Tsi301™ HyperTransport to PCI User Manual

80D3000_MA001_02

September 2009

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

GENERAL DISCLAIMER

Integrated Device Technology, Inc. reserves the right to make changes to its products or specifications at any time, without notice, in order to improve design or performance and to supply the best possible product. IDT does not assume any responsibility for use of any circuitry described other than the circuitry embodied in an IDT product. The Company makes no representations that circuitry described herein is free from patent infringement or other rights of third parties which may result from its use. No license is granted by implication or otherwise under any patent, patent rights or other rights, of Integrated Device Technology, Inc.

CODE DISCLAIMER

Code examples provided by IDT are for illustrative purposes only and should not be relied upon for developing applications. Any use of the code examples below is completely at your own risk. IDT MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND CONCERNING THE NONINFRINGEMENT, QUALITY, SAFETY OR SUITABILITY OF THE CODE, EITHER EXPRESS OR IMPLIED, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT. FURTHER, IDT MAKES NO REPRESENTATIONS OR WARRANTIES AS TO THE TRUTH, ACCURACY OR COMPLETENESS OF ANY STATEMENTS, INFORMATION OR MATERIALS CONCERNING CODE EXAMPLES CONTAINED IN ANY IDT PUBLICATION OR PUBLIC DISCLOSURE OR THAT IS CONTAINED ON ANY IDT INTERNET SITE. IN NO EVENT WILL IDT BE LIABLE FOR ANY DIRECT, CONSEQUENTIAL, INCIDENTAL, INDIRECT, PUNITIVE OR SPECIAL DAMAGES, HOWEVER THEY MAY ARISE, AND EVEN IF IDT HAS BEEN PREVIOUSLY ADVISED ABOUT THE POSSIBILITY OF SUCH DAMAGES. The code examples also may be subject to United States export control laws and may be subject to the export or import laws of other countries and it is your responsibility to comply with any applicable laws or regulations.

LIFE SUPPORT POLICY

Integrated Device Technology's products are not authorized for use as critical components in life support devices or systems unless a specific written agreement pertaining to such intended use is executed between the manufacturer and an officer of IDT.

1. Life support devices or systems are devices or systems which (a) are intended for surgical implant into the body or (b) support or sustain life and whose failure to perform, when properly used in accordance with instructions for use provided in the labeling, can be reasonably expected to result in a significant injury to the user.
2. A critical component is any components of a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system, or to affect its safety or effectiveness.

About This Manual

Notes

This reference manual contains technical specifications for the Tsi301 HyperTransport PCI bridge chip. The manual is intended for use by system designers and developers who are using the chipset in their designs.

HyperTransport™ and LDT

In this manual, HyperTransport™ technology and LDT (Lightning Data Transport) are sometimes used interchangeably. HyperTransport™ technology and LDT both refer to identical technology based on the same specification.

Throughout the remainder of this manual HyperTransport™ technology is referred to as HyperTransport.

Silicon Revisions: Pass1 and Pass2

Silicon revisions are referred to as Pass1 and Pass 2 throughout this document. Pass1 refers to silicon revisions 1.0 and 1.1. Pass2 refers to silicon revisions 2.0 and higher. Any information not flagged as specific to Pass1 or Pass 2 is applicable to all Tsi301 bridges. *Pass1* and *Pass2* are italicized throughout this document's text to help identify the relevant information.

The following sections describe the intended audience, scope, and the organization of this reference manual.

Audience

This reference manual is intended for system designers and others who design using the HyperTransport PCI bridge, or who evaluate computer systems based on this chip.

Scope

This manual describes the features, requirements, and configurations for the HyperTransport PCI bridge, including functional and physical details.

Specific details on industry standards (for example, PCI or ISA bus specifications) are not included. Additional information is available in the appropriate vendor and IEEE specifications.

Conventions and Definitions

This section describes stylistic conventions used in this manual, and defines product-specific terminology, acronyms, and other technical language used throughout this manual.

Information Flags

Note: Notes highlight information that provides additional clarification or will ease tasks related to the assembly and operation of the system.

Typographic Conventions

This manual uses the following type conventions:

- *Italic* type is used for document citations and information that provides supplemental clarification. *Pass1* and *Pass2* are italicized throughout this document's text to help identify the relevant information.
- `Courier font` is used for type that appears on a screen, such as an example of computer output.

Notes

Signals and Bits

- Active-Low Signals—Signal names ending in _N, such as SFILL_N, indicate active-low signals. Active-low signals are asserted in their low-voltage state and negated in their high-voltage state.
- Active-High Signals are asserted in their high-voltage state and negated in their low-voltage state.
- Differential Signals—Differential signals are indicated by the _H and _L suffixes. Signals ending in _H are the active-high half of a differential pair whereas signals ending in _L are the active-low half of a differential pair.
- Signal Ranges—The highest and lowest signal numbers in a range of signals are contained in brackets and separated by a colon; for example D[63:0].
- Reserved Bits and Signals—Signals or bus bits marked reserved must be driven inactive or left unconnected, as indicated in the signal descriptions. These bits and signals are reserved by API NetWorks, Inc. for future implementations. When software reads registers with reserved bits, the reserved bits must be masked. When software writes such registers, it must first read the register and change only the non-reserved bits before writing back to the register.

Data

The following list defines data terminology:

- Quantities
  - A *word* (W) is two bytes (16 bits).
  - A *doubleword* (DW) is four bytes (32 bits).
  - A *quadword* (QW) is eight bytes (64 bits).
- Abbreviations—The following notation is used for bits and bytes:
  - Kilo—K, as in 4-Kbyte page (2^{10}).
  - Mega—M, as in 4 Mbits/sec (2^{20}).
  - Giga—G, as in 4 Gbytes of memory space (2^{30}).
- Little-Endian Convention—The byte with the address xx...xx00 is in the least-significant byte position (little end). In byte diagrams, bit positions are numbered from right to left: The little end is on the right, and the big end is on the left.
- Bit Ranges—In text, bit ranges are shown with a colon (for example, 3:0). When accompanied by a signal or bus name, the highest and lowest bit numbers are contained in brackets and separated by a colon (for example, AD[31:0]).
- Bit Values—Bits can either be set to 1 or cleared/reset to 0.
- Hexadecimal and Binary Numbers—Unless the context makes interpretation clear, hexadecimal numbers are followed by an *h*, binary numbers are followed by a *b*, and decimal numbers are followed by a *d*.

Acronyms

The following is a list of acronyms used in this document.

Acronym	Definition
APIC	Advanced Programmable Interrupt Controller
BIOS	Basic Input/Output System
CSR	Control/Status Register
DAC	Dual Address Cycle
DDR	Double Data Rate
DMA	Direct Memory Access
DRAM	Direct Random Access Memory
ESBGA	Enhanced Super Ball Grid Array
FIFO	First In, First Out
HSTL	High-Speed Transistor Logic
I/O	Input/Output

Notes

Acronym	Definition
ISA	Industry Standard Architecture
LSB	Least Significant Bit
LVD	Low Voltage Differential
LVTTL	Low Voltage Transistor-Transistor Logic
MRL	Memory Read Line
MRM	Memory Read Multiple
MSB	Most Significant Bit
MWI	Memory Write Invalid
NDA	Non-disclosure Agreement
NMI	Non-maskable Interrupt
ORC	Outbound Request Controller
OS	Operating System
PBGA	Plastic Ball Grid Array
PCB	Printed Circuit Board
PIO	Programmed Input/Output
PCI	Peripheral Component Interconnect
PLL	Phase Locked Loop
SBGA	Super Ball Grid Array
SDRAM	Synchronous Direct Random Access Memory
SE	Single-ended
SIP	Serial Initialization Packet
SPD	Serial Presence Detect
SRI	Serial ROM Interface
SROM	Serial Read Only Memory
SRAM	Static Random Access Memory
SRM	System Reference Manual

Notes

Chapter 1 Bridge Features

Notes

This chapter describes the features and operation of the API NetWorks HyperTransport PCI bridge.

1.1 HyperTransport PCI Bridge Features

The HyperTransport PCI bridge is an I/O bridge from HyperTransport to PCI with the following features:

- An 8-bit HyperTransport primary interface capable of running at data rates up to 800 Mb/s and compliant with *Lightning Data Transport I/O Link Protocol Specification, Revision 1.0*.
- A 64-bit, 66 MHz PCI secondary interface compliant with the *PCI Local Bus Specification, Revision 2.2.*
- A PCI arbiter supporting six external bus masters (not counting the Tsi301) and compliant with the *PCI Local Bus Specification, Revision 2.2.*
- A HyperTransport interrupt controller with configurable support for up to 20 interrupt sources in *Pass1*, and 16 interrupt sources in *Pass2*.
- A configuration register set and programming model consistent with the *PCI-to-PCI Bridge Architecture Specification, Revision 1.1.*
- Power-up configuration through Serial Initialization Packet (SIP) or pin sampling.

1.2 HyperTransport Interface

The HyperTransport PCI bridge primary interface is a HyperTransport tunnel. The primary interface is compliant with *Lightning Data Transport I/O Link Protocol Specification, Revision 1.0* with the exception that the Tsi301 does not support 2 and 4 bit HyperTransport interfaces and does not support Dynamic Frequency Programming. The interface contains two HyperTransport links which allow the connection of multiple bridge chips in a daisy-chain configuration.

Each HyperTransport link has an 8-bit DDR transmit and an 8-bit DDR receive port running at clock speeds up to 400 MHz, allowing for raw bandwidth of 800 MB/s simultaneously in each direction. With protocol overhead, the maximum sustainable bandwidth in one direction is approximately 705 MB/s.

- For testing, or connection to slower devices, the HyperTransport PCI bridge may be programmed to operate at slower link clock rates.
- The HyperTransport PCI bridge supports both the synchronous and asynchronous modes of link initialization.

1.3 PCI Interface

The HyperTransport PCI bridge secondary interface is a 64-bit, 66 MHz capable PCI bus. The HyperTransport PCI bridge supports running the bus in 32-bit mode at either 33 MHz or 66 MHz. A lower performance mode is also supported in which the PCI bus runs at 25 or 50 MHz. The bus can be configured for compatibility with 3.3 V or 5.0 V operation, although 50 and 66 MHz are only supported at 3.3 V.

The HyperTransport PCI bridge supports the full 40-bit memory-mapped space and 25-bit I/O space described in the *Lightning Data Transport I/O Link Protocol Specification, Revision 1.0*. In *Pass1*, PCI addresses outside these spaces alias into them. In *Pass2*, PCI addresses outside these spaces are left on the PCI bus. PCI dual address cycle (DAC) support is provided both inbound and outbound to support the memory-mapped space.

- The HyperTransport PCI bridge supports configuration accesses to devices 0–15, using Address/Data (AD) bits 16–31 for IDSEL#.
- The HyperTransport PCI bridge implements all parity and error-checking features described in the *PCI Local Bus Specification, Revision 2.2*.

Notes

1.3.1 PCI Master

As a PCI master, the HyperTransport PCI bridge chip can generate MemRd/Wr, IORd/Wr, and ConfigRd/Wr cycles.

- The HyperTransport PCI bridge does not implement a cacheline size register and does not prefetch to PCI, so it never generates MemRdLine, MemRdMult, or MemWrInv cycles.
- The HyperTransport PCI bridge does not support a Southbridge connection to its PCI bus, so it never generates INTA cycles.
- The HyperTransport PCI bridge does not generate Special cycles.

PCI master cycles that are retried or disconnected on the PCI bus are reissued locally by the HyperTransport PCI bridge until they complete. The HyperTransport PCI bridge can track up to four outstanding requests in the Outbound Request Controller, of which a maximum of three may be nonposted requests. The HyperTransport PCI bridge rotates among these requests to maximize bandwidth in the presence of retries or disconnects.

1.3.2 PCI Slave

As a PCI slave, the HyperTransport PCI bridge can respond to all types of memory and I/O cycles. However, the HyperTransport PCI bridge never responds to PCI configuration cycles.

- The HyperTransport PCI bridge employs medium DEVSEL# timing.
- All PCI slave writes, including I/O writes, are posted.
- A total of 48 doublewords (DW) of write data buffering are provided on the chip.
- All PCI slave reads are implemented as delayed requests, with up to four delayed requests outstanding at once.

Prefetching is supported for all flavors of memory read cycle, with separate prefetch controls for each cycle type and a maximum prefetch per read of 128 DW. Prefetching may be done once at the beginning of each read, or it may be enabled to continuously issue requests as data is drained to PCI. All prefetch data is discarded when the read disconnects on the PCI bus. The bridge chip provides buffer space for a total of 256 DW of read prefetch data.

1.3.3 PCI Arbiter

The HyperTransport PCI bridge implements an on-chip two-level PCI Arbiter with request/grant pairs: 7 pairs in *Pass1*, 6 pairs in *Pass2*. The request/grant pairs in *Pass1* include two high-priority sets for the on-chip PCI master, one high priority set for an external requester, and five symmetrical sets for external device use. In *Pass2*, there is one high-priority set for the on-chip PCI master.

All connections to the arbiter are through external pins, so its use is optional. The HyperTransport PCI bridge chip may also be configured to use an external PCI arbiter.

1.4 Interrupt Controller

The HyperTransport PCI bridge implements a HyperTransport interrupt controller. In *Pass1*, there are 20 external interrupt sources. In *Pass2*, there are 16 external interrupt sources.

The interrupts pins may be programmed in groups of four to be level or edge-triggered and active high or low. In *Pass1*, one group can also be used to generate the special PC compatibility interrupts: SMI, NMI, INIT, and INTR. In *Pass2*, SMI, NMI, INIT, and INTR are not available.

1.5 Configuration

Most HyperTransport PCI bridge configuration is done under host software control through accesses across HyperTransport to the bridge chip's control/status register (CSR) set. However, some hardware initialization is required to bring up the HyperTransport links before software configuration can occur. To support hardware initialization, the HyperTransport PCI bridge provides a Serial Initialization Packet (SIP) interface to read an external SROM during a cold reset sequence.

Notes

To reduce external part count, a small number of configurations are pre-programmed into the HyperTransport PCI bridge. These pre-programmed configurations may be selected using PCI bus pins during cold reset in the absence of an SROM.

1.6 Interface Levels

A complete pinout of the HyperTransport PCI bridge is given in the Signals Chapter. The rough grouping of signal types is shown in Table 1.

Interface	Group	Voltages
PCI	PCI	3.3V, 5.0V tolerant
HyperTransport	HyperTransport	Differential, 600 mV swing, centered on 600 mV
Interrupts	INT	In *Pass1*, 2.5 V In *Pass2*, 3.3 V

Table 1 HyperTransport PCI Bridge Interface Voltages

1.7 Clocking

In normal operation, the HyperTransport PCI bridge's reference clock (REFCLK_H/L) comes from a PCI bus clock. This clock is received from the same source that drives clocks to devices on the bridge's PCI bus and is nominally in phase with it; although it may be delayed relative to the other PCI bus clocks. Reference clock frequency is indicated by the P_M66EN pin. From this input, an internal phase locked loop (PLL) generates the HyperTransport PCI bridge internal core clock and the HyperTransport transmit clocks.

> **Note:** In Pass1, the reference clock frequency and P_M66EN can be set at either 33 or 66 MHz at L_PWROK and must remain static. In Pass2, clocks and P_M66EN can be used in the same way; or the PCI bus frequency can be changed without resetting the HyperTransport PCI bridge. To support changing the PCI bus frequency without resetting the HyperTransport PCI bridge, REFCLK_H/L must always run at 33 MHz and P_M66EN must indicate 33 MHz at the rising edge of L_PWROK. The clock frequency to the PCI devices and P_M66EN can then be switched between 33 and 66 MHz whenever AP_RST is asserted. The clocks must still be nominally in phase and rising edges of the reference clock must be aligned with rising edges of the PCI bus clock.

If the HyperTransport PCI bridge is to perform synchronous link initialization with the HyperTransport devices on either side of it, the reference clock must be derived from the same base frequency source. If not, asynchronous link initialization must be used. No phase relationship is required in either mode.

For debug and test purposes, the HyperTransport PCI bridge allows bypassing of the PLL. It provides separate bypass clock inputs for the core and HyperTransport transmit clocks. The bypass clocks run directly at the frequency provided. Both bypass clocks must be derived from the same base frequency source.

Notes

Notes

2.1 HyperTransport Signals

Note: The Lx_ prefix denotes signals associated with either HyperTransport Link 0 (x=0) or HyperTransport Link 1 (x=1).

The PCI bridge HyperTransport implementation is a standard HyperTransport design. Table 2.1 lists all HyperTransport signals. For details on the characteristics of these signals, refer to the *Lightning Data Transport I/O Link Protocol Specification, Revision 1.0* and the *Lightning Data Transport Electrical Specification, Revision 0.77*, both from AMD.

Signal Name	I/O Type	Signal Type
Lx_RX_CLK_H/L	Input	HyperTransport
Lx_RX_CTL_H/L	Input	HyperTransport
Lx_RX_CAD[7:0]_H/L	Input	HyperTransport
Lx_TX_CLK_H/L	Output	HyperTransport
Lx_TX_CTL_H/L	Output	HyperTransport
Lx_TX_CAD[7:0]_H/L	Output	HyperTransport
Lx_VLDT[2:0]	Power	1.2 volt
L_TSTRST_N	Input	2.5 volt LVCMOS
L_POWER_OK	Input	2.5 volt LVCMOS
L_RST_N	Input	2.5 volt LVCMOS

Table 2.1 HyperTransport Bridge Signals

2.2 PCI Signals

The HyperTransport PCI bridge implements a standard PCI interface, as detailed in *PCI Local Bus Specification, Revision 2.2*. Table 2.2 lists all HyperTransport PCI bridge PCI signals.

Signal Name	I/O Type	Signal Type
P_CBE_N[7:0]	Bidirectional	PCI
P_AD[63:0]	Bidirectional	PCI
P_PAR	Bidirectional	PCI
P_SERR_N	Input	PCI
P_PERR_N	Bidirectional	PCI
P_LOCK_N	Bidirectional	PCI
P_STOP_N	Bidirectional	PCI
P_DEVSEL_N	Bidirectional	PCI
P_TRDY_N	Bidirectional	PCI

Table 2.2 HyperTransport Bridge PCI Signals

Notes

Signal Name	I/O Type	Signal Type
P_IRDY_N	Bidirectional	PCI
P_FRAME_N	Bidirectional	PCI
P_REQ64_N	Bidirectional	PCI
P_ACK64_N	Bidirectional	PCI
P_PAR64	Bidirectional	PCI
AP_RST_N	*Pass1* Output *Pass2* Input/Output	Open Drain
P_M66EN	Input	PCI
P_WSC_N	Reserved	PCI

Table 2.2 HyperTransport Bridge PCI Signals<Emphasis> (Continued)

Table 2.3 lists the PCI Arbitration signals, which are not defined by the PCI specification.

Signal Name	I/O Type	Signal Type
P_REQ_OUT_N	Output	PCI
P_GNT_IN_N	Input	PCI
P_REQ[5:0]_N	Input	PCI
P_GNT[5:0]_N	Output	PCI
P_PREQ_N	Input	PCI
P_PGNT_N	Output	PCI

Table 2.3 PCI Arbitration Signals

Note: Any unused P_REQ[5:0]_N or P_PREQ_N requests should be pulled to 3.3 volts (inactive).

2.2.1 PCI Signal Levels

The HyperTransport bridge supports PCI 3.3 V levels and PCI 5.0 V levels as determined by AP_TYPEDET_N and VDD3050_PCI[7:0]. AP_TYPEDET_N is related to PCI functionality but not part of the PCI specification.

If AP_TYPEDET_N is pulled low and VDD3050_PCI[7:0] is at 3.3 V, PCI signals conform to the PCI 3.3 V signalling rules.

If AP_TYPEDET_N is pulled high and VDD3050 PCI[7:0] is at 5.0 V, PCI signals conform to the PCI 5.0 V signalling rules.

No other combinations of AP_TYPEDET_N and VDD3050 PCI[7:0] are supported.

Notes

2.3 Interrupt and Error Signals

Note: In Pass2, SMI, NMI, INIT, and INTR are not available. Customers should use the BLKx_INT[y] signals instead.

BLKx_IRQ[y]

Functionally input only. In *Pass1*, 2.5 volt LVCMOS. In *Pass2*, 3.3 volt LVCMOS.

Generic interrupt input signals. The HyperTransport PCI bridge has four blocks of generic interrupts, with four interrupts per block. The lower 2 bits of the vector are contained in each interrupt. The upper 6 bits are in the interrupt block description register.

The y denotes four separate signals associated with each of the four interrupt blocks x, for a total of 16 signals.

Note: In Pass1 if this pin is not needed for an interrupt, it can be floated or pulled to 2.5 volts.

In Pass2 if this pin is not needed for an interrupt, it can be floated or pulled to 3.3 or 2.5 volts. 3.3 volts is preferable.

NMI

Bidirectional. In *Pass1*, 2.5 volt LVCMOS. In *Pass2*, this signal is absent.

Non-Maskable Interrupt Signal. This signal is not available in *Pass2*. Customers should use the BLKx_INT[y] signals instead.

Note: In Pass1 if this pin is not needed for an interrupt, it can be floated or pulled to 2.5 volts.

SMI_N

Bidirectional. In *Pass1*, 2.5 volt LVCMOS. In *Pass2*, this signal is absent.

System Management Interrupt Signal. This signal is not available in *Pass2*. Customers should use the BLKx_INT[y] signals instead.

Note: In Pass1 if this pin is not needed for an interrupt, it can be floated or pulled to 2.5 volts.

INTR

Input. In *Pass1*, 2.5 volt LVCMOS. In *Pass2*, this signal is absent.

Interrupt Input Signal. This signal is not available in *Pass2*. Customers should use the BLKx_INT[y] signals instead.

INIT

Bidirectional. In *Pass1*, 2.5 volt LVCMOS. In *Pass2*, this signal is absent.

x86 INIT signal. This signal is not available in *Pass2*. Customers should use the BLKx_INT[y] signals instead.

Note: In Pass1 if this pin is not needed for an interrupt, it can be floated or pulled to 2.5 volts.

FATAL_ERR_N

Open Drain Output. In *Pass1*, 2.5 volt open drain LVCMOS. In *Pass2*, 3.3 volt open drain LVCMOS.

The HyperTransport PCI bridge has detected a fatal error.

NONFATAL_ERR_N

Open Drain Output. In *Pass1*, 2.5 volt open drain LVCMOS. In *Pass2*, 3.3 volt open drain LVCMOS.

The HyperTransport PCI bridge has detected a non-fatal error.

Notes

2.4 SIP Signals

SROM_SCK

Bidirectional. In *Pass1*, 2.5 volt open drain LVCMOS. In *Pass2*, 3.3 volt open drain LVCMOS.

P_AD[1] drives the CLK pin of the SIP ROM. During reset, if this pin is pulled high with a 10 K resistor, the SROM supplies the entire SIP packet. If low, the SIP packet is generated internally.

SROM_SDA

Bidirectional. In *Pass1*, 2.5 volt LVCMOS. In *Pass2*, 3.3 volt LVCMOS.

This pin is the data pin of the SIP ROM.

2.5 Test Signals

For systems using Pass1 of the HyperTransport PCI bridge, the TDI and TDO pins are not guaranteed to be quiescent during scan operations. Customers should ensure their JTAG implementation on the PCB is tolerant of this.

TCK

Input. In *Pass1*, 2.5 volt LVCMOS. In *Pass2*, 3.3 volt LVCMOS.

In *Pass1*, this is a reserved input and can be left floating or pulled to GND.

In *Pass2*, this is the JTAG clock input.

TDI

***Pass1* output. *Pass2* input. In *Pass1*, 2.5 volt open drain LVCMOS. In *Pass2*, 3.3 volt open drain LVCMOS.**

In *Pass1*, this is a reserved output and not guaranteed to be quiet.

In *Pass2*, this is the JTAG data input.

TDO

Output. In *Pass1*, 2.5 volt LVCMOS. In *Pass2*, 3.3 volt LVCMOS.

In *Pass1*, this is a reserved output and not guaranteed to be quiet.

In *Pass2*, this is the JTAG data output.

TMS

Input. In *Pass1*, 2.5 volt LVCMOS. In *Pass2*, 3.3 volt LVCMOS.

In *Pass1*, this is a reserved input and can be left floating or pulled to GND.

In *Pass2*, this is the JTAG mode select.

TRST_N

Input. In *Pass1*, 2.5 volt LVCMOS. In *Pass2*, 3.3 volt LVCMOS.

In *Pass1*, this is a reserved input.

In *Pass2*, this is the JTAG reset input.

Note: This signal should be pulled low for normal functional mode.

TMODE [3:0]

Input. Test pins. In *Pass1*, 2.5 volt LVCMOS. In *Pass2*, 3.3 volt LVCMOS.

Notes

TRISTATE_N

Input. In *Pass1*, 2.5 volt LVCMOS. In *Pass2*, 3.3 volt LVCMOS.

Tristate enable, see the Signals Chapter.

VBB

This pin is tied to the substrate for testing purposes. It should not be connected on the PCB. This signal is not available in *Pass2.*

2.6 PLL/Clock Signals

REFCLK_H/L

Input. GTL+.

This 33/66 MHz input is the reference for the HyperTransport PCI bridge PLL. To improve 66 MHz timing, it may be delayed relative to other PCI clocks on the PCI bus. This is a differential signal. The _L input can be attached to the active-low half of a differential pair, or it can be tied to a reference voltage.

REFCLK_PLL_BYPASS

Input. 2.5 volt LVCMOS.

When this signal is asserted, the PLL is bypassed. REFCLK_H/L input pins are used directly to drive internal clocks.

REFCLK_PLL_VCC

This is the 2.5 V power input for the PLL.

REFCLK_PLL_GND

This is the ground input for the PLL.

2.7 Core and I/O Power Signals

VDD

Core power. 2.5 V.

VSS

Ground.

VDDQ_AP[18:0]

PCI VDD supply voltage. Always 3.3 V, regardless of whether the HyperTransport bridge is using 3.3 V PCI levels or 5.0 V PCI levels.

VDD3O5O_PCI[7:0]

If AP_TYPEDET_N is pulled low, these should be at 3.3 V. PCI signalling will conform to the 3.3 V rules.

If AP_TYPEDET _N is pulled high, these should be at 5.0 V. PCI signalling will conform to the 5.0 V rules.

VDD3P_AGP[3:1]

In *Pass1*, 3.3 V.

In *Pass2*, these signals are absent.

Lx_VLDT[y]

HyperTransport I/O power. 1.2 V.

Notes

2.8 Calibrator Logic Signals

Lx_RREF_GND/Lx_RREF

Bidirectional

These signals are used by the HyperTransport calibrator logic. A resistor should be placed between these two pins. The resistor value should be equal to half of the differential impedance of the HyperTransport signals, which is 100 ohms +/- 10%.

Note: The LDT_RREF_GND should not be connected to ground.

Chapter 3 Functional Operation

Notes

This chapter details the operation of the HyperTransport PCI Bridge chip.

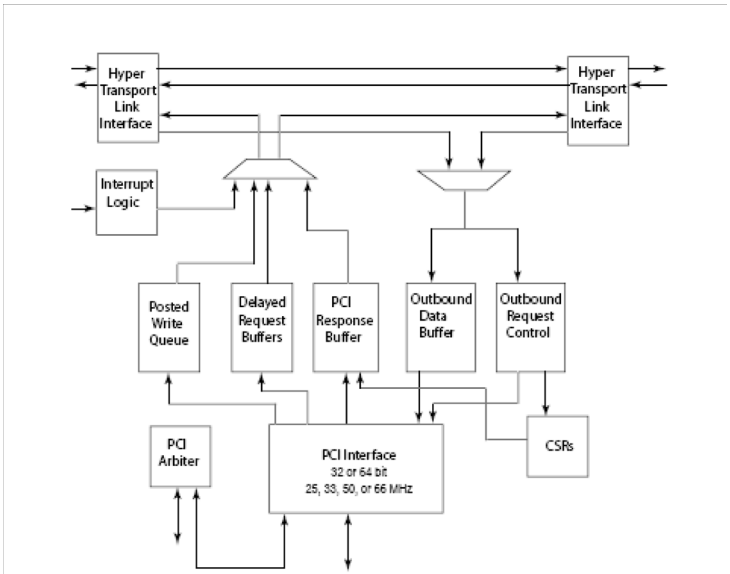

Figure 3.1 HyperTransport PCI Bridge Block Diagram

3.1 HyperTransport Interface

The HyperTransport PCI bridge HyperTransport interface consists of two identical link interfaces, each with a HyperTransport transmitter and receiver. Some central reset and error-handling logic is shared between the two links.

In the HyperTransport protocol, all logical packet transfer is between HyperTransport slaves and the host. Direct peer-to-peer communication is not allowed. To support peer-to-peer operations, packets are reflected through the host. Packets issued from the host to a HyperTransport slave are defined to travel downstream on the HyperTransport chain. Packets issued from a HyperTransport slave to the host are defined to travel upstream. Intermediate nodes in the daisy chain forward packets from link to link until they reach their final destination, which accepts the packet.

Link interfaces in the HyperTransport PCI bridge are symmetrical, which allows connection of either bridge link toward a host. The HyperTransport PCI bridge also supports being placed in a double-hosted chain with hosts at both ends.

Figure 3.2 shows a block diagram of a single HyperTransport link interface.

Notes

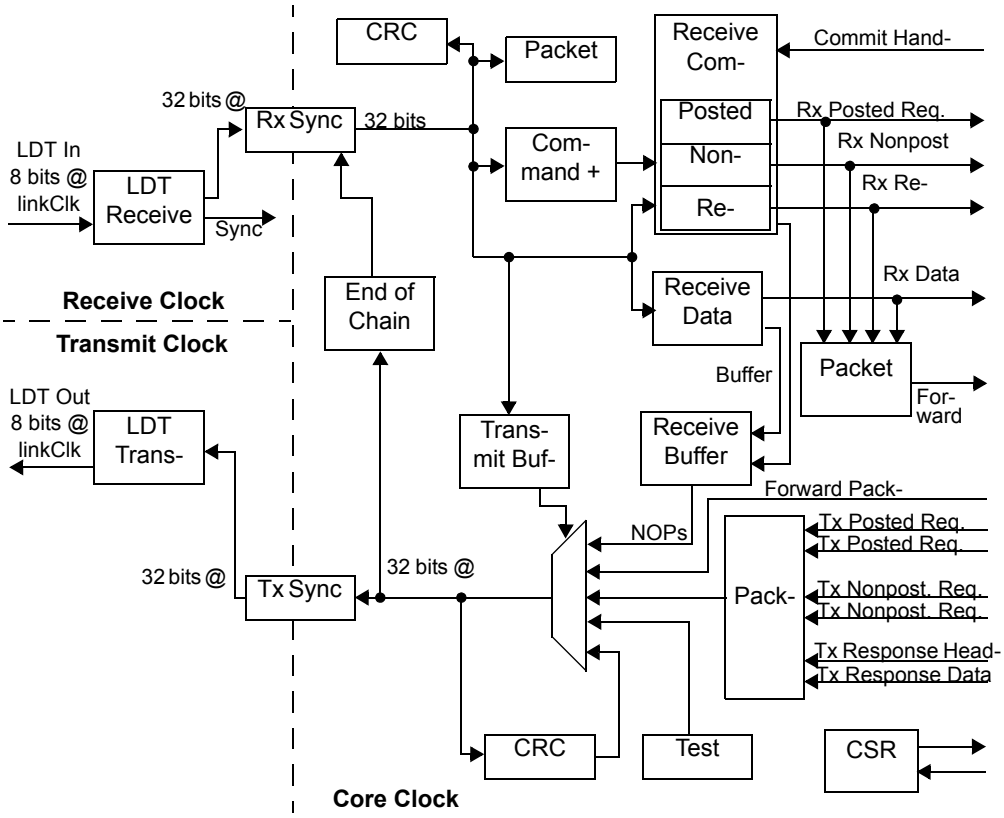

Figure 3.2 Single HyperTransport Link Interface Block Diagram

3.2 HyperTransport Packet Reception

Packets received from HyperTransport are placed into the HyperTransport Receive (Rx) buffers. The HyperTransport flow control algorithm guarantees that no packet is received without buffer space to store it. Packet contents are divided into command information (including address) and data, with separate buffers for each.

- Command buffers are statically partitioned among the three virtual channels (posted requests, non-posted requests, and responses), with each channel allocated space to hold four commands.
- Data buffers are shared among the three virtual channels and dynamically allocated among them.

A total of eight data buffers are provided. Allocation of buffers to virtual channels is controlled by the Data Buffer Allocation CSR (6Dh:6Ch). As data buffers are allocated, buffers are released from the pool to each channel. As the buffers are retired, they return to the pool.

3.2.1 Data Buffer Allocation

Each virtual channel always needs at least one data buffer allocated to it to prevent deadlock. A data buffer is considered allocated to a virtual channel if it has been released to that channel and is awaiting data, or if it has received data but has not yet been retired. Two types of user specified data buffer allocation are allowed:

- Guaranteed buffers specified in the NeedPReq, NeedNpReq, and NeedResp CSR fields.
- Assigned but not guaranteed buffers specified in the WantPReq, WantNpReq, and WantResp CSR fields.

Guaranteed data buffer allocations to each channel are user configurable using NeedPReq, NeedNpReq, and NeedResp CSR fields to set minimum buffer allocations for each virtual channel. If retiring a data buffer to the pool would cause a virtual channel to fall below its allocation, that buffer is immediately re-allocated to the

Notes

same channel. Re-allocation guarantees that the minimum allocation set in the Need CSR fields will always be met as long as the sum of the CSR values is less than or equal to the total number of data buffers in the link interface.

The buffer allocation strategy is to always keep some number of granted but unfilled buffers allocated to each virtual channel so that data is always able to move. Once the minimum configured allocations are met, remaining unallocated buffers are brought out of the pool and granted based on traffic demands. These are the assigned buffers specified in WantPReq, WantNpReq, and WantResp.

The WantPReq, WantNpReq, and WantResp CSR fields specify the number of free buffers attempted (not guaranteed) for each channel. Whenever the number of free buffers in a particular channel drops below its Want value, a new buffer is allocated to that channel from the pool based on buffer availability. Depending on the number of full data buffers, it may not be possible to always satisfy Want buffer assignments.

The WantPReq, WantNpReq, and WantResp CSR fields allow the user to assign free buffers to the traffic most important to latency and performance. The goal is to set the Want values high enough for channels that require maximum bandwidth so that there are enough buffers available to hide the latency of a buffer release issued back to the transmitter when the first beat of a data packet is received. The pitfall is setting a channel's Want values so high that it prevents buffers from being available to the other channels, inadvertently throttling performance.

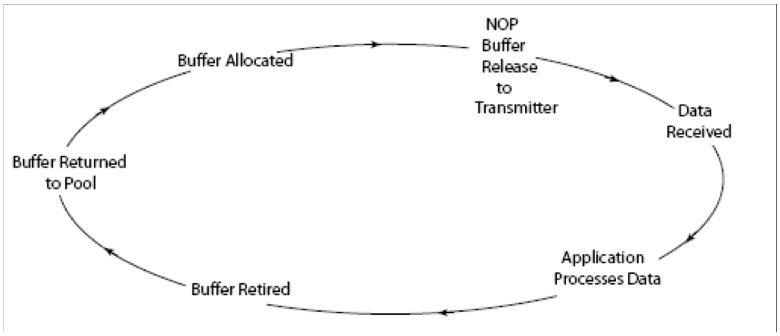

Figure 3.3 Data Buffer Life Cycle

3.2.2 Packet Decode

As packets are placed in the Rx buffers, the associated commands and addresses are decoded to determine whether the HyperTransport PCI bridge is the packet target on the HyperTransport chain. These packets are also checked for ordering collisions against other packets resident in the buffers. The decode and collision results are stored in the buffers with the packets.

- Packets received on the HyperTransport interface may be routed to internal logic, including the PCI interface. This is "accepting" a packet.
- Packets may also be routed to the other HyperTransport link interface for transmission to the next device in the chain. This is "forwarding" a packet.
- A particular packet may be accepted, forwarded, or both.

The HyperTransport PCI bridge HyperTransport packet decode first determines whether the incoming packet is travelling upstream or downstream. This determination is based on the packet source information contained in the packet itself, not on which link is the upstream or downstream link.

- Upstream packets are always forwarded toward the host and are never accepted by the HyperTransport PCI chip.
- Downstream RdSized and WrSized request packet addresses are decoded according to the HyperTransport Address Map described in Section 3.3 and are accepted if they match any HyperTransport PCI bridge address ranges.
- Downstream WrSized and RdSized that do not match any of the HyperTransport PCI bridge address ranges are forwarded to the next device in the chain.
- Broadcast request addresses are also decoded and accepted if they match a HyperTransport PCI

Notes

bridge address range. However, these packets are also always forwarded.

- Fence and Flush requests are never accepted by the HyperTransport PCI bridge and are always forwarded.
- Downstream response packets are accepted if their unitId field matches the value in the BaseUnitId field of the HyperTransport PCI bridge LdtCmd register; otherwise, they are forwarded.

3.2.3 Collision Checking

Collision checking is performed according to the HyperTransport protocol to determine if incoming packets are required to stay ordered behind packets already in the Rx buffers. Only packets headed to the same accept or forward destination may have ordering requirements. If a packet has an ordering collision, it may not be issued from the Rx buffers until the packet with which it collided has both been issued and reached an appropriate commit point to guarantee ordering. This ordering point varies by destination.

Because the Outbound Request Controller (ORC) can reorder requests, requests accepted by the HyperTransport PCI bridge may not be committed until they are retired by the ORC. For accesses to PCI, this means that the request reached the PCI bus and all data was transferred. Packets forwarded from one link to the other are streamed - meaning that transmission may start before the whole packet is received. It is the transmitter's responsibility to ensure that required packet ordering is maintained so packets can be committed as soon as they are passed to the other link controller.

Packets that do not have any ordering requirements may leave the Rx buffers in a different order than they reached them, both between and within virtual channels. In general, the Rx buffers select a packet choosing the oldest non-blocked packet in each channel to a given destination.

3.3 HyperTransport Address Map

HyperTransport implements a single flat 40-bit address space for all accesses. All address spaces that can be reached from HyperTransport are mapped into this space. The HyperTransport PCI bridge checks addresses on incoming packets in each space for subranges that it accepts.

3.3.1 Memory Mapped Space

The HyperTransport specification places Memory Mapped Space in the address range of 00_0000_0000h to FC_FFFF_FFFFh. The HyperTransport PCI bridge accepts two ranges within this space, as enabled by MemSpaceEn in the Command (Cmd) CSR, consisting of the following:

- Memory Space, defined by the MemBase and MemLimit CSRs.
- Prefetchable Memory Space, defined by the PrefMemBaseUpper/ PrefMemBase and PrefMemLimitUpper/PrefMemLimit CSRs.

Setting the VgaEn bit in the BrCtrl CSR creates the additional window of 00_000A_0000h – 00_000B_FFFFh, which is also accepted. The HyperTransport PCI bridge never does prefetching to PCI, so the prefetchable/nonprefetchable attribute of these ranges does not matter. RdSized requests to these ranges result in MemRd requests on the PCI bus. WrSized requests to these ranges result in MemWr requests on the PCI bus. If above 4 GB, addresses are passed straight through as a DAC.

3.3.2 I/O Space

The HyperTransport specification places PCI I/O space in the address range of FD_FC00_0000h to FD_FDFF_FFFFh. The HyperTransport PCI bridge strips the top 15 bits off of addresses in this range.

- If enabled by I/OSpaceEn in the Cmd CSR, the HyperTransport PCI bridge accepts requests that fall in the range defined by the I/O Base and I/O Range Base Upper, and I/O Limit and I/O Range Limit Upper CSRs.
- If set, the IsaEn bit in the Bridge Control CSR creates a series of holes (the top 768 bytes of each 1 KB block in the low 64 KB) in this space that the HyperTransport PCI bridge does not accept.
- Setting the VgaEn bit in the Bridge Control CSR creates an additional set of windows (all addresses in the low 64 KB where the bottom 10 bits are in the ranges 3B0h – 3BBh or 3C0h – 3DFh) which the HyperTransport PCI bridge accepts. Accepted RdSized requests result in IoRd requests on the PCI

Notes

bus, and WrSized requests result in IoWr requests, with the bottom 25 bits of the HyperTransport address passed through. Bits 31:26 are 0.

3.3.3 Configuration Space

The HyperTransport specification places PCI configuration space in the address range of FD_FE00_0000h to FD_FFFF_FFFFh. Address bit 24 identifies requests as Type 0 or Type 1 configuration requests.

- Type 0 requests are accepted and routed to the HyperTransport PCI bridge internal configuration registers if their device number (bits 15:11) matches the value of BaseUnitId in the HyperTransport Command CSR.
- Type 1 requests are accepted if their bus number (bits 23:16) falls within the range defined by the Secondary Bus Number and Subordinate Bus Number CSRs, inclusive. Type 1 requests are routed to PCI, as ConfigRd or ConfigWr cycles.

A Type 1 request with a bus number that exactly matches the Secondary Bus Number CSR becomes a PCI Type 0 configuration request, with AD[deviceNumber + 16] set.

Type 1 requests with a bus number greater than Secondary Bus number but less than or equal to Subordinate Bus number are passed on to PCI as Type 1 configuration cycles. Bits 23:2 of the address are left unchanged. The HyperTransport PCI bridge does not support the Type 1 configuration to special cycle mapping.

3.3.4 Interrupt Space

The HyperTransport specification places interrupt space in the address range of FD_F800_0000h to FD_F8FF_FFFFh. The HyperTransport PCI bridge only accepts End of Interrupt (EOI) requests in this range, which should always be broadcasts. These EOI requests are routed to the interrupt controller.

3.4 HyperTransport Packet Transmission

The HyperTransport packet generator logic is essentially a large arbiter/multiplexer that formats and combines packets from each of the three virtual channels issued from within the HyperTransport PCI bridge. The output stream is combined with the stream of packets forwarded through the HyperTransport PCI bridge from the far HyperTransport link receiver's Rx buffers. This later multiplexing is also used to insert NOP/buffer release messages to the transmitter on the link's other end.

Packet transmission is paced by the transmit buffer counters maintained in each virtual channel for both command/address and data, as the HyperTransport specification describes. These counters are decremented as packets are transmitted and incremented as buffer release messages are received from the transmitter at the link's other end. Transmit buffer counters can be throttled using the txBufCountMax CSR (C8h and CCh).

3.4.1 Packet Insertion

To prevent devices close to the host bridge from starving devices further out in the chain of bandwidth, the HyperTransport PCI bridge implements the packet insertion fairness algorithm described in the HyperTransport specification. This algorithm throttles the insertion rate of packets from the HyperTransport PCI bridge relative to packets being forwarded and attempts to balance the packet insertion rates of all devices on the chain.

Insertion of buffer release messages is forced, even when the outgoing transmission stream is busy. Forcing allows traffic to flow to the HyperTransport PCI bridge continuously while maintaining a relatively small number of Rx buffers. The HyperTransport PCI bridge forces a buffer release message as soon as possible when an Rx buffer is freed, subject to the requirements of the HyperTransport protocol. The frequency of buffer release messages is limited under the control of the LdtTx CSR to prevent them from occupying too much bandwidth in a busy stream. Throttling buffer releases clumps the released messages together and raises their efficiency.

In a single-hosted HyperTransport chain, the HyperTransport PCI bridge may be at the end of the chain furthest from the host and therefore have no downstream link connection. In this case, packets are routed to the End of Chain (EOC) logic in the unconnected link interface. The EOC logic drops responses and posted

Notes

requests and generates Non-Existent Access (NXA) Error responses back into the receiver for nonposted requests. These error responses then get forwarded back to the other HyperTransport link interface and back to the requesting device. Error logging for the dropped packets occurs in the LinkCtrl CSRs (44h and 48h).

3.5 Outbound Transactions

Outbound transactions to the HyperTransport PCI bridge are those accepted from the HyperTransport chain. All outbound requests go first from the HyperTransport link interface on which they are received to the Outbound Request Controller (ORC). This controller is responsible for issuing the request to the appropriate destination functional unit and for tracking the request state while it is outstanding.

The controller has four buffers which allow state tracking for up to four outstanding requests. If multiple requests are outstanding in the controller at once, it rotates among them in round-robin fashion to issue or reissue them. When the ORC file fills, subsequent requests back up to the HyperTransport Rx buffers. Since the ORC doesn't guarantee ordering, the HyperTransport Rx buffers must not issue the second in an ordered pair of transactions until the first has completed.

The ORC is also responsible for managing space in the PCI response data buffer. All outbound nonposted requests, regardless of destination, must be allocated space in the response data buffer before they can be accepted from the HyperTransport Rx buffers by the ORC. The data buffer is not large enough to allocate space for all four requests tracked in the ORC's buffers. This guarantees that the Outbound Request Buffers are never filled with nonposted requests, and posted requests can always get in. This, in turn, provides the deadlock-avoidance guarantee required by the PCI Local Bus specification (nonposted requests are never allowed to block posted requests).

As requests complete at their destinations that fact is signaled back to the ORC which allows the request buffer to be retired. If the request was nonposted, the transaction will require generation of a response to the host. OCR considers posted transactions as complete when the request completes at its destination and the buffer is retired. Nonposted transactions are complete when the response packet is issued to the HyperTransport transmit interface from which the request was received.

3.5.1 CSR Master

All configuration accesses to the HyperTransport PCI bridge CSRs are handled by the CSR master engine. The CSR master engine controls the registers in multiple CSR slaves. These slaves are dispersed around the chip so that registers can be physically close to the functions that they control or monitor. The partition between slave modules is invisible from software's point of view.

The CSR master can handle a single CSR read or write at a time. All CSR accesses complete in a fixed amount of time once they reach the master. Data returned from CSR reads is placed in the PCI response data buffer.

The HyperTransport PCI bridge implements a single-function PCI bridge header with a slave HyperTransport capability block. Because the HyperTransport PCI bridge primary bus is not PCI, not all bits of the standard PCI header are used, as the HyperTransport specification defines. The HyperTransport PCI bridge PCI device number is contained in the BaseUnitId CSR in the HyperTransport capability block and is written as part of the HyperTransport initialization sequence.

Only accesses that fit within a 32-bit aligned block are supported. Accesses that span multiple 32-bit blocks receive an HyperTransport error response, equivalent to a PCI target abort. All registers not listed in the CSR descriptions are considered reserved. Reserved bits return undefined values when read and have undefined results if written with values other than those that were read out. Writes to undefined registers have undefined results.

3.5.2 PCI Outbound Transactions

Outbound requests to PCI are handed to the PCI interface to be driven out to the bus. Write data comes from the Rx data buffers. Read data is returned from the bus and placed in the PCI Response Data Buffer.

Notes

The PCI interface can be enabled to perform fast back-to-back transactions. When the bus is configured as 64 bit at reset (PCI_AD[14]), the interface automatically asserts P_REQ64_N on all transactions for which it is legal. When the bus is configured as 32 bit, the interface automatically shuts down both its input and output buffers for P_AD[63:32], P_CBE_N[7:4], P_REQ64_N, P_ACK64_N, and P_PAR64.

The PCI interface supports Type 0 PCI configuration cycles to device numbers 0 through 15. P_AD[31:16] are driven with a one-hot encoding during these configuration cycles, with bit 16 asserted for accesses to device number 0. This logic assumes that one P_AD bit is connected to the IdSel# pin on each PCI slot through a series resistor on the board. If the resistor increases the settling time of IdSel# to the point at which it doesn't make timing in one PCI cycle, the IdSelCharge field of the PCI Control register may be set to a nonzero value to provide more time. Setting the IdSelCharge field of the PCI Control register to a nonzero value opens a window where the HyperTransport PCI bridge may lose control of the PCI bus because of its failure to drive P_FRAME_N immediately, which may have implications about forward progress.

If a request is retried or disconnected on the PCI bus, that fact is reported back to the ORC. The controller finishes any data movement associated with the disconnected transaction and then reissues the request from the point of disconnection. It continues to reissue a request until it completes or until the retry timer for the request expires. Because the ORC can handle multiple outstanding requests, transactions repeatedly retried or disconnected may be reordered or interleaved.

3.5.3 PCI Response Data Buffer

The PCI response data buffer contains read data returned from outbound reads to either the PCI interface or the CSR master. This data buffer is a RAM holding 48 DW of data, 16 DW each for three HyperTransport read requests.

Because multiple reads may be in progress to the PCI and CSR interfaces at one time, with their returning data getting interleaved by PCI disconnects, this cannot be a FIFO structure. Each buffer accumulates data for its request until all requested DW are returned. This data is then combined with response header information from the ORC to form the response packet for the operation.

Once the response is issued, the buffer is retired. Nonposted write requests still occupy space in the response data buffer, even though they have no read data.

3.5.4 End of Interrupt

When an interrupt is configured as level sensitive, upstream interrupt logic must respond to an interrupt request packet with an end of interrupt (EOI) packet. Until the EOI packet is received by the HyperTransport PCI bridge, no new interrupt request packets will be generated by that interrupt pin.

Note: See Interrupt Generation, Section 3.6.12.

3.6 Inbound Transactions

Inbound transactions are requests from the PCI bus or interrupt controller across HyperTransport to the host bridge. From there, they are routed to destinations behind the host bridge or reflected peer-to-peer back onto the HyperTransport chain. If the request is nonposted, the transaction also includes the response from the host bridge back to the original requesting unit.

The HyperTransport PCI bridge operates as a PCI target for requests from external PCI devices. All accepted requests are forwarded to the HyperTransport link interface leading to the host. Reads go though the delayed request buffers and are handled on the PCI bus as delayed requests. All writes, regardless of type, are posted into the posted request queue and allowed to immediately complete on the PCI bus.

3.6.1 PCI Address Map

Accesses on the PCI bus are checked against the following ranges to determine whether the HyperTransport PCI bridge is the target of the access and should assert P_DEVSEL_N to accept the request. The HyperTransport PCI bridge makes this determination with medium DEVSEL# timing. When configured as a 64-bit target at power up, the HyperTransport PCI bridge asserts P_ACK64_N in response to P_REQ64_N for requests it accepts.

Notes

- **Memory Mapped Cycles.** The HyperTransport PCI bridge implements a 40-bit space for memory mapped accesses and decodes DAC accesses for addresses above 4 GB. In *Pass1*, address bits above bit 39 are ignored and result in the 40-bit space aliasing through PCI's 64-bit memory mapped space. In *Pass2*, any address with a non-zero value in the top 24 bits of the 64 bit PCI address is left on the PCI bus.

Memory mapped addresses are compared to the range defined by the Memory Range Base Addr and Memory Range Limit Addr CSRs; and the range defined by the Prefetchable Memory Range Base Upper and Prefetchable Memory Range Base Addr, and Prefetchable Memory Range Limit Upper and Prefetchable Memory Range Limit Addr CSRs. If the VgaEn bit in the Bridge Control CSR is set, the address is also compared to the fixed range of 00_000A_0000h – 00_000B_FFFFh. Addresses that don't fall into any of these ranges are accepted for forwarding to HyperTransport, as long as the MasterEn bit in the Command CSR is set and bits [39:32] <= Fch.

- **I/O Cycles.** The HyperTransport PCI bridge implements a 25-bit space for I/O accesses. In *Pass1*, address bits above bit 24 are ignored and result in the 25-bit space aliasing through PCI's 32 bit I/O space. In *Pass2*, the top 7 of the 32 bits must be 0 for the access to reach HyperTransport.

I/O addresses are compared to the range defined by I/O Range Base Upper and I/O Base, and I/O Range Limit Upper and I/O Limit CSRs. If the IsaEn bit in the Bridge Control CSR is set, a series of holes are created in this range at the top 768 bytes of each 1 KB block in the low 64 KB. If the VgaEn bit in the Bridge Control CSR is set, I/O addresses in the low 64 KB have their bottom 10 bits compared to the ranges 3B0h – 3BBh and 3C0h – 3DFh. Accesses that miss all the enabled ranges are accepted for forwarding to HyperTransport, as long as the MasterEn bit in the Command CSR is set.

- **Configuration and Special Cycles.** The HyperTransport PCI bridge never acts as a target for configuration or special cycles on the PCI bus.

3.6.2 PCI Posted Write Queue

The HyperTransport PCI bridge responds as a PCI write target to PCI Memory Write, Memory Write Invalidate, and I/O Write commands. All of these writes are posted to the HyperTransport chain. The HyperTransport PCI bridge never responds to Configuration Writes. A total of 192 bytes of buffering for posted write data is provided.

Memory Write and Memory Write Invalidate commands stream data into the chip, disconnecting either on 4-KB boundaries or when all of the internal buffer space is filled. The HyperTransport PCI bridge generates the largest HyperTransport write operations possible, issuing them continuously as the data for each write is received from PCI.

As the bandwidth of HyperTransport exceeds the bandwidth of PCI, it is expected that the internal buffers will not fill and memory writes will proceed continuously at the full bandwidth of the PCI bus.

I/O writes are not allowed to stream and always disconnect after a single data beat on the PCI bus (32 bits). Each I/O write is issued to HyperTransport as an independent request.

3.6.3 PCI Delayed Request Buffers

The HyperTransport PCI bridge acts as a PCI read target for PCI Memory Read, Memory Read Line, Memory Read Multiple, and I/O Read commands. The HyperTransport PCI bridge never responds to configuration read or interrupt acknowledge accesses. All supported read transactions are implemented as PCI delayed reads.

Incoming reads are assigned to a delayed request buffer. There are four delayed request buffers, enabled under CSR control, allowing up to four PCI read requests to be in progress at one time. If no delayed request buffers are free, the incoming request is retried until one is available. Once the request is assigned to a buffer, the interface continues to retry it on the PCI bus while read requests are issued to the HyperTransport interface.

3.6.4 Prefetching

The HyperTransport PCI bridge supports a variety of prefetching options configured under CSR control using the Read Control CSR (62h:60h), however:

Notes

- I/O reads are never prefetchable.
- MemRdLines and MemRdMult may have prefetching individually configured.
- For systems in which MemRds are known to be side-effect free, MemReadPrefEn can be set to enable prefetchable behavior for MemReads using the same parameters as MemRdLines.
- PrefEn can be used to globally enable or disable all prefetching.
- Nonprefetchable reads always request only the bytes required to satisfy the initial data beat of four or eight bytes on the PCI bus, which may result in either one or two HyperTransport requests.

Transactions for which prefetching is enabled issue a HyperTransport read for the remainder of the 64-byte aligned block containing the original request. These transactions also issue HyperTransport reads for the zero to seven complete 64-byte blocks following, as determined by the Read Control CSR. The total number of reads that may be outstanding to HyperTransport at one time is limited by the Outbound Data Buffers.

When multiple reads to HyperTransport are issued for a single PCI read request due to prefetching or due to clear byte enables in a 64-bit nonprefetchable read on a 64-bit bus, each HyperTransport request is referred to as a subrequest of the PCI request. Each Delayed Request Buffer can track up to 8 subrequests at once. The total number of configured subrequests (number of enabled delayed request buffers * (the maximum number of subrequests each, rounded up to the next power of 2)) must not exceed the number of entries in the Outbound Data Buffers.

3.6.5 SrcTags

The SrcTag for each HyperTransport read request is formed by concatenating the delayed request buffer number with the number of the HyperTransport subrequest being issued by that buffer and adding a leading 0 (zero) bit.

- If three or four delayed request buffers are in use, a maximum of four reads may be outstanding for each one. The subrequest number is two bits. The delayed request buffer number is two bits.
- If only one or two delayed request buffers are in use, a 3-bit subrequest number is used. Only one bit of delayed request buffer number is needed and the top bit is dropped.

Either way, five bit srcTags are generated in the range of 00h – 0Fh.

3.6.6 Sequences

HyperTransport subrequests that are part of the same PCI request must be tagged with a matching nonzero seqId to guarantee ordering at the target. This 4-bit seqId is formed by concatenating a leading 1 (guaranteeing a nonzero result) with the 2-bit delayed request buffer number and one bit that toggles for each occupation of the delayed request buffer. The concatenation prevents consecutive PCI reads from being issued with the same seqId and appearing to have HyperTransport ordering requirements.

3.6.7 Read Responses

As the read responses return from HyperTransport, the data is stored in the Outbound Data Buffers. Even though sequenced requests are guaranteed to reach the target in order, responses may be received from the target out of order. When all data from the first HyperTransport requests is received (the amount required is controlled by the InitCount fields of the Read Control CSR), the PCI interface ceases retrying the request. Read data is supplied from the buffers when the request is next reissued. Data streams to the PCI bus until the transaction is disconnected by the PCI master or until the next data required is not present in the Outbound Data Buffers.

3.6.8 Continuous Prefetching

If continuous prefetching is enabled in the Read Control CSR, the HyperTransport PCI bridge issues further ascending read requests to fill buffers as they drain (up to a 4-KB page boundary) in an effort to make sure required data is always available. Otherwise, the transaction is disconnected as soon as all data from the initial reads is returned to the PCI bus.

Notes

3.6.9 Transaction Disconnects

Read transactions may be disconnected by the bridge when required data is not available in time, or by the master. The Delayed Request buffer remains in use until all outstanding HyperTransport prefetch requests receive their responses; then it is retired and any leftover data discarded. Each delayed request buffer also has an associated discard timer loaded with one of two values determined by the Secondary Discard Timer bit (9) of the Bridge Control CSR (3Eh) when the data is received from HyperTransport. If this timer expires before the data is called for by the PCI master, the data is discarded and the buffer retired.

3.6.10 Performance Variables

The following PCI inbound read performance characteristics can be tuned for best performance in a given system architecture or traffic load by using fields in the Read Control CSR. Independent prefetch controls are also provided for both MemRdMult and MemRdLine commands. If prefetching is enabled for MemRd commands, the MemRdLine values are used.

- **PCI Delayed Request.** The PCI Delayed Request field controls the number of PCI requests that can be fetching data at one time. Having multiple delayed requests generally reduces average latency on the PCI bus by fetching for multiple PCI requests simultaneously. However, setting the number of fetches too high may interfere with continuous prefetching.
- When the number of fetches is set too high, a large number of reads may be sent to HyperTransport due to other PCI reads between the initial fetches and the ones generated as the data buffers drain to PCI. If these subsequent fetches backup far enough, they may not succeed in returning data in time to keep the burst from ending on the PCI bus at which point their data might be discarded.
- **Ideal Prefetch Count.** If the available HyperTransport bandwidth exceeds the PCI bandwidth, there is an ideal prefetch count for large transfers. The ideal prefetch count is determined by dividing the number of bytes that can be transferred on the PCI bus in the round-trip read latency by 64 (the number of bytes in a prefetch request).
- The number of bytes that can be transferred on the PCI bus depends on the bus characteristics: 32 or 64 bit, 33 or 66 MHz, or 25 or 50 MHz.
- Round-trip read latency depends on the latency of and distance to the target.
- With continuous prefetching enabled, the HyperTransport PCI bridge will be able to keep up with an arbitrary length burst.
- **InitCount.** If the available HyperTransport bandwidth is less than the PCI bandwidth, there is no ideal prefetch count. HyperTransport will eventually be forced to disconnnect on a long burst because it will be out of data. Disconnects like this waste PCI bandwidth and waste HyperTransport bandwidth by discarding the prefetched data.
- In this case, setting the InitCount above zero will require more data to be on hand before the transaction is allowed to reconnect on PCI. The nonzero setting increases utilization of HyperTransport bandwidth at some cost in PCI latency, but reduces the number of wasteful disconnects. Continuous prefetching should also be disabled since the extra prefetches would probably not arrive in time to be used and would likely be discarded.

3.6.11 Outbound Data Buffer

- The HyperTransport PCI bridge contains a central data buffer (sixteen 64-byte entries) for the accumulation of read response data to return to the PCI bus. Entries in the buffer are assigned to PCI reads by the same algorithm used to assign srcTags—the bottom four bits of the srcTag are the entry number. Because of this one-to-one correspondence, no separate mechanism for allocation of data buffers is required. Data from returning HyperTransport responses is loaded into the buffer based on the srcTag in the response header and drained out to the PCI bus when the delayed request reconnects.

3.6.12 Interrupt Generation

The HyperTransport PCI bridge interrupt controller consists of four blocks of generic interrupts, four interrupts per block, and one set of special interrupts. In *Pass2*, the special interrupts are not available.

Notes

Each block, and the interrupts below that block, are configured by CSRs. Interrupt configuration options include: enable, edge versus level sensitivity, polarity, and vector ID.

An incoming interrupt is first synchronized to the core clock domain and masked with its CSR enable bit to set the interrupt's bid in the Interrupt Request Register (IRR). A round-robin arbiter will then examine each IRR bit and forward requests to the primary bus logic, setting its In Service Register (ISR) bit. New interrupts from that pin will not be accepted as long as the ISR register is set.

- For edge sensitive interrupts, the ISR bit is cleared as soon as the primary interface logic accepts the inbound interrupt request.
- For level interrupts, the ISR bit is not cleared until an End of Interrupt (EOI) packet is received with a vector ID matching the ISR.
- All interrupts and interrupt blocks are disabled at reset and must be enabled by software.
- The ISR can also be monitored via the Control Status Register (CSR).

3.6.13 Interrupt Diagnostic Mode

To simplify debugging interrupt software and hardware, each interrupt may be stimulated and monitored via CSR reads and writes (see the Interrupt Diagnostic Register for CSR details). Writing a 1 to the Initiate field of the CSR will generate an interrupt on the interrupt line in the Pin Number field. The ISR bit may be observed by monitoring the Active field of the same CSR.

Note: The Active field is always for the CSR indicated by the Pin Number field.

3.7 PCI Arbiter

The HyperTransport PCI bridge includes a PCI arbiter. This arbiter is an independent unit. The HyperTransport PCI bridge internal PCI request and PCI grant signals connect to pins as P_REQ_OUT_N and P_GNT_IN_N. It is possible to either use this arbiter or to bypass it and use an external arbiter.

The HyperTransport PCI bridge arbiter contains two round-robin arbitration groups: P_REQ0_N and P_PREQ_N; and P_REQ1_N through P_REQ5_N. In *Pass2*, P_PREQ_N is not available.

Arbitration is shared equally between the two groups. Within each group, arbitration is shared equally between the requests. Generally, P_REQ_OUT_N would be attached to P_REQ0_N and P_GNT_IN_N attached to P_GNT0_N, because this causes the arbiter to share equally between inbound and outbound PCI traffic.

When there are no requests present, the arbiter either parks the bus at the last grant or (based on a CSR bit) at P_GNT0_N which is presumed to be the HyperTransport PCI bridge internal requester.

The PCI Control CSR ParkMaster bit allows configuration of the internal PCI Arbiter parking.

3.8 Reset

3.8.1 Cold Reset

Cold reset of the HyperTransport PCI bridge is caused by the deassertion of the L_POWER_OK pin. At power-on, L_POWER_OK must remain deasserted until power and clocks are stable for at least 1 ms. L_RST_N must be asserted before L_POWER_OK deasserts and remains asserted for at least 1 ms following.

Cold reset results in the initialization of all internal state, including the loading of internal registers from strapped PCI bus pins and the SRI (Serial ROM Interface). The PCI bus is held in reset until the deassertion of L_RST_N.

3.8.2 Warm Reset

Warm reset of the HyperTransport PCI bridge is caused by the assertion of L_RST_N while leaving L_POWER_OK asserted. Once asserted, L_RST_N must remain asserted for at least 1 μs.

Notes

Warm reset results in the initialization of most internal state, with the exception of state loaded from PCI bus sampling or the external SROM interface, and persistent error state. The PCI bus is held in reset until the deassertion of L_RST_N.

3.8.3 Serial ROM Initialization

There are 224 bits loaded from a Serial Read Only Memory (SROM). The SRI (Serial ROM Interface) allows loading of configuration information into the HyperTransport PCI chip during reset.

On the rising edge of L_POWER_OK, the state of P_AD[1] is observed. If it is high, configuration information is loaded into the HyperTransport PCI chip from an external SROM.

Once L_POWER_OK is asserted, the SRI state machine begins to drive SROM_SCK to the external SROM. The state machine expects data to be valid by the falling edge of SROM_SCK. The state machine runs, shifting in data until all configuration information is read.

If P_AD[1] is low during the assertion of L_POWER_OK, internal SIP values are used for all configuration bits and no data is loaded.

3.8.4 Reset Configuration

In addition to loading configuration information from the SIP, information may also be loaded into the HyperTransport PCI bridge at reset from the PCI AD bus, P_AD[63:0].

Signal	I/O	Definition
minRstCnt	P_AD[31]	Reserved for API NetWorks. Must be tied to 0.
dbgSelCtl	P_AD[30]	Use SROM or CSR for dbg selects. 0 = SROM 1 = CSR
L0_clkSel	P_AD[29:27]	Link0 Clock Divider Select
L1_clkSel	P_AD[26:24]	Link1 Clock Divider Select
coreSel	P_AD[23:22]	Core Clock Divider Select
sri0_LdtSyncPtrCtlDflt	P_AD[19]	Link0 Sync Pointer Control values used only when external SROM is not being loaded. 1=Sync 0=Async
sri1_LdtSyncPtrCtlDflt	P_AD[18]	Link1 Sync Pointer Control values used only when external SROM is not being loaded. 1=Sync 0=Async
monitorMode	P_AD[17]	1=Enable monitor mode 0=Normal mode
monModeLink0Sel	P_AD[16]	0=Link1 1=Link0
debugEnable	P_AD[15]	Reserved for API Networks. Must be tied to 0.
pci64	P_AD[14]	PCI bus data width: 0 = 64 bit 1 = 32 bit
intDbgSrcSel	P_AD[13]	Reserved for API Networks. Must be tied to 0.
intDbgSel	P_AD[12:10]	Reserved for API Networks. Must be tied to 0.
intDbgEn	P_AD[9]	Reserved for API Networks. Must be tied to 0.

Table 3.1 Reset Configuration Strappings

Notes

Signal	I/O	Definition
enableSROM	P_AD[1]	Enable SROM: 1 = Load SIP configuration 0 = Load configuration from internal SIP
sel3x	P_AD[0]	PLL loop gain adjust. Set to 0.
sel9x	P_AD[63]	PLL loop gain adjust. Set to 1.

Table 3.1 Reset Configuration Strappings <Emphasis> (Continued)

3.8.5 HyperTransport Link Initialization

On the deassertion of L_RST_N as part of a warm or cold reset sequence, the HyperTransport PCI bridge attempts to initialize both of its HyperTransport links as described in the HyperTransport specification.

When link initialization is complete, the InitDone CSR bit is set. Software polls this CSR bit to determine that the link is live and may be included in fabric initialization. InitDone remains clear if the HyperTransport PCI bridge is unable to initialize the link because of any of the following:

- There is no device at the other end.
- Communication with the device at the other end is not possible.
- The link is disabled because of a previous failure.

3.8.6 HyperTransport Fabric Initialization

Once hardware initialization of the individual links in the HyperTransport chain completes, host software is responsible for initializing the HyperTransport fabric. A sample initialization sequence proceeds according to the following example. If starting from the host, initialization proceeds recursively for each link in the chain.

Example initialization sequence:

1. Read the InitDone bit for the link to determine whether the link is live. Also, read the various link error bits to determine if the link has taken errors since reset that prevent it from functioning correctly. If the link is not live, or is taking fatal errors, fabric sizing is complete and you can proceed to Step 6. All devices assume a HyperTransport unitId of 0 at reset. Therefore, a configuration access to PCI device number 0 is accepted by the first uninitialized device on the chain. This is the device at the far end of the link currently being sized.

   Performing a write to the HyperTransport Command register, without changing any fields, causes initialization of the master host bit. This indicates the HyperTransport link that connects toward the host bridge. Polling the error bits for that link determines whether the node on the far end is detecting any fatal errors on the link. If so, this link is not to be used, fabric sizing is complete, and you can proceed to Step 5.
2. Software reads the Class Code, Vendor ID, and Device ID from the device at the end of the current link to determine what type of device it is talking to.
3. Software writes the BaseUnitID register in the device with the next free HyperTransport unitID value, starting with 1 for the first device. It reads the unitCount register to determine how many unitID values the current device requires and increments the next free unitID value appropriately.
4. Return to step 1 to size the next link in the chain. Because step 3 wrote the base unitID of the last sized device to a nonzero value, it no longer accepts accesses to device 0. Instead, it forwards them to the next device in the chain.
5. When sizing completes to the last live link in the chain, set the EndOfChain and TransmitOff bits for the outgoing link of the last device. This prevents the unused link from being driven and enables proper handling of HyperTransport packets that traverse the HyperTransport link without finding their target node.

3.8.7 Secondary Bus Reset

The PCI bus may be placed and held in reset while the HyperTransport interface remains live. When the reset pin on the PCI bus (AP_RST_N) is asserted, all internal PCI buffers are flushed. Inbound writes that are in progress during the reset may be completed, depending on how far into the pipeline they are. Inbound reads are retired as their responses return from HyperTransport and the data dropped. Outbound operations

Notes

to PCI are dropped and error status is maintained and returned as if the operations had master aborted on the PCI bus. Software is responsible for coping with any transfers that were interrupted or dropped as a result of the reset. The interrupt controller is not affected by secondary bus reset.

Note: In Pass 1, only the Tsi301 can initiate the PCI bus reset. In Pass 2, an external device can also initiate the PCI bus reset.
In Pass1, the PCI bus is placed and held in reset under CSR control. In Pass 2, the PCI bus can also be placed and held in reset by an external device.

3.9 Error Handling

The HyperTransport PCI bridge provides a variety of error checking, logging, and containment functions to ensure correct operation and diagnose failures.

3.9.1 Reporting

The HyperTransport PCI bridge logs all errors it detects in CSRs that are persistent through a warm reset. CSR enables are used to mask and control routing of error notification. Not all signaling methods are available for all error types.

When the HyperTransport PCI bridge takes an error, it signals the system in one of three ways. The error-signaling methods are listed below in order of increasing severity:

- For errors detected by the HyperTransport PCI bridge as a transaction target, errors may be signaled in the bridge chip response. The method of signaling in the response depends on the protocol of the bus on which the error is detected.
  - *Transaction errors on nonposted HyperTransport requests may be indicated by an Error response.*
  - *Transaction errors on PCI may be indicated by a target abort, PERR# assertion, or premature disconnection before the data in error is transferred.*

  In each of these cases, the protocol on the given bus continues to run, and it is the requester's responsibility to take appropriate action on receipt of the error. Transmission of HyperTransport error responses (without NXA) sets the SigdTgtAbort bit in the Status CSR. Transmission of a target abort on PCI sets the SigdTgtAbort bit in the Secondary Bus Status CSR.
- Errors may be signaled to the system by the error interrupt pins. Two pins, FATAL_ERR_N and NONFATAL_ERR_N, allow division of errors into two different priority classes.

  These pins can be directly connected to input pins on the HyperTransport PCI bridge interrupt controller or to an external interrupt controller. They are active-low, open drain outputs, which allows the pins to be wire-ORed with other interrupt sources. They also provide edge-triggered interrupts, pulsing when an error is detected. The SerrEn bit in the Command CSR serves as a master enable for the error interrupts, in addition to the enables for individual error conditions. Assertion of either error interrupt sets the SigdSerr bit in the Status CSR.
- The HyperTransport transmitters can flood the link with synchronization packets. These propagate along the length of the chain and are detected by the host bridge. The host bridge is then responsible for taking appropriate action. The link has to pass through a warm reset sequence before it can be re-enabled, and all transactions in progress are lost. This option is only available for catastrophic HyperTransport errors that render the chain untrustworthy. Any error that causes sync flooding also sets the LinkFail bit in the HyperTransport Link 0/1 Control CSR for the link on which the error was detected. The setting of LinkFail will cause that link to not be re-initialized on the next warm reset event. The SigdSerr bit in the Status CSR is also set.

3.10 HyperTransport Errors

Link Errors — HyperTransport link errors are detectable at the lowest levels of the HyperTransport protocol and indicate a basic failing of the HyperTransport link. These errors are not localizable to individual transactions, and all can bring down the link through sync flooding.

Notes

CRC is checked by all of the HyperTransport link receivers in accordance with the HyperTransport specification. Basic protocol checks are performed for proper switching of the CTL signal and legal command encodings. An overflow error is detected if the HyperTransport flow control mechanism breaks down and packets are received with no space for them in the Rx buffers.

Table 3.2 indicates the CSR bits used to log and enable reporting of each HyperTransport link error type.

Error	Log Bit	Sync Flood	Fatal Interrupt	NonFatal Interrupt
Bad CRC	LinkCtrl/ CrcErr[0]	LinkCtrl/ CrcSyncFloodEn	ErrCtrl/ CrcFatalEn	ErrCtrl/ CrcNonFatalEn
Protocol Error	LinkCtrl/ ProtErr	ErrCtrl/ ProtSyncFloodEn	ErrCtrl/ ProtFatalEn	ErrCtrl/ ProtNonFatalEn
Rx Overflow	LinkCtrl/ OvfErr	ErrCtrl/ OvfSyncFloodEn	ErrCtrl/ OvfFatalEn	ErrCtrl/ OvfNonFatalEn

Table 3.2 HyperTransport Link Error CSR Bits

In addition to these HyperTransport link errors, the HyperTransport PCI bridge also propagates sync packets out of its transmitters if sync flooding is detected on either receiver. This is not logged or reported. This condition represents propagation of an error report from another device, not error detection by the HyperTransport PCI bridge.

Transmission Errors — End of Chain (EOC) accesses occur when the HyperTransport PCI bridge tries to transmit a packet from PCI, or the other HyperTransport link, on a link that is not live (there is nowhere to send the packet).

- If the outgoing packet was a nonposted request, the HyperTransport PCI chip generates a matching response with the error and NXA bits set and sends the response back to the requester. This is equivalent to a master abort on PCI and requires no logging or further action by the HyperTransport PCI chip.
- If the outgoing packet was a broadcast, it is silently dropped (it has traversed the whole chain).
- If the outgoing packet was not a nonposted request (either posted request or response) or a broadcast, then there is no in-band way to signal the error. The packet is dropped and may be signaled as an error, as shown in Table 3.3.

Error	Log Bit	Sync Flood	Fatal Interrupt	NonFatal Interrupt
End of Chain Error	LinkCtrl/ NxaErr	ErrCtrl/ NxaSyncFloodEn	ErrCtrl/ NxaFatalEn	ErrCtrl/ NxaNonFatalEn

Table 3.3 HyperTransport Forwarding Error CSR Bits

Master Errors — The HyperTransport PCI bridge accepts received responses that match its unitId and compares the response srcTags to the bridge's outstanding request srcTags. When the response does not match an outstanding request, it is called a Response Match error.

If the response does match an outstanding request, it is routed back to the PCI bus. The response may contain an error assertion, indicated by the Error bit. In this case, the NXA bit indicates whether the error was caused by the access failing to reach a target (value of 1) or signaled by the target (value of 0).

- An NXA bit value of 1 is roughly equivalent to a PCI master abort.
- An NXA bit value of 0 is equivalent to a PCI target abort.

In general, these errors are signaled to the PCI bus in the same way as in a standard PCI-PCI bridge. HyperTransport PCI bridge HyperTransport master error settings are described in Table 3.4.

Notes

Because the HyperTransport PCI bridge prefetches read data, it is possible that the error occurred on a location that the PCI device did not intend to access. If error responses are received for prefetch requests, the associated data is dropped. The prefetching stops at the point of the error, and the PCI transaction is disconnected when it reaches that point. If the PCI device then requests the data again, the request goes back to HyperTransport. If the error recurs, the response error is passed to PCI.

Once a response reaches the delayed request buffers, it must wait there until the requesting PCI master reconnects. As soon as the initial HyperTransport request completes and places its data in the buffer, a timer starts. The timer may be initialized to one of two values, as configured by the SecDiscardTimer bit in the Bridge Control CSR. If the timer expires before the data is called for, that may be treated as an error.

Error	Log Bit	Fatal Interrupt	NonFatal Interrupt	PCI
Response Match Error	Error/ RespMatchErr	Error/ RespMatchFatalEn	Error/ RespMatch NonFatalEn	No action
Error without NXA	Status/ RcvdTgtAbort	Not supported		Target Abort
Error with NXA	Status/ RcvdMstrAbort			If BrCtrl/ MstrAbortMode = 1, Target Abort; Else, complete normally returning all 1s data
Discard Timeout	BrCtrl/ DiscardStat	BrCtrl/ DiscardSerrEn & Error/ DiscardSerrFatal	BrCtrl/ DiscardSerrEn & !Error/ DiscardSerrFatal	Data dropped; PCI master must re-request.

Table 3.4 HyperTransport Master Errors CSR Bits

Slave Errors — The HyperTransport PCI bridge CSR master only supports accesses within a 32-bit aligned block. Accesses that span more than one 32-bit block receive HyperTransport error responses, equivalent to a PCI target abort. No other action is taken.

Error responses may also be signaled to HyperTransport because of errors taken when the request was issued to PCI.

3.11 PCI Errors

3.11.1 PCI System Errors

PCI devices may assert an unrecoverable system error by asserting SERR# on the secondary PCI bus. Settings for this error are described in Table 3.5.

Error	Log Bit	Fatal Int	NonFatal Int
SERR# Assertion	SecStatus/ DetSerr	BrCtrl/SerrEn & ErrCtrl/ SerrFatalEn	BrCtrl/SerrEn & !ErrCtrl/ SerrFatalEn

Table 3.5 PCI System Error CSR Bits

3.11.2 PCI Master Errors

PCI master errors refers to errors detected by the HyperTransport PCI bridge when acting as a master on the PCI bus. Master and Target Abort are defined in the *PCI Local Bus Specification, Revision 2.2*.

Notes

TRDY# timeout refers to a violation of the target latency requirements, as given by the Error/TrdyTimer CSR. Retry timeout refers to an excessive number of retries and/or disconnects, as given by the Error/Retry-Timer CSR.

All PCI requests issued are forwarded through the HyperTransport PCI bridge from HyperTransport. If the HyperTransport request was nonposted, error status may be returned to the HyperTransport requester in the response. If it was posted, the error may only be signaled by the error interrupts. A single set of error-reporting controls is used for all posted requests, regardless of the specific error taken.

Error	Log Bit	Nonposted	Posted
Master Abort	SecStatus/ RcvdMstrAbort	If BrCtrl/ MstrAbortMode = 1, return Error response	If Error/PostFatalEn = 1, assert FATAL_ERR_N. If Error/ PostNonFatalEn = 1, assert NONFATAL_ERR_N.
Target Abort	SecStatus/ RcvdTgtAbort	Return Error response	
TRDY# Timeout	Error/TrdyTimeout	Return Error response	
Retry Timeout	Error/RetryTimeout	Return Error response	

Table 3.6 PCI Master Errors CSR Bits

All of the above errors return all 1s data for read requests, whether or not the response Error bit is set.

3.11.3 PCI Parity Errors

All PCI devices are required to drive even parity on P_PAR when they are driving the bottom half of the P_AD bus, and on P_PAR64 when they are driving the top half.

The HyperTransport PCI bridge checks parity on the P_AD bus during command/address phases and data phases when it receives data. The HyperTransport PCI bridge then logs bad parity in the DetParErr bit of the Secondary Bus Status CSR. Other action is taken only if enabled by the ParErrRespEn bit in the Bridge Control CSR. The action taken depends on the type of information being transferred at the time of the error and in which direction the transfer was occurring.

Table 3.7 indicates the CSR bits used to log and enable reporting of each PCI parity error.

Error In	Fatal Interrupt	NonFatal Interrupt	PCI
Command/ Address	Error/ CmdPerrFatalEn	Error/ CmdPerrNonFatalEn	If decode has caused the HyperTransport PCI bridge to drive P_DEVSEL_N, Target Abort
Write data to HyperTransport PCI	Not Supported		Assert P_PERR_N
Read data to HyperTransport PCI			Set SecStatus/ MstrDParErr, return HyperTransport error response, and assert P_PERR_N

Table 3.7 PCI Parity Errors CSR Bits

The HyperTransport PCI bridge may also sample P_PERR_N, asserted when it is driving write data out, indicating that a parity error was detected by the target of the write. If the ParErrRespEn bit is set and the request was a nonposted write, it receives an error response. If the request was a posted write and the Post-FatalEn or NonPostFatalEn bits in the Error Control CSR are enabled, the error is signaled by one of the error interrupts.

3.12 Test Features

Several features are included in the HyperTransport PCI bridge to facilitate testing of the chip. Supported test modes are listed in Table 3.7 and are described in the following sections.

Notes

Signal Setting	Test Mode
01xxx0	Normal mode—Functional mode
11xxxx	JTAG mode—JTAG test mode
0000xx	Tristate mode—Tristate all outputs
0010xx	Parametric nandtree mode—Tristate all outputs except for nandtree outputs
0001xx	Scan chain output mode—Tristate all outputs except for scan chain outputs
0011xx	Powerdown inputs mode—Tristate all outputs, powerdown inputs
0xxx11	Internal scan mode—Tester bypass mode, enable scan shifting. This setting is used while scanning in/out of the internal scan chains
0xxx01	Tester bypass mode—Disable scan. This setting is used for the internal scan capture cycles

Table 3.8 HyperTransport PCI Bridge Test Modes

Note: Pin order for signal setting—
TRST_N, TRISTATE_N, TMODE[3], TMODE[2], TMODE[1], TMODE[0]

Note: Not all of the above conditions are exclusive. For example, the HyperTransport PCI bridge can be in internal scan mode and in scan chain output mode simultaneously.

3.12.1 JTAG Test Mode

In *Pass2*, the HyperTransport PCI bridge implements JTAG in accordance with IEEE Standard 1149.1. A BSDL description of the JTAG chain is available from API NetWorks.

3.12.2 Tristate

All HyperTransport PCI bridge outputs can be put in a tristate mode to put the chip into a quiescent state. There are three groups of outputs:

- scan chain
- nandtree
- other

Various combinations of these three groups can be tristated. See Table 3.8 for details.

3.12.3 Parametric Nandtree

On the HyperTransport PCI bridge, inputs and the input side of bidirectional I/Os go to parametric nandtrees which allow parametric testing of V_{il} and V_{ih} as well as some AC timing measurements. Refer to Table 3.8 for information about enabling the parametric nandtree test mode.

There are three different nandtrees:

- HyperTransport I/Os—the output is DBGOUT2
- PCI I/Os—the output is P_REQ_OUT_N
- Any other signals—the output is DBGCLK

A version of the HyperTransport nandtree output that provides a pulse every time the HyperTransport nandtree output changes is output on TDO. This allows some characterization of the speed of the chip.

Note: In Pass2, there is only one nandtree and its output is DBGCLK. There is also a version of the nandtree on DBGOUT[16] which pulses once for every nandtree output transition. DBGCLK is reserved for use by API NetWorks.

The following inputs/bidirects are not covered by the nandtree test:

- TMODE3
- TMODE2

Notes

- TMODE1
- TMODE0
- TRST_N
- TRISTATE_N
- AP_TYPEDET_N

3.12.4 Power-Down Inputs

It is possible to power-down comparator-type input cells to put the chip in a quiescent low-power mode. See Table 3.8 for details on how to enable this mode.

Inputs that are powered-down in power-down mode are the PCI I/O cells and the clock input (REFCLK_H/L). If the HyperTransport PCI bridge recognizes that it is only a 32-bit PCI bus when it comes out of reset, then the PCI signals used only in 64-bit mode (P_AD[63:32], P_CBE_N[7:4], P_PAR64, P_REQ64_N, and P_ACK64_N) are also powered-down while the rest of the PCI I/O are enabled.

3.12.5 Internal Scan

There are 16 internal scan chains. When the HyperTransport PCI bridge is in internal scan mode, all clock signals clock at the same frequency. Sixteen interrupt pins are used as the inputs to the scan chains, and 16 PCI AD pins are used as the outputs of the scan chains. See Table 3.8 for details on how to enable internal scan mode.

Scan outputs are provided on P_AD[24:9], and scan inputs are given on the BLK3_IRQ[3:0], BLK2_IRQ[3:0], BLK1_IRQ[3:0], BLK0_IRQ[3:0] signals.

> **Note:** In Pass2, scan outputs are on DBGOUT[19:0]. DBGOUT[19:0] are reserved for use by API NetWorks.

3.12.6 Tester Bypass

During scan testing, the HyperTransport PCI bridge should run from a common frequency. In tester bypass mode, internal clocks are muxed in from external pins, which the tester then drives to a common frequency. Asynchronous resets are also muxed in from an external pin so that no elements on the internal scan chains are asynchronously reset during the scan.

3.13 Calibration Circuit

Internal logic continually recalculates and assigns impedance values. The internal logic generated values may be overwritten by CSRs. See the CSR impedance section for CSR configuration of HyperTransport impedance values.

Notes

Chapter 4 Clock and Timing Relationships

Notes

The HyperTransport PCI bridge has only one PLL. The PCI clock (REFCLK_H/L input) is 1/4 of the core clock or 1/2 of the core clock. See Table 4.1 for core clock frequencies.

4.1 Clock Dividers

The PLL has three dividers for generating divided VCO clocks. The divider selects come from the P_AD bus during reset as follows:

- Link0 Clock Divider Select
  - *L0_clkSel* *P_AD[29:27]*
- Link1 Clock Divider Select
  - *L1_clkSel* *P_AD[26:24]*
- Core Clock Divider Select
  - *coreClkSel* *P_AD[23:22]*

coreClk MHz	Lx_TX_CLK_H/L MHz	REFCLK_H/L MHz	coreClkSel	Lx_clkSel	P_M66EN
100	200	25	01	001	0
100	200	50	01	001	1
100	400	25	01	000	0
100	400	50	01	000	1
133	200	33	00	001	0
133	200	66	00	001	1
133	400	33	00	000	0
133	400	66	00	000	1

Table 4.1 Clock Settings

Note: The frequencies given in Table 4-1 are the HyperTransport frequency per wire. The actual HyperTransport data rate is 2 times the rate of this frequency.

4.2 Clock Features

4.2.1 Definitions

Clock	Description	Max Speed	Pin
L0_RxWlnkClk	Link0, 1/2 LDT Rx Clock (internal)	200 MHz	TCK
L0_TxWlnkClk	Link0, 1/2 LDT Tx Clock (internal)	200 MHz	TCK
L1_RxWlnkClk	Link1, 1/2 LDT Rx Clock (internal)	200 MHz	TCK
L1_TxWlnkClk	Link1, 1/2 LDT Tx Clock (Internal)	200 MHz	TCK
ref33Clk	Non-PLL 33MHz Ref (internal)	33 MHz	REFCLK_H
clk	Core Clk (internal)	133 MHz	REFCLK_H

Table 4.2 HyperTransport PCI Bridge Internal Clocks

Notes

Note: Test assumes tmode and/or bypass enable on core clock is active.

4.2.2 Resets

Clock	Description	Sync[1]	Pin
coldReset_N	L_POWER_OK delayed by 0.5 ms[2]	clk	L_TSTRST_N
warmReset_N	Derived from L_RST_N	clk	L_TSTRST_N
rstClkDiv_N	Derived from L_POWER_OK	refclk	L_TSTRST_N
ldtRxWlnkRst_N	Synchronized warmReset_N	ldtRxWlnkClk	L_TSTRST_N
ldtTxWlnkRst_N	Synchronized coldReset_N[3]	ldtTxWlnkClk	L_TSTRST_N
ldtRxLoadPtrRst_N	Derived from ldtRxSync	ldtRxWlnkClk	L_TSTRST_N
ldtRxUnloadPtrRst_N	Link Synchronized ldtRxSync	clk	L_TSTRST_N

Table 4.3 HyperTransport PCI Bridge Internal Resets

Note: 1.All resets are asynchronously asserted and synchronously deasserted to the specified clock.
2.ColdReset is delayed so that the PLL has time to lock before reset is released.
3.ColdReset is sent to the PHY and synchronized to the link data clock (clock from the PLL).
The result is ldtTxWlnkRst_N and is used to reset some link data clock logic as well as the WlnkClk.

4.3 Performance Information

The following clocks are referenced in this section:

LDT Rx Clock = Lx_RX_CLK_H/L clock period (twice the HyperTransport bit time)

LDT Tx Clock = Lx_TX_CLK_H/L clock period (twice the HyperTransport bit time)

Core Clock = Period of the clock used for HyperTransport PCI Bridge core

Times are given from the first piece of the packet presented on one interface to the first piece of the packet being presented on the next interface. For the HyperTransport interface, this is the first byte of the packet on the link.

4.3.1 Inbound Operations

Latency through the HyperTransport PHY from the first byte of the packet on the link to the clock edge where the Rx Sync FIFO captures the doubleword of data = 3 * LDT Rx Clock + data alignment factor

Note: For an 8-byte header, the two 4-byte halves can be received aligned (so that they are received by the core in the same core clock cycle) or unaligned (split across two core clock cycles). For headers received aligned, there is an additional delay defined as the data alignment factor in the equation. This value can range from 0 to ((1*Core Clock) + (2*LDT Rx Clock)).

Minimum latency through Rx Sync FIFO = (2 * Core Clock) + clock alignment factor. Actual latency depends on SIP settings for Rx Sync FIFO.

Note: For the two clock domain crossings in the receive direction, there is a clock alignment factor in the calculation that can range from 0 to ((2 * Core Clocks) + (1 LDT Rx Clock)).

Latency from Rx Sync FIFO through the Rx buffers to the header presented at the HyperTransport receive interface = 2 * Core Clock

Notes

4.3.2 Outbound Operations

Latency from the header presented at the HyperTransport transmit interface through the packet generator to the Tx Sync FIFO = 2 * Core Clock

Minimum latency through the Tx Sync FIFO = 1 * Core Clock. Actual latency depends on SIP settings for the Tx Sync FIFO.

Latency through the HyperTransport PHY from the doubleword available at the output of the Tx Sync FIFO to the first byte of the packet on the link = 1.75 * LDT Tx Clock

4.3.3 Latency Through the HyperTransport PCI Bridge

Reference clock periods for the HyperTransport PCI bridge are:

- LDT Rx Clock at 400 MHz = 2.5 ns
- LDT Tx Clock at 400 MHz = 2.5 ns
- Core Clock at 133 MHz = 7.5 ns
- PCI Clock at 66 MHz = 15 ns

Forwarding Path Idle Latency

The formula for calculating the minimum forward path idle latency through the HyperTransport PCI bridge is

HyperTransport Rx PHY delay + Rx Sync FIFO delay + HyperTransport Link I/F delay + logic delay(forwarding) + HyperTransport Link I/F delay + Tx Sync FIFO delay + HyperTransport Tx PHY delay

The calculation is

(3 * LDT Rx Clock) + (2 * Core Clock) + (2 * Core Clock) + (0 * Core Clock) + (2 * Core Clock) + (1* Core Clock) + (1 * LDT Tx Clock)

= (3 * 2.5) + (7 * 7.5) + (1.75 *2.5)

= 7.5 + 52.5 + 4.4 = 64.4 ns

Data Alignment Factor (maximum) =

(1 * Core Clock) + (2 * LDT Rx Clock) = 11.5 ns

Clock Alignment Factor (maximim) =

(2 * Core Clock) + (1 * LDT Rx Clock) = 17 ns

Extraction Path Idle Latency

The formula for calculating the minimum extraction path idle latency through the HyperTransport PCI bridge is

HyperTransport Rx PHY delay + Rx Sync FIFO delay + HyperTransport PCI Link I/F delay + logic delay (extraction to PCI) + PCI I/F delay

The calculation is

(3 * LDT Rx Clock) + (2 * Core CLock) + (2 * Core Clock) + (3 * Core Clock) + (4 * PCI Clock)

= (3 * 2.5) + (7 * 7.5) + (4 * 15)

= 7.5 + 52.5 + 60 = 124 ns

Data Alignment Factor (maximum) =

(1 * Core Clock) + (2 * LDT Rx Clock) = 12.5 ns

Clock Alignment Factor (maximim) =

(2 * Core Clock) + (1 * LDT Rx Clock) = 17.5 ns

Notes

Insertion Path Idle Latency

The formula for calculating the minimum insertion path idle latency through the HyperTransport PCI bridge is

PCI I/F delay + Application logic delay (PCI insertion) + HyperTransport Link I/F delay + Tx Sync FIFO delay + HyperTransport Tx PHY delay

The calculation is

(4 * PCI Clock) + (3 * Core Clock) + (2 * Core Clock) + (2 * Core Clock) + (1.75 * LDT Tx Clock)

= (4 * 15) + (7 * 7.5) + (1.75 * 2.5)

= 60 + 52.5 + 4.4 = 116.9 ns

Note: Data Alignment Factor and Clock Alignment Factor do not apply.

Notes

Notes

Chapter 5 Configuration Registers

Notes

To select many of the many options available on the HyperTransport PCI bridge, you write to its configuration registers. Usually, these registers are programmed during system initialization and are not accessed during normal operation.

This section describes the mechanism used to access the HyperTransport PCI bridge configuration registers as well as the location and functional details of each register.

Configuration Mechanism

Configuration accesses are accepted from HyperTransport to the HyperTransport PCI bridge internal CSRs if they are Type 0 accesses with a device number equal to the value in the BaseUnitId field of the HyperTransport Command CSR.

5.1 Summary of Configuration Registers

Table 5.1 summarizes the HyperTransport PCI bridge configuration register offsets, devices, default values after reset, and access types.

5.1.1 Register Access Definitions

Access types are indicated as follows:

- R – ReadA read of this register returns the field.
- W – WriteA write of this register loads the value.
- T – ToggleA write of 1 to this field toggles the value.
- S – SetA write of 1 to this field sets the field.
- C – ClearA write of 1 to this field clears the field.

5.1.2 Register Access Rules

Several rules apply to HyperTransport PCI bridge CSR accesses.

- Reads to undefined fields return undefined data.
- Writes to reserved fields should only be done with previously read data.

Offset	Register Name	Reset	Page No.
01h–00h	Vendor ID (API NetWorks, Inc.)	14D9h	5-6
03h–02h	Device ID	0010h	-6
05h–04h	Command	0000h	-6
07h–06h	Status	0010h	-7
08h	Revision ID	See Note.	-8
0Bh–09h	Class Code	060400h	-8
0Ch	Cacheline	00h	-9
0Dh	Primary Latency Timer	00h	-9
0Eh	Header Type	01h	-9
0Fh	BIST	00h	-9
13h–10h	Base Address Register 0	00000000h	-9

Table 5.1 Function 0 Configuration Registers

Notes

Offset	Register Name	Reset	Page No.
17h–14h	Base Address Register 1	00000000h	-9
18h	Primary Bus Number	00h	-9
19h	Secondary Bus Number	00h	-10
1Ah	Subordinate Bus Number	00h	-10
1Bh	Secondary Latency Timer	00h	-10
1Ch	I/O Base Address	01h	-10
1Dh	I/O Limit Address	01h	-10
1Fh–1Eh	Secondary Bus Status	00A0h	-11
21h–20h	Memory Range Base Address	0000h	-11
23h–22h	Memory Range Limit Address	0000h	-12
25h–24h	Prefetchable Memory Range Base Address	0001h	-12
27h–26h	Prefetchable Memory Range Limit Address	0001h	-12
2Bh–28h	Prefetchable Memory Range Base Upper 32 Bits	00000000h	-12
2Fh–2Ch	Prefetchable Memory Range Limit Upper 32 Bits	00000000h	-13
31h–30h	I/O Range Base Upper 16 Bits	0000h	-14
33h–32h	I/O Range Limit Upper 16 Bits	0000h	-14
34h	Capability 1	40h	-14
3Bh–38h	Expansion ROM	00000000h	-14
3Ch	Interrupt Line	FFh	-14
3Dh	Interrupt Pin	00h	-14
3Fh–3Eh	Bridge Control	0000h	-15
40h	HyperTransport Capability ID	08h	-16
41h	Capability 2	00h	-16
43h–42h	HyperTransport Command	0020h	-17
45h–44h	HyperTransport Link 0 Control	0000h	-17
47h–46h	Link 0 Width Control	0000h	-18
49h–48h	HyperTransport Link 1 Control	0000h	-19
4Bh–4Ah	Link 1 Width Control	0000h	-20
4Ch	HyperTransport Revision ID	11h	-20
4Dh	Link Frequency	-	5-21
62h–60h	Read Control	000000h	-22
63h	PCI Control	00Fh	-23
67h–64h	Error Control	00008080h	-23
68h–69h	HyperTransport Error Control	0000h	-24
6Dh-6Ch	HyperTransport Rx Data BufferAllocation	1515h	-26
6Eh	HyperTransport Transmit Control	04h	-26
6Fh	PCI Control 2	0Dh	-26

Table 5.1 Function 0 Configuration Registers<Emphasis> (Continued)

Notes

Offset	Register Name	Reset	Page No.
73h–70h	Link Impedance Control	00000000h	-27
77h–74h	Link Impedance Control 1	00000000h	-27
83h–80h	Serial ROM (SROM) Interface 0	From SROM	-28
87h–84h	Serial ROM (SROM) Interface 0 Rx	From SROM	-29
8Bh–88h	Serial ROM (SROM) Interface 0 Tx	From SROM	-29
8Fh–8Ch	Serial ROM (SROM) Interface 1	From SROM	-29
93h–90h	Serial ROM (SROM) Interface 1 Rx	From SROM	-29
97h–94h	Serial ROM (SROM) Interface 1 Tx	From SROM	-29
9Bh–98h	Serial ROM (SROM) Interface Control	From SROM	-30
BFh–A0h	Interrupt Controllers	-	-30
C1h–C0h	Special Interrupts	FEDCh	-31
C2h	Interrupt Diagnostics	00h	-31
C3h	Interrupt Block Level 0	00h	-32
C4h	Interrupt Block Level 1	00h	-32
C5h	Interrupt Block Level 2	00h	-33
C6h	Interrupt Block Level 3	00h	-33
C7h	Interrupt Special Block	00h	-33
CAh-C8h	HyperTransport Transmit Buffer Control Limit 0	FFFFFFh	-33
CEh-CCh	HyperTransport Transmit Buffer Control Limit 1	FFFFFFh	-34
D7h–D4h	Debug	03CB089Fh	-34
D8h	HyperTransport Diagnostics	10h	-35
DFh–DCh	Diagnostics Link 0 CRC Expected	00000000h	-35
F0h–F3h	Diagnostics Link 0 CRC Received	00000000h	-36
F7h–F4h	Diagnostics Link 1 CRC Expected	00000000h	-36
FBh–F8h	Diagnostics Link 1 CRC Received	00000000h	-36
FFh-FCh	Scratch	00000000h	-36

Table 5.1 Function 0 Configuration Registers<Emphasis> (Continued)

Note: Revision ID changes for each device revision. Rev A= 00h, Rev B=01h, Rev C = 02h, Rev D = 03h, etc. Pass2 = 10h.

5.2 Processor-to-HyperTransport Bridge Registers

CSR space maps for the HyperTransport PCI bridge Function 0 are provided in Figure 5.1. Most CSRs are reset by either cold or warm reset. Some CSRs maintain their values through warm reset and are called "persistent" CSRs.

5.2.1 CSR Space Map

The following figure is a map of HyperTransport PCI bridge CSR space. For *Pass2*, Link Frequency is a new CSR.

Notes

31 24	23 16	15 8	7 0	
Device ID		Vendor ID		00h
Status		Command		04h
Class Code		Revision ID		08h
BIST	Header Type	Primary Latency Timer	Cacheline	0Ch
Base Address Register 0				10h
Base Address Register 1				14h
Secondary Latency Timer	Subordinate Bus Number	Secondary Bus Number	Primary Bus Number	18h
Secondary Bus Status	I/O Limit Addr	I/O Base Addr	1Ch	
Memory Range Limit Addr		Memory Range Base Addr		20h
Prefetchable Memory Range Limit Addr		Prefetchable Memory Range Base Addr		24h
Prefetchable Memory Range Base Upper 32 Bits				28h
Prefetchable Memory Range Limit Upper 32 Bits				2Ch
I/O Range Limit Upper 16 Bits		I/O Range Base Upper 16 Bits		30h
Reserved			Capability 1	34h
Expansion ROM				38h
Bridge Control		Interrupt Pin	Interrupt Line	3Ch
HyperTransport Command		Capability 2	HyperTransport Capability ID	40h
Link 0 Width Control		HyperTransport Link 0 Control		44h
Link 1 Width Control		HyperTransport Link 1 Control		48h
Reserved		Link Freq 1 / Link Freq 0	HyperTransport Revision ID	4Ch
Reserved				50h
Reserved				54h
Reserved				58h
Resereved				5Ch
PCI Control	Read Control			60h
Error Control				64h
Reserved		HyperTransport Error Control		68h
PCI Control 2	Transmit Control	HyperTransport Rx Data Buffer Allocation		6Ch
Link Impedance Control 0				70h
Link Impedance Control 1				74h
Reserved				78h
Reserved				7Ch
Serial ROM (SROM) Interface 0				80h
Serial ROM (SROM) Interface 0 Rx				84h
Serial ROM (SROM) Interface 0 Tx				88h
Serial ROM (SROM) Interface 1				8Ch

Figure 5.1 CSR Function 0 Space Map

Notes

Serial ROM (SROM) Interface 1 Rx				90h
Serial ROM (SROM) Interface 1 Tx				94h
Serial ROM (SROM) Interface Control				98h
Reserved				9Ch
Block 0, Interrupt 1		Block 0, Interrupt 0		A0h
Block 0, Interrupt 3		Block 0, Interrupt 2		A4h
Block 1, Interrupt 1		Block 1, Interrupt 0		A8h
Block 1, Interrupt 3		Block 1, Interrupt 2		ACh
Block 2, Interrupt 1		Block 2, Interrupt 0		B0h
Block 2, Interrupt 3		Block 2, Interrupt 2		B4h
Block 3, Interrupt 1		Block 3, Interrupt 0		B8h
Block 3, Interrupt 3		Block 3, Interrupt 2		BCh
Interrupt Block Level 0	Interrupt Diagnostics	Special Interrupts		C0h
Interrupt Special Block	Interrupt Block Level 3	Interrupt Block Level 2	Interrupt Block Level 1	C4h
Reserved	Transmit Buffer Counter Maximum 0			C8h
Reserved	Transmit Buffer Counter Maximum 1			CCh
Reserved				D0h
Debug				D4h
Reserved Space			HyperTransport Diagnostics	D8h
Diagnostics Link 0 Receive CRC Expected				DCh
ECh - EFh are Reserved				
Diagnostics Link 0 Receive CRC Received				F0h
Diagnostics Link 1 Receive CRC Expected				F4h
Diagnostics Link 1 Receive CRC Received				F8h
Scratch				FCh

Figure 5.1 CSR Function 0 Space Map

Table 5.2 summarizes the addresses and Vector bit values for the 16 interrupt controller registers.

Interrupt Register	Address	Vector [1:0] Bit Default Value
Block 0, Interrupt 0	A1h–A0h	00b
Block 0, Interrupt 1	A3h–A2h	01b
Block 0, Interrupt 2	A5h–A4h	10b
Block 0, Interrupt 3	A7h–A6h	11b
Block 1, Interrupt 0	A9h–A8h	00b
BBlock 1, Interrupt 1	ABh–AAh	01b
Block 1, Interrupt 2	ADh–ACh	10b
Block 1, Interrupt 3	AFh–AEh	11b

Table 5.2 Interrupt Controller Addresses and Vectors

Notes

Interrupt Register	Address	Vector [1:0] Bit Default Value
Block 2, Interrupt 0	B1h–B0h	00b
Block 2, Interrupt 1	B3h–B2h	01b
Block 2, Interrupt 2	B5h–B4h	10b
Block 2, Interrupt 3	B7h–B6h	11b
Block 3, Interrupt 0	B9h–B8h	00b
Block 3, Interrupt 1	BBh–BAh	01b
Block 3, Interrupt 2	BDh–BCh	10b
Block 3, Interrupt 3	BFh–BEh	11b

Table 5.2 Interrupt Controller Addresses and Vectors<Emphasis> (Continued)

5.2.2 Registers

VendorID **Offset 01h-00h**

Bits	Type	Reset	Description
15:0	R	14D9h	Vendor ID - this value is defined as 14D9h for API NetWorks.

Device ID **Offset 03h–02h**

Bits	Type	Reset	Description
15:0	R	0010h	Id - this Device ID value of 0010h represents the HyperTransport PCI bridge.

Command **Offset 05h–04h**

Bits	Type	Reset	Description
15:10	R	0	Reserved (always reads 0).
9	R	0	FastB2BEn - this has no meaning for HyperTransport. Always reads 0.
8	RW	0	SerrEn - this enables system error interrupt pins FATAL_ERR_N and NONFATAL_ERR_N to be driven. 0 = SERR_N output driver disabled (default). 1 = SERR_N output driver enabled. Not persistent through warm reset.
7	R	0	WaitCycCtrl - this has no meaning for HyperTransport. Always reads 0.
6	R	0	ParErrRespEn - controls the bridge's response to parity errors on its primary interface. There are no parity errors on HyperTransport. Always reads 0.
5	R	0	VGAPalSnpEn - controls the bridge's response to VGA-compatible palette write accesses. If enabled, the bridge decodes VGA palette accesses (I/O 3C6, 3C8, and 3C9) as belonging on the secondary bus. The HyperTransport PCI bridge does not support VGA palette snooping. Always reads 0.

Notes

Command **Offset 05h–04h<Emphasis> (Continued)**

Bits	Type	Reset	Description
4	R Pass1 RW Pass2	0	MemWrInvEn - controls the ability of the bridge to generate Memory Write and Invalidate transactions as a master on the PCI bus; as controlled by the CacheLine Size register. **Note:** In Pass1 this bit always reads 0.
3	R	0	SpecCycEn - controls the bridge's ability to respond to special cycle operations. Bridges do not respond to special cycle operations. The bit is hardwired to 0.
2	RW	0	MasterEn - controls the bridge's ability to operate as a master on the primary interface when forwarding memory or I/O transactions from the secondary interface on behalf of a master on the secondary bus. If clear, the bridge will not respond to memory or I/O transactions on its secondary bus. 0 = HyperTransport PCI bridge not enabled to drive memory and I/O requests on HyperTransport. 1 = HyperTransport PCI bridge enabled to drive memory and I/O requests on HyperTransport. Not persistent through warm reset.
1	RW	0	MemSpaceEn - controls the bridge's response as a target to memory space accesses on the primary interface. If clear, the bridge will not accept any requests within the memory space (LDT 00_0000_0000 - FC_FFFF_FFFF) range. 0 = Disable memory space. 1 = Respond to memory space accesses. Not persistent through warm reset.
0	RW	0	IoSpaceEn - controls the bridge's response as a target to I/O space transactions on the primary interface. If clear, the bridge does not accept any requests within the I/O space (LDT FD_FC00_0000 - FD_FDFF_FFFF) range. 0 = Disable I/O space. 1 = Respond to I/O space accesses. Not persistent through warm reset.

Status **Offset 07h–06h**

Bits	Type	Reset	Description
15	R	0	Data Parity Error Detected - this bit reports the detection of an address or data parity error by the bridge on its primary interface. HyperTransport does not have parity errors. Always reads 0.
14	RC	0	SERR Signaled - this bit reports the assertion of a system error by a bridge on its primary interface; it may be cleared by writing a 1 to it. Persistent through warm reset. 0 = No error signaled. 1 = HyperTransport PCI bridge has asserted FATAL_ERR_N or NONFATAL_ERR_N or initiated sync flooding on the HyperTransport chain.
13	RC	0	Received Master Abort - this bit reports the detection of a master abort termination by the bridge, when it is the master of a transaction on its primary interface. This is indicated on HyperTransport by an NXA error response. It may be cleared by writing a 1 to it. Persistent through warm reset. 0 = HyperTransport PCI bridge has not received an NXA error response on HyperTransport. 1 = HyperTransport PCI bridge has received an NXA error response on HyperTransport.

Notes

Status **Offset 07h–06h<Emphasis> (Continued)**

Bits	Type	Reset	Description
12	RC	0	Received Target Abort - this bit reports the detection of a target abort by the bridge when it is the master of a transaction on its primary interface. This is indicated on HyperTransport by an error response without NXA. It may be cleared by writing a 1 to it. Persistent through warm reset. 0 = No error received. 1 = HyperTransport PCI bridge has received an HyperTransport error response.
11	RC	0	Signaled Target Abort - this bit reports the signaling of a target abort termination by the bridge, when it responds as the target of a transaction on its primary interface. This is indicated on HyperTransport by a response with the error bit set. It may be cleared by writing a 1 to it. Persistent through warm reset.
10:9	R	00b	DEVSEL_N Timing - Encodes the timing of the primary interface's DEVSEL. This is not meaningful for HyperTransport. Always reads 01. **Note:** In Pass1, DEVSEL_N Timing always reads 00.
8	R	0	PCI Parity Error Detected - this bit is used to report the detection of a parity error by the bridge when it is the master of the transaction. HyperTransport doesn't have parity errors.
7	R	0	Fast Back-to-Back Capability - this is not meaningful for HyperTransport. Always reads 0.
6	R	0	Reserved (always reads 0).
5	R	0	66 MHz Capable PCI Bus - indicates whether the primary interface is 66 MHz capable. Not meaningful for HyperTransport. Always reads 0.
4	R	1b	Capabilities List - this bit indicates that the configuration space of this device contains a capabilities list. Always reads 1.
3–0	R	0	Reserved (always reads 0).

Revision ID **Offset 08h**

Bits	Type	Reset	Description
7:4	R	[1]	Revision ID.
3:0	R	[1]	Shipping code.

[1] Reset values are revision dependent.

Class Code **Offset 0B–09h**

Bits	Type	Reset	Description
23:16	R	06h	Base class of the device. 06h indicates a bridge.
15:8	R	04h	Subclass of the device. 04h indicates a PCI bridge.
7:0	R	00h	Programming Interface of the device. 00 indicates a positive decode device.

Notes

CacheLine Size **Offset 0Ch**

Bits	Type	Reset	Description
7:0	RW	00h	Cache line size (in bytes) - must be a power of 2. This value is used to control generation of MemRdLine, MemRdMult, and MemWrInv commands by the PCI master when forwarding memory accesses from HyperTransport. If this value is left 0, the HyperTransport PCI bridge only generates MemRd and MemWr commands. For non-zero values, reads greater than 1 DW to prefetch-able space generate MemRdMults if they cross a cacheline boundary. Other-wise, MemRdLines are generated. If enabled by the MemWrInvEn bit of the command register, writes that write an entire cacheline (all byte enables asserted) generate MemWrInv commands. **Note:** In Pass1, CacheLine Size always reads 0.

Primary Latency Timer **Offset 0Dh**

Bits	Type	Reset	Description
7:0	R	00h	Primary Latency Timer - this is not used by the HyperTransport PCI bridge. Always reads 0.

Header Type **Offset 0Eh**

Bits	Type	Reset	Description
7:0	R	01h	Type - a value of 01h indicates that this is a bridge header.

BIST **Offset 0Fh**

Bits	Type	Reset	Description
7:0	R	00h	BIST - always reads 0.

Base Address Register 0 **Offset 13h–10h**

Bits	Type	Reset	Description
31:0	R	00000000h	Not used in the HyperTransport PCI bridge. Always reads 0.

Base Address Register 1 **Offset 17h–14h**

Bits	Type	Reset	Description
31:0	R	00000000h	Not used in the HyperTransport PCI bridge. Always reads 0.

Primary Bus Number **Offset 18h**

Bits	Type	Reset	Description
7:0	RW	00h	Primary Bus Number - PCI bus number of the HyperTransport chain on which the HyperTransport PCI bridge is located. Not persistent through warm reset.

Notes

Secondary Bus Number **Offset 19h**

Bits	Type	Reset	Description
7:0	RW	00h	Secondary Bus Number - PCI bus number of the bus which the HyperTransport PCI bridge sources. Not persistent through warm reset.

Subordinate Bus Number **Offset 1Ah**

Bits	Type	Reset	Description
7:0	RW	00h	Subordinate Bus Number - PCI bus number of the highest numbered bus behind the HyperTransport PCI bridge. Not persistent through warm reset.

Secondary Latency Timer **Offset 1Bh**

Bits	Type	Reset	Description
7:0	RW	00h	Timer - controls how long the HyperTransport PCI bridge may continue to occupy the PCI once the arbiter has taken the grant away in PCI clocks. The bottom two bits are hard wired to 0. Not persistent through warm reset.

I/O Base Address **Offset 1Ch**

Bits	Type	Reset	Description
7:4	RW	0h	Address - bits 15:12 of the base of the I/O range. 11:0 are assumed to be 0, leading to a 4 KB granularity. Not persistent through warm reset.
3:0	R	1h	Addressing Capability - indicates the size of I/O addresses supported by the device, indicates 32 bits.

I/O Limit Address **Offset 1Dh**

Bits	Type	Reset	Description
7:4	RW	0h	Address - bits 15:12 of the top of the I/O range. 11:0 are assumed to be 1, leading to a 4 KB granularity. Not persistent through warm reset.
3:0	R	1h	Capability - indicates the size of I/O addresses supported by the device, 32 bits.

Notes

Secondary Bus Status **Offset 1Fh–1Eh**

Bits	Type	Reset	Description
15	RC	0	Parity Error Detected - this bit reports the detection of an address or data parity error by the bridge on its secondary interface. It may be cleared by writing a 1 to it. Persistent through warm reset.
14	RC	0	Detect System Error - this bit reports the detection of a system error by a bridge on its secondary interface. It may be cleared by writing a 1 to it. Persistent through warm reset. 0 = No error detected. 1 = P_SERR_N assertion detected on the HyperTransport PCI bridge PCI bus.
13	RC	0	Received Master Abort - this bit reports the detection of a master abort termination by the bridge, when it is the master of a transaction on its secondary interface. It may be cleared by writing a 1 to it. Persistent through warm reset. 0 = No master abort detected. 1 = HyperTransport PCI bridge has detected a master abort.
12	RC	0	Received Target Abort - this bit reports the detection of a target abort by the bridge when it is the master of a transaction on its secondary interface. It may be cleared by writing a 1 to it. Persistent through warm reset. 0 = No target abort received. 1 = Transaction aborted by target.
11	RC	0	Signaled Target Abort - this bit reports the signaling of a target abort termination by the bridge, when it responds as the target of a transaction on its secondary interface. It may be cleared by writing a 1 to it. Persistent through warm reset.
10:9	R	01	DEVSEL_N Timing - encodes the timing of the secondary interface's DEVSEL. The HyperTransport PCI bridge PCI interface supports medium DEVSEL decoding. 00 = Fast 01 = Medium (the HyperTransport PCI bridge implements only this timing) 10 = Slow 11 = Reserved
8	RC	0	Master Detect Parity Error - this bit is used to report the detection of a parity error by the bridge when it is the master of the transaction. It is cleared by writing a 1 to the bit. Not persistent through warm reset. 0 = No parity error detected. 1 = Parity error detected.
7	R	1b	Fast Back-2-Back Capability - the HyperTransport PCI bridge supports fast back-to-back transactions on the PCI interface. Always reads 1.
6	R	0	Reserved (always reads 0).
5	R	1b	66 MHz Capable PCI Bus—Indicates whether the secondary interface is 66 MHz capable. The HyperTransport PCI bridge is 66 MHz capable.
4:0	R	00h	Reserved (always reads 0).

Memory Range Base Address **Offset 21h–20h**

Bits	Type	Reset	Description
15:4	RW	000h	Address - bits 31:20 of the base of the memory range. Bits 19:0 are assumed to be 0, leading to a 1 MB granularity. Not persistent through warm reset.

Notes

Memory Range Base Address **Offset 21h–20h<Emphasis> (Continued)**

Bits	Type	Reset	Description
3:0	R	0h	Reserved.

Memory Range Limit Address **Offset 23h–22h**

Bits	Type	Reset	Description
15:4	RW	000h	Address - bits 31:20 of the top (inclusive) of the memory range. Bits 19:0 are assumed to be 0, leading to a 1 MB granularity. Not persistent through warm reset.
3:0	R	0h	Reserved.

Prefetchable Memory Range Base Address **Offset 25h—24h**

Bits	Type	Reset	Description
15:4	RW	000h	Address - bits 31:20 of the base of the prefetchable memory range. Bits 19:0 are assumed to be 0, leading to a 1 MB granularity. Not persistent through warm reset.
3:0	R	1h	Cap - indicates whether the bridge supports 32- or 64-bit addressing for prefetchable memory space. The HyperTransport PCI bridge supports 64-bit addressing, indicated by an encoding of 1h.

Prefetchable Memory Range Limit Address **Offset 27h–26h**

Bits	Type	Reset	Description
15:4	RW	000h	Address - bits 31:20 of the top (inclusive) of the prefetchable memory range. Bits 19:0 are assumed to be 0, leading to a 1 MB granularity. Not persistent through warm reset.
3:0	R	1h	Cap - indicates whether the bridge supports 32- or 64-bit addressing for prefetchable memory space. The HyperTransport PCI bridge supports 64-bit addressing, indicated by an encoding of 1h.

Prefetch Mem Range Base Upper 32-Bit Address **Offset 2Bh–28h**

Bits	Type	Reset	Description
31:8	R	000000h	Reserved - the HyperTransport PCI bridge does not support decode of address bits above bit 39.
7:0	RW	00h	Address - bits 39:32 of the prefetchable memory range base. Memory accesses above 1012 GB will not be accepted regardless of the setting of this register. Not persistent through warm reset.

Notes

Prefetch Mem Range Limit Upper 32-Bit Address **Offset 2Fh–2Ch**

Bits	Type	Reset	Description
31:8	R	000000h	Reserved - the HyperTransport PCI bridge does not support decode of address bits above bit 39. Not persistent through warm reset.
7:0	RW	00h	Address - bits 39:32 of the prefetchable memory range limit. Memory accesses above 1012 GB will not be accepted regardless of the setting of this register. Not persistent through warm reset.

Notes

I/O Range Base Upper 16 Bits **Offset 31h–30h**

Bits	Type	Reset	Description
15:9	R	00h	Reserved - the HyperTransport PCI bridge does not support decode of address bits above 24.
8:0	RW	000h	Address - bits 24:16 of the I/O range base. Not persistent through warm reset.

I/O Range Limit Upper 16 Bits **Offset 33h–32h**

Bits	Type	Reset	Description
15:9	R	00h	Reserved - the HyperTransport PCI bridge does not support decode of address bits above 24.
8:0	RW	000h	Address - bits 24:16 of the I/O range limit. Not persistent through warm reset.

Capability 1 **Offset 34h**

Bits	Type	Reset	Description
7:0	R	40h	Pointer - register number of the base of the first capabilities block. There is only one capability block, which is for HyperTransport.

Expansion ROM **Offset 3Bh–38h**

Bits	Type	Reset	Description
31:0	R	00000000h	Reserved.

Interrupt Line **Offset 3Ch**

Bits	Type	Reset	Description
7:0	RW	FFh	Register - the HyperTransport spec requires that this be a read/write register. Its value is not used internally. Not persistent through warm reset.

Interrupt Pin **Offset 3Dh**

Bits	Type	Reset	Description
7:0	R	00h	Reserved.

Notes

Bridge Control **Offset 3Fh–3Eh**

Bits	Type	Reset	Description
15:12	R	0h	Reserved.
11	RW	0	DiscardSerrEn - if set, treat a discard timer error as a system error. It can cause the assertion of either the FATAL_ERR_N or NONFATAL_ERR_N interrupts, depending on the state of the DiscardSerrFatal bit of the Error Control register (see “Error Control Offset 67h–64h” on page -23). Not persistent through warm reset.
10	RC	0	DiscardStat - this bit is set by hardware when a request is dropped due to an expired discard counter. It may be cleared by writing a 1 to it. Persistent through warm reset.
9	RW	0	SecDiscardTimer - sets the length of the timer on delayed requests. Once the initial subrequests of an inbound delayed transaction have completed on HyperTransport, the timer begins to run. If it expires before the PCI requester reissues the request, the transaction is dropped from the delayed request buffers. Not persistent through warm reset. 0 = count 2^{15} PCI clocks 1 = count 2^{10} PCI clocks
8	R	0	PrimDiscardTimer - not meaningful for HyperTransport. Always reads 0.
7	RW	0	FastB2BEn - enables the generation of fast back-to-back transactions when the HyperTransport PCI bridge is the master on the PCI bus. Not persistent through warm reset.
6	RW	0	SecBusReset - if written to a 1, hardware will perform a reset sequence on the secondary bus. Clearing the bit will bring the secondary bus out of reset. Not persistent through warm reset.
5	RW	0	MstrAbortMode - this bit controls the action taken by the bridge when a transaction that it is forwarding in either direction takes a master abort on the destination bus. The master abort is indicated on HyperTransport by an error response with NXA set • If this bit is clear, writes are allowed to complete normally on the source bus, and reads have all 1's returned. • If it is set, the master abort will be treated as an error, returning a Target Abort Response (indicated on HyperTransport by a set error bit without NXA) for nonposted requests and causing a sideband error assertion (indicated by asserting the FATAL_ERR_N or NONFATAL_ERR_N interrupt pins, as enabled) for posted requests. Not persistent through warm reset.
4	R	0	Reserved.

Notes

Bridge Control **Offset 3Fh–3Eh<Emphasis> (Continued)**

Bits	Type	Reset	Description
3	RW	0	VgaEn - this bit modifies the response by the bridge to VGA compatible addresses, which are defined as memory addresses in the range 000A_0000h - 000B_FFFFh, and I/O space addresses in the bottom 64KB of PCI I/O space, where the bottom 10 bits are in the range 3B0h - 3BBh or 3C0h - 3DFh. Address bits 15:10 of I/O addresses are not decoded, allowing for ISA aliasing of the above address ranges. If set, the bridge will forward these addresses from the primary to the secondary bus and block forwarding them from the secondary to the primary bus; regardless of the contents of the memory and I/O range registers, the ISA Enable bit (in this register), or the VGA Palette Snoop Enable bit (in the Command register). Not persistent through warm reset.
2	RW	0	IsaEn - this bit modifies the response by the bridge to ISA I/O addresses, which are defined as those addresses in the top 768 bytes of each 1 KB block of the first 64KB of PCI I/O space. If this bit is set, it will block forwarding these addresses from the primary to the secondary bus and cause forwarding of these addresses from the secondary to the primary bus; regardless of the contents of the IoBase and IoLimit registers. Not persistent through warm reset.
1	RW	0	SerrEn - this bit controls forwarding of system errors from the secondary interface to the primary interface. If it is set, SERR# on the secondary bus will cause a system error (indicated by the assertion of the FATAL_ERR_N or NONFATAL_ERR_N error interrupt pins), assuming that the SERR enable bit is set for the primary interface in the command register. Not persistent through warm reset.
0	RW	0	Parity Error Response Enable - enables parity errors to be reported by P_PERR_N, FATAL_ERR_N or NONFATAL_ERR_N error interrupt. Not persistent through warm reset.

HyperTransport Capability ID **Offset 40h**

Bits	Type	Reset	Description
7:0	R	08h	Id - HyperTransport Capability ID assigned by the PCI SIG.

Capability 2 **Offset 41h**

Bits	Type	Reset	Description
7:0	R	00h	Pointer - pointer to the next capability block. It always points to 0, indicating no additional capability blocks.

Notes

HyperTransport Command **Offset 43h–42h**

Bits	Type	Reset	Description
15:13	R	000b	Capability Type - indicates what type of HyperTransport capability block this is. For a primary block, it is always 000b.
12	R	0	Reserved.
11	RW	0	Default Direction - this determines which link requests initiated by this device will be placed on. Not persistent through warm reset. 0 = Send requests toward the Master Host Bridge (as determined by the Master Host field). 1 = Send requests in the opposite direction.
10	R	0	Master Host - contains the number of the link pointing to the master host bridge on the HyperTransport chain. Updated with the link number that the register was written from whenever the HyperTransport Command register is written.
9:5	R	01h	Unit Count - the number of UnitIDs consumed by the HyperTransport PCI bridge. This number is always 1.
4:0	RW	00h	Base Unit ID - this is the base of the range of HyperTransport UnitIDs occupied by this device. Not persistent through warm reset.

HyperTransport Link 0 Control **Offset 45h–44h**

Bits	Type	Reset	Description
15	R	0	Reserved.
14	RC	0	NXA Error - this bit is set whenever a response or posted write is dropped due to hitting end of chain. It can be cleared by writing a 1 to it. Persistent through warm reset.
13	RC	0	Overflow Error - this bit is set whenever an overflow error is detected. It may be cleared by writing a 1 to it. Persistent through warm reset.
12	RC	0	Protocol Error - this bit is set whenever a protocol error is detected. It may be cleared by writing a 1 to it. Persistent through warm reset.
11:8	RC	0h	CRC Err - each bit is set whenever a CRC error is detected on the corresponding byte lane of the link. Each bit may be cleared by writing a 1 to it. The HyperTransport PCI bridge only has one byte lane, so only bit 8 will ever get set. Persistent through warm reset.
7	RS	0	Xmit Off - this bit shuts off the link transmitter to reduce EMI and power. The EOC bit should always be set prior to setting the XmitOff bit. It may only be set by software, not cleared. It may only be cleared by a warm or cold reset sequence on HyperTransport.
6	RS	0	End Of Chain - this bit indicates that this link is not part of the logical HyperTransport chain and that this device should be considered the end of the chain for packets coming from the other direction. Packets directed toward this link are dropped. Nonposted requests result in nonexistent address (NXA) error responses. It may only be set, not cleared, by software. Not persistent through warm reset.

Notes

HyperTransport Link 0 Control — Offset 45h–44h<Emphasis> (Continued)

Bits	Type	Reset	Description
5	R	0	Init Done - this read-only bit indicates that low-level link initialization has successfully completed on the link.
4	RW	0	Link Fail - this bit is set to indicate that a failure has been detected on a link and it should not be used. It is persistent through warm reset and will prevent link initialization when set. It may be set either by hardware or software and cleared by software. Software within values do not take effect until the next warm reset.
3	RW	0	CRC Force Error - when this bit is a 1, bad CRC will be generated on all outgoing traffic on the link. Not persistent through warm reset.
2	RS	0	CRC Start Test - writing a 1 to this bit causes hardware to initiate a CRC test sequence on the link. When the test sequence has completed, hardware will clear the bit. Not persistent through warm reset.
1	RW	0	CRC Sync Flood Enable - if set, this bit causes CRC errors to be treated as fatal errors. When detected, they will cause all HyperTransport links from this device to be flooded with sync packets and the LinkFail bit to be set. Not persistent through warm reset.
0	R	0	Reserved.

Link 0 Width Control — Offset 47h–46h

Bits	Type	Reset	Description
15	R	0	Reserved.
14:12	R	000b	Out - this controls the used width of the outgoing link from this device. It must match the used incoming width of the device on the other end of the link: 000b = 8 bit 001b = 16 bit 011b = 32 bit The HyperTransport PCI bridge only supports 8-bit links.
11	R	0	Reserved.
10:8	R	000b	In - this controls the used width of the incoming link to this device. It must match the used outgoing width of the device on the other end of the link: 000b = 8 bit 001b = 16 bit 011b = 32 bit The HyperTransport PCI bridge only supports 8-bit links.
7	R	0	Reserved.
6:4	R	000b	Max Out - indicates the maximum width of the outgoing link supported by this device: 000b = 8 bit 001b = 16 bit 011b = 32 bit The HyperTransport PCI bridge only supports 8-bit links.
3	R	0	Reserved.

Notes

Link 0 Width Control **Offset 47h–46h<Emphasis> (Continued)**

Bits	Type	Reset	Description
2:0	R	000b	Max In - indicates the maximum width of the incoming link supported by this device: 000b = 8 bit 001b = 16 bit 011b = 32 bit The HyperTransport PCI bridge only supports 8-bit links.

HyperTransport Link 1 Control **Offset 49h–48h**

Bits	Type	Reset	Description
15	R	0	Reserved.
14	RC	0	NXA Error - this bit is set whenever a response or posted write is dropped due to hitting end of chain. It may be cleared by writing a 1 to it. Persistent through warm reset.
13	RC	0	Overflow Error - this bit is set whenever an overflow error is detected. It may be cleared by writing a 1 to it. Persistent through warm reset.
12	RC	0	Protocol Error - this bit is set whenever a protocol error is detected. It may be cleared by writing a 1 to it. Persistent through warm reset.
11:8	RC	0	CRC Error - each bit is set whenever a CRC error is detected on the corresponding byte lane of the link. Each bit may be cleared by writing a 1 to it. The HyperTransport PCI bridge only has one byte late, so only bit 8 will ever be set. Persistent through warm reset.
7	RS	0	Xmit Off - this bit shuts off the link transmitter to reduce EMI and power. The EOC bit should always be set prior to setting the XmitOff bit. It may only be set by software, not cleared. It may only be cleared by a warm or cold reset sequence on HyperTransport.
6	RS	0	End Of Chain - this bit indicates that this link is not part of the logical HyperTransport chain and that this device should be considered the end of the chain for packets coming from the other direction. Packets directed toward this link are dropped. Nonposted requests result in nonexistent address (NXA) error responses. It may only be set, not cleared, by software. Not persistent through warm reset.
5	R	0	Init Done - this read-only bit indicates that low-level link initialization has successfully completed on the link.
4	RW	0	Link Fail - this bit is set to indicate that a failure has been detected on a link and it should not be used. It may be set either by hardware or software and cleared by software. Software within values do not take effect until the next warm reset. It is persistent through warm reset and will prevent link initialization when set.
3	RW	0	CRC Force Error - when this bit is a 1, bad CRC will be generated on all outgoing traffic on the link. Not persistent through warm reset.
2	RS	0	CRC Start Test - writing a 1 to this bit causes hardware to initiate a CRC test sequence on the link. When the test sequence has completed, hardware will clear the bit. Not persistent through warm reset.

Notes

HyperTransport Link 1 Control **Offset 49h–48h<Emphasis> (Continued)**

Bits	Type	Reset	Description
1	RW	0	CRC Sync Flood Enable - if set, this bit causes CRC errors to be treated as fatal errors. When detected, they will cause all HyperTransport links from this device to be flooded with sync packets and the LinkFail bit to be set. Not persistent through warm reset.
0	R	0	Reserved.

Link 1 Width Control **Offset 4Bh–4Ah**

Bits	Type	Reset	Description
15	R	0	Reserved.
14:12	R	000b	Out - this controls the used width of the outgoing link from this device. It must match the used incoming width of the device on the other end of the link: 000b = 8 bit 001b = 16 bit 011b = 32 bit The HyperTransport PCI bridge only supports 8-bit links.
11	R	0	Reserved.
10:8	R	000b	In - this controls the used width of incoming link to this device. It must match the used outgoing width of the device on the other end of the link: 000b = 8 bit 001b = 16 bit 011b = 32 bit The HyperTransport PCI bridge only supports 8-bit links.
7	R	0	Reserved.
6:4	R	000b	Max Out - indicates the maximum width of the outgoing link supported by this device: 000b = 8 bit 001b = 16 bit 011b = 32 bit The HyperTransport PCI bridge only supports 8-bit links.
3	R	0	Reserved.
2:0	R	000b	Max In - indicates the maximum width of the incoming link supported by this device: 000b = 8 bit 001b = 16 bit 011b = 32 bit The HyperTransport PCI bridge only supports 8-bit links.

HyperTransport Revision ID **Offset 4Ch**

Bits	Type	Reset	Description
7:5	R	Pass1 000b Pass2 001b	Major - major revision ID of the HyperTransport specification supported by the HyperTransport PCI bridge.

Notes

HyperTransport Revision ID **Offset 4Ch<Emphasis> (Continued)**

Bits	Type	Reset	Description
4:0	R	Pass1 10001b Pass2 00000b	Minor - minor revision ID of the HyperTransport specification supported by the HyperTransport PCI bridge.

Note: In rev 1.0 of the Tsi301 HyperTransport to PCI bridge, the HyperTransport Revision ID is given as 0.17. However, Pass1 is also compliant to LDT I/O Specification Revision 1.0, with the exceptions that the Tsi301 does not support 2 and 4 bit interfaces or standard frequency control.

In Pass2, the HyperTransport Revision ID is 1.0. 2 and 4 bit interfaces are not supported.

Link Frequency **Offset 4Dh**

Bits	Type	Reset	Description
7:4	RW	0000b	Link 1 Frequency Control - controls the link transmitter frequency. • If using sync mode, these bits also control the receiver frequency. • If the core133Sel and Lx_clkSel strappings are set to a value logic can recognize as generating a 200 MHz clock, the register reads 0000 at cold reset. • If logic can not recognize the core133Sel and Lx-clkSel strappings as generating a 200 MHz clock, the register reads 1111 at cold reset. • If SIP is used, the register reads 1111 at cold reset. If the register comes up 0000 at cold reset, software can write it to either 0000 or 0010 and go through warm reset to switch between 200 and 400 MHz operation. All other encodings are reserved and can lead to undefined behavior. Each 4-bit field contains one of the following values: 0000 = 200 MHz 0010 = 400 MHz 1111 = Vendor specified Persistent through warm reset.
3:0	RW	0000b	Link 0 Frequency Control - controls the link transmitter frequency. • If using sync mode, these bits also control the receiver frequency. • If the core133Sel and Lx_clkSel strappings are set to a value logic can recognize as generating a 200 MHz clock, the register reads 0000 at cold reset. • If logic can not recognize the core133Sel and Lx-clkSel strappings as generating a 200 MHz clock, the register reads 1111 at cold reset. • If SIP is used, the register reads 1111 at cold reset. If the register comes up 0000 at cold reset, software can write it to either 0000 or 0010 and go through warm reset to switch between 200 and 400 MHz operation. All other encodings are reserved and can lead to undefined behavior. Each 4-bit field contains one of the following values: 0000 = 200 MHz 0010 = 400 MHz 1111 = Vendor specified Persistent through warm reset.

Note: In Pass1, the Link Frequency CSR did not exist.

Notes

Read Control **Offset 62h–60h**

Bits	Type	Reset	Description
23:22	RW	000b	Reserved.
21:19	RW	000b	Line Prefetch Initial Count - indicates the minimum number of lines that must be successfully prefetched from memory on a MemRdLine (or MemRd if prefetching is enabled for MemRd commands) before allowing the PCI requester to reconnect. Not persistent through warm reset.
18:16	RW	000b	Multiple Prefetch Intitial Count - indicates the minimum number of lines that must be successfully prefetched from memory on a MemRdMult before allowing the PCI requester to reconnect. Not persistent through warm reset.
15:12	RW	0h	Reserved.
11	RW	0	Line Prefetch Continue - if set, and prefetching for MemRdLine commands is enabled, MemRdLine (and MemRd, if prefetching is enabled for MemRd commands) prefetching will be continuous. As each line of data is returned to PCI, another line will be read from HyperTransport, creating a moving prefetch window. Otherwise, prefetching will end when the specified number of lines has been fetched. Not persistent through warm reset.
10	RW	0	MultPrefetchContinue - if set, and prefetching for MemRdMult commands is enabled, MemRdMult prefetching will be continuous. As each line of data is returned to PCI, another line is read from HyperTransport, creating a moving prefetch window. Otherwise, prefetching will end when the specified number of lines has been fetched. Not persistent through warm reset.
9:8	RW	00b	PCI Delayed Requests - this controls the number of PCI delayed requests that may be outstanding at one time. The value in the register plus 1 is the number that will be allowed, enabling from one to four buffers. Not persistent through warm reset.
7:5	RW	000b	Line Prefetch Count - this indicates the number of lines to be prefetched for MemRdLine commands, and MemRd commands if prefetching is enabled for them, in addition to the line containing the original request address. MemRdLine (and MemRd, if enabled) always prefetch at least to the end of the first line. This field may not be set to a larger value than Multiple Prefetch Count. Not persistent through warm reset.
4:2	RW	000b	Multiple Prefetch Count - this indicates the number of lines to be prefetched for MemRdMult commands, in addition to the line containing the original request address. MemRdMults always prefetch at least to the end of the first line. Not persistent through warm reset.
1	RW	0	Mem Rd Prefetch Enable - if set, PCI MemRd commands are treated as prefetchable using the same prefetch controls as for memRdLine. Otherwise, no prefetching is performed for MemRds, and they fetch only the initially requested DW or QW. Not persistent through warm reset.
0	RW	0	Prefetch Enable - this bit enables prefetching for prefetchable read requests. If clear, no prefetching of any kind is performed. Not persistent through warm reset.

Notes

PCI Control **Offset 63h**

Bits	Type	Reset	Description
7:6	R	00b	Reserved.
5	RW	0	Target Receive FIFO - if asserted, new requests from the PCI bus are only accepted when the PCI interface's target receive FIFO is completely empty. When deasserted, writes are accepted as long as there is space in the FIFO for the write command and the first beat of data. Not persistent through warm reset.
4	RW	0	Park Master - this bit controls where the arbiter defaults to when there is no PCI request outstanding. The default can be to grant the PCI bus to P_GNT0_N (assumed to be connected to the HyperTransport PCI bridge) or to grant the PCI bus to the most recent master on the bus. 0 = Park PCI bus on most recent master. 1 = Park PCI bus on P_GNT0_N. Not persistent through warm reset.
3:0	RW	Fh	ID Sel Charge - number of PCI clocks to charge the AD lines on a Type 0 configuration cycle before asserting P_FRAME_N. Not persistent through warm reset.

Error Control **Offset 67h–64h**

Bits	Type	Reset	Description
31	R	0	Reserved.
30	RC	0	PCI Command/Address Parity Error - a parity error was detected on the PCI bus during the address phase. This is only logged if the Parity Error Response Enable bit in the Bridge Control register is set. Not persistent through warm reset.
29	RC	0	Master Posted PCI Command Error - this is set whenever a PCI posted write intended by the HyperTransport PCI bridge fails to complete. Possible reasons are: • Master Abort Received with Master Abort Mode = 1 • Target Abort Received • TRDY# Timeout • Retry Timeout • Write received PERR with Parity Error Response Enable = 1 Not persistent through warm reset.
28	RW	0	Discard SERR Fatal - if a secondary bus discard timer expiration occurs with the Discard Timer SERR# enable asserted, this bit controls whether the SERR gets mapped to a fatal or nonfatal interrupt: 0 = Nonfatal 1 = Fatal Not persistent through warm reset.
27	RC	0	Response Match Error - this chip received a response packet to its unitID, indicating that it belonged with a request issued from here, but no request with that srcTag is outstanding. Persistent through warm reset.

Notes

Error Control **Offset 67h–64h<Emphasis> (Continued)**

Bits	Type	Reset	Description
26	RW	0	Response Match Error Nonfatal Enable - a nonmatching response causes a nonfatal interrupt to be asserted, as enabled by SerrEn in the Command register (see “Command Offset 05h–04h” on page -6). Not persistent through warm reset.
25	RW	0	Response Match Error Fatal Enable - a nonmatching response causes a fatal interrupt to be asserted, as enabled by SerrEn in the Command register (see “Command Offset 05h–04h” on page -6). Not persistent through warm reset.
24:22	RC	0	Reserved.
21	RW	0	Command/Address Parity Error Nonfatal Enable - if asserted, detection of a command phase parity error on the PCI bus causes a nonfatal interrupt to be asserted, as enabled by SerrEn in the Command register (see “Command Offset 05h–04h” on page -6). Not persistent through warm reset.
20	RW	0	Command/Address Parity Error Fatal Enable - if asserted, detection of a command phase parity error on the PCI bus causes a fatal interrupt to be asserted, as enabled by SerrEn in the Command register (see “Command Offset 05h–04h” on page -6). Not persistent through warm reset.
19	RW	0	Post Nonfatal Enable - if asserted, PCI master posted writes that fail to complete successfully on the bus result in a nonfatal interrupt assertion, if enabled by SerrEn in the Command register (see “Command Offset 05h–04h” on page -6). Not persistent through warm reset.
18	RW	0	Post Fatal Enable - if asserted, PCI master posted writes that fail to complete successfully on the bus result in a fatal interrupt assertion, if enabled by SerrEn in the Command register (see “Command Offset 05h–04h” on page -6). Not persistent through warm reset.
17	RC	0	Retry Timeout - a master transaction was retried beyond the value in Retry Timer. Not persistent through warm reset.
16	RC	0	TRDY Timeout - a master transaction took a TRDY timeout. Not persistent through warm reset. Not persistent through warm reset.
15:8	RW	Pass1 80h Pass2 0	Retry Timer - controls how many retries of a particular operation HyperTransport tries before giving up. Setting the disconnects timer to 0 disables it. Not persistent through warm reset.
7:0	RW	Pass1 80h Pass2 0	TRDY Timer - controls the number of PCI clocks that the HyperTransport PCI bridge waits for TRDY or STOP when acting as a master before timing out. Setting timer to 0 disables it. Not persistent through warm reset.

HyperTransport Error Control **Offset 69h–68h**

Bits	Type	Reset	Description
15:12	R	0h	Reserved.
11	RW	0	SERR Fatal - BrCtrlSerrEn controls whether an interrupt gets generated at all when SERR assertion is detected. This bit controls whether the interrupt is Fatal or Nonfatal: 0 = Nonfatal Enable 1 = Fatal Enable Not persistent through warm reset.

Notes

HyperTransport Error Control — Offset 69h–68h<Emphasis> (Continued)

Bits	Type	Reset	Description
10	RW	0	CRC Nonfatal Enable - if asserted, detection of an HyperTransport CRC error causes a nonfatal interrupt. Not persistent through warm reset.
9	RW	0	CRC Fatal Enable - if asserted, detection of an HyperTransport CRC error causes a fatal interrupt. Not persistent through warm reset.
8	RW	0	NXA Error, Sync Flood Enable - if asserted, detection of a posted HyperTransport request or HyperTransport response hitting the end of the HyperTransport chain causes sync flooding and sets the LinkFail bit. Not persistent through warm reset.
7	RW	0	NXA Error, Nonfatal Enable - if asserted, detection of a posted HyperTransport request or HyperTransport response hitting the end of the HyperTransport chain causes a nonfatal interrupt. Not persistent through warm reset.
6	RW	0	NXA Error, Fatal Enable - if asserted, detection of a posted HyperTransport request or HyperTransport response hitting the end of the HyperTransport chain causes a fatal interrupt. Not persistent through warm reset.
5	RW	0	Overflow Error, Sync Flood Enable - if asserted, detection of an HyperTransport receive buffer overflow error causes sync flooding and sets the LinkFail bit. Not persistent through warm reset.
4	RW	0	Overflow Error, Nonfatal Enable - if asserted, detection of an HyperTransport receive buffer overflow error causes a nonfatal interrupt. Not persistent through warm reset.
3	RW	0	Overflow Error, Fatal Enable - if asserted, detection of an HyperTransport receive buffer overflow error causes a fatal interrupt. Not persistent through warm reset.
2	RW	0	Protocol Error, Sync Flood Enable - if asserted, detection of a protocol error causes sync flooding and sets the LinkFail bit. Not persistent through warm reset.
1	RW	0	Protocol Error, Nonfatal Enable - if asserted, detection of a protocol error causes a nonfatal interrupt. Not persistent through warm reset.
0	RW	0	Protocol Error, Fatal Enable - if asserted, detection of a protocol error causes a fatal interrupt. Not persistent through warm reset.

Notes

HyperTransport Rx Data Buffer Allocation **Offset 6Dh-6Ch**

Bits	Type	Reset	Description
15:14	R	00b	Reserved.
13:12	RW	01b	WantPReq - the number of buffers minus one to try to keep released in the posted request channel. Not persistent through warm reset.
11:10	RW	01b	WantNpReq - the number of buffers minus one to try to keep released in the nonposted request channnel. Not persistent through warm reset.
9:8	RW	01b	WantResp - the number of buffers minus one to try to keep released in the response channel. Not persistent through warm reset.
7:6	R	00b	Reserved.
5:4	RW	01b	NeedPReq - the mimimum data buffer allocation to the posted request channel. Not persistent through warm reset.
3:2	RW	01b	NeedNpReq - the minimum data buffer allocation to the nonposted request channel. Not persistent through warm reset.
0:1	RW	01b	NeedResp - the minimum data buffer allocation to the response channel. Not persistent through warm reset.

HyperTransport Transmit Control **Offset 6Eh:6Eh**

Bit	Type	Reset	Description
7:4	R	0h	Reserved
3:0	RW	4h	BufRelSpace - controls the throttling of buffer release messages on a busy bus. If the bus is idle, buffer releases always get issued immediately. When the bus is busy, buffer release messages are forced into the packet stream. This field gives the minimum number of DW that must be allowed to pass between forced buffer releases to prevent them from absorbing too much bandwidth. Not persistent through warm reset.

PCI Control 2 **Offset 6Fh–6Fh**

Bit	Type	Reset	Description
7:4	R	0h	Reserved.
3:3	RW_R	1b	IntPassPW - value of the PassPW bit for interrupt packets. This bit controls whether interrupts are allowed to pass other posted request packets. A value of 0 orders interrupts behind other posted writes (including other interrupts) from this chip. **Note:** In Pass1, this CSR did not exist. Hardware behaved as if this CSR was always set to 1.

Notes

PCI Control 2 — Offset 6Fh–6Fh<Emphasis> (Continued)

Bit	Type	Reset	Description
2:0	RW_R	5h	TrcvThrshld - this register sets the threshold for determining when the target receive FIFO is full. The FIFO is considered full when the count of items in the FIFO exceeds this value. Since 2 or more items may be received by PCI after the disconnect is initiated, this value should never be set above 5 and must never be set to 0. **Note:** The target receive FIFO is used to store requests and write data when the Tsi301 is acting as a target on the PCI bus. When the FIFO is considered full, it causes PCI disconnects. *In Pass1, this CSR did not exist. Hardware behaved as if this CSR was always set to 5.*

Link Impedance Control 0 — Offset 73h–70h

Bits	Type	Reset	Description
31:27	R	00h	Current TxUp - current impedance value for transmit pull-up resistor.
26:22	R	00h	Current TxDown - current impedance value for transmit pull-down resistor.
21:17	R	00h	Current RxTerm - current impedance value for receive termination resistor.
16:12	RW	00h	RxTerm - impedance value for receive termination resistor. Used when RxSel is set. Not persistent through warm reset.
11	RW	0	RxSel - enables use of CSR receive impedance values. Not persistent through warm reset.
10:6	RW	00h	TxUp - impedance value for transmit pull-up resistor. Used when TxSel is set. Not persistent through warm reset.
5:1	RW	00h	TxDown - impedance value for transmit pull-down resistor. Used when TxSel is set. Not persistent through warm reset.
0	RW	0	TxSel - enables use of CSR transmit impedance values. Not persistent through warm reset.

Link Impedance Control 1 — Offset 77h–74h

Bits	Type	Reset	Description
31:27	R	00h	Current Tx Up - current impedance value for transmit pull-up resistor.
26:22	R	00h	Current Tx Down - current impedance value for transmit pull-down resistor.
21:17	R	00h	Current Rx Term - current impedance value for receive termination resistor.
16:12	RW	00h	Rx Term - impedance value for receive termination resistor. Used when RxSel is set. Not persistent through warm reset.
11	RW	0	Rx Select - enables use of CSR receive impedance values. Not persistent through warm reset.
10:6	RW	00h	Tx Up - impedance value for transmit pull-up resistor. Used when TxSel is set. Not persistent through warm reset.
5:1	RW	00h	Tx Down - impedance value for transmit pull-down resistor. Used when TxSel is set. Not persistent through warm reset.
0	RW	0	Tx Select - enables use of CSR transmit impedance values. Not persistent through warm reset.

Notes

Serial ROM (SROM) Interface 0 Offset 83h–80h

Bits	Type	Reset	Description
31:20	R	Values come from SROM.	Reserved.
19	R	Values come from SROM.	Reduce Sync Zero - reduce the number of zeros in Sync Sequence.
18	R	Values come from SROM.	Sync Pointer Control - use Rx Sync Pointer Control.
17:15	R	Values come from SROM.	Tx Initial Offset - initial Pointer Offset for Tx HyperTransport FIFO Pointers.
14:10	R	Values come from SROM.	Rx Margin - HyperTransport Receive Pointer margin.
9:5	R	Values come from SROM.	Tx Denom - denominator value for Tx HyperTransport/Clock ratio.
4:0	R	Values come from SROM.	Rx Denominator - denominator value for HyperTransport/Clock ratio.

Notes

Serial ROM (SROM) Interface 0 Rx Offset 87h–84h

Bits	Type	Reset	Description
31:0	R	Values come from SROM.	Numerator - numerator value for HyperTransport/Clock ratio.

Serial ROM (SROM) Interface 0 Tx Offset 8Bh–88h

Bits	Type	Reset	Description
31:0	R	Values come from SROM.	Numerator - numerator value for HyperTransport/Clock ratio.

Serial ROM (SROM) Interface 1 Offset 8Fh–8Ch

Bits	Type	Reset	Description
31:20	R	Values come from SROM.	Reserved.
19	R	Values come from SROM.	Reduce Sync Zero - reduce the number of zeros in Sync Sequence.
18	R	Values come from SROM.	Sync Pointer Control - use Rx Sync Pointer Control.
17:15	R	Values come from SROM.	Tx Initial Offset - initial Pointer Offset for Tx HyperTransport FIFO Pointers.
14:10	R	Values come from SROM.	Rx Margin - HyperTransport Receive Pointer margin.
9:5	R	Values come from SROM.	Tx Denominator - denominator value for Tx HyperTransport/Clock ratio.
4:0	R	Values come from SROM.	Rx Denominator - denominator value for HyperTransport/Clock ratio.

Serial ROM (SROM) Interface 1 Rx Offset 93h–90h

Bits	Type	Reset	Description
31:0	R	Values come from SROM.	Numerator - numerator value for HyperTransport/Clock ratio.

Serial ROM (SROM) Interface 1 Tx Offset 97h–94h

Bits	Type	Reset	Description
31:0	R	Values come from SROM.	Numerator - numerator value for HyperTransport/Clock ratio.

Notes

Serial ROM (SROM) Interface Control Offset 9Bh–98h

Bits	Type	Reset	Description
31:26	R	Values come from SROM.	Reserved.
25:20	R	Values come from SROM.	*Pass1* Debug Select 3 - Port 3 debug select. Used only in monitor mode. *Pass2* - Reserved.
19:14	R	Values come from SROM.	*Pass 1* Debug Select 2 - Port 2 debug select. Used only in monitor mode. *Pass2* - Reserved.
13:8	R	Values come from SROM.	Debug Select 1 - Port 1 debug select.
7:2	R	Values come from SROM.	Debug Select 0 - Port 0 debug select.
1	R	Values come from SROM.	Reserved.
0	R	Values come from SROM.	Reserved (always reads 0).

BlockxInterrupty Offsets BFh–A0h

Bits	Type	Reset	Description
15	RW	0	Interrupt Enable. Not persistent through warm reset. 0 = Disabled. 1 = Enabled.
14	RW	0	Destination Mode. Not persistent through warm reset. 0 = Physical 1 = Logical
13:6	RW	FFh	Destination ID - 8 bit physical mask or logical ID interpreted by host bridges. Not persistent through warm reset. FF = Broadcast.
5:4	RW	0	Message Type. Not persistent through warm reset. 00 = Fixed 01 = Arbitrated 10 = SMI 11 = NMI
3	RW	1	Polarity - active polarity of interrupt. Not persistent through warm reset. 0 = Active low 1 = Active High
2	RW	0	Trigger Mode - Level/Edge trigger. Not persistent through warm reset. 0 = Edge 1 = Level
1:0	RW	See Table 5.2	Vector - lower two bits of interrupt vector. Not persistent through warm reset. 00 = Interrupt 0 01 = Interrupt 1 10 = Interrupt 2 11 = Interrupt 3

Notes

Special Interrupts **Offset C1h–C0h**

Bits	Type	Reset	Description
15:14	RW	11b	INTR Message Type. Not persistent through warm reset. 00 = Fixed 01 = Arbitrated 10 = ExtInt 11 = Disabled
13:12	RW	11b	INTR Interrupt Vector - lower two bits of the INTR interrupt vector. Not persistent through warm reset.
11:10	RW	11b	INIT Message Type. Not persistent through warm reset. 00 = Fixed 01 = Arbitrated 10 = INIT 11 = Disabled
9:8	RW	10b	INIT Vector - lower two bits of the INIT interrupt vector. Not persistent through warm reset.
7:6	RW	11b	NMI Message Type. Not persistent through warm reset. 00=NMI 01 = Startup 10 = ExtInt 11 = Disabled
5:4	RW	01b	NMI Vector - lower two bits of NMI interrupt vector. Not persistent through warm reset.
3:2	RW	11b	SMI Message Type. Not persistent through warm reset. 00 = NMI 01 = Startup 10 = SMI 11 = Disabled
1:0	RW	00b	SMI Vector - lower two bits of the SMI# interrupt vector. Not persistent through warm reset.

Interrupt Diagnostics **Offset C2h**

Bits	Type	Reset	Description
7	RW	0b	Reserved by API NetWorks.
6	RS_RC	0b	Initiate - writing a 1 asserts an interrupt as if it was coming from the pin. This bit is cleared by hardware once the interrupt is sent. Not persistent through warm reset.
5	R_P	0	Active - if 1, the given pin has an asserted (level) interrupt. Not persistent through warm reset.

Notes

Interrupt Diagnostics **Offset C2h<Emphasis> (Continued)**

Bits	Type	Reset	Description
4:0	RW	00000b	Pin Number - determines which pin the Activate and Initiate fields affect (0–19 only; others are reserved). 3 - 0 Special Interrups 7 - 4 Block0 11 - 8 Block1 12 - 15 Block2 19 - 16 Block3 Not persistent through warm reset.

Interrupt Block Level 0 **Offset C3h**

Bits	Type	Reset	Description
7	R	0b	Reserved (always reads 0).
6	RW	0b	Block Enable - indicates that the current group is being used. Not persistent through warm reset.
5:0	RW	000000b	Block Vector - upper vector bits for Block 0 interrupts. Not persistent through warm reset.

Interrupt Block Level 1 **Offset C4h**

Bits	Type	Reset	Description
7	R	0b	Reserved (always reads 0).
6	RW	0b	Block Enable - indicates that the current group is being used. Not persistent through warm reset.
5:0	RW	000000b	Block Vector - upper vector bits for Block 1 interrupts. Not persistent through warm reset.

Notes

Interrupt Block Level 2 **Offset C5h**

Bits	Type	Reset	Description
7	R	0b	Reserved (always reads 0).
6	RW	0b	Block Enable - indicates that the current group is being used. Not persistent through warm reset.
5:0	RW	000000b	Block Vector - upper vector bits for Block 2 interrupts. Not persistent through warm reset.

Interrupt Block Level 3 **Offset C6h**

Bits	Type	Reset	Description
7	R	0b	Reserved (always reads 0).
6	RW	0b	Block Enable - indicates that the current group is being used.
5:0	RW	000000b	Block Vector - upper vector bits for Block 3 interrupts.

Interrupt Special Block **Offset C7h**

Bits	Type	Reset	Description
7	R	0b	Reserved (always reads 0).
6	RW	0b	Block Enable - indicates that the current group is being used. Not persistent through warm reset.
5:0	RW	000000b	Block Vector - upper vector bits. Not persistent through warm reset.

Transmit Buffer Counter Maximum Count0
Offset CAh-C8

Bits	Type	Reset	Description
3:0	RW	Fh	PCmd - posted command buffer threshold. Not persistent through warm reset.
7:4	RW	Fh	PData - posted data buffer threshold. Not persistent through warm reset.
11:8	RW	Fh	NpCmd - nonposted command buffer threshold. Not persistent through warm reset.
15:12	RW	Fh	NpData - nonposted data buffer threshold. Not persistent through warm reset.
19:16	RW	Fh	RCmd - response command buffer threshold. Not persistent through warm reset.
23:20	RW	Fh	RData - response data buffer threshold. Not persistent through warm reset.

Notes

Note: Buffer releases in excess of these thresholds are discarded to allow throttling of traffic.

Transmit Buffer Counter Maximum Count1 Offset CEh-CCh

Bits	Type	Reset	Description
3:0	RW	Fh	PCmd - posted command buffer threshold. Not persistent through warm reset.
7:4	RW	Fh	PData - posted data buffer threshold. Not persistent through warm reset.
11:8	RW	Fh	NpCmd - nonposted command buffer threshold. Not persistent through warm reset.
15:12	RW	Fh	NpData - nonposted data buffer threshold. Not persistent through warm reset.
19:16	RW	Fh	RCmd - response command buffer threshold. Not persistent through warm reset.
23:20	RW	Fh	RData - response data buffer threshold. Not persistent through warm reset.

Debug Offset D7h–D4h

Bits	Type	Reset	Description
31:28	R	0	Reserved.
27	RW	0	Interrupt Debug Enable. Not persistent through warm reset.
26:24	RW	011b	Interrupt Debug Select. Not persistent through warm reset.
23:18	RW	110010b	Select 3 - select for Debug Port 3. Not persistent through warm reset. Bits 18–20: *Pass1* Level 1 MUX select 8–1. *Pass2* - Reserved. Bit 21:0 = Rx, 1 = Tx Bit 22:0 = Port 0, 1 = Port 1 Bit 23:0 = Core Debug, 1 = HyperTransport Debug
17:12	RW	110000b	Select 2 - select for Debug Port 2. Not persistent through warm reset. Bits 12–14: *Pass1* Level 1 MUX select 8–1. *Pass2* - Reserved. Bit 15:0 = Rx, 1 = Tx Bit 16:0 = Port 0, 1 = Port 1 Bit 17:0 = Core Debug, 1 = HyperTransport Debug
11:6	RW	100010b	Select 1 - select for Debug Port 1. Not persistent through warm reset. Bits 6–8:Level 1 MUX select 8–1 Bit 9:0 = Rx, 1 = Tx Bit 10:0 = Port 0, 1 = Port 1 Bit 11:0 = Core Debug, 1 = HyperTransport Debug
5:0	RW	011111b	Select 0 - select for Debug Port 0. Not persistent through warm reset. Bits 0–2:Level 1 MUX select 8–1 Bit 3:0 = Rx, 1 = Tx Bit 4:0 = Port 0, 1 = Port 1 Bit 5:0 = Core Debug, 1 = HyperTransport Debug

Notes

HyperTransport PCI Bridge Diagnostics **Offset D8h**

Bits	Type	Reset	Description
7:5	R	0	Reserved.
4	RW	1	Broadcast - broadcast enable bit for CSR read commands. If cleared to 0, Slave ID (bits 3–0) is used. If set, read data is combined from all sources. Not persistent through warm reset.
3:0	RW	0	Slave ID - slave ID for CSR read commands. Used to specify a particular internal CSR slave module to read from. Not persistent through warm reset.

Diagnostics Link 0 Receive CRC Expected **Offset DFh–DCh**

Bits	Type	Reset	Description
31:0	R		Expected CRC value for Link 0. This register is for software use and will survive cold and warm reset as long as there is no power off.

Note: The value of this register is captured from the CRC logic following detection of a CRC error. The value is unaffected by any form of reset. The value is undefined at power up.

Notes

Diagnostics Link 0 Receive CRC Received **Offset F3h–F0h**

Bits	Type	Reset	Description
31:0	R		Received CRC value for Link 0. This register is for software use and will survive cold and warm reset as long as there is no power off.

Note: The value of this register is captured from the CRC logic following detection of a CRC error. The value is unaffected by any form of reset. The value is undefined at power up.

Diagnostics Link 1 Receive CRC Received **Offset F7h–F4h**

Bits	Type	Reset	Description
31:0	R	0	Expected CRC value for Link 0. This register is for software use and will survive cold and warm reset as long as there is no power off.

Note: The value of this register is captured from the CRC logic following detection of a CRC error. The value is unaffected by any form of reset. The value is undefined at power up.

Note:

Diagnostics Link 1 Receive CRC Received **Offset FBh–F8h**

Bits	Type	Reset	Description
31:0	R	0	Received CRC value for Link 0. This register is for software use and will survive cold and warm reset as long as there is no power off.

Note: The value of this register is captured from the CRC logic following detection of a CRC error. The value is unaffected by any form of reset. The value is undefined at power up.

Scratch **Offset FFh–FCh**

Bits	Type	Reset	Description
31:0	RW	00000000h	Scratch - read/write software scratch register. Not persistent through warm reset.

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 who are 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 to develop 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 from 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.

(Disclaimer Rev.1.01)

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-us/.

Trademarks

Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners.