds to the I2C peripheral addresses 0x14 through 0x17.

Register name: EXI2C_REG_RDATA Reset value: 0x0000_0000	Register offset: 0x1D214

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

Bits	Name	Description	Type	Reset Value
0:31	RDATA	Internal Register Read Data Data read by the external I2C master as a side-effect of reading this register. When RSIZE is configured for 4-byte access in the "Externally Visible I2C Internal Access Control Register" on page 438, this register is updated by the read of an internal register when the LSB is read by an external I2C master (peripheral address 0x14 = RDATA[24:31]). The register bus address is taken from the "Externally Visible I2C Internal Read Address Register" on page 434. Reads of subsequent bytes (peripheral addresses 0x15-17) do not invoke internal accesses. Note: When the MSB of this register is read (peripheral address 0x17), the slave peripheral address wraps to 0x14 (the LSB of this register) instead of incrementing to 0x18. This allows an external master to read a block of internal registers without having to change the slave peripheral address, assuming RINC in the "Externally Visible I2C Internal Access Control Register" is set to auto-increment the RADDR. When 0x17 is written, the peripheral address increments to 0x18.	R	0

13.2.17 Externally Visible I2C Slave Access Status Register

This register provides status indications to an external I2C master. It is read-only from both the register bus and the I2C bus through the slave interface. This register corresponds to the I2C peripheral addresses 0x20 through 0x23.

Note: This register is affected by a reset controlled by the "I2C Reset Register". All status will be cleared.

Register name: EXI2C_ACC_STAT **Reset value: 0x0000_0000**	**Register offset: 0x1D220**

Bits	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
00:07	Reserved							
08:15	Reserved							
16:23	Reserved							
24:31	ACC_OK	Reserved			OMB_ FLAG	IMB_ FLAG	Reserved	ALERT_ FLAG

Bits	**Name**	**Description**	**Type**	**Reset Value**
00:23	Reserved	Reserved	R	0x00_0000
24	ACC_OK	Internal Register Access OK 0 = No access, or access in progress 1 = Access was successful This bit is set when a slave access successfully reads or writes data to an internal register through the "Externally Visible I2C Internal Write Data Register" or "Externally Visible I2C Internal Read Data Register". Reading this bit returns the last status of the bit. If read through the slave interface (through peripheral address 0x20), the bit is then cleared to 0. The bit is not cleared to 0 when read by a host or indirectly through the EXI2C_REG_RADDR / EXI2C_REG_RDATA function.	R	0
25:27	Reserved	Reserved	R	000

(Continued)

Bits	Name	Description	Type	Reset Value
28	OMB_FLAG	Outgoing Mailbox Flag 0 = Outgoing mailbox empty 1 = New data in the outgoing mailbox This bit is set when data is written to the outgoing mailbox register (“Externally Visible I2C Outgoing Mailbox Register”) by software. This bit remains set (flag up) until an external I2C master reads the outgoing mailbox register, and the bit is then cleared (flag down). When the mailbox is read, the OMB_EMPTY interrupt is asserted. A mailbox read is considered complete when the external master issues a STOP condition to end the transaction during which any bytes in the mailbox were written.	R	0
29	IMB_FLAG	Incoming Mailbox Flag 0 = Incoming mailbox empty 1 = New data in the incoming mailbox This bit is set when data is written to the incoming mailbox register (“Externally Visible I2C Incoming Mailbox Register”) by an external I2C master. This bit remains set (flag up) until software reads the incoming mailbox register, and the bit is then cleared (flag down). When the mailbox is written and the flag is set, the IMB_FULL interrupt is asserted. A mailbox read is considered complete when the external master issues a STOP condition to end the transaction during which any bytes in the mailbox were written.	R	0
30	Reserved	Reserved	R	0
31	ALERT_FLAG	Alert Response Flag 0 = No alert 1 = Alert response active This bit is set when the Alert Response would trigger, as defined in the “Externally Visible I2C Status Register”. It is cleared by a successful response to the Alert Response Address or if the global status no longer requires the alert to be asserted. On a hard reset, this flag will assert immediately due to EXI2C_STAT[RESET] asserting.	R	0

13.2.18 Externally Visible I2C Internal Access Control Register

This register allows an external I2C master to configure the functionality for internal register accesses through the slave interface. This register is read-only from the register bus and R/W from the I2C bus through the slave interface.

The fields in this register control the size and auto-increment functions when internal register accesses are performed by an external master through the slave interface.

This register corresponds to the I2C peripheral addresses 0x24 through 0x27.

Register name: EXI2C_ACC_CNTRL **Reset value: 0x0000_00A0**	**Register offset: 0x1D224**

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

Bits	**Name**	**Description**	**Type**	**Reset Value**
00:23	Reserved	Reserved	R	0x00_0000
24:25	RSIZE	Internal Register Read Access Size 00 = 1 byte (Reserved) 01 = 2 bytes 10 = 4 bytes – An internal register read is invoked once for each internal register, loading all 4 bytes in the "Externally Visible I2C Internal Read Data Register". The read is performed when the LSB of the data register is read (peripheral address 0x14). 11 = 8 bytes (Reserved) All Reserved settings will result in internal read accesses being disabled.	R	10

(Continued)

Bits	Name	Description	Type	Reset Value
26:27	WSIZE	Internal Register Write Access Size 00 = 1 byte (Reserved) 01 = 2 bytes 10 = 4 bytes – An internal register write is invoked once for each internal register, writing all 4 bytes from the "Externally Visible I2C Internal Write Data Register". The write is performed when the MSB of the data register is written (peripheral address 0x07). 11 = 8 bytes (Reserved) All Reserved settings will result in internal writes accesses being disabled.	R	10
28	RINC	Enable Auto-Incrementing on Internal Register Reads 0 = The "Externally Visible I2C Internal Read Address Register" is unchanged after reads performed to the "Externally Visible I2C Internal Read Data Register" 1 = The "Externally Visible I2C Internal Read Address Register" is incremented by 4 after reads performed to the LSB of the "Externally Visible I2C Internal Read Data Register" (peripheral address 0x14) so that the address points to the next internal register address. When auto-incrementing is on, consecutive internal registers can be read in one I2C transaction without the need to reset the peripheral address because the peripheral address wraps from 0x17 back to 0x14. If auto-incrementing is off, then the same internal register can be read multiple times in a single I2C transaction. The latter could be useful for polling a status register.	R	0
29	WINC	Enable Auto-Incrementing on Internal Register Writes 0 = The "Externally Visible I2C Internal Write Address Register" is unchanged after reads performed to the "Externally Visible I2C Internal Write Data Register". 1 = The "Externally Visible I2C Internal Write Address Register" is incremented by 4 after writes performed to the MSB of the "Externally Visible I2C Internal Write Data Register" (peripheral address 0x07) so that the address points to the next internal register address. When auto-incrementing is on, consecutive internal registers can be written in one I2C transaction with the need to reset the peripheral address because the peripheral address wraps from 0x07 back to 0x04. If auto-incrementing is off, then the same internal register can be written multiple times in a single I2C transaction.	R	0
30:31	Reserved	Reserved	R	00

13.2.19 Externally Visible I2C Status Register

This register provides a summary view of status of the Tsi574. It can be polled by an external system management device. Any bit masked by its related enable, changing from 0 to 1, will cause ALERT_FLAG to be set in the "Externally Visible I2C Slave Access Status Register", and the Tsi574 to respond to the Alert Response Address if the ALRT_EN bit is set in the "I2C Slave Configuration Register". The related enables are present in the "Externally Visible I2C Enable Register". If all masked status bits are 0, then the ALERT_FLAG clears. The ALERT_FLAG also clears when the slave responds to the Alert Response Address, and not set again until there is a change in the status.

Bits [0] are read only from the register bus, but R/W1C from the I2C bus through the slave interface. They are set when the corresponding event occurs within the Tsi574, and held asserted until an external I2C master writes a 1 to that position to clear the event. If an event is still asserting at the time the W1C occurs, the bit remains set.

The software status bits [1:3] are R/W from the register bus. They can be set or cleared by software, and thereby used for any system purpose. An external I2C master can write 1 to those bits to clear them. If the W1C occurs at the same time as software is writing the bit, the software written value will take precedence.

This register corresponds to the I2C peripheral addresses 0x80 through 0x83. This register is affected by a reset controlled by the "I2C Reset Register". All status will be cleared, including the software status bits. Chip status will re-assert after a SRESET is released in the "I2C Reset Register" only if that chip event occurs again.

Register name: EXI2C_STAT **Reset value: 0xFFFF_FFFF**	**Register offset: 0x1D280**

Bits	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
00:07	RESET	SW_ STAT2	SW_ STAT1	SW_ STAT0	OMBW	IMBR	I2C	TEA
08:15	RCS	MCS	Reserved			LOGICAL	MC_LAT	MCE
16:23	Reserved							
24:31	PORT7	PORT6	PORT5	PORT4	PORT3	PORT2	PORT1	PORT0

Bits	**Name**	**Description**	**Type**	**Reset Value**
0	RESET	Enable RESET Alert Response 0 = Status asserted will not enable setting ALERT_FLAG 1 = Status asserted will enable setting ALERT_FLAG	R/W	1
1	SW_STAT2	Enable SW_STAT2 Alert Response 0 = Status asserted will not enable setting ALERT_FLAG 1 = Status asserted will enable setting ALERT_FLAG	R/W	1

(Continued)

Bits	Name	Description	Type	Reset Value
2	SW_STAT1	Enable SW_STAT1 Alert Response 0 = Status asserted will not enable setting ALERT_FLAG 1 = Status asserted will enable setting ALERT_FLAG	R/W	1
3	SW_STAT0	Enable SW_STAT0 Alert Response 0 = Status asserted will not enable setting ALERT_FLAG 1 = Status asserted will enable setting ALERT_FLAG	R/W	1
4	OMBW	Enable Outgoing Mailbox Written 0 = Status asserted will not enable setting ALERT_FLAG 1 = Status asserted will enable setting ALERT_FLAG	R/W	1
5	IMBR	Enable Incoming Mailbox Read 0 = Status asserted will not enable setting ALERT_FLAG 1 = Status asserted will enable setting ALERT_FLAG	R/W	1
6	I2C	Enable I^2C Alert Response 0 = Status asserted will not enable setting ALERT_FLAG 1 = Status asserted will enable setting ALERT_FLAG	R/W	1
7	TEA	Enable TEA Alert Response 0 = Status asserted will not enable setting ALERT_FLAG 1 = Status asserted will enable setting ALERT_FLAG	R/W	1
8	RCS	Enable RCS Alert Response 0 = Status asserted will not enable setting ALERT_FLAG 1 = Status asserted will enable setting ALERT_FLAG	R/W	1
9	MCS	Enable MCS Alert Response 0 = Status asserted will not enable setting ALERT_FLAG 1 = Status asserted will enable setting ALERT_FLAG	R/W	1
10:12	Reserved	Reserved These bits are unused in the Tsi574. The enables can be changed, but have no effect.	R/W	111
13	LOGICAL	Enable LOGICAL Alert Response 0 = Status asserted will not enable setting ALERT_FLAG 1 = Status asserted will enable setting ALERT_FLAG	R/W	1
14	MC_LAT	Enable MC_LAT Alert Response 0 = Status asserted will not enable setting ALERT_FLAG 1 = Status asserted will enable setting ALERT_FLAG	R/W	1
15	MCE	Enable MCE Alert Response 0 = Status asserted will not enable setting ALERT_FLAG 1 = Status asserted will enable setting ALERT_FLAG	R/W	1

(Continued)

Bits	Name	Description	Type	Reset Value
16:23	Reserved	Reserved	R/W	0xFF
24	PORT7	Enable PORT7 Alert Response 0 = Status asserted will not enable setting ALERT_FLAG 1 = Status asserted will enable setting ALERT_FLAG	R/W	1
25	PORT6	Enable PORT6 Alert Response 0 = Status asserted will not enable setting ALERT_FLAG 1 = Status asserted will enable setting ALERT_FLAG	R/W	1
26	PORT5	Enable PORT5 Alert Response 0 = Status asserted will not enable setting ALERT_FLAG 1 = Status asserted will enable setting ALERT_FLAG	R/W	1
27	PORT4	Enable PORT4 Alert Response 0 = Status asserted will not enable setting ALERT_FLAG 1 = Status asserted will enable setting ALERT_FLAG	R/W	1
28	PORT3	Enable PORT3 Alert Response 0 = Status asserted will not enable setting ALERT_FLAG 1 = Status asserted will enable setting ALERT_FLAG	R/W	1
29	PORT2	Enable PORT2 Alert Response 0 = Status asserted will not enable setting ALERT_FLAG 1 = Status asserted will enable setting ALERT_FLAG	R/W	1
30	PORT1	Enable PORT1 Alert Response 0 = Status asserted will not enable setting ALERT_FLAG 1 = Status asserted will enable setting ALERT_FLAG	R/W	1
31	PORT0	Enable PORT0 Alert Response 0 = Status asserted will not enable setting ALERT_FLAG 1 = Status asserted will enable setting ALERT_FLAG	R/W	1

13.2.20 Externally Visible I2C Enable Register

Any bit set in this register will enable the equivalent bit in the "Externally Visible I2C Status Register" to set the ALERT_FLAG. These enables do not affect whether events are set in the global status register, only whether the asserted events are allowed to set the ALERT_FLAG when changing from 0 to 1. If an event is already asserted in the status when the related enable is changed from 0 to 1, this is equivalent to the event asserting, and the ALERT_FLAG will be set.

This register is R/W from either the register bus or from the I2C bus through the slave interface. If the register is written by both at the same time, the register bus interface will take precedence.

This register corresponds to the I2C peripheral addresses 0x84 through 0x87.

Register name: EXI2C_STAT_ENABLE **Reset value: 0xFFFF_FFFF**	**Register offset: 0x1D284**

Bits	0	1	2	3	4	5	6	7
00:07	RESET	SW_ STAT2	SW_ STAT1	SW_ STAT0	OMBW	IMBR	I2C	TEA
08:15	RCS	MCS	Reserved			LOGICAL	MC_LAT	MCE
16:23	Reserved							
24:31	PORT7	PORT6	PORT5	PORT4	PORT3	PORT2	PORT1	PORT0

Bits	Name	Description	Type	Reset Value
0	RESET	Enable RESET Alert Response 0 = Status asserted will not enable setting ALERT_FLAG 1 = Status asserted will enable setting ALERT_FLAG	R/W	1
1	SW_STAT2	Enable SW_STAT2 Alert Response 0 = Status asserted will not enable setting ALERT_FLAG 1 = Status asserted will enable setting ALERT_FLAG	R/W	1
2	SW_STAT1	Enable SW_STAT1 Alert Response 0 = Status asserted will not enable setting ALERT_FLAG 1 = Status asserted will enable setting ALERT_FLAG	R/W	1
3	SW_STAT0	Enable SW_STAT0 Alert Response 0 = Status asserted will not enable setting ALERT_FLAG 1 = Status asserted will enable setting ALERT_FLAG	R/W	1
4	OMBW	Enable Outgoing Mailbox Written 0 = Status asserted will not enable setting ALERT_FLAG 1 = Status asserted will enable setting ALERT_FLAG	R/W	1

(Continued)

Bits	Name	Description	Type	Reset Value
5	IMBR	Enable Incoming Mailbox Read 0 = Status asserted will not enable setting ALERT_FLAG 1 = Status asserted will enable setting ALERT_FLAG	R/W	1
6	I2C	Enable I^2C Alert Response 0 = Status asserted will not enable setting ALERT_FLAG 1 = Status asserted will enable setting ALERT_FLAG	R/W	1
7	TEA	Enable TEA Alert Response 0 = Status asserted will not enable setting ALERT_FLAG 1 = Status asserted will enable setting ALERT_FLAG	R/W	1
8	RCS	Enable RCS Alert Response 0 = Status asserted will not enable setting ALERT_FLAG 1 = Status asserted will enable setting ALERT_FLAG	R/W	1
9	MCS	Enable MCS Alert Response 0 = Status asserted will not enable setting ALERT_FLAG 1 = Status asserted will enable setting ALERT_FLAG	R/W	1
10:12	Reserved	Reserved These bits are unused in the Tsi574. The enables can be changed, but have no effect.	R/W	111
13	LOGICAL	Enable LOGICAL Alert Response 0 = Status asserted will not enable setting ALERT_FLAG 1 = Status asserted will enable setting ALERT_FLAG	R/W	1
14	MC_LAT	Enable MC_LAT Alert Response 0 = Status asserted will not enable setting ALERT_FLAG 1 = Status asserted will enable setting ALERT_FLAG	R/W	1
15	MCE	Enable MCE Alert Response 0 = Status asserted will not enable setting ALERT_FLAG 1 = Status asserted will enable setting ALERT_FLAG	R/W	1
16:23	Reserved	Reserved	R/W	0xFF
24	PORT7	Enable PORT7 Alert Response 0 = Status asserted will not enable setting ALERT_FLAG 1 = Status asserted will enable setting ALERT_FLAG	R/W	1
25	PORT6	Enable PORT6 Alert Response 0 = Status asserted will not enable setting ALERT_FLAG 1 = Status asserted will enable setting ALERT_FLAG	R/W	1

(Continued)

Bits	Name	Description	Type	Reset Value
26	PORT5	Enable PORT5 Alert Response 0 = Status asserted will not enable setting ALERT_FLAG 1 = Status asserted will enable setting ALERT_FLAG	R/W	1
27	PORT4	Enable PORT4 Alert Response 0 = Status asserted will not enable setting ALERT_FLAG 1 = Status asserted will enable setting ALERT_FLAG	R/W	1
28	PORT3	Enable PORT3 Alert Response 0 = Status asserted will not enable setting ALERT_FLAG 1 = Status asserted will enable setting ALERT_FLAG	R/W	1
29	PORT2	Enable PORT2 Alert Response 0 = Status asserted will not enable setting ALERT_FLAG 1 = Status asserted will enable setting ALERT_FLAG	R/W	1
30	PORT1	Enable PORT1 Alert Response 0 = Status asserted will not enable setting ALERT_FLAG 1 = Status asserted will enable setting ALERT_FLAG	R/W	1
31	PORT0	Enable PORT0 Alert Response 0 = Status asserted will not enable setting ALERT_FLAG 1 = Status asserted will enable setting ALERT_FLAG	R/W	1

13.2.21 Externally Visible I2C Outgoing Mailbox Register

This register is the outgoing mailbox, allowing the processor to communicate data to an external I2C master. The register is R/W from the register bus, and read-only from the I2C bus through the slave interface.

This register corresponds to the I2C peripheral addresses 0x90 through 0x93.

Register name: EXI2C_MBOX_OUT Reset value: 0x0000_0000	Register offset: 0x1D290

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

Bits	Name	Description	Type	Reset Value
0:31	DATA	Mailbox data to be transferred to an external I2C master. Every write to this register by software sets the OMB_FLAG bit in the "Externally Visible I2C Slave Access Status Register", indicating data is available in the outgoing mailbox. When this register is read by an external master, the OMB_FLAG bit is cleared, and an OMB_EMPTY interrupt is asserted if the OMB_FLAG bit was set. This register is read-only through the I2C slave interface. **Note**: A read is considered complete when the STOP condition is seen on the I2C bus, and one or more bytes in this register were read by the external master since the preceding START condition.	R/W	0

13.2.22 Externally Visible I2C Incoming Mailbox Register

This register is the incoming mailbox, allowing an external I2C master to communicate data to the host processor. The register is read-only from the register bus, and R/W from the I2C bus through the slave interface.

This register corresponds to the I2C peripheral addresses 0x94 through 0x97.

Register name: EXI2C_MBOX_IN Reset value: 0x0000_0000	Register offset: 0x1D294

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

Bits	Name	Description	Type	Reset value
0:31	DATA	Mailbox data transferred from an external I2C master. Every write to this register sets the IMB_FLAG in the "Externally Visible I2C Slave Access Status Register", and asserts an IMB_FULL interrupt. When software reads this register, the IMB_FLAG is cleared. **Note**: This register is writable only through the I2C slave interface.	R	0

13.2.23 I²C Event and Event Snapshot Registers

These registers indicate events that occur within the I²C block. For the I2C_EVENT register, each bit is an "or" of the corresponding bit in the "I²C New Event Register" and the I2C_SNAP_EVENT register. The I2C_SNAP_EVENT register contains those events that were asserted prior to the last snapshot. A snapshot is taken when the "I²C Interrupt Status Register" is read.

Each bit in these registers are write-one-to-clear. Writing a 1 to a bit position in the I2C_EVENT register will clear the event in both the snapshot and new event registers. Writing a 1 to a bit position in the I2C_SNAP_EVENT register will clear the bit only in that register. Writing a 1 to a bit position in the I2C_INT_STAT register will clear all related event bits in the I2C_SNAP_EVENT register, provided those events are enabled in the I2C_EVENT_ENB register. Bits from the I2C_EVENT register are masked (enabled) by the corresponding bits in the "I²C Enable Event Register", and then determine whether a related bit in the I2C_INT_STAT register is set.

Note: These registers are affected by a reset controlled by the "I²C Reset Register". All events will be cleared and will not assert while SRESET is asserted in the "I²C Reset Register".

Register name: I2C_{EVENT, SNAP_EVENT} **Reset value: 0x0000_0000**	**Register offset: 0x1D300, 1D304**

Bits	0	1	2	3	4	5	6	7
00:07	Reserved	SDW	SDR	SD	Reserved	DTIMER	DHIST	DCMDD
08:15	IMBW	OMBR	Reserved	SCOL	STRTO	SBTTO	SSCLTO	Reserved
16:23	Reserved	MTD	Reserved	BLTO	BLERR	BLSZ	BLNOD	BLOK
24:31	Reserved		MNACK	MCOL	MTRTO	MBTTO	MSCLTO	MARBTO

Bits	Name	Description	Type	Reset Value
00	Reserved	Reserved	R	0
01	SDW	Slave Internal Register Write Done Event 0 = Event not asserted 1 = Slave interface completed a transaction for an external master that resulted in a write to an internal register	R/W1C	0
02	SDR	Slave Internal Register Read Done Event 0 = Event not asserted 1 = Slave interface completed a transaction for an external master that resulted in a read to an internal register	R/W1C	0
03	SD	Slave Transaction Done Event 0 = Event not asserted 1 = Slave interface completed an I²C transaction for an external master with no detectable error	R/W1C	0

(Continued)

Bits	Name	Description	Type	Reset Value
04	Reserved	Reserved	R	0
05	DTIMER	Diagnostic Timer Expired Event 0 = Event not asserted 1 = Diagnostic timer has expired This event does not assert during the boot load sequence. The BLTO will assert instead.	R/W1C	0
06	DHIST	Diagnostic History Filling Event 0 = Event not asserted 1 = Diagnostic history recorded the 8th event	R/W1C	0
07	DCMDD	Diagnostic Command Done Event 0 = Event not asserted 1 = Master interface completed the diagnostic command	R/W1C	0
08	IMBW	Incoming Mailbox Write Event 0 = Event not asserted 1 = Slave interface completed a write transaction to the incoming mailbox	R/W1C	0
09	OMBR	Outgoing Mailbox Read Event 0 = Event not asserted 1 = Slave interface completed a read transaction to the outgoing mailbox when the OMB_FLAG was set The event is asserted only if the mailbox was full.	R/W1C	0
10	Reserved	Reserved	R	0
11	SCOL	Slave Collision Detect Event 0 = Event not asserted 1 = Slave interface detected a bit collision on the I^2C bus during a slave transaction initiated by an external master. The slave interface was not able to successfully assert a 1 for a data bit or for a NACK.	R/W1C	0
12	STRTO	Slave Transaction Timeout Event 0 = Event not asserted 1 = Transaction timer expired during a slave transaction initiated by an external master	R/W1C	0
13	SBTTO	Slave Byte Timeout Event 0 = Event not asserted 1 = Byte timer expired during a slave transaction initiated by an external master	R/W1C	0

(Continued)

Bits	Name	Description	Type	Reset Value
14	SSCLTO	Slave I2C_SCLK Low Timeout Event 0 = Event not asserted 1 = I2C_SCLK low timer expired during a slave transaction initiated by an external master	R/W1C	0
15:16	Reserved	Reserved	R	00
17	MTD	Master Transaction Done Event 0 = Event not asserted 1 = Master interface completed the I^2C transaction initiated through the "I^2C Master Control Register"	R/W1C	0
18	Reserved	Reserved	R	0
19	BLTO	Boot Load Timeout Event 0 = Event not asserted 1 = Boot load timer expired This event only asserts during the boot load sequence if the Boot/Diagnostic timer expires. During normal operation, the DTIMER event will assert.	R/W1C	0
20	BLERR	Boot Load Error Event 0 = Event not asserted 1 = The boot load sequence failed due to an error during register reading: a protocol error (NACK when ACK expected), an I2C_SCLK low timer, collision after the device addressing phase, or the last six bytes of a register count field not being 0xFF. This error will be qualified by the MNACK, MCOL or MSCLTO event. The last data read from the EEPROM is visible in the EXI2C_REG_WADDR and EXI2C_REG_WDATA registers.	R/W1C	0
21	BLSZ	Boot Load Size Error Event 0 = Event not asserted 1 = The boot load sequence aborted because the count field is incorrect, indicating an improperly loaded boot EEPROM. The register count is visible in the EXI2C_REG_WADDR register.	R/W1C	0
22	BLNOD	Boot Load No Device Event 0 = Event not asserted 1 = The boot load sequencer received a NACK 6 times when trying to address the slave device. No device is responding to the boot load device address.	R/W1C	0

(Continued)

Bits	Name	Description	Type	Reset Value
23	BLOK	Boot Load OK Event 0 = Event not asserted 1 = The boot load sequence completed with no detectable errors. This bit is also asserted if boot load is disabled upon power up.	R/W1C	0
24:25	Reserved	Reserved	R	00
26	MNACK	Master NACK Received Event 0 = Event not asserted 1 = Master interface received a NACK from a slave device during a transaction initiated through the "I2C Master Control Register". This event can also assert during boot load, and provides more information on the source of a BLERR event.	R/W1C	0
27	MCOL	Master Collision Detect Event 0 = Event not asserted 1 = Master interface lost arbitration after it addressed the slave device during a transaction initiated through the "I2C Master Control Register". Another master is competing for access to the same slave device. This event can also assert during boot load, and provides more information on the source of a BLERR event.	R/W1C	0
28	MTRTO	Master Transaction Timeout Event 0 = Event not asserted 1 = Transaction timeout timer expired during a transaction initiated through the "I2C Master Control Register"	R/W1C	0
29	MBTTO	Master Byte Timeout Event 0 = Event not asserted 1 = Byte timeout timer expired during a transaction initiated through the "I2C Master Control Register"	R/W1C	0
30	MSCLTO	Master I2C_SCLK Low Timeout Event 0 = Event not asserted 1 = SCL_TO timeout timer expired during a transaction initiated through the "I2C Master Control Register". Another device is holding the I2C_SCLK signal low. This event can also assert during boot load, and provides more information on the source of a BLERR event.	R/W1C	0
31	MARBTO	Master Arbitration Timeout Event 0 = Event not asserted 1 = Arbitration timeout timer expired during a transaction initiated through the "I2C Master Control Register". Another master has control of the I2C bus.	R/W1C	0

13.2.24 I²C New Event Register

This register indicates events that occurred since the last snapshot. This register is write-one-to-set. Writing a 1 to a bit position will set the event for diagnostic purposes. The register is cleared by writing to the I2C_EVENT register (see "I²C Event and Event Snapshot Registers") or by creating a snapshot by reading the "I²C Interrupt Status Register". For individual event descriptions, see the I2C_EVENT register.

Note: This register is affected by a reset controlled by the "I²C Reset Register". All events will be cleared and will not assert while SRESET is asserted in the "I²C Reset Register".

Register name: I2C_NEW_EVENT **Reset value: 0x0000_0000**	**Register offset: 0x1D308**

Bits	0	1	2	3	4	5	6	7
00:07	Reserved	SDW	SDR	SD	Reserved	DTIMER	DHIST	DCMDD
08:15	IMBW	OMBR	Reserved	SCOL	STRTO	SBTTO	SSCLTO	Reserved
16:23	Reserved	MTD	Reserved	BLTO	BLERR	BLSZ	BLNOD	BLOK
24:31	Reserved		MNACK	MCOL	MTRTO	MBTTO	MSCLTO	MARBTO

Bits	Name	Description	Type	Reset Value
00	Reserved	Reserved	R	0
01	SDW	Slave Internal Register Write Done Event 0 = Event not asserted 1 = Event asserted	R/W1S	0
02	SDR	Slave Internal Register Read Done Event 0 = Event not asserted 1 = Event asserted	R/W1S	0
03	SD	Slave Transaction Done Event 0 = Event not asserted 1 = Event asserted	R/W1S	0
04	Reserved	Reserved	R	0
05	DTIMER	Diagnostic Timer Expired Event 0 = Event not asserted 1 = Event asserted	R/W1S	0
06	DHIST	Diagnostic History Filling Event 0 = Event not asserted 1 = Event asserted	R/W1S	0

(Continued)

Bits	Name	Description	Type	Reset Value
07	DCMDD	Diagnostic Command Done Event 0 = Event not asserted 1 = Event asserted	R/W1S	0
08	IMBW	Incoming Mailbox Write Event 0 = Event not asserted 1 = Event asserted	R/W1S	0
09	OMBR	Outgoing Mailbox Read Event 0 = Event not asserted 1 = Event asserted	R/W1S	0
10	Reserved	Reserved	R	0
11	SCOL	Slave Collision Detect Event 0 = Event not asserted 1 = Event asserted	R/W1S	0
12	STRTO	Slave Transaction Timeout Event 0 = Event not asserted 1 = Event asserted	R/W1S	0
13	SBTTO	Slave Byte Timeout Event 0 = Event not asserted 1 = Event asserted	R/W1S	0
14	SSCLTO	Slave I2C_SCLK Low Timeout Event 0 = Event not asserted 1 = Event asserted	R/W1S	0
15:16	Reserved	Reserved	R	00
17	MTD	Master Transaction Done Event 0 = Event not asserted 1 = Event asserted	R/W1S	0
18	Reserved	Reserved	R	0
19	BLTO	Boot Load Timeout Event 0 = Event not asserted 1 = Event asserted	R/W1S	0
20	BLERR	Boot Load Error Event 0 = Event not asserted 1 = Event asserted	R/W1S	0

(Continued)

Bits	Name	Description	Type	Reset Value
21	BLSZ	Boot Load Size Error Event 0 = Event not asserted 1 = Event asserted	R/W1S	0
22	BLNOD	Boot Load No Device Event 0 = Event not asserted 1 = Event asserted	R/W1S	0
23	BLOK	Boot Load OK Event 0 = Event not asserted 1 = Event asserted	R/W1S	0
24:25	Reserved	Reserved	R	00
26	MNACK	Master NACK Received Event 0 = Event not asserted 1 = Event asserted	R/W1S	0
27	MCOL	Master Collision Detect Event 0 = Event not asserted 1 = Event asserted	R/W1S	0
28	MTRTO	Master Transaction Timeout Event 0 = Event not asserted 1 = Event asserted	R/W1S	0
29	MBTTO	Master Byte Timeout Event 0 = Event not asserted 1 = Event asserted	R/W1S	0
30	MSCLTO	Master I2C_SCLK Low Timeout Event 0 = Event not asserted 1 = Event asserted	R/W1S	0
31	MARBTO	Master Arbitration Timeout Event 0 = Event not asserted 1 = Event asserted	R/W1S	0

13.2.25 I2C Enable Event Register

This register modifies the function of the I2C_EVENT register (see "I2C Event and Event Snapshot Registers"). Each bit in this register enables (1) or disables (0) the corresponding event in the I2C_EVENT register from asserting in the "I2C Interrupt Status Register".

Register name: I2C_EVENT_ENB Reset value: 0x74DE_5F3F	Register offset: 0x1D30C

Bits	0	1	2	3	4	5	6	7
00:07	Reserved	SDW	SDR	SD	Reserved	DTIMER	DHIST	DCMDD
08:15	IMBW	OMBR	Reserved	SCOL	STRTO	SBTTO	SSCLTO	Reserved
16:23	Reserved	MTD	Reserved	BLTO	BLERR	BLSZ	BLNOD	BLOK
24:31	Reserved		MNACK	MCOL	MTRTO	MBTTO	MSCLTO	MARBTO

Bits	Name	Description	Type	Reset Value
00	Reserved	Reserved	R	0
01	SDW	Slave Internal Register Write Done Enable 0 = Event does not assert to interrupt status 1 = Event will assert in interrupt status	R/W	1
02	SDR	Slave Internal Register Read Done Enable 0 = Event does not assert to interrupt status 1 = Event will assert in the interrupt status	R/W	1
03	SD	Slave Transaction Done Enable 0 = Event does not assert to interrupt status 1 = Event will assert in the interrupt status	R/W	1
04	Reserved	Reserved	R	0
05	DTIMER	Diagnostic Timer Expired Enable 0 = Event does not assert to interrupt status 1 = Event will assert in the interrupt status	R/W	1
06	DHIST	Diagnostic History Filling Enable 0 = Event does not assert to interrupt status 1 = Event will assert in the interrupt status	R/W	0
07	DCMDD	Diagnostic Command Done Enable 0 = Event does not assert to interrupt status 1 = Event will assert in the interrupt status	R/W	0

(Continued)

Bits	Name	Description	Type	Reset Value
08	IMBW	Incoming Mailbox Write Enable 0 = Event does not assert to interrupt status 1 = Event will assert in the interrupt status	R/W	1
09	OMBR	Outgoing Mailbox Read Enable 0 = Event does not assert to interrupt status 1 = Event will assert in the interrupt status	R/W	1
10	Reserved	Reserved	R	0
11	SCOL	Slave Collision Detect Enable 0 = Event does not assert to interrupt status 1 = Event will assert in the interrupt status	R/W	1
12	STRTO	Slave Transaction Timeout Enable 0 = Event does not assert to interrupt status 1 = Event will assert in the interrupt status	R/W	1
13	SBTTO	Slave Byte Timeout Enable 0 = Event does not assert to interrupt status 1 = Event will assert in the interrupt status	R/W	1
14	SSCLTO	Slave I2C_SCLK Low Timeout Enable 0 = Event does not assert to interrupt status 1 = Event will assert in the interrupt status	R/W	1
15:16	Reserved	Reserved	R	00
17	MTD	Master Transaction Done Enable 0 = Event does not assert to interrupt status 1 = Event will assert in the interrupt status	R/W	1
18	Reserved	Reserved	R	0
19	BLTO	Boot Load Timeout Enable 0 = Event does not assert to interrupt status 1 = Event will assert in the interrupt status	R/W	1
20	BLERR	Boot Load Error Enable 0 = Event does not assert to interrupt status 1 = Event will assert in the interrupt status	R/W	1
21	BLSZ	Boot Load Size Error Enable 0 = Event does not assert to interrupt status 1 = Event will assert in the interrupt status	R/W	1

(Continued)

Bits	Name	Description	Type	Reset Value
22	BLNOD	Boot Load No Device Enable 0 = Event does not assert to interrupt status 1 = Event will assert in the interrupt status	R/W	1
23	BLOK	Boot Load OK Enable 0 = Event does not assert to interrupt status 1 = Event will assert in the interrupt status	R/W	1
24:25	Reserved	Reserved	R	00
26	MNACK	Master NACK Received Enable 0 = Event does not assert to interrupt status 1 = Event will assert in the interrupt status	R/W	1
27	MCOL	Master Collision Detect Enable 0 = Event does not assert to interrupt status 1 = Event will assert in the interrupt status	R/W	1
28	MTRTO	Master Transaction Timeout Enable 0 = Event does not assert to interrupt status 1 = Event will assert in the interrupt status	R/W	1
29	MBTTO	Master Byte Timeout Enable 0 = Event does not assert to interrupt status 1 = Event will assert in the interrupt status	R/W	1
30	MSCLTO	Master I2C_SCLK Low Timeout Enable 0 = Event does not assert to interrupt status 1 = Event will assert in the interrupt status	R/W	1
31	MARBTO	Master Arbitration Timeout Enable 0 = Event does not assert to interrupt status 1 = Event will assert in the interrupt status	R/W	1

13.2.26 I2C Time Period Divider Register

This register provides programmable extension of the reference clock period into longer periods used by the timeout and idle detect timers.

Register name: I2C_DIVIDER **Reset value:** 0x0063_03E7	**Register offset: 0x1D320**

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

Bits	Name	Description	Type	Reset Value
00:03	Reserved	Reserved	R	0x0
04:15	USDIV	Period Divider for Micro-Second Based Timers This field divides the reference clock down for use by the Idle Detect Timer, the Byte Timeout Timer, the I2C_SCLK Low Timeout Timer, and the Milli-Second Period Divider. Period(USDIV) = Period(P_CLK) * (USDIV + 1), where P_CLK is 10 ns. Reset period is 1 microsecond.	R/W	0x0063
16:19	Reserved	Reserved	R	0x0
20:31	MSDIV	Period Divider for Milli-Second Based Timers This field divides the USDIV period down further for use by the Arbitration Timeout Timer, the Transaction Timeout Timer, and the Boot/Diag Timeout Timer. Period(MSDIV) = Period(USDIV) * (MSDIV + 1). Reset period is 1 millisecond.	R/W	0x03E7

13.2.27 I²C Start Condition Setup/Hold Timing Register

This register programs the setup and hold timing for the Start condition when generated by the master control logic. The timer periods are relative to the reference clock. This register is shadowed during boot loading, and can be reprogrammed prior to a chain operation without affecting the bus timing for the current EEPROM.

Register name: I2C_START_SETUP_HOLD **Reset value:** 0x01D7_0191	**Register offset: 0x1D340**

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

Bits	**Name**	**Description**	**Type**	**Reset Value**
00:15	START_SETUP	Count for the START Condition Setup Period Defines the minimum setup time for the START condition; that is, both I2C_SCLK and I2C_SD seen high prior to I2C_SD pulled low. This is a master-only timing parameter. This value also doubles as the effective Stop Hold time. Period(START_SETUP) = (START_SETUP * Period(PCLK)), where PCLK is 10ns. Reset time is 4.71 microseconds.	R/W	0x01D7
16:31	START_HOLD	Count for the START Condition Hold Period Defines the minimum hold time for the START condition; that is, from I2C_SD seen low to I2C_SCLK pulled low. This is a master only timing parameter. Period(START_HOLD) = (START_HOLD * Period(PCLK)), where PCLK is 10 ns. Reset time is 4.01 microseconds.	R/W	0x0191

13.2.28 I2C Stop/Idle Timing Register

This register programs the setup timing for the Stop condition when generated by the master control logic, and the Idle Detect timer. The Start Setup time doubles as the Stop Hold. The timer period for the Stop setup is relative to the reference clock. The timer period for the Idle Detect is relative to the USDIV period. The STOP setup time is shadowed during boot loading, and can be reprogrammed prior to a chain operation without affecting the bus timing for the current EEPROM.

Register name: I2C_STOP_IDLE **Reset value:** 0x0191_0033	**Register offset: 0x1D344**

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

Bits	Name	Description	Type	Reset Value
00:15	STOP_SETUP	Count for STOP Condition Setup Period Defines the minimum setup time for the STOP condition; that is, both I2C_SCLK seen high and I2C_SD seen low prior to I2C_SD released high. This is a master-only timing parameter. Period(STOP_SETUP) = (STOP_SETUP * Period(P_CLK)), where P_CLK is 10ns. Reset time is 4.01 microseconds.	R/W	0x0191
16:31	IDLE_DET	Count for Idle Detect Period Used in two cases. First, defines the period after reset during which the I2C_SCLK signal must be seen high to call the bus idle. This period is needed to avoid interfering with an ongoing transaction after reset. Second, defines the period before a master transaction during which the I2C_SCLK and I2C_SD signals must both be seen high to call the bus idle. This period is a protection against external master devices not correctly idling the bus. Period(IDLE_DET) = (IDLE_DET * Period(USDIV)), where USDIV is the microsecond time defined in the "I2C Time Period Divider Register". A value of zero results in no idle detect period, meaning the bus will be sensed as idle immediately. Reset time is 51 microseconds.	R/W	0x0033

13.2.29 I2C_SD Setup and Hold Timing Register

This register programs the setup and hold times for the I2C_SD signal when output by either the master or slave interface. It is shadowed during boot loading, and can be reprogrammed prior to a chain operation without affecting the bus timing for the current EEPROM.

Register name: I2C_SDA_SETUP_HOLD **Reset value:** 0x007E_001F	**Register offset: 0x1D348**

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

Bits	**Name**	**Description**	**Type**	**Reset Value**
00:15	SDA_SETUP	Count for the I2C_SD Setup Period Defines the minimum setup time for the I2C_SD signal; that is, I2C_SD set to desired value prior to rising edge of I2C_SCLK. This applies to both slave and master interface. Note: This value should be set to the sum of the I2C_SD setup time and the maximum rise/fall time of the I2C_SD signal to ensure that the signal is valid on the output at the correct time. This time is different than the raw I2C_SD setup time in the *I^2C Specification*. Period(SDA_SETUP) = (SDA_SETUP * Period(P_CLK)), where P_CLK is 10ns. Reset time is 1260 nanoseconds.	R/W	0x007E
16:31	SDA_HOLD	Count for I2C_SD Hold Period Defines the minimum hold time for the I2C_SD signal; that is, I2C_SD valid past the falling edge of I2C_SCLK. This applies to both slave and master interface. Period(SDA_HOLD) = (SDA_HOLD * Period(P_CLK)), where P_CLK is 10 ns. Reset time is 310 nanoseconds.	R/W	0x001F

13.2.30 I2C_SCLK High and Low Timing Register

This register programs the nominal high and low periods of the I2C_SCLK signal when generated by the master interface. It is shadowed during boot loading, and can be reprogrammed prior to a chain operation without affecting the bus timing for the current EEPROM.

Register name: I2C_SCL_PERIOD **Reset value:** 0x01F4_01F4	**Register offset: 0x1D34C**

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

Bits	**Name**	**Description**	**Type**	**Reset Value**
00:15	SCL_HIGH	Count for I2C_SCLK High Period Defines the nominal high period of the clock, from rising edge to falling edge of I2C_SCLK. This is a master-only parameter. The observed period may be shorter if other devices pull the clock low. Period(SCL_HIGH) = (SCL_HIGH * Period(P_CLK)), where P_CLK is 10 ns. Reset time is 5.00 microseconds (100 kHz).	R/W	0x01F4
16:31	SCL_LOW	Count for I2C_SCLK Low Period Defines the nominal low period of the clock, from falling edge to rising edge of I2C_SCLK. This is a master-only parameter. The observed period may be longer if other devices pull the clock low. Period(SCL_LOW) = (SCL_LOW * Period(P_CLK)), where P_CLK is 10 ns. Reset time is 5.00 microseconds (100 kHz).	R/W	0x01F4

13.2.31 I2C_SCLK Minimum High and Low Timing Register

This register programs the minimum high and low periods of the I2C_SCLK signal when generated by the master interface. It is shadowed during boot loading, and can be reprogrammed prior to a chain operation without affecting the bus timing for the current EEPROM.

Register name: I2C_SCL_MIN_PERIOD **Reset value:** 0x0191_01D7	**Register offset: 0x1D350**

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

Bits	**Name**	**Description**	**Type**	**Reset Value**
00:15	SCL_MINH	Count for I2C_SCLK High Minimum Period Defines the minimum high period of the clock, from rising edge seen high to falling edge of I2C_SCLK. This is a master-only parameter. The observed period may be shorter if other devices pull the clock low. Period(SCL_MINH) = (SCL_MINH * Period(P_CLK)), where P_CLK is 10 ns. Reset time is 4.01 microseconds.	R/W	0x0191
16:31	SCL_MINL	Count for I2C_SCLK Low Minimum Period Defines the minimum low period of the clock, from falling edge seen low to rising edge of I2C_SCLK. This is a master-only parameter. The observed period may be longer if other devices pull the clock low. Period(SCL_MINL) = (SCL_MINL * Period(P_CLK)), where P_CLK is 10 ns. Reset time is 4.71 microseconds.	R/W	0x01D7

13.2.32 I2C_SCLK Low and Arbitration Timeout Register

This register programs the I2C_SCLK low timeout and the Arbitration timeout. The arbitration timer period is relative to the MSDIV period, and the I2C_SCLK low timeout period is relative to the USDIV period.

Register name: I2C_SCL_ARB_TIMEOUT **Reset value: 0x65BB_0033**	**Register offset: 0x1D354**

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

Bits	**Name**	**Description**	**Type**	**Reset Value**
00:15	SCL_TO	Count for I2C_SCLK Low Timeout Period Defines the maximum amount of time for a slave device holding the I2C_SCLK signal low. This timeout covers the period from I2C_SCLK falling edge to the next I2C_SCLK rising edge. Value 0x0 disables the timeout. Period(SCL_TO) = (SCL_TO * Period(USDIV)) where USDIV is the microsecond time defined in the "I2C Time Period Divider Register". The reset value of this timeout is 26,000 microseconds (26 milliseconds).	R/W	0x65BB
16:31	ARB_TO	Count for Arbitration Timeout Period Defines the maximum amount of time for the master interface to arbitrate for the bus before aborting the transaction. This timeout covers the period from master operation start (setting the START bit in the "I2C Master Control Register") until the ACK/NACK is received from the external slave for the slave device address. A value of 0 disables the timeout. Period(ARB_TO) = (ARB_TO * Period(MSDIV)) where MSDIV is the millisecond time defined in "I2C Time Period Divider Register". The reset value of this timeout is 51 milliseconds. However, this timeout is not active during the boot load sequence.	R/W	0x0033

13.2.33 I2C Byte/Transaction Timeout Register

This register programs the Transaction and Byte timeouts. The timer periods are relative to the USDIV period for the byte timeout, and relative to the MSDIV period for the transaction timeout.

Register name: I2C_BYTE_TRAN_TIMEOUT Reset value: 0x0000_0000	Register offset: 0x1D358

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

Bits	Name	Description	Type	Reset Value
00:15	BYTE_TO	Count for Byte Timeout Period Defines the maximum amount of time for a byte to be transferred on the I2C bus. This covers the period from Start condition to next ACK/NACK, between two successive ACK/NACK bits, or from ACK/NACK to Stop/Restart condition. A value of 0 disables the timeout. Period(BYTE_TO) = (BYTE_TO * Period(USDIV)) where USDIV is the microsecond time defined in "I2C Time Period Divider Register". This timeout is disabled on reset, and is not used during boot load.	R/W	0x0000
16:31	TRAN_TO	Count for Transaction Timeout Period Defines the maximum amount of time for a transaction on the I2C bus. This covers the period from Start to Stop. A value of 0 disables the timeout. Period(TRAN_TO) = (TRAN_TO * Period(MSDIV)) where MSDIV is the millisecond time defined in "I2C Time Period Divider Register". This timeout is disabled on reset, and is not used during boot load.	R/W	0x0000

13.2.34 I2C Boot and Diagnostic Timer

This register programs a timer that times out the boot load sequence, and can be used after boot load as a general purpose timer.

Register name: I2C_BOOT_DIAG_TIMER Reset value: 0x0000_0FA0	Register offset: 0x1D35C

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

Bits	Name	Description	Type	Reset Value
00	FREERUN	Free Running Timer 0 = Timer is not automatically restarted when expires 1 = When timer expires, timer is restarted with same COUNT	R/W	0
01:15	Reserved	Reserved	R	0x0000
16:31	COUNT	Count for Timer Period Defines period for timer. Initial reset value is used for overall boot load timeout. During normal operation, this timer can be used for any general purpose timing. A value of 0 disables the timeout. Period(DTIMER) = (COUNT * Period(MSDIV)), where MSDIV is the millisecond period define in "I2C Time Period Divider Register". Timer begins counting when this register is written. If this register is written while the counter is running, the timer is immediately restarted with the new COUNT, and the DTIMER/BLTO event is not generated. When the timer expires, either the BLTO or DTIMER event is generated, depending on whether the boot load sequence is active. If FREERUN is set to 1 when timer expires, then the timer is restarted immediately (the event is still generated), providing a periodic interrupt capability. The reset value for the boot load timeout is 4 seconds. If the boot load completes before the timer expires, the timer is set to zero (disabled).	R/W	0x0FA0

13.2.35 I²C Boot Load Diagnostic Progress Register

This register provides visibility of the register count and peripheral address during the boot load sequence.

Register name: I2C_BOOT_DIAG_PROGRESS Reset value: 0x0000_0000	Register offset: 0x1D3B8

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

Bits	Name	Description	Type	Reset Value
00:15	REGCNT	Register Count The number of registers remaining to load from the current EEPROM during the boot load sequence. This register is initialized to the count read from the first two bytes of the EEPROM after reset, or to the first two byte read after a boot chaining operation. The field counts down as each register address/data pair is read.	R	0x0000_0000
16:31	PADDR	Peripheral Address Value of current peripheral address used by the boot load sequence. This field is initialized to zero at reset, and increments as the boot load sequence progresses. If a chain operation is performed, this field is loaded with the new peripheral address from the "I²C Boot Control Register" (with the 3 LSBs set to zero).	R	0x0000_0000

This is a diagnostic register. Documentation is provided for reference purposes only. The function of this register is not guaranteed in future versions and usage thereof is not supported.

13.2.36 I2C Boot Load Diagnostic Configuration Register

This register provides visibility of boot sequence information.

Register name: I2C_BOOT_DIAG_CFG Reset value: Undefined	Register offset: 0x1D3BC

Bits	0	1	2	3	4	5	6	7
00:07	BOOTING	BDIS	PASIZE	PINC	Reserved			
08:15	Reserved							
16:23	Reserved							
24:31	Reserved	BOOT_ADDR						

Bits	Name	Description	Type	Reset Value
00	BOOTING	Booting 0 = Boot sequence not active 1 = Boot sequence in progress	R	0
01	BDIS	Boot Disabled 0 = Boot enabled 1 = Boot disabled	R	Undefined
02	PASIZE	Peripheral Address Size 0 = 1-byte peripheral address 1 = 2-byte peripheral address Note: This is the state of the I2C_BOOT_CNTRL.PSIZE field at boot load start or after a chain operation.	R	Undefined
03	PINC	Page Increment 0 = Page increment disabled 1 = Page increment enabled	R	0
04:24	Reserved	Reserved	R	0x000
25:31	BOOT_ADDR	Boot Device Address Current value of the boot device address in use by the boot load sequence. This value is incremented during the bootload if the page increment feature is enabled.	R	0x00

This is a diagnostic register. Documentation is provided for reference purposes only. The function of this register is not guaranteed in future versions and usage thereof is not supported.

A. Serial RapidIO Protocol Overview

The *RapidIO Physical Layer 1x/4x LP-Serial Specification* addresses the physical layer requirements for devices utilizing an electrical serial connection medium. This specification defines a full duplex serial physical layer interface (link) between devices using unidirectional differential signals in each direction. Further, it allows ganging of four serial links for applications requiring higher link performance. It also defines a protocol for link management and packet transport over a link.

RapidIO systems are comprised of end point processing elements and switch processing elements. The RapidIO interconnect architecture is partitioned into a layered hierarchy of specifications which includes the *Logical*, *Common Transport*, and *Physical* layers. The Logical layer specifications define the operations and associated transactions by which end point processing elements communicate with each other. The Common Transport layer defines how transactions are routed from one end point processing element to another through switch processing elements. The Physical Layer defines how adjacent processing elements electrically connect to each other. RapidIO packets are formed through the combination of bit fields defined in the Logical, Common Transport, and Physical Layer specifications. TheTsi574 fully manages the end to end link on each port.

A.1 Protocol

The RapidIO Physical Layer 1x/4x LP-Serial specification defines the protocol for packet delivery between serial RapidIO devices including packet and control symbol transmission, flow control, error management, and other device to device functions. A particular device may not implement all of the mode selectable features presented in the RapidIO specification.

The 1x/4x LP-Serial physical layer specification has the following properties:

- Embeds the transmission clock with data using an 8B/10B encoding scheme.
- Supports one serial differential pair, referred to as one lane, or four ganged serial differential pairs, referred to as four lanes, in each direction.
- Allows switching packets between RapidIO 1x/4x LP-Serial Ports and RapidIO Physical Layer 8/16 LP-LVDS ports without requiring packet manipulation.
- Employs similar retry and error recovery protocols as the RapidIO Physical Layer 8/16 LP-LVDS specification.
- Supports transmission rates of 1.25, 2.5, and 3.125 Gbaud (data rates of 1.0, 2.0, and 2.5 Gbit/s) per lane.

A.2 Packets

A RapidIO 1x/4x LP-Serial packet is formed by prefixing a 10-bit physical layer header to the combined RapidIO transport and logical layer bit fields followed by an appended 16-bit CRC field. The sum of all of the bit fields adds 20 bytes to the encapsulated data packet size. The maximum data field size is 256 bytes resulting in a maximum packet size of 276 bytes.

A.2.1 Control Symbols

Two classes of control symbols are defined (stype0 and stype1) and are used for packet acknowledgment, link utility functions, link maintenance, and packet delineation. A control symbol is a 24-bit entity (including a 5-bit CRC code). The control symbol is used for packet delineation by placement at the beginning of a packet. The control symbol may also be embedded within a packet for message passing and link status notification as well as sent when the link is idle.

Acknowledgment control symbols are used by processing elements to indicate packet transmission status. Utility control symbols are used to communicate buffer status and link recovery synchronization. Link maintenance control symbols are used by link partner devices to communicate physical layer status, synchronization requests, and device reset.

A.3 Physical Layer

The physical layer is broken into two sub-layers, the PCS and PMA Layers describes the Physical Coding Sub-layer (PCS) functionality as well as the Physical Media Attachment (PMA) functionality.

A.3.1 PCS Layer

The PCS layer functionality includes 8B/10B encoding scheme for embedding the clock with the data. It also manages the transmission rules for the 1x and 4x interfaces and defines the link initialization sequence for clock synchronization. The PCS function is also responsible for idle sequence generation, encoding for transmission and lane striping, and decoding, lane alignment and de-striping on reception.

The PCS layer also provides methods for determining the operational mode of the port as 4-lane or 1-lane operation, and means to detect link states. It provides for clock difference tolerance between the sender and receiver without requiring flow control.

A.3.2 PMA Layer

The PMA function is responsible for serializing 10-bit parallel code-groups to/from a serial bit stream on a lane-by-lane basis. Upon receiving data, the PMA function provides alignment of the received bit stream to 10-bit code-group boundaries, done independently on a lane-by-lane basis. It then provides a continuous stream of 10-bit code-groups to the PCS, one stream for each lane. The 10-bit code-groups are not observable by layers higher than the PCS.

A.3.3 Physical Protocol

The physical connection of a RapidIO link is managed by a series of control symbols transmitted on a transmit - response basis. These control symbols are made up of 10-bit encoded special characters and 3 byte control symbols. Encoded 8-bit characters are given encoding values and names that allow easy and unique detection of the character. Detailed explanation of the encoding values and names can be found in Chapter 4, *“PCS and PMA Layers” of “Part VI Physical Layer 1x/4x LP-Serial RapidIO Specification”*.

The Table 51 illustrates the Special Characters and their function as it applies to the serial protocol.

Table 51: Special Characters and Encoding

Code Group	Use	Number of Groups	Encoding	8-bit Value
/PD/	packet delimiter	1	/K28.3/	0x7C
/SC/	start of Control Symbol	1	/K28.0/	0x1C
/I/	1x Idle	/K/ or /R/ or /A/	see below	0xBC or 0xFD or 0xFB
/K/	1x Sync	1	/K28.5/	0xBC
/R/	1x Skip	1	/K29.7/	0xFD
/A/	1x Align	1	/K27.7/	0xFB
//I//	Idle Column	//K// or //R// or //A//	see below	0xBC or 0xFD or 0xFB
//K//	4x Idle	4	/K28.5/K28.5/K28.5/K28.5/	0xBC on each lane
//R//	4x Sync	4	/K29.7/K29.7/K29.7/K29.7/	0xFD on each lane
//A//	4x Skip	4	/K27.7/K27.7/K27.7/K27.7/	0xFB on each lane

Table 52 illustrates the control symbol construction in 8-bit values. Further detail on the usage of the control symbols may be found in *Part VI Physical Layer 1x/4x LP-Serial RapidIO Specification”*.

Table 52: Control Symbol Construction

3-bits	---	5-bits	5-bits	3-bits	---	3-bits	5-bits	--
stype0 [0-2]	definition	P-0	P-1	stype1 [0-2]		cmd	CRC	
000	pkt-accepted	pkt-ackID	buf_status					
001	pkt-rtry	pkt-ackID	buf_status					
010	pkt-not-accepted	pkt-ackID	cause (see below)					
011	reserved	---	---					
100	status	ackID_status	buf_status					
101	reserved	---	---					
110	link-response	ackID_status	port_status (see below)					
111	reserved	---	---					
		pkt-not-accepted cause [0-4]		definition				
			00000	reserved				
			00001	recvd unexpected ackID on pkt				
			00010	recvd a ctrl symbol w/ bad CRC				
			00011	non-maint pkt reception is stopped				
			00100	recvd pkt w/ bad CRC				
			00101	recvd invalid char or valid but illegal char				
			000110-11110	reserved				
			11111	general error				

Table 52: Control Symbol Construction

3-bits	---	5-bits	5-bits	3-bits	---	3-bits	5-bits	--
		link response port_status [0-4]		definition				
			00000	reserved				
			00001	reserved				
			00010	unrecoverable error unable to accept pkts				
			00011	reserved				
			00100	retry-stopped state				
			00101	error-stopped state				
			00110-01111	reserved				
			10000	OK				
			10001-11111	reserved				
				stype1				
				stype1[0-2]	definition	cmd[0-2]	cmd function	pkt delimiter
				000	start of pkt	000	reserved	yes
				001	stomp	000	reserved	yes
				010	end of pkt	000	reserved	yes
				011	restart from rtry	000	reserved	N/A
				100	link req	000-010	reserved	N/A
						011	reset device	N/A
						100	input status	N/A
						101-111	reserved	N/A
				101	multicast event	000	reserved	no
				110	reserved	000	reserved	no
				111	NOP	000	reserved	no

B. Clocking

This appendix describes device behavior outside the *RapidIO Interconnect Specification (Revision 1.3)* recommended operating line rates and clock frequencies.

The following topics are discussed:

B.1 Line Rate Support

The Tsi574 supports all of the *RapidIO Interconnect Specification (Revision 1.3)* specified line rates of 1.25, 2.50, and 3.125 Gbaud. The device also supports line rates that are outside of the RapidIO specification. The ability to support multiple line rates gives the Tsi574 flexibility in both application support and power consumption.

Table 53 shows the supported line rates for the Tsi574. The Serial Port Select pin, SP_IO_SPEED[1,0] must be set to the values shown in Table 53 to achieve the documented line rates.

Table 53: Tsi574 Supported Line Rates [a]

S_CLK_p/n (MHz)	Baud Rate (Gbaud)	SP_IO_SPEED[1,0] Bit Settings	Register Settings
153.60	1.2288 CPRI Line Rate	0,0	-
153.60	1.536 OBSAI Line Rate	0,1	-
153.60	2.4576 CPRI Line Rate	0,1	-
153.60	3.0720 CPRI Line Rate	1,0	-
156.25	1.2500 Standard RapidIO Line Rate	0,0	-
156.25	2.5000 Standard RapidIO Line Rate	0,1	-
156.25	3.1250 Standard RapidIO Line Rate	1,0	-

Table 53: Tsi574 Supported Line Rates *(Continued)*[a]

S_CLK_p/n (MHz)	Baud Rate (Gbaud)	SP_IO_SPEED[1,0] Bit Settings	Register Settings
125.00	1.2500 Standard RapidIO Line Rate	1,1	-
125.00	2.5000 Standard RapidIO Line Rate	1,0	-
125.00	3.1250 Standard RapidIO Line Rate	1,0	See "Register Requirements Using 125 MHz S_CLK for a 3.125 Gbps Link Rate" on page 476

a. This information assumes a +/- 100 ppm clock tolerance that must be obeyed between link partners.

All bit and register settings that are documented for operation with S_CLK = 156.25 .MHz also apply to the use of 153.6 MHz and 125 MHz. For more clocking information, see "Clocks" in the *Tsi574 User Manual*.

B.1.1 Register Requirements Using 125 MHz S_CLK for a 3.125 Gbps Link Rate

In order to use S_CLK at 125 MHz to create a 3.125 Gbps link baud rate, the default values in the SerDes PLL Control Register must be modified from a x20 multiplier to a x25 multiplier. On power-up, the default PLL multipliers of x20 causes the 125 MHz source to create a 2.5 Gbps link rate. Changing this link rate to 3.125 Gbps requires either intervention by the I^2C boot EEPROM during boot loading to reconfigure the SerDes, or the intervention of an external host to modify the SerDes registers through the use of maintenance transactions. However, modifying by EEPROM is the recommended method.

The SerDes PLL Control Registers are volatile. Applying HARD_RST_b or asserting PWRDN_x4 results in the SerDes PLL Control Register default value being re-applied.

B.1.1.1 Modification by EEPROM Boot Load

Modifying the EEPROM is the recommended method for using the S_CLK at 125 MHz to create a 3.125 Gbps link baud rate because the EEPROM boot load accesses the required configuration registers before the SerDes are released from reset. This can be performed by modifying the EEPROM loading script (for more information, see "EEPROM Scripts" in the *Tsi574 User Manual*).

Once the boot load is complete, the modified switch ports operate at 3.125 Gbps, while the remaining ports operate at 2.5 Gbps.

Using the Script

The example EEPROM loading script in the “EEPROM Scripts” appendix of the *Tsi574 User Manual* configures ports six and eight of the Tsi574. Other ports can be added to the script and configured by editing the text. The script is written assuming that no other contents are required in the EEPROM. Additional register configurations may be appended to the script as required, as well as the value written to location 0 of the EEPROM to indicate the number (hex) of registers the bootloader is required to initialize. For more information regarding configuring the contents of the EEPROM, see “I2C Interface” in the *Tsi574 User Manual*.

B.1.1.2 Modification by Maintenance Transaction

Modification by maintenance transactions must occur after the link to the host processor tasked with changing the port speeds has initialized. The process involves performing the sequence of operations listed in “Example Maintenance Transaction Sequence” on page 477.

The possibility of link instability exists should the process not be followed in the stated sequence

Example Maintenance Transaction Sequence

The following procedure configures port two. After these steps are complete, port two can train with its link partner at a baud rate of 3.125Gbps.

1. Reset the MAC by asserting SOFT_RST_x4 and leave the IO_SPEED set to 3.125
   - Write offset 0x132C8 with 0x7FFF0012
2. Set the BYPASS_INIT bit to enable control of the following: MPLL_CK_OFF, SERDES_RESET, MPLL_PWRON, TX_EN, RX_PLL_PWRON, RX_EN
   - Write offset 0x132C0 with 0xCA060084
3. Clear the RX_EN bit in the SMAC_x SerDes Configuration Register Channel 0 - 3
   - Write offset 0x132B0 with 0x203CA513
   - Write offset 0x132B4 with 0x203CA513
   - Write offset 0x132B8 with 0x203CA513
   - Write offset 0x132BC with 0x203CA513
4. Clear the RX_PLL_PWRON bit in the SMAC_x SerDes Configuration Register Channel 0 - 3
   - Write offset 0x132B0 with 0x203C2513
   - Write offset 0x132B4 with 0x203C2513
   - Write offset 0x132B8 with 0x203C2513
   - Write offset 0x132BC with 0x203C2513
5. Clear the TX_EN field in the SMACx_CFG_CH0
   - Write offset 0x132B0 with 0x200C2513
   - Write offset 0x132B4 with 0x200C2513

— Write offset 0x132B8 with 0x200C2513

— Write offset 0x132BC with 0x200C2513

6. Clear the MPLL_PWRON bit in the SMACx_CFG_GLOBAL register
   — Write offset 0x132c0 with 0xCA060004
   — Ensure that BYPASS_INIT remains asserted
7. Set the MPLL_CK_OFF bit in the SMACx_CFG_GLOBAL register
   — Write offset 0x132c0 with 0xCA060044
8. Change the multipliers by:
   — Write offset 0x132C4 with 0x002C0545
   — Write offset 0x132C0 with 0xCA060045
9. Clear the MPLL_CK_OFF bit in the SMACx_CFG_GBL register
   — Write offset 0x132C0 with 0xCA060005
10. Toggle the SERDES_RSTN bit in the SMACx_CFG_GBL register
    — Write offset 0x132C0 with 0x4A060005
    — Write offset 0x132c0 with 0xCA060005
11. Set the MPLL_PWRON bit in the SMACx_CFG_GLOBAL register
    — Write offset 0x132C0 with 0xCA060085
    — Ensure that BYPASS_INIT remains asserted
12. Set TX_EN[2:0] to 0b011 in the SMACx_CFG_CH0-3 register
    — Write offset 0x132B0 with 0x203C2513
    — Write offset 0x132B4 with 0x203C2513
    — Write offset 0x132B8 with 0x203C2513
    — Write offset 0x132BC with 0x203C2513
13. Set the RX_PLL_PWRON bit in the SMACx_CFG_CH0-3 register
    — Write offset 0x132B0 with 0x203CA513
    — Write offset 0x132B4 with 0x203CA513
    — Write offset 0x132B8 with 0x203CA513
    — Write offset 0x132BC with 0x203CA513
14. Set the RX_EN bit in the SMACx_CFG_CH0-3 register
    — Write offset 0x132B0 with 0x203CE513
    — Write offset 0x132B4 with 0x203CE513
    — Write offset 0x132B8 with 0x203CE513
    — Write offset 0x132BC with 0x203CE513

15. Release the MAC from reset
    - Write offset 0x132c8 with 0x7FFF0002

B.2 P_CLK Programming

The Tsi574 recommends a P_CLK operating frequency of 100 MHz. However, the device also supports P_CLK frequencies less than the recommended 100 MHz. The ability to support other P_CLK frequencies gives the Tsi574 flexibility in both application support and design.

The minimum frequency supported by the P_CLK input is 25 MHz. Operation above 100 MHz or below 25 MHz is not tested or guaranteed.

The following sections describe the effects on the Tsi574 when the input frequency of the P_CLK source is decreased from the recommended 100 MHz operating frequency.

B.2.1 RapidIO Specifications Directly Affected by Changes in the P_CLK Frequency

The following sections describe how changing the P_CLK frequency to below the recommended 100 MHz operation affect the counters and state machines in the Tsi574 that are defined in the *RapidIO Interconnect Specification (Revision 1.3)*.

B.2.1.1 Port Link Time-out CSR

RapidIO Part 6: 1x/4x LP-Serial Physical Layer Specification Revision 1.3: Section 6.6.2.2 Port Link Time-out CSR (Block Offset 0x20)

The *RapidIO Interconnect Specification (Revision 1.3)* defines the Port Link Time-out CSR as follows:

> The port-link time-out control register contains the time-out timer value for all ports on a device. This time-out is for link events, such as sending a packet to receiving the corresponding acknowledge and sending a link-request to receiving the corresponding link-response. The reset value is the maximum time-out interval, and represents between three and six seconds.

IDT Implementation

The Tsi574 supports this timer in the RapidIO Switch Port Link Time Out Control CSR. Effects of changing the P_CLK frequency are shown in the following formula:

- Time-out = 32/F x TVAL
  - F is P_CLK frequency in MHz
  - TVAL is the 24-bit counter setting
    - Maximum TVAL decimal value of 16,777,215 (0xFFFFFF)

Effects of changing the P_CLK frequency and TVAL setting can be seen in Table 54.

Table 54: Timer Values with P_CLK and TVAL Variations

P_CLK Setting	TVAL Setting	Equation	Timer Value
25 MHz	2,343,750 (0x23C346)	32/25 x 2,343,750	3 seconds
25 MHz	4,687,500 (0x47868C)	32/25 x 4,687,500	6 seconds
50 MHz	4,687,500 (0x47868C)	32/50 x 4,687,500	3 seconds
50 MHz	9,375,000 (0x8F0D18)	32/50 x 9,375,000	6 seconds
50 MHz	16,777,215 (0xFFFFFF)	32/50 x 16,777,215	10.4 seconds
100 MHz	9,375,000 (0x8F0D18)	32/100 x 9,375,000	3 seconds
100 MHz	16,777,215 (0xFFFFFF)	32/100 x 16,777,215	5.4 seconds

B.2.1.2 *RapidIO Part 6: 1x/4x LP-Serial Physical Layer Specification Revision 1.3: Section 4.7.3.2 State Machine Variables and Functions*

SILENCE_TIMER_DONE

The *RapidIO Interconnect Specification (Revision 1.3)* defines the SILENCE_TIMER_DONE as follows:

> Asserted when the SILENCE_TIMER_EN has been continuously asserted for 120 +/- 40 µs and the state machine is in the SILENT state. The assertion of SILENCE_TIMER_DONE causes SILENCE_TIMER_EN to be de-asserted. When the state machine is not in the SILENT state, SILENCE_TIMER_DONE is de-asserted

IDT Implementation

The Tsi574's silence timer does not have user programmable registers. The silence timer is sourced from the P_CLK and any changes to P_CLK are directly reflected in the timer timeout period.

DISCOVERY_TIMER_DONE

The *RapidIO Interconnect Specification (Revision 1.3)* defines the DISCOVERY_TIMER_DONE as follows:

> Asserted when DISCOVERY_TIMER_EN has been continuously asserted for 12 +/- 4msec and the state machine is in the DISCOVERY state. The assertion of DISCOVERY_TIMER_DONE causes DISCOVERY_TIMER_EN to be de-asserted. When the state machine is not in the DISCOVERY state, DISCOVERY_TIMER_DONE is de-asserted.

IDT Implementation

The Tsi574's discovery timer is programmed in the RapidIO Port x Discovery Timer. The DISCOVERY_TIMER field is used by serial ports configured to operate in 4x mode. The DISCOVERY_TIMER allows time for the link partner to enter its discovery state, and if the link partner supports 4x mode, for all four lanes to be aligned.

The DISCOVERY_TIMER field is a 4-bit field whose value is used as a pre-scaler for a 17-bit counter clocked by P_CLK.

The DISCOVERY_TIMER has a default value of 9 decimal, but can be programmed to various values. The results of changing the DISCOVERY_TIMER value and P_CLK are shown in Table 55.

Table 55: Timer Values with DISCOVERY_TIMER and P_CLK Variations

P_CLK Setting	DISCOVERY_TIMER Setting	Equation	Timer Value
100 MHz	9 decimal	9 * 0x1FFFF * 1/ P_CLK	11.79 mS
100 MHz	9 decimal	9 * 131071 * 1/ P_CLK	11.79 mS
25 MHz	1 decimal	1 * 131071 * 1/25 MHz	5.24 mS
25 MHz	2 decimal	2 * 131071 * 1/25 MHz	10.48 mS
25 MHz	15 decimal	15 * 131071 * 1/25 MHz	78.6 mS
50 MHz	1 decimal	1 * 131071 * 1/ 50 MHz	2.62 mS
50 MHz	5 decimal	5 * 131071 * 1/ 50 MHz	13.1 mS
50 MHz	15 decimal	15 * 131071 *1/ 50 MHz	19.7 mS
100 MHz	1 decimal	1 * 131071 * 1/ 100 MHz	1.31 mS
100 MHz	9 decimal	9 * 131071 * 1/ 100 MHz	11.79 mS
100 MHz	15 decimal	15 * 131071 *1/ 100 MHz	19.7 mS

B.2.2 IDT Specific Timers

The following sections describe how changing the P_CLK frequency to below the recommended 100 MHz operation affect the IDT-specific counters and state machines in the Tsi574.

B.2.2.1 Dead Link Timer

The Dead Link Timer period is controlled by the DLT_THRESH field in the SRIO MAC x Digital Loopback and Clock Selection Register.

Each time a silence is detected on a link, the counter is reloaded from this register and starts to count down. When the count reaches 0, the link is declared dead, which means that all packets are flushed from the transmit queue and no new packets are admitted to the queue until the link comes up.

The duration of the dead link timer is computed by the following formula:

- 2^^13 * DLT_THRESH * P_CLK period
  - P_CLK is 100 MHz (which gives a P_CLK period of 10nS)
  - Default value of DLT_THRESH is 0x7FFF (which corresponds to 32767)
- Using these parameters, the populated formula is 8192*32767*10e-9 = 2.68 seconds

When enabled, this timer is used to determine when a link is powered up and enabled, but dead (that is, there is no link partner responding). When a link is declared dead, the transmitting port on the Tsi574 removes all packets from its transmit queue and ensure that all new packets sent to port are dropped rather than placed in the transmit queue.

The DLT_THRESH is a 15-bit counter with a maximum value of 32767. Table 56 shows equations using different values for DLT_THRESH and P_CLK.

Table 56: Timer Values with P_CLK and DLT_THRESH Variations

P_CLK Setting	Equation	Timer Value
25 MHz	8192 * 1 * 1/25 MHz	327 uS
	8192 * 32767 * 1/25 MHz	10.74 seconds
50 MHz	8192 * 1 * 1/50 MHz	163.8 uS
	8192 * 32767 * 1/50 MHz	5.37 seconds
100 MHz	8192 * 1 * 1/100 MHz	81.9 uS
	8192 * 32767 * 1/100 MHz	2.68 seconds

B.2.3 I²C interface and Timers

The I²C interface clock is derived from the P_CLK. Decreasing the frequency of P_CLK causes a proportional decrease in the I²C serial clock and affects the I²C timers. The timer values can be re-programmed during boot loading but the changes does not take effect until after the boot load has completed. As a result, a decrease from 100 MHz to 50 MHz of P_CLK causes a doubling of the boot load time of the EEPROM. Once boot loading has completed, the new values take effect and the I²C interface can operate at the optimum rate of the attached devices.

B.2.3.1 I²C Time Period Divider Register

The I2C Time Period Divider Register provides programmable extension of the reference clock period into longer periods used by the timeout and idle detect timers.

USDIV Period Divider for Micro-Second Based Timers

The USDIV field divides the reference clock down for use by the Idle Detect Timer, the Byte Timeout Timer, the I2C_SCLK Low Timeout Timer, and the Milli-Second Period Divider.

- Period(USDIV) = Period(P_CLK) * (USDIV + 1)
- P_CLK is 10 ns
- Tsi574 reset value is 0x0063

MSDIV Period Divider for Milli-Second Based Timers

The MSDIV field divides the USDIV period down further for use by the Arbitration Timeout Timer, the Transaction Timeout Timer, and the Boot/Diagnostic Timeout Timer.

- Period (MSDIV) = Period(USDIV) * (MSDIV + 1)
- Tsi574 reset value is 0x03E7

B.2.3.2 I2C Start Condition Setup/Hold Timing Register

The I2C Start Condition Setup/Hold Timing Register programs the setup and hold timing for the start condition when generated by the master control logic. The timer periods are relative to the reference clock.

This register is shadowed during boot loading, and can be reprogrammed prior to a chain operation without affecting the bus timing for the current EEPROM.

START_SETUP Count for the START Condition Setup Period

The START_SETUP field defines the minimum setup time for the START condition; that is, both I2C_SCLK and I2C_SD seen high prior to I2C_SD pulled low. This is a master-only timing parameter.

This value also doubles as the effective Stop Hold time.

- Period (START_SETUP) = (START_SETUP * Period(PCLK))
  - PCLK is 10ns
  - Reset time is 4.71 microseconds.
  - Tsi574 reset value is 0x01D7

START_HOLD Count for the START Condition Hold Period

The START_HOLD field defines the minimum hold time for the START condition; that is, from I2C_SD seen low to I2C_SCLK pulled low. This is a master only timing parameter.

- Period (START_HOLD) = (START_HOLD * Period(P_CLK))
- P_CLK is 10 ns
- Reset time is 4.01 microseconds
- Tsi574 reset value is 0x0191

B.2.3.3 I2C Stop/Idle Timing Register

The I2C Stop/Idle Timing Register programs the setup timing for the Stop condition when generated by the master control logic and the Idle Detect timer.

The START_SETUP time doubles as the Stop Hold.

The Stop/Idle register is broken down as follows:

- The timer period for the STOP_SETUP is relative to the reference clock
- The timer period for the Idle Detect is relative to the USDIV period
- The STOP_SETUP time is shadowed during boot loading, and can be reprogrammed prior to a chain operation without affecting the bus timing for the current EEPROM.

STOP_SETUP Count for STOP Condition Setup Period

The STOP_SETUP field defines the minimum setup time for the STOP condition (that is, both I2C_SCLK seen high and I2C_SD seen low prior to I2C_SD released high). This is a master-only timing parameter.

- Period(STOP_SETUP) = (STOP_SETUP * Period(P_CLK))
  - P_CLK is 10ns
  - Reset time is 4.01 microseconds
  - Tsi574 reset value is 0x0191

IDLE_DET Count for Idle Detect Period

The IDLE_DET field is used in two cases. First, it defines the period after reset during which the I2C_SCLK signal must be seen high in order to call the bus idle. This period is needed to avoid interfering with an ongoing transaction after reset. Second, it defines the period before a master transaction during which the I2C_SCLK and I2C_SD signals must both be seen high in order to call the bus idle.

This period is a protection against external master devices not correctly idling the bus.

- Period(IDLE_DET) = (IDLE_DET * Period(USDIV)), where USDIV is the microsecond time defined in the I2C Time Period Divider Register

A value of zero results in no idle detect period, meaning the bus will be sensed as idle immediately.

  - Reset time is 51 microseconds
  - Tsi574 reset value is 0x0033

B.2.3.4 I2C_SD Setup and Hold Timing Register

The I2C_SD Setup and Hold Timing Register programs the setup and hold times for the I2C_SD signal when output by either the master or slave interface. It is shadowed during boot loading, and can be reprogrammed prior to a chain operation without affecting the bus timing for the current EEPROM.

SDA_SETUP Count for the I2C_SD Setup Period

The SDA_SETUP field defines the minimum setup time for the I2C_SD signal; that is, I2C_SD is set to a desired value prior to rising edge of I2C_SCLK. This applies to both slave and master interface.

This value should be set to the sum of the I2C_SD setup time and the maximum rise/fall time of the I2C_SD signal in order to ensure that the signal is valid on the output at the correct time. This time is different than the raw I2C_SD setup time in the *I^2C Specification*.

- Period(SDA_SETUP) = (SDA_SETUP * Period(P_CLK)), where P_CLK is 10ns.
  - Reset time is 1260 nanoseconds
  - Tsi574 reset value is 0x007E

SDA_HOLD Count for I2C_SD Hold Period

The SDA_HOLD field defines the minimum hold time for the I2C_SD signal; that is, I2C_SD valid past the falling edge of I2C_SCLK. This applies to both slave and master interface.

- Period(SDA_HOLD) = (SDA_HOLD * Period(P_CLK)), where P_CLK is 10 ns.
  - Reset time is 310 nanoseconds
  - Tsi574 reset value is 0x001F

B.2.3.5 I2C_SCLK High and Low Timing Register

The I2C_SCLK High and Low Timing Register programs the nominal high and low periods of the I2C_SCLK signal when generated by the master interface.

It is shadowed during boot loading, and can be reprogrammed prior to a chain operation without affecting the bus timing for the current EEPROM.

SCL_HIGH Count for I2C_SCLK High Period

The SCL_HIGH field defines the nominal high period of the clock, from rising edge to falling edge of I2C_SCLK. This is a master-only parameter.

The actual observed period may be shorter if other devices pull the clock low.

- Period(SCL_HIGH) = (SCL_HIGH * Period(P_CLK))
  - P_CLK is 10 ns
  - Reset time is 5.00 microseconds (100 kHz)
  - Tsi574 reset value is 0x01F4

SCL_LOW Count for I2C_SCLK Low Period

The SCL_LOW field defines the nominal low period of the clock, from falling edge to rising edge of I2C_SCLK. This is a master-only parameter.

The actual observed period may be longer if other devices pull the clock low.

- Period(SCL_LOW) = (SCL_LOW * Period(P_CLK))
  - P_CLK is 10 ns
  - Reset time is 5.00 microseconds (100 kHz)
  - Tsi574 reset value is 0x01F4

B.2.3.6 I2C_SCLK Minimum High and Low Timing Register

The I2C_SCLK Minimum High and Low Timing Register programs the minimum high and low periods of the I2C_SCLK signal when generated by the master interface. It is shadowed during boot loading, and can be reprogrammed prior to a chain operation without affecting the bus timing for the current EEPROM.

SCL_MINH Count for I2C_SCLK High Minimum Period

The SCL_MINH field defines the minimum high period of the clock, from rising edge seen high to falling edge of I2C_SCLK. This is a master-only parameter.

The actual observed period may be shorter if other devices pull the clock low.

- Period(SCL_MINH) = (SCL_MINH * Period(P_CLK))
  - P_CLK is 10 ns
  - Reset time is 4.01 microseconds
  - Tsi574 reset value is 0x0191

SCL_MINL Count for I2C_SCLK Low Minimum Period

The SCL_MINL defines the minimum low period of the clock, from falling edge seen low to rising edge of I2C_SCLK. This is a master-only parameter.

The actual observed period may be longer if other devices pull the clock low.

- Period(SCL_MINL) = (SCL_MINL * Period(P_CLK))
  - P_CLK is 10 ns
  - Reset time is 4.71 microseconds
  - Tsi574 reset value is 0x01D7

B.2.3.7 I2C_SCLK Low and Arbitration Timeout Register

The I2C_SCLK Low and Arbitration Timeout Register programs the I2C_SCLK low timeout and the Arbitration timeout. The arbitration timer period is relative to the MSDIV period, and the I2C_SCLK low timeout period is relative to the USDIV period.

SCL_TO Count for I2C_SCLK Low Timeout Period

The SCL_TO field defines the maximum amount of time for a slave device holding the I2C_SCLK signal low. This timeout covers the period from I2C_SCLK falling edge to the next I2C_SCLK rising edge. A value of 0 disables the timeout.

- Period(SCL_TO) = (SCL_TO * Period(USDIV))
  - USDIV is the microsecond time defined in the I2C Time Period Divider Register.
  - The reset value of this timeout is 26 milliseconds
  - Tsi574 reset value is 0x65BB

ARB_TO Count for Arbitration Timeout Period

The ARB_TO field defines the maximum amount of time for the master interface to arbitrate for the bus before aborting the transaction. This timeout covers the period from master operation start (see setting the START bit in the I2C Master Control Register) until the ACK/NACK is received from the external slave for the slave device address. A value of 0 disables the timeout.

- Period(ARB_TO) = (ARB_TO * Period(MSDIV))
  - MSDIV is the millisecond time defined in I2C Time Period Divider Register.
  - The reset value of this timeout is 51 milliseconds
    - This timeout is not active during the boot load sequence.
  - Tsi574 reset value is 0x0033

B.2.3.8 I2C Byte/Transaction Timeout Register

The I2C Byte/Transaction Timeout Register programs the Transaction and Byte time-outs. The timer periods are relative to the USDIV period for the byte timeout, and relative to the MSDIV period for the transaction timeout.

BYTE_TO Count for Byte Timeout Period

The BYTE_TO field defines the maximum amount of time for a byte to be transferred on the I^2C bus. This covers the period from Start condition to next ACK/NACK, between two successive ACK/NACK bits, or from ACK/NACK to Stop/Restart condition. A value of 0 disables the timeout.

- Period(BYTE_TO) = (BYTE_TO * Period(USDIV))
  - USDIV is the microsecond time defined in I2C Time Period Divider Register.
  - This timeout is disabled on reset, and is not used during boot load.
  - Tsi574 reset value is 0x0000

TRAN_TO Count for Transaction Timeout Period

The TRAN_TO field defines the maximum amount of time for a transaction on the I2C bus. This covers the period from Start to Stop. A value of 0 disables the timeout.

- Period(TRAN_TO) = (TRAN_TO * Period(MSDIV))
  - MSDIV is the millisecond time defined in I2C Time Period Divider Register.
  - This timeout is disabled on reset, and is not used during boot load
  - Tsi574 reset value is 0x0000

B.2.3.9 I2C Boot and Diagnostic Timer

The I2C Boot and Diagnostic Timer programs a timer used to timeout the boot load sequence, and can be used after boot load as a general purpose timer.

COUNT Count for Timer Period

The COUNT field defines the period for the timer. The initial reset value is used for overall boot load timeout. A value of 0 disables the timeout.

During normal operation, this timer can be used for any general purpose timing.

The timer begins counting when this register is written. If this register is written while the counter is running, the timer is immediately restarted with the new COUNT, and the DTIMER/BLTO event is not generated.

When the timer expires, either the BLTO or DTIMER event is generated, depending on whether the boot load sequence is active. If FREERUN is set to 1 when timer expires, then the timer is restarted immediately (the event is still generated), providing a periodic interrupt capability.

- Period(DTIMER) = (COUNT * Period(MSDIV))
  - MSDIV is the millisecond period define in I2C Time PeriodDivider Register.
  - The reset value for the boot load timeout is four seconds. If the boot load completes before the timer expires, the timer is set to zero (disabled).
  - Tsi574 reset value is 0x0FA0

B.2.4 Other Performance Factors

This section describes any other factors that may impact the performance of the Tsi574 if P-CLK is programmed to operate lower than the recommended 100 MHz frequency.

B.2.4.1 Internal Register Bus Operation

The internal register bus, where all the internal registers reside, is a synchronous bus clocked by the P_CLK source. A decrease in the P_CLK frequency causes a proportional increase in register access time during RapidIO maintenance transactions, JTAG registers accesses, and I^2C register accesses.

RapidIO Maintenance Transaction

Maintenance transactions use the internal register bus to read and write registers in the Tsi574. If the P_CLK frequency is decreased, it may be necessary to review the end point's response latency timer value to ensure that it does not expire before the response is returned.

Changing the frequency of the P_CLK does not affect the operation or performance of the RapidIO portion of the switch, in particular its ability to route or multicast packets between ports.

JTAG Register Interface

Changing the P_CLK frequency affects accesses to the internal registers through the JTAG register interface because the interface uses the internal register bus. However, the decreased performance will not be noticeable.

Boundary scan operations are not affected by a change in the P_CLK frequency because these transactions use the JTAG TCK clock signal and do not access the internal register bus.

C. PRBS Scripts

The following sections show the PRBS scripts used in "Using PRBS Scripts for the Transmitters and Receivers". All of the PRBS scripts affect all of the ports, therefore editing the files to comment out the respective transmitting and receiving ports is required.

Topics discussed include the following:

- "Tsi574_start_prbs_all.txt Script"
- "Tsi574_framer_disable.txt Script"
- "Tsi574_sync_prbs_all.txt Script"
- "Tsi574_read_prbs_all.txt Script"

C.1 Tsi574_start_prbs_all.txt Script

This JTAG script is used to turn on the PRBS pattern generator for each lane to be tested. The SerDes in a port are offset by 0x40 from the lane 0 register of the port.

The SerDes Lane 0 Pattern Generator Control Register is located at offset 0x1e020. Therefore lane 1 is located at offset 0x1e060, lane 2 is located at 0x1e0a0 and lane 3 is located at 0x1e0e0.

```
//i 0

//Port 0

w 1e020 00000002 //Start 2^7 Pattern Generator

w 1e060 00000002

w 1e0a0 00000002

w 1e0e0 00000002

//Port 2

w 1e220 00000002 //Start 2^7 Pattern Generator

w 1e260 00000002

w 1e2a0 00000002

w 1e2e0 00000002

//Port 4

w 1e420 00000002 //Start 2^7 Pattern Generator

w 1e460 00000002
```

```
w 1e4a0 00000002
w 1e4e0 00000002
//Port 6
w 1e620 00000002 //Start 2^7 Pattern Generator
w 1e660 00000002
w 1e6a0 00000002
w 1e6e0 00000002
//Port 8
w 1e820 00000002 //Start 2^7 Pattern Generator
w 1e860 00000002
w 1e8a0 00000002
w 1e8e0 00000002
//Port a
w 1ea20 00000002 //Start 2^7 Pattern Generator
w 1ea60 00000002
w 1eaa0 00000002
w 1eae0 00000002
//Port c
w 1ec20 00000002 //Start 2^7 Pattern Generator
w 1ec60 00000002
w 1eca0 00000002
w 1ece0 00000002
//Port e
w 1ee20 00000002 //Start 2^7 Pattern Generator
w 1ee60 00000002
w 1eea0 00000002
w 1eee0 00000002
```

C.2 Tsi574_framer_disable.txt Script

This script turns off the word alignment framer because PRBS patterns are not word aligned. This script is only for use with PRBS patterns.

In this script, the HALF_RATE bit is set corresponding to port operation at 1.25 Gbps. PRBS testing at 2.5 or 3.125 Gbps requires the HALF_RATE bit to be cleared. The data to be written therefore becomes 0x203CE511.

```
//i 0

//Port 0

w 130b0 A03CE511 //Clear RX_ALIGN_EN

w 130b4 A03CE511

w 130b8 A03CE511

w 130bc A03CE511

//Port 2

w 132b0 A03CE511 //Clear RX_ALIGN_EN

w 132b4 A03CE511

w 132b8 A03CE511

w 132bc A03CE511

//Port 4

w 134b0 A03CE511 //Clear RX_ALIGN_EN

w 134b4 A03CE511

w 134b8 A03CE511

w 134bc A03CE511

//Port 6

w 136b0 A03CE511 //Clear RX_ALIGN_EN

w 136b4 A03CE511

w 136b8 A03CE511

w 136bc A03CE511

//Port 8

w 138b0 A03CE511 //Clear RX_ALIGN_EN

w 138b4 A03CE511
```

```
w 138b8 A03CE511

w 138bc A03CE511

//Port a

w 13ab0 A03CE511 //Clear RX_ALIGN_EN

w 13ab4 A03CE511

w 13ab8 A03CE511

w 13abc A03CE511

//Port c

w 13cb0 A03CE511 //Clear RX_ALIGN_EN

w 13cb4 A03CE511

w 13cb8 A03CE511

w 13cbc A03CE511

//Port e

w 13eb0 A03CE511 //Clear RX_ALIGN_EN

w 13eb4 A03CE511

w 13eb8 A03CE511

w 13ebc A03CE511
```

C.3 Tsi574_sync_prbs_all.txt Script

This JTAG script is used to turn on the PRBS pattern matcher for each lane to be tested. The SerDes in a port are offset by 0x40 from the lane 0 register of the port. The SerDes Lane 0 Pattern Generator Control Register is located at offset 0x1e020. Therefore lane 1 is located at offset 0x1e060, lane 2 is located at 0x1e0a0 and lane 3 is located at 0x1e0e0.

```
//i 0

//Port 0

w 1e030 0000000a //turn on Sync pattern matcher

w 1e070 0000000a

w 1e0b0 0000000a

w 1e0f0 0000000a

w 1e030 00000002 //turn off sync pattern matcher

w 1e070 00000002
```

```
w 1e0b0 00000002
w 1e0f0 00000002
//Port 2
w 1e230 0000000a //Sync pattern matcher
w 1e270 0000000a
w 1e2b0 0000000a
w 1e2f0 0000000a
w 1e230 00000002
w 1e270 00000002
w 1e2b0 00000002
w 1e2f0 00000002
//Port 4
w 1e430 0000000a //Sync pattern matcher
w 1e470 0000000a
w 1e4b0 0000000a
w 1e4f0 0000000a
w 1e430 00000002
w 1e470 00000002
w 1e4b0 00000002
w 1e4f0 00000002
//Port 6
w 1e630 0000000a //Sync pattern matcher
w 1e670 0000000a
w 1e6b0 0000000a
w 1e6f0 0000000a
w 1e630 00000002
w 1e670 00000002
w 1e6b0 00000002
w 1e6f0 00000002
```

```
//Port 8
w 1e830 0000000a //Sync pattern matcher
w 1e870 0000000a
w 1e8b0 0000000a
w 1e8f0 0000000a
w 1e830 00000002
w 1e870 00000002
w 1e8b0 00000002
w 1e8f0 00000002
//Port a
w 1ea30 0000000a //Sync pattern matcher
w 1ea70 0000000a
w 1eab0 0000000a
w 1eaf0 0000000a
w 1ea30 00000002
w 1ea70 00000002
w 1eab0 00000002
w 1eaf0 00000002
//Port c
w 1ec30 0000000a //Sync pattern matcher
w 1ec70 0000000a
w 1ecb0 0000000a
w 1ecf0 0000000a
w 1ec30 00000002
w 1ec70 00000002
w 1ecb0 00000002
w 1ecf0 00000002
//Port e
w 1ee30 0000000a //Sync pattern matcher
```

```
w 1ee70 0000000a
w 1eeb0 0000000a
w 1eef0 0000000a
w 1ee30 00000002
w 1ee70 00000002
w 1eeb0 00000002
w 1eef0 00000002
```

C.4 Tsi574_read_prbs_all.txt Script

This script is used to read the PRBS values. Note that the PRBS error counter and overflow bit fields must be read twice to determine the correct value. The result of the first read is invalid and should be discarded. The result of the second read is correct and should be kept.

```
//Port0
r 1e030
r 1e030
r 1e070
r 1e070
r 1e0b0
r 1e0b0
r 1e0f0
r 1e0f0
//Port2
r 1e230
r 1e230
r 1e270
r 1e270
r 1e2b0
r 1e2b0
r 1e2f0
r 1e2f0
```

```
//Port4
r 1e430
r 1e430
r 1e470
r 1e470
r 1e4b0
r 1e4b0
r 1e4f0
r 1e4f0
//Port6
r 1e630
r 1e630
r 1e670
r 1e670
r 1e6b0
r 1e6b0
r 1e6f0
r 1e6f0
//Port8
r 1e830
r 1e830
r 1e870
r 1e870
r 1e8b0
r 1e8b0
r 1e8f0
r 1e8f0
//Porta
r 1ea30
```

```
r 1ea30
r 1ea70
r 1ea70
r 1eab0
r 1eab0
r 1eaf0
r 1eaf0
//Portc
r 1ec30
r 1ec30
r 1ec70
r 1ec70
r 1ecb0
r 1ecb0
r 1ecf0
r 1ecf0
//Porte
r 1ee30
r 1ee30
r 1ee70
r 1ee70
r 1eeb0
r 1eeb0
r 1eef0
r 1eef0
```

D. EEPROM Scripts

The following section shows the EEPROM script used in "Modification by EEPROM Boot Load".

D.1 Script

```
    ew 0 0047FFFF
    ew 4 FFFFFFFF
/1
    ew 8 138c8
    ew c 7FFF0012
/2
    ew 10 138c0
    ew 14 CA060084
/3
    ew 18 138B0
    ew 1c 203CA513
/4
    ew 20 138B4
    ew 24 203CA513
/5
    ew 28 138B8
    ew 2c 203CA513
/6
    ew 30 138BC
    ew 34 203CA513
/7
    ew 38 138B0
    ew 3c 203C2513
```

```
/8
    ew 40 138B4
    ew 44 203C2513
/9
    ew 48 138B8
    ew 4c 203C2513
/A
    ew 50 138BC
    ew 54 203C2513
/B --
    ew 58 138B0
    ew 5c 200C2513
/C
    ew 60 138B4
    ew 64 200C2513
/D
    ew 68 138B8
    ew 6c 200C2513
/E
    ew 70 138BC
    ew 74 200C2513
/F --
    ew 78 138c0
    ew 7c CA060004
/10
    ew 80 138c0
    ew 84 CA060044
/11
    ew 88 138C4
    ew 8c 002C0545
```

```
/12
    ew 90 138c0
    ew 94 CA060045
/13
    ew 98 138c0
    ew 9c CA060005
/14
    ew a0 138c0
    ew a4 4A060005
/15
    ew a8 138c0
    ew ac CA060005
/16 --
    ew b0 138c0
    ew b4 CA060085
/17
    ew b8 138B0
    ew bc 203C2513
/18
    ew c0 138B4
    ew c4 203C2513
/19
    ew c8 138B8
    ew cc 203C2513
/1A
    ew d0 138BC
    ew d4 203C2513
/1B
    ew d8 138B0
    ew dc 203CA513
```

```
/1C
    ew e0 138B4
    ew e4 203CA513
/1D
    ew e8 138B8
    ew ec 203CA513
/1E
    ew f0 138BC
    ew f4 203CA513
/1F
    ew f8 138B0
    ew fc 203CE513
/20
    ew 100 138B4
    ew 104 203CE513
/21
    ew 108 138B8
    ew 10c 203CE513
/22
    ew 110 138BC
    ew 114 203CE513
/23
    ew 118 138c8
    ew 11c 7FFF0002

/24 start of port 6 initialization
    ew 120 136c8
    ew 124 7FFF0012
/25
    ew 128 136c0
    ew 12c CA060084
```

```
/26
    ew 130 136B0
    ew 134 203CA513
/27
    ew 138 136B4
    ew 13c 203CA513
/28
    ew 140 136B8
    ew 144 203CA513
/29
    ew 148 136BC
    ew 14c 203CA513
/2a
    ew 150 136B0
    ew 154 203C2513
/2b
    ew 158 136B4
    ew 15c 203C2513
/2c
    ew 160 136B8
    ew 164 203C2513
/2d
    ew 168 136BC
    ew 16c 203C2513
/2e
    ew 170 136B0
    ew 174 200C2513
/2f
    ew 178 136B4
    ew 17c 200C2513
```

```
/30
    ew 180 136B8
    ew 184 200C2513
/31
    ew 188 136BC
    ew 18c 200C2513
/32
    ew 190 136c0
    ew 194 CA060004
/33
    ew 198 136c0
    ew 19c CA060044
/34
    ew 1a0 136C4
    ew 1a4 002C0545
/35
    ew 1a8 136c0
    ew 1ac CA060045
/36
    ew 1b0 136c0
    ew 1b4 CA060005
/37
    ew 1b8 136c0
    ew 1bc 4A060005
/38
    ew 1c0 136c0
    ew 1c4 CA060005
/39
    ew 1c8 136c0
    ew 1cc CA060085
```

```
/3a
    ew 1d0 136B0
    ew 1d4 203C2513
/3b
    ew 1d8 136B4
    ew 1dc 203C2513
/3c
    ew 1e0 136B8
    ew 1e4 203C2513
/3d
    ew 1e8 136BC
    ew 1ec 203C2513
/3e
    ew 1f0 136B0
    ew 1f4 203CA513
/3f
    ew 1f8 136B4
    ew 1fc 203CA513
/40
    ew 200 136B8
    ew 204 203CA513
/41
    ew 208 136BC
    ew 20c 203CA513
/42
    ew 210 136B0
    ew 214 203CE513
/43
    ew 218 136B4
    ew 21c 203CE513
```

```
/44
    ew 220 136B8
    ew 224 203CE513
/45
    ew 228 136BC
    ew 22c 203CE513
/46
    ew 230 136c8
    ew 234 7FFF0002
/47
    ew 238 8
    ew 23c deadbeef
```

Index

Numerics

A

B

C

D

E

F

G

H

I

J

L

M

P

R

S

T

U

W

IMPORTANT NOTICE AND DISCLAIMER

RENESAS ELECTRONICS CORPORATION AND ITS SUBSIDIARIES (“RENESAS”) PROVIDES TECHNICAL SPECIFICATIONS AND RELIABILITY DATA (INCLUDING DATASHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING, WITHOUT LIMITATION, ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for developers skilled in the art designing with Renesas products. You are solely responsible for (1) selecting the appropriate products for your application, (2) designing, validating, and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, or other requirements. These resources are subject to change without notice. Renesas grants you permission to use these resources only for development of an application that uses Renesas products. Other reproduction or use of these resources is strictly prohibited. No license is granted to any other Renesas intellectual property or to any third party intellectual property. Renesas disclaims responsibility for, and you will fully indemnify Renesas and its representatives against, any claims, damages, costs, losses, or liabilities arising out of your use of these resources. Renesas' products are provided only subject to Renesas' Terms and Conditions of Sale or other applicable terms agreed to in writing. No use of any Renesas resources expands or otherwise alters any applicable warranties or warranty disclaimers for these products.

(Rev.1.0 Mar 2020)

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan
www.renesas.com

Contact Information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit:
www.renesas.com/contact/

Trademarks

Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners.