IBM 133 PCI-X Bridge R2.0

Datasheet

October 15, 2001

Printed in the United States of America October 2001

IBM Microelectronics Division
1580 Route 52, Bldg. 504
Hopewell Junction,
NY 12533-6351

The IBM home page can be found at
http://www.ibm.com

The IBM Microelectronics Division home page
can be found at http://www.chips.ibm.com

ppb20_titlepage.fm.01
October 15, 2001

List of Tables

List of Figures

1. Preface

1.1 About This Book

This book describes the features and operation of the IBM 133 PCI-X Bridge R2.0. While every effort has been made to ensure that all information contained herein is as accurate as possible, users of this document should be aware that it is subject to change by IBM without notice. Any comments, corrections, and recommended additions to improve this book may be forwarded to the address inside the front cover.

1.2 Who Should Read This Manual

This document is designed for engineers and system designers interested in implementing PCI-X capable systems or adapters using a PCI-X to PCI-X transparent bridge device.

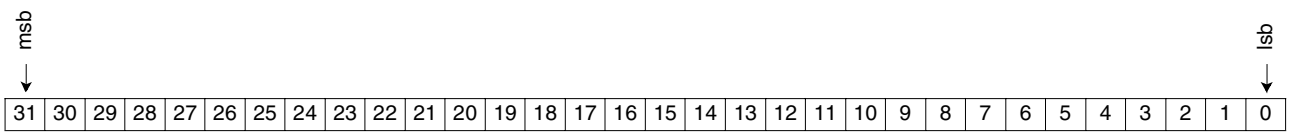
It is assumed that the reader is acquainted with the PCI and PCI-X bus specifications. For details about bus protocols and general bridge architecture, see the documents listed in *References* on page 12.

1.3 Document Contents

Chapter	Description
1. *Preface*	Document conventions, audience, references, and terminology
2. *General Information*	Architectural overview with highlights of the features and functions of the IBM 133 PCI-X Bridge R2.0 as well as a functional block diagram
3. *Bus Operation*	Detailed description of the operation of the device in its various supported modes
4. *Transaction Ordering*	Description of the ordering rules that control the forwarding of transactions across the bridge
5. *Configuration Registers*	An explanation of the configuration registers and programming interface for the device
6. *Clocking and Reset*	Clocking requirements, reset functions, and related parameters
7. *Signal Descriptions*	A list of device signals and descriptions of the their use
8.*Package Pin Assignments*	The signal-to-package pin assignment list
9. *JTAG Boundary Scan*	The mapping of the device I/Os to bits in the JTAG Boundary Scan Ring
10. *Electrical Information*	Electrical specifications for the device
11. *Mechanical Information*	Mechanical and thermal specifications for the device

1.4 Conventions and Notation

Throughout this document, bit notation is non-IBM, meaning that bits and bytes are numbered in descending order from left to right. Thus, for a four-byte word, bit 31 is the most significant bit and bit zero is the least significant bit.



1.4.1 Units of Measure

Prefixes: K=1024 k=1000.

Bits and Bytes: An upper-case ‘B’ stands for bytes. For example: 1 KB means 1024 bytes.

A lower-case ‘b’ refers to bits (see *Numeric Notation* below for when ‘b’ denotes a binary value). For example: 1Kb means 1024 bits.

Signals that are active low are designated by a # symbol at the end of the signal name. For example: P_ACK64#.

1.4.2 Numeric Notation

Hexadecimal values are in single quotation marks and preceded by an x. For example: x‘0B00’.

Undefined hexadecimal values are indicated by a capital X. For example: x’X1’ = undefined on reset.

Binary values are spelled out (zero and one) or appear in single quotation marks and are preceded by b. For example: b‘0101’.

1.5 References

1. *PCI Local Bus Specification*, Revision 2.2, December 18, 1998
2. *PCI-X Addendum to the PCI Local Bus Specification*, Revision 1.0a, July 24, 2000
3. *PCI-to-PCI Bridge Architecture Specification*, Revision 2.0, December 18, 1998
4. *PCI Power Management Interface Specification*, Revision 2.0, December 18, 1998

These specifications are downloadable for members of the PCI Special Interest Group (you will be prompted for a username and password) from the group’s web page at http://www.pcisig.com.

1.6 Document Nomenclature

Bytes, Words, DWords, and QWords	When used in this document, the term *byte* refers to a field containing eight bits and the term *word* means a field containing sixteen bits or two bytes. Hence, a double word (abbreviated DWord or DW) contains 32 bits or four bytes, and a quadword (abbreviated QWord or QW) is made up of 64 bits or eight bytes. These entities are assumed to be naturally aligned on the appropriate boundary, thus the least significant three bits of address of a QWord are zeros.
Bit and Byte Numbering	All bit and byte numbering in this specification is given in “little endian” notation. Bit 0 is the least significant bit (lsb) and byte 0 is the least significant byte (LSB). Any exceptions to this will be specifically noted to avoid any confusion. Bit ranges are given by two numbers separated by a colon and enclosed in parentheses (for example, (7:0)); the first number in the range indicates the msb or most significant bit and the last number indicates the lsb.

Destination Bus	For transactions that cross the bridge, the destination bus is the bus on which the the completer of a transaction resides.
Downstream	Transactions forwarded from the primary interface to the secondary interface are referred to as flowing downstream, regardless of the direction of the data flow.
Originating Bus	For transactions that cross the bridge, the originating bus is the bus on which the initiator of a transaction resides.
Parity	Throughout this specification, *odd parity* should be interpreted as the total number of bits with a b‘1’ value within a field and the field’s associated parity bit is an odd number. Similarly, *even parity* should be interpreted as the total number of bits with a b‘1’ value within a field and the field’s associated parity bit is an even number.
Primary Interface	The interface connected to the bus closest to the Host Processor.
Reserved Bits	The term ‘reserved’ is applied to any undefined, unimplemented, or spare bits within the registers of the device. No assumptions may be made about these bits in any way, as these bits may have meaning assigned to them in the future. Care must be taken when accessing registers with reserved bit fields. For read accesses, such bits should not be expected to have any specific or consistent value and appropriate masks should be used to extract only those bits that are defined. For write accesses, the values of reserved bit fields should be preserved by performing a read-modify-write sequence.
Secondary Interface	The interface connected to the bus farthest from the Host Processor.
Upstream	Transactions forwarded from the secondary interface to the primary interface are referred to as flowing upstream, regardless of the direction of the data flow.

2. General Information

2.1 Features

PCI-X Interfaces

- Complies with *PCI Local Bus Specification, Revision 2.2, December 18, 1998*; and *PCI-to-PCI Bridge Architecture Specification, Revision 2.0, December 18, 1998; and PCI-X Addendum to the PCI Local Bus specification, Revision 1.0a, July 24, 2000*
- Uses the 3.3V signaling environment and does not support the optional 5 V I/O signaling levels
- Primary and secondary interface clocks may be run synchronously or asynchronously
- Concurrent primary and secondary bus operations
- Supports configurations of PCI mode or PCI-X mode on either bus in any combination

Memory Buffer Architecture

- 4KB of buffering for upstream memory burst read commands, with up to eight active transactions allowed
- 4KB of buffering for downstream memory burst read commands, with up to eight active transactions allowed
- 1KB of buffering for upstream posted memory write commands, with up to eight active transactions allowed
- 1KB of buffering for downstream posted memory write commands, with up to eight active transactions allowed
- Allows one active single data phase (4-byte) delayed or split transaction in each direction

Power Management

- Supports D0 and D3 power states

Transaction Forwarding

- I/O, Memory, and Prefetchable Memory base and limit registers for downstream forwarding
- Inverse address decoding for upstream forwarding
- Flat addressing model
- Supports VGA-compatible addressing and palette snooping for upstream transactions
- Supports full 64-bit addressing and Dual Address Cycles
- Responds as medium-speed device on both interfaces

Configuration Registers

- 1 set of standard PCI and device specific configuration registers, accessible from both the primary and secondary interfaces
- Supports Type 0 and Type 1 configuration cycles

Optional Features

- Capable of defining an optional opaque (undecoded) memory address region to facilitate applications with embedded processors
- Supports secondary side PCI-X device privatization
- Optional Definable Base Address Register for use by embedded sub-systems on the secondary bus
- Optional access to configuration register space from the secondary bus

Bus Arbitration

- On-chip programmable bus arbiter for the secondary bus with support for up to six external masters
- Priority and masking control for each agent

IEEE® 1149.1 JTAG port

- Performs boundary-scan testing

2.2 Description

The IBM 133 PCI-X Bridge R2.0 transparently connects two electrically separate PCI-X bus domains, allowing concurrent operations on both buses. This results in good utilization of the buses in various system configurations and enables hierarchical expansion of I/O bus structures.

As described by the PCI-X architecture, the IBM 133 PCI-X Bridge R2.0 is capable of handling 64-bit data at a maximum bus frequency of 133 MHz (depending upon the bus topology and load) and is backward compatible with all 3.3V I/O conventional PCI interfaces.

The IBM 133 PCI-X Bridge R2.0 also provides extensive buffering and prefetching mechanisms for efficient transfer of data through the device, facilitating multi-threaded operation and high system throughput.

2.3 Technology Highlights

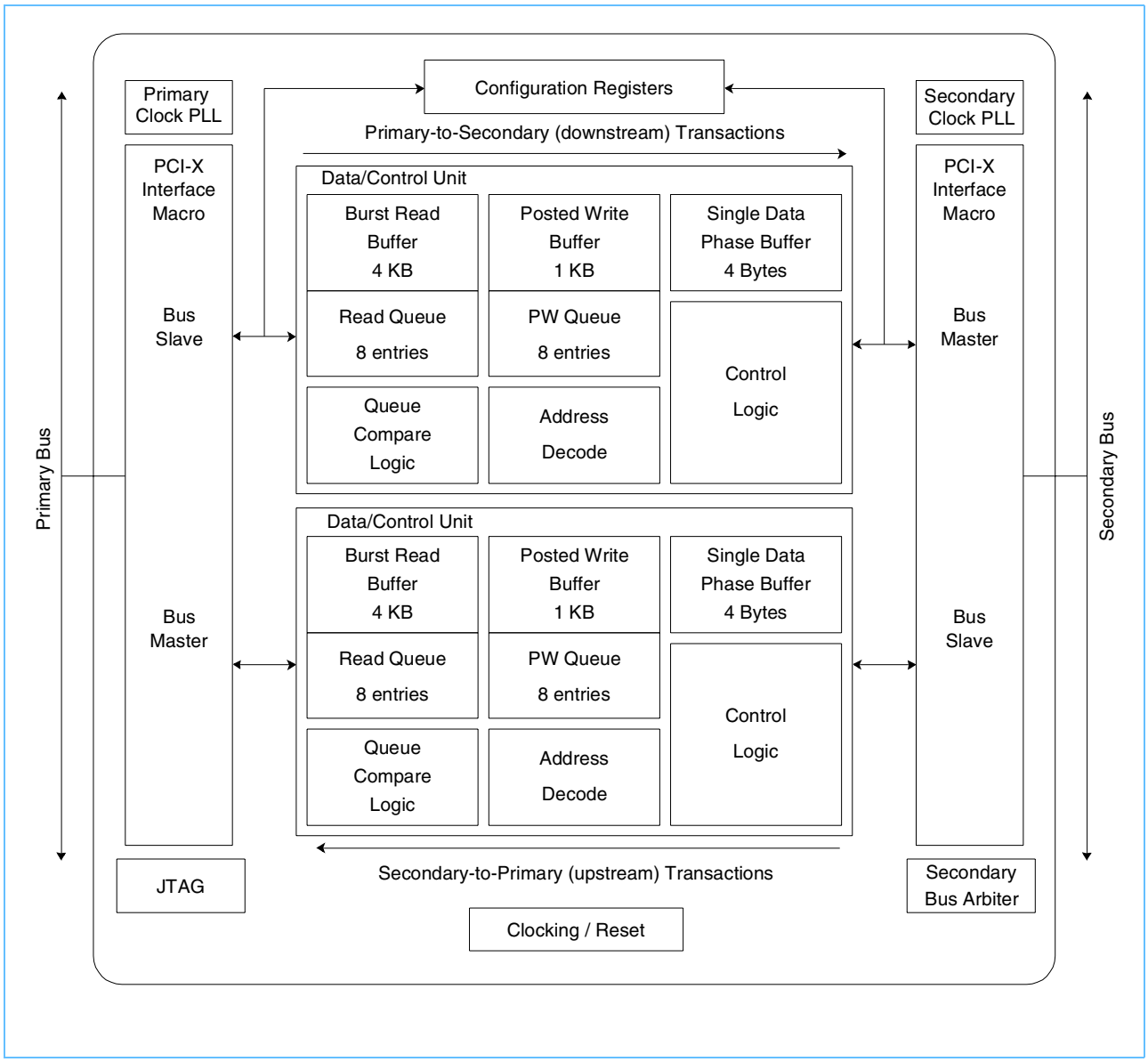
The IBM 133 PCI-X Bridge R2.0 is implemented using IBM's CMOS 6SF technology, which is a 0.25 micron (µm) lithography process with a 0.18 µm $L_{effective}$. The device requires two power supplies, one at 2.5 V for internal logic and the other at 3.3 V to power the device I/O circuits. The device is packaged in a 31mm thermally and electrically enhanced plastic ball grid array (H-PBGA) with 304 balls. See the *Electrical Information* on page 139 and the *Mechanical Information* on page 141 for further details.

2.4 Block Diagram

The IBM 133 PCI-X Bridge R2.0 is composed of the following major functional blocks as shown in *Figure 2-1*:

- Two PCI-X interface macros, one of the elements in the IBM Blue Logic™ Core Library. Each macro handles the PCI/PCI-X protocol for its respective bus and depending on the type of transaction, can act as either a bus master or a bus slave. These macros transfer data and control information flowing to and from the blocks shown in the middle of the diagram.
- Two phase-locked loops (PLLs), one for the primary clock domain and one for the secondary clock domain. The PLL for each clock domain is used when that bus is running in PCI-X mode; in PCI mode, the PLL is bypassed to allow the full frequency range as defined by the bus architecture. The two bus clocks may be run synchronously or asynchronously. A spread-spectrum clock input, within the architectural bounds, is supported for either or both interfaces.
- One set of configuration registers, programmable either from the primary or secondary interface. The first 64 bytes of this address space conform to the architectural format for bridge devices, called Header Type 1. The remaining 192 bytes are device-specific registers. Each register is fully defined in *PCI Configuration Space Header Registers* on page 41.
- Two data/control units, one for downstream transactions and one for upstream transactions. These symmetric units each contain separate buffers for burst read, posted write, and single data phase operations. Read and write queues, queue compare logic, address decoding, control logic, and other control functions are also included in these blocks.
- An arbiter for the secondary bus, which can be disabled if an external arbiter is employed. When enabled, bus arbitration is provided for the IBM 133 PCI-X Bridge R2.0 and up to six other masters. Each client can be assigned high or low priority, or can be masked off.
- A clocking and reset control unit to manage these common device functions.
- A JTAG controller, compliant with IEEE Standard 1149.1, to facilitate boundary scan testing.

Figure 2-1. Block Diagram



2.5 Operation Overview

This section gives a brief description of operation for various aspects of the IBM 133 PCI-X Bridge R2.0. More detailed information on these topics can be found in subsequent chapters.

2.5.1 Supported Modes

The IBM 133 PCI-X Bridge R2.0 is a full-function transparent PCI-X to PCI-X bridge. As such, either interface may be configured to operate using the conventional PCI bus protocol or the PCI-X bus protocol. In mixed-mode configurations, the IBM 133 PCI-X Bridge R2.0 hardware handles the conversion from one protocol to the other.

Any allowed bus clock frequency range for a particular mode may be used, up to 66 MHz for PCI mode and up to 133 MHz per section 9.4.1 of the PCI-X 1.0a specification. Operation at a particular speed depends on the bus topology and loading. Since the two clock domains are asynchronous and independent, a different bus speed may be used on each interface. Speed-matching is accomplished via the buffering structure of the IBM 133 PCI-X Bridge R2.0 design.

The IBM 133 PCI-X Bridge R2.0 implements a 64-bit bus on both interfaces. The PCI architecture also allows either side to be connected to a 32-bit bus or to 32-bit devices. Full 64-bit addressing capability is also provided, including support for dual address cycles (DAC).

Note: The IBM 133 PCI-X Bridge R2.0 uses the 3.3 V signaling environment and is not tolerant of 5 V signal levels. When the IBM 133 PCI-X Bridge R2.0 is mounted on an adapter card, the card must use the 3.3 V connector keying scheme.

2.5.2 Buffer Structure

The IBM 133 PCI-X Bridge R2.0 contains two symmetric sets of buffers with associated queues, one for upstream transactions and the other for downstream transactions.

2.5.2.1 Burst Read Buffers

Each burst read buffer shown in *Figure 2-1* contains 4 KB to hold data from memory burst read transactions. Each buffer is logically divided into eight independent 512-byte buffers to allow for multi-threading. Each 512-byte buffer has a read queue providing up to eight active read transactions in each direction.

Every 512-byte buffer is further divided into four 128-byte subsections. Activity generally occurs on these 128-byte boundaries. Filling and/or emptying 128 bytes causes bus transactions to be initiated. While each read queue entry has up to 512 bytes of buffer space associated with it, to keep data flowing efficiently the 128-byte subsections are re-used as needed when they are emptied. This means that when the primary and secondary interfaces are running at similar frequencies and there is little bus contention, long transfers can proceed without disconnection, after the initial latency needed to fill the first 128-byte subsection. For large transfers when the two buses are running at vastly dissimilar frequencies, disconnections may occur on the faster bus as often as every 128 bytes as the 512-byte buffer becomes completely full or empty.

2.5.2.2 Posted Write Buffers

The posted write buffers each have a capacity of 1 KB to hold data from posted memory write transactions. Each is logically divided into eight independent 128-byte segments to allow transactions to be issued on the destination bus before they have been completed on the originating bus. Unlike the burst read buffers, the

amount of space assigned to each transaction is dynamic. A single transaction can utilize from one to eight 128-byte subsections as needed. Each posted write queue is an 8-entry FIFO, providing up to eight active write transactions in each direction. Activity generally occurs when a 128-byte segment is filled or emptied, this keeps data flowing by re-using 128-byte subsections as they become available.

2.5.2.3 Single Data Phase Buffers

There is one single data phase buffer for each direction to hold read or write data from 4-byte split or delayed transactions. These transactions include all I/O or configuration operations as well as doubleword memory read operations.

2.5.3 Address Decoding

The IBM 133 PCI-X Bridge R2.0 is a transparent bridge and uses a flat addressing model. Both the PCI and PCI-X address spaces are split between the primary bus and the secondary bus. Address ranges residing on the secondary bus are defined by the I/O, memory, prefetchable memory base and limit, and the optional base address registers 0 and 1 in the bridge configuration space. All other addresses are assumed to reside on the primary bus. Inverse address decoding is used to determine when to forward transactions upstream. The only exception to this is when the optional opaque address range is enabled and defined by its base and limit registers. The IBM 133 PCI-X Bridge R2.0 does not recognize transactions for addresses within the opaque range on either bus. This region may be used, for example, for peer-to-peer communication between devices on the secondary bus.

The IBM 133 PCI-X Bridge R2.0 supports full 64-bit addressing and handles dual address cycles on both interfaces. The device provides no capability for translating addresses.

The bridge configuration registers are accessible from either the primary or secondary interface through Type 0 configuration reads and writes. On the secondary interface, the bridge claims Type 1 configuration write transactions that specify conversion to a special cycle on an upstream bus segment.

2.5.4 Bus Arbitration

The IBM 133 PCI-X Bridge R2.0 contains an arbiter for the secondary interface that is enabled or disabled via an input signal pin. It provides bus arbitration for up to six additional masters, each of which may be assigned high or low priority or may be masked off. When the internal arbiter is being used and the IBM 133 PCI-X Bridge R2.0 request is not masked off, the bus will be parked at the bridge whenever there are no pending requests.

The arbiter implements a two-level fairness algorithm that allows each device within a level to receive grant requests cyclically. The arbiter uses the arbitration priority register to determine which agents are high priority (HP) devices and which are low priority (LP) devices. At different points in time, snapshots are taken of all pending requests for each priority level. All captured HP requests are serviced first, then one of the captured LP requests is serviced. At this point, a new HP snapshot is taken, picking up any new HP requests. All captured HP requests are serviced before continuing with the next LP request still pending from the previous LP snapshot. A new snapshot of pending LP requests is taken only after all requests from the previous LP snapshot have been serviced.

3. Bus Operation

This chapter presents a summary of the PCI and PCI-X transactions, transaction forwarding across the IBM 133 PCI-X Bridge R2.0, and transaction termination.

3.1 Types of Transactions

Table 3-1 and *Table 3-2* list the command code and name for each PCI and PCI-X transaction. For each transaction type, the middle two columns indicate whether the IBM 133 PCI-X Bridge R2.0 can initiate the transaction as a master on the primary bus and on the secondary bus; the last two columns indicate whether the bridge responds to the transaction as a target on the primary bus and on the secondary bus.

Table 3-1. PCI Transactions

Command Code	Type of Transaction	Initiates as Master		Responds as Target	
		Primary	Secondary	Primary	Secondary
0000	Interrupt Acknowledge	No	No	No	No
0001	Special Cycle	Yes	Yes	No	No
0010	I/O Read	Yes	Yes	Yes	Yes
0011	I/O Write	Yes	Yes	Yes	Yes
0100	Reserved	No	No	No	No
0101	Reserved	No	No	No	No
0110	Memory Read	Yes	Yes	Yes	Yes
0111	Memory Write	Yes	Yes	Yes	Yes
1000	Reserved	No	No	No	No
1001	Reserved	No	No	No	No
1010	Configuration Read	No	Yes	Yes	Type 0
1011	Configuration Write	Type 1	Yes	Yes	Yes
1100	Memory Read Multiple	Yes	Yes	Yes	Yes
1101	Dual Address Cycle	Yes	Yes	Yes	Yes
1110	Memory Read Line	Yes	Yes	Yes	Yes
1111	Memory Write and Invalidate	Yes	Yes	Yes	Yes

Table 3-2. PCI-X Transactions

Command Code	Type of Transaction	Initiates as Master		Responds as Target	
		Primary	Secondary	Primary	Secondary
0000	Interrupt Acknowledge	No	No	No	No
0001	Special Cycle	Yes	Yes	No	No
0010	I/O Read	Yes	Yes	Yes	Yes
0011	I/O Write	Yes	Yes	Yes	Yes
0100	Reserved	No	No	No	No
0101	Reserved	No	No	No	No
0110	Memory Read	Yes	Yes	Yes	Yes
0111	Memory Write	Yes	Yes	Yes	Yes
1000	Alias to Memory Read Block	No	No	Yes	Yes
1001	Alias to Memory Write Block	No	No	Yes	Yes
1010	Configuration Read	No	Yes	Yes	Type 0
1011	Configuration Write	Type 1	Yes	Yes	Yes
1100	Split Completion	Yes	Yes	Yes	Yes
1101	Dual Address Cycle	Yes	Yes	Yes	Yes
1110	Memory Read Block	Yes	Yes	Yes	Yes
1111	Memory Write Block	Yes	Yes	Yes	Yes

As indicated in *Table 3-1* and *Table 3-2* certain commands are not supported by the IBM 133 PCI-X Bridge R2.0:

- The bridge never initiates a transaction with a reserved command code and, as a target, the bridge ignores reserved command codes.
- The bridge never initiates an interrupt acknowledge transaction and, as a target, the bridge ignores interrupt acknowledge transactions. Interrupt acknowledge transactions are expected to reside entirely on the primary bus closest to the host bridge.
- The bridge does not respond to special cycle transactions. To generate special cycle transactions on other buses, either upstream or downstream, a Type 1 configuration command must be used.

The bridge does not generate Type 0 configuration transactions on the primary interface.

3.2 Write Transactions

Write transactions are treated as either posted, delayed/split (PCI-X), or immediate write transactions as shown in *Table 3-3*.

Table 3-3. Write Transaction Handling

Type of Transaction	Type of Handling
Memory Write	Posted
Memory Write and Invalidate	Posted
Memory Write Block (PCI-X)	Posted
I/O Write	Delayed/Split (PCI-X)
Type 0 Configuration Write	Immediate on the primary bus, Delayed/Split (PCI-X) on the secondary bus.
Type 1 Configuration Write	Delayed/Split (PCI-X)

3.2.1 Posted Write Transactions

The posted mode is the default mode used for the memory-write and memory-write-and-invalidate transactions. The memory-write-block transaction also uses the posted mode. Posted is the only mode used for the memory-write-block command.

When the IBM 133 PCI-X Bridge R2.0 determines that a memory write transaction is to be forwarded across the bridge, it first checks for empty space in the posted write buffer. If space is available in the posted write buffer, the bridge accepts data until the buffer is full or the transaction is terminated. If the transaction is terminated because the buffer is full, the transaction is terminated on a 128-byte boundary. If there is no space in the posted write buffer, the transaction is terminated with retry.

Up to eight posted write transactions can be enqueued on the bridge.

3.2.1.1 PCI to PCI-X Transactions

When the originating bus is operating in the conventional PCI mode and the destination bus is operating in the PCI-X mode, the bridge must buffer memory write transactions from the conventional PCI interface and count the number of bytes to be forwarded to the PCI-X interface. If the conventional PCI transaction uses the memory write command and some byte enables are not asserted, the bridge must use the PCI-X memory write command. If the conventional PCI command is memory write and all byte enables are asserted, the bridge will use the memory write PCI-X command. If the conventional transaction uses the memory write and invalidate command, the bridge uses the PCI-X memory write block command.

The bridge attempts to transfer the write data on the PCI-X interface as soon as the transaction ends or a 128-byte boundary is crossed, whichever comes first. Writes of greater than 128 bytes are possible only if more than one 128-byte sector fills up before the write operation is issued on the PCI-X interface.

3.2.1.2 PCI-X to PCI Transactions

When the originating bus is operating in the PCI-X mode and the destination bus is operating in the conventional PCI mode, the bridge uses the conventional memory write command for both the PCI-X memory write and PCI-X memory write block commands.

The bridge attempts to transfer write data on the conventional PCI interface when the PCI-X data crosses a 128-byte boundary or the end of the PCI-X transfer occurs, whichever comes first. As long as a 128-byte buffer is full, or the end of transfer remains from the PCI-X memory write command when a 128-byte boundary is crossed, the transfer will continue on the conventional PCI interface.

3.2.1.3 PCI to PCI Transactions

When both buses are operating in conventional PCI mode, the bridge passes the memory write command that it receives to the destination interface, unless the bridge is disconnected in the middle of a memory write and invalidate and is not on a cache line boundary. In this event, the command will continue as a memory write when the bridge attempts to reconnect.

The bridge attempts to transfer a memory write command when the transaction ends or a 128-byte boundary is crossed, whichever comes first. As long as a 128-byte buffer is full or the end of transfer remains from the PCI memory write command when a 128-byte boundary is crossed, the transfer will continue.

3.2.1.4 PCI-X to PCI-X Transactions

When both buses are operating in the PCI-X mode, the bridge passes the memory write command that it receives to the destination interface along with the originating byte count and transaction ID.

The bridge attempts to transfer a memory write command when the transaction ends or a 128-byte boundary is crossed, whichever comes first. As long as a 128-byte buffer is full or the end of transfer remains from the PCI-X memory write command when a 128-byte boundary is crossed, the transfer will continue.

If a transaction is disconnected on the destination interface in the middle of a continuing transfer, the byte count and address are updated and the transaction is presented again on the destination interface. If a transaction is disconnected in the middle of a continuing transfer on the originating interface, the originator must present the transaction again with the updated byte count and address.

3.2.2 Delayed/Split Write Transactions

I/O writes, Type 1 configuration writes, and Type 0 configuration writes on the secondary bus are treated as delayed transactions in the bridge. These commands are retried on the originating bus, completed on the destination bus if necessary, and then completed on the originating bus. The bridge executes DWord transactions only as delayed transactions in the conventional PCI mode and as split requests in PCI-X mode.

There is only one request queue entry for either delayed or split write transactions.

3.2.3 Immediate Write Transactions

Type 0 configuration writes on the primary PCI interface meant for the bridge are treated as an immediate write transaction by the bridge. The bridge executes the transaction and indicates its completion by accepting the DWord of data immediately.

3.3 Read Transactions

Read transactions are treated as either delayed (PCI), split (PCI-X), or immediate read transactions, as shown in *Table 3-4. Read Transaction Handling*.

Table 3-4. Read Transaction Handling

Type of Transaction	Type of Handling
Memory Read	Delayed
Memory Read Line	Delayed
Memory Read Multiple	Delayed
Memory Read DWord (PCI-X)	Split (PCI-X)
Memory Read Block (PCI-X)	Split (PCI-X)
I/O Read	Delayed/Split (PCI-X)
Type 0 Configuration Read	Immediate on the primary bus, Delayed/Split (PCI-X) on the secondary bus
Type 1 Configuration Read	Delayed/Split (PCI-X)

3.3.1 Memory Read Transactions

The conventional PCI memory-read, memory-read-line, memory-read-multiple, PCI-X memory-read-DWord, and PCI-X memory-read-block transactions are used to transfer memory data from the originating side of the bridge to the destination side of the bridge. All memory read transactions are either delayed or split on the originating interface depending on the mode of the originating interface.

3.3.1.1 PCI to PCI-X Transactions

The IBM 133 PCI-X Bridge R2.0 must translate the conventional memory read command to either the memory read DWord or the memory read block PCI-X Command. If the conventional memory read command targets non-prefetchable memory space, the command is translated into a memory read DWord. In any other instance the conventional memory read command gets translated into a memory read block PCI-X command. The prefetching algorithm for the conventional memory read command in the prefetchable space is controlled by bits 9:8 of the primary and secondary data buffering control registers. The default value of these bits indicates that one cache line will be prefetched.

The bridge must translate the conventional memory read line command to the memory read block PCI-X command. The prefetching algorithm is controlled by bits 7:6 of the primary and secondary data buffering control registers. The default value of these bits indicates that one cache line will be prefetched.

The bridge must translate the conventional memory read multiple command to the memory read block PCI-X command. The prefetching algorithm is controlled by bits 5:4 of the primary and secondary data buffering control registers. The default value of these bits indicates that a full prefetch will be done, subject to the limit imposed by the maximum memory read byte count value set by bits (14:12) of the same register. The default value for this field is 512 bytes or an entire read buffer. Using a value greater than this is possible, but it may be constrained by the setting of the split transaction commitment limit value in the upstream or downstream split transaction register, since the target bus is in PCI-X mode. Data fetching operations will be disconnected at all 1MB boundaries.

3.3.1.2 PCI-X to PCI Transactions

The IBM 133 PCI-X Bridge R2.0 translates PCI-X memory read DWord commands into conventional memory read commands.

The bridge translates a PCI-X memory read block command into one of three conventional PCI memory read commands based on the byte count and starting address. If the starting address and byte count are such that only a single DWord (or less) is being read, the conventional transaction uses the memory read command. If the PCI-X transaction reads more than one DWord, but does not cross a cache line boundary (as indicated by the Cache Line Size register in the conventional Configuration Space header), the conventional transaction uses the memory read line command. If the PCI-X transaction crosses a cache line boundary, the conventional transaction uses the memory read multiple command.

If a disconnect occurs before the byte count of the PCI-X memory read block command is exhausted, the bridge continues to issue the command until all the bytes in the count are received. The bridge disconnects once the buffer is filled and prefetches more data as 128-byte sectors of the buffer become free when split completion data is returned to the originator, until the byte count is exhausted.

3.3.1.3 PCI to PCI Transactions

This mode does not involve any translation.

If the memory read command targets non-prefetchable memory space, the memory read fetches only the requested double word. Bits 9:8 of the primary and secondary data buffering control registers control the prefetching algorithm for the memory read command in prefetchable space. The default value of these bits indicates that up to one cache line will be prefetched.

For the memory read line command, the prefetching algorithm is controlled by bits 7:6 of the primary and secondary data buffering control registers. The default value of these bits indicates that up to one cache line will be prefetched.

For the memory read multiple command, the prefetching algorithm is controlled by bits 5:4 of the primary and secondary data buffering control registers. The default value of these bits indicates that a full prefetch will be done, subject to the limit imposed by the maximum memory read byte count value set by bits (14:12) of the same register. The default value is 512 bytes or an entire read buffer. Data fetching operations will be disconnected at all 1 MB boundaries.

3.3.1.4 PCI-X to PCI-X Transactions

This mode does not involve any translation.

The amount of data that is fetched is controlled by the downstream and upstream split transaction control register. The split transaction capacity and split transaction commitment limit fields control how much data is requested at any one time.

3.3.2 I/O Read

The I/O Read command is not translated and fetches a DWord of data. The command will either be split in the PCI-X mode or delayed in the conventional PCI mode.

3.3.3 Configuration Read

3.3.3.1 Type 1 Configuration Read

The Type 1 configuration read command is only accepted on the primary interface. The command will either be split in the PCI-X mode or delayed in the conventional PCI mode.

3.3.3.2 Type 0 Configuration Read

The Type 0 configuration read command is accepted on either the primary or secondary interface. The command returns immediate data on the primary interface regardless of the interface mode. On the secondary interface the command is treated as a delayed transaction in the PCI mode and as a split transaction in the PCI-X mode.

3.3.4 Non-Prefetchable and DWord Reads

A non-prefetchable read transaction is a read transaction in which the bridge requests exactly one DWord from the target and disconnects the initiator after delivering that one DWord of read data. Unlike prefetchable read transactions, the bridge forwards the read byte enable information for the data phase.

Non-prefetchable behavior is used for I/O, configuration, memory read transactions that fall into the non-prefetchable memory space for PCI mode, and all DWord read transactions in PCI-X mode.

3.3.5 Prefetchable Reads

A prefetchable read transaction is a read transaction where the bridge performs speculative reads, transferring data from the target before it is requested from the initiator. This behavior allows a prefetchable read transaction to consist of multiple data transfers. For prefetchable read transactions, all byte enables are asserted for all data phases.

Prefetchable behavior is used for memory read line and memory read multiple transactions, as well as for memory read transactions that fall into prefetchable memory space and are allowed to fetch more than a DWord. The amount of data that is prefetched depends on the type of transaction and the setting of bits in the primary and secondary data buffering control registers in configuration space. The amount of prefetching may also be affected by the amount of free buffer space available in the bridge, and by any read address boundaries encountered. Examples of these boundaries are cache line for cache line reads and 1M address boundary to ensure that a read does not cross into another devices' space.

3.3.5.1 Algorithm for PCI-to-PCI Mode

The algorithm used for transfers in PCI-to-PCI mode is user defined in the primary and secondary data buffering control registers. These registers have bits for memory read to prefetchable space, memory read line, and memory read multiple transactions. For memory read, the bits select whether to read a DWord, read to a cache line boundary, or to fill the prefetch buffer. For memory read line and memory read multiple transactions, the bits select whether to read to a cache line boundary or to fill the prefetch buffer. In all cases, if the bits are selected to fill the prefetch buffer, the maximum amount of data that is requested on the target interface is controllable by the setting of the maximum memory read byte count bits of the Primary and Secondary Data Buffering Control registers. When more than 512 bytes are requested, the bridge fetches data to fill the buffer and then fetches more data to keep the buffer filled as sectors (128 bytes) are emptied and become free to use again.

3.3.5.2 Algorithm for PCI-to-PCI-X Mode

The algorithm for transfers in this mode is much the same as for transfers in PCI-to-PCI mode, except that the maximum request amount may be additionally constrained by the setting of the split transaction commitment limit value in the upstream or downstream split transaction register. The only other difference is that prefetching will not cease when the originating master disconnects. Prefetching will only cease when all of the requested data is received, as required by the PCI-X architecture.

3.3.5.3 Algorithm for PCI-X-to-PCI and PCI-X-to-PCI-X Mode

The algorithm for transfers in these modes is to transfer the amount of requested data.

In the PCI-X-to-PCI mode the bridge continues to generate data requests to the PCI interface and keeps the prefetch buffer full until the entire amount of data requested is transferred.

In the PCI-X to PCI-X mode the algorithm is controlled by the split transaction commitment limit value contained in the upstream or downstream split transaction register. If the value is greater than or equal to the split transaction capacity (4KB) but less than 32KB, the maximum request amount is 512 bytes. Larger transfers will be decomposed into a series of smaller transfers, until the original byte count has been satisfied. If the commitment limit value indicates 32KB or more, the original request amount is used and decomposition is not performed.

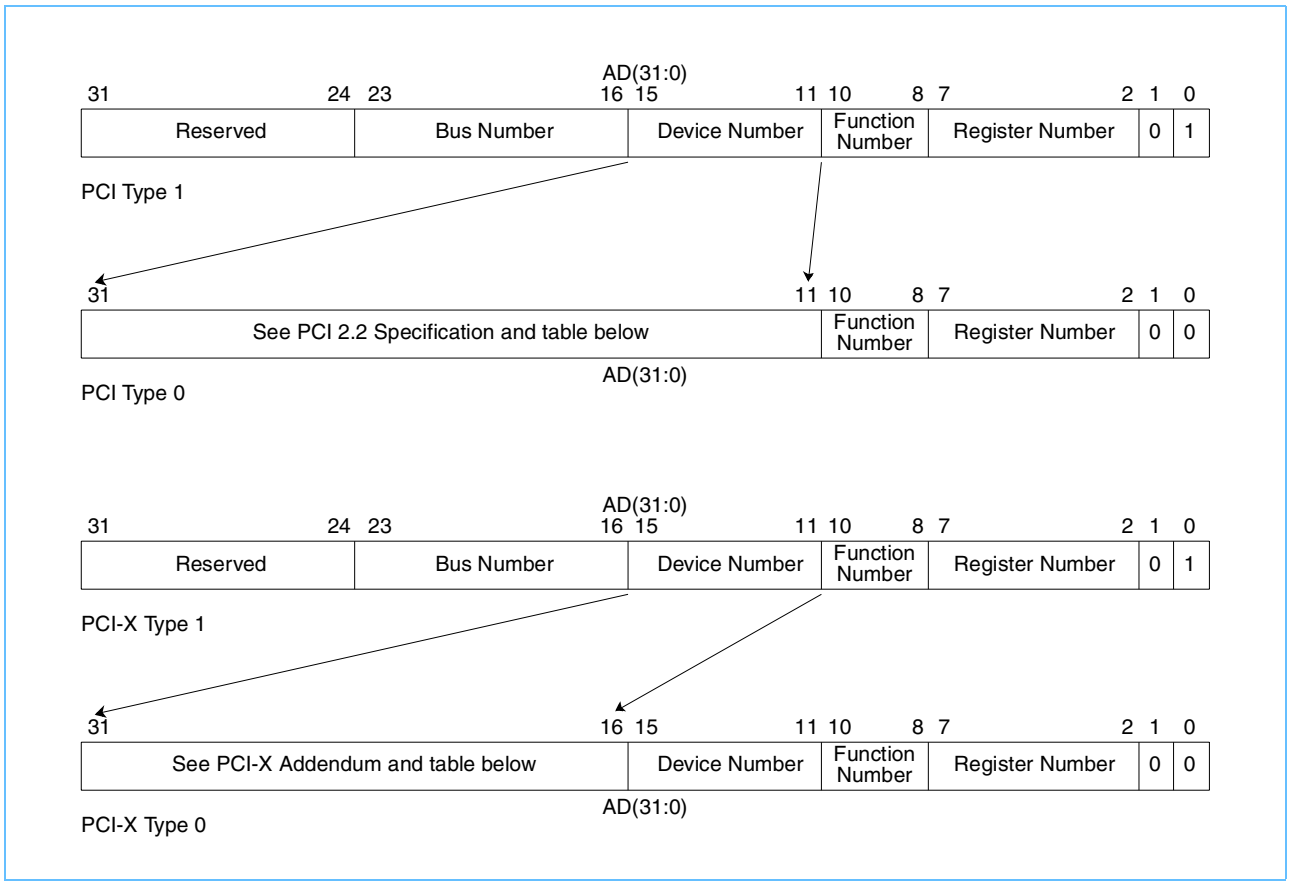
If the original request is broken into smaller requests the bridge waits until the previous completion has been totally received before a new request is issued. This ensures that the data does not get out of order and that two requests with the same sequence ID are not issued. In any case, the bridge generates a new requester ID for each request passed through the bridge.

3.4 Configuration Transactions

The *PCI Local Bus Specification, Revision 2.2, December 18, 1998* defines two configuration transaction types, Type 0 and Type 1. These two configuration formats are distinguished by the value of bus address bits (1:0). If address bits (1:0) are b‘00’ during a configuration transaction, a Type 0 configuration transaction is being indicated. A Type 0 configuration transaction is used to access configuration information for devices on the current bus segment. A Type 0 configuration transaction is not forwarded across the bridge, but rather is used to configure the bridge itself. If address bits (1:0) are b‘01’ during a configuration transaction, a Type 1 configuration transaction is being indicated. Type 1 configuration transactions are used to access devices that reside behind one or more bridges.

Figure 3-1 shows the address formats for Type 0 and Type 1 configuration transactions:

Figure 3-1. Configuration Transaction Address Formats



The register number is found in both Type 0 and Type 1 formats and gives the DWord address of the configuration register to be accessed. The function number is also included in both formats and indicates which function of a multi-function device is to be accessed. For single-function devices, this value is not decoded. Configuration transaction addresses for Type 1 and PCI-X Type 0 formats also include a 5-bit field designating the device number that identifies the device on the target PCI bus that is to be accessed. This device number is decoded to determine which IDSEL signal is asserted for the transaction. Finally, Type 1 transactions also include a bus number field that is used by the bridge to determine where in the bus hierarchy the transaction is targeted.

3.4.1 Configuration Type 0 Access to Bridge

The configuration space of the bridge is accessed by Type 0 configuration transactions. The configuration space of the bridge can be accessed from either the primary bus or the secondary bus. Applications which do not require access from the secondary bus should tie down the S_IDSEL pin.

On the primary bus, the bridge responds to a Type 0 configuration transaction by accepting the transaction when the following conditions are met during the address phase:

- The command on P_C/BE(3:0)# indicates a configuration read or configuration write transaction;
- P_AD(1:0) are b'00';
- P_IDSEL is asserted; and

- Bit 2 of the Miscellaneous Control Register is b'0'.

On the secondary bus, the bridge responds to a Type 0 configuration transaction by accepting the transaction when the following conditions are met during the address phase:

- The command on S_C/BE(3:0)# indicates a configuration read or configuration write transaction;
- S_AD(1:0) are b'00';and
- S_IDSEL is asserted.

The function number is not decoded since the bridge is a single-function device. All configuration transactions to the bridge are handled as DWord (single data phase) operations.

3.4.2 Type 1 to Type 0 Translation by Bridge

Type 1 configuration transactions are used to configure devices in a hierarchical bus structure having one or more bridges. A bridge is the only type of device that should respond to a Type 1 configuration transaction. Type 1configuration commands are used to access PCI/PCI-X devices that are located on a bus segment other than the one where the Type 1 transaction is initiated.

The bridge performs a Type 1 to Type 0 translation when a Type 1 transaction is presented on the primary interface that is destined for a device attached directly to the secondary interface. In this case, the bridge must convert the configuration transaction to a Type 0 format so that the secondary bus device can accept it. Type 1 to Type 0 translations are never performed in the upstream direction.

The bridge claims a Type 1 configuration transaction on its primary bus and translates it into a Type 0 transaction on the secondary bus when the following conditions are met during the address phase:

- The command on P_C/BE(3:0)# indicates a configuration read or configuration write transaction;
- P_AD(1:0) are b‘01’; and
- The bus number on P_AD(23:16) is the same as the value in the secondary bus number register in the bridge’s configuration space.

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

- Sets S_AD(1:0) to b‘00’;
- Decodes the device number specified and drives the bit pattern specified in *Table 3-5* on S_AD(31:16) for use in the assertion of the device’s IDSEL signal;
- Sets S_AD(15:11) to b‘00000’ if the secondary bus is operating in conventional PCI mode (in the PCI-X mode, the device number is passed through unchanged); and
- Leaves the function number and register number fields unchanged.

The bridge asserts a unique address signal based on the device number. These address signals may be used as secondary bus IDSEL signals. Mapping of the address signals depends on the device number on P_AD(15:11) of the Type1 configuration transaction.

Table 3-5 indicates how the bridge decodes the device number field. This default mapping may be modified by the secondary bus private device mask register. See *Section 5.2.5.29* on page 94 for a description of how the mapping may optionally be modified.

Table 3-5. IDSEL Generation

Device Number	Primary Address P_AD(15:11)	Secondary Address S_AD(31:16)
x'00'	00000	0000 0000 0000 0001
x'01'	00001	0000 0000 0000 0010
x'02'	00010	0000 0000 0000 0100
x'03'	00011	0000 0000 0000 1000
x'04'	00100	0000 0000 0001 0000
x'05'	00101	0000 0000 0010 0000
x'06'	00110	0000 0000 0100 0000
x'07'	00111	0000 0000 1000 0000
x'08'	01000	0000 0001 0000 0000
x'09'	01001	0000 0010 0000 0000
x'0A'	01010	0000 0100 0000 0000
x'0B'	01011	0000 1000 0000 0000
x'0C'	01100	0001 0000 0000 0000
x'0D'	01101	0010 0000 0000 0000
x'0E'	01110	0100 0000 0000 0000
x'0F'	01111	1000 0000 0000 0000
x'10' - x'1E'	10000 - 11110	0000 0000 0000 0000
x'1F'	11111	0000 0000 0000 0000 or may convert to a special cycle transaction if all criteria are met (see *Section 3.4.4* on page 32)

The bridge forwards Type 1 to Type 0 configuration read or configuration write transactions as delayed transactions in the PCI mode or as split transactions in the PCI-X mode.

3.4.3 Type 1 to Type 1 Forwarding by Bridge

Type 1 to Type 1 transaction forwarding provides a means to configure devices when a hierarchical bus structure containing two or more levels of bridges is used.

When the bridge accepts a Type 1 configuration transaction destined for a PCI/PCI-X bus downstream from its secondary interface, the bridge forwards the transaction unchanged to the secondary bus. Eventually, this transaction will be translated to a Type 0 configuration transaction or to a special cycle transaction by a downstream bridge.

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

- The command on P_C/BE(3:0)# indicates a configuration read or configuration write transaction;
- P_AD(1:0) are equal to b'01'; and
- the specified bus number is within the range defined by the secondary bus number register (exclusive) and the subordinate bus number register (inclusive).

The bridge also supports Type 1 to Type 1 forwarding of configuration write transactions upstream to allow for the generation of upstream special cycle transactions, as described in *Section 3.4.4*. All upstream Type 1 configuration read transactions are ignored by the bridge.

The bridge forwards Type 1 to Type 1 configuration read and configuration write transactions as delayed transactions in the PCI mode and as split transactions in the PCI-X mode.

3.4.4 Special Cycle Generation by the Bridge

The Type 1 configuration transaction format may be used to generate special cycle transactions in hierarchical PCI/PCI-X systems. Special cycle transactions can be generated from Type 1 configuration write transactions in either the upstream or the downstream direction.

The bridge initiates a special cycle on the destination bus when a Type 1 configuration write transaction is detected on the initiating bus and the following conditions are met during the address phase:

- The command is a configuration write;
- Address bits AD(1:0) are b‘01’;
- The device number in address bits AD(15:11) is equal to b‘11111’;
- The function number in address bits AD(10:8) is equal to b‘111’;
- The register number in address bits AD(7:2) is equal to b‘000000’; and
- The specified bus number is the same as the value in the bridge’s secondary bus number register (for downstream transactions) or matches the value in its primary bus number register (for upstream transactions).

When the bridge initiates the transaction on the destination interface, the command is changed from a configuration write to a special cycle. Devices that use special cycles ignore the address and decode only the bus command. The data phase contains the special cycle message. The transaction is forwarded as a delayed transaction in PCI mode and as a split transaction in PCI-X mode. Once the transaction is completed on the destination bus through the detection of the master abort condition, the bridge completes the transaction on the originating bus by accepting the retry of the delayed command in the PCI mode or by generating a completion message in the PCI-X mode.

Special cycle transactions received by the bridge as a target are ignored.

4. Transaction Ordering

To maintain data coherence and consistency, the IBM 133 PCI-X Bridge R2.0 complies with the ordering rules set forth in the *PCI Local Bus Specification, Revision 2.2* for the PCI mode and the *PCI-X Addendum to the PCI Local Bus Specification, Revision 1.0* for the PCI-X mode.

This chapter describes the ordering rules that control transaction forwarding across the bridge. For a more detailed discussion of transaction ordering see Appendix E of the *PCI Local Bus Specification, Revision 2.2* for the PCI mode and Section 8.4.4 of the *PCI-X Addendum to the PCI Local Bus Specification, Revision 1.0* for the PCI-X mode.

4.1 General Ordering Guidelines

Independent transactions on the primary and secondary buses have a relationship only when those transactions cross the IBM 133 PCI-X Bridge R2.0.

The following general ordering guidelines govern transactions crossing the bridge:

- Requests terminated with target retry can be accepted and completed in any order with respect to other transactions that have been terminated with target retry. If the order of delayed or split requests is important, the initiator should not start a second delayed or split transaction until the first transaction has been completed. If more than one delayed or split transaction is initiated, the initiator should repeat all retried requests, using some fairness algorithm. Repeating a delayed or split transaction cannot be contingent upon the completion of another delayed transaction; otherwise, a deadlock can occur.
- Write transactions flowing in one direction have no ordering requirements with respect to write transactions flowing in the opposite direction. The bridge can accept posted write transactions on both interfaces at the same time, and also can initiate posted write transactions on both interfaces at the same time.
- The acceptance of posted memory or memory write transactions as a target can never be contingent on the completion of a non-locked, non-posted transaction as a master. This is true of the bridge and must also be true of other bus agents; otherwise, a deadlock can occur.
- The bridge accepts posted memory or memory write transactions, regardless of the state of completion of any delayed or split transactions being forwarded across the bridge.

4.2 Ordering Rules

Table 4-1 and *Table 4-2* describe the ordering rules for the IBM 133 PCI-X Bridge R2.0 in the PCI-X and the PCI modes.

Table 4-1. IBM 133 PCI-X Bridge R2.0 Ordering Rules in PCI-X Mode

Bus Operation	Can Row Pass Column?				
	Memory Write	Split Read Request	Split Write Request	Split Read Completion	Split Write Completion
Memory Write	no	yes	yes	yes	yes
Split Read Request	no	yes	yes	yes	yes
Split Write Request	no	yes	no	yes	yes
Split Read Completion	a) no b) yes[1]	yes	yes	a) yes b) no[2]	yes
Split Write Completion	no	yes	yes	yes	no

1. If the relaxed ordering bit is set in PCI-X-to-PCI-X mode or the enable relaxed ordering bit in the primary and/or secondary data buffering control register in any other mode, Read completions can pass memory writes. See the bit descriptions in *Primary Data Buffering Control Register* on page 65 and *Secondary Data Buffering Control Register* on page 67 for details.
2. Split read completions with the same sequence ID must remain in address order.

Table 4-2. IBM 133 PCI-X Bridge R2.0 Ordering Rules in PCI Mode

Bus Operation	Can Row Pass Column?				
	Posted Memory Write	Delayed Read Request	Delayed Write Request	Delayed Read Completion	Delayed Write Completion
Posted Memory Write	no	yes	yes	yes	yes
Delayed Read Request	no	yes	yes	yes	yes
Delayed Write Request	no	yes	no	yes	yes
Delayed Read Completion	a) no b) yes[1]	yes	yes	yes	yes
Delayed Write Completion	no	yes	yes	yes	no

1. If the relaxed ordering bit is set in PCI-X-to-PCI-X mode or the enable relaxed ordering bit in the primary and/or secondary data buffering control register in any other mode, read completions can pass memory writes. See the bit description for details.

5. Configuration Registers

This section describes the standard and device specific configuration registers contained within the bridge. These registers provide various control and status reporting functions. Registers are accessible from the primary interface using the configuration read and configuration write commands.

5.1 Configuration Space

The PCI/PCI-X specifications define a separate address space called the configuration space in which the configuration registers are located. It is a contiguous block of 256 bytes, subdivided into two regions:

1. The first 64 bytes are the PCI configuration space header region, which provides for identification, configuration, and recovery capabilities.
2. The remaining 192 bytes are the PCI device dependent region, which provides for device specific configuration data. Modification of the reserved bits contained within this region may have serious and unpredictable effects on the operation of the IBM 133 PCI-X Bridge R2.0.

In addition, there are two versions of the PCI configuration space:

1. The standard configuration is often referred to as a “Type x‘00’ PCI configuration space header” because the value x‘00’ is stored in the PCI header type register (at offset x‘0E’).
2. The PCI-to-PCI bridge configuration is often referred to as a “Type x‘01’ PCI configuration space header” because the value x‘01’ is stored in the PCI header type register (at offset x‘0E’).

This chapter provides a specification for only the Type x‘01’ PCI configuration space header.

The mandatory PCI configuration space registers perform these operations:

- Detect PCI devices and their functions
- Initialize PCI devices
- Assign system resources to PCI devices
- Support catastrophic error recovery

5.2 Type x‘01’ PCI Configuration Space Header

5.2.1 Description

The 64-byte PCI configuration space header region has two subregions:

- The first 16 bytes are the PCI device independent region with registers for:
  - Vendor ID
  - Device ID
  - Command
  - Status

  - Revision ID
  - Class Code
  - Cache Line Size
  - Latency Timer
  - Header Type
  - Built-in Self Test (BIST)

- The remaining 48 bytes are the PCI Device Header Type region with registers for:
  - Base Address Registers
  - Primary Bus Number
  - Secondary Bus Number
  - Subordinate Bus Number
  - Secondary Latency Timer
  - I/O Base and Limit
  - Secondary Status
  - Memory Base and Limit
  - Prefetchable Memory Base and Limit
  - Prefetchable Base and Limit Upper 32 Bits
  - I/O Base and Limit Upper 16 Bits
  - Capabilities Pointer
  - Expansion ROM Addresses
  - Interrupt Line
  - Interrupt Pin
  - Bridge Control

5.2.2 Summary of Registers

Table 5-1 provides a list of these registers and an index to their detailed descriptions.

Table 5-1. Register Summary (Page 1 of 2)

Register Name	Starting Address	Description	See Page
PCI Configuration Space Header Registers			
Vendor ID	x‘00’	Manufacturer ID, assigned by the PCI Special Interest Group	41
Device ID	x‘02’	Device ID number	42
Command	x‘04’	PCI bus configuration parameters	43
Status	x‘06’	PCI event status	45
Revision ID	x‘08’	Revision ID number	47
Class Code	x‘09’	Class Code designator	47
Cache Line Size	x‘0C’	PCI cache line size in DWords	48
Latency Timer	x‘0D’	Latency value of bus master	49
Header Type	x‘0E’	Header type	49
BIST	x‘0F’	not supported	49
Base Address	x‘10’ x‘14’	Optional base address register	50
Primary Bus Number	x‘18’	Bus number of primary interface PCI segment	51
Secondary Bus Number	x‘19’	Bus number of secondary interface PCI segment	52
Subordinate Bus Number	x‘1A’	Bus number of highest PCI segment behind bridge	52
Secondary Latency Timer	x‘1B’	Value of secondary latency timer as bus master	53
I/O Base	x‘1C’	Base of I/O address range bits	53
I/O Limit	x‘1D’	Upper address of I/O address range bits	54
Secondary Status	x‘1E’	Secondary interface event status	55
Memory Base	x‘20	Base of memory mapped I/O address range bits	57
Memory Limit	x‘22’	Upper limit of memory mapped I/O address range bits	57
Prefetchable Memory Base	x‘24’	Base of prefetchable memory address range bits	58
Prefetchable Memory Limit	x‘26’	Upper limit of prefetchable memory address range bits	58
Prefetchable Base Upper 32 Bits	x‘28’	Base of prefetchable address range bits 63:32	59
Prefetchable Limit Upper 32 Bits	x‘2C’	Upper limit of prefetchable address range bits 63:32	59
I/O Base Upper 16 Bits	x‘30’	Base of I/O address range bits 63:32	60
I/O Limit Upper 16 Bits	x‘32’	Upper limit of I/O address range bits 63:32	60
Capabilities Pointer	x‘34’	Specifies a pointer to a capabilities list item	61
Reserved Registers	x‘35’	Reserved	61
Interrupt Line	x‘3C’	Communicates interrupt line routing information between initialization code and device driver	61
Interrupt Pin	x‘3D’	not supported	62

Table 5-1. Register Summary (Page 2 of 2)

Register Name	Starting Address	Description	See Page
Bridge Control	x‘3E’	Provides bridge-specific Command register extensions	63
Device-Specific Configuration Space Registers			
Primary Data Buffering Control	x‘40’	Provides controls for primary bus memory operations	65
Secondary Data Buffering Control	x‘42’	Provides controls for secondary bus memory operations	67
Miscellaneous Control	x‘44’	Controls miscellaneous functions, such as parity error operations	69
Arbiter Mode	x‘50’	Controls secondary bus arbitration logic	70
Arbiter Enable	x‘54’	Enables arbitration for requestors of internal secondary bus arbitration logic	71
Arbiter Priority	x‘58’	Indicates whether high or low priority is assigned to internal secondary bus arbitration logic requests	72
SERR# Disable	x‘5C’	Controls assertion of SERR# on primary bus	73
Primary Retry Counter	x‘60’	Defines number of primary bus retries	75
Secondary Retry Counter	x‘64’	Defines number of secondary bus retries	76
Discard Timer Control	x‘68’	Controls duration and enabling of discard timer	77
Retry and Timer Status	x‘6C’	Indicates expiration of a retry counter or discard timer	78
Opaque Memory Enable	x‘70’	Enables all opaque memory registers	79
Opaque Memory Base	x‘74’	Specifies base of opaque memory address range (bits 31:20)	80
Opaque Memory Limit	x‘76’	Specifies upper limit of opaque memory address range (bits 31:20)	81
Opaque Memory Base Upper 32 Bits	x‘78’	Specifies base of opaque memory address range (bits 63:32)	81
Opaque Memory Limit Upper 32 Bits	x‘7C’	Specifies upper limit of opaque memory address range (bits 63:32)	82
PCI-X ID	x‘80’	Identifies register set in Capabilities List as a PCI-X register set	82
Next Capabilities Pointer	x‘81’	Indicates more list items in Capabilities List	83
PCI-X Secondary Status	x‘82’	Reports status information about secondary interface	84
PCI-X Bridge Status	x‘84’	Identifies bridge capabilities and operating mode	86
Secondary Bus Upstream Split Transaction	x‘88’	Controls bridge buffers for forwarding Split Transactions from secondary requester to primary completer	88
Primary Bus Downstream Split Transaction	x‘8C’	Controls bridge buffers for forwarding Split Transactions from primary requester to secondary completer	89
Power Management ID	x‘90’	Identifies this register set in Capabilities List as a Power Management register set.	90
Next Capabilities Pointer	x‘91’	Indicates that there are no more items in the Capabilities List	90
Power Management Capabilities	x‘92’	Reports secondary interface power management capabilities	91
Power Management Control/ Status	x‘94’	Reports status of secondary register	92
PCI-to-PCI Bridge Support Extensions	x‘96’	Indicates that clocks are not stopped on a change of power state	93
Data Register	x‘97’	not implemented	93
Secondary Bus Private Device Mask	x‘B0’	Masks devices on the secondary interface	94
Miscellaneous Control Register 2	x‘B8’	Provides additional control over the memory read prefetch algorithm	96

5.2.3 Register Map

See *Table 5-2* for a map of the registers in the PCI Configuration Space. The reserved registers and bits return zeros when read.

Table 5-2. Register Map (Page 1 of 2)

Bits/Register Names				Starting Address	Region	Subregion
31 . . .24	23 . . .16	15 . . . 08	07 . . . 00			
Device ID		Vendor ID		x‘00’	PCI Configuration Space Header Region 64 bytes	PCI Device Independent Region 16 bytes
Status		Command		x‘04’		
Class Code			Revision ID	x‘08’		
BIST	Header Type	Latency Timer	Cache Line Size	x‘0C’		
Base Address Register 0				x‘10’		PCI Device Header Type Region 48 bytes
Base Address Register 1				x‘14’		
Secondary Latency Timer	Subordinate Bus Number	Secondary Bus Number	Primary Bus Number	x‘18’		
Secondary Status		I/O Limit	I/O Base	x‘1C’		
Memory Limit		Memory Base		x‘20’		
Prefetchable Memory Limit		Prefetchable Memory Base		x‘24’		
Prefetchable Base Upper 32 Bits				x‘28’		
Prefetchable Limit Upper 32 Bits				x‘2C’		
I/O Limit Upper 16 Bits		I/O Base Upper 16 Bits		x‘30’		
Reserved			Capabilities Pointer	x‘34’		
Expansion ROM Base Addresses				x‘38’		
Bridge Control		Interrupt Pin	Interrupt Line	x‘3C’		

Table 5-2. Register Map (Page 2 of 2)

Bits/Register Names				Starting Address	Region	Subregion
31 . . .24	23 . . .16	15 . . . 08	07 . . . 00			
Secondary Data Buffering Control		Primary Data Buffering Control		x‘40’	Device Dependent Region 192 bytes	N/A
Reserved			Miscellaneous Control	x‘44’		
Reserved				x‘48’ - x‘4C’		
Reserved			Arbiter Mode	x‘50’		
Reserved			Arbiter Enable	x‘54’		
Reserved			Arbiter Priority	x‘58’		
Reserved			SERR# Disable	x‘5C’		
Primary Retry Counter				x‘60’		
Secondary Retry Counter				x‘64’		
Reserved			Discard Timer Control	x‘68’		
Reserved			Retry & Timer Status	x‘6C’		
Reserved			Opaque Memory Enable	x‘70’		
Opaque Memory Limit		Opaque Memory Base		x‘74’		
Opaque Memory Base Upper 32				x‘78’		
Opaque Memory Limit Upper 32				x‘7C’		
PCI-X Secondary Status		Next Capabilities Pointer	PCI-X ID	x‘80’		
PCI-X Bridge Status				x‘84’		
Upstream Split Transaction				x‘88’		
Downstream Split Transaction				x‘8C’		
Power Management Capabilities		Next Capabilities Pointer	Power Management ID	x‘90’		
Data Register	Bridge Support Extensions	Power Management Control/Status		x‘94’		
Reserved				x‘98’ - x‘AC’		
Secondary Bus Private Device Mask				x‘B0’		
Reserved				x‘B4’		
Reserved		Miscellaneous Control 2		x‘B8’		
Reserved				x‘BC’ - x‘FC’		

5.2.4 PCI Configuration Space Header Registers

These sections describe the individual registers of the entire 64-byte PCI configuration space header region.

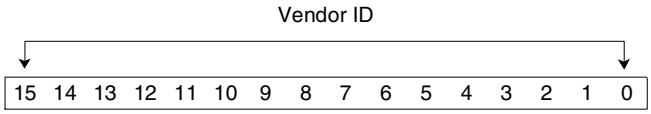
5.2.4.1 Vendor ID Register

This register identifies the manufacturer, in this case IBM, using a unique vendor ID assigned by the PCI special interest group.

Address Offset x‘00’

Access Read only

Reset Value x‘1014’



Bit(s)	Access	Field Name and Description
15:0	RO	Vendor ID. This read-only register contains IBM’s Vendor ID. The value assigned to IBM is x‘1014’.

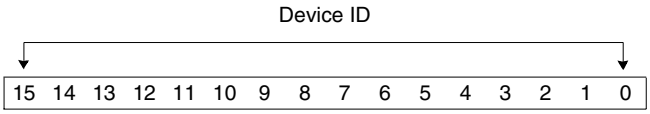
5.2.4.2 Device ID Register

This register identifies the device, using a unique device ID assigned by IBM.

Address Offset x‘02’

Access Read only

Reset Value x‘01A7’



Bit(s)	Access	Field Name and Description
15:0	RO	Device ID. This read-only register contains IBM’s Device ID. The value assigned by IBM is x‘01A7’.

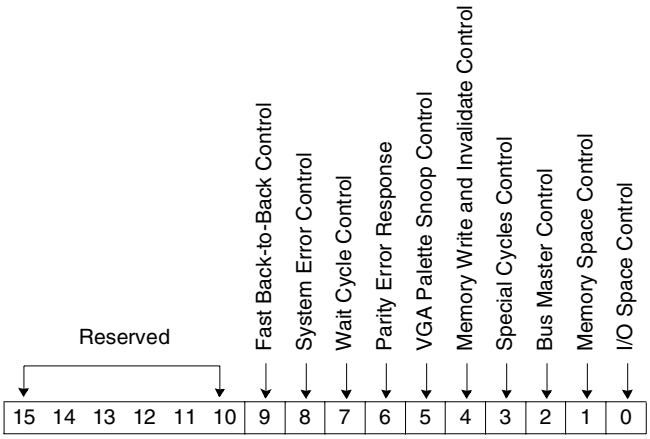
5.2.4.3 Command Register

This register provides a variety of configurable parameters defining the device's interaction with the PCI bus.

Address Offset x'04'

Access See individual bit fields.

Reset Value x'0000'



Bit(s)	Access	Field Name and Description
15:10	RO	Reserved
9	RO	Fast Back-to-Back Control 0 Fast back-to-back transactions are allowed only for the same agent This bit is ignored in the PCI-X mode
8	RW	System Error Control 0 Disable the SERR# output driver. 1 Enable the SERR# output driver.
7	RO	Wait Cycle Control 0 Disable Address/Data stepping. This bit is ignored in the PCI-X mode.
6	RW	Parity Error Response 0 Ignore detected parity errors. 1 Respond to detected parity errors. Controls the response to address and data parity errors on the primary interface. If this bit is set, the bridge must take its normal action when a parity error is detected. If this bit is cleared, the bridge must ignore any parity errors that it detects and continue normal operation. In either case, the parity error detected bit of the Status register gets set if an address or data parity error is detected.
5	RW	VGA Palette Snoop Control 0 Disable palette snooping, treat palette accesses like all other accesses. 1 Enable palette snooping.
4	RO	Memory Write and Invalidate Control 0 Disable MWI. This bit is ignored in the PCI-X mode.
3	RO	Special Cycles Control 0 Ignore special cycle operations.

Bit(s)	Access	Field Name and Description
2	RW	Bus Master Control 0 Disable generating of the PCI accesses. 1 Enable generating of the PCI accesses. Device in the PCI-X mode is allowed to initiate a split completion transaction regardless of the state of this bit.
1	RW	Memory Space Control 0 Disable memory space accesses. 1 Enable memory space accesses.
0	RW	I/O Space Control 0 Disable device response. 1 Enable device response.

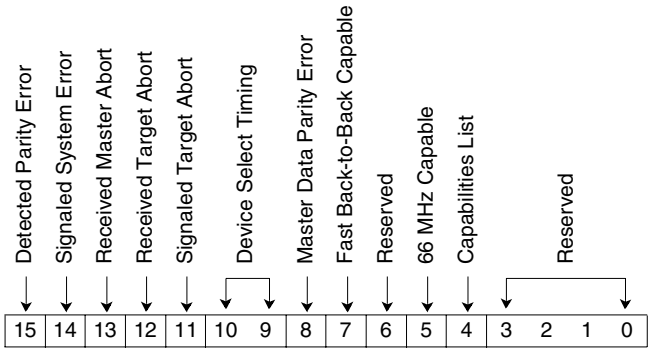
5.2.4.4 Status Register

This register records the status of PCI events.

Address Offset x‘06’

Access See individual bit fields. Reads to this register behave normally. Writes are slightly different in that bits can be reset, but not set. A bit is reset whenever the register is written, and the data in the corresponding bit location is a ‘1’.

Reset Value x‘02B0’ in the PCI mode, x‘0230’ in the PCI-X mode



Bit(s)	Access	Field Name and Description
15	RW	Detected Parity Error Status 0 Device did not detect a parity error. 1 Device detected a parity error.
14	RW	Signaled System Error Status 0 Device did not generate a SERR# signal. 1 Device generated a SERR# signal.
13	RW	Received Master Abort Status 0 Bus master transaction was not terminated with a bus master abort. 1 Bus master transaction terminated with bus master abort.
12	RW	Received Target Abort Status 0 Bus master transaction was not terminated by a target abort. 1 Bus master transaction terminated by a target abort.
11	RW	Signaled Target Abort Status 0 Target device did not terminate a transaction with a target abort. 1 Target device terminated a transaction with a target abort.
10:9	RO	Device Select (DEVSEL) Timing Status 01 Medium.
8	RW	Data Parity Status 0 No data parity errors encountered. 1 Data parity errors encountered (this bit for bus masters only).
7	RO	Fast Back-to-Back Status 0 Target not capable of accepting fast back-to-back transactions in the PCI-X mode. 1 Target capable of accepting fast back-to-back transactions in the conventional PCI mode. This bit is set by hardware when the primary interface is in the PCI mode and is set to a b‘0’ when the primary interface is in the PCI-X mode.
6	RO	Reserved

Bit(s)	Access	Field Name and Description
5	RO	66 MHz Capable Status 1 Capable of 66 MHz.
4	RO	Capabilities List 1 The capabilities linked list is available and the value read at offset x‘34’ is a pointer in configuration space to a linked list of new capabilities.
3:0	RO	Reserved

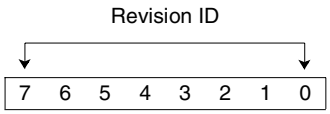
5.2.4.5 Revision ID Register

This register specifies the Revision ID for the PCI-X to PCI-X Bridge.

Address Offset x‘08’

Access Read only

Reset Value x‘02’



Bit(s)	Access	Field Name and Description
7:0	RO	x‘00’ for Rev. 1.0 of the device x‘01’ for Rev. 1.1 of the device x‘02’ for Rev. 2.0 of the device

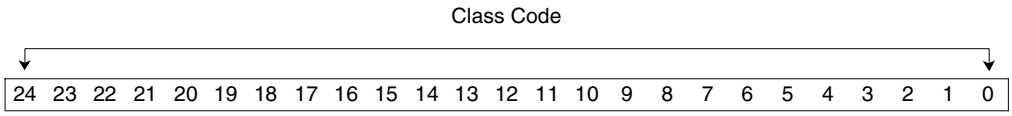
5.2.4.6 Class Code Register

This register specifies the class code for a PCI-to-PCI Bridge device.

Address Offset x‘09’

Access Read only

Reset Value x‘060400’



Bit(s)	Access	Field Name and Description
23:0	RO	x‘060400’ for a PCI-to-PCI Bridge device which does not support subtractive decode.

5.2.4.7 Cache Line Size Register

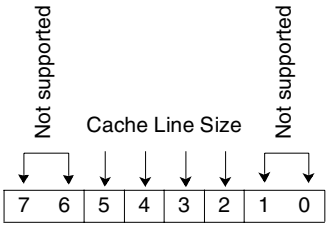
This register specifies the cache line size in 32-bit DWord units (not used when the interface is in the PCI-X mode).

Address Offset x‘0C’

Access Read/Write

Reset Value x‘00’

Restrictions Only one bit can be set at any time, if multiple bits are set or if the bits are in an invalid setting, these bits default to the 32 DWords setting.

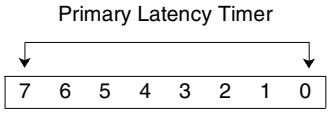


Bit(s)	Access	Field Name and Description
7:6	RW	Not supported, must equal b‘00’.
5	RW	If ‘1’, Cache Line = 32 DWords.
4	RW	If ‘1’, Cache Line = 16 DWords.
3	RW	If ‘1’, Cache Line = 8 DWords.
2	RW	If ‘1’, Cache Line = 4 DWords.
1:0	RW	Not supported, must equal b‘00’.

5.2.4.8 Latency Timer Register

This register specifies, in PCI bus clock units, the value of the latency timer for this device as a bus master. Masters that can burst for more than two data phases must implement this register as Read/Write.

Address Offset x‘0D’
Access See individual fields
Reset Value x'00' in the PCI mode, x'40' in the PCI-X mode

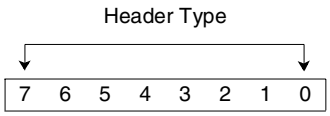


Bit(s)	Access	Field Name and Description
7:3	RW	Read/Write to set granularity in 8-cycle increments.
2:0	RO	Set to b‘000’ to force 8-cycle increments for the latency timer.

5.2.4.9 Header Type Register

This read only register specifies that a type x‘01’ header is being used for this device.

Address Offset x‘0E’
Access Read only
Reset Value x‘01’



Bit(s)	Access	Field Name and Description
7:0	RO	x‘01’

5.2.4.10 BIST Register

This register is not supported in this device.

Address Offset x‘0F’
Width 8 bits
Access Read only
Reset Value x‘00’
Restrictions Not supported

5.2.4.11 Lower Memory Base Address Register

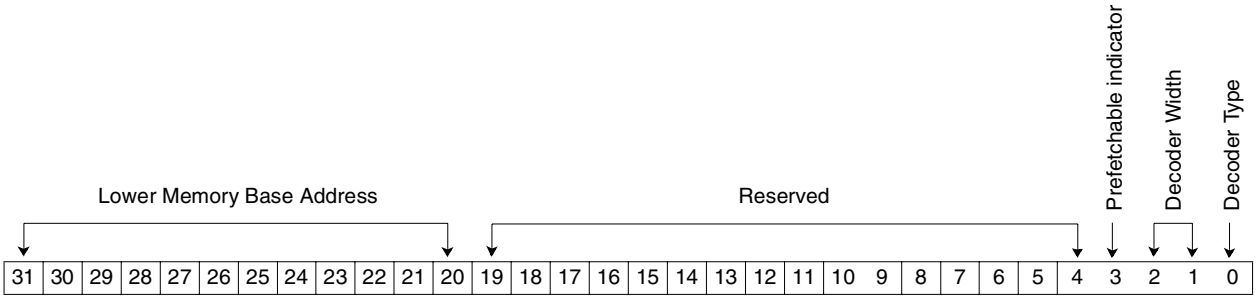
This register and the memory space defined by it are enabled by the strapping pin, BAR_EN. When the BAR_EN pin is pulled low, this register location returns zeros for reads and cannot be written. When the BAR_EN pin is pulled high, the lower memory base address register specifies address bits 31:20 of the 64 bit memory base address register. Bits 3:0 are encoded to indicate that this is part of a 64 bit register, and that it defines a prefetchable memory space. Memory accesses on the primary bus are compared against this register, if address bits 63:20 are equal to bits 63:20 of the address defined by the combination of the lower memory base address register and the upper memory base address register, the access is claimed by the bridge and passed through to the secondary bus. Memory accesses on the secondary bus are also compared against this register, if address bits 63:20 are equal to bits 63:20 of the address defined by the combination of the lower memory base address register and the upper memory base address register, the access is ignored by the bridge.

Address Offset x‘10’

Access See individual fields

Reset Value x‘0000 000C’ When BAR_EN (pin G2) is tied high. See *Table 7-3* on page 113 for details of strapping considerations.

x‘0000 0000’ When BAR_EN (pin G2) is tied low. See *Table 7-3* on page 113 for details of strapping considerations.



Bit(s)	Access	Field Name and Description
31:20	RW	Lower Memory Base Address Address bits 31:20 of the base address for an address range of prefetchable memory operations that are passed from the primary to the secondary PCI bus.
19:4	RO	Reserved
3	RO	Prefetchable indicator Identifies the address range defined by this register as prefetchable.
2:1	RO	Decoder Width Indicates that this is the lower portion of a 64 bit register.
0	RO	Decoder Type Indicates that this register is a memory decoder.

5.2.4.12 Upper Memory Base Address Register

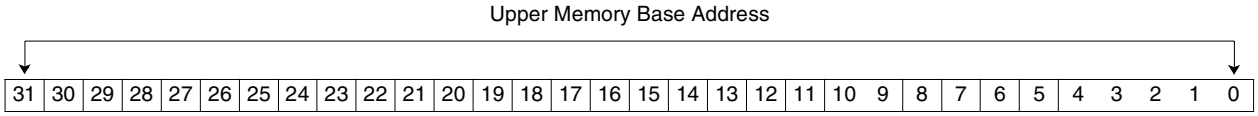
This register and the memory space defined by it are enabled by the strapping pin, BAR_EN. When the BAR_EN pin is pulled low, this register location returns zeros for reads and cannot be written. When the BAR_EN pin is pulled high, the upper memory base address register specifies address bits 63:32 of the 64 bit memory base address register. Memory accesses on the primary bus are compared against this register, if address bits 63:20 are equal to bits 63:20 of the address defined by the combination of the lower memory base address register and the upper memory base address register, the access is claimed by the bridge and passed through to the secondary bus. Memory accesses on the secondary bus are also compared against this register, if address bits 63:20 are equal to bits 63:20 of the address defined by the combination of the lower memory base address register and the upper memory base address register, the access is ignored by the bridge.

Address Offset x‘14’

Access See individual fields

Reset Value x‘0000 0000’

Note: When BAR_EN (pin G2) is tied low this register returns zero and cannot be written. *Table 7-3* on page 113 for details of strapping considerations.



Bit(s)	Access	Field Name and Description
31:0	RW	Upper Memory Base Address Address bits 63:32 of the base address for an address range of prefetchable memory operations that are passed from the primary to the secondary PCI bus.

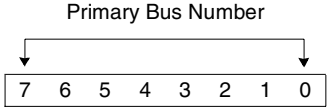
5.2.4.13 Primary Bus Number Register

The Primary Bus Number register is used to record the bus number of the PCI bus segment to which the primary interface of the bridge is connected. The configuration software programs the value in this register. The bridge uses this register to decode Type 1 configuration transactions on the secondary interface that must be converted to special cycle transactions on the primary interface.

Address Offset x‘18’

Access Read/Write

Reset Value x‘00’



Bit(s)	Access	Field Name and Description
7:0	RW	Software sets this register to the bus number of the bus segment that is attached to the primary interface of the bridge.

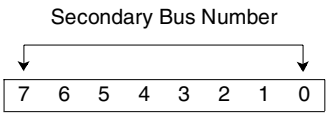
5.2.4.14 Secondary Bus Number Register

The secondary bus number register is used to record the bus number of the PCI bus segment to which the secondary interface of the bridge is connected. The configuration software programs the value in this register. The bridge uses this register to decode Type 1 configuration transactions on the primary interface that must be converted to Type 0 configuration transactions on the secondary interface. The bridge also uses the secondary bus number register and the subordinate bus number register to determine when to forward Type 1 configuration transactions upstream.

Address Offset x‘19’

Access Read/Write

Reset Value x‘00’



Bit(s)	Access	Field Name and Description
7:0	RW	Software sets this register to the bus number of the bus segment that is attached to the secondary interface of the bridge.

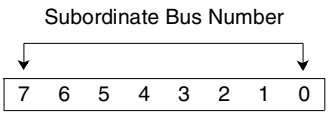
5.2.4.15 Subordinate Bus Number Register

The subordinate bus number register is used to record the bus number of the highest numbered PCI bus segment which is behind (or subordinate to) the bridge. The configuration software programs the value in this register. The bridge uses this register in conjunction with the secondary bus number register to determine when to respond to a Type 1 configuration transaction on the primary interface and pass it to the secondary interface. The bridge also uses the secondary bus number register and the subordinate bus number register to determine when to forward Type 1 configuration transactions upstream.

Address Offset x‘1A’

Access Read/Write

Reset Value x‘00’



Bit(s)	Access	Field Name and Description
7:0	RW	Software sets this register to the bus number of the highest numbered bus segment behind (or subordinate to) the bridge.

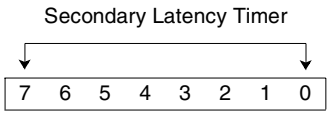
5.2.4.16 Secondary Latency Timer Register

This register specifies, in PCI bus clock units, the value of the secondary latency timer for this device as a bus master. Bus masters that can burst for more than two data phases must implement this register as Read/ Write.

Address Offset x‘1B’

Access See individual fields

Reset Value x'00' in the PCI mode, x'40' in the PCI-X mode



Bit(s)	Access	Field Name and Description
7:3	RW	Read/Write to set granularity in 8-cycle increments.
2:0	RO	Forced to b‘000’ to force 8-cycle increments for the latency timer.

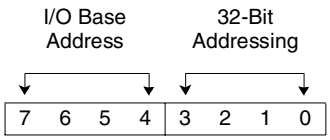
5.2.4.17 I/O Base Register

The I/O Base register specifies the base of the I/O address range bits 15:12 and is used in conjunction with the I/O limit register and I/O base upper 16 bits and I/O limit upper 16 bits registers to specify a range of 32-bit addresses supported for I/O transactions on the PCI bus. Address bits 11:0 are assumed to be x‘000’ for the base address. This register also specifies that the bridge supports 32-bit I/O addressing.

Address Offset x‘1C’

Access See individual fields

Reset Value x‘X1’



Bit(s)	Access	Field Name and Description
7:4	RW	I/O Base Address Address bits 15:12 of the base address for the address range of I/O operations that are passed from the primary to the secondary PCI bus.
3:0	RO	Set to b‘0001’ to indicate that 32-bit I/O addressing is supported.

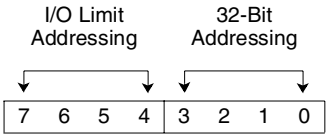
5.2.4.18 I/O Limit Register

This register specifies the upper address of the I/O address range bits 15:12 and is used in conjunction with the I/O base register and I/O base upper 16 bits and I/O limit upper 16 bits to specify a range of 32-bit addresses supported for I/O transactions on the PCI bus. Address bits 11:0 are assumed to be x‘FFF’ for the limit address. This register also specifies that the bridge supports 32-bit I/O addressing.

Address Offset x‘1D’

Access See individual fields

Reset Value x‘X1’



Bit(s)	Access	Field Name and Description
7:4	RW	I/O Limit Address Address bits 15:12 of the limit address for the address range of I/O operations that are passed from the primary to the secondary PCI bus.
3:0	RO	Set to b‘0001’ to indicate that 32-bit I/O addressing is supported.

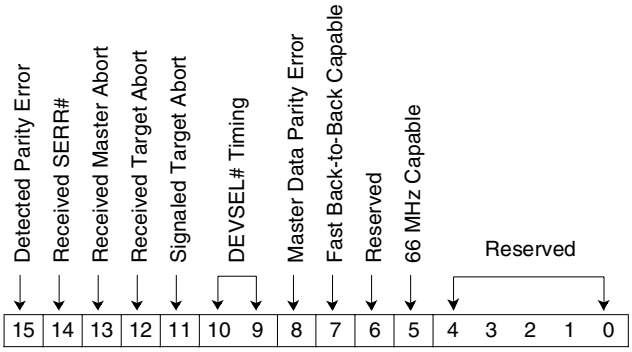
5.2.4.19 Secondary Status Register

This register is similar in function and bit definition to the Status register. However, its bits reflect status conditions of the secondary interface.

Address Offset x‘1E’

Access See individual bit fields. Writes are slightly different in that bits can be reset, but not set. A bit is reset whenever the register is written, and the data in the corresponding bit location is a ‘1’.

Reset Value x‘02A0’ in the PCI mode, x‘0220’ in the PCI-X mode



Bit(s)	Access	Field Name and Description
15	RW	Detected Parity Error Status 0 Device did not detect a parity error. 1 Device detected a parity error.
14	RW	Signaled System Error Status 0 Device did not receive a SERR# signal on the secondary interface. 1 Device received a SERR# signal on the secondary interface.
13	RW	Received Master Abort Status 0 Bus master transaction was not terminated with bus master abort. 1 Bus master transaction terminated with bus master abort.
12	RW	Received Target Abort Status 0 Bus master transaction was not terminated by target abort. 1 Bus master transaction terminated by target abort.
11	RW	Signaled Target Abort Status 0 Target device did not terminate a transaction with target abort. 1 Target device terminated a transaction with target abort.
10:9	RO	Device Select (DEVSEL#) Timing Status 01 Medium.
8	RW	Data Parity Status 0 No data parity errors encountered. 1 Data parity errors encountered (this bit for bus masters only).
7	RO	Fast Back-to-Back Capable 0 Target not capable of accepting fast back-to-back transactions in the PCI-X mode. 1 Target capable of accepting fast back-to-back transactions in conventional mode. This bit is set to a b‘1’ by hardware when the secondary interface is in the PCI mode and is set to a b‘0’ when the secondary interface is in the PCI-X mode.

Bit(s)	Access	Field Name and Description
6	RO	Reserved
5	RO	66 MHz Capable 1 Capable of 66 MHz.
4:0	RO	Reserved.

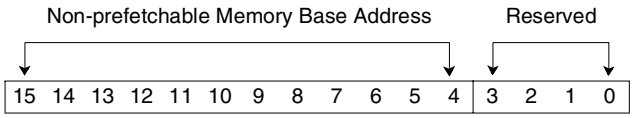
5.2.4.20 Memory Base Register

This register specifies the base of the memory mapped I/O address range bits 31:20 and is used in conjunction with the Memory Limit register to specify a range of 32-bit addresses supported for memory mapped I/O transactions on the PCI Bus. Address bits 19:0 are assumed to be x‘0 0000’ for the base address.

Address Offset x‘20’

Access See individual fields

Reset Value x‘8000’



Bit(s)	Access	Field Name and Description
15:4	RW	Non-prefetchable memory base address Address bits 31:20 of the base address for the address range of memory mapped I/O operations that are passed from the primary to the secondary PCI bus.
3:0	RO	Reserved

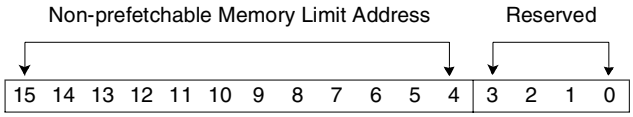
5.2.4.21 Memory Limit Register

This register specifies the upper address of the memory-mapped I/O address range bits 31:20 and is used in conjunction with the memory base register to specify a range of 32-bit addresses supported for memory mapped I/O transactions on the PCI bus. Address bits 19:0 are assumed to be x‘F FFFF’ for the limit address.

Address Offset x‘22’

Access See individual fields

Reset Value x‘0000’



Bit(s)	Access	Field Name and Description
15:4	RW	Non-prefetchable Memory Limit Address Address bits 31:20 of the limit address for the address range of memory mapped I/O operations that are passed from the primary to the secondary PCI bus.
3:0	RO	Reserved

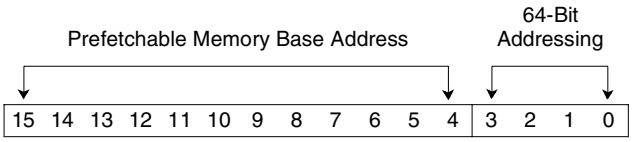
5.2.4.22 Prefetchable Memory Base Register

This register specifies the base of the prefetchable memory address range bits 31:20 and is used in conjunction with the prefetchable memory limit register, the prefetchable base upper 32 bits register, and the prefetchable limit upper 32 bits register to specify a range of 64-bit addresses supported for prefetchable memory transactions on the PCI bus. Address bits 19:0 are assumed to be x‘0 0000’ for the base address. This register also specifies that the bridge supports 64-bit prefetchable memory addressing.

Address Offset x‘24’

Access See individual fields

Reset Value x‘8001’



Bit(s)	Access	Field Name and Description
15:4	RW	Prefetchable Memory Base Address Address bits 31:20 of the base address for the address range of prefetchable memory operations that are passed from the primary to the secondary PCI bus.
3:0	RO	64-bit addressing. Set to b‘0001’ to indicate support of 64-bit addressing.

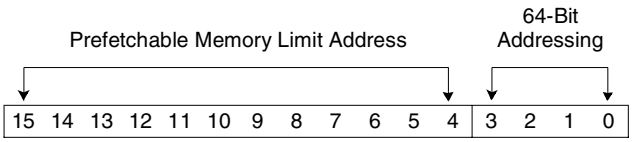
5.2.4.23 Prefetchable Memory Limit Register

This register specifies the upper address of the prefetchable memory address range bits 31:20 and is used in conjunction with the prefetchable memory base register, the prefetchable base upper 32 bits register, and the prefetchable limit upper 32 bits register to specify a range of 64-bit addresses supported for prefetchable memory transactions on the PCI bus. Address bits 19:0 are assumed to be x‘F FFFF’ for the limit address. This register also specifies that the bridge supports 64-bit prefetchable memory addressing.

Address Offset x‘26’

Access See individual fields

Reset Value x‘0001’



Bit(s)	Access	Field Name and Description
15:4	RW	Prefetchable Memory Limit Address Address bits 31:20 of the limit address for the address range of prefetchable memory operations that are passed from the primary to the secondary PCI bus.
3:0	RO	64-bit addressing. Set to b‘0001’ to indicate support of 64-bit addressing.

5.2.4.24 Prefetchable Base Upper 32 Bits Register

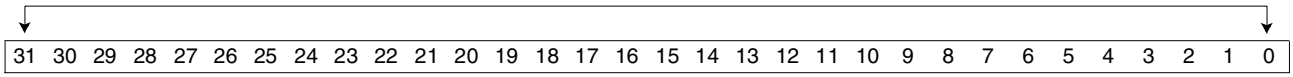
This register specifies the base of the prefetchable memory address range bits 63:32 and is used in conjunction with the prefetchable memory base register, the prefetchable memory limit register, and the prefetchable limit upper 32 bits register to specify a range of 64-bit addresses supported for prefetchable memory transactions on the PCI bus. Address bits 19:0 are assumed to be x‘0 0000’ for the base address.

Address Offset x‘28’

Access See individual fields

Reset Value x‘0000 0000’

Prefetchable Base Upper 32 Bits



Bit(s)	Access	Field Name and Description
31:0	RW	Address bits 63:32 of the base address for the address range of prefetchable memory operations that are passed from the primary to the secondary PCI bus.

5.2.4.25 Prefetchable Limit Upper 32 Bits Register

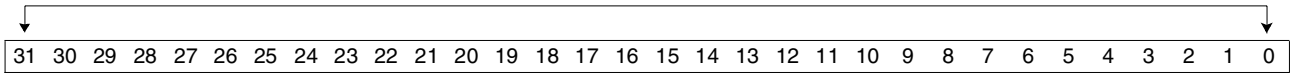
This register specifies the upper address of the prefetchable memory address range bits 63:32 and is used in conjunction with the prefetchable memory base register, the prefetchable memory limit register, and the prefetchable base upper 32 bits register to specify a range of 64-bit addresses supported for prefetchable memory transactions on the PCI bus. Address bits 19:0 are assumed to be x‘F FFFF’ for the limit address.

Address Offset x‘2C’

Access See individual fields

Reset Value x‘0000 0000’

Prefetchable Limit Upper 32 Bits



Bit(s)	Access	Field Name and Description
31:0	RW	Address bits 63:32 of the limit address for the address range of prefetchable memory operations that are passed from the primary to the secondary PCI bus.

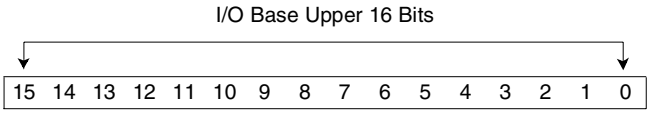
5.2.4.26 I/O Base Upper 16 Bits Register

This register specifies the base of the I/O address range bits 31:16 and is used in conjunction with the I/O base register, the I/O limit register, and I/O limit upper 16 bits register to specify a range of 32-bit addresses supported for I/O transactions on the PCI bus. Address bits 11:0 are assumed to be x‘000’ for the base address.

Address Offset x‘30’

Access See individual fields

Reset Value x‘0000’



Bit(s)	Access	Field Name and Description
15:0	RW	Address bits 31:16 of the base address for the address range of I/O operations that are passed from the primary to the secondary PCI bus.

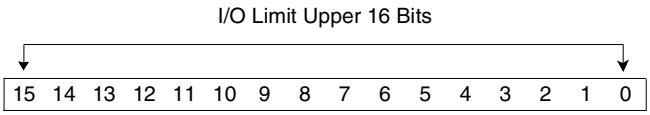
5.2.4.27 I/O Limit Upper 16 Bits Register

This register specifies the upper address of the I/O address range bits 31:16 and is used in conjunction with the I/O base register, I/O limit register and I/O base upper 16 bits register to specify a range of 32-bit addresses supported for I/O transactions on the PCI bus. Address bits 11:0 are assumed to be x‘FFF’ for the limit address.

Address Offset x‘32’

Access See individual fields

Reset Value x‘0000’

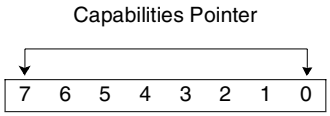


Bit(s)	Access	Field Name and Description
15:0	RW	Address bits 31:16 of the base address for the address range of I/O operations that are passed from the primary to the secondary PCI bus.

5.2.4.28 Capabilities Pointer Register

This register specifies a pointer to a capabilities list item in configuration space.

Address Offset x‘34’
Access Read only
Reset Value x‘80’



Bit(s)	Access	Field Name and Description
7:0	RO	Capabilities Pointer Read-only pointer to a capabilities list in configuration space.

5.2.4.29 Reserved Registers

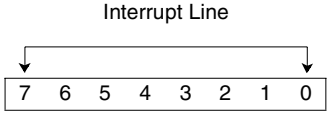
These registers are reserved and return zeros when read.

Address Offset x‘35’
Width 24 bits
Access Read only
Reset Value x‘000000’

5.2.4.30 Interrupt Line Register

This register is a read/write register that is used to communicate interrupt line routing information between initialization code and the device driver. If a bridge does not implement an interrupt signal pin, then the power on self test (POST) code must write x‘FF’ to this register.

Address Offset x‘3C’
Access Read/Write
Reset Value x‘XX’



Bit(s)	Access	Field Name and Description
7:0	RW	Interrupt Line POST code must initialize this register to x‘FF’.

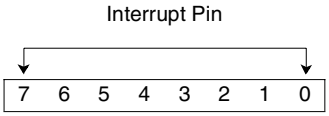
5.2.4.31 Interrupt Pin Register

This register is a read only register that returns x‘00’ when read because the bridge does not implement any interrupt pins.

Address Offset x‘3D’

Access Read only

Reset Value x‘00’



Bit(s)	Access	Field Name and Description
7:0	RO	Interrupt Pin Set to x‘00’.

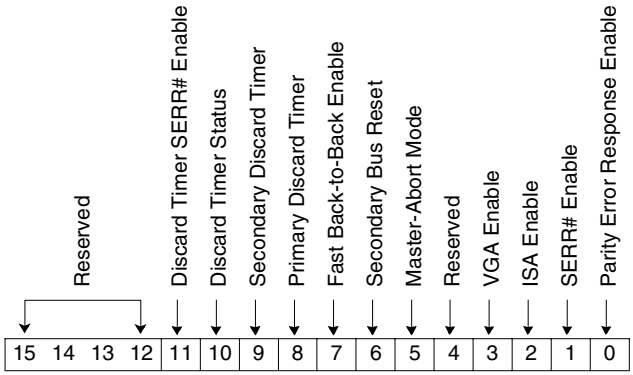
5.2.4.32 Bridge Control Register

This register provides extensions to the command register that are specific to a bridge. The bridge control register provides many of the same controls for the secondary interface that are provided by the command register for the primary interface. Some bits affect the operation of both bridge interfaces.

Address Offset x‘3E’

Access See individual bit fields. Reads and writes to this register behave normally for all bits except bit 10. Writes to bit 10 are slightly different as this bit can be reset, but not set. The bit is reset whenever the register is written, and the data in the corresponding bit location is a ‘1’.

Reset Value x‘0000’



Bit(s)	Access	Field Name and Description
15:12	RO	Reserved
11	RW	Discard Timer SERR# Enable 0 Do not assert SERR# on the primary interface as a result of the expiration of either the Primary Discard Timer or Secondary Discard Timer. 1 Assert SERR# on the primary interface as a result of the expiration of either the Primary Discard Timer or Secondary Discard Timer. This bit is ignored by a bridge in the PCI-X mode.
10	RW	Discard Timer Status 0 No discard timer error. 1 Discard timer error. This bit is never set for an interface that is in the PCI-X mode.
9	RW	Secondary Discard Timer 0 Secondary Discard Timer counts 2^{15} PCI clock cycles. 1 Secondary Discard Timer counts 2^{10} PCI clock cycles. Ignored by the bridge if the secondary interface is in the PCI-X mode.
8	RW	Primary Discard Timer 0 Primary Discard Timer counts 2^{15} PCI clock cycles. 1 Primary Discard Timer counts 2^{10} PCI clock cycles. Ignored by the bridge if the primary interface is in the PCI-X mode.
7	RO	Fast Back-to-Back Enable 0 Bridge does not generate fast back-to-back transactions.

Bit(s)	Access	Field Name and Description
6	RW	Secondary Bus Reset 0 Do not force the assertion of the secondary interface signal RST#. 1 Force the assertion of the secondary interface signal RST#.
5	RW	Master-Abort Mode 0 Do not report master-aborts, return x'FFFF FFFF' on reads and discard data on writes. 1 Report master-aborts by signaling target-abort if possible or by assertion of SERR# (if enabled). If in the PCI-X mode the bridge will return a split completion message and it is up to the host bridge to return x'FFFF FFFF' on any non-posted transaction when the non-posted transaction ends in a master abort.
4	RO	Reserved 0 Must return 0.
3	RW	VGA Enable 0 Do not forward VGA compatible memory and I/O addresses from the primary to the secondary interface unless they are enabled for forwarding by the defined I/O and memory address ranges. 1 Forward VGA compatible memory and I/O addresses from the primary interface to the secondary interface (if the I/O Enable and Memory Enable bits are set) independent of the I/O and memory address ranges and independent of the ISA Enable bit.
2	RW	ISA Enable 0 Forward downstream all I/O addresses in address range defined by I/O Base and I/O Limit registers. 1 Forward upstream ISA I/O addresses in address range defined by I/O Base and I/O Limit registers in the first 64 KB of PCI I/O address space (top 768 bytes of each 1 KB block).
1	RW	SERR# Enable 0 Disable the forwarding of secondary SERR# to primary SERR#. 1 Enable the forwarding of secondary SERR# to primary SERR#.
0	RW	Parity Error Response Enable 0 Ignore address and data parity errors on the secondary interface. 1 Enable parity error detection and reporting on the secondary interface. Controls the response to address and data parity errors on the secondary interface. If this bit is set, the bridge must take its normal action when a parity error is detected. If this bit is cleared, the bridge must ignore any parity errors that it detects and continue normal operation. In either case, the parity error detected bit of the secondary status register gets set if an address or data parity error is detected.

5.2.5 Device-Specific Configuration Space Registers

The following sections describe the individual registers of the device-specific configuration space region.

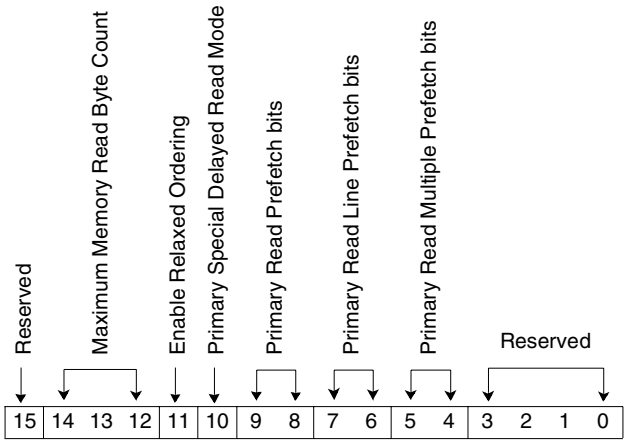
5.2.5.1 Primary Data Buffering Control Register

This register provides controls for memory read transactions that are initiated on the primary interface.

Address Offset x‘40’

Access See individual bit fields.

Reset Value x‘0020’



Bit(s)	Access	Field Name and Description
15	RO	Reserved
14:12	RW	Maximum Memory Read Byte Count These bits set the maximum byte count used by the bridge when generating read requests on the secondary bus in response to a memory read operation initiated on the primary bus (when the primary bus is in the PCI mode). Device drivers must not modify these bits without considering the impact on the rest of the system. These bits only have an effect if the prefetch mode bits for the PCI read command in use (bits (9:8), (7:6), or (5:4) below) are set to “full prefetch”. The most recent value of this register is used each time the bridge makes a new read request. The bits have the following meaning: 000 Default to 512 bytes. 001 128 bytes. 010 256 bytes. 011 512 bytes. 100 1024 bytes. 101 2048 bytes. 110 4096 bytes. 111 Default to 512 bytes.

Bit(s)	Access	Field Name and Description
11	RW	Enable Relaxed Ordering For a read operation, this bit enables read completions that occur after the first read completion to bypass posted writes and complete with a higher priority. Only read completions being presented on the primary interface are affected by this bit. 0 Conventional mode, read command relaxed ordering disabled: each target bus read completion transaction will be ordered with posted writes in the opposite direction (traveling the same direction as the completion data). see the PCI Bridge Specification 2.0, Table 5-2. 1 Conventional mode, read command relaxed ordering enabled: only the first read completion of a read transaction will be required to meet the PCI ordering requirement for delayed read completion vs. posted writes (PCI Bridge Specification 2.0, Table 5-2) all subsequent read completions for the same master transaction will be allowed to pass posted writes in the opposite direction. This bit is used only for requests issued in the PCI mode. For requests issued in the PCI-X mode the relaxed ordering bit in the attribute field takes precedence.
10	RW	Primary Special Delayed Read Mode Enable bit Allows a primary master to change memory read command code (MR, MRL, or MRM) after it has received retry to another memory read command code (MR, MRL, or MRM). 0 Retry any primary master which repeats its transaction with command code changes. 1 Completed MR, MRL, or MRM transaction on the secondary bus and the initiator (or another) master on the primary bus initiates a MR, MRL, or MRM transaction with the same address and byte enables, then the bridge will complete it normally. This bit is ignored when the primary interface is in the PCI-X mode.
9:8	RW	Primary Read prefetch mode bits (prefetchable range only) Controls prefetching for memory read transactions in prefetchable range that are initiated on the primary bus. 00 One cache line prefetch. 01 Reserved. 10 Full prefetch. 11 No prefetching, full handshake between initiator and target. Disconnect on first DWord. These bits are ignored when the primary interface is in the PCI-X mode.
7:6	RW	Primary Read Line prefetch mode bits Controls prefetching for Memory Read Line that are initiated on the primary bus. 00 One cache line prefetch. 01 Reserved. 10 Full prefetch. 11 Reserved. These bits are ignored when the primary interface is in the PCI-X mode.
5:4	RW	Primary Read Multiple prefetch mode bits Controls prefetching for Memory Read Multiple transactions that are initiated on the primary bus. 00 One cache line prefetch. 01 Reserved. 10 Full prefetch. 11 Reserved. These bits are ignored when the primary interface is in the PCI-X mode.
3:0	RO	Reserved.

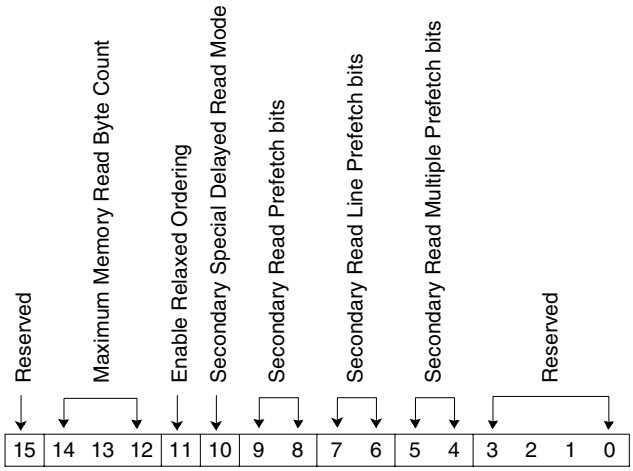
5.2.5.2 Secondary Data Buffering Control Register

This register provides controls for memory read transactions that are initiated on the secondary interface.

Address Offset x‘42’

Access See individual bit fields.

Reset Value x‘0020’



Bit(s)	Access	Field Name and Description
15	RO	Reserved
14:12	RW	Maximum Memory Read Byte Count These bits set the maximum byte count used by the bridge when generating read requests on the primary bus in response to a memory read operation initiated on the secondary bus (when the secondary bus is in the PCI mode). Device drivers must not modify these bits without considering the impact on the rest of the system. These bits only have an effect if the prefetch mode bits for the PCI read command in use (bits (9:8), (7:6), or (5:4) below) are set to “full prefetch”. The most recent value of this register is used each time the bridge makes a new read request. The bits have the following meaning: 000 Default to 512 bytes. 001 128 bytes. 010 256 bytes. 011 512 bytes. 100 1024 bytes. 101 2048 bytes. 110 4096 bytes. 111 Default to 512 bytes.

Bit(s)	Access	Field Name and Description
11	RW	Enable Relaxed Ordering For a read operation, this bit enables read completions that occur after the first read completion to bypass posted writes and complete with a higher priority. Only read completions being presented on the primary interface are affected by this bit. 0 Conventional mode, read command relaxed ordering disabled: each target bus read completion transaction will be ordered with posted writes in the opposite direction (traveling the same direction as the completion data). see the PCI Bridge Specification 2.0, Table 5-2. 1 Conventional mode, read command relaxed ordering enabled: only the first read completion of a read transaction will be required to meet the PCI ordering requirement for delayed read completion vs. posted writes (PCI Bridge Specification 2.0, Table 5-2) all subsequent read completions for the same master transaction will be allowed to pass posted writes in the opposite direction. This bit is used only for requests issued in the PCI mode. For requests issued in the PCI-X mode the relaxed ordering bit in the attribute field takes precedence.
10	RW	Secondary Special Delayed Read Mode Enable bit Allows a secondary master to change the memory read command (MR, MRL, or MRM) code after it has received retry to another memory read command (MR, MRL, or MRM) code. 0 Retry any secondary master which repeats its transaction with command code changes. 1 Completed a MR, MRL, or MRM transaction on the secondary bus and the initiator (or another) master on the secondary bus initiates a MR, MRL, or MRM transaction with the same address and byte enables, then the bridge will complete it normally. This bit is ignored when the secondary interface is in the PCI-X mode.
9:8	RW	Secondary Read prefetch mode bits Controls prefetching for memory read transactions, that are initiated on the secondary bus. 00 One cache line prefetch. 01 Reserved. 10 Full prefetch. 11 No prefetching, full handshake between initiator and target. Disconnect on first DWord. These bits are ignored when the secondary interface is in the PCI-X mode.
7:6	RW	Secondary Read Line prefetch mode bits Controls prefetching for memory read line transactions, that are initiated on the secondary bus. 00 One cache line prefetch. 01 Reserved. 10 Full prefetch. 11 Reserved. These bits are ignored when the secondary interface is in the PCI-X mode.
5:4	RW	Secondary Read Multiple prefetch mode bits Controls prefetching for memory read multiple transactions that are initiated on the secondary bus. 00 One cache line prefetch. 01 Reserved. 10 Full prefetch. 11 Reserved. These bits are ignored when the secondary interface is in the PCI-X mode.
3:0	RO	Reserved.

5.2.5.3 Miscellaneous Control Register

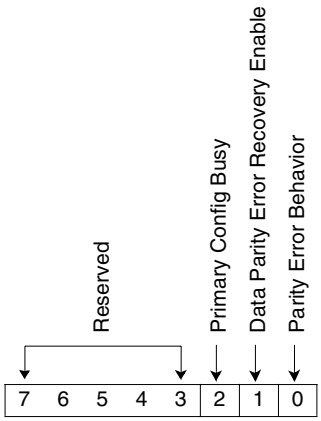
This register provides controls for miscellaneous functions, such as handling parity errors, in the bridge.

Address Offset x‘44’

Access Read/Write

Reset Value x‘03’ When P_CFG_BUSY (pin C6) is tied low. *Table 7-3* on page 113 for details of strapping considerations.

x‘07’ When P_CFG_BUSY (pin C6) is tied high. *Table 7-3* on page 113 for details of strapping considerations.



Bit(s)	Access	Field Name and Description
7:3	RO	Reserved
2	RW	Primary Config Busy This bit has R/W access from the secondary bus only. Configuration commands received from the primary bus can read, but cannot write this bit. The reset value of this bit depends on how the P_CFG_BUSY strapping pin is tied. When this bit is a b'1', all type 0 configuration commands received on the primary bus are retried. 0 Type 0 configuration commands accepted normally on the primary bus. 1 Type 0 configuration commands retried on the primary bus.
1	RW	Data Parity Error Recovery Enable 0 Allow the bridge to pass parity errors through the bridge. 1 Cause SERR# to be asserted whenever either Master Data Parity Error bit (bit 8 in either the Status or the Secondary Status register) is set. The default value after reset is b‘1’.
0	RW	Parity Error Behavior This bit defines the bridge’s behavior when detecting a data parity error on a non-posted write transaction. 0 The bridge will pass the corrupted data sequence, PERR# will be asserted (if enabled), but the bridge will not compare the data and BE# for performing completion on the initiating bus. 1 The transaction will be completed on the originating bus, PERR# will be asserted (if enabled), the appropriate status bits will be set, the data will be discarded and no request will be enqueued. The default value after reset is b‘1’.

5.2.5.4 Arbiter Mode Register

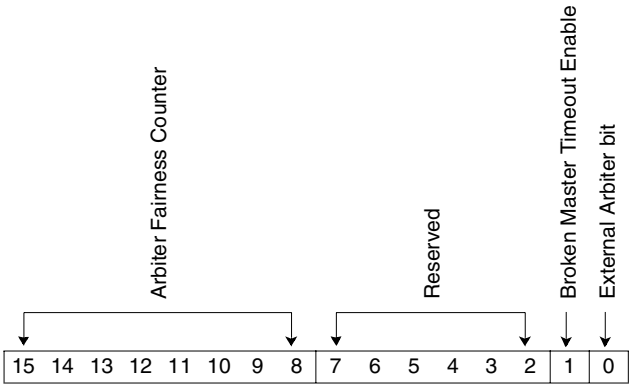
This register provides controls for the secondary bus arbitration logic on the bridge.

Address Offset x‘50’

Access See bit descriptions for details

Reset Value x’0800’ When S_INT_ARB_EN# (pin T21) is tied low. See *Table 7-3* on page 113 for details of strapping considerations.

Reset Value x’0801’ When S_INT_ARB_EN# (pin T21) is tied high. See *Table 7-3* on page 113 for details of strapping considerations.



Bit(s)	Access	Field Name and Description
15:8	RW	Arbiter Fairness Counter This is the initialization value of a counter used by the internal arbiter. It controls the number of PCI bus cycles that the arbiter holds a device’s PCI bus grant active after detecting a PCI bus request from another device. The counter is reloaded whenever a new PCI bus grant is asserted. For every new PCI bus grant, the counter is armed to decrement when it detects the new fall of FRAME#. If the arbiter fairness counter is set to x‘00’, the arbiter will not remove a device’s PCI bus grant until the device has deasserted its PCI bus request. The reset value is x‘08’.
7:2	RO	Reserved
1	RW	Broken Master Timeout Enable This bit enables the internal arbiter to count 16 PCI bus cycles while waiting for FRAME# to become active when a device’s PCI Bus Grant is active and the PCI bus is idle. If the Broken Master Timeout expires the PCI Bus Grant for the device is de-asserted. 0 Broken Master Timeout disabled. 1 Broken Master Timeout enabled. The reset value is b‘0’.
0	RO	External Arbiter bit This status bit reflects whether the bridge is in internal or external arbiter mode. The value is set at reset time, according to the polarity of the S_INT_ARB_EN# I/O pin. 0 Internal arbiter in control. 1 External arbiter in control.

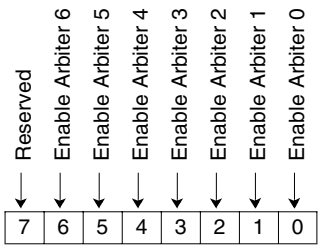
5.2.5.5 Arbiter Enable Register

This register enables arbitration for the requestors of the internal secondary bus arbitration logic on the bridge.

Address Offset x‘54’

Access Read/Write

Reset Value x‘7F’



Bit(s)	Access	Field Name and Description
7	RO	Reserved
6	RW	Enable Arbiter 6 0 Disable arbitration. 1 Enable arbitration.
5	RW	Enable Arbiter 5 0 Disable arbitration. 1 Enable arbitration.
4	RW	Enable Arbiter 4 0 Disable arbitration. 1 Enable arbitration.
3	RW	Enable Arbiter 3 0 Disable arbitration. 1 Enable arbitration.
2	RW	Enable Arbiter 2 0 Disable arbitration. 1 Enable arbitration.
1	RW	Enable Arbiter 1 0 Disable arbitration. 1 Enable arbitration.
0	RW	Enable Arbiter 0 This bit enables arbitration for the internal bridge requests. 0 Disable arbitration. 1 Enable arbitration.

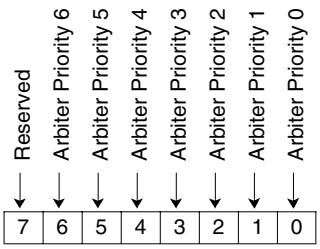
5.2.5.6 Arbiter Priority Register

This register indicates whether high or low priority is assigned to the corresponding request of the internal secondary bus arbitration logic on the bridge.

Address Offset x‘58’

Access Read/Write

Reset Value x‘01’



Bit(s)	Access	Field Name and Description
7	RO	Reserved
6	RW	Arbiter Priority 6 0 Low priority request. 1 High priority request.
5	RW	Arbiter Priority 5 0 Low priority request. 1 High priority request.
4	RW	Arbiter Priority 4 0 Low priority request. 1 High priority request.
3	RW	Arbiter Priority 3 0 Low priority request. 1 High priority request.
2	RW	Arbiter Priority 2 0 Low priority request. 1 High priority request.
1	RW	Arbiter Priority 1 0 Low priority request. 1 High priority request.
0	RW	Arbiter Priority 0 This bit indicates the priority for the internal bridge requests. 0 Low priority request. 1 High priority request.

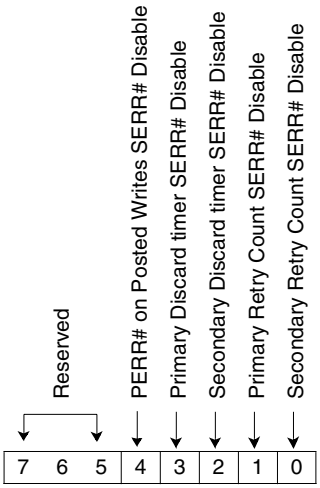
5.2.5.7 SERR# Disable Register

This register controls the assertion of the SERR# signal on the primary bus due to certain errors.

Address Offset x‘5C’

Access Read/Write

Reset Value x‘00’



Bit(s)	Access	Field Name and Description
7:5	RO	Reserved
4	RW	PERR# on Posted Writes SERR# Disable Controls the SERR# assertion when a PERR# is detected on the destination bus on an error free posted write. 0 Assert SERR# and set bit 14 of the status register if the SERR# enable bit 8 in the command register is set. Discard the delayed transaction. 1 Disable the assertion of SERR#.
3	RW	Primary Discard Timer SERR# Disable Controls the SERR# assertion when the primary discard timer has expired. 0 Assert SERR# and update status bit 14 in the status register if the primary discard timer expires, the SERR# enable bit 8 in the Command register is set, and bit 11 of the bridge control register is set. Discard the delayed transaction and set bit 3 of the retry and timer status register. 1 Disable the assertion of SERR# if the primary discard timer expires. Discard the delayed transaction and set bit 3 of the retry and timer status register.
2	RW	Secondary Discard Timer SERR# Disable Controls the SERR# assertion when the secondary discard timer has expired. 0 Assert SERR# and update status bit 14 in the Status register if the secondary discard timer expires, the SERR# enable bit 8 in the command register is set, and bit 11 of the bridge control register is set. Discard the delayed transaction and set bit 2 of the retry and timer status register. 1 Disable the assertion of SERR# if the secondary discard timer expires. Discard the delayed transaction and set bit 2 of the retry and timer status register.

Bit(s)	Access	Field Name and Description
1	RW	Primary Retry Count SERR# Disable Controls the SERR# assertion when the primary retry counter has expired. 0 Assert SERR# and update status bit 14 in the status register if the primary retry counter expires and SERR# enable bit 8 in the command register is set. Discard the transaction and set bit 1 of the retry and timer status register. 1 Disable the assertion of SERR# if the primary retry counter expires. Discard the transaction and set bit 1 of the retry and timer status register.
0	RW	Secondary Retry Count SERR# Disable Controls the SERR# assertion when the secondary retry counter has expired. 0 Assert SERR# and update status bit 14 in the status register if the secondary retry counter expires and SERR# enable bit 8 in the command register is set. Discard the transaction and set bit 1 of the retry and timer status register. 1 Disable the assertion of SERR# if the secondary retry counter expires. Discard the transaction and set bit 1 of the retry and timer status register.

5.2.5.8 Primary Retry Counter Register

This register defines the number of retries that the bridge receives on the secondary bus for a requested transaction, before its internal retry counter expires. When the counter expires, the bridge discards the request, and, if enabled, will issue SERR# on the primary bus.

This mechanism prevents deadlock when the target is unable to receive the request, and allows recovery through the use of SERR#.

The only allowed values are:

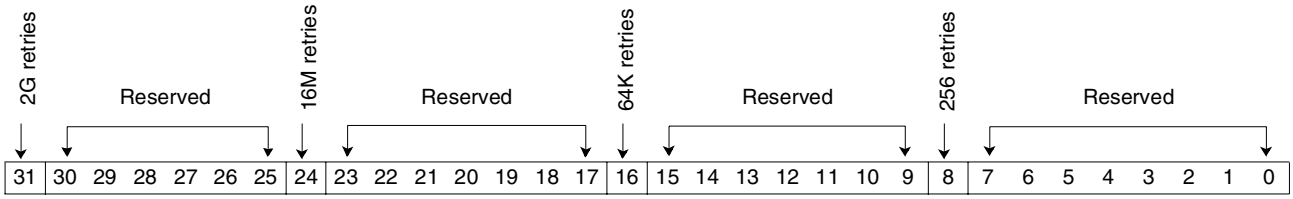
- x‘0000 0000’: Counting disabled (No expiration)
- x‘0000 0100’: 256 retries before expiration
- x‘0001 0000’: 64K retries before expiration
- x‘0100 0000’: 16M retries before expiration
- x‘8000 0000’: 2G retries before expiration

Address Offset x‘60’

Access Read/Write

Reset Value x‘0000 0000’

Restrictions Only one bit can be set at any time, setting multiple bits results in the smaller retry count.



Bit(s)	Access	Field Name and Description
31	RW	This bit used to indicate 2G retries before expiration.
30:25	RO	Reserved.
24	RW	This bit used to indicate 16M retries before expiration.
23:17	RO	Reserved.
16	RW	This bit used to indicate 64K retries before expiration.
15:9	RO	Reserved.
8	RW	This bit used to indicate 256 retries before expiration.
7:0	RO	Reserved.

5.2.5.9 Secondary Retry Counter Register

This register defines the number of retries that the bridge will receive on the secondary bus for a requested transaction, before its internal retry counter expires. When the counter expires, the bridge discards the request, and, if enabled, issues SERR# on the primary bus.

This mechanism prevents deadlock when the target is unable to receive the request, and allows recovery through the use of SERR#.

The only allowed values are:

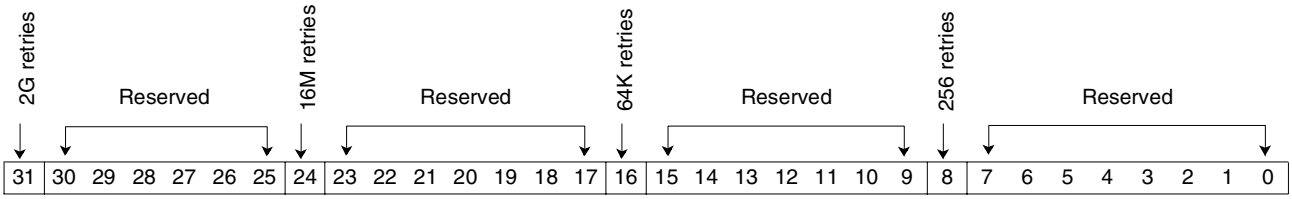
- x‘0000 0000’: Counting disabled (No expiration)
- x‘0000 0100’: 256 retries before expiration
- x‘0001 0000’: 64K retries before expiration
- x‘0100 0000’: 16M retries before expiration
- x‘8000 0000’: 2G retries before expiration

Address Offset x‘64’

Access Read/Write

Reset Value x‘0000 0000’

Restrictions Only one bit can be set at any time, setting multiple bits results in the smaller retry count.



Bit(s)	Access	Field Name and Description
31	RW	This bit used to indicate 2G retries before expiration.
30:25	RO	Reserved.
24	RW	This bit used to indicate 16M retries before expiration.
23:17	RO	Reserved.
16	RW	This bit used to indicate 64K retries before expiration.
15:9	RO	Reserved.
8	RW	This bit used to indicate 256 retries before expiration.
7:0	RO	Reserved.

5.2.5.10 Discard Timer Control Register

This register controls the duration and enabling of the discard timers.

There is a unique discard timer for every delayed transaction enqueued in the bridge. A discard timer begins counting at the receipt of a delayed transaction in the conventional PCI mode. The timer is reset each time the initiating bus master retries the transaction. The timer is stopped and reset when the transaction is completed. If a discard timer expires, the associated transaction is discarded, data buffers and control resources are freed, and the appropriate status bit is set in the retry and timer status register.

Address Offset x‘68’

Access Read/Write

Reset Value x‘00’

Bit(s)	Access	Field Name and Description
7:4	RO	Reserved
3	RW	Primary Discard Timer Short Duration Controls the number of PCI clocks that the primary discard timer allows before it expires. 0 Use bit 8 of the bridge control register to indicate how many PCI clocks should be allowed before the primary discard timer expires. 1 Use 2^6 PCI clocks for the value of the primary discard timer.
2	RW	Secondary Discard Timer Short Duration Controls the number of PCI clocks that the secondary discard timer allows before it expires. 0 Use bit 9 of the bridge control register to indicate how many PCI clocks should be allowed before the secondary discard timer expires. 1 Use 2^6 PCI clocks for the value of the secondary discard timer.
1	RW	Primary Discard Timer Disable Controls the disabling of the primary discard timer in conjunction with bit 11 of the bridge control register. 0 Enable the primary discard timer. 1 Disable primary discard timer.
0	RW	Secondary Discard Timer Disable Controls the disabling of the secondary discard timer in conjunction with bit 11 of the bridge control register. 0 Enable the secondary discard timer. 1 Disable secondary discard timer.

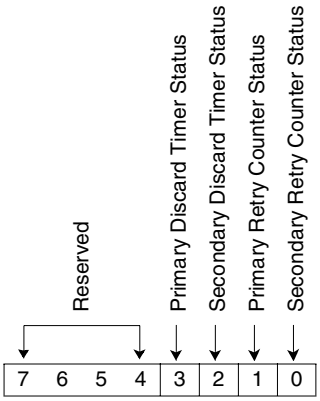
5.2.5.11 Retry and Timer Status Register

This register indicates if a retry counter or a discard timer has expired since the register was last reset.

Address Offset x‘6C’

Access See individual bit descriptions. Reads to this register behave normally. Writes are slightly different in that bits can be reset, but not set. A bit is reset whenever the register is written, and the data in the corresponding bit location is a ‘1’.

Reset Value x‘00’



Bit(s)	Access	Field Name and Description
7:4	RO	Reserved
3	RW	Primary Discard Timer Status 0 The primary discard timer has not expired since the last reset. 1 The primary discard timer has expired since the last reset.
2	RW	Secondary Discard Timer Status 0 The secondary discard timer has not expired since the last reset. 1 The secondary discard timer has expired since the last reset.
1	RW	Primary Retry Counter Status 0 The primary retry counter has not expired since the last reset. 1 The primary retry counter has expired since the last reset.
0	RW	Secondary Retry Counter Status 0 The secondary retry counter has not expired since the last reset. 1 The secondary retry counter has expired since the last reset.

5.2.5.12 Opaque Memory Enable Register

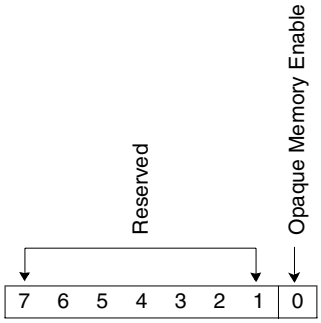
This register enables the opaque memory base, opaque memory limit, opaque memory base upper 32 bits, and the opaque memory limit upper 32 bits registers. These registers specify a range of 64-bit memory addresses that are used exclusively on the secondary PCI bus and are not to be accepted by the bridge on either the primary or secondary interfaces.

Address Offset x‘70’

Access Read/Write

Reset Value x‘00’ When OPAQUE_EN (pin AA18) is tied low. See *Table 7-3* on page 113 for details of strapping considerations.

Reset Value x‘01’ When OPAQUE_EN (pin AA18) is tied high. See *Table 7-3* on page 113 for details of strapping considerations.



Bit(s)	Access	Field Name and Description
7:1	RO	Reserved
0	RW	Opaque Memory Enable 0 Disable the opaque memory address range. 1 Enable the opaque memory address range. This bit is reset according to the tie value of module pin AA18, OPAQUE_EN.

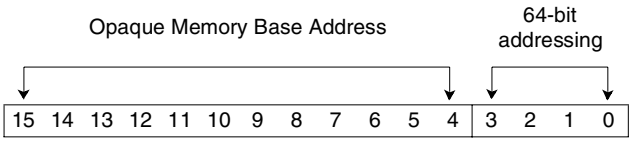
5.2.5.13 Opaque Memory Base Register

This register specifies base address bits 31:20 of the opaque memory address range. It is used in conjunction with the opaque memory limit register, the opaque memory base upper 32 bits register, and the opaque memory limit upper 32 bits register to specify a range of 64-bit memory addresses that are used exclusively on the secondary bus. These memory addresses will not be accepted by the bridge on the primary or the secondary buses. This address range is enabled by bit 0 of the opaque memory enable register. Bits 19:0 of the base address are assumed to be x‘0 0000’. This register also specifies that the bridge supports 64-bit opaque memory addressing.

Address Offset x‘74’

Access See individual fields

Reset Value x‘0001’



Bit(s)	Access	Field Name and Description
15:4	RW	Opaque Memory Base Address Address bits 31:20 of the base address for the opaque memory address range. Memory operations in this range are not accepted by the bridge on either the primary or secondary interfaces.
3:0	RO	64-bit addressing Set to b‘0001’ to indicate support of 64-bit addressing.

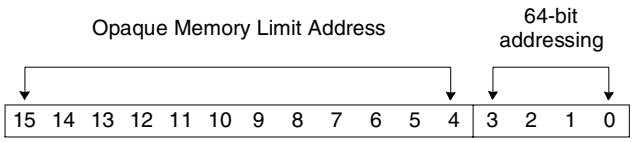
5.2.5.14 Opaque Memory Limit Register

This register specifies upper address bits 31:20 of the opaque memory address range. It is used in conjunction with the opaque memory base register, the opaque memory base upper 32 bits register, and the opaque memory limit upper 32 bits register to specify a range of 64-bit addresses that are used exclusively on the secondary bus. These memory addresses will not be accepted by the bridge on the primary or the secondary buses. This address range is enabled by bit 0 of the opaque memory enable register. Bits 19:0 of the limit address are assumed to be x'F FFFF'. This register also specifies that the bridge supports 64-bit opaque memory addressing.

Address Offset x'76'

Access See individual fields

Reset Value x'FFF1'



Bit(s)	Access	Field Name and Description
15:4	RW	Opaque Memory Limit Address Address bits 31:20 of the limit address for the opaque memory address range. Memory operations in this range are not accepted by the bridge on either the primary or secondary interfaces.
3:0	RO	Set to b'0001' to indicate support of 64-bit addressing.

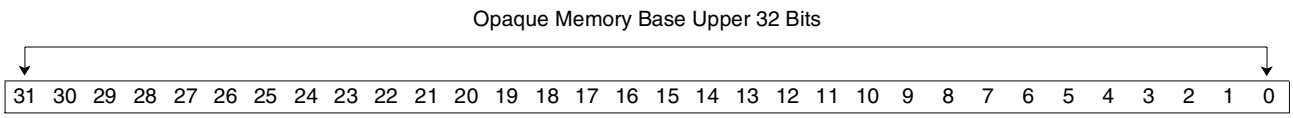
5.2.5.15 Opaque Memory Base Upper 32 Bits Register

This register specifies bits 63:32 of the base address of the opaque memory address range. It is used in conjunction with the opaque memory base register, the opaque memory limit register, and the opaque memory limit upper 32 bits register to specify a range of 64-bit addresses that are used exclusively on the secondary bus. These memory addresses will not be accepted by the bridge on the primary or the secondary buses. This address range is enabled by bit 0 of the opaque memory enable register.

Address Offset x'78'

Access Read/Write

Reset Value x'FFFF FFFF'



Bit(s)	Access	Field Name and Description
31:0	RW	Opaque Memory Base Upper 32 Bits Address bits 63:32 of the base address for the opaque memory address range. Memory operations in this range are not accepted by the bridge on either the primary or secondary interfaces.

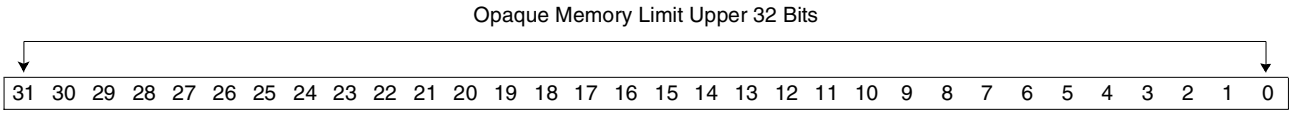
5.2.5.16 Opaque Memory Limit Upper 32 Bits Register

This register specifies upper address bits 63:32 of the opaque memory address range. It is used in conjunction with the opaque memory base register, the opaque memory limit register, and the opaque memory base upper 32 bits register to specify a range of 64-bit addresses that are used exclusively on the secondary bus. These memory addresses will not be accepted by the bridge on the primary or the secondary buses. This address range is enabled by bit 0 of the Opaque Memory Enable register.

Address Offset x‘7C’

Access Read/Write

Reset Value x‘FFFF FFFF’



Bit(s)	Access	Field Name and Description
31:0	RW	Opaque Memory Limit Upper 32 Bits Address bits 63:32 of the limit address for the opaque memory address range. Memory operations in this range are not accepted by the bridge on either the primary or secondary interfaces.

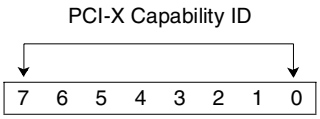
5.2.5.17 PCI-X ID Register

This register identifies this register set in the capabilities list as a PCI-X register set. It is read-only, returning x‘07’ when read.

Address Offset x‘80’

Access Read only

Reset Value x‘07’



Bit(s)	Access	Field Name and Description
7:0	RO	PCI-X Capability ID Returns x‘07’ when read indicating that this register set of the Capabilities List is a PCI-X register set.

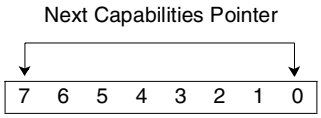
5.2.5.18 Next Capabilities Pointer Register

This register of the PCI-X register set is a read-only register returning x‘90’ when read, indicating that there are more list items in the capabilities list.

Address Offset x‘81’

Access Read only

Reset Value x‘90’



Bit(s)	Access	Field Name and Description
7:0	RO	Next Capabilities Pointer Returns x‘90’ when read indicating that there are more list items in the Capabilities List.

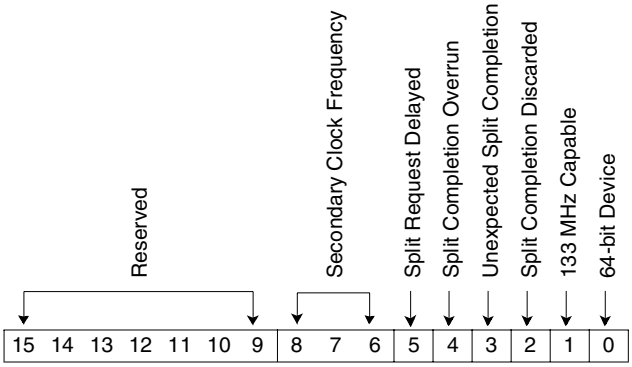
5.2.5.19 PCI-X Secondary Status Register

This register reports status information about the secondary interface.

Address Offset x‘82’

Access See individual bit descriptions. Reads to this register behave normally. Writes are slightly different in that bits can be reset, but not set. A bit is reset whenever the register is written, and the data in the corresponding bit location is a ‘1’.

Reset Value x‘0003’



Bit(s)	Access	Field Name and Description
15:9	RO	Reserved.
8:6	RO	Secondary Clock Frequency This register enables the configuration software to determine to what mode and (in the PCI-X mode) what frequency the bridge set the secondary bus to the last time the secondary RST# was asserted. This is the same information the bridge used to create the PCI-X initialization pattern on the secondary bus the last time the secondary RST# was asserted. Value — Max Clock Frequency (MHz) — Minimum Clock Period (ns) 000 — Conventional mode. — N/A 001 — 66 — 15 010 — 100 — 10 011 — 133 — 7.5 100 — Reserved. — Reserved. 101 — Reserved. — Reserved. 110 — Reserved. — Reserved. 111 — Reserved. — Reserved.
5	RW	Split Request Delayed This bit is set any time the bridge has a request to forward a transaction to the secondary bus, but cannot because there is not enough room within the limit specified in the split transaction commitment limit field in the downstream split transaction control register. It is used by algorithms that optimize the setting of the downstream split transaction commitment limit register. 0 — The bridge has not delayed a split request. 1 — The bridge has delayed a split request.

Bit(s)	Access	Field Name and Description
4	RW	Split Completion Overrun This bit is set if the bridge terminates a split completion on the secondary bus with retry or disconnect at next ADB because the bridge buffers are full. It is used by algorithms that optimize the setting of the downstream split transaction commitment limit register. 0 The bridge has accepted all split completions. 1 The bridge has terminated a split completion with retry or disconnect at next ADB because the bridge buffers were full.
3	RW	Unexpected Split Completion This bit is set if an unexpected split completion with a requester ID equal to the bridge's secondary bus number, device number x'00', and function number 0 is received on the bridge's secondary interface. 0 No unexpected split completion has been received. 1 An unexpected split completion has been received.
2	RW	Split Completion Discarded This bit is set if the bridge discards a split completion moving toward the secondary bus because the requester would not accept it. 0 No split completion has been discarded. 1 A split completion has been discarded.
1	RO	133 MHz Capable This bit is read-only and is a b'1' indicating that this bridge is capable of 133 MHz operation on the secondary interface.
0	RO	64-bit Device This bit is read-only and is a b'1' indicating that the width of the bridge's secondary AD interface is 64 bits.

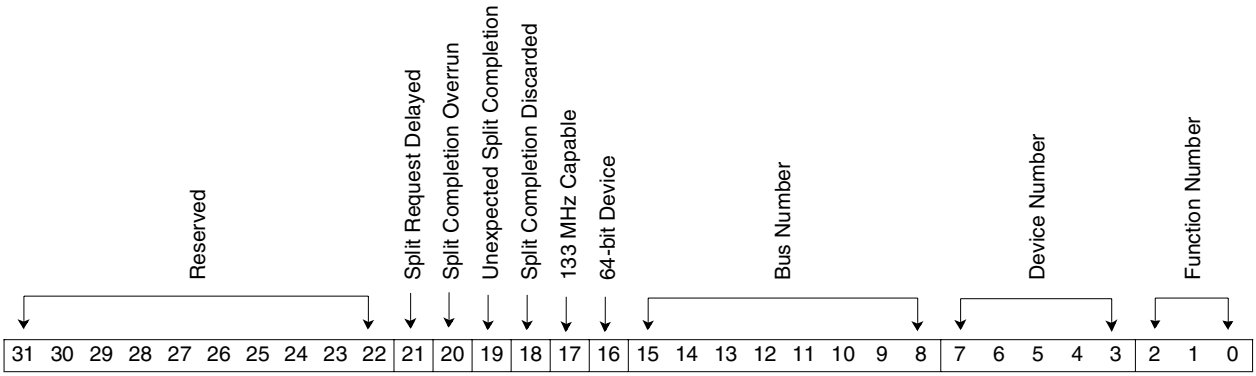
5.2.5.20 PCI-X Bridge Status Register

This register identifies the capabilities and current operating mode of the bridge on its primary bus.

Address Offset x‘84’

Access See individual bit descriptions. Reads to this register behave normally. Writes are slightly different in that bits can be reset, but not set. A bit is reset whenever the register is written, and the data in the corresponding bit location is a ‘1’.

Reset Value x‘0003 00F8’



Bit(s)	Access	Field Name and Description
31:22	RO	Reserved
21	RW	Split Request Delayed This bit is set any time the bridge has a request to forward a transaction to the primary bus, but cannot because there is not enough room within the limit specified in the split transaction commitment limit field in the upstream split transaction control register. It is used by algorithms that optimize the setting of the upstream split transaction commitment limit register. 0 The bridge has not delayed a split request. 1 The bridge has delayed a split request.
20	RW	Split Completion Overrun This bit is set if the bridge terminates a Split Completion on the primary bus with retry or disconnect at next ADB because the bridge buffers are full. It is used by algorithms that optimize the setting of the upstream split transaction commitment limit register. 0 The bridge has accepted all split completions. 1 The bridge has terminated a split completion with retry or disconnect at next ADB because the bridge buffers were full.
19	RW	Unexpected Split Completion This bit is set if an unexpected split completion with a requester ID equal to the bridge’s primary bus number, device number, and function number is received on the bridge’s primary interface. 0 No unexpected split completion has been received. 1 An unexpected split completion has been received.
18	RW	Split Completion Discarded This bit is set if the bridge discards a split completion moving toward the primary bus because the requester would not accept it. 0 No split completion has been discarded. 1 A split completion has been discarded.

Bit(s)	Access	Field Name and Description
17	RO	133 MHz Capable This bit is read-only and is a b‘1’ indicating that this bridge is capable of 133 MHz operation on the primary interface.
16	RO	64-bit Device This bit is read-only and is a b‘1’ indicating that the width of the bridge’s primary AD interface is 64 bits.
15:8	RO	Bus Number These bits are read for diagnostic purposes only. It is an additional address from which the contents of the primary bus number register in the Type x‘01’ configuration space header is read. The bridge uses the bus number, device number, and function number fields to create the completer ID when responding with a split completion to a read of and internal bridge register. These fields are also used for cases when one interface is in conventional mode and the other is in the PCI-X mode.
7:3	RO	Device Number These bits are read for diagnostic purposes only. They indicate the number of this device, which is the number in the device number field (AD[15:11]) of the address of a Type 0 configuration transaction that is assigned to this bridge by the connection of the system hardware. The bridge uses this number as described for the bus number field above. Each time the bridge is addressed by a configuration write transaction, the bridge updates this register with the contents of AD[15:11] of the address phase of the configuration write, regardless of which register in the bridge is addressed by the transaction. The bridge is addressed by a configuration write transaction if all of the following are true: • The transaction uses a configuration write command. • IDSEL is asserted during the address phase. • AD[1:0] are b‘00’ (Type 0 configuration transaction). • AD[10:08] of the configuration address contain the appropriate function number. • State after RST# is x‘1F’.
2:0	RO	Function Number These bits are read for diagnostic purposes only. They indicate the number of this function, which is the number in the function number field (AD[10:08]) of the address of a Type 0 configuration transaction to which this bridge responds. The bridge uses this number as described for the bus number field above. State after RST# is b‘000’.

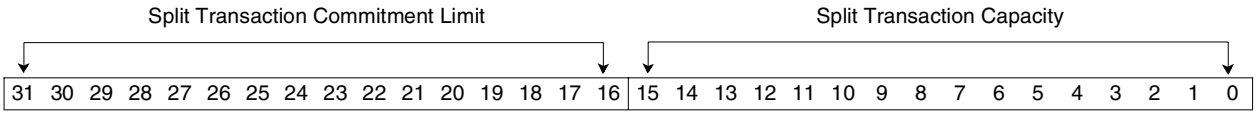
5.2.5.21 Secondary Bus Upstream Split Transaction Register

This register controls the behavior of the bridge buffers for forwarding split transactions from a secondary bus requester to a primary bus completer. When the completer bus is in the PCI-X mode, the split transaction commitment limit field only affects the byte count used when issuing read requests.

Address Offset x‘88’

Access See bit descriptions

Reset Value x‘0020 0020’



Bit(s)	Access	Field Name and Description
31:16	RW	Split Transaction Commitment Limit This field indicates the cumulative sequence size for all memory read transactions forwarded by the bridge from requesters on the secondary bus addressing completers on the primary bus. This field indicates the size of the commitment limit in units of ADQs. Software is permitted to program this field to any value greater than or equal to the contents of the split transaction capacity field. A value less than the contents of the split transaction capacity field causes unspecified results. If this field is set to x‘FFFF’, the bridge is permitted to forward all split requests of any size regardless of the amount of buffer space available. A value of x‘0100’ or greater will cause the bridge to forward accepted split requests of any size regardless of the amount of buffer space available. Software is permitted to change this field at any time. The most recent value of the field is used each time the bridge forwards a split transaction. State after RST# is the same as the split transaction capacity field.
15:0	RO	Split Transaction Capacity This read-only field indicates the size of the buffer (in number of ADQs) for storing split completions for memory reads. This applies to requesters on the secondary bus addressing completers on the primary bus. The bridge returns x‘0020’ to indicate that there are 32 ADQs (4K bytes) available buffer space.

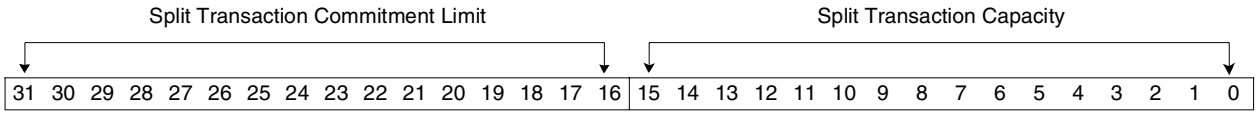
5.2.5.22 Primary Bus Downstream Split Transaction Register

This register controls the behavior of the bridge buffers when forwarding split transactions from a primary bus requester to a secondary bus completer. When the completer bus is in the PCI-X mode, the split transaction commitment limit field only affects the byte count used when issuing read requests.

Address Offset x‘8C’

Access See bit descriptions

Reset Value x‘0020 0020’



Bit(s)	Access	Field Name and Description
31:16	RW	Split Transaction Commitment Limit This field indicates the cumulative sequence size for all memory read transactions forwarded by the bridge from requesters on the primary bus addressing completers on the secondary bus. This field indicates the size of the commitment limit in units of ADQs. Software is permitted to program this field to any value greater than or equal to the contents of the split transaction capacity field. A value less than the contents of the split transaction capacity field causes unspecified results. If this field is set to x‘FFFF’, the bridge is permitted to forward all split requests of any size regardless of the amount of buffer space available. A value of x‘0100’ or greater will cause the bridge to forward accepted split requests of any size regardless of the amount of buffer space available. Software is permitted to change this field at any time. The most recent value of the field is used each time the bridge forwards a split transaction. The state of this field after a RST# is the same as the split transaction capacity field.
15:0	RO	Split Transaction Capacity This read-only field indicates the size of the buffer (in number of ADQs) for storing split completions for memory reads for requesters on the primary bus addressing completers on the secondary bus. The bridge returns x‘0020’ to indicate that there are 32 ADQs (4K bytes) available buffer space.

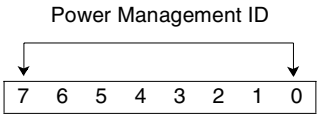
5.2.5.23 Power Management ID Register

This register identifies this register set in the capabilities list as a power management register set. It is read-only, returning x‘01’ when read.

Address Offset x‘90’

Access Read only

Reset Value x‘01’



Bit(s)	Access	Field Name and Description
7:0	RO	Power Management ID Read-only. Returns x‘01’ when read indicating that this register set of the capabilities list is a power management register set.

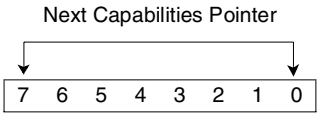
5.2.5.24 Next Capabilities Pointer Register

This register of the power management register set is a read-only register returning x‘00’ when read indicating that there are no more list items in the capabilities list.

Address Offset x‘91’

Access Read only

Reset Value x‘00’



Bit(s)	Access	Field Name and Description
7:0	RO	Next Capabilities Pointer Returns x‘00’ when read indicating that there are no more list items in the capabilities list.

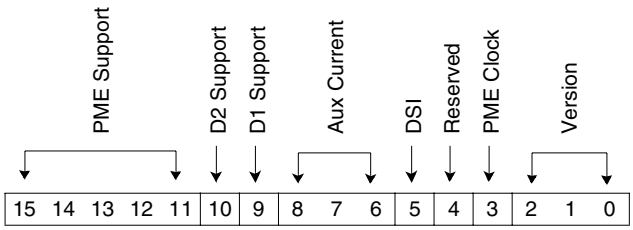
5.2.5.25 Power Management Capabilities Register

This register reports information about the capabilities of the secondary interface with regard to power management functions.

Address Offset x‘92’

Access Read only

Reset Value x‘0002’



Bit(s)	Access	Field Name and Description
15:11	RO	PME Support Forced to b‘00000’ to indicate that this device does not support the PME# pin.
10	RO	D2 Support Forced to b‘0’ to indicate that this device does not support the D2 power management state.
9	RO	D1 Support Forced to b‘0’ to indicate that this device does not support the D1 power management state.
8:6	RO	Aux Current Forced to b‘000’ to indicate that this device does not support PME# generation in the $D3_{cold}$ power management state.
5	RO	DSI The device specific initialization bit reads as b‘0’ to indicate that no special initialization of this function beyond the standard PCI configuration header is required following transition to the D0 un-initialized state.
4	RO	Reserved.
3	RO	PME Clock Forced to b‘0’ to indicate that this device does not support PME# generation.
2:0	RO	Version Forced to b‘010’ to indicate that this device complies with Revision 2.0 of the *PCI Power Management Interface Specification*.

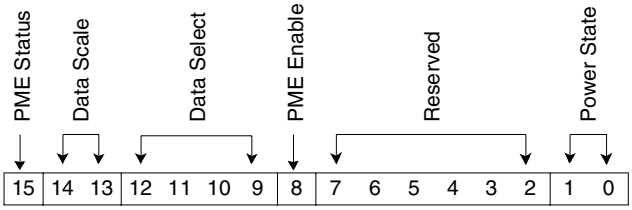
5.2.5.26 Power Management Control/Status Register

This register reports status information about the secondary interface.

Address Offset x‘94’

Access See bit descriptions

Reset Value x‘0000’



Bit(s)	Access	Field Name and Description
15	RO	PME Status Forced to b‘0’ to indicate that this device does not support the PME# pin.
14:13	RO	Data Scale Forced to b‘00’ to indicate that this device does not implement the Data register.
12:9	RO	Data Select Forced to b‘0000’ to indicate that this device does not implement the Data register.
8	RO	PME Enable Forced to b‘0’ to indicate that this device does not support PME# generation.
7:2	RO	Reserved.
1:0	RW	Power State This 2-bit field is used to both determine and reflect the current power state of the device. If an un-implemented power state is written to this register, the bridge completes the write transaction, ignores the write data, and does not change the value of this field. Writing a value of D0 when the previous state was D3 will cause a device reset to occur (without activating the secondary RST#). 00 D0 01 D1 (not implemented) 10 D2 (not implemented) 11 D3

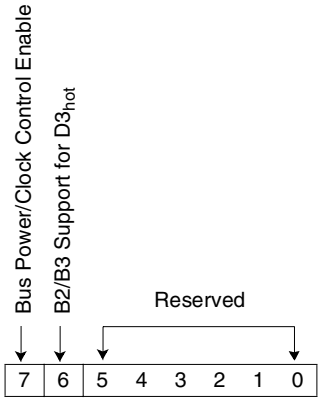
5.2.5.27 PCI-to-PCI Bridge Support Extensions Register

This register is read only and indicates that the bridge will not stop the clocks on a change of power state.

Address Offset x‘96’

Access Read only

Reset Value x‘00’



Bit(s)	Access	Field Name and Description
7	RO	Bus Power/Clock Control Enable Forced to b‘0’ to indicate that the secondary clock cannot be controlled by this device and that the power/clock control capabilities have been disabled.
6	RO	B2 B3# (B2/B3 support for $D3_{hot}$) Forced to b‘0’ and has no meaning because bit 7 indicates that the power/clock control capabilities have been disabled.
5:0	RO	Reserved.

5.2.5.28 Data Register

This register is read only, since the bridge does not implement the data register.

Address Offset x‘97’

Access Read Only

Reset Value x‘00’

5.2.5.29 Secondary Bus Private Device Mask Register

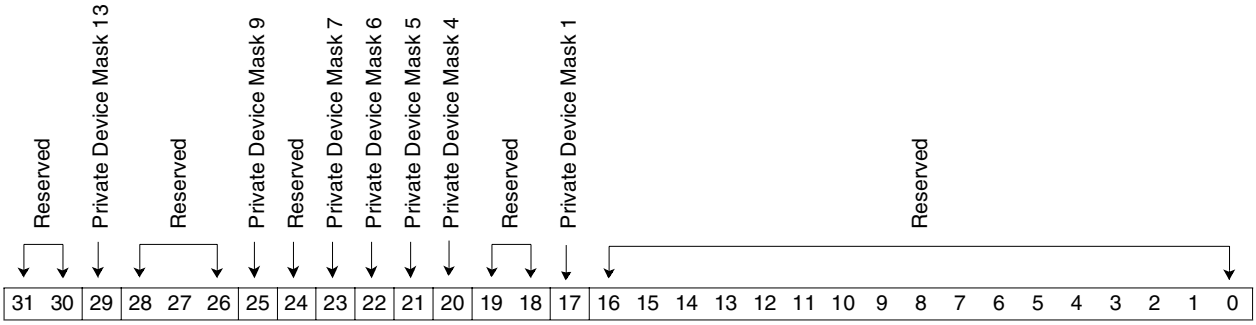
This register provides a means to implement private devices on the secondary PCI bus. The process of converting Type 1 configuration transactions to Type 0 configuration transactions is modified by the contents of this register. A configuration transaction that targets a device masked by this register is rerouted to device 15. Setting the secondary bus private device mask register to zeros disables this function.

Address Offset x'B0'

Access R/W

Reset Value x'0000 0000' When IDSEL_REROUTE_EN (pin AC22) is tied low. See *Table 7-3* on page 113 for details of strapping considerations.

Reset Value x'22F2 0000' When IDSEL_REROUTE_EN (pin AC22) is tied high. See *Table 7-3* on page 113 for details of strapping considerations.



Bit(s)	Access	Field Name and Description
31:30	RW	Reserved. Masking for devices 14 and 15 is not implemented. Operation of the IBM 133 PCI-X Bridge R2.0 is unaffected by the value of these bits.
29	RW	Private Device Mask 13 0 Rerouting disabled for device 13. 1 Block assertion of S_AD(pin 29) for configuration transactions to device 13, assert S_AD(pin 31) instead.
28:26	RW	Reserved. Masking for devices 12, 11, and 10 is not implemented. Operation of the IBM 133 PCI-X Bridge R2.0 is unaffected by the value of these bits.
25	RW	Private Device Mask 9 0 Rerouting disabled for device 9. 1 Block assertion of S_AD(pin 25) for configuration transactions to device 9, assert S_AD(pin 31) instead.
24	RW	Reserved Masking for devices 8 is not implemented. Operation of the IBM 133 PCI-X Bridge R2.0 is unaffected by the value of this bit.
23	RW	Private Device Mask 7 0 Rerouting disabled for device 7. 1 Block assertion of S_AD(pin 23) for configuration transactions to device 7, assert S_AD(pin 31) instead.

Bit(s)	Access	Field Name and Description
22	RW	Private Device Mask 6 0 Rerouting disabled for device 6. 1 Block assertion of S_AD(pin 22) for configuration transactions to device 6, assert S_AD(pin 31) instead.
21	RW	Private Device Mask 5 0 Rerouting disabled for device 5. 1 Block assertion of S_AD(pin 21) for configuration transactions to device 5, assert S_AD(pin 31) instead.
20	RW	Private Device Mask 4 0 Rerouting disabled for device 4. 1 Block assertion of S_AD(pin 20) for configuration transactions to device 4, assert S_AD(pin 31) instead.
19:18	RW	Reserved Masking for devices 3 and 2 is not implemented. Operation of the IBM 133 PCI-X Bridge R2.0 is unaffected by the value of these bits.
17	RW	Private Device Mask 1 0 Rerouting disabled for device 1. 1 Block assertion of S_AD(pin 17) for configuration transactions to device 1, assert S_AD(pin 31) instead.
16:0	RW	Reserved. Operation of the IBM 133 PCI-X Bridge R2.0 is unaffected by the value of these bits.

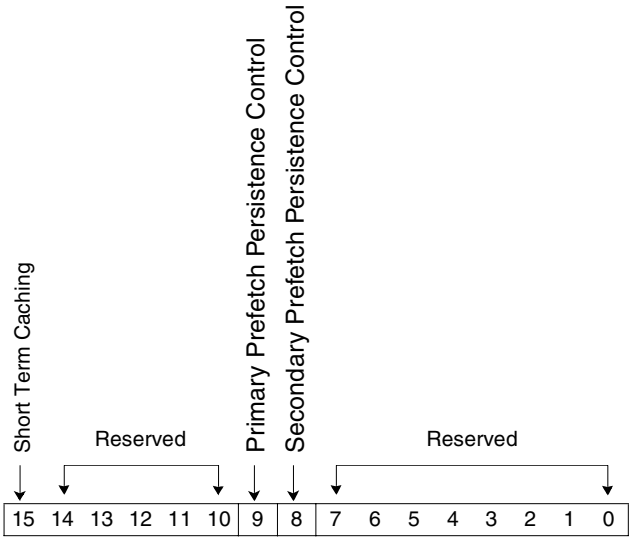
5.2.5.30 Miscellaneous Control Register 2

This register provides additional control over the memory read prefetch algorithm employed by the IBM 133 PCI-X Bridge R2.0 when both the primary and secondary buses are in PCI mode.

Address Offset x‘B8’

Access See individual fields.

Reset Value b‘0000 0000 0000 00x0’



Bit(s)	Access	Field Name and Description
15	R/W	Short Term Caching This is used to control the Short Term Caching feature of the IBM 133 PCI-X Bridge R2.0. 0 Disabled 1 Enabled. **Note:** A clear understanding all of the secondary side device’s device drivers and memory architectures, and ensuring that the *PCI to PCI Bridge Architecture Specification* as stated in Chapter 5, sections 5.1 Prefetching Read Data and 5.6.2 Stale Data has been strongly adhered to, is required to prevent stale data from being delivered to the master.
14:10	RW	Reserved Modifying the contents of this field will have serious and unpredictable affects on the operation of the IBM 133 PCI-X Bridge R2.0
9	RW	Primary Prefetch Persistence Control Affects the how the bridge reacts to target disconnect when prefetching data on the secondary bus for read transactions that are initiated on the primary bus. 0 Discontinue prefetching when the target disconnects regardless of how much data has been buffered 1 Continue prefetching despite target disconnects until either: the byte count specified by Primary Data Buffering Control Register has been prefetched, or the initiator disconnects.
8	RW	Secondary Prefetch Persistence Control Affects the how the bridge reacts to target disconnect when prefetching data on the primary bus for read transactions that are initiated on the secondary bus. 0 Discontinue prefetching when the target disconnects regardless of how much data has been buffered 1 Continue prefetching despite target disconnects until either: the byte count specified by Secondary Data Buffering Control Register has been prefetched, or the initiator disconnects.

Bit(s)	Access	Field Name and Description
7:0	RW	Reserved. Modifying the contents of this field will have serious and unpredictable affects on the operation of the IBM 133 PCI-X Bridge R2.0

6. Clocking and Reset

This section explains the clocking requirements and reset functions of the IBM 133 PCI-X Bridge R2.0.

6.1 Clocking Domains

The IBM 133 PCI-X Bridge R2.0 has two clocking domains, one for the primary interface and one for the secondary interface. Each interface has its own clock input pin. The primary interface is controlled by the P_CLK input. The secondary interface and the internal arbiter are controlled by the S_CLK input. The bridge does not supply the clocks on either interface. The two bus clocks may be run synchronously or asynchronously to one another. The two clock frequencies are independent of each other and each may have any value allowed by the PCI/PCI-X bus architectures. A spread spectrum clock input is supported for either or both interfaces within the architectural bounds.

The bridge contains a separate internal phase-locked loop (PLL) circuit for each clocking domain. The PLL for each interface is employed when its bus is running in the PCI-X mode, as determined by the bus initialization process described below. When either bus is running in the PCI mode, the respective PLL is bypassed to allow for any clock frequency from zero to 66 MHz.

6.2 Clock Jitter

Clock jitter is defined as the relationship of one clock edge to a subsequent clock edge, measured at the same point. If these two edges are separated by one clock cycle, it is called cycle-to-cycle or short term jitter. If they are separated by hundreds or thousands of cycles, it is called long term jitter. As specified in *Electrical Information* on page 139, the IBM 133 PCI-X Bridge R2.0 tolerates a maximum of ± 250 ps of short term and long term jitter on each of its clock inputs. Clock jitter introduced by the internal PLLs of the bridge is accounted for within this maximum specification.

Careful design of the clock generation circuitry is an important factor in determining the speed of the bus. As indicated in the PCI and PCI-X architectures, all sources of clock jitter must be considered when determining the bus clock frequency. The minimum and maximum clock period specifications must not be violated for any single clock cycle. The system clock output period, including all sources of clock period variation such as jitter and component tolerances, must always be within the minimum and maximum limits defined for the mode in which the bus is configured. For example, if a specific system clock design has a maximum clock period variation of 180 ps, then the nominal clock period for the PCI-X 133 range needs to be at least 7.68ns (7.5ns + 0.180ns), making the maximum frequency allowed for this case is just over 130 MHz.

6.3 Mode and Clock Frequency Determination

As explained in Sections 6.2 and 8.9 of the *PCI-X Addendum to the PCI Local Bus Specification*, the mode and frequency range of each bus is determined by the values on its M66EN and PCIXCAP signals when the bus reset signal is active. Each bus client is then informed of the determination via an initialization pattern that is broadcast at the de-assertion or rising edge of the reset signal. This process is accomplished on the secondary interface differently than on the primary interface, due to architectural requirements for PCI-X bridges. The details for each are given below.

6.3.1 Primary Interface

The primary interface is capable of operating in either the conventional PCI mode or in the PCI-X mode, at any of the defined frequency ranges. When the IBM 133 PCI-X Bridge R2.0 is used on an add-in card, the M66EN and PCIXCAP pins on the card edge connector should be left unconnected (except for a required decoupling capacitor to provide an AC return path) to indicate this maximum capability. As defined by Section 9.10 of the *PCI-X Addendum to the PCI Local Bus Specification*, an add-in card's PCIXCAP pin must be consistent with the 133 MHz Capable bit in the PCI-X Bridge Status register. When the bridge is used on a motherboard, the system designer should wire the M66EN and PCIXCAP signal networks to all clients on the bus in the architected fashion to achieve the desired results.

The bridge does not have I/O pins for the M66EN or PCIXCAP signals on its primary interface. The bridge adjusts its internal configuration (including its internal PLL, if appropriate) solely on the basis of the initialization pattern it detects on the signals P_DEVSEL#, P_STOP#, and P_TRDY# at the rising edge of P_RST#. If the internal PLL is being used (the bus is configured in the PCI-X mode), a maximum of 100 µs from the rising edge of P_RST# is required to lock the PLL to the frequency of the clock supplied on the P_CLK input.

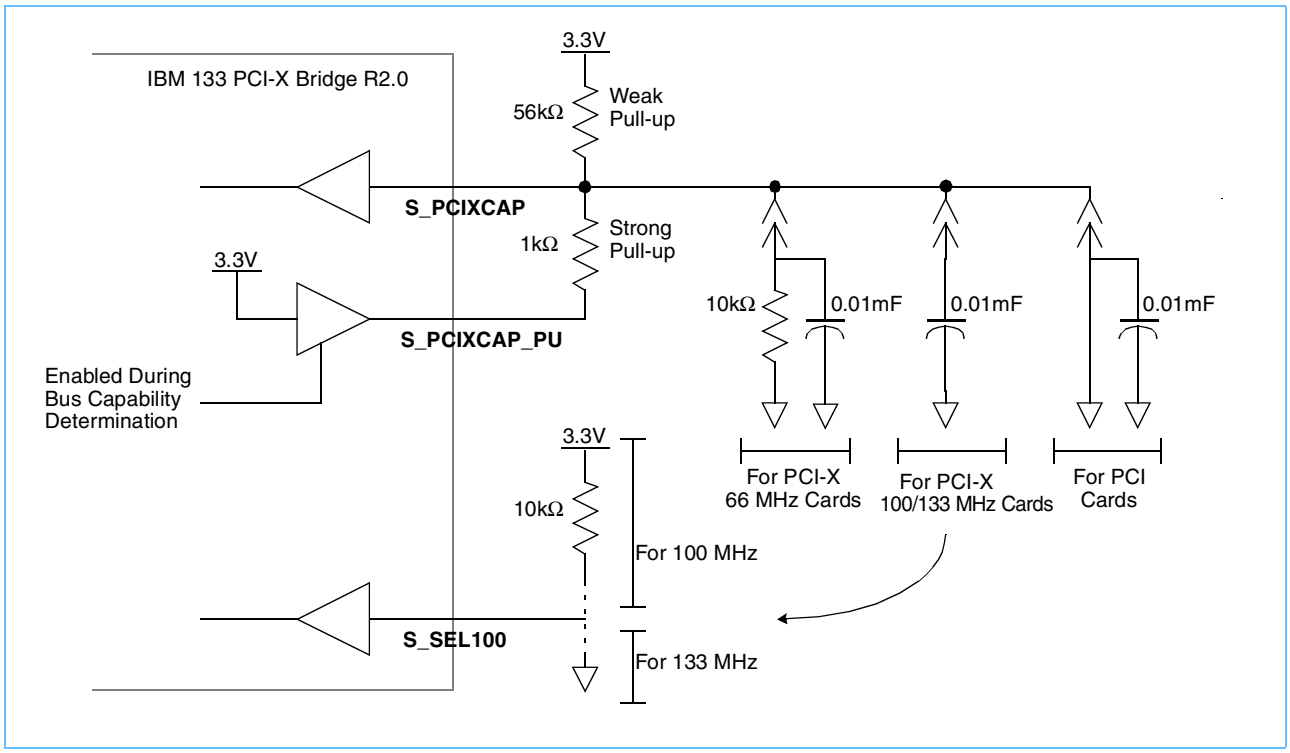
6.3.2 Secondary Interface

The secondary interface is also capable of operating in either the conventional PCI mode or in the PCI-X mode, at any of the defined frequency ranges. Since the bridge acts as the central resource for the secondary bus, it controls the mode and frequency determination process. As recommended by Section 14.2 of the *PCI-X Addendum to the PCI Local Bus Specification*, the bridge uses a programmable pull-up circuit to accomplish this. *Figure 6-1* shows the programmable pull-up circuit and the structure of the network that is connected to the S_PCIXCAP input of the bridge. Examples of the connections that are expected when PCI-X 66, PCI-X 133, and conventional PCI clients are attached to the bus are also shown (see the *PCI-X Addendum* for details). There are two pull-up resistors in the circuit. The first is an external weak pull-up whose value of 56kΩ is selected to set the voltage of the S_PCIXCAP input below its low threshold when a PCI-X 66 client is attached. The second is a strong pull-up, externally wired between the S_PCIXCAP and S_PCIXCAP_PU pins on the module. Its value of 1kΩ is selected to set the voltage of the S_PCIXCAP input above its high threshold when all clients on the bus are only PCI-X 66 capable.

During the mode and frequency determination process, the S_PCIXCAP_PU driver is initially disabled, effectively removing the strong pull-up resistor from the circuit. The bridge begins by sampling the value on its S_PCIXCAP input. If it detects a b'0' value, one or more clients have either pulled the network to ground (if they are PCI-X 66 capable) or tied it to ground (if they are only capable of conventional PCI operation). To distinguish between these two cases, the bridge then enables its S_PCIXCAP_PU driver to put the strong pull-up into the circuit. If, after a sufficient time, the S_PCIXCAP input remains at a b'0' value, the network must be tied to ground by one or more clients, and the bus is initialized to the conventional PCI mode. If the network can be pulled up, one or more clients are capable of only PCI-X 66 operation (and there are no conventional PCI devices), so the bus is initialized to PCI-X 66 mode.

If the bridge initially samples a b'1' value on the S_PCIXCAP input, then all clients on the bus are capable of PCI-X 133 operation. The bridge then samples the S_SEL100 input to distinguish between the 66-100 MHz and the 100-133 MHz clock frequency ranges. If it detects a b'1' value on the S_SEL100 input, the bus is initialized with the PCI-X 100 initialization pattern. If the value is b'0', the PCI-X 133 initialization pattern is used. These two ranges allow adjustment of the clock frequency to account for bus loading conditions.

Figure 6-1. Programmable Pull-up Circuit



Since the internal PLL is bypassed in the PCI mode and the S_CLK input is used directly, the IBM 133 PCI-X Bridge R2.0 has no need to distinguish between the PCI 66 and PCI 33 modes. Therefore the bridge does not have an I/O pin for the M66EN signal on its secondary interface. This signal should be routed to all devices on the bus in case the other clients require it.

6.4 Clock Stability

The PCI/PCI-X architectures specify that the bus clock must be stable and running at its designated frequency for at least 100µs prior to the de-assertion of the bus reset signal. As the IBM 133 PCI-X Bridge R2.0 does not generate the secondary bus clock but does control the secondary bus reset signal, it must detect when the S_CLK input has become stable in order to meet this requirement. The input signal S_CLK_STABLE is provided for this purpose. During a bus reset, the bridge will wait for the assertion of the S_CLK_STABLE input before executing the mode and frequency determination sequence described in *Section 6.3.2* on page 100.

The bridge is expecting at most one transition on the S_CLK_STABLE input from the not stable to the stable state. The S_CLK_STABLE input may also be tied-up if the secondary clock input will always be stable prior to the de-assertion of the primary bus reset signal or the secondary bus reset bit of the bridge control register (see *Section 6.6.2* on page 103).There are several possibilities for the source of the S_CLK_STABLE input signal. For example: some clock generation circuits that use phase-locked loops provide a lock indicator that may be used for this purpose. Care must be taken to assure that the lock indicator does not toggle randomly while the PLL is locking to the desired frequency before reaching a steady state. Another possibility is to

tie-up the signal, this may be useful for fixed frequency applications with simple clock generators or oscillators. A third possibility may be to use a 'power good' indicator, if the proper stability assurances can be made. Other ways to provide the S_CLK_STABLE input signal may also be possible.

The S_CLK_STABLE input provides another measure of control for cases where the secondary bus mode and clock frequency could vary from reset to reset, as in motherboard applications with pluggable slots. In these applications the external clock generation circuitry will need to adapt to the changes along with the bridge. If the S_CLK_STABLE signal is initially held low during reset, the bridge will not control the S_PCIXCAP network and the clock generation circuitry is free to do its own mode and frequency determination sequence. The clock frequency may be adjusted based on the number of populated slots, determined by the PRSNT pins of the bus. Once the frequency of the S_CLK input is stable, the clock circuit can assert the S_CLK_STABLE signal to allow the bridge to complete the reset sequence. The clock generation circuitry must ensure that the clock frequency it provides falls within the range that the bridge will ultimately determine and broadcast on the initialization pattern. To do this, the clock generator may need to drive the proper values on the S_SEL100 and S_PCIXCAP inputs, in addition to controlling the S_CLK_STABLE signal. A mismatch between the broadcast initialization pattern and the actual operating mode and frequency of the bus is a violation of the architecture and will cause unpredictable results.

6.5 Driver Impedance Selection

On the IBM 133 PCI-X Bridge R2.0, the output drivers for the bussed PCI/PCI-X interface signals are capable of two different output impedances, a 40 ohm output impedance for point-to-point applications and a 20 ohm output impedance for multi-point configurations. The output impedance for the primary and secondary interfaces is separately controlled. The bridge selects a default impedance value at the de-assertion of the bus reset on the basis of the bus mode and frequency initialization pattern which was received on the primary interface or generated on the secondary interface. It is assumed that if a bus is configured to be in the PCI-X 133 mode, it will be lightly loaded and therefore have a higher impedance. The drivers are put into point-to-point mode for this case. For all other PCI-X and all PCI configurations, the bridge assumes that the bus is more heavily loaded and has a lower impedance, so the drivers are set to multi-point mode.

There may be some applications for which these assumptions are inaccurate. For example, a conventional PCI device may be connected in a point-to-point manner. For exceptions like this, two control input signals are provided, P_DRVR_MODE for the primary interface and S_DRVR_MODE for the secondary interface. When these inputs are pulled high, the bridge will change the output impedance of the drivers on their respective interfaces to the opposite state than was assumed by default, as shown in *Table 6-1*.

Table 6-1. Driver Impedance Selection

Primary Bus Mode	Default Driver Mode (P_DRVR_MODE=0)	Driver Mode if P_DRVR_MODE=1	Secondary Bus Mode	Default Driver Mode (S_DRVR_MODE=0)	Driver Mode if S_DRVR_MODE=1
Conventional PCI	Multi-point (20Ω)	Point-to-point (40Ω)	Conventional PCI	Multi-point (20Ω)	Point-to-point (40Ω)
PCI-X 66	Multi-point (20Ω)	Point-to-point (40Ω)	PCI-X 66	Multi-point (20Ω)	Point-to-point (40Ω)
PCI-X 100	Multi-point (20Ω)	Point-to-point (40Ω)	PCI-X 100	Multi-point (20Ω)	Point-to-point (40Ω)
PCI-X 133	Point-to-point (40Ω)	Multi-point (20Ω)	PCI-X 133	Point-to-point (40Ω)	Multi-point (20Ω)

Note that the values on these inputs are only valid at reset time; they may not be used to change the driver mode dynamically, though they may be set differently at each reset if desired to account for changes in bus loading and mode. Regardless of the driver impedance used, signal analysis should always be done with the actual or expected bus topology and wiring to verify proper operation. Nets may need to be tuned and series terminations or other adjustments may be required in order to meet the frequency targets.

6.6 Reset Functions and Operations

Each bus interface has an asynchronous bus reset signal that is used at power-on and other times to place the IBM 133 PCI-X Bridge R2.0 into a known state. On the primary interface, the reset signal is an input to the bridge. On the secondary interface, the reset signal is an output driven by the bridge in its role as the central resource for that bus.

6.6.1 Primary Reset

The bus reset for the primary interface is called P_RST#. It is an input to the bridge and is controlled by an external upstream central resource. When asserted (see *Figure 6-2* and *Table 6-2*) it forces all bus output signals from the bridge into their benign states and sets all configuration registers within the bridge to their reset values as defined in *Configuration Transactions* on page 28. Activating the P_RST# signal also causes the secondary bus reset signal to be asserted, as required by the PCI/PCI-X specifications.

On the de-assertion or rising edge of the P_RST# signal, the initialization pattern is received off of the bus and latched into the bridge. If the indication is that the primary bus is operating in the PCI-X mode, an internal PLL is used to source the clock tree for the primary clock domain. The appropriate range and tuning bits for the PLL are set according to the indicated frequency range, and an internal PLL reset signal is deactivated to allow the PLL to begin locking to the P_CLK input frequency. Since the PLL requires an allowance of 100 µs to accomplish this frequency lock, an internal reset is held on the logic in the primary clock domain until this time period has elapsed. While the internal logic reset is active, the bridge will not respond to any primary bus transactions. When the primary bus is operating in the PCI mode, the internal PLL for the primary interface is not used. In this case, the internal PLL reset remains activated, keeping the PLL in the bypass mode, and the internal logic reset is held for only seven additional primary clock cycles after the rising edge of P_RST#.

6.6.2 Secondary Reset

The bus reset for the secondary interface is called S_RST#, it is an output from the bridge. Whenever P_RST# is asserted or when the secondary bus reset bit (bit 6) of the bridge control register is set to b'1', S_RST# is asserted immediately, asychronously to the secondary bus clock. When the secondary bus reset bit is being used to control S_RST#, the software must be sure that the required minimum reset active time (T_{rst}) of 1 ms is met.

Several things must occur at or prior to the de-assertion of the secondary bus reset signal. Once P_RST# is de-asserted or the secondary bus reset bit is changed from b'1' to b'0', indicating that S_RST# should be deactivated, the bridge will wait for the S_CLK_STABLE signal to be asserted before proceeding. The S_CLK input must be stable at a frequency within the bus capability limits prior to the assertion of S_CLK_STABLE. Since the *PCI Local Bus Specification* requires that the bus clock be stable for at least 100 µs prior to the de-assertion of the bus reset, S_CLK_STABLE serves as a gate to a timer that ensures that this requirement is met. During this time delay period, the determination of the secondary bus mode and frequency capability is made through the use of the programmable pull-up circuit described in *Section 6.3.2*. This process may include up to 80 µs for the capacitive load on the S_PCIXCAP net to be charged, making it prudent to overlap

the two functions. By the time the 100 µs timer expires, the bus capability will have been determined and the appropriate initialization pattern can be driven on the secondary interface. The S_RST# signal is then de-asserted a minimum of ten secondary bus cycles later.

When the secondary bus is operating in the PCI-X mode, an internal PLL is used to source the clock tree for the secondary clock domain inside the bridge. The appropriate range and tuning bits for the PLL are set once the initialization pattern is known, and an internal PLL reset signal is deactivated to allow the PLL to begin locking to the S_CLK input frequency. The PLL requires an allowance of 100 µs to accomplish this frequency lock. An internal reset is held on the logic in the secondary clock domain until this time period has elapsed. While the internal reset is active, the bridge will not respond to any secondary bus transactions. When the secondary bus is operating in the PCI mode, the internal PLL for the secondary interface is not used. In this case, the internal PLL reset remains activated, keeping the PLL in the bypass mode, and the internal logic reset is held for only five additional secondary clock cycles.

Figure 6-2. De-assertion of S_RST#

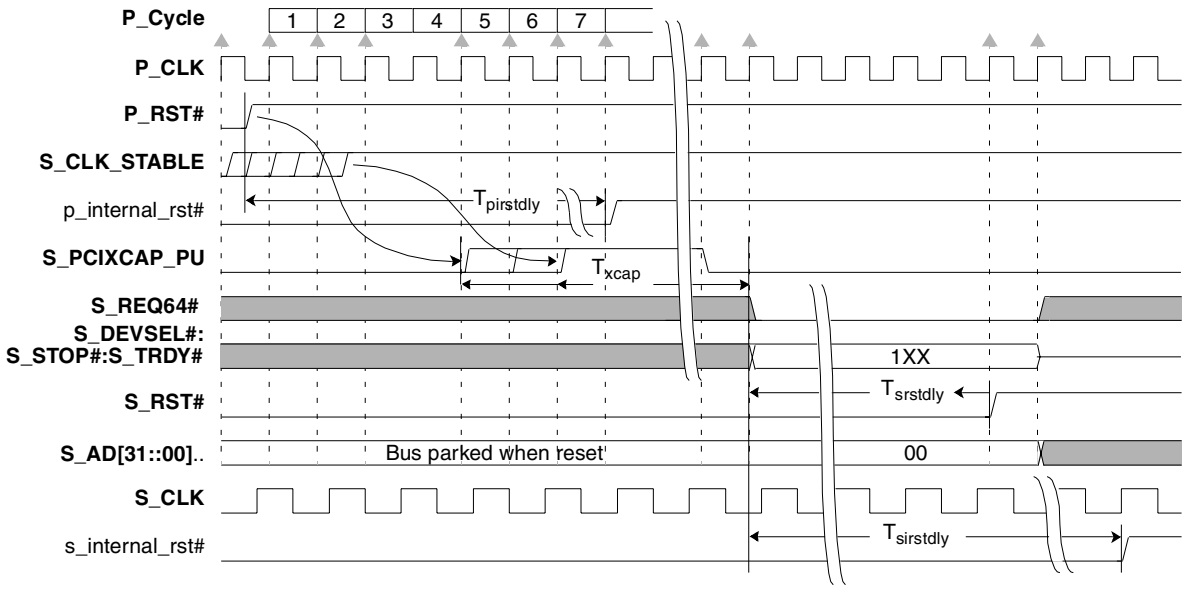


Table 6-2. Delay Times for De-assertion of S_RST#

	PCI	PCI-X (66 MHz)	PCI-X (100 MHz)	PCI-X (133 MHz)
$T_{pirstdly}$	7 P_cycles	6678 P_cycles 100 µs - 133 µs	13350 P_cycles 133 µs - 200 µs	13350 P_cycles 100 µs - 133 µs
T_{xcap}	6675 P_cycles	6675 P_cycles 100 µs - 133 µs	13347 P_cycles 133 µs - 200 µs	13347 P_cycles 100 µs - 133 µs
$T_{srstdly}$	11 S + 7 P_cycles	11 S + 7 P_cycles	11 S + 7 P_cycles	11 S + 7 P_cycles
$T_{sirstdly}$	16 S_cycles	6687 S_cycles 100 µs - 133 µs	13359 S_cycles 133 µs - 200 µs	13359 S_cycles 100 µs - 133 µs

In *Table 6-2*, the terms “P_cycles” and “S_cycles” refer to clock cycles whose period is determined by the P_CLK and S_CLK input frequencies, respectively. Since the time periods listed in the table are based on counters, different clock rates will result in different effective delays, as shown. The counter values have been selected to meet the various minimum delay requirements, but will result in longer times when the clock period lengthens.

6.7 Bus Parking and Bus Width Determination

On the secondary interface, as required by the *PCI-to-PCI Bridge Specification*, the S_AD(31:0), S_C/BE(3:0), and PAR signals will be driven to zeros whenever S_RST# is asserted. This is known as bus parking. The signals are driven low within a few cycles of the falling edge of S_RST#; they are released (placed in the high-Z state) in the cycle following the rising edge of S_RST#.

The bridge is also required to drive S_REQ64# low for at least ten cycles prior to the de-assertion of S_RST#, to allow devices to determine whether they are connected on a 64-bit data path or a 32-bit data path. For convenience, this is done coincident with the broadcasting of the initialization pattern, as shown in *Figure 6-2* and *Table 6-2*.

6.8 Power Management and Hot-Plug

The IBM 133 PCI-X Bridge R2.0 is compliant with the minimum requirements of the *PCI Power Management Interface Specification,* as it supports the D0 and D3 power management states and the power management capabilities registers. No other power management functions are implemented by the bridge. Power management events (PMEs) are not supported.

The transition into a D3 power management state by the bridge will be the result of either a software action or the removal of power. The D3 state has two variants that are supported, $D3_{hot}$ and $D3_{cold}$. When the bridge transitions from the D0 state to the $D3_{hot}$ state, the secondary bus signals are driven to their benign state and the bridge only accepts Type 0 configuration transactions on the primary interface. On the transition from the $D3_{hot}$ to the D0 state, all configuration registers are returned to their reset values without the generation of a secondary side PCI reset (S_RST#). The generation of a secondary side PCI reset after transitioning to the DO state is supported by software writing to the bridge control register x'3E' bit 6.

The transition to the $D3_{cold}$ state occurs when power is removed from the device. The bridge will be in the uninitialized D0 state once power is reapplied and the power-on sequences associated with P_RST# and S_RST# described in *Section 6.6.1* and *Section 6.6.2* are complete. These power-on sequences require software to fully initialize and configure the bridge.

The bridge contains no functions to specifically assist or preclude its use in a hot-plug system. In such an environment, each hot-plug slot must be independently controlled via an external hot-plug controller. Such a controller is required to perform the various initialization and reset functions described above for the slots under its control. In addition, before connecting those slots to the rest of the bus, it must assure that the capabilities of devices plugged into the slots match the mode and operating frequency of the bus. Presumably, the hot-plug controller will need to remember the initialization pattern broadcast at the last bus reset. If it detects a device with the same or greater capability than what the bus is running, it should initialize the card with the stored pattern before connecting to the bus. If it detects a lower capability, then a bus reset is required and the entire bus must be reconfigured.

6.9 Secondary Device Masking

The IBM 133 PCI-X Bridge R2.0 supports the masking of secondary devices through configuration/power strapping of the secondary bus private device mask register. The process of converting Type 1 configuration transactions to Type 0 configuration transactions is modified by the contents of the secondary bus private device mask register. A configuration transaction that targets a device masked by this register is routed to device 15. Secondary bus architectures which are designed to support masking of devices should not implement a device number 15 (i.e., S_AD(31)).

The device mask bit options (device numbers 1, 4, 5, 6, 7, 9, and 13) defined by the bridge allow architectures to support private device groupings that use a single or multiple interrupt binding per *Table 9-1 of the PCI-to-PCI Bridge specification*.

6.10 Handling of Address Phase Parity Errors

When an address parity error is detected by the bridge, the transaction will not be claimed (by not asserting DEVSEL#) and is allowed to terminate with a master abort. The bridge will detect address parity errors for all transactions on both the primary and secondary interfaces. The result of an address parity error will be controlled by the parity error response bits in both the command register and the bridge control register.

6.11 Optional Base Address Register

The 64 bit Base Address register located in the bridge configuration space at offsets x'10' and x'14' can optionally be used to acquire a 1 MB memory region at system initialization. The PCI specification calls for the region that is defined by this register to be used by the bridge itself. The IBM 133 PCI-X Bridge uses this register to claim an additional prefetchable memory region for the secondary bus. When used in conjunction with the secondary device masking, this allows for the acquisition of memory space for private devices that are not otherwise viewable by the system software.

This 64 bit base address register and the memory space defined by it are enabled by the strapping pin, BAR_EN. When BAR_EN is pulled low, this register location returns zeros for reads and cannot be written. When BAR_EN is pulled high, the upper memory base address register and lower memory base address registers combined specify address bits 63:20 of a memory region. Memory accesses on the primary bus are compared against this register, if address bits 63:20 are equal to bits 63:20 of the address defined by the combination of the lower memory base address register and the upper memory base address register, the access is claimed by the bridge and passed through to the secondary bus. Memory accesses on the secondary bus are also compared against this register, if address bits 63:20 are equal to bits 63:20 of the address defined by the combination of the lower memory base address register and the upper memory base address register, the access is ignored by the bridge.

6.12 Optional Configuration Register Access from the Secondary Bus

On the secondary bus, the bridge responds to a Type 0 configuration transaction by accepting the transaction when the following conditions are met during the address phase:

- The command on S_C/BE(3:0)# indicates a configuration read or configuration write transaction,
- S_AD(1:0) are b'00', and
- S_IDSEL is asserted.

Applications that require access to the bridge configuration registers via the secondary bus may optionally control the initialization sequence through the P_CFG_BUSY pin and bit 2 of the miscellaneous control register. When P_CFG_BUSY is pulled high, bit 2 of the miscellaneous control register is set to b'1' at power up and reset. This causes the bridge to retry Type 0 configuration transactions on the primary bus that would otherwise be accepted. The bridge continues to retry these transactions until such time as bit 2 of the miscellaneous control register is set to b'0' by a configuration write initiated on the secondary bus. This mechanism allows an agent on the secondary bus to initialize the bridge and any private devices on the secondary bus without contention from agents accessing the bridge via the primary bus.

Applications that do not require access to the bridge configuration registers via the secondary bus should pull both the S_IDSEL and P_CFG_BUSY pins low.

6.13 Short Term Caching

Short Term Caching was developed to provide performance improvements where upstream devices are not able to stream data continuously to meet the prefetching needs of the IBM 133 PCI-X Bridge. As defined in the *PCI to PCI Bridge Architecture Specification*, when the master completes the transaction, the bridge is required to discard the balance of any data that was prefetched for the master. To prevent performance impacts when dealing with target devices that can only stream data of from 128 to 512 bytes before disconnecting, the IBM 133 PCI-X Bridge has a feature called “Short Term Caching”. This feature applies only when the secondary bus is operating in PCI mode and provides a time limited read data cache in which the bridge will not discard prefetched read data after the request has been completed on the initiating bus.

Short Term Caching is an optional feature which is enabled by setting the “Miscellaneous Control Register 2” bits 8 and 15. When enabled, the IBM 133 PCI-X Bridge will not discard the additional prefetched data when the read transaction has been completed on the initiating bus. As such, the IBM 133 PCI-X bridge will continue to prefetch data up to the amount specified by the “Secondary Data Buffering Control Register” offset x’42’ bits 14:12. Should the initiator generate a new transaction requesting the previously prefetched data, the IBM 133 PCI-X Bridge will return that data. However, the IBM 133 PCI-X Bridge will discard the data approximately 64 secondary bus side clocks after some of the data for a request has been returned to the initiator, and the initiator has not requested additional data.

If this feature is enabled, it will apply to all devices attached to the secondary side of the IBM 133 PCI-X Bridge. System designers need to ensure that all attached devices have memory region(s) that are architected to be accessed by only one master and that the additional prefetching will present data to the initiator in the same state as if the initial transaction were continued. This feature should only be used in system designs that are able to ensure that the data provided to the master has not been modified since the initial transaction.

Note: A clear understanding of all the secondary side device’s device drivers and memory architectures, and ensuring that the PCI to PCI Bridge Architecture Specification as stated in Chapter 5, sections 5.1 *Prefetching Read Data* and 5.6.2 *Stale Data* has been strongly adhered to, is required to prevent stale data from being delivered to the master.

7. Signal Descriptions

This section describes in detail each input and output signal of the IBM 133 PCI-X Bridge R2.0. Throughout this section, trailing pound signs, for example P_ACK64#, designate signals that are active low.

7.1 Primary Interface Signals

Table 7-1. Primary Interface Signal List (Page 1 of 2)

Signal Name	I/O	Width	Description
P_ACK64#	I/O	1	Acknowledge 64-Bit Transfer Asserted by the currently addressed target on the primary bus to indicate its willingness to transfer data using 64 bits.
P_AD(63:00)	I/O	64	Multiplexed Address and Data 64-bit multiplexed address and data bus, shared by other devices on the primary bus. During a transaction, this bus contains the physical bus address, attributes, or data, or it may be reserved.
P_C/BE(7:0)#	I/O	8	Multiplexed Bus Command and Byte Enables During a transaction, these eight bits define the bus command, attributes, or byte enables for the transfer. These signals are shared with other agents on the primary bus and at times may be reserved.
P_CLK	I	1	Clock Received by the bridge and provides timing for all operations on the primary interface.
P_DEVSEL#	I/O	1	Device Select Asserted by the target on the primary bus that decoded the address of the current transaction as being within one of its address ranges. P_DEVSEL# is monitored by the bridge when performing a primary bus transaction on behalf of a secondary bus master. P_DEVSEL# is driven by the bridge when a primary bus master is performing a transaction on the primary bus intended for a secondary bus slave or the bridge's configuration registers.
P_FRAME#	I/O	1	Cycle Frame Defines the beginning and duration of each primary bus transaction and is controlled by the initiator of the operation. P_FRAME# is driven by the bridge when performing a primary bus transaction on behalf of a secondary bus master. P_FRAME# is monitored by the bridge when a primary bus master is performing a transaction on the primary bus.
P_GNT#	I	1	Grant Indicates that the bridge has been granted access to the primary bus.
P_IDSEL	I	1	Initialization Device Select Used as a chip select during configuration read and write transactions on the primary bus.
P_IRDY#	I/O	1	Initiator Ready Indicates the ability of the initiator on the primary bus to complete the current data phase of the transaction. It is used in conjunction with P_TRDY#. P_IRDY# is driven by the bridge when performing a primary bus transaction on behalf of a secondary bus master. P_IRDY# is monitored by the bridge when a primary bus master is performing a transaction on the primary bus to or through the bridge.

Table 7-1. Primary Interface Signal List (Page 2 of 2)

Signal Name	I/O	Width	Description
P_LOCK#	I	1	Lock Indicates that an atomic (unbroken) operation is required that may need multiple primary bus transactions to complete. P_LOCK# is monitored by the bridge when serving as the selected primary bus target in order to detect exclusive access.
P_PAR	I/O	1	Parity Parity protection bit for the lower half of the address/data and command/byte enable buses on the primary interface. It provides even parity across P_AD(31:00) and P_C/BE(3:0)#.
P_PAR64	I/O	1	Parity Upper DWord Parity protection bit for the upper half of the address/data and command/byte enable buses on the primary interface. It provides even parity across P_AD(63:32) and P_C/BE(7:4)#.
P_PERR#	I/O	1	Parity Error Used to report data parity errors on the primary interface. P_PERR# is monitored by the bridge when performing a primary bus write transaction on behalf of a secondary bus master or when serving as the selected slave for a primary bus read transaction. P_PERR# is driven by the bridge when performing a primary bus read transaction on behalf of a secondary bus master or when serving as the selected slave during a primary bus write transaction.
P_REQ#	O	1	Request Driven by the bridge to request use of the primary bus.
P_REQ64#	I/O	1	Request 64-Bit Transfer When asserted by the current master on the primary bus, this signal indicates a desire to transfer data using 64 bits.
P_RST#	I	1	Primary Bus Reset This signal is used to: Initialize the bridge to a known state, Drive the Secondary Bus Reset signal (S_RST#), and Drive all primary bus and secondary bus output signals to their benign state.
P_SERR#	O	1	System Error This signal is used to report address parity errors and other system errors where the results will be catastrophic. P_SERR# is driven by the bridge when detecting such errors on the primary bus or in the case of an error in which the initiator of the transaction cannot be otherwise notified. It is also driven as the result of S_SERR# being asserted on the secondary bus.
P_STOP#	I/O	1	Stop Indicates that the current target is requesting that the initiator stop the current transaction on the primary bus. P_STOP# is monitored by the bridge when performing a primary bus transaction on behalf of a secondary bus master. P_STOP# is driven by the bridge when addressed as a target and is asserted low to indicate target termination.
P_TRDY#	I/O	1	Target Ready Indicates the ability of the target on the primary bus to complete the current data phase of the transaction. It is used in conjunction with the P_IRDY# signal. P_TRDY# is monitored by the bridge when performing a primary bus transaction on behalf of a secondary bus master. P_TRDY# is driven by the bridge when a primary bus master is performing a transaction in which the bridge is the selected target.
Total		89	

7.2 Secondary Interface Signals

Table 7-2. Secondary Interface Signal List (Page 1 of 2)

Signal Name	I/O	Width	Description
S_ACK64#	I/O	1	Acknowledge 64-Bit Transfer Asserted by the currently addressed target on the secondary bus to indicate its willingness to transfer data using 64 bits.
S_AD(63:00)	I/O	64	Multiplexed Address and Data These signals are the 64-bit multiplexed address and data bus, shared by other devices on the secondary bus. During a transaction, this bus contains the physical bus address, attributes, or data, or it may be reserved.
S_C/BE(7:0)#	I/O	8	Multiplexed Bus Command and Byte Enables During a transaction, these eight bits define the bus command, attributes, or byte enables for the transfer. These signals are shared with other agents on the secondary bus and at times may be reserved.
S_CLK	I	1	Clock Received by the bridge and provides timing for all operations on the secondary interface.
S_DEVSEL#	I/O	1	Device Select Asserted by the target on the secondary bus that decoded the address of the current transaction as being within one of its address ranges. S_DEVSEL# is monitored by the bridge when performing a secondary bus transaction on behalf of a primary bus master. S_DEVSEL# is driven by the bridge when a secondary bus master is performing a transaction on the secondary bus intended for a primary bus slave.
S_FRAME#	I/O	1	Cycle Frame Defines the beginning and duration of each secondary bus transaction and is controlled by the initiator of the operation. S_FRAME# is driven by the bridge when performing a secondary bus transaction on behalf of a primary bus master. S_FRAME# is monitored by the bridge when a secondary bus master is performing a transaction on the secondary bus.
S_GNT1REQ#	O	1	Grant 1 This is a dual-purpose signal: When the bridge's internal arbiter is enabled, this signal is used as a grant output, activated by a the bridge to grant the use of the secondary bus to the master who requested the use with the S_REQ1GNT# signal. When the internal arbiter is disabled, this signal is used by the bridge as its request output signal.
S_GNT2# - S_GNT6#	O	5	Grants 2-6 Driven by the bridge's internal arbiter to grant usage of the secondary bus to the master that activated the corresponding request signal.
S_IRDY#	I/O	1	Initiator Ready This signal indicates the ability of the initiator on the secondary bus to complete the current data phase of the transaction. It is used in conjunction with S_TRDY#. S_IRDY# is driven by the bridge when performing a secondary bus transaction on behalf of a primary bus master. S_IRDY# is monitored by the bridge when a secondary bus master is performing a transaction on the secondary bus through the bridge.
S_LOCK#	I/O	1	Lock Indicates that an atomic (unbroken) operation is required that may need multiple secondary bus transactions to complete. The bridge drives S_LOCK# only to propagate an exclusive access from the primary bus to the secondary bus and monitors S_LOCK# as part of that protocol. When acting as a target on the secondary interface, the bridge ignores S_LOCK#.

Table 7-2. Secondary Interface Signal List (Page 2 of 2)

Signal Name	I/O	Width	Description
S_PAR	I/O	1	Parity Parity protection bit for the lower half of the address/data and command/byte enable buses on the secondary interface. It provides even parity across S_AD(31:00) and S_C/BE(3:0)#.
S_PAR64	I/O	1	Parity Upper DWord Parity protection bit for the upper half of the address/data and command/byte enable buses on the secondary interface. It provides even parity across S_AD(63:32) and S_C/BE(7:4)#.
S_PERR#	I/O	1	Parity Error Used to report data parity errors on the secondary interface. S_PERR# is monitored by the bridge when performing a secondary bus write transaction on behalf of a primary bus master or when serving as the selected slave for a secondary bus read transaction. S_PERR# is driven by the bridge when performing a secondary bus read transaction on behalf of a primary bus master or when serving as the selected slave during a secondary bus write transaction.
S_REQ1GNT#	I	1	Request 1 This is a dual-purpose signal: When the bridge's internal arbiter is enabled, this signal is used as a request input, to be activated by a secondary bus master requesting the use of the secondary bus. When the internal arbiter is disabled, this signal is used by the bridge as its grant input signal.
S_REQ2# - S_REQ6#	I	5	Requests 2-6 Activated by the secondary bus masters to request the use of the secondary bus.
S_REQ64#	I/O	1	Request 64-Bit Transfer This signal, when asserted by the current master on the secondary bus, indicates a desire to transfer data using 64 bits.
S_RST#	O	1	Secondary Bus Reset S_RST#, driven by the bridge, is the secondary bus reset signal. Asserted when P_RST# is active or when the secondary bus reset bit in the bridge control register is set.
S_SERR#	I	1	System Error Used to report address parity errors and other system errors where the results will be catastrophic. S_SERR# is monitored by the bridge to detect these errors on the secondary bus. When S_SERR# is asserted, it results in P_SERR# being asserted on the primary bus by the bridge.
S_STOP#	I/O	1	Stop Indicates that the current target is requesting that the initiator stop the current transaction on the secondary bus. S_STOP# is monitored by the bridge when performing a secondary bus transaction on behalf of a primary pus master. S_STOP# is driven by the bridge when addressed as a target and is asserted low to indicate target termination.
S_TRDY#	I/O	1	Target Ready Indicates the ability of the target on the secondary bus to complete the current data phase of the transaction. It is used in conjunction with the S_IRDY# signal. S_TRDY# is monitored by the bridge when performing a secondary bus transaction on behalf of a primary bus master. S_TRDY# is driven by the bridge when a secondary bus master is performing a transaction in which the bridge is the selected target.
Total		98	

7.3 Strapping Pins and Other Signals

Table 7-3. List of Strapping Pins and Other Signals (Page 1 of 2)

Signal Name	I/O	Width	Description
64_BIT_DEVICE#	I	1	Physical bus width of the PCI-X device Used only when the IBM 133 PCI-X Bridge R2.0 is employed as the bus interface on a PCI-X add-in card. The PCI-X Specification requires that such devices indicate the physical width of their bus in bit 16 of the PCI-X bridge status register. Bit 16 of the PCI-X bridge status register is set directly from the inverse of the 64_BIT_DEVICE# pin. This information is used solely by the configuration software; operation of the IBM 133 PCI-X Bridge R2.0 is unaffected. 0 bit 16 of the PCI-X bridge status register is set to b'1', indicating a 64 bit bus. 1 bit 16 of the PCI-X bridge status register is set to b'0', indicating a 32 bit bus.
BAR_EN	I	1	Base Address Register Enable Used to enable the base address register at reset or power up. The 64 bit register located at offsets x'10' and x'14' is used to claim a 1 MB memory region when enabled. The register returns all zeroes to read accesses and the associated memory region is not claimed when disabled. 0 BAR disabled, register reads returns 0's, no memory region claimed. 1 BAR enabled, bits 63:20 can be written by software to claim a 1 MB memory region.
IDSEL_REROUTE_EN	I	1	IDSEL Reroute Enable Used to enable the IDSEL reroute function at reset or power up. The reset value of the secondary bus private device mask register is modified according to the tie value of the IDSEL_REROUTE_EN pin. Note that configuration software can subsequently modify the secondary bus private device mask register, regardless of how the IDSEL_REROUTE_EN pin is tied. 0 reset value of the secondary bus private device mask register is x'00000000'. 1 reset value of the secondary bus private device mask register is x'22F20000'.
OPAQUE_EN	I	1	Opaque Region Enable Used to enable the opaque memory region at reset or power up. The reset value of bit 0 of the opaque memory enable register is modified according to the tie value of the OPAQUE_EN pin. The configuration software can subsequently modify bit 0 of the opaque memory enable register, regardless of how the OPAQUE_EN pin is tied. 0 reset value of bit 0 of the opaque memory enable register is b'0'. 1 reset value of bit 0 of the opaque memory enable register is b'1'.
P_CFG_BUSY	I	1	Primary Configuration Busy Controls the reset and power up value of bit 2 of the miscellaneous control register. Used to sequence initialization with regard to the primary and secondary buses for applications that require access to the bridge configuration registers from the secondary bus. When pulled high, the configuration commands received on the primary bus are retried until such time as bit 2 of the miscellaneous control register is set to b'0' by a configuration write initiated from the secondary bus. Applications that do not require access to the bridge configuration registers from the secondary bus should pull this signal to ground. 0 reset value of bit 2 of the miscellaneous control register is b'0'. 1 reset value of bit 2 of the miscellaneous control register is b'1'.
P_DRVR_MODE	I	1	Primary driver mode control Used to alter the output impedance of the primary bus PCI/PCI-X drivers, to account for how many drops are on the bus. This line should be pulled through a resistor to a high or a low as needed. Internal pull down. 0 use the default impedance value. 1 select the alternate impedance value.
RESERVED2		1	Reserved pin RESERVED2 should be pulled to ground.

Note: Each strapping pin or reserved pin should have an unshared series resistor tying it to either ground or 3.3 V. The value of the resistor may be selected to limit part number count, but the value should be greater than or equal to 100Ω and less than or equal to 5kΩ.

Table 7-3. List of Strapping Pins and Other Signals (Page 2 of 2)

Signal Name	I/O	Width	Description
S_DRVR_MODE	I	1	Secondary driver mode control Used to alter the output impedance of the secondary bus PCI/PCI-X drivers, to account for how many drops are on the bus. This line should be pulled through a resistor to a high or a low as needed. Internal pull down. 0 use default impedance value. 1 select alternate impedance value.
S_CLK_STABLE	I	1	S_CLK Input Stable Indicates when the S_CLK input to the bridge is stable. It is used to determine when the S_RST# signal may be de-asserted. 0 S_CLK input is not yet stable. 1 S_CLK input is stable.
S_IDSEL	I	1	Initialization Device Select Used as a chip select during configuration read and write transactions on the secondary bus. Applications that do not require access to the bridge configuration registers from the secondary bus should pull this pin low.
S_INT_ARB_EN#	I	1	Internal Arbiter Enable Used to choose between the internal arbiter and external arbiter for the secondary bus. 0 Use the internal arbiter. 1 Disable the internal arbiter, use an external arbiter.
S_PCIXCAP	I	1	Secondary Bus PCI-X Capable Used in conjunction with the S_SEL100 signal to determine the operating frequency and mode of the secondary interface.
S_PCIXCAP_PU	O	1	S_PCIXCAP Pull-up Driver Part of a programmable pull-up circuit used to detect the three possible states of the S_PCIXCAP input signal. A 1kΩ resistor must be placed on the board and wired between this signal and S_PCIXCAP.
S_SEL100	I	1	Secondary Bus 100MHz Indicator Used to choose between 100MHz and 133MHz maximum operating frequency on the secondary interface when in the PCI-X mode. It has no meaning in the PCI mode, but should be tied to a stable value. 0 133MHz. 1 100MHz.
Total		14	

Note: Each strapping pin or reserved pin should have an unshared series resistor tying it to either ground or 3.3 V. The value of the resistor may be selected to limit part number count, but the value should be greater than or equal to 100Ω and less than or equal to 5kΩ.

7.4 Test Signals

Table 7-4. List of Test Signals

Signal Name	I/O	Width	Description
JTG_TCK	I	1	JTAG Test Clock Used to clock state information and test data into and out of the bridge during operation of the IEEE 1149.1 test access port (TAP). Internal pull-up.
JTG_TDI	I	1	JTAG Test Data Input Used to serially shift test data and test instructions into the bridge during TAP operation. Internal pull-up.
JTG_TDO	O	1	JTAG Test Output Used to serially shift test data and test instructions out of the bridge during TAP operation.
JTG_TMS	I	1	JTAG Test Mode Select Used to control the state of the TAP controller within the bridge. Internal pull-up.
JTG_TRST#	I	1	JTAG Test Reset Provides an asynchronous initialization of the TAP controller within the bridge. Internal pull-up.
T_DI1#	I	1	Driver Inhibit 1 Used to tri-state the outputs of non-test drivers during manufacturing test. It must be tied high for normal system operation. Internal pull-up.
T_DI2#	I	1	Driver Inhibit 2 Used to tri-state the outputs of the test drivers during manufacturing test. It must be tied high for normal system operation. Internal pull-up.
T_MODECTL	I	1	Driver Mode Control Used to control the PCI/PCI-X driver impedance during manufacturing test. It should be tied low for normal system operation.
T_RI#	I	1	Receiver Inhibit Used to gate all receivers during manufacturing test. It must be tied high for normal system operation.
TEST_CE0	I	1	Test Mode Enable Used to enable scan testing of the bridge device during manufacturing test. It must be tied low for normal system operation. Internal pull-down.
XCLK_OUT	O	1	PLL Output Monitor This signal may be used to monitor the output of the phase-locked loop circuits for the primary and secondary interfaces. During normal system operation, this output is in a high-impedance state and should be pulled down on the board.
Total		11	

Note: With the bridge containing internal pull-up resistors on TDI, TMS, TRST#, and TCK, system designers need to assure voltage dividers are not generated by the possible implementation of pull-down resistors as defined in sections 4.3.3 and 4.4.1 of the PCI 2.2 specification.

7.5 Power and Ground Connections

Table 7-5. List of Power and Ground Connections

Input Name	Number	Description	See Note
P_VDDA	1	Quiet 2.5 V power supply connection to the PLL for the primary clock domain.	1
S_VDDA	1	Quiet 2.5 V power supply connection to the PLL for the secondary clock domain.	1
VDD	16	2.5 V power supply connections for the internal logic.	
VDD2	26	3.3 V power supply connections for the I/O circuits.	
GND	48	Ground connections.	
Total	92		

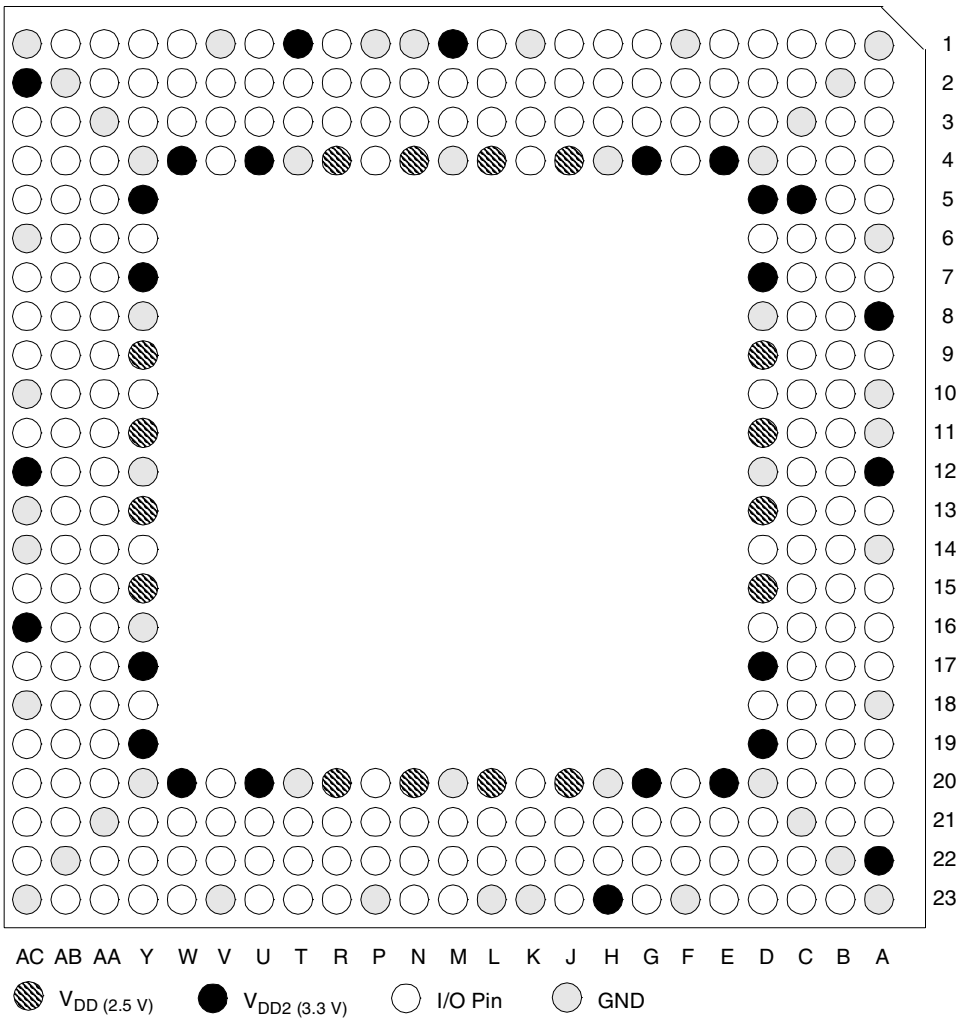
1. A filtering circuit may be required to ensure a quiet supply at this pin (only if the associated bus is operated in PCI-X mode). A suggested filtering circuit can be found in the ‘Phase-Locked Loop’ chapter of the IBM ‘ASIC SA-12E Databook’.

8. Package Pin Assignments

8.1 Physical Information

Note: Trailing pound signs, as in P_ACK64#, designate signals that are active low.

Figure 8-1. Pinout Diagram Viewed from above, looking through the module (balls are on the underside). For complete lists of pin numbers and names, see *Signal Pin Listing by Signal Name* on page 118 and *Signal Pin Listing by Grid Position* on page 121.



8.2 Complete Signal Pin List, Sorted by Signal Name

Table 8-1. Signal Pin Listing by Signal Name

Signal Name	Grid Position	Signal Name	Grid Position	Signal Name	Grid Position
64_BIT_DEVICE#	Y22	GND	Y12	P_AD(49)	B5
BAR_EN	G2	GND	Y16	P_AD(48)	C2
GND	A1	GND	Y20	P_AD(47)	D2
GND	A6	GND	AA3	P_AD(46)	F4
GND	A10	GND	AA21	P_AD(45)	E3
GND	A11	GND	AB2	P_AD(44)	F3
GND	A14	GND	AB22	P_AD(43)	B1
GND	A18	GND	AC1	P_AD(42)	F2
GND	A23	GND	AC6	P_AD(41)	G3
GND	B2	GND	AC10	P_AD(40)	H3
GND	B22	GND	AC13	P_AD(39)	H2
GND	C3	GND	AC14	P_AD(38)	E1
GND	C21	GND	AC18	P_AD(37)	J3
GND	D4	GND	AC23	P_AD(36)	G1
GND	D8	IDSEL_REROUTE_EN	AC22	P_AD(35)	H1
GND	D12	JTG_TCK	F21	P_AD(34)	J2
GND	D16	JTG_TDI	C22	P_AD(33)	J1
GND	D20	JTG_TDO	B23	P_AD(32)	L1
GND	F1	JTG_TMS	D22	P_AD(31)	J23
GND	F23	JTG_TRST#	C23	P_AD(30)	M21
GND	H4	OPAQUE_EN	AA18	P_AD(29)	M22
GND	H20	P_ACK64#	A2	P_AD(28)	L21
GND	K1	P_AD(63)	B11	P_AD(27)	L22
GND	K23	P_AD(62)	D10	P_AD(26)	G23
GND	L23	P_AD(61)	C10	P_AD(25)	K20
GND	M4	P_AD(60)	A4	P_AD(24)	E23
GND	M20	P_AD(59)	B10	P_AD(23)	K21
GND	N1	P_AD(58)	C9	P_AD(22)	D23
GND	P1	P_AD(57)	B9	P_AD(21)	K22
GND	P23	P_AD(56)	A3	P_AD(20)	J21
GND	T4	P_AD(55)	B8	P_AD(19)	J22
GND	T20	P_AD(54)	B3	P_AD(18)	H21
GND	V1	P_AD(53)	C7	P_AD(17)	H22
GND	V23	P_AD(52)	B7	P_AD(16)	G21
GND	Y4	P_AD(51)	D6	P_AD(15)	B20
GND	Y8	P_AD(50)	B6	P_AD(14)	G22

Table 8-1. Signal Pin Listing by Signal Name

Signal Name	Grid Position	Signal Name	Grid Position	Signal Name	Grid Position
P_AD(13)	F20	P_SERR#	B4	S_AD(31)	N22
P_AD(12)	F22	P_STOP#	C4	S_AD(30)	N21
P_AD(11)	D18	P_TRDY#	B15	S_AD(29)	P22
P_AD(10)	C19	RESERVED2	D1	S_AD(28)	P21
P_AD(09)	C17	S_ACK64#	AA8	S_AD(27)	M23
P_AD(08)	B17	S_AD(63)	AB8	S_AD(26)	P20
P_AD(07)	A20	S_AD(62)	AB7	S_AD(25)	N23
P_AD(06)	C16	S_AD(61)	AA7	S_AD(24)	R22
P_AD(05)	B16	S_AD(60)	AB6	S_AD(23)	T23
P_AD(04)	A19	S_AD(59)	AA6	S_AD(22)	R21
P_AD(03)	C15	S_AD(58)	AA5	S_AD(21)	W23
P_AD(02)	B14	S_AD(57)	Y6	S_AD(20)	T22
P_AD(01)	C13	S_AD(56)	Y3	S_AD(19)	U22
P_AD(00)	B13	S_AD(55)	V2	S_AD(18)	U21
P_C/BE(7)#	A7	S_AD(54)	V4	S_AD(17)	V22
P_C/BE(6)#	B12	S_AD(53)	U2	S_AD(16)	V21
P_C/BE(5)#	C11	S_AD(52)	U3	S_AD(15)	W21
P_C/BE(4)#	A5	S_AD(51)	T2	S_AD(14)	V20
P_C/BE(3)#	A15	S_AD(50)	T3	S_AD(13)	AA20
P_C/BE(2)#	D14	S_AD(49)	R2	S_AD(12)	AB18
P_C/BE(1)#	B18	S_AD(48)	R3	S_AD(11)	Y18
P_C/BE(0)#	A13	S_AD(47)	P2	S_AD(10)	AA16
P_CFG_BUSY	C6	S_AD(46)	Y1	S_AD(09)	AB15
P_CLK	E21	S_AD(45)	P3	S_AD(08)	AC17
P_DEVSEL#	D21	S_AD(44)	W1	S_AD(07)	AA13
P_DRVR_MODE	E2	S_AD(43)	P4	S_AD(06)	AA12
P_FRAME#	A17	S_AD(42)	U1	S_AD(05)	AC15
P_GNT#	C20	S_AD(41)	N2	S_AD(04)	AB11
P_IDSEL	B19	S_AD(40)	N3	S_AD(03)	AC11
P_IRDY#	A16	S_AD(39)	M2	S_AD(02)	AC9
P_LOCK#	C14	S_AD(38)	M3	S_AD(01)	AB9
P_PAR	C18	S_AD(37)	R1	S_AD(00)	AA9
P_PAR64	A9	S_AD(36)	L2	S_C/BE(7)#	Y10
P_PERR#	C8	S_AD(35)	L3	S_C/BE(6)#	AB10
P_REQ#	B21	S_AD(34)	K2	S_C/BE(5)#	AA11
P_REQ64#	C12	S_AD(33)	K3	S_C/BE(4)#	AC8
P_RST#	E22	S_AD(32)	K4	S_C/BE(3)#	AA15

Table 8-1. Signal Pin Listing by Signal Name

Signal Name	Grid Position	Signal Name	Grid Position	Signal Name	Grid Position
S_C/BE(2)#	AB14	S_REQ64#	AB13	VDD2	A8
S_C/BE(1)#	AB16	S_RST#	U23	VDD2	A12
S_C/BE(0)#	AB12	S_SEL100	V3	VDD2	A22
S_CLK	AB23	S_SERR#	AB19	VDD2	C5
S_CLK_STABLE	W3	S_STOP#	AB20	VDD2	D5
S_DEVSEL#	AC21	S_TRDY#	Y14	VDD2	D7
S_DRVR_MODE	AC7	TEST_CE0	Y23	VDD2	D17
S_FRAME#	AA14	T_DI1	Y21	VDD2	D19
S_GNT1REQ#	AA19	T_DI2	AA4	VDD2	E4
S_GNT2#	AB1	T_MODECTL	C1	VDD2	E20
S_GNT3#	Y2	T_RI	W22	VDD2	G4
S_GNT4#	AC5	P_VDDA	A21	VDD2	G20
S_GNT5#	AB4	S_VDDA	AB21	VDD2	H23
S_GNT6#	AC4	VDD	D9	VDD2	M1
S_IDSEL	AA22	VDD	D11	VDD2	T1
S_INT_ARB_EN#	T21	VDD	D13	VDD2	U4
S_IRDY#	AC19	VDD	D15	VDD2	U20
S_LOCK#	AC20	VDD	J4	VDD2	W4
S_PAR	AA17	VDD	J20	VDD2	W20
S_PAR64	AA10	VDD	L4	VDD2	Y5
S_PCIXCAP	R23	VDD	L20	VDD2	Y7
S_PCIXCAP_PU	AA1	VDD	N4	VDD2	Y17
S_PERR#	AB17	VDD	N20	VDD2	Y19
S_REQ1GNT#	AA23	VDD	R4	VDD2	AC2
S_REQ2#	AA2	VDD	R20	VDD2	AC12
S_REQ3#	W2	VDD	Y9	VDD2	AC16
S_REQ4#	AB3	VDD	Y11	XCLK_OUT	D3
S_REQ5#	AB5	VDD	Y13		
S_REQ6#	AC3	VDD	Y15		

8.3 Complete Signal Pin List, Sorted by Grid Position

Table 8-2. Signal Pin Listing by Grid Position

Signal Name	Grid Position	Signal Name	Grid Position	Signal Name	Grid Position
GND	A1	P_AD(05)	B16	GND	D8
P_ACK64#	A2	P_AD(08)	B17	VDD	D9
P_AD(56)	A3	P_C/BE(1)#	B18	P_AD(62)	D10
P_AD(60)	A4	P_IDSEL	B19	VDD	D11
P_C/BE(4)#	A5	P_AD(15)	B20	GND	D12
GND	A6	P_REQ#	B21	VDD	D13
P_C/BE(7)#	A7	GND	B22	P_C/BE(2)#	D14
VDD2	A8	JTG_TDO	B23	VDD	D15
P_PAR64	A9	T_MODECTL	C1	GND	D16
GND	A10	P_AD(48)	C2	VDD2	D17
GND	A11	GND	C3	P_AD(11)	D18
VDD2	A12	P_STOP#	C4	VDD2	D19
P_C/BE(0)#	A13	VDD2	C5	GND	D20
GND	A14	P_CFG_BUSY	C6	P_DEVSEL#	D21
P_C/BE(3)#	A15	P_AD(53)	C7	JTG_TMS	D22
P_IRDY#	A16	P_PERR#	C8	P_AD(22)	D23
P_FRAME#	A17	P_AD(58)	C9	P_AD(38)	E1
GND	A18	P_AD(61)	C10	P_DRVR_MODE	E2
P_AD(04)	A19	P_C/BE(5)#	C11	P_AD(45)	E3
P_AD(07)	A20	P_REQ64#	C12	VDD2	E4
P_VDDA	A21	P_AD(01)	C13	VDD2	E20
VDD2	A22	P_LOCK#	C14	P_CLK	E21
GND	A23	P_AD(03)	C15	P_RST#	E22
P_AD(43)	B1	P_AD(06)	C16	P_AD(24)	E23
GND	B2	P_AD(09)	C17	GND	F1
P_AD(54)	B3	P_PAR	C18	P_AD(42)	F2
P_SERR#	B4	P_AD(10)	C19	P_AD(44)	F3
P_AD(49)	B5	P_GNT#	C20	P_AD(46)	F4
P_AD(50)	B6	GND	C21	P_AD(13)	F20
P_AD(52)	B7	JTG_TDI	C22	JTG_TCK	F21
P_AD(55)	B8	JTG_TRST#	C23	P_AD(12)	F22
P_AD(57)	B9	RESERVED2	D1	GND	F23
P_AD(59)	B10	P_AD(47)	D2	P_AD(36)	G1
P_AD(63)	B11	XCLK_OUT	D3	BAR_EN	G2
P_C/BE(6)#	B12	GND	D4	P_AD(41)	G3
P_AD(00)	B13	VDD2	D5	VDD2	G4
P_AD(02)	B14	P_AD(51)	D6	VDD2	G20
P_TRDY#	B15	VDD2	D7	P_AD(16)	G21

Table 8-2. Signal Pin Listing by Grid Position

Signal Name	Grid Position	Signal Name	Grid Position	Signal Name	Grid Position
P_AD(14)	G22	P_AD(30)	M21	VDD2	U20
P_AD(26)	G23	P_AD(29)	M22	S_AD(18)	U21
P_AD(35)	H1	S_AD(27)	M23	S_AD(19)	U22
P_AD(39)	H2	GND	N1	S_RST#	U23
P_AD(40)	H3	S_AD(41)	N2	GND	V1
GND	H4	S_AD(40)	N3	S_AD(55)	V2
GND	H20	VDD	N4	S_SEL100	V3
P_AD(18)	H21	VDD	N20	S_AD(54)	V4
P_AD(17)	H22	S_AD(30)	N21	S_AD(14)	V20
VDD2	H23	S_AD(31)	N22	S_AD(16)	V21
P_AD(33)	J1	S_AD(25)	N23	S_AD(17)	V22
P_AD(34)	J2	GND	P1	GND	V23
P_AD(37)	J3	S_AD(47)	P2	S_AD(44)	W1
VDD	J4	S_AD(45)	P3	S_REQ3#	W2
VDD	J20	S_AD(43)	P4	S_CLK_STABLE	W3
P_AD(20)	J21	S_AD(26)	P20	VDD2	W4
P_AD(19)	J22	S_AD(28)	P21	VDD2	W20
P_AD(31)	J23	S_AD(29)	P22	S_AD(15)	W21
GND	K1	GND	P23	T_RI	W22
S_AD(34)	K2	S_AD(37)	R1	S_AD(21)	W23
S_AD(33)	K3	S_AD(49)	R2	S_AD(46)	Y1
S_AD(32)	K4	S_AD(48)	R3	S_GNT3#	Y2
P_AD(25)	K20	VDD	R4	S_AD(56)	Y3
P_AD(23)	K21	VDD	R20	GND	Y4
P_AD(21)	K22	S_AD(22)	R21	VDD2	Y5
GND	K23	S_AD(24)	R22	S_AD(57)	Y6
P_AD(32)	L1	S_PCIXCAP	R23	VDD2	Y7
S_AD(36)	L2	VDD2	T1	GND	Y8
S_AD(35)	L3	S_AD(51)	T2	VDD	Y9
VDD	L4	S_AD(50)	T3	S_C/BE(7)#	Y10
VDD	L20	GND	T4	VDD	Y11
P_AD(28)	L21	GND	T20	GND	Y12
P_AD(27)	L22	S_INT_ARB_EN#	T21	VDD	Y13
GND	L23	S_AD(20)	T22	S_TRDY#	Y14
VDD2	M1	S_AD(23)	T23	VDD	Y15
S_AD(39)	M2	S_AD(42)	U1	GND	Y16
S_AD(38)	M3	S_AD(53)	U2	VDD2	Y17
GND	M4	S_AD(52)	U3	S_AD(11)	Y18
GND	M20	VDD2	U4	VDD2	Y19

Table 8-2. Signal Pin Listing by Grid Position

Signal Name	Grid Position	Signal Name	Grid Position	Signal Name	Grid Position
GND	Y20	S_IDSEL	AA22	GND	AC1
T_DI1	Y21	S_REQ1GNT#	AA23	VDD2	AC2
64_BIT_DEVICE#	Y22	S_GNT2#	AB1	S_REQ6#	AC3
TEST_CE0	Y23	GND	AB2	S_GNT6#	AC4
S_PCIXCAP_PU	AA1	S_REQ4#	AB3	S_GNT4#	AC5
S_REQ2#	AA2	S_GNT5#	AB4	GND	AC6
GND	AA3	S_REQ5#	AB5	S_DRVR_MODE	AC7
T_DI2	AA4	S_AD(60)	AB6	S_C/BE(4)#	AC8
S_AD(58)	AA5	S_AD(62)	AB7	S_AD(02)	AC9
S_AD(59)	AA6	S_AD(63)	AB8	GND	AC10
S_AD(61)	AA7	S_AD(01)	AB9	S_AD(03)	AC11
S_ACK64#	AA8	S_C/BE(6)#	AB10	VDD2	AC12
S_AD(00)	AA9	S_AD(04)	AB11	GND	AC13
S_PAR64	AA10	S_C/BE(0)#	AB12	GND	AC14
S_C/BE(5)#	AA11	S_REQ64#	AB13	S_AD(05)	AC15
S_AD(06)	AA12	S_C/BE(2)#	AB14	VDD2	AC16
S_AD(07)	AA13	S_AD(09)	AB15	S_AD(08)	AC17
S_FRAME#	AA14	S_C/BE(1)#	AB16	GND	AC18
S_C/BE(3)#	AA15	S_PERR#	AB17	S_IRDY#	AC19
S_AD(10)	AA16	S_AD(12)	AB18	S_LOCK#	AC20
S_PAR	AA17	S_SERR#	AB19	S_DEVSEL#	AC21
OPAQUE_EN	AA18	S_STOP#	AB20	IDSEL_REROUTE_EN	AC22
S_GNT1REQ#	AA19	S_VDDA	AB21	GND	AC23
S_AD(13)	AA20	GND	AB22		
GND	AA21	S_CLK	AB23		

9. JTAG Boundary Scan

The IBM 133 PCI-X Bridge R2.0 provides a JTAG test port that is compliant with IEEE Standard 1149.1, *IEEE Standard Test Access Port and Boundary-Scan Architecture,* to facilitate card and board testing using boundary-scan techniques. This function consists of the following major components:

- the five signal test port interface with the signals JTG_TCK, JTG_TDI, JTG_TDO, JTG_TMS, and JTG_TRST# (see *Signal Descriptions* on page 109 for further details on these signals);
- the test access port (TAP) controller;
- the instruction register (IR) and its associated instruction decoder;
- the bypass register;
- a device ID register, and
- the boundary-scan register.

The JTAG test port is intended to be used only while the bridge is not operating in a functional mode.

The information presented in this section is available in a standardized, soft-copy format called a boundary scan description language (BSDL) file. Contact your IBM sales representative for further information.

9.1 TAP Controller Initialization

After power-up of the IBM 133 PCI-X Bridge R2.0 device, the TAP Controller must be put into its test-logic-reset state to disable the JTAG logic and allow the bridge to function normally. This may be done by driving the JTG_TMS signal high and pulsing the JTG_TCK signal five or more times or by asserting the JTG_TRST# signal. If the bridge will not be connected into a boundary scan ring, the JTG_TRST# signal may be tied directly to GND.

9.2 Instruction Register and Codes

The IBM 133 PCI-X Bridge R2.0 implements a 4-bit Instruction register to control the operation of the JTAG logic. The defined instruction codes are shown in *Table 9-1*. Those bit combinations that are not listed are equivalent to the BYPASS (b‘1111’) instruction:

Table 9-1. JTAG Logic Instruction Codes

Instruction Code	Instruction Name	Register Selected	Operation
b‘0000’	EXTEST	Boundary-Scan	Drives / receives off-chip test data
b‘0100’	SAMPLE	Boundary-Scan	Samples inputs / pre-loads outputs
b‘0101’	HIGHZ	Bypass	Tri-states outputs
b‘0110’	IDCODE	Device ID	Accesses the Device ID register, to read manufacturer ID, part number, and version number
b‘1111’	BYPASS	Bypass	Selects Bypass register

9.3 Bypass Register

This register is a 1-bit shift register that provides a single bit scan path between the JTG_TDI input and the JTG_TDO output. This abbreviated scan path is selected by the BYPASS instruction code and is used to shorten the overall scan ring length during board-level testing when the bridge is not involved.

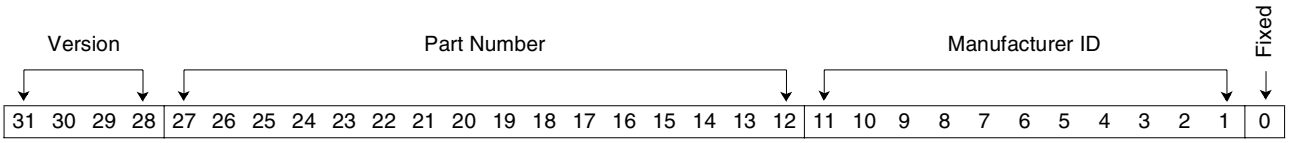
9.4 Device ID Register

This register, defined by IEEE Standard 1149.1, identifies IBM as the manufacturer of the IBM 133 PCI-X Bridge R2.0 and provides a portion of the die part number and a revision code for the device.

Width 32 bits

Access Read only

Reset Value x‘1490049’



Bits	Access	Field Name and Description
31:28	RO	Version number of the device.
27:12	RO	Last four hex digits of the BEOL die part number.
11:1	RO	JEDEC-assigned identifier for manufacturer (IBM).
0	RO	Fixed bit equal to b‘1’.

9.5 Boundary-Scan Register

The Boundary-Scan register is formed by connecting boundary scan cells placed at the device’s signal pins into a shift register path. The input to the shift register is the JTG_TDI signal and the output from the shift register is the JTG_TDO signal. When selected and controlled properly by the TAP Controller, this structure provides the output drive and input visibility features necessary for effective boundary-scan testing.

Several varieties of boundary-scan cells are possible, selected according to the function of the signal pin with which they are associated:

Input-only pins: For device inputs that are not shared with component test functions, the boundary-scan cell IBM1149_BSR_BIDIIN is used. Inputs that are shared with component test functions use the IBM1149_BSR_IN boundary-scan cell. Both have the BSDL function label of ‘INPUT’.

Output-only pins: For device outputs that are not shared with component test functions, the boundary-scan cell IBM1149_BSR_BIDIOUT is used. Outputs that are shared with component test inputs use the IBM1149_BSR_OUT boundary-scan cell. Those that are shared with component test outputs use the IBM1149_BSR_TESTOUT boundary-scan cell. The BSDL function label of OUTPUT2 is used for two-state outputs, OUTPUT3 is used for three-state outputs.

Bidirectional pins: For bidirectional device pins that are not shared with component test outputs, the boundary-scan cell IBM1149_BSR_BIDI is used. Bidirectional pins that are shared with component test inputs use the IBM1149_BSR_BIDI_SIO boundary-scan cell. Those that are shared with component test outputs use the IBM1149_BSR_BIDITESTOUT boundary-scan cell. All have the BSDL function label of BIDIR.

Control Functions: Two different boundary-scan cells are used to control the device's driver enables, these are IBM1149_BSR_ENAB and IBM1149_BSR_FASTENAB. While they perform the same boundary-scan function, they differ in that the IBM1149_BSR_FASTENAB cell has a faster functional path through it. It is often used on timing-critical signals. Both have the BSDL function label of CONTROL.

9.5.1 Boundary-Scan Register Bit Map

Table 9-2 gives the mapping between the bit position in the boundary-scan register and the signal or control function that it represents. The index is ordered from JTG_TDO to JTG_TDI. The value on JTG_TDI is applied on the rising edge of JTG_TCK, and JTG_TDO is valid on the falling edge of JTG_TCK. The JTG_TDO driver is enabled by the TAP controller, all other tristate drivers are controlled by the boundary-scan register control cells listed in the *Table 9-2*, and by the HIGHZ instruction code.

Table 9-2. Boundary-Scan Register Bit Map (Page 1 of 12)

Bit Position	Cell Type	Port Name	Function	Tristate Control Cell
1	IBM1149_BSR_BIDI	P_AD(0)	BIDIR	209
2	IBM1149_BSR_BIDI	P_AD(1)	BIDIR	210
3	IBM1149_BSR_BIDI	P_AD(2)	BIDIR	211
4	IBM1149_BSR_BIDI	P_AD(3)	BIDIR	212
5	IBM1149_BSR_BIDI	P_AD(4)	BIDIR	213
6	IBM1149_BSR_BIDI	P_AD(5)	BIDIR	214
7	IBM1149_BSR_BIDI	P_AD(6)	BIDIR	215
8	IBM1149_BSR_BIDI	P_AD(7)	BIDIR	216
9	IBM1149_BSR_BIDI	P_AD(8)	BIDIR	217
10	IBM1149_BSR_BIDI	P_AD(9)	BIDIR	218
11	IBM1149_BSR_BIDI	P_AD(10)	BIDIR	219
12	IBM1149_BSR_BIDI	P_AD(11)	BIDIR	220
13	IBM1149_BSR_BIDI	P_AD(12)	BIDIR	221
14	IBM1149_BSR_BIDI	P_AD(13)	BIDIR	222

Table 9-2. Boundary-Scan Register Bit Map (Page 2 of 12)

Bit Position	Cell Type	Port Name	Function	Tristate Control Cell
15	IBM1149_BSR_BIDI	P_AD(14)	BIDIR	223
16	IBM1149_BSR_BIDI_SIO	P_AD(15)	BIDIR	224
17	IBM1149_BSR_BIDI	P_AD(16)	BIDIR	225
18	IBM1149_BSR_BIDI	P_AD(17)	BIDIR	226
19	IBM1149_BSR_BIDI	P_AD(18)	BIDIR	227
20	IBM1149_BSR_BIDI	P_AD(19)	BIDIR	228
21	IBM1149_BSR_BIDI	P_AD(20)	BIDIR	229
22	IBM1149_BSR_BIDI	P_AD(21)	BIDIR	230
23	IBM1149_BSR_BIDI	P_AD(22)	BIDIR	231
24	IBM1149_BSR_BIDI	P_AD(23)	BIDIR	232
25	IBM1149_BSR_BIDI	P_AD(24)	BIDIR	233
26	IBM1149_BSR_BIDI	P_AD(25)	BIDIR	234
27	IBM1149_BSR_BIDI	P_AD(26)	BIDIR	235
28	IBM1149_BSR_BIDI	P_AD(27)	BIDIR	236
29	IBM1149_BSR_BIDI	P_AD(28)	BIDIR	237
30	IBM1149_BSR_BIDI	P_AD(29)	BIDIR	238
31	IBM1149_BSR_BIDI	P_AD(30)	BIDIR	239
32	IBM1149_BSR_BIDI	P_AD(31)	BIDIR	240
33	IBM1149_BSR_BIDI	P_AD(32)	BIDIR	241
34	IBM1149_BSR_BIDI	P_AD(33)	BIDIR	242
35	IBM1149_BSR_BIDI	P_AD(34)	BIDIR	243
36	IBM1149_BSR_BIDI	P_AD(35)	BIDIR	244
37	IBM1149_BSR_BIDI	P_AD(36)	BIDIR	245
38	IBM1149_BSR_BIDI	P_AD(37)	BIDIR	246
39	IBM1149_BSR_BIDI	P_AD(38)	BIDIR	247
40	IBM1149_BSR_BIDI	P_AD(39)	BIDIR	248
41	IBM1149_BSR_BIDI	P_AD(40)	BIDIR	249
42	IBM1149_BSR_BIDI	P_AD(41)	BIDIR	250
43	IBM1149_BSR_BIDI	P_AD(42)	BIDIR	251
44	IBM1149_BSR_BIDI	P_AD(43)	BIDIR	252
45	IBM1149_BSR_BIDI	P_AD(44)	BIDIR	253
46	IBM1149_BSR_BIDI	P_AD(45)	BIDIR	254
47	IBM1149_BSR_BIDI	P_AD(46)	BIDIR	255
48	IBM1149_BSR_BIDI	P_AD(47)	BIDIR	256
49	IBM1149_BSR_BIDI	P_AD(48)	BIDIR	257
50	IBM1149_BSR_BIDI	P_AD(49)	BIDIR	258

Table 9-2. Boundary-Scan Register Bit Map (Page 3 of 12)

Bit Position	Cell Type	Port Name	Function	Tristate Control Cell
51	IBM1149_BSR_BIDI	P_AD(50)	BIDIR	259
52	IBM1149_BSR_BIDI	P_AD(51)	BIDIR	260
53	IBM1149_BSR_BIDI	P_AD(52)	BIDIR	261
54	IBM1149_BSR_BIDI	P_AD(53)	BIDIR	262
55	IBM1149_BSR_BIDI	P_AD(54)	BIDIR	263
56	IBM1149_BSR_BIDI	P_AD(55)	BIDIR	264
57	IBM1149_BSR_BIDITESTOUT	P_AD(56)	BIDIR	265
58	IBM1149_BSR_BIDI	P_AD(57)	BIDIR	266
59	IBM1149_BSR_BIDI	P_AD(58)	BIDIR	267
60	IBM1149_BSR_BIDI	P_AD(59)	BIDIR	268
61	IBM1149_BSR_BIDI	P_AD(60)	BIDIR	269
62	IBM1149_BSR_BIDI	P_AD(61)	BIDIR	270
63	IBM1149_BSR_BIDI	P_AD(62)	BIDIR	271
64	IBM1149_BSR_BIDI	P_AD(63)	BIDIR	272
65	IBM1149_BSR_BIDI	P_CBE(0)	BIDIR	273
66	IBM1149_BSR_BIDI	P_CBE(1)	BIDIR	274
67	IBM1149_BSR_BIDI	P_CBE(2)	BIDIR	275
68	IBM1149_BSR_BIDI	P_CBE(3)	BIDIR	276
69	IBM1149_BSR_BIDI	P_CBE(4)	BIDIR	277
70	IBM1149_BSR_BIDI	P_CBE(5)	BIDIR	278
71	IBM1149_BSR_BIDI	P_CBE(6)	BIDIR	279
72	IBM1149_BSR_BIDI	P_CBE(7)	BIDIR	280
73	IBM1149_BSR_IN	P_CLK	INPUT	---
74	IBM1149_BSR_BIDI	P_DEVSEL	BIDIR	281
75	IBM1149_BSR_BIDI	P_FRAME	BIDIR	282
76	IBM1149_BSR_IN	P_GNT	INPUT	---
77	IBM1149_BSR_IN_SIO	P_IDSEL	INPUT	---
78	IBM1149_BSR_BIDI	P_IRDY	BIDIR	283
79	IBM1149_BSR_BIDIIN	P_LOCK	INPUT	---
80	IBM1149_BSR_IN_SIO	P_DRVR_MODE	INPUT	---
81	IBM1149_BSR_BIDI	P_PAR	BIDIR	284
82	IBM1149_BSR_BIDI	P_PAR64	BIDIR	285
83	IBM1149_BSR_IN_SIO	P_CFG_BUSY	INPUT	---
84	IBM1149_BSR_BIDI	P_PERR	BIDIR	286
85	IBM1149_BSR_BIDI	P_REQ64	BIDIR	287
86	IBM1149_BSR_OUT	P_REQ	OUTPUT3	288

Table 9-2. Boundary-Scan Register Bit Map (Page 4 of 12)

Bit Position	Cell Type	Port Name	Function	Tristate Control Cell
87	IBM1149_BSR_IN	P_RST	INPUT	---
88	IBM1149_BSR_OUT	P_SERR	OUTPUT3	289
89	IBM1149_BSR_BIDI	P_STOP	BIDIR	290
90	IBM1149_BSR_BIDI	P_TRDY	BIDIR	291
91	IBM1149_BSR_BIDI	S_ACK64	BIDIR	292
92	IBM1149_BSR_BIDI	S_AD(0)	BIDIR	293
93	IBM1149_BSR_BIDI	S_AD(1)	BIDIR	294
94	IBM1149_BSR_BIDI	S_AD(2)	BIDIR	295
95	IBM1149_BSR_BIDI	S_AD(3)	BIDIR	296
96	IBM1149_BSR_BIDI	S_AD(4)	BIDIR	297
97	IBM1149_BSR_BIDI	S_AD(5)	BIDIR	298
98	IBM1149_BSR_BIDI	S_AD(6)	BIDIR	299
99	IBM1149_BSR_BIDI	S_AD(7)	BIDIR	300
100	IBM1149_BSR_BIDI	S_AD(8)	BIDIR	301
101	IBM1149_BSR_BIDI	S_AD(9)	BIDIR	302
102	IBM1149_BSR_BIDI	S_AD(10)	BIDIR	303
103	IBM1149_BSR_BIDI	S_AD(11)	BIDIR	304
104	IBM1149_BSR_BIDI	S_AD(12)	BIDIR	305
105	IBM1149_BSR_BIDI	S_AD(13)	BIDIR	306
106	IBM1149_BSR_BIDI	S_AD(14)	BIDIR	307
107	IBM1149_BSR_BIDI	S_AD(15)	BIDIR	308
108	IBM1149_BSR_BIDI	S_AD(16)	BIDIR	309
109	IBM1149_BSR_BIDI	S_AD(17)	BIDIR	310
110	IBM1149_BSR_BIDI	S_AD(18)	BIDIR	311
111	IBM1149_BSR_BIDI	S_AD(19)	BIDIR	312
112	IBM1149_BSR_BIDI	S_AD(20)	BIDIR	313
113	IBM1149_BSR_BIDI	S_AD(21)	BIDIR	314
114	IBM1149_BSR_BIDI	S_AD(22)	BIDIR	315
115	IBM1149_BSR_BIDI	S_AD(23)	BIDIR	316
116	IBM1149_BSR_BIDI	S_AD(24)	BIDIR	317
117	IBM1149_BSR_BIDI	S_AD(25)	BIDIR	318
118	IBM1149_BSR_BIDI	S_AD(26)	BIDIR	319
119	IBM1149_BSR_BIDI	S_AD(27)	BIDIR	320
120	IBM1149_BSR_BIDI	S_AD(28)	BIDIR	321
121	IBM1149_BSR_BIDI	S_AD(29)	BIDIR	322
122	IBM1149_BSR_BIDI	S_AD(30)	BIDIR	323

Table 9-2. Boundary-Scan Register Bit Map (Page 5 of 12)

Bit Position	Cell Type	Port Name	Function	Tristate Control Cell
123	IBM1149_BSR_BIDI	S_AD(31)	BIDIR	324
124	IBM1149_BSR_BIDI	S_AD(32)	BIDIR	325
125	IBM1149_BSR_BIDI	S_AD(33)	BIDIR	326
126	IBM1149_BSR_BIDI	S_AD(34)	BIDIR	327
127	IBM1149_BSR_BIDI	S_AD(35)	BIDIR	328
128	IBM1149_BSR_BIDI	S_AD(36)	BIDIR	329
129	IBM1149_BSR_BIDI	S_AD(37)	BIDIR	330
130	IBM1149_BSR_BIDI	S_AD(38)	BIDIR	331
131	IBM1149_BSR_BIDI	S_AD(39)	BIDIR	332
132	IBM1149_BSR_BIDI	S_AD(40)	BIDIR	333
133	IBM1149_BSR_BIDI	S_AD(41)	BIDIR	334
134	IBM1149_BSR_BIDI	S_AD(42)	BIDIR	335
135	IBM1149_BSR_BIDI	S_AD(43)	BIDIR	336
136	IBM1149_BSR_BIDI	S_AD(44)	BIDIR	337
137	IBM1149_BSR_BIDI	S_AD(45)	BIDIR	338
138	IBM1149_BSR_BIDI	S_AD(46)	BIDIR	339
139	IBM1149_BSR_BIDI	S_AD(47)	BIDIR	340
140	IBM1149_BSR_BIDI	S_AD(48)	BIDIR	341
141	IBM1149_BSR_BIDI	S_AD(49)	BIDIR	342
142	IBM1149_BSR_BIDI	S_AD(50)	BIDIR	343
143	IBM1149_BSR_BIDI	S_AD(51)	BIDIR	344
144	IBM1149_BSR_BIDI	S_AD(52)	BIDIR	345
145	IBM1149_BSR_BIDI	S_AD(53)	BIDIR	346
146	IBM1149_BSR_BIDI	S_AD(54)	BIDIR	347
147	IBM1149_BSR_BIDI	S_AD(55)	BIDIR	348
148	IBM1149_BSR_BIDI	S_AD(56)	BIDIR	349
149	IBM1149_BSR_BIDI	S_AD(57)	BIDIR	350
150	IBM1149_BSR_BIDI	S_AD(58)	BIDIR	351
151	IBM1149_BSR_BIDI	S_AD(59)	BIDIR	352
152	IBM1149_BSR_BIDI	S_AD(60)	BIDIR	353
153	IBM1149_BSR_BIDI	S_AD(61)	BIDIR	354
154	IBM1149_BSR_BIDI	S_AD(62)	BIDIR	355
155	IBM1149_BSR_BIDI	S_AD(63)	BIDIR	356
156	IBM1149_BSR_BIDI	S_CBE(0)	BIDIR	357
157	IBM1149_BSR_BIDI	S_CBE(1)	BIDIR	358
158	IBM1149_BSR_BIDI	S_CBE(2)	BIDIR	359

Table 9-2. Boundary-Scan Register Bit Map (Page 6 of 12)

Bit Position	Cell Type	Port Name	Function	Tristate Control Cell
159	IBM1149_BSR_BIDI	S_CBE(3)	BIDIR	360
160	IBM1149_BSR_BIDI	S_CBE(4)	BIDIR	361
161	IBM1149_BSR_BIDI	S_CBE(5)	BIDIR	362
162	IBM1149_BSR_BIDI	S_CBE(6)	BIDIR	363
163	IBM1149_BSR_BIDI	S_CBE(7)	BIDIR	364
164	IBM1149_BSR_IN	S_CLK	INPUT	---
165	IBM1149_BSR_IN	S_CLK_STABLE	INPUT	---
166	IBM1149_BSR_BIDI	S_DEVSEL	BIDIR	365
167	IBM1149_BSR_BIDI	S_FRAME	BIDIR	366
168	IBM1149_BSR_OUT	S_GNT1REQ	OUTPUT3	202
169	IBM1149_BSR_OUT	S_GNT2	OUTPUT3	203
170	IBM1149_BSR_OUT	S_GNT3	OUTPUT3	204
171	IBM1149_BSR_OUT	S_GNT4	OUTPUT3	205
172	IBM1149_BSR_OUT	S_GNT5	OUTPUT3	206
173	IBM1149_BSR_OUT	S_GNT6	OUTPUT3	207
174	IBM1149_BSR_BIDIIN	S_INT_ARB_EN	INPUT	---
175	IBM1149_BSR_BIDI	S_IRDY	BIDIR	367
176	IBM1149_BSR_BIDI	S_LOCK	BIDIR	368
177	IBM1149_BSR_IN	S_DRVR_MODE	INPUT	---
178	IBM1149_BSR_BIDI	S_PAR	BIDIR	369
179	IBM1149_BSR_BIDI	S_PAR64	BIDIR	370
180	IBM1149_BSR_BIDIIN	S_PCIXCAP	INPUT	---
181	IBM1149_BSR_OUT	S_PCIXCAP_PU	OUTPUT3	376
182	IBM1149_BSR_BIDI	S_PERR	BIDIR	371
183	IBM1149_BSR_IN	S_REQ1GNT	INPUT	---
184	IBM1149_BSR_IN	S_REQ2	INPUT	---
185	IBM1149_BSR_IN	S_REQ3	INPUT	---
186	IBM1149_BSR_IN	S_REQ4	INPUT	---
187	IBM1149_BSR_IN	S_REQ5	INPUT	---
188	IBM1149_BSR_BIDI	S_REQ64	BIDIR	372
189	IBM1149_BSR_IN	S_REQ6	INPUT	---
190	IBM1149_BSR_TESTOUT	S_RST	OUTPUT2	---
191	IBM1149_BSR_IN	S_SEL100	INPUT	---
192	IBM1149_BSR_IN	S_SERR	INPUT	---
193	IBM1149_BSR_BIDI	S_STOP	BIDIR	373
194	IBM1149_BSR_BIDI	S_TRDY	BIDIR	374

Table 9-2. Boundary-Scan Register Bit Map (Page 7 of 12)

Bit Position	Cell Type	Port Name	Function	Tristate Control Cell
195	IBM1149_BSR_BIDIIN	BAR_EN	INPUT	---
196	IBM1149_BSR_IN	RESERVED2	INPUT	---
197	IBM1149_BSR_BIDIOUT	XCLK_OUT	OUTPUT3	375
198	IBM1149_BSR_IN	S_IDSEL	INPUT	---
199	IBM1149_BSR_IN	EN64_BIT_DEVICE	INPUT	---
200	IBM1149_BSR_IN	IDSEL_REROUTE_EN INPUT	---	
201	IBM1149_BSR_IN	OPAQUE_EN	INPUT	---
202	IBM1149_BSR_FASTENAB	---	CONTROL	---
203	IBM1149_BSR_FASTENAB	---	CONTROL	---
204	IBM1149_BSR_FASTENAB	---	CONTROL	---
205	IBM1149_BSR_FASTENAB	---	CONTROL	---
206	IBM1149_BSR_FASTENAB	---	CONTROL	---
207	IBM1149_BSR_FASTENAB	---	CONTROL	---
208	IBM1149_BSR_FASTENAB	---	CONTROL	---
209	IBM1149_BSR_FASTENAB	---	CONTROL	---
210	IBM1149_BSR_FASTENAB	---	CONTROL	---
211	IBM1149_BSR_FASTENAB	---	CONTROL	---
212	IBM1149_BSR_FASTENAB	---	CONTROL	---
213	IBM1149_BSR_FASTENAB	---	CONTROL	---
214	IBM1149_BSR_FASTENAB	---	CONTROL	---
215	IBM1149_BSR_FASTENAB	---	CONTROL	---
216	IBM1149_BSR_FASTENAB	---	CONTROL	---
217	IBM1149_BSR_FASTENAB	---	CONTROL	---
218	IBM1149_BSR_FASTENAB	---	CONTROL	---
219	IBM1149_BSR_FASTENAB	---	CONTROL	---
220	IBM1149_BSR_FASTENAB	---	CONTROL	---
221	IBM1149_BSR_FASTENAB	---	CONTROL	---
222	IBM1149_BSR_FASTENAB	---	CONTROL	---
223	IBM1149_BSR_FASTENAB	---	CONTROL	---
224	IBM1149_BSR_FASTENAB	---	CONTROL	---
225	IBM1149_BSR_FASTENAB	---	CONTROL	---
226	IBM1149_BSR_FASTENAB	---	CONTROL	---
227	IBM1149_BSR_FASTENAB	---	CONTROL	---
228	IBM1149_BSR_FASTENAB	---	CONTROL	---
229	IBM1149_BSR_FASTENAB	---	CONTROL	---
230	IBM1149_BSR_FASTENAB	---	CONTROL	---

Table 9-2. Boundary-Scan Register Bit Map (Page 8 of 12)

Bit Position	Cell Type	Port Name	Function	Tristate Control Cell
231	IBM1149_BSR_FASTENAB	---	CONTROL	---
232	IBM1149_BSR_FASTENAB	---	CONTROL	---
233	IBM1149_BSR_FASTENAB	---	CONTROL	---
234	IBM1149_BSR_FASTENAB	---	CONTROL	---
235	IBM1149_BSR_FASTENAB	---	CONTROL	---
236	IBM1149_BSR_FASTENAB	---	CONTROL	---
237	IBM1149_BSR_FASTENAB	---	CONTROL	---
238	IBM1149_BSR_FASTENAB	---	CONTROL	---
239	IBM1149_BSR_FASTENAB	---	CONTROL	---
240	IBM1149_BSR_FASTENAB	---	CONTROL	---
241	IBM1149_BSR_FASTENAB	---	CONTROL	---
242	IBM1149_BSR_FASTENAB	---	CONTROL	---
243	IBM1149_BSR_FASTENAB	---	CONTROL	---
244	IBM1149_BSR_FASTENAB	---	CONTROL	---
245	IBM1149_BSR_FASTENAB	---	CONTROL	---
246	IBM1149_BSR_FASTENAB	---	CONTROL	---
247	IBM1149_BSR_FASTENAB	---	CONTROL	---
248	IBM1149_BSR_FASTENAB	---	CONTROL	---
249	IBM1149_BSR_FASTENAB	---	CONTROL	---
250	IBM1149_BSR_FASTENAB	---	CONTROL	---
251	IBM1149_BSR_FASTENAB	---	CONTROL	---
252	IBM1149_BSR_FASTENAB	---	CONTROL	---
253	IBM1149_BSR_FASTENAB	---	CONTROL	---
254	IBM1149_BSR_FASTENAB	---	CONTROL	---
255	IBM1149_BSR_FASTENAB	---	CONTROL	---
256	IBM1149_BSR_FASTENAB	---	CONTROL	---
257	IBM1149_BSR_FASTENAB	---	CONTROL	---
258	IBM1149_BSR_FASTENAB	---	CONTROL	---
259	IBM1149_BSR_FASTENAB	---	CONTROL	---
260	IBM1149_BSR_FASTENAB	---	CONTROL	---
261	IBM1149_BSR_FASTENAB	---	CONTROL	---
262	IBM1149_BSR_FASTENAB	---	CONTROL	---
263	IBM1149_BSR_FASTENAB	---	CONTROL	---
264	IBM1149_BSR_FASTENAB	---	CONTROL	---
265	IBM1149_BSR_FASTENAB	---	CONTROL	---
266	IBM1149_BSR_FASTENAB	---	CONTROL	---

Table 9-2. Boundary-Scan Register Bit Map (Page 9 of 12)

Bit Position	Cell Type	Port Name	Function	Tristate Control Cell
267	IBM1149_BSR_FASTENAB	---	CONTROL	---
268	IBM1149_BSR_FASTENAB	---	CONTROL	---
269	IBM1149_BSR_FASTENAB	---	CONTROL	---
270	IBM1149_BSR_FASTENAB	---	CONTROL	---
271	IBM1149_BSR_FASTENAB	---	CONTROL	---
272	IBM1149_BSR_FASTENAB	---	CONTROL	---
273	IBM1149_BSR_FASTENAB	---	CONTROL	---
274	IBM1149_BSR_FASTENAB	---	CONTROL	---
275	IBM1149_BSR_FASTENAB	---	CONTROL	---
276	IBM1149_BSR_FASTENAB	---	CONTROL	---
277	IBM1149_BSR_FASTENAB	---	CONTROL	---
278	IBM1149_BSR_FASTENAB	---	CONTROL	---
279	IBM1149_BSR_FASTENAB	---	CONTROL	---
280	IBM1149_BSR_FASTENAB	---	CONTROL	---
281	IBM1149_BSR_FASTENAB	---	CONTROL	---
282	IBM1149_BSR_FASTENAB	---	CONTROL	---
283	IBM1149_BSR_FASTENAB	---	CONTROL	---
284	IBM1149_BSR_FASTENAB	---	CONTROL	---
285	IBM1149_BSR_FASTENAB	---	CONTROL	---
286	IBM1149_BSR_FASTENAB	---	CONTROL	---
287	IBM1149_BSR_FASTENAB	---	CONTROL	---
288	IBM1149_BSR_FASTENAB	---	CONTROL	---
289	IBM1149_BSR_FASTENAB	---	CONTROL	---
290	IBM1149_BSR_FASTENAB	---	CONTROL	---
291	IBM1149_BSR_FASTENAB	---	CONTROL	---
292	IBM1149_BSR_FASTENAB	---	CONTROL	---
293	IBM1149_BSR_FASTENAB	---	CONTROL	---
294	IBM1149_BSR_FASTENAB	---	CONTROL	---
295	IBM1149_BSR_FASTENAB	---	CONTROL	---
296	IBM1149_BSR_FASTENAB	---	CONTROL	---
297	IBM1149_BSR_FASTENAB	---	CONTROL	---
298	IBM1149_BSR_FASTENAB	---	CONTROL	---
299	IBM1149_BSR_FASTENAB	---	CONTROL	---
300	IBM1149_BSR_FASTENAB	---	CONTROL	---
301	IBM1149_BSR_FASTENAB	---	CONTROL	---
302	IBM1149_BSR_FASTENAB	---	CONTROL	---

Table 9-2. Boundary-Scan Register Bit Map (Page 10 of 12)

Bit Position	Cell Type	Port Name	Function	Tristate Control Cell
303	IBM1149_BSR_FASTENAB	---	CONTROL	---
304	IBM1149_BSR_FASTENAB	---	CONTROL	---
305	IBM1149_BSR_FASTENAB	---	CONTROL	---
306	IBM1149_BSR_FASTENAB	---	CONTROL	---
307	IBM1149_BSR_FASTENAB	---	CONTROL	---
308	IBM1149_BSR_FASTENAB	---	CONTROL	---
309	IBM1149_BSR_FASTENAB	---	CONTROL	---
310	IBM1149_BSR_FASTENAB	---	CONTROL	---
311	IBM1149_BSR_FASTENAB	---	CONTROL	---
312	IBM1149_BSR_FASTENAB	---	CONTROL	---
313	IBM1149_BSR_FASTENAB	---	CONTROL	---
314	IBM1149_BSR_FASTENAB	---	CONTROL	---
315	IBM1149_BSR_FASTENAB	---	CONTROL	---
316	IBM1149_BSR_FASTENAB	---	CONTROL	---
317	IBM1149_BSR_FASTENAB	---	CONTROL	---
318	IBM1149_BSR_FASTENAB	---	CONTROL	---
319	IBM1149_BSR_FASTENAB	---	CONTROL	---
320	IBM1149_BSR_FASTENAB	---	CONTROL	---
321	IBM1149_BSR_FASTENAB	---	CONTROL	---
322	IBM1149_BSR_FASTENAB	---	CONTROL	---
323	IBM1149_BSR_FASTENAB	---	CONTROL	---
324	IBM1149_BSR_FASTENAB	---	CONTROL	---
325	IBM1149_BSR_FASTENAB	---	CONTROL	---
326	IBM1149_BSR_FASTENAB	---	CONTROL	---
327	IBM1149_BSR_FASTENAB	---	CONTROL	---
328	IBM1149_BSR_FASTENAB	---	CONTROL	---
329	IBM1149_BSR_FASTENAB	---	CONTROL	---
330	IBM1149_BSR_FASTENAB	---	CONTROL	---
331	IBM1149_BSR_FASTENAB	---	CONTROL	---
332	IBM1149_BSR_FASTENAB	---	CONTROL	---
333	IBM1149_BSR_FASTENAB	---	CONTROL	---
334	IBM1149_BSR_FASTENAB	---	CONTROL	---
335	IBM1149_BSR_FASTENAB	---	CONTROL	---
336	IBM1149_BSR_FASTENAB	---	CONTROL	---
337	IBM1149_BSR_FASTENAB	---	CONTROL	---
338	IBM1149_BSR_FASTENAB	---	CONTROL	---

Table 9-2. Boundary-Scan Register Bit Map (Page 11 of 12)

Bit Position	Cell Type	Port Name	Function	Tristate Control Cell
339	IBM1149_BSR_FASTENAB	---	CONTROL	---
340	IBM1149_BSR_FASTENAB	---	CONTROL	---
341	IBM1149_BSR_FASTENAB	---	CONTROL	---
342	IBM1149_BSR_FASTENAB	---	CONTROL	---
343	IBM1149_BSR_FASTENAB	---	CONTROL	---
344	IBM1149_BSR_FASTENAB	---	CONTROL	---
345	IBM1149_BSR_FASTENAB	---	CONTROL	---
346	IBM1149_BSR_FASTENAB	---	CONTROL	---
347	IBM1149_BSR_FASTENAB	---	CONTROL	---
348	IBM1149_BSR_FASTENAB	---	CONTROL	---
349	IBM1149_BSR_FASTENAB	---	CONTROL	---
350	IBM1149_BSR_FASTENAB	---	CONTROL	---
351	IBM1149_BSR_FASTENAB	---	CONTROL	---
352	IBM1149_BSR_FASTENAB	---	CONTROL	---
353	IBM1149_BSR_FASTENAB	---	CONTROL	---
354	IBM1149_BSR_FASTENAB	---	CONTROL	---
355	IBM1149_BSR_FASTENAB	---	CONTROL	---
356	IBM1149_BSR_FASTENAB	---	CONTROL	---
357	IBM1149_BSR_FASTENAB	---	CONTROL	---
358	IBM1149_BSR_FASTENAB	---	CONTROL	---
359	IBM1149_BSR_FASTENAB	---	CONTROL	---
360	IBM1149_BSR_FASTENAB	---	CONTROL	---
361	IBM1149_BSR_FASTENAB	---	CONTROL	---
362	IBM1149_BSR_FASTENAB	---	CONTROL	---
363	IBM1149_BSR_FASTENAB	---	CONTROL	---
364	IBM1149_BSR_FASTENAB	---	CONTROL	---
365	IBM1149_BSR_FASTENAB	---	CONTROL	---
366	IBM1149_BSR_FASTENAB	---	CONTROL	---
367	IBM1149_BSR_FASTENAB	---	CONTROL	---
368	IBM1149_BSR_FASTENAB	---	CONTROL	---
369	IBM1149_BSR_FASTENAB	---	CONTROL	---
370	IBM1149_BSR_FASTENAB	---	CONTROL	---
371	IBM1149_BSR_FASTENAB	---	CONTROL	---
372	IBM1149_BSR_FASTENAB	---	CONTROL	---
373	IBM1149_BSR_FASTENAB	---	CONTROL	---
374	IBM1149_BSR_FASTENAB	---	CONTROL	---

Table 9-2. Boundary-Scan Register Bit Map (Page 12 of 12)

Bit Position	Cell Type	Port Name	Function	Tristate Control Cell
375	IBM1149_BSR_ENAB	---	CONTROL	---
376	IBM1149_BSR_ENAB	---	CONTROL	---

10. Electrical Information

This section specifies the electrical and thermal behavior of the IBM 133 PCI-X Bridge R2.0.

10.1 PCI/PCI-X Specification Conformance

Most of the IBM 133 PCI-X Bridge R2.0 interface signals are delineated by the PCI/PCI-X specifications. These interface signals conform to electrical requirements of the PCI/PCI-X specification documents, that information is not duplicated here.

Note: The IBM 133 PCI-X Bridge R2.0 uses the 3.3 V signaling environment, it is not a 5V-tolerant device.

10.2 Absolute Maximum Ratings

Stresses greater than the absolute maximum ratings listed in *Table 10-1* may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions outside of the limits indicated in this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect the reliability of the bridge.

Table 10-1. Absolute Maximum Ratings

Symbol	Parameter	Rating	Units
V_{DD}	Core Logic Power Supply Voltage		V
V_{DD2}	I/O Power Supply Voltage		V
V_{IN}	Input Voltage		V
V_{OUT}	Output Voltage		V
T_A	Operating Temperature (ambient)	0 to +70	°C
T_J	Maximum Junction Temperature	+125	°C
T_{STG}	Storage Temperature	-55 to +125	°C
P_{WC}	Worst Case Power Dissipation	4.0	W
I_{OUT}	Short Circuit Output Current		mA

10.3 Recommended DC Operating Conditions

Table 10-2. Recommended DC Operating Conditions (T_A= 0 to 70°C)

Symbol	Parameter	Rating			Units	Notes
		Min.	Typ.	Max.		
V_{DD}	Core Logic Power Supply Voltage	2.3	2.5	2.7	V	1
V_{DD2}	I/O Power Supply Voltage	3.0	3.3	3.6	V	1
V_{IH}	Input High Voltage	2.0	—	V_{DD2} + 0.3	V	1
V_{IL}	Input Low Voltage	-0.3	—	0.8	V	1
C_{IN}	Input Pin Capacitance	—	—	8.0	pF	

1. All voltages referenced to GND.

10.4 AC Operating Conditions

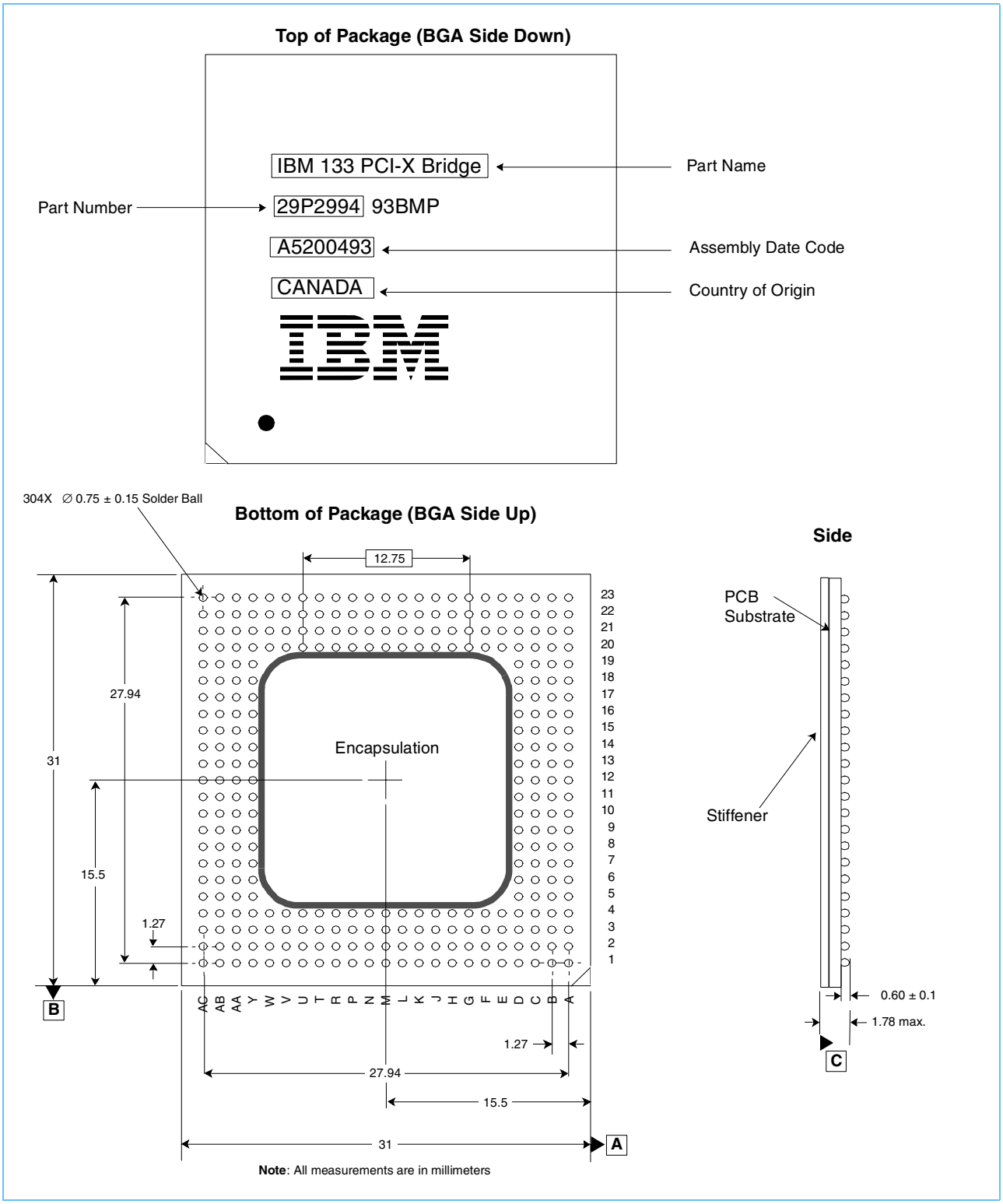
Table 10-3. AC Operating Conditions (T_A= 0 to 70°C)

Symbol	Parameter	Rating			Units	Notes
		Min.	Typ.	Max.		
$P_{WC\text{-}VDD}$	Worst case power dissipation from V_{DD}			3.1	W	1,3
$P_{WC\text{-}VDD2}$	Worst case power dissipation from V_{DD2}			1.2	W	1,2,3

1. Highly dependent on operating frequency and bus traffic.
2. Highly dependent on output load.
3. $P_{WC\text{-}VDD}$ and $P_{WC\text{-}VDD2}$ are inter-related but cannot occur at the same time: high output loads preclude operation at high frequencies and vice-versa.

11. Mechanical Information

Figure 11-1. Package Diagram 31 x 31 mm 304-lead H-PBGA Package



11.1 Package Information

Table 11-1. Package Information

Package Type	BGA
Leads (Pins)	304
Power Supplies	3.3 V 2.5 V

Revision Log

Date	Description of Modification
August 22, 2001	Initial Release.
August 30, 2001	Added *Table 9-2*.
Sep. 1, 2001	Changed P_M66EN and S_M66EN in Boundary Scan table 9-2 to P and S_DRVR_MODE
Sep. 4, 2001	Declassified
October 15, 2001	Short Term Caching function documented and PCIXCAP consistency added.