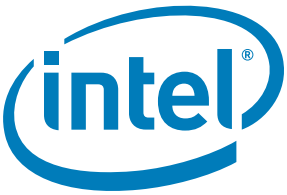


Intel® 5000X Chipset Memory Controller Hub (MCH)

Datasheet

May 2006

Document Number: 313070-001

Contents

Figures

Tables

Revision History

Revision Number	Description	Date
0.5	Initial release.	January 2004
001	Final release	May 2006

§

1 Introduction

The Intel® 5000X Chipset is designed for systems based on the Dual-Core Intel® Xeon® Processor 5000 Series and supports FSB frequencies of 1066 MTS. The Intel 5000X Chipset contains two main components: Memory Controller Hub (MCH) for the host bridge and the I/O controller hub for the I/O subsystem. The Intel 5000X Chipset uses the Intel® 631xESB/632xESB I/O Controller Hub for the I/O Controller Hub. This document is the datasheet for the Intel® 5000X Chipset Memory Controller Hub (MCH) components.

The Intel 5000X Chipset is packaged in a 1432 pin FCBGA package with pins on 1.092 mm (37 mil) centers. The overall package dimensions are 42.5 mm by 42.5 mm.

The Intel 5000X Chipset platform supports the Dual-Core Intel Xeon Processor 5000 series processor (1066 MHz) with 2 MB L2 cache on 90 nm process in a 771-land, FC-LGA4 (Flip Chip Land Grid Array 4) package. This package uses the matching LGA771 socket. The surface mount, LGA771 socket supports Direct Socket Loading (DSL). The Dual-Core Intel Xeon Processor 5000 series (1066 MHz) returns a processor signature of, 0F5xh where x is the stepping number, when the CPUID instruction is executed with EAX=1.

Note: Unless otherwise specified, the term processor in this document refers to the Dual-Core Intel Xeon Processor 5000 series processor (1066 MHz) with 2 MB L2 cache on 90 nm process in the 771-pin FC-LGA4 package.

1.1 Terminology

This section provides the definitions of some of the terms used in this document.

Table 1-1. General Terminology (Sheet 1 of 6)

Terminology	Description
Agent	A logical device connected to a bus or shared interconnect that can either initiate accesses or be the target of accesses. Each thread executing within a processor is a unique agent.
aka	also known as
Asserted	Asserted Signal is set to a level that represents logical true. For signals that end with “#” this means driving a low voltage. For other signals, it is a high voltage.
Atomic operation	A series of operations, any one of which cannot be observed to complete unless all are observed to complete.
AGP	Accelerated Graphics Port. In this document AGP refers to the AGP/PCI interface that is in the MCH. The MCH AGP interface supports only 0.8 V/1.5 V AGP 2.0/AGP 3.0 compliant devices using PCI (66 MHz), AGP 1X (66 MHz), 4X (266 MHz), and 8X (533 MHz) transfers. MCH does not support any 3.3 V devices. For AGP 2.0, PIPE# and SBA addressing cycles and their associated data phases are generally referred to as AGP transactions. FRAME# cycles are generally referred to as AGP/PCI transactions
Bank	DRAM chips are divided into multiple banks internally. Commodity parts are all 4 bank, which is the only type the MCH supports. Each bank acts somewhat like a separate DRAM, opening and closing pages independently, allowing different pages to be open in each. Most commands to a DRAM target a specific bank, but some commands (that is, Precharge All) are targeted at all banks. Multiple banks allows higher performance by interleaving the banks and reducing page miss cycles.
Buffer	1. A random access memory structure. 2. The term I/O buffer is also used to describe a low level input receiver and output driver combination.

Table 1-1. General Terminology (Sheet 2 of 6)

Terminology	Description
Cache Line	The unit of memory that is copied to and individually tracked in a cache. Specifically, 64 bytes of data or instructions aligned on a 64-byte physical address boundary.
CDM	Central Data Manager. A custom array within the Intel 5000X Chipset that acts as a temporary repository for system data in flight between the various ports: FSB's, FBD's, ESI, and PCI Express*.
Cfg, Config	Abbreviation for "Configuration".
Channel	In the MCH a FBD DRAM channel is the set of signals that connects to one set of FBD DIMMs. The MCH has up to four DRAM channels.
Character	The raw data byte in an encoded system (for example, the 8b value in a 8b/10b encoding scheme). This is the meaningful quantum of information to be transmitted or that is received across an encoded transmission path.
Chipset Core	The MCH internal base logic.
Coherent	Transactions that ensure that the processor's view of memory through the cache is consistent with that obtained through the I/O subsystem.
Command	The distinct phases, cycles, or packets that make up a transaction. Requests and completions are referred to generically as Commands.
Completion	A packet, phase, or cycle used to terminate a transaction on a interface, or within a component. A Completion will always refer to a preceding request and may or may not include data and/or other information.
Core	The internal base logic in the Intel 5000X Chipset.
CRC	Cyclic Redundancy Check; A number derived from, and stored or transmitted with, a block of data in order to detect corruption. By recalculating the CRC and comparing it to the value originally transmitted, the receiver can detect some types of transmission errors.
Critical Word First	On the DRAM, Processor, and Memory interfaces, the requestor may specify a particular word to be delivered first. This is done using address bits of lower significance than those required to specify the cache line to be accessed. The remaining data is then returned in a standardized specified order.
DDR	Double Data Rate SDRAM. DDR describes the type of DRAMs that transfers two data items per clock on each pin. This is the only type of DRAM supported by the MCH.
Deasserted	Signal is set to a level that represents logical false.
Deferred Transaction	A processor bus Split Transaction. On the processor bus, the requesting agent receives a Deferred Response which allows other transactions to occur on the bus. Later, the response agent completes the original request with a separate Deferred Reply transaction or by Deferred Phase.
Delayed Transaction	A transaction where the target retries an initial request, but without notification to the initiator, forwards or services the request on behalf of the initiator and stores the completion or the result of the request. The original initiator subsequently re-issues the request and receives the stored completion
DFx (DFD, DFM, DFT, DFV)	DFD=Design for Debug DFM=Design for Manufacturing DFT=Design for Testability DFV=Design for Validation
DIMM	Dual-in-Line Memory Module. A packaging arrangement of memory devices on a socketable substrate.
Double-Sided DIMM	Terminology often used to describe a DIMM that contain two DRAM rows. Generally a Double-sided DIMM contains two rows, with the exception noted above. This terminology is not used within this document.
Downstream	See Terminology entry of "Inbound (IB)/Outbound (OB), AKA Upstream/DownStream, Northbound/Southbound, Upbound/Downbound"
DRAM Page (Row)	The DRAM cells selected by the Row Address.
Dword	A reference to 32 bits of data on a naturally aligned four-byte boundary (that is, the least significant two bits of the address are 00b).
ECC	Error Correcting Code
ESB2	Intel® 631xESB/632xESB I/O Controller Hub

Table 1-1. General Terminology (Sheet 3 of 6)

Terminology	Description
FBD	Fully Buffered DDRII
FBD Channel	One electrical interface to one or more Fully Buffered DDRII DIMM.
FSB	Processor Front-Side Bus. This is the bus that connects the processor to the MCH.
Full Duplex	A connection or channel that allows data or messages to be transmitted in opposite directions simultaneously.
GART	Graphics Aperture Re-map Table. GART is a table in memory containing the page re-map information used during AGP aperture address translations.
GB/s	Gigabytes per second (10^9 bytes per second).
Gb/s	Gigabits per second (10^9 bits per second).
GTLB	Graphics Translation Look-aside Buffer. A cache used to store frequently used GART entries.
Hardwired	A parameter that has a fixed value.
Half Duplex	A connection or channel that allows data or messages to be transmitted in either direction, but not simultaneously.
Host	This term is used synonymously with processor.
I/O	1. Input/Output. 2. When used as a qualifier to a transaction type, specifies that transaction targets Intel architecture-specific I/O space. (for example, I/O read)
Intel® 631xESB/ 632xESB I/O Controller Hub	6th Generation I/O Controller Hub. The IO Controller Hub component that contains the legacy I/O functions.
Implicit Writeback	A snoop initiated data transfer from the bus agent with the modified Cache Line to the memory controller due to an access to that line.
Inband	Communication that is multiplexed on the standard lines of an interface, rather than requiring a dedicated signal.
Inbound	See Terminology entry of “Inbound (IB)/Outbound (OB), AKA Upstream/DownStream, Northbound/Southbound, Upbound/Downbound.”
Incoming	A transaction or data that enters the Intel 5000X Chipset.
Inbound (IB)/ Outbound (OB), AKA Upstream/ DownStream, Northbound/ Southbound, Upbound/Downbound	Up, North, or Inbound is in the direction of the processor, Down, South, or Outbound is in the direction of IO (SDRAM, SMBus).
Initiator	The source of requests. An agent sending a request packet on PCI Express is referred to as the Initiator for that transaction. The Initiator may receive a completion for the request.
Isochronous	A classification of transactions or a stream of transactions that require service within a fixed time interval.
Layer	A level of abstraction commonly used in interface specifications as a tool to group elements related to a basic function of the interface within a layer and to identify key interactions between layers.
Legacy	Functional requirements handed down from previous chipsets or PC compatibility requirements from the past.
Line	Cache line.
Link	The layer of an interface that handles flow control and often error correction by retry.
Lock	A sequence of transactions that must be completed atomically.
LSb	Least Significant Bit
LSB	Least Significant Byte
Master	A device or logical entity that is capable of initiating transactions. A Master is any potential Initiator.

Table 1-1. General Terminology (Sheet 4 of 6)

Terminology	Description
Master Abort	A response to an illegal request. Reads receive all ones. Writes have no effect.
MB/s	Megabytes per second (10^6 bytes per second)
MCH	The Memory Controller Hub component that contains the processor interface, DRAM controller, PCI Express interface, and AGP interface. It communicates with the I/O controller hub (Intel 631xESB/632xESB I/O Controller Hub) over a proprietary interconnect called the Enterprise South Bridge Interface (ESI).
Mem	Used as a qualifier for transactions that target memory space. (for example, A Mem read to I/O).
Memory Issue	Committing a request to DDR or, in the case of a read, returning the read header.
Mesochronous	Distributed or common referenced clock.
Metastability	A characteristic of flip flops that describes the state where the output becomes non-deterministic. Most commonly caused by a setup or hold time violation.
Mirroring	RAID-1. Please see RAID for detail descriptions.
MMIO	Memory Mapped IO. Any memory access to PCI Express or 3GIOC ports.
MMCFG	Memory Mapped Configuration. A memory transaction that accesses configuration space.
MSb	Most Significant Bit.
MSB	Most Significant Byte.
MTBF	Mean Time Between Failure.
Non-Coherent	Transactions that may cause the processor's view of memory through the cache to be different with that obtained through the I/O subsystem.
Outbound	See Terminology entry of “Inbound (IB)/Outbound (OB), AKA Upstream/DownStream, Northbound/Southbound, Upbound/Downbound.”
Outgoing	A transaction or completion that exits the Intel 5000X Chipset. Peer-to-Peer Transactions that occur between two devices below the PCI Express or ESI ports.
Packet	The indivisible unit of data transfer and routing, consisting of a header, data, and CRC.
Page Hit.	An access to an open page, or DRAM row. The data can be supplied from the sense amps at low latency.
Page Miss (Empty Page)	An access to a page that is not buffered in sense amps and must be fetched from DRAM array. Address Bit Permuting Address bits are distributed among channel selects, DRAM selects, bank selects to so that a linear address stream accesses these resources in a certain sequence.
Page Replace Aka Page Miss, Row Hit/ Page Miss.	An access to a row that has another page open. The page must be transferred back from the sense amps to the array, and the bank must be precharged.
PCI	Peripheral Component Interconnect Local Bus. A 32-bit or 64-bit bus with multiplexed address and data lines that is primarily intended for use as an interconnect mechanism within a system between processor/memory and peripheral components or add-in cards.
PCI 2.3 compliant	Refers to compliance to the *PCI Local Bus Specification*, Revision 2.3.
Plesiochronous	Each end of a link uses an independent clock reference. Support of this operational mode places restrictions on the absolute frequency difference, as specified by PCI Express, which can be tolerated between the two independent clock references.
Posted	A transaction that is considered complete by the initiating agent or source before it actually completes at the target of the request or destination. All agents or devices handling the request on behalf of the original Initiator must then treat the transaction as being system visible from the initiating interface all the way to the final destination. Commonly refers to memory writes.
Primary PCI	The physical PCI bus that is driven directly by the Intel® 631xESB/632xESB I/O Controller Hub component. Communication between PCI and the MCH occurs over ESI. Note that even though the Primary PCI bus is referred to as PCI it is **not** PCI Bus 0 from a configuration standpoint.
Push Model	Method of messaging or data transfer that predominately uses writes instead of reads.

Table 1-1. General Terminology (Sheet 5 of 6)

Terminology	Description
Queue	A storage structure for information. Anything that enters a queue will exit eventually. The most common policy to select an entry to read from the queue is FIFO (First In First Out).
RAID	Redundant Array of Independent Disks. RAID improves performance by disk striping, which interleaves bytes or groups of bytes across multiple drives, so more than one disk is reading and writing simultaneously. Fault tolerance is achieved by mirroring or parity. Mirroring is 100% duplication of the data on two drives (RAID-1), and parity is used (RAID-3 and 5) to calculate the data in two drives and store the results on a third: a bit from drive 1 is XOR'd with a bit from drive 2, and the result bit is stored on drive 3 (see OR for an explanation of XOR). A failed drive can be hot swapped with a new one, and the RAID controller automatically rebuilds the lost data. RAID can be classified into the following categories: RAID-0 • RAID-0 is disk striping only, which interleaves data across multiple disks for better performance. It does not provide safeguards against failure. RAID-1 • Uses disk mirroring, which provides 100% duplication of data. Offers highest reliability, but doubles storage cost. RAID-2 • Bits (rather than bytes or groups of bytes) are interleaved across multiple disks. The Connection Machine used this technique, but this is a rare method. RAID-3 • Data are striped across three or more drives. Used to achieve the highest data transfer, because all drives operate in parallel. Parity bits are stored on separate, dedicated drives. RAID-4 • Similar to RAID-3, but manages disks independently rather than in unison. Not often used. RAID-5 • Most widely used. Data are striped across three or more drives for performance, and parity bits are used for fault tolerance. The parity bits from two drives are stored on a third drive. RAID-6 • Highest reliability, but not widely used. Similar to RAID-5, but does two different parity computations or the same computation on overlapping subsets of the data. RAID-10 • Actually RAID-1,0. A combination of RAID-1 and RAID-0 (mirroring and striping). Above definitions can be extended to DRAM memory system as well. To avoid confusion, the RAID scheme for memory is referred as memory-RAID. Memory mirroring scheme is actually memory-RAID-1.
RASUM	Reliability, Availability, Serviceability, Usability, and Manageability, which are all important characteristics of servers.
Receiver, Rcvr	1. The Agent that receives a packet across an interface regardless of whether it is the ultimate destination of the packet. 2. More narrowly, the circuitry required to convert incoming signals from the physical medium to more perceptible forms.
Request	A packet, phase, or cycle used to initiate a transaction on a interface, or within a component.
Reserved	The contents or undefined states or information are not defined at this time. Using any reserved area is not permitted.
RMW	Read-Modify-Write operation.
Row	A group of DRAM chips that fill out the data bus width of the system and are accessed in parallel by each DRAM command.
Row Address	The row address is presented to the DRAMs during an activate command, and indicates which page to open within the specified bank (the bank number is presented also).
Scalable Bus	Processor-to-MCH interface. The compatible mode of the Scalable Bus is the P6 Bus. The enhanced mode of the Scalable Bus is the P6 Bus plus enhancements primarily consisting of source synchronous transfers for address and data, and FSB interrupt delivery. The Intel® Pentium® 4 processor implements a subset of the enhanced mode.

Table 1-1. General Terminology (Sheet 6 of 6)

Terminology	Description
SDDC	Single Device Disable Code; aka x4 or x8 chip-disable Hamming code to protect single DRAM device (x4 or x8 data width) failure.
SDR	Single Data Rate SDRAM.
SDRAM	Synchronous Dynamic Random Access Memory.
SEC/DED	Single-bit Error Correct / Double-symbol Error Detect
Secondary PCI	The physical PCI interface that is a subset of the AGP bus driven directly by the MCH. It supports a subset of 32-bit, 66 MHz PCI 2.0 compliant components, but only at 1.5 V (not 3.3 V or 5 V).
Serial Presence Detect (SPD)	A two-signal serial bus used to read and write Control registers in the SDRAMs via the SMBus protocol.
Single-Sided DIMM	Terminology often used to describe a DIMM that contains one DRAM row. Usually one row fits on a single side of the DIMM allowing the backside to be empty.
Simplex	A connection or channel that allows data or messages to be transmitted in one direction only.
SMBus	System Management Bus. Mastered by a system management controller to read and write configuration registers. Signaling and protocol are loosely based on I2C, limited to 100 KHz.
Snooping	A means of ensuring cache coherency by monitoring all coherent accesses on a common multi-drop bus to determine if an access is to information resident within a cache. The Intel 5000X Chipset MCH ensures coherency by initiating snoops on the processor busses with the address of any line that might appear in a cache on that bus.
Split Lock Sequence	A sequence of transactions that occurs when the target of a lock operation is split across a processor bus data alignment or Cache Line boundary, resulting in two read transactions and two write transactions to accomplish a read-modify-write operation.
Split Transaction	A transaction that consists of distinct Request and Completion phases or packets that allow use of bus, or interconnect, by other transactions while the Target is servicing the Request.
SSTL	Stub-Series Terminated Logic
SSTL_2	Stub Series Terminated Logic for 2.6 Volts (DDR)
Symbol	An expanded and encoded representation of a data Byte in an encoded system (for example, the 10-bit value in a 8-bit/10-bit encoding scheme). This is the value that is transmitted over the physical medium.
Symbol Time	The amount of time required to transmit a symbol.
System Bus	Processor-to-Intel 5000X Chipset interface. The system bus in this document refers to operation at 266/533/1066 MHz (Bus Clock/Address/Data). The system bus is not compatible with the P6 system bus.
Target	A device that responds to bus Transactions. The agent receiving a request packet is referred to as the Target for that Transaction.
Tenured Transaction	A transaction that holds the bus, or interconnect, until complete, effectively blocking all other transactions while the Target is servicing the Request.
TID	Transaction Identifier: A multi-bit field used to uniquely identify a transaction. Commonly used to relate a Completion with its originating Request in a Split Transaction system.
Transaction, Txn	An overloaded term that represents an operation between two or more agents that can be comprised of multiple phases, cycles, or packets.
Transmitter	1. The Agent that sends a Packet across an interface regardless of whether it was the original generator of the packet. 2. More narrowly, the circuitry required to drive signals onto the physical medium.
Upstream	See Terminology entry of “Inbound (IB)/Outbound (OB), AKA Upstream/DownStream, Northbound/Southbound, Upbound/Downbound”

1.2 Related Documents

Document	Document Number/ Location
Intel® 5000 Sequence Chipset: For use with Dual Core Intel® Xeon® 5000 Series Processor with 2-MB L2 Cache on 90 nm Process Platform Design Guide	http://developer.intel.com/design/
Intel® 5000 Chipset: Intel® 5000 Sequence Chipset Thermal Design Guide	http://developer.intel.com/design/chipsets/
Intel® 6402/6400 Advanced Memory Buffer Component External Design Specification	http://developer.intel.com/design/
Intel® 631xESB/632xESB I/O Controller Hub Datasheet	http://developer.intel.com/design/chipsets/
Dual-Core Intel® Xeon® Processor 5000 series (1066 MHz) Electrical, Mechanical, and Thermal Specifications (EMTS)	http://developer.intel.com/design/
Intel® 5000 Series Chipsets MCH BIOS Specification	http://developer.intel.com/design/
JEDEC FB-DIMM Memory Specification	www.jedec.org
PCI Local Bus Specification, Rev 2.3.	www.pcisig.org
PCI Express Interface Specification, Rev 1.0a	www.pcisig.org

1.3 Intel® 5000X Chipset Overview

Figure 1-1 shows an example block diagram of a Intel 5000X Chipset-based platform. The Intel 5000X Chipset is designed for use in high performance workstations based on the Dual-Core Intel Xeon Processor 5000 Series. The Intel 5000X Chipset supports two processors on independent point to point system buses operating at 333 MHz (1333 MTS). The bandwidth of the two processor busses is 21 GB/s.

Intel 5000X Chipset features a high performance PCI Express* graphics port capable of through puts of 4 GB/s. This graphics port contains several architectural enhancements designed to optimize graphics performance in demanding video applications. One of the architectural enhancements in Intel 5000X Chipset is the inclusion of a Snoop Filter to eliminate snoop traffic to the graphics port. Reduction of this traffic results in significant performance increases in graphics intensive applications.

The Dual-Core Intel Xeon Processor 5000 series has a 2 MB L2 cache, a 266 MHz (1066 MTS) system bus and fabricated using a 90 nm process in a 771-pin package.

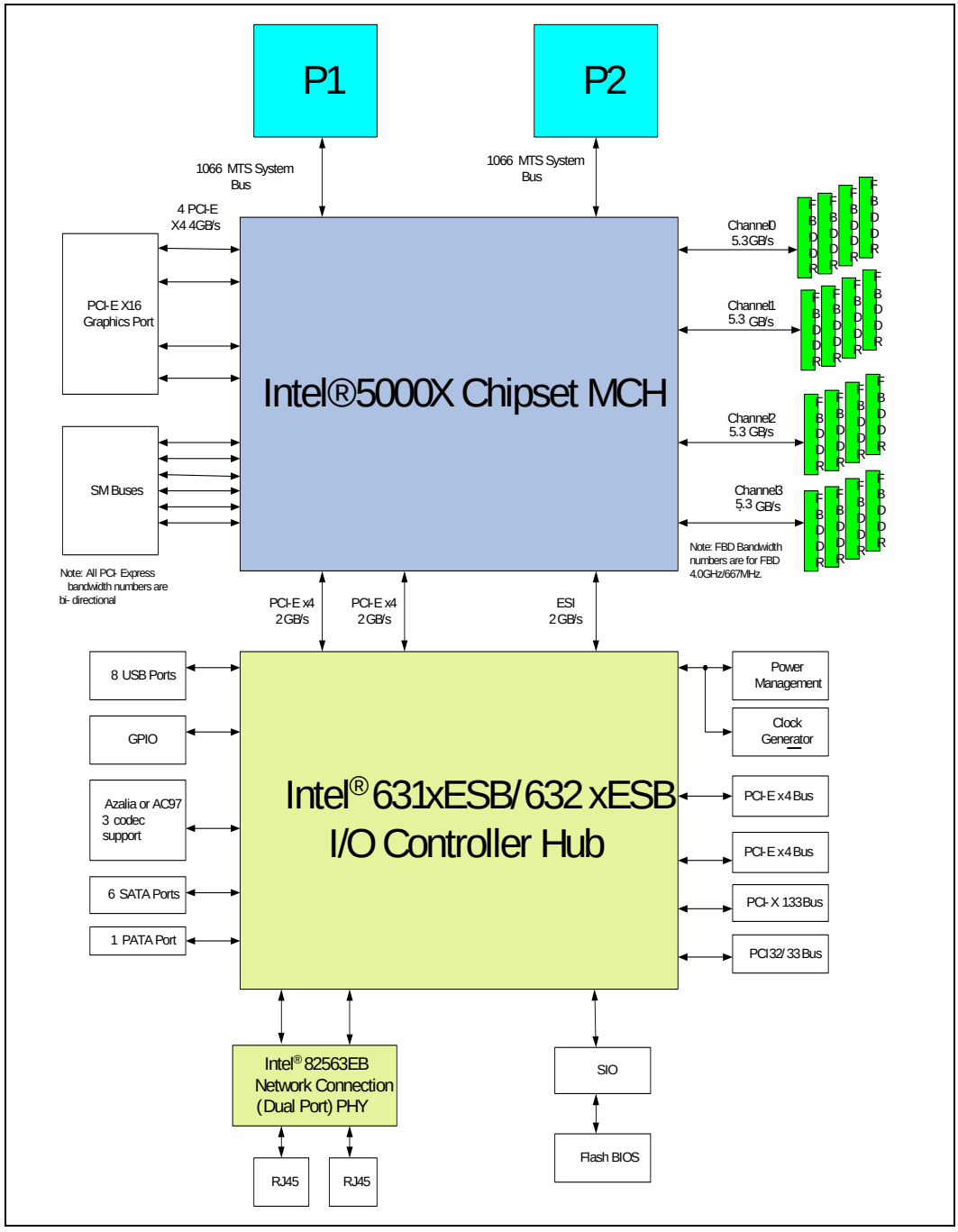
In a Intel 5000X Chipset-based platform, the MCH provides the processor interface, fully buffered DIMM memory interfaces, PCI Express bus interfaces, ESI interface, and SM Bus interfaces.

The MCH provides four channels of Fully Buffered DIMM (FB-DIMM) memory. Each channel can support up to 4 Dual Ranked FB-DIMM DDR2 DIMMs. FB-DIMM memory channels are organized in to two branches for support of RAID 1 (mirroring). The MCH can support up to 16 DIMMs or a maximum memory size of 64 GB physical memory in non-mirrored mode and 32 GB physical memory in mirrored configuration. The read bandwidth for each FB-DIMM channel is 4.25 GB/s for DDR2 533 FB-DIMM memory which gives a total read bandwidth of 17GB/s for four FB-DIMM channels. Thus this provides 8.5 GB/s of write memory bandwith for four FB-DIMM channels. The read bandwidth for each FB-DIMM channel is 5.325 GB/s for DDR2 667FB-DIMM memory which gives a total read bandwidth of 21.3 GB/s for four FB-DIMM channels. Thus this provides 10.7GB/s of write memory bandwith for four FB-DIMM channels. The total bandwidth is based on read bandwidth therefore the total bandwidth is 17 GB/s for 533 and 21.3 GB/s for 667.

The Intel® 631xESB/632xESB I/O Controller Hub integrates an Ultra ATA 100 controller, six Serial ATA host controller ports, one EHCI host controller, and four UHCI host controllers supporting eight external USB 2.0 ports, LPC interface controller, flash BIOS interface controller, PCI interface controller, Azalia / AC'97 digital controller, integrated LAN controller, an ASF controller and a ESI for communication with the MCH. The Intel 631xESB/632xESB I/O Controller Hub component provides the data buffering and interface arbitration required to ensure that system interfaces operate efficiently and provide the bandwidth necessary to enable the system to obtain peak performance. The Intel 631xESB/632xESB I/O Controller Hub elevates Serial ATA storage performance to the next level with Intel® RAID.

The ACPI compliant Intel 631xESB/632xESB I/O Controller Hub platform can support the Full-on, Stop Grant, Suspend to RAM, Suspend to Disk, and Soft-Off power management states. Through the use of the integrated LAN functions, the Intel® 631xESB/632xESB I/O Controller Hub also supports Alert Standard Format for remote management.

Figure 1-1. Intel® 5000X Chipset System Block Diagram



§

2 Signal Description

This section provides a detailed description of MCH signals. The signals are arranged in functional groups according to their associated interface. The signals presented in this section may not be present in all Intel 5000 Sequence chipsets. To determine if a signal is in a particular version, consult Section 8. Throughout this section the following conventions are used:

The terms *assertion* and *deassertion* are to avoid confusion when working with a mix of active-high and active-low signals. The terms *assert*, or *assertion*, indicates that the signal is active, independent of whether the active level is represented by a high or low voltage. The terms *deassert*, or *deassertion*, indicates that the signal is inactive.

Signal names may or may not have a "#" appended to them. The "#" symbol at the end of a signal name indicates that the active, or asserted state occurs when the signal is at a low voltage level. When "#" is not present after the signal name, the signal is asserted when at the high voltage level.

Differential signal pairs adopt a "{P/N}" suffix to indicate the "positive" (P) or "negative" (N) signal in the pair. If a "#" is appended, it is appended to the positive and negative signals in a pair.

Typical frequencies of operation for the fastest operating modes are indicated. No frequency is specified for asynchronous or analog signals.

Some signals or groups of signals have multiple versions. These signal groups may represent distinct but similar ports or interfaces, or may represent identical copies of the signal used to reduce loading effects.

Curly-bracketed non-trailing numerical indices, for example, "{X/Y}", represent replications of major buses. Square-bracketed numerical indices, for example, "[n:m]" represent functionally similar but logically distinct bus signals; each signal provides an independent control, and may or may not be asserted at the same time as the other signals in the grouping. In contrast, trailing curly-bracketed numerical indices, for example, "{x/y}" typically represent identical duplicates of a signal; such duplicates are provided for electrical reasons.

The following notations are used to describe the signal type:

I Input pin

O Output pin

I/O Bi-directional Input/Output pin

s/t/s Sustained Tri-state. This pin is driven to its inactive state prior to tri-stating.

The signal description also includes the type of buffer used for the particular signal:

AGTL+ Open Drain AGTL+ interface signal. Refer to the AGTL+ I/O Specification for complete details. The MCH integrates AGTL+ termination resistors, and supports VTT from 1.15 V to 1.55 V.

LVTTL Low Voltage TTL 3.3 V compatible signals

SSTL_2 Stub Series Terminated Logic 2.6 V compatible signals

2.6 VGPIO 2.6 V buffers used for miscellaneous GPIO signals

CMOS CMOS buffers

Host Interface signals that perform multiple transfers per clock cycle may be marked as either “4X” (for signals that are “quad-pumped”) or 2X (for signals that are “double-pumped”).

Note: Processor address and data bus signals are logically inverted signals. In other words, the actual values are inverted of what appears on the processor bus. This must be taken into account and the addresses and data bus signals must be inverted inside the MCH host bridge. All processor control signals follow normal convention. A 0 indicates an active level (low voltage) if the signal is followed by # symbol and a 1 indicates an active level (high voltage) if the signal has no # suffix.

Table 2-1. Signal Naming Conventions

Convention	Expands to
RR{0/1/2}XX	Expands to: RR0XX, RR1XX, and RR2XX. This denotes similar signals on replicated buses.
RR[2:0]	Expands to: RR[2], RR[1], and RR[0]. This denotes a bus.
RR{0/1/2}	Expands to: RR2, RR1, and RR0. This denotes electrical duplicates.
RR# or RR[2:0]#	Denotes an active low signal or bus.

Table 2-2 lists the reference terminology used for signal types.

Table 2-2. Buffer Signal Types

Buffer Direction	Description
I	Input signal
O	Output signal
A	Analog
I/O	Bidirectional (input/output) signal

2.1 Processor Front Side Bus Signals

2.1.1 Processor Front Side Bus 0

Signal Name	Type	Description
FSB0A[35:3]#	I/O	**Processor 0 Address Bus:** FSB0A[35:3]# connect to the processor address bus. During processor cycles, FSB0A[35:3]# are inputs. The MCH drives FSB0A[35:3]# during snoop cycles on behalf of ESI and AGP/Secondary PCI initiators. FSB0A[35:3]# are transferred at 2X rate. Note that the address is inverted on the processor bus. The MCH drives the FSB0A7# signal, which is then sampled by the processor and the MCH on the active-to-inactive transition of FSB0RESET#. The minimum setup time for this signal is 4 FSB0CLKs. The minimum hold time is 2 clocks and the maximum hold time is 20 FSB0CLKs.
FSB0ADS#	I/O	**Processor 0 Address Strobe:** The processor bus owner asserts FSB0ADS# to indicate the first of two cycles of a request phase. The MCH can assert this signal for snoop cycles and interrupt messages.
FSB0ADSTB[1:0]#	I/O	**Processor 0 Address Strobe:** FSB0ADSTB[1:0]# are source synchronous strobes used to transfer FSB0A[35:3]# and FSB0REQ[4:0]# at the 2X transfer rate. **StrobeAddress Bits** FSB0ADSTB0#FSB0A[16:3]#, FSB0REQ[4:0]# FSB0ADSTB1#FSB0A[35:17]#
FSB0AP[1:0]#	I/O	**Processor 0 Address Parity:** FSB0AP[1:0]# provide parity protection on the address bus
FSB0BINIT#	I/O	**Processor 0 Bus Initialization:** This signal causes a reset of the bus state machines.
FSB0BNR#	I/O	**Processor 0 Block Next Request:** This signal is used to block the current request bus owner from issuing a new request. This signal is used to dynamically control the processor bus pipeline depth.
FSB0BPM[5:4]	I/O	**Breakpoint /Debug Bus:** These signals are breakpoint and performance monitor signals. These are output from the processor to indicate the status of breakpoints and programmable counters used for monitoring processor performance.
FSB0BPRI#	O	**Processor 0 Priority Agent Bus Request:** The MCH is the only Priority Agent on the processor bus. It asserts this signal to obtain ownership of the address bus. This signal has priority over symmetric bus requests and cause the current symmetric owner to stop issuing new transactions unless the FSB0LOCK# signal was asserted.
FSB0BREQ[1:0]#	I/O	**Processor 0 Bus Requests**: The MCH pulls the FSB0BREQ0# signal low during RESET#. The signal is sampled by the processor on the active-to-inactive transition of FSB0RESET#. The minimum setup time for this signal is 4 FSB0CLKs. The minimum hold time is 2 clocks and the maximum hold time is 20 FSB0CLKs.
FSB0D[63:0]#	I/O	**Processor 0 Data Bus:** These signals are connected to the processor data bus. Data on FSB0D[63:0]# is transferred at a 4X rate. Note that the data signals may be inverted on the processor bus, depending on the **P0DBI**[3:0] signals.
FSB0DBI[3:0]#	I/O	**Processor 0 Dynamic Bus Inversion:** These signals are driven along with the FSB0D[63:0]# signals. They indicate if the associated signals are inverted. FSB0DBI[3:0]# are asserted such that the number of data bits driven electrically low (low voFSB0ltage) within the corresponding 16-bit group never exceeds 8. **FSB0DBI[x]#Data Bits** FSB0DBI3#FSB0D[63:48]# FSB0DBI2#FSB0D[47:32]# FSB0DBI1#FSB0D[31:16]# FSB0DBI0#FSB0D[15:0]#
FSB0DBSY#	I/O	**Processor 0 Data Bus Busy:** This signal is used by the data bus owner to hold the data bus for transfers requiring more than one cycle.
FSB0DEFER#	O	**Processor 0 Data Bus Defer:** Defer indicates that the MCH will terminate the transaction currently being snooped with either a deferred response or with a retry response.

Signal Name	Type	Description
FSB0DP[3:0]#	I/O	**Processor 0 Data Bus Parity:** FSB0DP[3:0]# provide parity protection on the data bus.
FSB0DRDY#	I/O	**Processor 0 Data Ready:** This signal is asserted for each cycle that data is transferred.
FSB0DSTBP[3:0]# FSB0DSTBN[3:0]#	I/O	**Processor 0 Differential Host Data Strobes:** The differential source synchronous strobes used to transfer FSB0D[63:0]# and FSB0DBI[3:0]# at the 4X transfer rate. **StrobeData Bits** FSB0DSTBP3#, FSB0DSTBN3#FSB0D[63:48]#, FSB0DBI3# FSB0DSTBP2#, FSB0DSTBN2#FSB0D[47:32]#, FSB0DBI2# FSB0DSTBP1#, FSB0DSTBN1#FSB0D[31:16]#, FSB0DBI1# FSB0DSTBP0#, FSB0DSTBN0#FSB0D[15:0]#, FSB0DBI0#
FSB0HIT#	I/O	**Processor 0 Cache Hit:** This signal indicates that a caching agent holds an unmodified version of the requested line. FSB0HIT# is also driven in conjunction with FSB0HITM# by the target to extend the snoop window.
FSB0HITM#	I/O	**Processor 0 Cache Hit Modified:** This signal indicates that a caching agent holds a modified version of the requested line and that this agent assumes responsibility for providing the line. FSB0HITM# is also driven in conjunction with FSB0HIT# to extend the snoop window.
FSB0LOCK#	I/O	**Processor 0 Lock:** This signal indicates to the system that a transaction must occur atomically. This signal must connect the appropriate pins of all processor FSB agents. For a locked sequence of transactions, LOCK# is asserted from the beginning of the first transaction to the end of the last transaction. When the priority agent asserts BPRI# to arbitrate for ownership of the processor FSB, it will wait until it observes LOCK# deasserted. This enables symmetric agents to retain ownership of the processor FSB throughout the bus locked operation and ensure the atomicity of lock.
FSB0MCERR#	I/O	**Processor 0 Machine Check Error:** Machine check error
FSB0REQ[4:0]#	I/O	**Processor Bus 0 Request Command:** These signals define the attributes of the request. FSB0REQ[4:0]# are transferred at 2X rate. They are asserted by the requesting agent during both halves of request phase. In the first half the signals define the transaction type to a level of detail that is sufficient to begin a snoop request. In the second half the signals carry additional information to define the complete transaction type.
FSB0RESET#	O	**Processor 0 Reset:** The FSB0RESET# pin is an output from the MCH. The MCH asserts FSB0RESET# while RSTIN# (PCIRST# from Intel® 631xESB/ 632xESB I/O Controller Hub) is asserted and for approximately 1 ms after RSTIN# is deasserted. The FSB0RESET# allows the processors to begin execution in a known state.
FSB0RS[2:0]#	O	**Processor 0 Response Status Signals:** These signals indicate the type of response according to the following: **Encoding Response Type** 000 Idle state 001 Retry response 010 Deferred response 011 Reserved (not driven by MCH) 100 Hard Failure (not driven by MCH) 101 No data response 110 Implicit Writeback 111 Normal data response
FSB0RSP#	O	**Processor 0 Response Status Parity:**
FSB0TRDY#	O	**Processor Bus 0 Target Ready:** This signal indicates that the target of the processor transaction is able to enter the data transfer phase.
FSB0VREF	Analog	**Processor 0 Voltage Reference:** Processor 0 voltage reference.

2.1.2 Processor Front Side Bus 1

Signal Name	Type	Description
FSB1A[35:3]#	I/O	**Processor 1 Address Bus:** FSB1A[35:3]# connect to the processor address bus. During processor cycles, FSB1A[35:3]# are inputs. The MCH drives FSB1A[35:3]# during snoop cycles on behalf of ESI and AGP/Secondary PCI initiators. FSB1A[35:3]# are transferred at 2X rate. Note that the address is inverted on the processor bus. ***Note:*** The MCH drives the FSB1A7# signal, which is then sampled by the processor and the MCH on the active-to-inactive transition of FSB1RESET#. The minimum setup time for this signal is 4 FSB0CLKs. The minimum hold time is 2 clocks and the maximum hold time is 20 FSB1CLKs.
FSB1ADS#	I/O	**Processor 1 Address Strobe:** The processor bus owner asserts FSB1ADS# to indicate the first of two cycles of a request phase. The MCH can assert this signal for snoop cycles and interrupt messages.
FSB1ADSTB[1:0]#	I/O	**Processor 1 Address Strobe:** FSB1ADSTB[1:0]# are source synchronous strobes used to transfer FSB1A[35:3]# and FSB1REQ[4:0]# at the 2X transfer rate. **StrobeAddress Bits** FSB1ADSTB0#FSB1A[16:3]#, FSB1REQ[4:0]# FSB1ADSTB1#FSB1A[35:17]#
FSB1AP[1:0]#	I/O	**Processor 1 Address Parity:** FSB0AP[1:0]# provide parity protection on the address bus
FSB1BINIT#	I/O	**Processor 1 Bus Initialization:** This signal causes a reset of the bus state machines.
FSB1BNR#	I/O	**Processor 1 Block Next Request:** This signal is used to block the current request bus owner from issuing a new request. This signal is used to dynamically control the processor bus pipeline depth.
FSB1BPM[5:4]	I/O	**Breakpoint /Debug Bus:** These signals are breakpoint and performance monitor signals. These are output from the processor to indicate the status of breakpoints and programmable counters used for monitoring processor performance.
FSB1BPRI#	O	**Processor 1 Priority Agent Bus Request:** The MCH is the only Priority Agent on the processor bus. It asserts this signal to obtain ownership of the address bus. This signal has priority over symmetric bus requests and cause the current symmetric owner to stop issuing new transactions unless the FSB1LOCK# signal was asserted.
FSB1BREQ[1:0]#	I/O	**Processor 1 Bus Requests:** The MCH pulls the FSB1BREQ1# & FSB1BREQ0# signals low during RESET#. The signal is sampled by the processor on the active-to-inactive transition of FSB1RESET#. The minimum setup time for this signal is 4 FSB1CLKs. The minimum hold time is 2 clocks and the maximum hold time is 20 FSB1CLKs.
FSB1D[63:0]#	I/O	**Processor 1 Data Bus:** These signals are connected to the processor data bus. Data on FSB1D[63:0]# is transferred at a 4X rate. Note that the data signals may be inverted on the processor bus, depending on the FSB1DBI[3:0] signals.
FSB1DBI[3:0]#	I/O	**Processor 1 Dynamic Bus Inversion:** These signals are driven along with the FSB1D[63:0]# signals. They indicate if the associated signals are inverted. FSB1DBI[3:0]# are asserted such that the number of data bits driven electrically low (low voltage) within the corresponding 16-bit group never exceeds 8. **FSB1DBI[x]#Data Bits** FSB1DBI3#FSB1D[63:48]# FSB1DBI2#FSB1D[47:32]# FSB1DBI1#FSB1D[31:16]# FSB1DBI0#FSB1D[15:0]#
FSB1DBSY#	I/O	**Processor 1 Data Bus Busy:** This signal is used by the data bus owner to hold the data bus for transfers requiring more than one cycle.
FSB1DEFER#	O	**Processor 1 Data Bus Defer:** Defer indicates that the MCH will terminate the transaction currently being snooped with either a deferred response or with a retry response.

Signal Name	Type	Description
FSB1DP[3:0]#	I/O	**Processor 1 Data Bus Parity:** FSB1DP[3:0]# provide parity protection on the data bus.
FSB1DRDY#	I/O	**Processor 1 Data Ready:** This signal is asserted for each cycle that data is transferred.
FSB1DSTBP[3:0]# FSB1DSTBN[3:0]#	I/O	**Processor 1 Differential Host Data Strobes:** The differential source synchronous strobes used to transfer FSB1D[63:0]# and FSB1DBI[3:0]# at the 4X transfer rate. **StrobeData Bits** FSB1DSTBP3#, FSB10DSTBN3#FSB1D[63:48]#, FSB1DBI3# FSB1DSTBP2#, FSB1DSTBN2#FSB1D[47:32]#, FSB1DBI2# FSB1DSTBP1#, FSB1DSTBN1#FSB1D[31:16]#, FSB1DBI1# FSB1DSTBP0#, FSB1DSTBN0#FSB1D[15:0]#, FSB1DBI0#
FSB1HIT#	I/O	**Processor 1 Cache Hit:** This signal indicates that a caching agent holds an unmodified version of the requested line. FSB1HIT# is also driven in conjunction with FSB1HITM# by the target to extend the snoop window.
FSB1HITM#	I/O	**Processor 1 Cache Hit Modified:** This signal indicates that a caching agent holds a modified version of the requested line and that this agent assumes responsibility for providing the line. FSB1HITM# is also driven in conjunction with FSB1HIT# to extend the snoop window.
FSB1LOCK#	I/O	**Processor 1 Lock:** This signal indicates to the system that a transaction must occur atomically. This signal must connect the appropriate pins of all processor FSB agents. For a locked sequence of transactions, LOCK# is asserted from the beginning of the first transaction to the end of the last transaction. When the priority agent asserts BPRI# to arbitrate for ownership of the processor FSB, it will wait until it observes LOCK# deasserted. This enables symmetric agents to retain ownership of the processor FSB throughout the bus locked operation and ensure the atomicity of lock.
FSB1MCERR#	I/O	**Processor 1 Machine Check Error:** Machine check error
FSB1REQ[4:0]#	I/O	**Processor Bus 1 Request Command:** These signals define the attributes of the request. FSB1REQ[4:0]# are transferred at 2X rate. They are asserted by the requesting agent during both halves of request phase. In the first half the signals define the transaction type to a level of detail that is sufficient to begin a snoop request. In the second half the signals carry additional information to define the complete transaction type.
FSB1RESET#	O	**Processor 1 Reset:** The FSB1RESET# pin is an output from the MCH. The MCH asserts FSB1RESET# while RSTIN# (PCIRST# from Intel® 631xESB/ 632xESB I/O Controller Hub) is asserted and for approximately 1 ms after RSTIN# is deasserted. The FSB1RESET# allows the processors to begin execution in a known state.
FSB1RS[2:0]#	O	**Processor 1 Response Status Signals:** These signals indicates type of response according to the following: **EncodingResponse Type** 000 Idle state 001 Retry response 010 Deferred response 011 Reserved (not driven by MCH) 100 Hard Failure (not driven by MCH) 101 No data response 110 Implicit Writeback 111 Normal data response
FSB1RSP#	O	Processor 1 Response Status Parity:
FSB1TRDY#	O	**Processor Bus 1 Target Ready:** This signal indicates that the target of the processor transaction is able to enter the data transfer phase.
FSB1VREF	Analog	**Processor 1 Voltage Reference:** Processor 1 voltage reference.

2.2 Fully Buffered DIMM Memory Channels

The following reference and compensation signals are common to all FB-DIMM channels.

Signal Name	Type	Description
FBDBGBIASEXT	Analog	FB-DIMM Bypass Bias Input for Band Gap Circuit:
FBDICOMPBIAS	Analog	FB-DIMM Transmitter Swing Bias:
FBDRESIN	Analog	FB-DIMM On-die Impedance Compensation:

2.2.1 FB-DIMM Branch 0

FB-DIMM branch 0 contains FB-DIMM channels 0 and 1. The following signals are common to both FB-DIMM channels.

Signal Name	Type	Description
FBD01CLKN	Analog	**FB-DIMM Clock Negative:** Core Clock Negative Phase
FBD01CLKP	Analog	**FB-DIMM Clock Positive:** Core Clock Positive Phase
FBD01VCCA	Analog	**FB-DIMM VCC:** Analog Voltage for the PLL
FBD01VSSA	Analog	**FB-DIMM VSS:** Analog Voltage for PLL

2.2.1.1 FB-DIMM Channel 0

Signal Name	Type	Description
FBD0NBIN[13:0]	I	FB-DIMM Channel 0 Northbound Input Data Negative Phase: NOTE: FBD0NBIN[13] is not an active signal carrying signal but must be connected to properly terminate the FB DIMM component.
FBD0NBIP[13:0]	I	FB-DIMM Channel 0 Northbound Input Data Positive Phase: NOTE: FBD0NBIP[13] is not an active signal carrying signal but must be connected to properly terminate the FB DIMM component.
FBD0SBON[9:0]	O	FB-DIMM Channel 0 Southbound Output Negative Phase:
FBD0SBOP[9:0]	O	FB-DIMM Channel 0 Southbound Output Positive Phase:

2.2.1.2 FB-DIMM Channel 1

Signal Name	Type	Description
FBD1NBIN[13:0]	I	FB-DIMM Channel 1 Northbound Input Data Negative Phase: NOTE: FBD1NBIN[13] is not an active signal carrying signal but must be connected to properly terminate the FB DIMM component.
FBD1NBIP[13:0]	I	FB-DIMM Channel 1 Northbound Input Data Positive Phase: NOTE: FBD1NBIP[13] is not an active signal carrying signal but must be connected to properly terminate the FB DIMM component.
FBD1SBON[9:0]	O	FB-DIMM Channel 1 Southbound Output Negative Phase:
FBD1SBOP[9:0]	O	FB-DIMM Channel 1 Southbound Output Positive Phase:

2.2.2 FB-DIMM Branch 1

FB-DIMM branch 1 contains FB-DIMM channels 2 and 3. The following signals are common to both FB-DIMM channels.

Signal Name	Type	Description
FBD23CLKN	Analog	**FB-DIMM Clock Negative:** Core Clock Negative Phase
FBD23CLKP	Analog	**FB-DIMM Clock Positive:** Core Clock Positive Phase
FBD23VCCA	Analog	**FB-DIMM VCC:** Analog Voltage for the PLL
FBD23VSSA	Analog	**FB-DIMM VSS:** Analog Voltage for PLL

2.2.2.1 FB-DIMM Channel 2

Signal Name	Type	Description
FBD2NBIN[13:0]	I	FB-DIMM Channel 2 Northbound Input Data Negative Phase: NOTE: FBD2NBIN[13] is not an active signal carrying signal but must be connected to properly terminate the FB DIMM component.
FBD2NBIP[13:0]	I	FB-DIMM Channel 2 Northbound Input Data Positive Phase: NOTE: FBD2NBIP[13] is not an active signal carrying signal but must be connected to properly terminate the FB DIMM component.
FBD2SBON[9:0]	O	FB-DIMM Channel 2 Southbound Output Negative Phase:
FBD2SBOP[9:0]	O	FB-DIMM Channel 2 Southbound Output Positive Phase:

2.2.2.2 FB-DIMM Channel 3

Signal Name	Type	Description
FBD3NBIN[13:0]	I	FB-DIMM Channel 3 Northbound Input Data Negative Phase: NOTE: FBD3NBIN[13] is not an active signal carrying signal but must be connected to properly terminate the FB DIMM component.
FBD3NBIP[13:0]	I	FB-DIMM Channel 3 Northbound Input Data Positive Phase: NOTE: FBD3NBIP[13] is not an active signal carrying signal but must be connected to properly terminate the FB DIMM component.
FBD3SBON[9:0]	O	FB-DIMM Channel 3 Southbound Output Negative Phase:
FBD3SBOP[9:0]	O	FB-DIMM Channel 3 Southbound Output Positive Phase:

2.3 PCI Express* Signal List

2.3.1 PCI Express* Common Signals

Signal Name	Type	Description
PECLKN	Analog	PCI Express* Common Clock Negative Phase:
PECLKP	Analog	PCI Express Common Clock Positive Phase:
PEICOMPI	Analog	PCI Express Impedance Compensation:
PEICOMPO	Analog	PCI Express Impedance Compensation:
PEVCCA	Analog	PCI Express VCC: Analog Voltage for the PCI Express PLL:
PEVCCBG	Analog	PCI Express Band Gap VCC: Band Gap Voltage
PEVSSA	Analog	PCI Express VSS: Analog Voltage for PCI Express PLL:
PEVSSBG	Analog	PCI Express Band Gap VSS: Band Gap Voltage
PEWIDTH[3:0]	Power/ Other	PCI Express Port Width Strapping Pins:

2.3.2 PCI Express Port 0, Enterprise South Bridge Interface (ESI)

PCI Express port 0 is a x4 port dedicated to providing the ESI link between the Intel 5000X Chipset MCH and the Intel 631xESB/632xESB I/O Controller Hub.

Signal Name	Type	Description Reference
PE0RP[3:0]	I	**PCI Express Port 0 (ESI) Positive Phase Inbound:** (Receive) Signals
PE0RN[3:0]	I	**PCI Express Port 0 (ESI) Negative Phase Inbound:** (Receive) Signals
PE0TP[3:0]	O	**PCI Express Port 0 (ESI) Positive Phase Outbound:** (Transmit) Signals
PE0TN[3:0]	O	**PCI Express Port 0 (ESI) Negative Phase Outbound:** (Transmit) Signals

2.3.3 PCI Express Port 2

PCI Express port 2 is a x4 port. PCI Express port 2 is combinable with PCI Express port 3 to form a single PCI Express x8 port. Normally port 2 and port 3 are used to increase the bandwidth between the Intel 5000X Chipset MCH and the Intel 631xESB/632xESB I/O Controller Hub.

Signal Name	Type	Description
PE2RP[3:0]	I	**PCI Express Port 2 Positive Phase Inbound:** (Receive) Signals
PE2RN[3:0]	I	**PCI Express Port 2 Negative Phase Inbound:** (Receive) Signals
PE2TP[3:0]	O	**PCI Express Port 2 Positive Phase Outbound:** (Transmit) Signals
PE2TN[3:0]	O	**PCI Express Port 2 Negative Phase Outbound:** (Transmit) Signals

2.3.4 PCI Express Port 3

PCI Express port 3 is combinable with PCI Express port 2 to form a single PCI Express x8 port. Normally port 2 and port 3 are used to increase the bandwidth between the Intel 5000X Chipset MCH and the Intel 631xESB/632xESB I/O Controller Hub.

Signal Name	Type	Description
PE3RP[3:0]	I	**PCI Express Port 3 Positive Phase:** Inbound (Receive) Signals
PE3RN[3:0]	I	**PCI Express Port 3 Negative Phase:** Inbound (Receive) Signals
PE3TP[3:0]	O	**PCI Express Port 3 Positive Phase:** Outbound (Transmit) Signals
PE3TN[3:0]	O	**PCI Express Port 3 Negative Phase:** Outbound (Transmit) Signals

2.3.5 PCI Express* Graphics Port

In the Intel® 5000X Chipset MCH PCI Express ports 4, 5, 6, and 7 are combined to form a single high performance x16 graphics port.

Signal Name	Type	Description
PE4RP[3:0]	I	PCI Express* Graphics Port First x4, Positive Phase Inbound (Receive) Signals:
PE4RN[3:0]	I	PCI Express Graphics Port First x4, Negative Phase Inbound (Receive) Signals:
PE4TP[3:0]	O	PCI Express Graphics Port First x4, Positive Phase Outbound (Transmit) Signals:
PE4TN[3:0]	O	PCI Express Graphics Port First x4, Negative Phase Outbound (Transmit) Signals:
PE5RP[3:0]	I	PCI Express Graphics Port Second x4, Positive Phase Inbound (Receive) Signals:
PE5RN[3:0]	I	PCI Express Graphics Port Second x4, Negative Phase Inbound (Receive) Signals:
PE5TP[3:0]	O	PCI Express Graphics Port Second x4, Positive Phase Outbound (Transmit) Signals:
PE5TN[3:0]	O	PCI Express Graphics Port Second x4, Negative Phase Outbound (Transmit) Signals:
PE6RP[3:0]	I	PCI Express Graphics Port Third x4, Positive Phase Inbound (Receive) Signals:
PE6RN[3:0]	I	PCI Express Graphics Port Third x4, Negative Phase Inbound (Receive) Signals:
PE6TP[3:0]	O	PCI Express Graphics Port Third x4, Positive Phase Outbound (Transmit) Signals:
PE6TN[3:0]	O	PCI Express Graphics Port Third x4, Negative Phase Outbound (Transmit) Signals:
PE7RP[3:0]	I	PCI Express Graphics Port Fourth x4, Positive Phase Inbound (Receive) Signals:
PE7RN[3:0]	I	PCI Express Graphics Port Fourth x4, Negative Phase Inbound (Receive) Signals:
PE7TP[3:0]	O	PCI Express Graphics Port Fourth x4, Positive Phase Outbound (Transmit) Signals:
PE7TN[3:0]	O	PCI Express Graphics Port Fourth x4, Negative Phase Outbound (Transmit) Signals:

2.4 System Management Bus Interfaces

There are seven SM Bus interfaces dedicated to specific functions. These functions are:

- System Management
- Four buses dedicated to FB-DIMM serial presents detect, one for each channel
- PCI Hot-Plug

Signal Name	Type	Description
CFGSMBCLK	I/O	**Slave SMB Clock:** System Management Bus Clock
CFGSMBDATA	I/O	**Slave SMB Data:** SMB Address/Data
GPIOSMBCLK	I/O	**PCI SMB Clock:** PCI Hot-Plug Master VPI, System Management Bus Clock
GPIOSMBDATA	I/O	**PCI SMB Data:** PCI Hot-Plug Master VPI, SMB Address/Data
SPD0SMBCLK	I/O	**FB-DIMM Channel 0 SMB Clock:** FB-DIMM Memory Serial Presents Detect 0, System Management Bus Clock
SPD0SMBDATA	I/O	**FB-DIMM Channel 0 SMB Data:** FB-DIMM Memory Serial Presents Detect 0, SMB Address/Data
SPD1SMBCLK	I/O	**FB-DIMM Channel 1 SMB Clock:** FB-DIMM Memory Serial Presents Detect 1, System Management Bus Clock
SPD1SMBDATA	I/O	**FB-DIMM Channel 1 SMB Data:** FB-DIMM Memory Serial Presents Detect 1, SMB Address/Data
SPD2SMBCLK	I/O	**FB-DIMM Channel 2 SMB Clock:** FB-DIMM Memory Serial Presents Detect 2, System Management Bus Clock
SPD2SMBDATA	I/O	**FB-DIMM Channel 2 SMB Data:** FB-DIMM Memory Serial Presents Detect 2, SMB Address/Data
SPD3SMBCLK	I/O	**FB-DIMM Channel 3 SMB Clock:** FB-DIMM Memory Serial Presents Detect 3, System Management Bus Clock
SPD3SMBDATA	I/O	**FB-DIMM Channel 3 SMB Data:** FB-DIMM Memory Serial Presents Detect 3, SMB Address/Data

2.5 XD Port Signal List

Signal Name	Type	Description
XDPCOMCRES	Analog	**XDP Bus Compensation:**
XDPD[15:0]#	I/O	**Data Bus:**.
XDPSTBN# XDPSTBP#	I/O	**Data Bus Strobe Negative and Positive Phases:**
XDPODTCRES	Analog	**XDP Bus Compensation:**
XDPRDY#	I/O	**Data Bus Ready:**
XDPSLWCRES	Analog	**XDP Bus Slew Rate Compensation:**.

2.6 JTAG Bus Signal List

Signal Name	Type	Description
TCK	I	**Clock:** Clock pin of the JTAG.
TDI	I	**Data Input:** Serial chain input of the JTAG.
TDO	O	**Data Output:** Serial chain output of the JTAG.
TMS	I	**State Machine:** JTAG State machine control
TRST#	I	**Reset:** Asynchronous reset of the JTAG.

2.7 Clocks, Reset and Miscellaneous

Signal Name	Type	Description
CORECLKN	Analog	**Differential Processor Core Clock Negative Phase:** These pins receive a low-voltage differential host clock from the external clock synthesizer. This clock is used by all of the MCH logic that is in the Host clock domain.
CORECLKP	Analog	**Differential Processor Core Clock Positive Phase:** These pins receive a low-voltage differential host clock from the external clock synthesizer. This clock is used by all of the MCH logic that is in the Host clock domain.
COREVCCA	Analog	**Core VCC:** Analog Voltage for the PLL
COREVSSA	Analog	**Core VSS:** Analog Voltage for PLL
ERR[2:0]#	O	**Error Output:** Error output signal: ERR[0] = Correctable and recoverable error from the memory subsystem ERR[1] = Uncorrectable error from the Intel 5000X Chipset MCH ERR[2] = Fatal error from the Intel 5000X Chipset MCH
FSBCRES	Analog	**Processor Bus Compensation**:
ODTCRES	Analog	**Processor Bus Compensation:**
FSBSLWCRES	Analog	**Processor Bus Slew Rate Compensation:**
FSBSLWCTRL	Power/ Other	**Processor Bus Slew Rate Control:**
FSBVCCA	Analog	**FSB VCC:** Analog Voltage for the FB-DIMM channel PLL
PWRGOOD	I	**Power OK:** When asserted, this signal indicates that all power supplies are in specification.
PSEL[2:0]	I	**Processor Speed Select:**
RESETI#	I	**MCH Reset:** This is the hard reset
RSVD	No Connect	**Reserved Pin:**
TDIOANODE	Analog	**Thermal Diode Anode:** This is the anode of the thermal diode
TDIOCATHODE	Analog	**Thermal Diode Cathode:** This is the cathode of the thermal diode
TESTHI	Power	**1.5 Volt Pullup:**
TESTHI_V3REF	Power/ Other	**3.3 Volt Pullup:**
VSSQUIET	Analog	**Quiet VSS:** Quiet VSS for ODDD
VSSSEN	Analog	**Quiet VSS:** Quiet VSS for Thermal Sensor
VCCSEN	Analog	**Quiet VCC:** Quiet VCC for Thermal Sensor

2.8 Power and Ground Signals

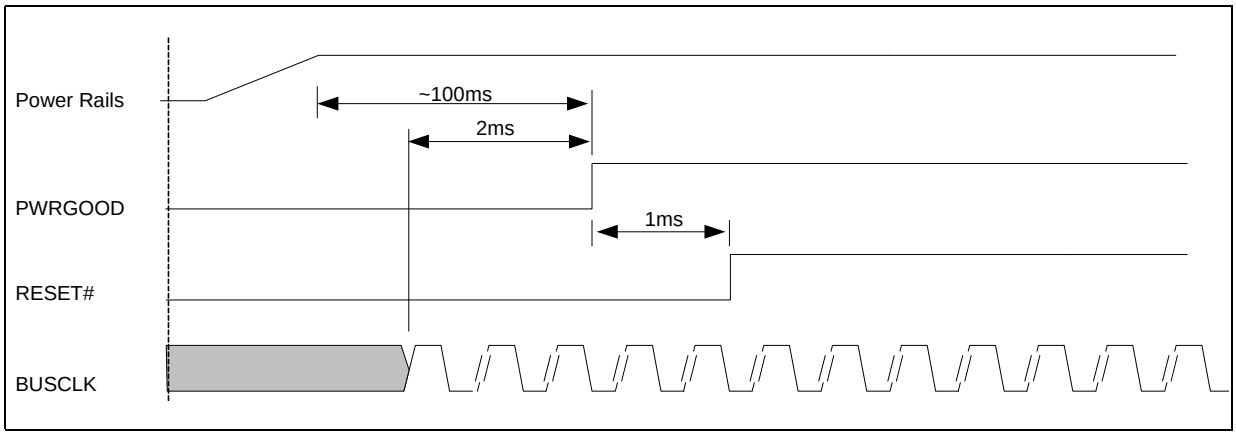
Signal Name	Description
V3REF	**SMB VCC:** Common 3.3 for SMB buses
VCC	**VCC Supply:** This is the 1.5 V core voltage.
VCCSF	**VCC Snoop Filter Supply:** This is the 1.5 V core voltage on a separate plane for the snoop filter.
VSS	**Ground Return:** Common return for power supplies
VTT	**VTT Supply:** VTT is a 1.2 V FSB supply
VCCFBD	**VCC for System Memory:** VCCFBD is 1.8 V for DDR2 power.
VCCPE	VCC for PCI Express ports.

2.9 MCH Sequencing Requirements

Power Plane and Sequencing Requirements:

- Clock Valid Timing:
- BUSCLK must be valid at least 2ms prior to rising edge of PWRGOOD.

Figure 2-1. Intel 5000X Chipset Clock and Reset Requirements



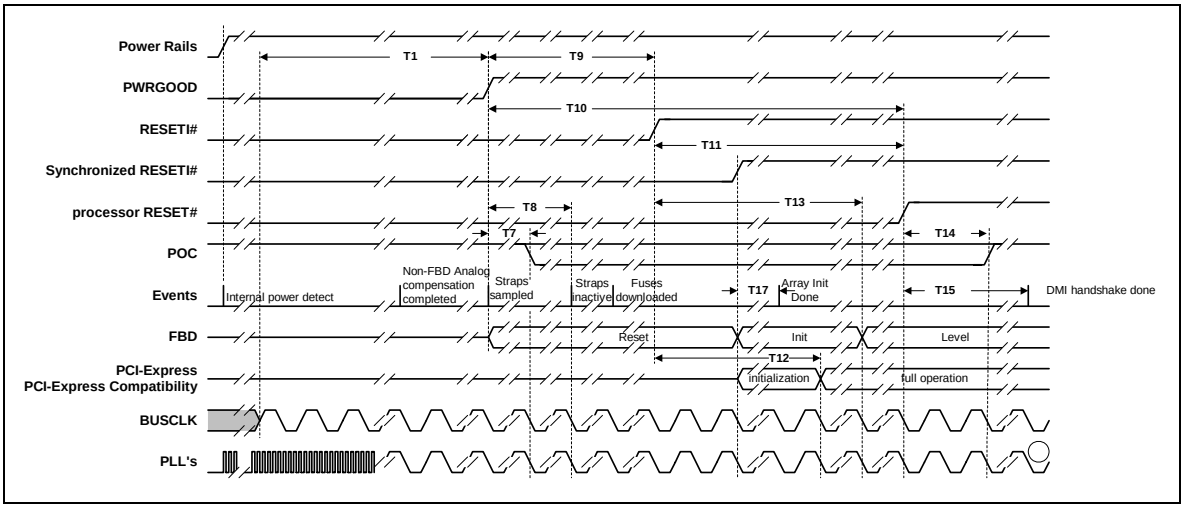
2.10 Reset Requirements

2.10.1 Timing Diagrams

2.10.1.1 Power-Up

The power-up sequence is illustrated in Figure 2-2.

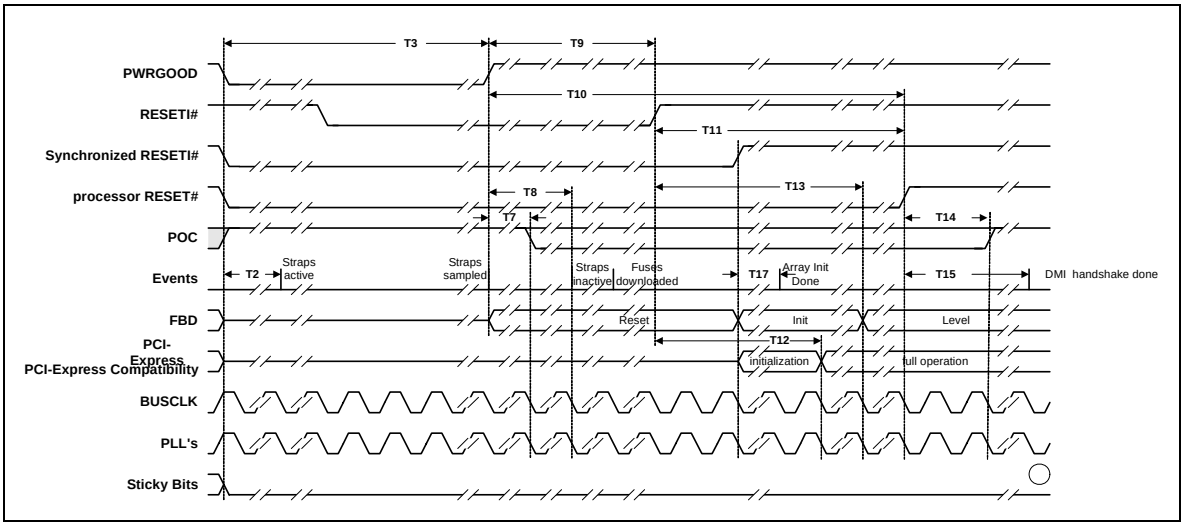
Figure 2-2. Power-Up



2.10.1.2 Power Good

The PWRGOOD reset sequence is illustrated in Figure 2-3.

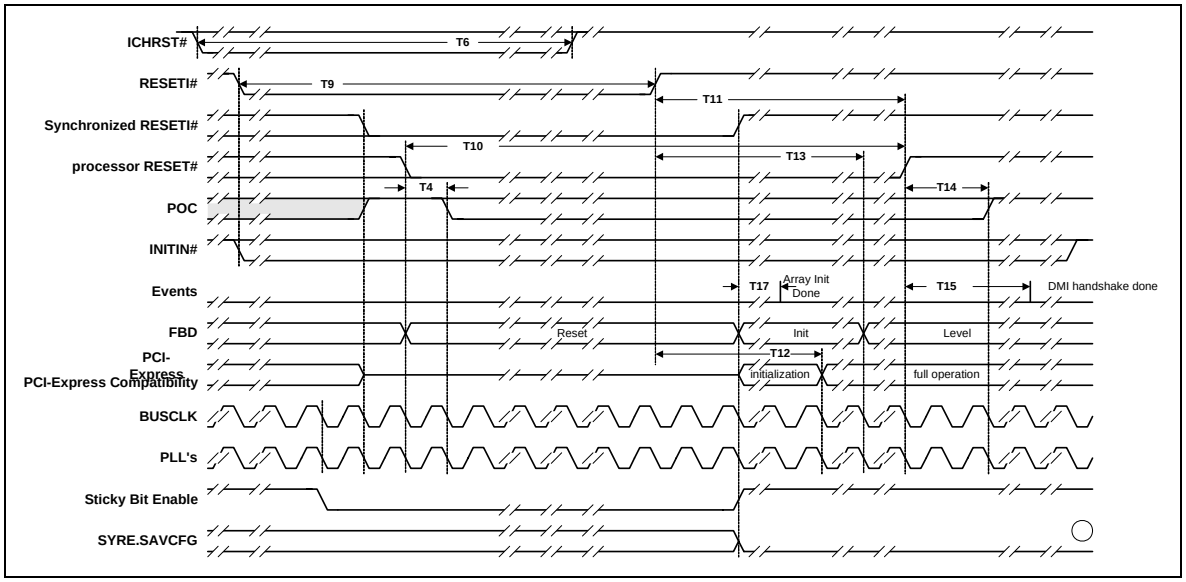
Figure 2-3. PWRGOOD



2.10.1.3 Hard Reset

The Hard Reset sequence is illustrated in Figure 2-4.

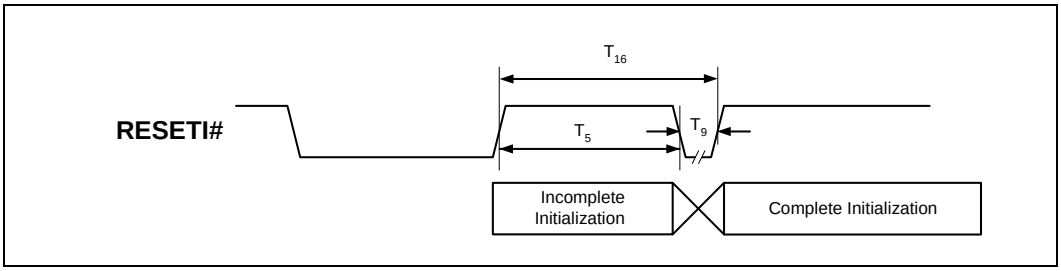
Figure 2-4. Hard Reset



2.10.1.4 RESETI# Retriggering Limitations

Figure 2-5 shows the timing for a RESETI# retrigger.

Figure 2-5. RESETI# Retriggering Limitations



2.10.2 Reset Timing Requirements

Table 2-3 specifies the timings drawn in Figure 2-2, Figure 2-3, Figure 2-4, and Figure 2-5. Nominal clock frequencies are described. Specifications still hold for derated clock frequencies.

Table 2-3. Power Up and Hard Reset Timings

Timing	Description	Min	Max	Comments
T1	Power and master clocks stable to PWRGOOD signal assertion	2ms		3GIO PLL specification
T2	PWRGOOD de-assertion to straps active		40ns	
T3	PWRGOOD de-assertion	80ns		Minimum PWRGOOD de-assertion time while power and platform clocks are stable.
T4	POC after RESET# assertion delay	1 BUSCLK		
T5	Platform reset de-assertion to platform reset assertion	50 BUSCLK's		Minimum re-trigger time on RESETI# de-assertion.
T7	PWRGOOD assertion to POC active	2 BUSCLK's		POC turn-on delay after strap disable
T8	PWRGOOD assertion to straps inactive	12ns	18ns	Strap Hold Time
T9	RESETI# signal assertion during PWRGOOD / PWROK signal assertion	1ms		This delay can be provided by the ICH6 or by system logic
T10	RESET# assertion during processor PWRGOOD assertion	1ms	10ms	Processor EMTS specification.
T11	RESETI# signal de-assertion to processor RESET# signal de-assertion	480us[a]		**Note**: This is a special Woodcrest requirement to have a longer POC assertion setup time on the FSB and the Intel® 5000X Chipset has added a fix in B0 RTL to increase this time period from 160us to 480us.
T12	RESETI# signal de-assertion to completion of PCI-Express initialization sequence		1,250,000 PECLK's	PCI-Express clock is 100MHz
T13	Array Initialization duration		200 cycles	
T14	POC hold time after RESET# de-assertion	2 BUSCLK's	19 BUSCLK's	Processor EMTS specification
T15	Initiation of DMI reset sequence to processor RESET# signal de-assertion		10,000 PECLK's + T17	ICH6 specification
T16	RESETI# re-trigger delay	T5 + T9		
T17	CPU_RESET_DONE capture timer	2,000 BUSCLK's		

Notes:

a. In the Intel® 5000X Chipset B0 RTL, the T11 duration is implemented through a counter with max value of 162,000 core clocks. For 333 Mhz, this gives a period of 486us for the POC setup time while @266 Mhz, the period is 607.5 us.

Table 2-4 summarizes the Product Name Initialization timings.

Table 2-4. Critical Intel® 5000X Initialization Timings

Sequence	Started by	Maximum Length	Covered by Timing parameter
Intel® 5000X Chipset Core, FSB, FB-DIMM PLL lock	Stable power and master clock	666,667 333 MHz cycles	T_1
Intel® 5000X Chipset MCH PCI Express PLL lock	Stable power and master clock	200,000 100 MHz cycles	
Array initialization	Synchronized RESETI# Deassertion	200 cycles	T_{17}
Fuse download	PWRGOOD Assertion	333,333 333 MHz cycles	T_9

2.10.3 Miscellaneous Requirements and Limitations

- Power rails and stable BUSCLK, FBD{0/1}CLK, and PECLK master clocks remain within specifications through all but power-up reset.
- Frequencies (for example, 266 MHz) described in this chapter are nominal. The Intel 5000X Chipset MCH reset sequences must work for the frequency of operation range specified in the Clocking chapter.
- Hard Reset can be initiated by code running on a processor, JTAG, SMBus, or PCI agents.
- Hard Reset is not guaranteed to correct all illegal configurations or malfunctions. Software can configure sticky bits in the Intel 5000X Chipset MCH to disable interfaces that will not be accessible after Hard Reset. Signaling errors or protocol violations prior to reset (from processor bus, FB-DIMM, or PCI Express) may hang interfaces that are not cleared by Hard Reset.
- System activity is initiated by a request from a processor bus. No I/O devices will initiate requests until configured by a processor to do so.
- The FB-DIMM channels will be enabled for packet levelization (Intel 5000X Chipset MCH.FBDST.STATE=“Ready” or “RecoveryReady” state) upon completion of a hard reset. Software should inspect the Intel 5000X Chipset MCH.FBDST.STATE configuration bits to determine which FB-DIMM channels are available.
- The default values of the POC configuration register bits do not require any processor request signals to be asserted when PWRGOOD is first asserted. Software sets these configuration registers to define these values, then initiates a hard reset that causes them to be driven during processor RESET# signal assertion.
- Cleanly aborting an in-progress SPD command during a PWRGOOD deassertion is problematic. No guarantee can be issued as to the final state of the EEPROM in this situation. The Intel 5000X Chipset MCH cannot meet the SPD data $t_{SU,STO}$ timing specification. Since the Intel 5000X Chipset MCH floats the data output into a pull-up on the platform, a read will not degrade to a write. However, if the PWRGOOD deassertion occurs after the EEPROM has received the write bit, the data will be corrupted. The platform pull-up must be strong enough to complete a low-to-high transition on the clock signal within t_R = 1 microsecond (ATMEL AT24C01 timing specification) after deassertion of PWRGOOD to prevent clock glitches. Within these constraints, an in-progress write address will not be corrupted.

2.11 Intel 5000X Chipset Platform Signal Routing Topology Diagrams

Figure 2-6. Simplest Power Good Distribution

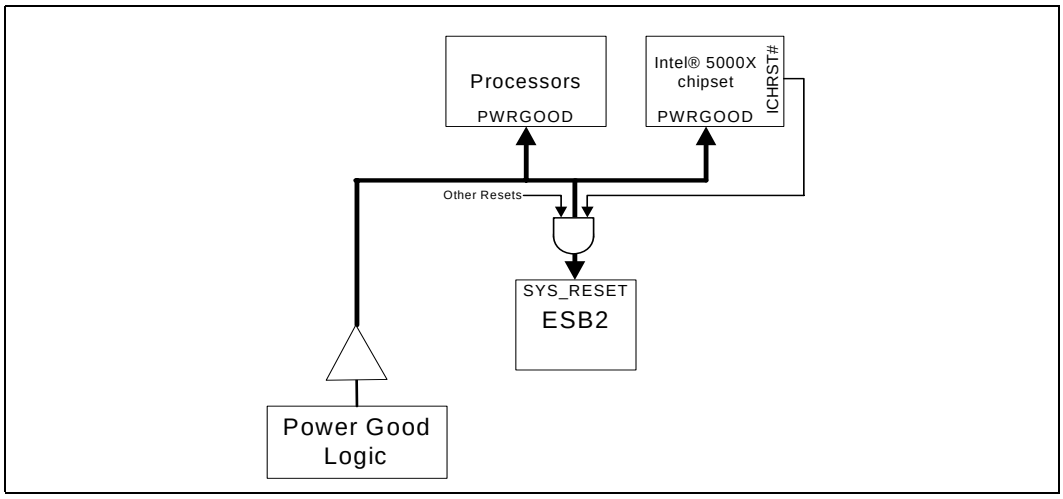


Figure 2-7. Basic System Reset Distribution

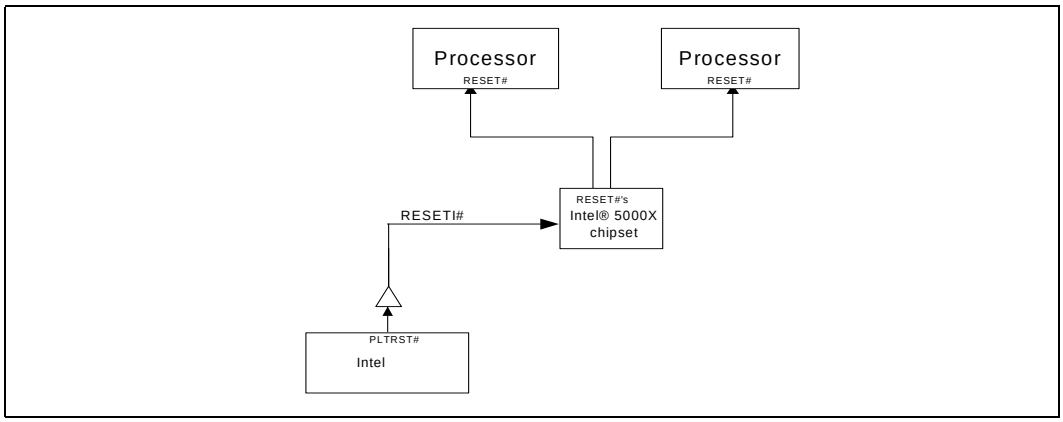
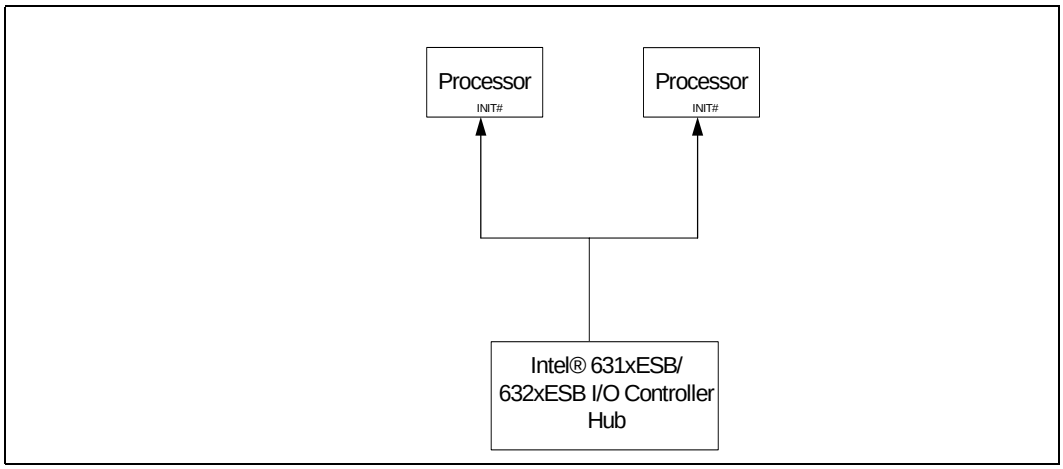


Figure 2-8. Basic INIT# Distribution



2.11.1 Intel 5000X Customer Reference Platform (SRP) Reset Topology

Typical platform level reset implementation is described in the *Dual-Core Intel® Xeon® Processor 5000 series (1066 MHz) and Intel® 5000 Sequence Chipsets Platform Design Guide* (PDG).

2.12 Signals Used as Straps

2.12.1 Functional Straps

The PEWIDTH signals are used to determine the widths of the 7 PCI Express ports.

Signal Name	Type	Description
PEWIDTH[3:0]	Power/ Other	PCI Express Port Width Strapping Pins:

§

3 Register Description

The Intel 5000X Chipset MCH contains three sets of software accessible registers, accessed via the host processor I/O address space:

- Control registers I/O mapped into the processor I/O space that controls access to PCI and AGP configuration spaces.
- Internal configuration registers residing within the MCH are partitioned into logical device register sets (“logical” since they reside within a single physical device). The first register set is dedicated to MCH functionality (controls PCI bus 0, that is, DRAM configuration, other chipset operating parameters, and optional features). The second register block is dedicated to host-AGP bridge functions (controls AGP interface configurations and operating parameters). The third register set is dedicated to ESI control.

The MCH supports PCI configuration space accesses using the mechanism denoted as Configuration Mechanism 1 in the PCI specification as defined in the *PCI Local Bus Specification,* Revision 2.3. All the registers are organized by bus, device, function, and so forth, as defined in the *PCI Express Base Specification*, Revision 1.0a. The MCH supports registers in PCI Express extended space. All MCH registers in Intel 5000X Chipset appear on PCI Bus #0.

In addition, the MCH registers can be accessed by a memory mapped register access mechanism (as MMIO), a PCI configuration access mechanism (only PCI space registers), and register access mechanisms through JTAG and SMBus. The memory mapped access mechanism is further broken down into different ranges. The internal registers of this chip set can be accessed in 8-bit, 16-bit, or 32-bit quantities, with the exception of CFGADR which can only be accessed as a 32-bit. All multi-byte numeric fields use “little-endian” ordering (that is, lower addresses contain the least significant parts of the field).

In addition, the MCH can forward accesses to all PCI/PCI Express configuration registers south of the MCH through the same mechanisms.

3.1 Register Terminology

Registers and register bits are assigned one or more of the following attributes. These attributes define the behavior of register and the bit(s) that are contained with in. All bits are set to default values by hard reset. Sticky bits retain their states between hard resets.

Term	Description
RO	**Read Only**. If a register bit is read only, the hardware sets its state. The bit may be read by software. Writes to this bit have no effect.
WO	**Write Only.** The register bit is not implemented as a bit. The write causes some hardware event to take place.
RW	**Read/Write**. A register bit with this attribute can be read and written by software.
RC	**Read Clear:** The bit or bits can be read by software, but the act of reading causes the value to be cleared.
RCW	**Read Clear/Write:** A register bit with this attribute, will get cleared after the read. The register bit can be written.

Term	Description
RWC	**Read / Write Clear**. A register bit with this attribute, can be read or cleared by software. In order to clear this bit, a one must be written to it. Writing a zero will have no effect.
RWS	**Read / Write / Set:** A register bit can be either read or set by software. In order to set this bit, a one must be written to it. Writing a zero to this bit has no effect. Hardware will clear this bit.
RWL	**Read / Write / Lock**. A register bit with this attribute can be read or written by software. Hardware or a configuration bit can lock bit and prevent it from updated.
RWO	**Read / Write Once.** A register bit with this attribute can be written to only once after power up. After the first write, the bit becomes read only. This attribute is applied on a bit by bit basis. For example, if the RWO attribute is applied to a 2 bit field, and only one bit is written, then the written bit cannot be rewritten (unless reset). The unwritten bit, of the field, may still be written once. This is special case of RWL.
RRW	**Read / Restricted Write.** This bit can be read and written by software. However, only supported values will be written. Writes of non supported values will have no effect.
L	**Lock.** A register bit with this attribute becomes Read Only after a lock bit is set.
RV	**Reserved Bit.** This bit is reserved for future expansion and must not be written. The *PCI Local Bus Specification,* Revision 2.2 requires that reserved bits must be preserved. Any software that modifies a register that contains a reserved bit is responsible for reading the register, modifying the desired bits, and writing back the result.
Reserved Bits	Some of the MCH registers described in this section contain reserved bits. These bits are labeled “Reserved”. Software must deal correctly with fields that are reserved. On reads, software must use appropriate masks to extract the defined bits and not rely on reserved bits being any particular value. On writes, software must ensure that the values of reserved bit positions are preserved. That is, the values of reserved bit positions must first be read, merged with the new values for other bit positions and then written back. Note that software does not need to perform a read-merge-write operation for the Configuration Address (CONFIG_ADDRESS) register.
Reserved Registers	In addition to reserved bits within a register, the MCH contains address locations in the configuration space of the Host-ESI bridge entity that are marked either “Reserved” or “Intel Reserved”. The MCH responds to accesses to “Reserved” address locations by completing the host cycle. When a “Reserved” register location is read, a zero value is returned. (“Reserved” registers can be 8, 16, or 32 bits in size). Writes to “Reserved” registers have no effect on the MCH. Registers that are marked as “Intel Reserved” must not be modified by system software. Writes to “Intel Reserved” registers may cause system failure. Reads to “Intel Reserved” registers may return a non-zero value.
Default Value upon a Reset	Upon a reset, the MCH sets all of its internal configuration registers to predetermined default states. Some register values at reset are determined by external strapping options. The default state represents the minimum functionality feature set required to successfully bring up the system. Hence, it does not represent the optimal system configuration. It is the responsibility of the system initialization software (usually BIOS) to properly determine the DRAM configurations, operating parameters and optional system features that are applicable, and to program the MCH registers accordingly.
“ST” appended to the end of a bit name	The bit is “sticky” or unchanged by a hard reset. These bits can only be cleared by a PWRGOOD reset.

3.2 Platform Configuration Structure

In some previous chipsets, the MCH and the South Bridge were physically connected by PCI bus 0. From a configuration standpoint, both components appeared to be on PCI bus 0 which was also the system’s primary PCI expansion bus. The MCH contained two PCI devices while the south bridge was considered one PCI device with multiple functions.

In the Intel 5000X Chipset platform the configuration structure is significantly different. The MCH and the Intel 631xESB/632xESB I/O Controller Hub are physically connected by the ESI interface; thus, from a configuration standpoint, the ESI interface is logically PCI bus 0. As a result, all devices internal to the MCH and Intel 631xESB/632xESB I/O Controller Hub appear to be on PCI bus 0. The system’s primary PCI expansion bus is

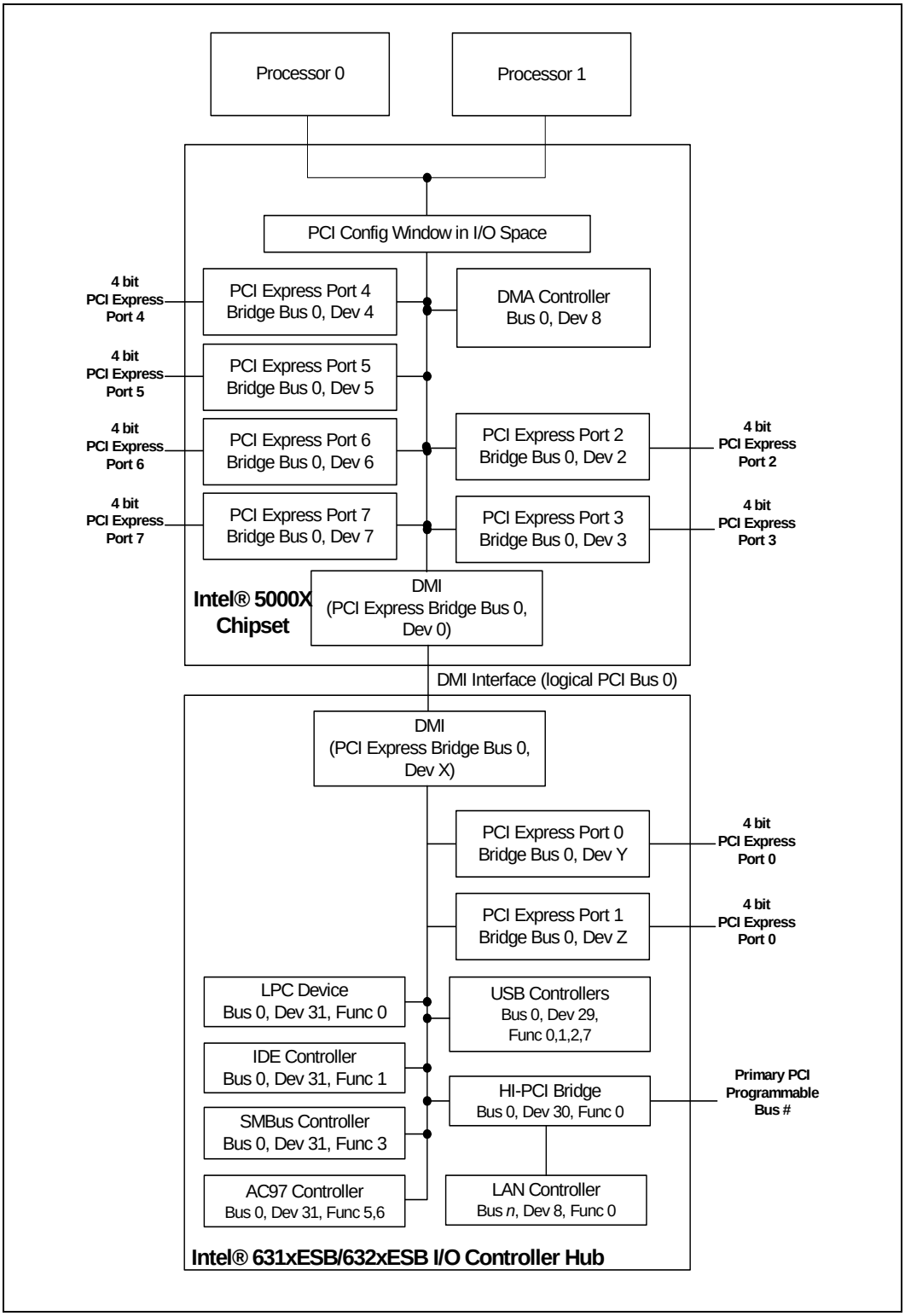
physically attached to the Intel 631xESB/632xESB I/O Controller Hub and, from a configuration perspective, appears to be a hierarchical PCI bus behind a PCI-to-PCI bridge; therefore, it has a programmable PCI Bus number.

The MCH contains 14 PCI devices within a single physical component. The configuration registers for these devices are mapped as devices residing on PCI bus 0.

- **Device 0:** ESI bridge/PCI Express Port 0. Logically, this appears as a PCI device that resides on PCI bus 0. Physically Device 0, Function 0 contains the PCI Express configuration registers for the ESI port, and other MCH specific registers.
- **Device 2:** PCI Express 2. Logically this appears as a PCI device residing on bus 0. Device 2, Function 0 is routed to the PCI Express configuration registers for PCI Express port 2. When PCI Express ports 2 and 3 are combined into a single x8 port, controlled by port 2 registers, Device 3, Function 0 (port 3) configuration registers are inactive. PCI Express port 2 resides at DID of 25E2h.
- **Device 3:** PCI Express 3. Logically this appears as a PCI device that resides on bus 0. Device 3, Function 0 contains the PCI Express configuration registers for PCI Express port 3. When PCI Express ports 2 and 3 are combined into a single x8 port, controlled by port 2 registers, these configuration registers are inactive. PCI Express port 3 resides at DID of 25E3h.
- **Device 4:** PCI Express 4. Logically this appears as a PCI device that resides on bus 0. Device 4, Function 0 contains the PCI Express configuration registers for PCI Express port 4. When PCI Express ports 4, 5, 6, and 7 are combined into a single x16 graphics port, Device 4, Function 0 contains the configuration registers and Device 5, Function 0 (port 5), Device 6, Function 0 (port 6), and Device 7, Function 0 (port 7), configuration registers are inactive. PCI Express port 4 resides at DID of 25E4h.
- **Device 5:** PCI Express 5. Logically this appears as a PCI device that resides on bus 0. Device 5, Function 0 contains the PCI Express configuration registers for PCI Express port 5. When PCI Express ports 4, 5, 6 and 7 are combined into a single x16 graphics port Device 4, Function 0 contains the configuration registers, and these configuration registers are inactive. PCI Express port 5 resides at DID of 25E5h.
- **Device 6:** PCI Express 6. Logically this appears as a PCI device residing on bus 0. Device 6, Function 0 contains the PCI Express configuration registers for PCI Express port 6. When PCI Express ports 4, 5, 6 and 7 are combined into a single x16 graphics port Device 4, Function 0 contains the configuration registers, and these configuration registers are inactive. PCI Express port 6 resides at DID of 25E6h.
- **Device 7:** PCI Express 7. Logically this appears as a PCI device residing on bus 0. Device 7, Function 0 contains the PCI Express configuration registers for PCI Express port 7. When PCI Express ports 4, 5, 6 and 7 are combined into a single x16 graphics port Device 4, Function 0 contains the configuration registers, and these configuration registers are inactive. PCI Express port 2 resides at DID of 25E7h.
- **Device 8:** DMA Engine Controller. Logically this appears as DMA device residing on bus 0. Device 8, Function 0 contains the DMA controller configuration registers for the DMA channels. Device 8, Function 1 contains the MMIO space configuration registers. The DMA Engine controller resides at DID 1A38h.
- **Device 9:** Device 9, Function 0 is routed to the Advanced Memory Buffer memory map. This interface is supported through the JTAG and SMBus interfaces and AMBSELECT register only.

- **Device 16**: Device 16, Function 0 is routed to the Frontside Bus (FSB) Controller, Interrupt and System Address registers. Function 1 is routed to the Frontside Bus Address Mapping, Memory Control, and Error registers. Function 2 is routed to FSB Error Registers. These devices reside at DID 25F0h.
- **Device 17**: Device 17, Function 0 is routed to the Coherency Engine and Data Manager registers. These devices reside at DID 25F1h.
- **Device 19:** Device 19, Function 0 is routed to the Debug and Miscellaneous registers. These devices reside at DID 25F3h.
- **Device 21:** Device 21, Function 0, FBD Branch 0 Memory Map, Error Flag/Mask, and Channel Control registers. These devices reside at DID 25F5h.
- **Device 22:** Device 22, Function 0, FBD Branch 1 Memory Map, Error Flag/Mask, and Channel Control registers. These devices reside at DID 25F6h.

Figure 3-1. Conceptual Intel 5000X Chipset MCH PCI Configuration Diagram



3.3 Routing Configuration Accesses

Intel 5000X Chipset MCH supports both PCI Type 0 and Type 1 configuration access mechanisms as defined in the PCI Local Bus Specification, Revision 2.3. PCI Revision 2.3 defines hierarchical PCI busses. Type 0 configuration access are used for registers located within a PCI device that resides on the local PCI bus. that is, The PCI bus the transaction is initiated on. Type 0 configuration transactions are not propagated beyond the local PCI bus. Type 0 configuration transactions must be claimed by a local device or master aborted.

Type 1 configuration accesses are used for devices residing on subordinate PCI buses. i.e Devices that are connected via PCI-to-PCI bridges. All targets except PCI-to-PCI bridges ignore Type 1 configuration transactions. PCI-to-PCI bridges decode the bus number information in Type 1 transactions. It the transaction is targeted to a device local to the PCI-to-PCI bridge it is translated into a Type 0 transaction and issued to the device. If the transaction is targeted to a bus subordinate (behind) to PCI-to-PCI bridge, it passed through unchanged. Otherwise the Type 1 transaction is dropped.

Accesses to non operational or non existent devices are master aborted. This means that writes are dropped and reads return all 1's.

3.3.1 Standard PCI Bus Configuration Mechanism

The PCI Bus defines a slot based "configuration space" that supports up to 32 devices. Each device is allowed to contain up to eight functions with each function containing up to 256, 8-bit configuration registers. The PCI specification defines two bus cycles to access the PCI configuration space: Configuration Read and Configuration Write. Memory and I/O spaces are supported directly by the processor. Configuration space is supported by a mapping mechanism implemented within the MCH. The PCI 2.3 specification defines the configuration mechanism to access configuration space. The configuration access mechanism makes use of the CONFIG_ADDRESS Register (at I/O address 0CF8h through 0CFBh) and CONFIG_DATA Register (at I/O address 0CFCh through 0CFFh). To reference a configuration register a DWord I/O write cycle is used to place a value into CONFIG_ADDRESS that specifies the PCI bus, the device on that bus, the function within the device, and a specific configuration register of the device function being accessed. CONFIG_ADDRESS[31] must be set to 1b, to enable a configuration cycle. CONFIG_DATA then becomes a window into the four bytes of configuration space specified by the contents of CONFIG_ADDRESS. Any read or write to CONFIG_DATA will result in the MCH translating the CONFIG_ADDRESS into the appropriate configuration cycle.

The MCH is responsible for translating and routing the processor's I/O accesses to the CONFIG_ADDRESS and CONFIG_DATA registers to internal MCH configuration registers.

3.3.2 PCI Bus 0 Configuration Mechanism

The MCH decodes the Bus Number (bits 23:16) and the Device Number fields of the CONFIG_ADDRESS register. If the Bus Number field of CONFIG_ADDRESS is 0, the configuration cycle is targeting a device on PCI Bus 0.

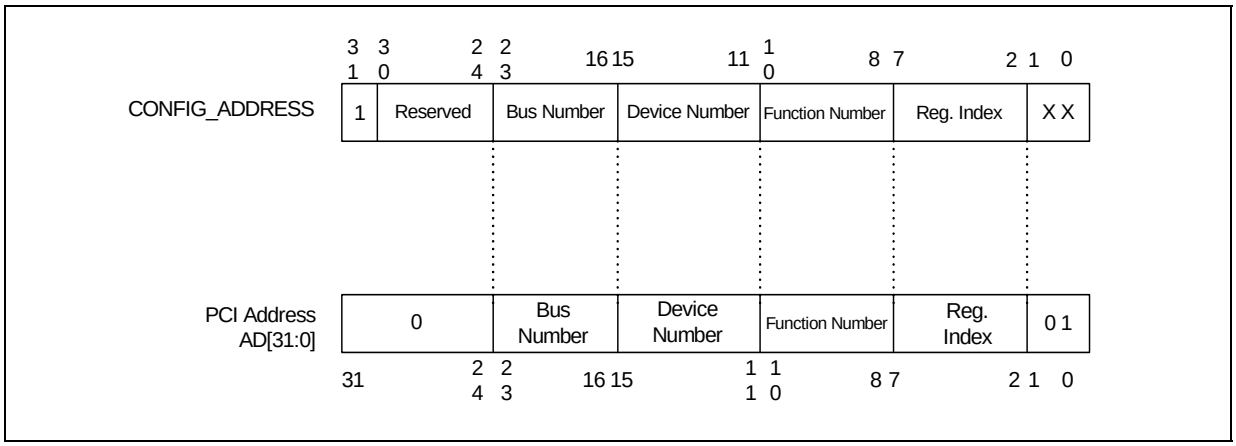
The ESI bridge entity within the MCH is hardwired as Device 0 on PCI Bus 0. The ESI bridge passes PCI south bridge configuration requests to the south bridge.

3.3.3 Primary PCI and Downstream Configuration Mechanism

If the Bus Number in the CONFIG_ADDRESS is non-zero, the MCH will generate a Type 1 PCI configuration cycle. A[1:0] of the ESI request packet for the Type 1 configuration cycle will be 01. Bits 31:2 of the CONFIG_ADDRESS register will be translated to the A[31:2] field of the ESI request packet of the configuration cycle as shown in Figure 3-2. This configuration cycle will be sent over the ESI to Intel 631xESB/632xESB I/O Controller Hub.

If the cycle is forwarded to the Intel® 631xESB/632xESB I/O Controller Hub via ESI, the Intel 631xESB/632xESB I/O Controller Hub compares the non-zero Bus Number with the Secondary Bus Number and Subordinate Bus Number Registers of its PCI-to-PCI bridges to determine if the configuration cycle is meant for primary PCI bus, one of the Intel 631xESB/632xESB I/O Controller Hub's PCI Express ports, or a downstream PCI bus.

Figure 3-2. Type 1 Configuration Address to PCI Address Mapping



3.4 Device Mapping

Each component in a Intel 5000X Chipset system is uniquely identified by a PCI bus address consisting of; Bus Number, Device Number and Function Number. Device configuration is based on the PCI Type 0 configuration conventions. All PCI devices with in a Intel 5000X Chipset platform must support Type 0 configuration accesses. All MCH registers in the Intel 5000X Chipset MCH appear on Bus #0.

All Intel 5000X Chipset MCH configuration registers reside in the configuration space defined by Bus, Device, Function, Register address. Some registers do not appear in all portions of this space and some mechanisms do not access all portions of this space. In general the configuration space is sparsely populated. The following table defines where the various fields of configuration register addresses appear. Each row defines a different access mechanism, register, interface, or decoder. Each column defines a different field of the configuration address.

Table 3-1. Configuration Address Bit Mapping

	Source/ Destination	Bus	Device	Function	Dword Offset		Byte in Dword	Type
					[11:8]	[5:0]		
PCI Express Config Txns (including ESI)	Both	Bus[7:0]	Device[4:0]	Function[2:0]	Extended Register Addr[3:0]	Register [5:0]	1st DW BE[3:0]	Fmt, Type
PCI Express MMCFG on FSB	Source	A[27:20]	A[19:15]	A[14:12]	A[11:8]	A[7:3] BE[7:4]	BE[7:0]	n/a
PCI Express MMCFG from ESI or PCI Express	Not permitted to access MCH or FB-DIMM regs and will be master aborted. Peer to Peer accesses targeting valid MMIO space will be forwarded through appropriate decoding.							
CPU/Inbound CB_BAR MMIO Access	Source	0	8	1	A[11:8]	A[7:3] BE[7:4]	BE[7:0]	n/a
CFGADR Register	Source	Bus Number [7:0]	DeviceID [4:0]	Function Number[2:0]	not present	Register Address [5:0]	Not present	n/a
CFC on FSB	Source	CFGADR Register, see row above					BE[7:4]	n/a
JTAG Config Access	Source	Bus Number [7:0]	DeviceID [4:0]	Function Number[2:0]	Extended Register Addr[3:0]	Register Address[7:2]	Register Address [1:0]	n/a
SMBus Config Access	Source	Bus Number [7:0]	Dev[4:0]	Func[2:0]	Reg Number [11:8]	Reg[7:2]	command, Register Number	n/a
Fixed MCH Memory Mapped on FSB	Source	0	16	0	cannot access	A[15:10]	All accesses are 4 byte	n/a
MCH Register Decoding	Destination	00000000	See Table 14-4	Function[2:0]	Dword Offset[9:6]	Dword Offset[5:0]	Byte[3:0]	n/a
FB-DIMM Config Cmds	Destination	A[23:15] always 0	See Note[a]		Cannot access	A[7:3] BE[7:4]	BE[7:0]	n/a

Notes:

a. These accesses are used to select channel/DIMM based on the AMBASE register.

3.4.1 Device Identification for Intel 5000X Chipset

Table 3-2. Memory Control Hub ESI Device Identification

Component	Register Group	DID	Device	Function
Intel 5000X Chipset	Enterprise South Bridge Interface	25C0h	0	0

3.4.2 Special Device and Function Routing

All devices in the Intel 5000X Chipset MCH reside on Bus 0. The following table describes the devices and functions that the MCH implements or routes specially. The DIMM component designator consists of a three-digit code: the first digit is the branch, the second digit is the channel on the branch, and the third digit is the DIMM (FB-DIMM command “DS” field) on the channel.

Table 3-3. Functions Specially Handled by the MCH

Component	Register Group	DID	Device	Function	Comment
MCH	PCI Express Port 2	25E2h	2	0	Depending on what is connected to these ports, some may not be accessible.
MCH	PCI Express Port 3	25E3h	3	0	
MCH	PCI Express Port 4	25E4h	4	0	
MCH	PCI Express Port 5	25E5h	5	0	
MCH	PCI Express Port 6	25E6h	6	0	
MCH	PCI Express Port 7	25E7h	7	0	
MCH	DMA Engine	1A38h	8	0	New device mapping for DMA Engine
MCH	DMA Engine MMIO Space	N/A	8	1	
MCH	Memory Map, Error Flag/Mask, RAS, Channel Control for FB-DIMM Branch 0	25F5h	21	0	Debug and DFT in higher address offsets.
MCH	Memory Map, Error Flag/Mask, RAS, Channel Control for FB-DIMM Branch 1	25F6h	22	0	Debug and DFT in higher address offsets.
MCH	Processor Bus, Boot, Interrupt, System Address	25F0h	16	0	Debug and DFT in higher address offsets.
DIMM	AMB Memory Mapped registers	25E8h	9	0	Route out to AMB per AMBSELECT register only for JTAG/SMBus.
MCH	Address Mapping, Memory Control, Error Logs	25F0h	16	1	Debug and DFT in higher address offsets.
MCH	FSB Error Registers	25F0h	16	2	
MCH	Reserved	TBD	20	all	TBD
MCH	PCI Express Port 2-3	25F7h	2	0	x8 mode. Only port 2 is active
MCH	PCI Express Port 4-5	25F8h	4	0	x8 mode. Only port 4 is active
MCH	PCI Express Port 6-7	25F9h	6	0	x8 mode. Only port 6 is active
MCH	PCI Express Port 4-7	25FAh	4	0	x16 mode. Only port 4 is active

To comply with the PCI specification, accesses to non-existent functions, registers, and bits will be master aborted. This behavior is defined in the following table:

Table 3-4. Access to “Non-Existent” Register Bits

Access to	Writes	Reads
Devices listed in Table 3-2, “Memory Control Hub ESI Device Identification” on page 52 but to functions not listed	Have no effect	MCH returns all ones
Devices listed inTable 3-2, “Memory Control Hub ESI Device Identification” on page 52 but to registers not listed in Section 3.8, “Register Definitions”	Have no effect	MCH returns all zeroes
Reserved bits in registers	Software must read-modify-write to preserve the value	MCH returns all zeroes

3.5 I/O Mapped Registers

There are only two I/O addresses that affect Intel 5000X Chipset MCH state. The first address is the DWORD location (CF8h) references a read/write register that is named CONFIG_ADDRESS. The second DWORD address (CFCh) references a read/write register named CONFIG_DATA. These two addresses are used for the PCI CFCh / CF8h configuration access mechanism.

3.5.1 CFGADR: Configuration Address Register

CFGADR is written only when a processor I/O transaction to I/O location CF8h is referenced as a DWord; a Byte or Word reference will not access this register, but will generate an I/O space access. Therefor the only I/O space taken up by this register is the DWORD at location CF8h. I/O devices that share the same address but use BYTE or WORD registers are not affected because their transactions will pass through the host bridge unchanged.

The CFGADR register contains the Bus Number, Device Number, Function Number, and Register Offset for which a subsequent CFGDAT access is intended. The mapping between fields in this register and PCI Express configuration transactions is defined by Table 3-1.

Table 3-5. I/O Address: CF8h

Bit	Attr	Default	Description
31	RW	0h	**CFGE: Configuration Enable** Unless this bit is set, accesses to the **CFGDAT** register will not produce a configuration access, but will be treated as other I/O accesses. This bit is strictly an enable for the CFC/CF8 access mechanism and is not forwarded to ESI or PCI Express.
30:24	RV	00h	Reserved.
23:16	RW	00h	**Bus Number** If 0, the MCH examines device to determine where to route. If non-zero, route as per PBUSN and SBUSN registers.
15:11	RW	0h	**Device Number** This field is used to select one of the 32 possible devices per bus.
10:8	RW	0h	**Function Number** This field is used to select the function of a locally addressed register.
7:2	RW	00h	**Register Offset** If this register specifies an access to MCH registers, this field specifies a group of four bytes to be addressed. The bytes accessed are defined by the Byte enables of the CFGDAT register access
1:0	RW	0h	Writes to these bits have no effect, reads return 0

3.5.2 CFGDAT: Configuration Data Register

CFGDAT provides data for the 4 bytes of configuration space defined by CFGADR. This register is only accessed if there is an access to I/O address, CFCh on the processor bus and CFGADR.CFGE (configuration enable) bit is set. The byte enables with the I/O access define how many configuration bytes are accessed.

Table 3-6. I/O Address: CFCh

Bit	Attr	Default	Description
31:0	RW	0	**Configuration Data Window** The data written or read to the configuration register (if any) specified by **CFGADR**

3.6 MCH Fixed Memory Mapped Registers

These registers are mapped into the fixed chipset specific range located from FE60 0000h - FE6F FFFFh.These appear at fixed addresses to support the boot process. These registers also appear in the regular PCI Express configuration space.

The following table defines the memory address of the registers in this region.

Table 3-7. Mapping for Fixed Memory Mapped Registers

Register	Memory Address
BOFL0	FE60_C000
BOFL1	FE60_C400
BOFL2	FE60_C800
BOFL3	FE60_CC00
SPAD0	FE60_D000
SPAD1	FE60_D400
SPAD2	FE60_D800
SPAD3	FE60_DC00
SPADS0	FE60_E000
SPADS1	FE60_E400
SPADS2	FE60_E800
SPADS3	FE60_EC00
AMBASE[31:0]	FE61_4800
AMBASE[63:32]	FE61_4C00
HECBASE	FE61_6400

3.7 Detailed Configuration Space Maps

Table 3-8. Device 0, Function 0: PCI Express PCI Space

DID			VID	00h	PEXSLOTCAP		80h
PCISTS			PCICMD	04h	PEXSLOTSTS	PEXSLOTCTRL	84h
CCR			RID	08h		PEXRTCTRL	88h
BIST	HDR	PRI_LT	CLS	0Ch	PEXRTSTS		8Ch
				10h			90h
				14h			94h
				18h			98h
				1Ch			9Ch
				20h			A0h
				24h			A4h
				28h			A8h
SID			SVID	2Ch			ACh
				30h			B0h
			CAPPTR	34h			B4h
				38h			B8h
		INTP	INTL	3Ch			BCh
		PEXLWSTPCTRL		40h			C0h
SSCTRL		CBPRES		44h			C4h
PEXCTRL				48h			C8h
INTXSWZ CTRL		PEXCTRL3	PEXCTRL2	4Ch			CCh
PMCAP				50h			D0h
PMCSR				54h	ESICTRL		D4h
MSICTRL		MSINXPTR	MSICAPID	58h			D8h
MSIAR				5Ch			DCh
MSIDR				60h			E0h
				64h			E4h
				68h			E8h
PEXCAP		PEXCAPL		6Ch			ECh
PEXDEVCAP				70h			F0h
PEXDEVSTS		PEXDEVCTRL		74h			F4h
PEXLNKCAP				78h			F8h
PEXLNKSTS		PEXLNKCTRL		7Ch			FCh

Table 3-9. Device 0, Function 0: PCI Express Extended Registers

PEXENHCAP	100h		180h
UNCERRSTS	104h		184h
UNCERRMSK	108h		188h
UNCERRSEV	10Ch		18Ch
CORERRSTS	110h		190h
CORERRMSK	114h		194h
AERRCAPCTRL	118h		198h
HDRLOG0	11Ch		19Ch
HDRLOG1	120h		1A0h
HDRLOG2	124h		1A4h
HDRLOG3	128h		1A8h
RPERRCMD	12Ch		1ACh
RPERRSTS	130h		1B0h
RPERRSID	134h		1B4h
	138h		1B8h
	13Ch		1BCh
Intel® 5000P Series chipset MCHSPCAPID	140h		1C0h
PEX_ERR_DOCMD	144h		1C4h
EMASK_UNCOR_PEX	148h		1C8h
EMASK_COR_PEX	14Ch		1CCh
EMASK_RP_PEX	150h		1D0h
PEX_FAT_FERR	154h		1D4h
PEX_NF_COR_FERR	158h		1D8h
PEX_FAT_NERR	15Ch		1DCh
PEX_NF_COR_NERR	160h		1E0h
	164h		1E4h
	168h		1E8h
	16Ch		1ECh
PEX_SSERR	170h		1F0h
	174h		1F4h
	178h		1F8h
	17Ch		1FCh

Table 3-10. Device 0, Function 0: PCI Express Intel® Interconnect BIST (Intel® IBIST) Registers

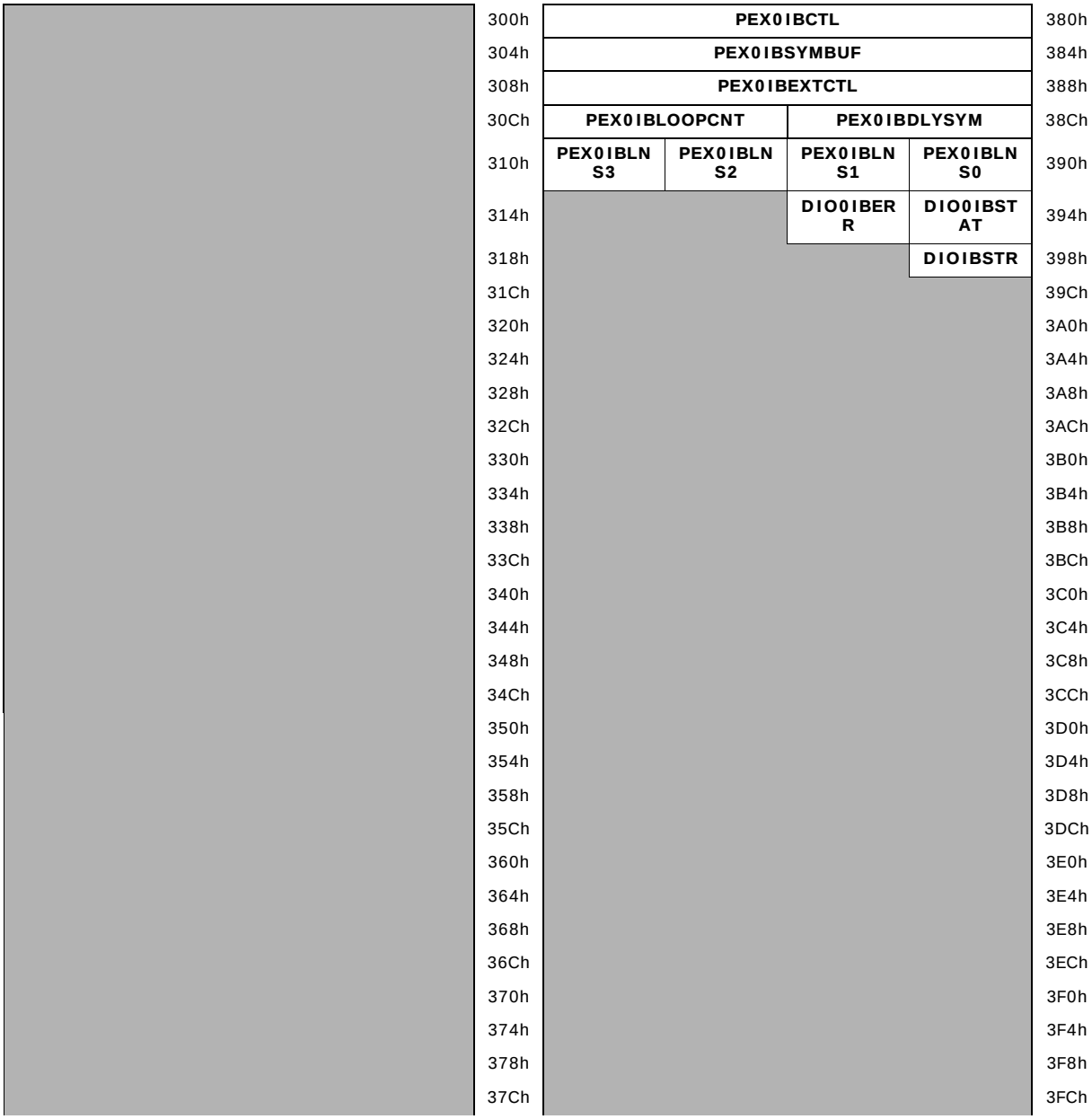


Table 3-11. Device 2-3, Function 0: PCI Express PCI Space

DID		VID		00h	PEXSLOTCAP		80h
PCISTS		PCICMD		04h	PEXSLOTSTS	PEXSLOTCTRL	84h
CCR		RID		08h		PEXRTCTRL	88h
BIST	HDR	PRI_LT	CLS	0Ch	PEXRTSTS		8Ch
				10h			90h
				14h			94h
SEC_LT	SUBUSN	SBUSN	PBUSN	18h			98h
SECSTS		IOLIM	IOBASE	1Ch			9Ch
MLIM		MBASE		20h			A0h
PMLIM		PMBASE		24h			A4h
PMBU				28h			A8h
PMLU				2Ch			ACh
				30h			B0h
			CAPPTR	34h			B4h
				38h			B8h
BCTRL		INTP	INTL	3Ch			BCh
				40h			C0h
SSCTRL				44h			C4h
PEXCTRL				48h			C8h
INTXSWZ CTRL	CBCTRL	PEXCTRL3	PEXCTRL2	4Ch			CCh
PMCAP				50h			D0h
PMCSR				54h			D4h
MSICTRL		MSINXPTR	MSICAPID	58h			D8h
MSIAR				5Ch			DCh
MSIDR				60h			E0h
				64h			E4h
				68h			E8h
PEXCAP		PEXCAPL		6Ch			ECh
PEXDEVCAP				70h			F0h
PEXDEVSTS		PEXDEVCTRL		74h			F4h
PEXLNKCAP				78h			F8h
PEXLNKSTS		PEXLNKCTRL		7Ch			FCh

Table 3-12. Device 2-3, Function 0: PCI Express Extended Registers

PEXENHCAP	100h		180h
UNCERRSTS	104h		184h
UNCERRMSK	108h		188h
UNCERRSEV	10Ch		18Ch
CORERRSTS	110h		190h
CORERRMSK	114h		194h
AERRCAPCTRL	118h		198h
HDRLOG0	11Ch		19Ch
HDRLOG1	120h		1A0h
HDRLOG2	124h		1A4h
HDRLOG3	128h		1A8h
RPERRCMD	12Ch		1ACh
RPERRSTS	130h		1B0h
RPERRSID	134h		1B4h
	138h		1B8h
	13Ch		1BCh
Intel® 5000P Series chipset MCHSPCAPID	140h		1C0h
PEX_ERR_DOCMD	144h		1C4h
EMASK_UNCOR_PEX	148h		1C8h
EMASK_COR_PEX	14Ch		1CCh
EMASK_RP_PEX	150h		1D0h
PEX_FAT_FERR	154h		1D4h
PEX_NF_COR_FERR	158h		1D8h
PEX_FAT_NERR	15Ch		1DCh
PEX_NF_COR_NERR	160h		1E0h
	164h		1E4h
PEX_UNIT_FERR	168h		1E8h
PEX_UNIT_NERR	16Ch		1ECh
PEX_SSER R	170h		1F0h
	174h		1F4h
	178h		1F8h
	17Ch		1FCh

Table 3-13. Device 2-3, Function 0: PCI Express Intel IBIST Registers

	300h	PEX[3:2]IBCTL				380h
	304h	PEX[3:2]IBSYMBUF				384h
	308h	PEX[3:2]IBEXTCTL				388h
	30Ch	PEX[3:2]IBLOOPCNT		PEX[3:2]IBDLYSYM		38Ch
	310h	PEX[3:2]IBLNS3	PEX[3:2]IBLNS2	PEX[3:2]IBLNS1	PEX[3:2]IBLNS0	390h
	314h					394h
	318h					398h
	31Ch					39Ch
	320h					3A0h
	324h					3A4h
	328h					3A8h
	32Ch					3ACh
	330h					3B0h
	334h					3B4h
	338h					3B8h
	33Ch					3BCh
	340h					3C0h
	344h					3C4h
	348h					3C8h
	34Ch					3CCh
	350h					3D0h
	354h					3D4h
	358h					3D8h
	35Ch					3DCh
	360h					3E0h
	364h					3E4h
	368h					3E8h
	36Ch					3ECh
	370h					3F0h
	374h					3F4h
	378h					3F8h
	37Ch					3FCh

Table 3-14. Device 4, Function 0: PCI Express PCI Space

DID			VID	00h	PEXSLOTCAP			80h
PCISTS			PCICMD	04h	PEXSLOTSTS		PEXSLOTCTRL	84h
CCR			RID	08h			PEXRTCTRL	88h
BIST	HDR	PRI_LT	CLS	0Ch	PEXRTSTS			8Ch
				10h				90h
				14h				94h
SEC_LT	SUBUSN	SBUSN	PBUSN	18h				98h
SECSTS		IOLIM	IOBASE	1Ch				9Ch
MLIM		MBASE		20h				A0h
PMLIM		PMBASE		24h				A4h
PMBU				28h				A8h
PMLU				2Ch				ACh
				30h				B0h
			CAPPTR	34h				B4h
				38h				B8h
BCTRL		INTP	INTL	3Ch				BCh
				40h				C0h
SSCTRL				44h				C4h
PEXCTRL				48h				C8h
INTXSWZ CTRL		PEXCTRL3	PEXCTRL2	4Ch				CCh
PMCAP				50h				D0h
PMCSR				54h				D4h
MSICTRL		MSINXPTR	MSICAPID	58h				D8h
MSIAR				5Ch				DCh
MSIDR				60h				E0h
				64h				E4h
				68h				E8h
PEXCAP		PEXCAPL		6Ch				ECh
PEXDEVCAP				70h				F0h
PEXDEVSTS		PEXDEVCTRL		74h				F4h
PEXLNKCAP				78h				F8h
PEXLNKSTS		PEXLNKCTRL		7Ch				FCh

Table 3-15. Device 4, Function 0: PCI Express Extended Registers

PEXENHCAP		100h		180h
UNCERRSTS		104h		184h
UNCERRMSK		108h		188h
UNCERRSEV		10Ch		18Ch
CORERRSTS		110h		190h
CORERRMSK		114h		194h
AERRCAPCTRL		118h		198h
HDRLOG0		11Ch		19Ch
HDRLOG1		120h		1A0h
HDRLOG2		124h		1A4h
HDRLOG3		128h		1A8h
RPERRCMD		12Ch		1ACh
RPERRSTS		130h		1B0h
RPERRSID		134h		1B4h
		138h		1B8h
		13Ch		1BCh
Intel® 5000P Series chipset MCHSPCAPID		140h		1C0h
PEX_ERR_DOCMD		144h		1C4h
EMASK_UNCOR_PEX		148h		1C8h
EMASK_COR_PEX		14Ch		1CCh
EMASK_RP_PEX		150h		1D0h
PEX_FAT_FERR		154h		1D4h
PEX_NF_COR_FERR		158h		1D8h
PEX_FAT_NERR		15Ch		1DCh
PEX_NF_COR_NERR		160h		1E0h
		164h		1E4h
PEX_UNIT_FERR		168h		1E8h
PEX_UNIT_NERR		16Ch		1ECh
	PEX_SSERR	170h		1F0h
		174h		1F4h
		178h		1F8h
		17Ch		1FCh

Table 3-16. Device 4, Function 0: PCI Express Intel IBIST Registers

	300h	PEX4IBCTL			380h
	304h	PEX4IBSYMBUF			384h
	308h	PEX4IBEXTCTL			388h
	30Ch	PEX4IBLOOPCNT		PEX4IBDLYSYM	38Ch
	310h	PEX4IBLNS3	PEX4IBLNS2	PEX4IBLNS1	PEX4IBLNS0	390h
	314h					394h
	318h					398h
	31Ch					39Ch
	320h					3A0h
	324h					3A4h
	328h					3A8h
	32Ch					3ACh
	330h					3B0h
	334h					3B4h
	338h					3B8h
	33Ch					3BCh
	340h					3C0h
	344h					3C4h
	348h					3C8h
	34Ch					3CCh
	350h					3D0h
	354h					3D4h
	358h					3D8h
	35Ch					3DCh
	360h					3E0h
	364h					3E4h
	368h					3E8h
	36Ch					3ECh
	370h					3F0h
	374h					3F4h
	378h					3F8h
	37Ch					3FCh

Table 3-17. Device 5-7, Function 0: PCI Express PCI Space

DID			VID	00h	PEXSLOTCAP		80h
PCISTS			PCICMD	04h	PEXSLOTSTS	PEXSLOTCTRL	84h
CCR			RID	08h		PEXRTCTRL	88h
BIST	HDR	PRI_LT	CLS	0Ch	PEXRTSTS		8Ch
				10h			90h
				14h			94h
SEC_LT	SUBUSN	SBUSN	PBUSN	18h			98h
SECSTS		IOLIM	IOBASE	1Ch			9Ch
MLIM		MBASE		20h			A0h
PMLIM		PMBASE		24h			A4h
PMBU				28h			A8h
PMLU				2Ch			ACh
				30h			B0h
			CAPPTR	34h			B4h
				38h			B8h
BCTRL		INTP	INTL	3Ch			BCh
				40h			C0h
SSCTRL				44h			C4h
PEXCTRL				48h			C8h
INTXSWZ CTRL		PEXCTRL3	PEXCTRL2	4Ch			CCh
PMCAP				50h			D0h
PMCSR				54h			D4h
MSICTRL		MSINXPTR	MSICAPID	58h			D8h
MSIAR				5Ch			DCh
MSIDR				60h			E0h
				64h			E4h
				68h			E8h
PEXCAP		PEXCAPL		6Ch			ECh
PEXDEVCAP				70h			F0h
PEXDEVSTS		PEXDEVCTRL		74h			F4h
PEXLNKCAP				78h			F8h
PEXLNKSTS		PEXLNKCTRL		7Ch			FCh

Table 3-18. Device 5-7, Function 0: PCI Express Extended Registers

PEXENHCAP	100h		180h
UNCERRSTS	104h		184h
UNCERRMSK	108h		188h
UNCERRSEV	10Ch		18Ch
CORERRSTS	110h		190h
CORERRMSK	114h		194h
AERRCAPCTRL	118h		198h
HDRLOG0	11Ch		19Ch
HDRLOG1	120h		1A0h
HDRLOG2	124h		1A4h
HDRLOG3	128h		1A8h
RPERRCMD	12Ch		1ACh
RPERRSTS	130h		1B0h
RPERRSID	134h		1B4h
	138h		1B8h
	13Ch		1BCh
Intel® 5000P Series chipset MCHSPCAPID	140h		1C0h
PEX_ERR_DOCMD	144h		1C4h
EMASK_UNCOR_PEX	148h		1C8h
EMASK_COR_PEX	14Ch		1CCh
EMASK_RP_PEX	150h		1D0h
PEX_FAT_FERR	154h		1D4h
PEX_NF_COR_FERR	158h		1D8h
PEX_FAT_NERR	15Ch		1DCh
PEX_NF_COR_NERR	160h		1E0h
	164h		1E4h
PEX_UNIT_FERR	168h		1E8h
PEX_UNIT_NERR	16Ch		1ECh
PEX_SSERR	170h		1F0h
	174h		1F4h
	178h		1F8h
	17Ch		1FCh

Table 3-19. Device 5-7, Function 0: PCI Express Intel IBIST Registers

	300h	PEX[7:5]IBCTL			380h
	304h	PEX[7:5]IBSYMBUF			384h
	308h	PEX[7:5]IBEXTCTL			388h
	30Ch	PEX[7:5]IBLOOPCNT		PE[7:5]IBDLYSYM	38Ch
	310h	PEX[7:5]IBLNS3	PEX[7:5]IBLNS2	PEX[7:5]IBLNS1	PEX[7:5]IBLNS0	390h
	314h					394h
	318h					398h
	31Ch					39Ch
	320h					3A0h
	324h					3A4h
	328h					3A8h
	32Ch					3ACh
	330h					3B0h
	334h					3B4h
	338h					3B8h
	33Ch					3BCh
	340h					3C0h
	344h					3C4h
	348h					3C8h
	34Ch					3CCh
	350h					3D0h
	354h					3D4h
	358h					3D8h
	35Ch					3DCh
	360h					3E0h
	364h					3E4h
	368h					3E8h
	36Ch					3ECh
	370h					3F0h
	374h					3F4h
	378h					3F8h
	37Ch					3FCh

Table 3-20. Device 9, Function 0: AMB Switching Window Registers

Route out AMB as per AMBSELECT register	0h - 255h

This function is only accessible by SMBus or JTAG. Other accesses will be routed to ESI and get master aborted.

Accessing to this function is routed out to FB-DIMM channel as per AMBSELECT register subject to AMBPRESENT register settings.

Table 3-21. Device 16, Function 0: Processor Bus, Boot, and Interrupt

DID		VID		00h	XTPR0	80h
				04h	XTPR1	84h
CCR			RID	08h	XTPR2	88h
	HDR			0Ch	XTPR3	8Ch
				10h	XTPR4	90h
				14h	XTPR5	94h
				18h	XTPR6	98h
				1Ch	XTPR7	9Ch
				20h	XTPR8	A0h
				24h	XTPR9	A4h
				28h	XTPR10	A8h
SID		SVID		2Ch	XTPR11	ACh
				30h	XTPR12	B0h
				34h	XTPR13	B4h
				38h	XTPR14	B8h
				3Ch	XTPR15	BCh
CPURSTCAPTMR		SYRE		40h	BOFL0	C0h
POC				44h	BOFL1	C4h
AMBASE				48h	BOFL2	C8h
				4Ch	BOFL3	CCh
AMR				50h	SPAD0	D0h
MAXAMB PERCH	MAXCH	AMBSELECT		54h	SPAD1	D4h
PAM2	PAM1	PAM0		58h	SPAD2	D8h
PAM6	PAM5	PAM4	PAM3	5Ch	SPAD3	DCh
EXSMRTOP	EXSMRC	SMRAMC	EXSMRAMC	60h	SPADS0	E0h
HECBASE				64h	SPADS1	E4h
REDIRBUCKETS				68h	SPADS2	E8h
REDIRCTL				6Ch	SPADS3	ECh
				70h	**PROCENABLE**	F0h
				74h		F4h
				78h		F8h
				7Ch		FCh

Table 3-22. Device 16, Function 1: Memory Branch Map, Control, Errors

DID	VID		00h		MIR0		80h
			04h		MIR1		84h
CCR		RID	08h		MIR2		88h
	HDR		0Ch		AMIR0		8Ch
			10h		AMIR1		90h
			14h		AMIR2		94h
			18h	FERR_FAT_FBD			98h
			1Ch	NERR_FAT_FBD			9Ch
			20h	FERR_NF_FBD			A0h
			24h	NERR_NF_FBD			A4h
			28h	EMASK_FBD			A8h
SID	SVID		2Ch	ERR0_FBD			ACh
			30h	ERR1_FBD			B0h
			34h	ERR2_FBD			B4h
			38h	MCERR_FBD			B8h
			3Ch	NRECMEMA			BCh
MC			40h	NRECMEMB			C0h
			44h	NRECFGLOG			C4h
DRTA			48h	NRECFBDA			C8h
DRTB			4Ch	NRECFBDB			CCh
ERRPER			50h	NRECFBDC			D0h
	DDRFRQ		54h	NRECFBDD			D4h
MCA			58h	NRECFBDE			D8h
			5Ch	RESERVED			DCh
		GBLACT	60h	RECMEMA			E0h
THRTCTRL	THRTHI	THRTMID	THRTLOW	64h	RECMEMB			E4h
THRTSTS1		THRTSTS0		68h	RECFGLOG			E8h
		TOLM		6Ch	RECFBDA			ECh
				70h	RECFBDB			F0h
				74h	RECFBDC			F4h
				78h	RECFBDD			F8h
REDMEMB				7Ch	RECFBDE			FCh

Table 3-23. Device 16, Function 2: RAS

DID		VID		00h				80h
				04h				84h
CCR			RID	08h				88h
	HDR			0Ch				8Ch
				10h				90h
				14h				94h
				18h				98h
				1Ch				9Ch
				20h				A0h
				24h				A4h
				28h				A8h
SID		SVID		2Ch				ACh
				30h	NRECSF			B0h
				34h				B4h
				38h	RECSF			B8h
				3Ch				BCh
FERR_Global				40h	NERR_NF_INT	NERR_FAT_INT	FERR_NF_INT	FERR_FAT_INT	C0h
NERR_Global				44h	NRECINT				C4h
				48h	RECINT				C8h
				4Ch				EMASK_INT	CCh
				50h	MCERR_INT	ERR2_INT	ERR1_INT	ERR0_INT	D0h
				54h					D4h
				58h					D8h
				5Ch					DCh
				60h	THRMSR_THRLD_HI	THRMSR_THRLD_LO	THRMSR_OP		E0h
				64h	THRMSR_STS_HI		THRMSR_STS_LO		E4h
				68h	THRMSR_CTRL		THRMSR_DOCMD	THRMSR_MSK	E8h
				6Ch					ECh
				70h					F0h
				74h					F4h
				78h					F8h
				7Ch					FCh

Table 3-24. Device 21, 22, Function 0: FB-DIMM Map, Control, RAS

DID		VID		00h			MTR0		80h
				04h			MTR1		84h
CCR			RID	08h			MTR2		88h
	HDR			0Ch			MTR3		8Ch
				10h	DMIR0				90h
				14h	DMIR1				94h
				18h	DMIR2				98h
				1Ch	DMIR3				9Ch
				20h	DMIR4				A0h
				24h	UERRCNT				A4h
				28h	CERRCNT				A8h
SID		SVID		2Ch	BADRAMA				ACh
				30h			BADRAMB		B0h
				34h	BADCNT				B4h
				38h					B8h
				3Ch					BCh
		SPCPS	SPCPC	40h			FBDSBTXCFG1	FBDSBTXCFG0	C0h
FBDICMD1	FBDICMD0	FBDLVL1	FBDLVL0	44h					C4h
FBDST	HPST0			48h					C8h
FBDHPC	HPHPC0			4Ch					CCh
	HPST1			50h					D0h
	HPCTL1			54h					D4h
FBDISTS1		FBDISTS0		58h					D8h
				5Ch					DCh
				60h					E0h
AMBPRESENT1		AMBPRESENT0		64h					E4h
				68h					E8h
				6Ch					ECh
				70h					F0h
SPD1		SPD0		74h					F4h
SPDCMD0				78h					F8h
SPDCMD1				7Ch					FCh

Table 3-25. Device 21, Function 0: FB-DIMM 0 Intel IBIST Registers

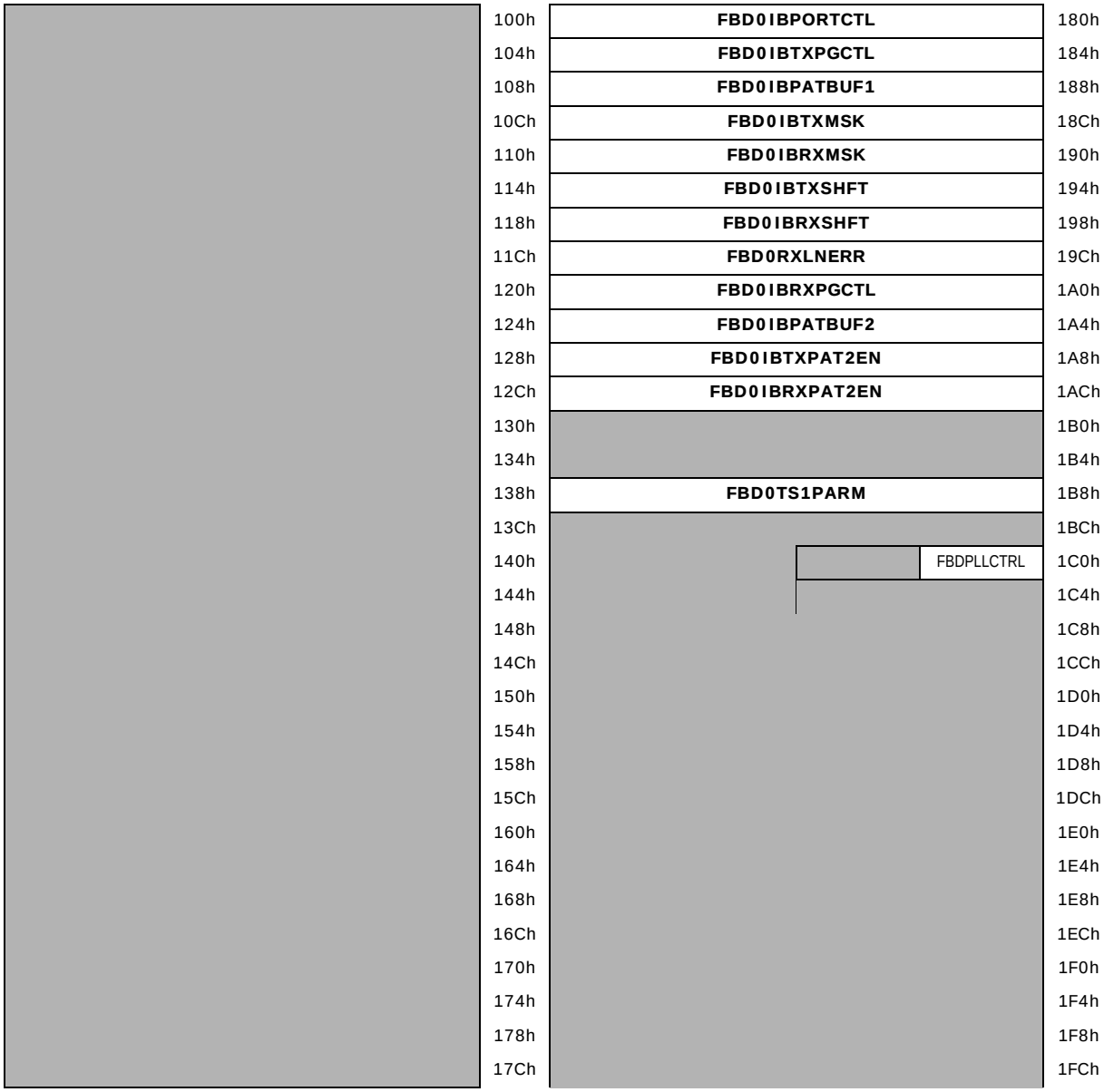


Table 3-26. Device 21, Function 0: FB-DIMM 1 IBST Registers

	200h	FBD1IBPORTCTL	280h
	204h	FBD1IBTXPGCTL	284h
	208h	FBD1IBPATBUF	288h
	20Ch	FBD1IBTXMSK	28Ch
	210h	FBD1IBRXMSK	290h
	214h	FBD1IBTXSHFT	294h
	218h	FBD1IBRXSHFT	298h
	21Ch	FBD1RXLNERR	29Ch
	220h	FBD1IBRXPGCTL	2A0h
	224h	FBD1IBPATBUF2	2A4h
	228h	FBD1IBTXPAT2EN	2A8h
	22Ch	FBD1IBRXPAT2EN	2ACh
	230h		2B0h
	234h		2B4h
	238h	FBD1TS1PARM	2B8h
	23Ch		2BCh
	240h		2C0h
	244h		2C4h
	248h		2C8h
	24Ch		2CCh
	250h		2D0h
	254h		2D4h
	258h		2D8h
	25Ch		2DCh
	260h		2E0h
	264h		2E4h
	268h		2E8h
	26Ch		2ECh
	270h		2F0h
	274h		2F4h
	278h		2F8h
	27Ch		2FCh

Table 3-27. Device 22, Function 0: FB-DIMM 2 IBST Registers

	100h	FBD2IBPORTCTL	180h
	104h	FBD2IBPGCTL	184h
	108h	FBD2IBPATBUF	188h
	10Ch	FBD2IBTXMSK	18Ch
	110h	FBD2IBRXMSK	190h
	114h	FBD2IBTXSHFT	194h
	118h	FBD2IBRXSHFT	198h
	11Ch	FBD2RXLNERR	19Ch
	120h	FBD2IBRXPGCTL	1A0h
	124h	FBD2IBPATBUF2	1A4h
	128h	FBD2IBTXPAT2EN	1A8h
	12Ch	FBD2IBRXPAT2EN	1ACh
	130h		1B0h
	134h		1B4h
	138h	FBD2TS1PARM	1B8h
	13Ch		1BCh
	140h		1C0h
	144h		1C4h
	148h		1C8h
	14Ch		1CCh
	150h		1D0h
	154h		1D4h
	158h		1D8h
	15Ch		1DCh
	160h		1E0h
	164h		1E4h
	168h		1E8h
	16Ch		1ECh
	170h		1F0h
	174h		1F4h
	178h		1F8h
	17Ch		1FCh

Table 3-28. Device 22, Function 0: FB-DIMM 3 Intel IBIST Registers

	200h	FBD3IBPORTCTL
	204h	FBD3IBPGCTL
	208h	FBD3IBPATBUF
	20Ch	FBD3IBTXMSK
	210h	FBD3IBRXMSK
	214h	FBD3IBTXSHFT
	218h	FBD3IBRXSHFT
	21Ch	FBD3RXLNERR
	220h	FBD3IBRXPGCTL
	224h	FBD3IBPATBUF2
	228h	FBD3IBTXPAT2EN
	22Ch	FBD3IBRXPAT2EN
	230h	
	234h	
	238h	FBD3TS1PARM
	23Ch	
	240h	
	244h	
	248h	
	24Ch	
	250h	
	254h	
	258h	
	25Ch	
	260h	
	264h	
	268h	
	26Ch	
	270h	
	274h	
	278h	
	27Ch	

280h	
284h	
288h	
28Ch	
290h	
294h	
298h	
29Ch	
2A0h	
2A4h	
2A8h	
2ACh	
2B0h	
2B4h	
2B8h	
2BCh	
2C0h	
2C4h	
2C8h	
2CCh	
2D0h	
2D4h	
2D8h	
2DCh	
2E0h	
2E4h	
2E8h	
2ECh	
2F0h	
2F4h	
2F8h	
2FCh	

3.8 Register Definitions

3.8.1 PCI Standard Registers

These registers appear in every function for every device.

3.8.1.1 VID - Vendor Identification Register

The VID Register contains the vendor identification number. This 16-bit register, combined with the Device Identification Register uniquely identifies the manufacturer of the function with in the MCH. Writes to this register have no effect.

Device:	**0, 2-9, 17, 21, 22**
Function:	**0**
Offset:	**00h**
Device:	**16**
Function:	**0, 2**
Offset:	**00h**

Bit	Attr	Default	Description
15:0	RO	8086h	**Vendor Identification Number** The value assigned to Intel.

3.8.1.2 DID - Device Identification Register

This 16-bit register combined with the Vendor Identification register uniquely identifies the Function with in the MCH. Writes to this register have no effect. See Table 3-3 for the DID of each MCH function.

Device:	**0, 2-9, 17, 21, 22**
Function:	**0**
Offset:	**02h**
Device:	**16**
Function:	**0, 2**
Offset:	**02h**

Bit	Attr	Default	Description
15:0	RWO	*See Table 3-2	Device Identification Number Identifies each function of the MCH

3.8.1.3 RID - Revision Identification Register

This register contains the revision number of the MCH. The Revision ID (RID) is a traditional 8-bit Read Only (RO) register located at offset 08h in the standard PCI header of every PCI/PCI Express compatible device and function. Previously, a new value for RID was assigned for Intel chipsets for every stepping. There is a a need to provide an alternative value for software compatibility when a particular driver or patch unique to that stepping or an earlier stepping is required, for instance, to prevent Windows software from flagging differences in RID during device enumeration. The solution is to implement a mechanism to read one of two possible values from the RID register:

1. **Stepping Revision ID** (SRID): This is the default power on value for mask/metal steppings
2. **Compatible Revision ID** (CRID): The CRID functionality gives BIOS the flexibility to load OS drivers optimized for a previous revision of the silicon instead of the current revision of the silicon in order to reduce drivers updates and minimize changes to the OS image for minor optimizations to the silicon for yield improvement, or feature enhancement reasons that do not negatively impact the OS driver functionality.

Reading the RID in the Intel 5000X Chipset MCH returns either the SRID or CRID depending on the state of a register select flip-flop. Following reset, the register select flip flop is reset and the SRID is returned when the RID is read at offset 08h. The SRID value reflects the actual product stepping. To select the CRID value, BIOS/configuration software writes a key value of 79h to Bus 0, Device 0, Function 0 (ESI port) of the Intel 5000X Chipset MCH's RID register at offset 08h. This sets the SRID/CRID register select flip-flop and causes the CRID to be returned when the RID is read at offset 08h.

The RID register in the ESI port (Bus 0 device 0 Function 0) is a “write-once” sticky register and gets locked after the first write. This causes the CRID to be returned on all subsequent RID register reads. Software should read and save all device SRID values by reading Intel 5000X Chipset MCH device RID registers before setting the SRID/CRID register select flip flop.

The RID values for all devices and functions in Intel 5000X Chipset MCH are controlled by the SRID/CRID register select flip flop, thus writing the key value (79h) to the RID register in Bus 0, Device 0, Function 0 sets all Intel 5000X Chipset MCH device RID registers to return the CRID. Writing to the RID register of other devices has no effect on the SRID/CRID register select flip-flop. Only a power good reset can change the RID selection back to SRID.

Device: 0, 2-9, 17, 21, 22
Function: 0
Offset: 08h

Device: 16
Function: 0, 2
Offset: 08h

Bit	Attr	Default	Description
7:4	if (DEV 0) {RWOST} else {RO} endif	0h	**Major Revision** Steppings which require all masks to be regenerated[a]. 0000: A stepping for Intel 5000 Series Chipset with SF 0001: B stepping for Intel 5000 Series Chipset with SF 0010: C stepping for Intel 5000 Series Chipset with SF 1000: A stepping with Intel 5000 Series Chipset without SF 1001: B Stepping with Intel 5000 Series Chipset without SF 1010: C Stepping with Intel 5000 Series Chipset without SF Others: *Reserved*
3:0	if (DEV 0) {RWOST} else {RO} endif	0h	**Minor Revision** Incremented for each stepping which does not modify all masks. Reset for each major revision. 0x0: M0 stepping 0x1: M1 stepping 0x2: M2 stepping Others: *Reserved* Note: The Metal steppings indicated are a subset of the Major revision. For for example, an A stepping with M0 as minor revision typically means A0.

Notes:

a. Even though the contents of the RID have an attribute as “RO”, it is ultimately dictated by the comparator flop (attribute “RWOST” in Device 0, function 0) that selects between the CRID/SRID outputs. The comparator is set by BIOS/SW writing a specific value to offset 08h in dev0, fn 0 based on Figure 3-3.

3.8.1.3.1 Stepping Revision ID (SRID)

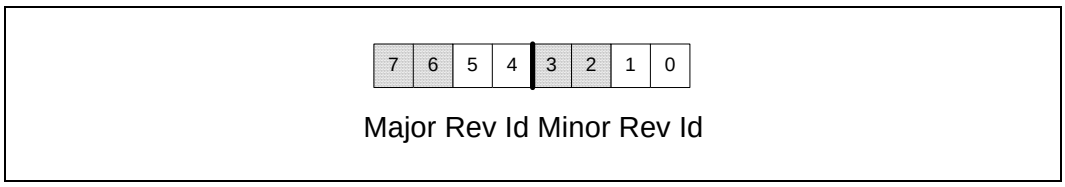
The SRID is a 4-bit hardwired value assigned by Intel, based on product’s stepping. The SRID is not a directly addressable PCI register. The SRID value is reflected through the RID register when appropriately addressed. The 4 bits of the SRID are reflected as the two least significant bits of the major and minor revision field respectively. See Figure 3-3.

3.8.1.3.2 Compatible Revision ID (CRID)

The CRID is an 4-bit hardwired value assigned by Intel during manufacturing process. Normally, the value assigned as the CRID will be identical to the SRID value of a previous stepping of the product with which the new product is deemed “compatible”.

The CRID is not a directly addressable PCI register. The CRID value is reflected through the RID register when appropriately addressed.The 4 bits of the CRID are reflected as the two least significant bits of the major and minor revision field respectively. See Figure 3-3.

Figure 3-3. Intel 5000X Chipset MCH implementation of SRID and CRID Registers



3.8.1.4 CCR - Class Code Register

This register contains the Class Code for the device. Writes to this register have no effect.

Device[a]:0, 2-7, 9, 17, 21, 22
Function: 0
Offset: 09h

Device: 16
Function: 0, 2
Offset: 09h

Bit	Attr	Default	Description
23:16	RO	06h	**Base Class**. This field indicates the general device category. For the MCH, this field is hardwired to 06h, indicating it is a “Bridge Device”.
15:8	RO	if (DEV2-7) {04h} else {00h}	**Sub-Class.** This field qualifies the Base Class, providing a more detailed specification of the device function. For PCI Express Devices 2,3,4,5,6,7 default is 040h, indicating “PCI to PCI Bridge” For all other Devices: 0,9,10,12,14,16,17,18,19 default is 00h, indicating “Host Bridge”. See footnote a, for DMA Engine device CCR.
7:0	RO	00h	**Register-Level Programming Interface.** This field identifies a specific programming interface (if any), that device independent software can use to interact with the device. There are no such interfaces defined for “Host Bridge” types, and this field is hardwired to 00h.

Notes:

a. The DMA Engine CCR for device 8 is defined separately in Section 3.11.3.

3.8.1.5 HDR - Header Type Register

This register identifies the header layout of the configuration space.

Device: 0, 2-9, 17, 21, 22
Function: 0
Offset: 0Eh

Device: 16
Function: 0, 2
Offset: 0Eh

Bit	Attr	Default	Description
7	RO	if (DEV16) {1h} else {0h} endif	**Multi-function Device.** Selects whether this is a multi-function device, that may have alternative configuration layouts. This bit is hardwired to '0' for devices for the MCH with the exception of device 16 fn 0-2, which it is set to '1'.
6:0	RO	if (DEV2-7) {01h} else {00h} endif	**Configuration Layout.** This field identifies the format of the configuration header layout for a PCI-to-PCI bridge from bytes 10h through 3Fh. For PCI Express Devices 2,3,4,5,6,7 default is 01h, indicating "PCI to PCI Bridge" For all other Devices: 0,8,9,16,17,21,22 default is 00h, indicating a conventional type 00h PCI header

3.8.1.6 SVID - Subsystem Vendor Identification Register

This register identifies the manufacturer of the system. This 16-bit register combined with the Device Identification Register uniquely identify any PCI device. They appear in every function except the PCI Express functions.

Device: 0, 8, 17, 21, 22
Function: 0
Offset: 2Ch

Device: 16
Function: 0, 2
Offset: 2Ch

Bit	Attr	Default	Description
15:0	RWO	8086h	**Vendor Identification Number.** The default value specifies Intel. Each byte of this register will be writable once. Second and successive writes to a byte will have no effect.

A write to any of the above registers on the MCH will write to all of them.

3.8.1.7 SID - Subsystem Identity

This register identifies the system. They appear in every function except the PCI Express functions.

Device:	**0, 8, 17, 21, 22**
Function:	**0**
Offset:	**2Eh**
Device:	**16**
Function:	**0, 2**
Offset:	**2Eh**

Bit	Attr	Default	Description
15:0	RWO	8086h	**Subsystem Identification Number:** The default value specifies Intel. Each byte of this register will be writable once. Second and successive writes to a byte will have no effect.

3.8.2 Address Mapping Registers

These registers control transaction routing to one of the three interface types (Memory, PCI Express, or ESI) based on transaction addresses. The memory mapping registers in this section are made read-only by the LT.LOCK-MEMCONFIG command. Routing to particular ports of a given interface type are defined by the following registers:

Table 3-29. Address Mapping Registers

Interface type	Address Routing registers
Memory	MIR, AMIR, PAM, SMRAM, EXSMRC, EXSMRAMC, TOLM, EXSMRTOP, AMBASE, AMR
PCI Express	MBASE/MLIM (devices 2-7) PMBASE/PMLIM (devices 2-7) PMBU/PMBL (devices 2-7) IOBASE/IOLIM (devices 2-7) SBUSN,SUBUSN (devices 2-7) BCTRL, HECBASE, PCICMD (devices 2-7)
ESI	Subtractive decode[a] (device 0)

Notes:

a. Any request not falling in the above ranges will be subtractively decoded and sent to Intel 631xESB/632xESB I/O Controller Hub via the ESI.

The MCH allows programmable memory attributes on 13 Legacy memory segments of various sizes in the 640 Kilobytes to 1 Megabytes address range. Seven Programmable Attribute Map (PAM) Registers are used to support these features.

Each PAM Register controls one or two regions, typically 16 Kilobytes in size

3.8.2.1 PAM0 - Programmable Attribute Map Register 0

This register controls the read, write, and shadowing attributes of the BIOS area which extends from 0F 0000h - 0F FFFFh.

Two bits are used to specify memory attributes for each memory segment. These bits apply to both host accesses and PCI initiator accesses to the PAM areas. These attributes are:

RE - Read Enable. When RE = 1, the CPU read accesses to the corresponding memory segment are claimed by the MCH and directed to main memory. Conversely, when RE = 0, the host read accesses are directed to ESI (Intel 631xESB/632xESB I/O Controller Hub) to be directed to the PCI bus.

WE - Write Enable. When WE = 1, the host write accesses to the corresponding memory segment are claimed by the MCH and directed to main memory. Conversely, when WE = 0, the host write accesses are directed to ESI (Intel 631xESB/632xESB I/O Controller Hub) to be directed to the PCI bus.

The RE and WE attributes permit a memory segment to be Read Only, Write Only, Read/Write, or disabled. For example, if a memory segment has RE = 1 and WE = 0, the segment is Read Only.

Device: 16
Function: 0
Offset: 59h

Bit	Attr	Default	Description
7:6	RV	00	Reserved
5:4	RW	00	**ESIENABLE0: 0F0000-0FFFFF Attribute Register** This field controls the steering of read and write cycles that address the BIOS area from 0F0000 to 0FFFFF. Bit5 = Write enable, Bit4 = Read enable. Encoding Description 00: DRAM Disabled - All accesses are directed to ESI 01: Read Only - All reads are serviced by DRAM. Writes are forwarded to ESI 10: Write Only - All writes are sent to DRAM. Reads are serviced by ESI 11: Normal DRAM Operation - All reads and writes are serviced by DRAM
3:0	RV	0h	Reserved

3.8.2.2 PAM1 - Programmable Attribute Map Register 1

This register controls the read, write, and shadowing attributes of the BIOS areas which extend from 0C 0000h-0C 7FFFh.

Device: 16
Function: 0
Offset: 5Ah

Bit	Attr	Default	Description
7:6	RV	00	Reserved
5:4	RW	00	**ESIENABLE1: 0C4000-0C7FFF Attribute Register** This field controls the steering of read and write cycles that address the BIOS area from 0C4000 to 0C7FFF Bit5 = Write enable, Bit4 = Read enable. Encoding Description 00: DRAM Disabled - All accesses are directed to ESI 01: Read Only - All reads are serviced by DRAM. Writes are forwarded to ESI 10: Write Only - All writes are sent to DRAM. Reads are serviced by ESI 11: Normal DRAM Operation - All reads and writes are serviced by DRAM
3:2	RV	00	Reserved
1:0	RW	00	**LOENABLE1: 0C0000-0C3FFF Attribute Register** This field controls the steering of read and write cycles that address the BIOS area from 0C0000 to 0C3FFF. Bit1 = Write enable, Bit0 = Read enable Encoding Description 00: DRAM Disabled - All accesses are directed to ESI 01: Read Only - All reads are serviced by DRAM. Writes are forwarded to ESI 10: Write Only - All writes are sent to DRAM. Reads are serviced by ESI 11: Normal DRAM Operation - All reads and writes are serviced by DRAM

3.8.2.3 PAM2 - Programmable Attribute Map Register 2

This register controls the read, write, and shadowing attributes of the BIOS areas which extend from 0C 8000h -0C FFFF h.

Device: 16
Function: 0
Offset: 5Bh

Bit	Attr	Default	Description
7:6	RV	00	Reserved
5:4	RW	00	**ESIENABLE2: 0CC000-0CFFFF Attribute Register** This field controls the steering of read and write cycles that address the BIOS area from 0CC000-0CFFFF. Bit5 = Write enable, Bit4 = Read enable. Encoding Description 00: DRAM Disabled - All accesses are directed to ESI 01: Read Only - All reads are serviced by DRAM. Writes are forwarded to ESI 10: Write Only - All writes are sent to DRAM. Reads are serviced by ESI 11: Normal DRAM Operation - All reads and writes are serviced by DRAM
3:2	RV	00	Reserved
1:0	RW	00	**LOENABLE2: 0C8000-0CBFFF Attribute Register** This field controls the steering of read and write cycles that address the BIOS area from 0C8000-0CBFFF. Bit1 = Write enable, Bit0 = Read enable Encoding Description 00: DRAM Disabled - All accesses are directed to ESI 01: Read Only - All reads are serviced by DRAM. Writes are forwarded to ESI 10: Write Only - All writes are sent to DRAM. Reads are serviced by ESI 11: Normal DRAM Operation - All reads and writes are serviced by DRAM

3.8.2.4 PAM3 - Programmable Attribute Map Register 3

This register controls the read, write, and shadowing attributes of the BIOS areas which extend from 0D 0000h - 0D 7FFFh.

Device: 16
Function: 0
Offset: 5Ch

Bit	Attr	Default	Description
7:6	RV	00	Reserved
5:4	RW	00	**ESIENABLE3: 0D 4000h - 0D 7FFFh Attribute Register** This field controls the steering of read and write cycles that address the BIOS area from 0D 4000h -0D 7FFFh. Bit5 = Write enable, Bit4 = Read enable. Encoding Description 00: DRAM Disabled - All accesses are directed to ESI 01: Read Only - All reads are serviced by DRAM. Writes are forwarded to ESI 10: Write Only - All writes are sent to DRAM. Reads are serviced by ESI 11: Normal DRAM Operation - All reads and writes are serviced by DRAM
3:2	RV	00	Reserved
1:0	RW	00	**LOENABLE3: 0D 0000h - 0D 3FFFh Attribute Register** This field controls the steering of read and write cycles that address the BIOS area from 0D 0000h -0D 3FFFh. Bit1 = Write enable, Bit0 = Read enable Encoding Description 00: DRAM Disabled - All accesses are directed to ESI 01: Read Only - All reads are serviced by DRAM. Writes are forwarded to ESI 10: Write Only - All writes are sent to DRAM. Reads are serviced by ESI 11: Normal DRAM Operation - All reads and writes are serviced by DRAM

3.8.2.5 PAM4 - Programmable Attribute Map Registers 4

This register controls the read, write, and shadowing attributes of the BIOS areas which extend from 0D 8000h - 0D FFFFh.

Device: 16
Function: 0
Offset: 5Dh

Bit	Attr	Default	Description
7:6	RV	00	Reserved
5:4	RW	00	**ESIENABLE4: 0DC000-0DFFFF Attribute Register** This field controls the steering of read and write cycles that address the BIOS area from 0DC000-0DFFFF. Bit5 = Write enable, Bit4 = Read enable. Encoding Description 00: DRAM Disabled - All accesses are directed to ESI 01: Read Only - All reads are serviced by DRAM. Writes are forwarded to ESI 10: Write Only - All writes are sent to DRAM. Reads are serviced by ESI 11: Normal DRAM Operation - All reads and writes are serviced by DRAM
3:2	RV	00	Reserved
1:0	RW	00	**LOENABLE4: 0D8000-0DBFFF Attribute Register** This field controls the steering of read and write cycles that address the BIOS area from 0D8000-0DBFFF. Bit1 = Write enable, Bit0 = Read enable Encoding Description 00: DRAM Disabled - All accesses are directed to ESI 01: Read Only - All reads are serviced by DRAM. Writes are forwarded to ESI 10: Write Only - All writes are sent to DRAM. Reads are serviced by ESI 11: Normal DRAM Operation - All reads and writes are serviced by DRAM

3.8.2.6 PAM5 - Programmable Attribute Map Register 5

This register controls the read, write, and shadowing attributes of the BIOS areas which extend from 0E 0000h -0E 7FFFh.

Device: 16
Function: 0
Offset: 5Eh

Bit	Attr	Default	Description
7:6	RV	00	Reserved
5:4	RW	00	**ESIENABLE5: 0E4000-0E7FFF Attribute Register** This field controls the steering of read and write cycles that address the BIOS area from 0E4000-0E7FFF. Bit5 = Write enable, Bit4 = Read enable. Encoding Description 00: DRAM Disabled - All accesses are directed to ESI 01: Read Only - All reads are serviced by DRAM. Writes are forwarded to ESI 10: Write Only - All writes are sent to DRAM. Reads are serviced by ESI 11: Normal DRAM Operation - All reads and writes are serviced by DRAM
3:2	RV	00	Reserved
1:0	RW	00	**LOENABLE5: 0E0000-0E3FFF Attribute Register** This field controls the steering of read and write cycles that address the BIOS area from 0E0000-0E3FFF. Bit1 = Write enable, Bit0 = Read enable Encoding Description 00: DRAM Disabled - All accesses are directed to ESI 01: Read Only - All reads are serviced by DRAM. Writes are forwarded to ESI 10: Write Only - All writes are sent to DRAM. Reads are serviced by ESI 11: Normal DRAM Operation - All reads and writes are serviced by DRAM

3.8.2.7 PAM6 - Programmable Attribute Map Register 6

This register controls the read, write, and shadowing attributes of the BIOS areas which extend from 0E 8000h -0E FFFFh.

Device: 16
Function: 0
Offset: 5Fh

Bit	Attr	Default	Description
7:6	RV	00	Reserved
5:4	RW	00	**ESIENABLE6: 0EC000-0DFFFF Attribute Register** This field controls the steering of read and write cycles that address the BIOS area from 0EC000-0DFFFF. Bit5 = Write enable, Bit4 = Read enable. Encoding Description 00: DRAM Disabled - All accesses are directed to ESI 01: Read Only - All reads are serviced by DRAM. Writes are forwarded to ESI 10: Write Only - All writes are sent to DRAM. Reads are serviced by ESI 11: Normal DRAM Operation - All reads and writes are serviced by DRAM
3:2	RV	00	Reserved
1:0	RW	00	**LOENABLE6: 0E8000-0EBFFF Attribute Register** This field controls the steering of read and write cycles that address the BIOS area from 0E8000-0EBFFF. Bit1 = Write enable, Bit0 = Read enable Encoding Description 00: DRAM Disabled - All accesses are directed to ESI 01: Read Only - All reads are serviced by DRAM. Writes are forwarded to ESI 10: Write Only - All writes are sent to DRAM. Reads are serviced by ESI 11: Normal DRAM Operation - All reads and writes are serviced by DRAM

3.8.2.8 SMRAMC - System Management RAM Control Register

The SMRAMC register controls how accesses to Compatible and Extended SMRAM spaces are treated. The Open, Close, and Lock bits function only when EXSMRC.G_SMRAME bit is set to a 1. Also, the OPEN bit must be reset before the LOCK bit is set.

Device: 16
Function: 0
Offset: 61h

Bit	Attr	Default	Description
7	RV	0	Reserved
6	RWL	0	**D_OPEN: SMM Space Open** When D_OPEN=1 and D_LCK=0, the SMM space DRAM is made visible even when SMM decode is not active. This is intended to help BIOS initialize SMM space. Software should ensure that D_OPEN=1 and D_CLS=1 are not set at the same time. This register can be locked by D_LCK.
5	RW	0	**D_CLS: SMM Space Closed** When D_CLS = 1 SMM space DRAM is not accessible to data references, even if SMM decode is active. Code references may still access SMM space DRAM. This will allow SMM software to reference through SMM space to update the display even when SMM is mapped over the VGA range. Software should ensure that D_OPEN=1 and D_CLS=1 are not set at the same time. Note that the D_CLS bit only applies to Compatible SMM space.
4	RWL	0	**D_LCK: SMM Space Locked** When D_LCK is set to 1 then D_OPEN is reset to 0 and D_LCK, D_OPEN, C_BASE_SEG, H_SMRAME, G_SMRAME, all LT.MSEG.BASE, all LT.MSEG.SIZE, ESMMTOP, TSEG_SZ and T_EN become read only. D_LCK can be set to 1 via a normal configuration space write but can only be cleared by a Full Reset. The combination of D_LCK and D_OPEN provide convenience with security. The BIOS can use the D_OPEN function to initialize SMM space and then use D_LCK to “lock down” SMM space in the future so that no application software (or BIOS itself) can violate the integrity of SMM space, even if the program has knowledge of the D_OPEN function.
3	RV	0	Reserved
2:0	RO	010	**C_BASE_SEG: Compatible SMM Space Base Segment** This field indicates the location of legacy SMM space. SMM DRAM is not remapped. It is simply made visible if the conditions are right to access SMM space, otherwise the access is forwarded to ESI/VGA. Since the MCH supports only the SMM space between A 0000h and B FFFFh, this field is hardwired to 010.

3.8.2.9 EXSMRC - Extended System Management RAM Control Register

The Extended SMRAM register controls the configuration of Extended SMRAM space. The Extended SMRAM (E_SMRAM) memory provides a write-back cacheable SMRAM memory space that is above 1 MByte.

Device: 16
Function: 0
Offset: 62h

Bit	Attr	Default	Description
7	RWL	0	**H_SMRAME: Enable High SMRAM** Controls the SMM memory space location (that is, above 1 MByte or below 1 MByte) When G_SMRAME is 1 and H_SMRAME is set to 1, the high SMRAM memory space is enabled. SMRAM accesses within the range FEDA_0000h to FEDB FFFFh are remapped to DRAM addresses within the range 000A_0000h to 000B_FFFFh. Once D_LCK has been set, this bit becomes read only.
6	RO	0	**MDAP: MDA Present** Since the MCH does not support MDA, this bit has no meaning.
5	RV	0	**Reserved**
4	RV	0	Reserved
3	RWL	0	**G_SMRAME: Global SMRAM Enable** If set to a 1, then Compatible SMRAM functions are enabled, providing 128 KB of DRAM accessible at the A0000h address while in SMM (ADS# with SMM decode). To enable Extended SMRAM function this bit has be set to 1. Refer to the section on SMM for more details. Once D_LCK is set, this bit becomes read only. (Moved from SMRAM bit3)
2:1	RWL	00	**TSEG_SZ: TSEG Size** Selects the size of the TSEG memory block if enabled. Memory from (ESMMTOP - TSEG_SZ) to ESMMTOP - 1 is partitioned away so that it may only be accessed by the processor interface and only then when the SMM bit (SMMEM#) is set in the request packet. Non-SMM accesses to this memory region are sent to ESI when the TSEG memory block is enabled. Note that once D_LCK is set, these bits become read only. 00: 512 kB 01: 1 MB 10: 2 MB 11: 4 MB
0	RWL	0	**T_EN: TSEG Enable** Enabling of SMRAM memory for Extended SMRAM space only. When G_SMRAME =1 and TSEG_EN = 1, the TSEG is enabled to appear in the appropriate physical address space. Note that once D_LCK is set, this bit becomes read only.

3.8.2.10 EXSMRTOP - Extended System Management RAM Top Register

This register defines the location of the Extended (TSEG) SMM range by defining the top of the TSEG SMM range (ESMMTOP).

Device: 16
Function: 0
Offset: 63h

Bit	Attr	Default	Description
7:4	RV	0h	Reserved

Device: 16 Function: 0 Offset: 63h			
Bit	**Attr**	**Default**	**Description**
3:0	RWL	1h	**ESMMTOP: Top of Extended SMM Space (TSEG)** This field contains the address that corresponds to address bits 31 to 28. This field points to the top (+1) of extended SMM space below 4 GB. Addresses below 4GB (A[39:32] must be 0) that fall in this range are decoded to be in the extended SMM space and should be routed according to Section 4.3.3: ESMMTOP-TSEG_SZ <= Address < ESMMTOP TSEG_SZ can be 512KB, 1MB, 2MB, or 4MB, depending on the value of EXSMRC.TSEG_SZ. ESMMTOP is relocatable to accommodate software that wishes to configure the TSEG SMM space before MMIO space is known. This field defaults to point to the same address as TOLM. Note that ESMMTOP cannot be greater than TOLM otherwise the chipset will not function deterministically. Note that once D_LCK is set, this field becomes read only.

3.8.2.11 EXSMRAMC - Expansion System Management RAM Control Register

Device: 16 Function: 2 Offset: 60h			
Bit	**Attr**	**Default**	**Description**
7	RWC	0	**E_SMERR: Invalid SMRAM Access** This bit is set when CPU has accessed the defined memory ranges in High SMM Memory and Extended SMRAM (T-segment) while not in SMM space and with the D-OPEN bit = 0. The MCH will set this bit if any In-Bound access from I/O device targeting SMM range that gets routed to the ESI port (master abort). Refer to Section 4.4.3 for details. The MCH will not set this bit when processor does a cache line eviction (EWB or IWB) to SMM ranges regardless of SMMEM# on FSB. It is software's responsibility to clear this bit. The software must write a 1 to this bit to clear it.
6:0	RV	0h	Reserved

Other address mapping registers such as BCTRL (VGAEN), MBASE/LIMIT, PMBASE/LIMIT, and so forth, are included with the PCI Express registers described in this chapter.

3.8.2.12 HECBASE - PCI Express Extended Configuration Base Address Register

This register defines the base address of the enhanced PCI Express configuration memory.

Device: 16 Function: 0 Offset: 64h			
Bit	**Attr**	**Default**	**Description**
31:24	RV	0h	Reserved
23:12	RW	001h	**HECBASE: PCI Express Extended Configuration Base** This register contains the address that corresponds to bits 39 to 28 of the base address for PCI Express extended configuration space. Configuration software will read this register to determine where the 256 MB range of addresses resides for this particular host bridge. This register defaults to the same address as the default value for TOLM.

Device: 16 Function: 0 Offset: 64h			
Bit	**Attr**	**Default**	**Description**
11:0	RV	0h	Reserved

3.8.3 AMB Memory Mapped Registers

The MCH supports four FB-DIMM channels. The MCH supports up to 16 FB-DIMM (each with its Advanced Memory Buffer (AMB)) on four channels. Software needs to program AMBPRESENT for each AMB on the platform. There are up to eight functions per AMB component with 256 B of register space per function.

The MCH supports memory mapped register regions for software to access individual AMB configuration registers. Memory mapped access to AMB register regions are converted by the MCH to FB-DIMM channel command encodings subject to AMBPRESENT register settings (see Section 3.9.24.11). This region is relocatable by programming the AMBASE register. Software is required to program the AMR for the size of AMB register regions. The size of this region is 128KB. It is mapped to each AMB addressing slot in 2 KB blocks. If the corresponding AMBPRESENT bit is not set, then MCH will not send configuration transaction to that AMB addressing slot.

To support SMBus and JTAG access using traditional PCI configuration mechanism, MCH provides a “switching window” using a dedicated PCI device/function and AMBSELECT register. AMBSELECT register can be programmed to select an AMB. Bus 0, device 9, function 0 is mapped to the selected AMB’s configuration registers.

Access to bus 0, device 9, function 0 is limited to SMBus and JTAG only, FSB access to this function will be mastered aborted by MCH as non-existent PCI function.
AMB register spaces are accessible through the SMBus by the programming of the AMBSELECT Function_Select field. This field is used select one of the AMB 8 register spaces.

3.8.3.1 AMBASE: AMB Memory Mapped Register Region Base Register

Device: 16 Function: 0 Offset: 48h			
Bit	**Attr**	**Default**	**Description**
63:40	RV	0h	**Reserved**
39:17	RW	007F00h	**AMBASE:** This marks the 128 KB memory-mapped registers region used for accessing AMB registers. It can be placed as MMIO region within the physical limits of the system. Since the MCH uses only 40-bit addressable space, hence only bits 39:17 are valid. The default base address is at: 0xFE00_0000. This field could be relocated by software.
16:0	RV	0h	Reserved

3.8.3.2 AMR - AMB Memory Mapped Registers Region Range Register

Device: 16
Function: 0
Offset: 50h

Bit	Attr	Default	Description
31:0	RW	0002_0000h	**AMBASE_Region_Size:** The size of AMB memory mapped register region in bytes. For MCH, the value is 128 KB: 2 KB per AMB for a total of 16 AMB per channel, 32 KB per FB-DIMM channel for a total of four channels.

3.8.3.3 AMBSELECT - AMB Switching Window Select Register

Device: 16
Function: 0
Offset: 54h

Bit	Attr	Default	Description
15:9	RV	0h	Reserved
8:7	RW	0h	**Channel_Select:** Specify the FB-DIMM channel being accessed via bus 0, device 9, function 0 for SM Bus and JTAG only.
6:3	RW	0h	**AMB_Select:** Specify the AMB slot being accessed via bus 0, device 9, function 0 for SM Bus and JTAG only.
2:0	RW	0h	**Function_Select:** Specify the function being accessed via bus 0, device 9, function 0 for SM Bus and JTAG only.

3.8.3.4 MAXCH - Maximum Channel Number Register

Device: 16
Function: 0
Offset: 56h

Bit	Attr	Default	Description
7:0	RO	04h	**Maximum_number_channels:** Set by hardware to indicate the maximum number of FB-DIMM channels that MCH supports.

3.8.3.5 MAXDIMMPERCH - Maximum DIMM PER Channel Number Register

This register controls the maximum number of AMB DIMMs per FB-DIMM channel that MCH supports for AMB configuration register access. This register applies only to DIMM modules in the FB-DIMM channel, that is, those AMB with DS[3:0] encoding from 0h to 7h.

Device: 16
Function: 0
Offset: 57h

Bit	Attr	Default	Description
7:0	RO	04h	**Maximum_number_DIMM_per_channel:** Set by hardware to indicate the maximum number of FB-DIMM AMBs per channel that the MCH supports.

3.8.3.6 Map to AMB Registers

In Table 3-30, each 2 KB range is mapped to individual AMB registers by address translation of MCH. The address of this relocatable register area is specified in the AMBASE register. Configuration transactions targeting these ranges are converted to FB-DIMM commands by the MCH and sent to the FB-DIMM channel subject to AMBPRESENT register settings.

The AMB register's PCI function (3 bits) and offset (8 bits) are used as the offset (11 bits) from the base of each 2 KB range for the specific AMB register space.

Table 3-30. Register Offsets in AMB Memory Mapped Registers Region (Sheet 1 of 2)

7FFh-0h	map to channel_0, AMB_0 registers
FFFh-800h	map to channel_0, AMB_1 registers
17FFh-1000h	map to channel_0, AMB_2 registers
1FFFh-1800h	map to channel_0, AMB_3 registers
27FFh-2000h	map to channel_0, AMB_4 registers
2FFFh-2800h	map to channel_0, AMB_5 registers
37FFh-3000h	map to channel_0, AMB_6 registers
3FFFh-3800h	map to channel_0, AMB_7 registers
47FFh-4000h	map to channel_0, AMB_8 registers
4FFFh-4800h	map to channel_0, AMB_9 registers
57FFh-5000h	map to channel_0, AMB_A registers
5FFFh-5800h	map to channel_0, AMB_B registers
67FFh-6000h	map to channel_0, AMB_C registers
6FFFh-6800h	map to channel_0, AMB_D registers
77FFh-7000h	map to channel_0, AMB_E registers
7FFFh-7800h	map to channel_0, AMB_F registers
87FFh-8000h	map to channel_1, AMB_0 registers
8FFFh-8800h	map to channel_1, AMB_1 registers
97FFh-9000h	map to channel_1, AMB_2 registers
9FFFh-9800h	map to channel_1, AMB_3 registers
A7FFh-A000h	map to channel_1, AMB_4 registers
AFFFh-A800h	map to channel_1, AMB_5 registers
B7FFh-B000h	map to channel_1, AMB_6 registers
BFFFh-B800h	map to channel_1, AMB_7 registers
C7FFh-C000h	map to channel_1, AMB_8 registers
CFFFh-C800h	map to channel_1, AMB_9 registers
D7FFh-D000h	map to channel_1, AMB_A registers
DFFFh-D800h	map to channel_1, AMB_B registers
E7FFh-E000h	map to channel_1, AMB_C registers
EFFFh-E800h	map to channel_1, AMB_D registers
F7FFh-F000h	map to channel_1, AMB_E registers
FFFFh-F800h	map to channel_1, AMB_F registers
107FFh-10000h	map to channel_2, AMB_0 registers
10FFFh-10800h	map to channel_2, AMB_1 registers
117FFh-11000h	map to channel_2, AMB_2 registers
11FFFh-11800h	map to channel_2, AMB_3 registers
127FFh-12000h	map to channel_2, AMB_4 registers
12FFFh-12800h	map to channel_2, AMB_5 registers
137FFh-13000h	map to channel_2, AMB_6 registers
13FFFh-13800h	map to channel_2, AMB_7 registers

Table 3-30. Register Offsets in AMB Memory Mapped Registers Region (Sheet 2 of 2)

147FFh-14000h	map to channel_2, AMB_8 registers
14FFFh-14800h	map to channel_2, AMB_9 registers
157FFh-15000h	map to channel_2, AMB_A registers
15FFFh-15800h	map to channel_2, AMB_B registers
167FFh-16000h	map to channel_2, AMB_C registers
16FFFh-16800h	map to channel_2, AMB_D registers
177FFh-17000h	map to channel_2, AMB_E registers
17FFFh-17800h	map to channel_2, AMB_F registers
187FFh-18000h	map to channel_3, AMB_0 registers
18FFFh-18800h	map to channel_3, AMB_1 registers
197FFh-19000h	map to channel_3, AMB_2 registers
19FFFh-19800h	map to channel_3, AMB_3 registers
1A7FFh-1A000h	map to channel_3, AMB_4 registers
1AFFFh-1A800h	map to channel_3, AMB_5 registers
1B7FFh-1B000h	map to channel_3, AMB_6 registers
1BFFFh-1B800h	map to channel_3, AMB_7 registers
1C7FFh-1C000h	map to channel_3, AMB_8 registers
1CFFFh-1C800h	map to channel_3, AMB_9 registers
1D7FFh-1D000h	map to channel_3, AMB_A registers
1DFFFh-1D800h	map to channel_3, AMB_B registers
1E7FFh-1E000h	map to channel_3, AMB_C registers
1EFFFh-1E800h	map to channel_3, AMB_D registers
1F7FFh-1F000h	map to channel_3, AMB_E registers
1FFFFh-1F800h	map to channel_3, AMB_F registers

3.8.4 Interrupt Redirection Registers

3.8.4.1 REDIRCTL - Redirection Control Register

This register controls the priority algorithm of the XTPR interrupt redirection mechanism.

Device: 16
Function: 0
Offset: 6Eh

Bit	Attr	Default	Description
15:12	RV	0	**Reserved**
11:8	RW	0h	**BUCKET2: First priority number not in BUCKET0, BUCKET1, or BUCKET2.** Must be programmed with a larger value than BUCKET1. A suggested value is Ch.
7:4	RW	0h	**BUCKET1: First priority number not in BUCKET0 or BUCKET1.** Must be programmed with a larger value than **BUCKET0.** A suggested value is 8h.
3:0	RW	0h	**BUCKET0:** First priority number not in **BUCKET0**. A suggested value is 0h.

3.8.4.2 REDIRBUCKETS - Redirection Bucket Number Register

This register allows software to read the current hardware bucket number assigned to each XTPR register.

Device: 16 Function: 0 Offset: 68h			
Bit	**Attr**	**Default**	**Description**
31:30	RO	0	**BUCKET15:** Redirection bucket number for XTPR[15].
29:28	RO	0	**BUCKET14:** Redirection bucket number for XTPR[14].
27:26	RO	0	**BUCKET13:** Redirection bucket number for XTPR[13].
25:24	RO	0	**BUCKET12:** Redirection bucket number for XTPR[12].
23:22	RO	0	**BUCKET11:** Redirection bucket number for XTPR[11].
21:20	RO	0	**BUCKET10:** Redirection bucket number for XTPR[10].
19:18	RO	0	**BUCKET9:** Redirection bucket number for XTPR[9].
17:16	RO	0	**BUCKET8:** Redirection bucket number for XTPR[8].
15:14	RO	0	**BUCKET7:** Redirection bucket number for XTPR[7].
13:12	RO	0	**BUCKET6:** Redirection bucket number for XTPR[6].
11:10	RO	0	**BUCKET5:** Redirection bucket number for XTPR[5].
9:8	RO	0	**BUCKET4:** Redirection bucket number for XTPR[4].
7:6	RO	0	**BUCKET3:** Redirection bucket number for XTPR[3].
5:4	RO	0	**BUCKET2:** Redirection bucket number for XTPR[2].
3:2	RO	0	**BUCKET1:** Redirection bucket number for XTPR[1].
1:0	RO	0	**BUCKET0:** Redirection bucket number for XTPR[0].

3.8.5 Boot and Reset Registers

3.8.5.1 SYRE - System Reset Register

This register controls MCH reset behavior. Any resets produced by a write to this register must be delayed until the configuration write is completed on the initiating interface (PCI Express, ESI, processor bus, SMBus, JTAG).

There is no “SOFT RESET” bit in this register. That function is invoked through the ESI. There are no CORE:FBD gear ratio definitions in this register. Those are located in the DDRFRQ register.

Device: 16 Function: 0 Offset: 40h			
Bit	**Attr**	**Default**	**Description**
15	RW	0	**SAVCFG: Preserve Configuration** When this bit is set, MCH configuration register contents (except for this bit) are not cleared by hard reset. As this bit is cleared by reset, software must set it after each reset if this behavior is desired for the next reset. If this bit is set, BOFL will not be cleared by reset. Software should use the Boot Flag Reset bit to re-enable the BOFL mechanism.

Device: 16 Function: 0 Offset: 40h			
Bit	**Attr**	**Default**	**Description**
14	RW	0	**CPURST: Processor Reset** If set, the MCH will assert processor RESET# on both buses as soon as the MCH has no pending transactions. The chipset will then deassert RESET# following the timing rules described in the Reset Chapter. The MCH does not have any mechanism to drain transactions before effecting the CPU RESET#. It is the responsibility of software to ensure that the system is quiet before sending the configuration write (last command) to set this field in the MCH in order to drive the CPU RESET# signal. Any violation of this usage pattern would render the system unstable and potentially catastrophic.
13	RV	0	**CPUBIST: Processor Built-In-Self-Test** If set, A[3]# is asserted during Power-On-Configuration (POC), and the processor will run BIST before engaging processor bus protocol.
12:11	RV	0	Reserved1
10	ROST	0	**S3: S3 Sleep State** The MCH sets this bit when it sends an Ack-S3 message to the ESI port. The MCH clears this bit after it has placed appropriate FB-DIMM channels into the FB-DIMM.Calibrate state in response to deassertion of the RESETI# signal.
9	RW	0	**ROR: Processor Reset on Refresh** If set, the MCH will assert processor RESET# on both busses when a refresh cycle completes.
8	RWST	0	**BNR_INDP_BINIT_MODE: BNR independent of BINIT Mode** 0: The chipset associates BNR with BINIT and for CPUs that do NOT follow the “BNR independent of BINIT” feature set. 1: Enables the chipset to use the “BNR independent of BINIT” feature set. i.e no dependency is required between BNR and BINIT. Refer to the BNR#, BINIT# sampling rules in the Intel® Pentium® 4 and Intel® Xeon® Processor External Hardware Specification, Rev 2.5, Ref#14035
7:0	RV	0h	Reserved

3.8.5.2 CPURSTCAPTMR: CPU Reset Done Cap Latency Timer

This register implements the cap latency method for the CPU_RST_DONE/ CPU_RST_DONE_ACK using a 12-bit variable timer.

Device: 16 Function: 0 Offset: 42h			
Bit	**Attr**	**Default**	**Description**
15:12	RV	0h	***Reserved***
11:0	RWST	7FFh	**DCRT: ESI CPU Reset Done Ack Determinism Timer** This field provides the determinism timer threshold for the Intel 5000X Chipset MCH for handling the CPU_RESET_DONE/CPU_RESET_DONE_ACK message before deasserting the CPU_RESET#. It uses this 12-bit counter to schedule the CPU_RESET_DONE message on the DMI and then waits for the CPU_RESET_DONE_ACK message to come back and waits for the timer expiry before deasserting CPU_RESET#. Cap_latency = Max(CPU_RST_DONE_ACK_round trip_latency, DCRT). It is expected that the DCRT field is set larger than the expected round trip latency. This provides the necessary leeway for absorbing clock synchronization, jitter, deskew and other variations that will affect the determinism on the DMI port. Hence the data is always sent back only after the expiry of the DCRT field at the heartbeat boundary. It is sticky through reset to permit to allow different types of BIOS flows that may require a hard reset of the Intel 5000X Chipset MCH. Maximum value is 4095 core clocks A default of 2047 clocks (7FFh) is used.

3.8.5.3 POC - Power-On Configuration Register

Contrary to its name, this register defines configuration values driven at reset. At power-on, no bits in this register are active as PWRGOOD clears them all. This register only activates configuration on subsequent resets.

The MCH drives the contents of this register on A[35:4]# whenever it asserts processor RESET#. These values are driven during processor RESET# assertion, and for two host clocks past the trailing edge of processor RESET#.

This register is sticky through reset; that is, the contents of the register remain unchanged during and following a Hard Reset. This allows system configuration software to modify the default values and reset the system to pass those values to all host bus devices.

The POC bits do not affect MCH operation except for driving A[35:4]#.

Read after write to POC register will read updated value but the architectural behavior will not be affected until hard-reset deassertion. A warm reset (CPU reset) will not cause the contents of the POC register to be altered.

There are other power-on configuration bits in the SYRE register.

Device: 16 Function: 0 Offset: 44h			
Bit	**Attr**	**Default**	**Description**
31:28	RV	0h	**Reserved**
27	RWST	0	**MTDIS: Disable Multi-Threading** If set, A[31]# is asserted, and the processor will disable Multi-threading.
26:12	RV	0h	**Reserved**
11	RWST	1	**BUSPARK: Request Bus Parking Disable** If set, A[15]# is asserted and the processor may not park on the system bus. Default is to disable busparking

Device: 16 Function: 0 Offset: 44h			
Bit	**Attr**	**Default**	**Description**
10:0	RV	0h	**Reserved**

3.8.5.4 SPAD[3:0] - Scratch Pad Registers

These scratch pad registers each provide 32 read/writable bits that can be used by software. They are also aliased to fixed memory addresses.

Device: 16 Function: 0 Offset: DCh, D8h, D4h, D0h			
Bit	**Attr**	**Default**	**Description**
31:0	RW	00000000h	Scratch Pad value. These bits have no effect on the hardware.

3.8.5.5 SPADS[3:0] - Sticky Scratch Pad

These sticky scratch pad registers each provide 32 read/writable bits that can be used by software. They are also aliased to fixed memory addresses.

Device: 16 Function: 0 Offset: ECh, E8h, E4h, E0h			
Bit	**Attr**	**Default**	**Description**
31:0	RWST	00000000h	Scratch Pad value. These sticky bits have no effect on the hardware.

3.8.5.6 BOFL[3:0] - Boot Flag Register

These registers can be used to select the system boot strap processor or for other cross processor communication purposes. When this register is read, the contents of the register is cleared. Therefore, a processor that reads a non-zero value owns the semaphore. Any value can be written to this register at any time.

An example of usage would be for all processors to read the register. The first one that gets a non-zero value owns the semaphore. Since the read clears the value of the register, all other processors will see a zero value and will spin until they receive further notification. After the winning processor is done, it writes a non-zero value of its choice into the register, arming it for subsequent uses. These registers are also aliased to fixed memory I/O addresses.

Device: 16 Function: 0 Offset: C0h, C4h, C8h, CCh			
Bit	**Attr**	**Default**	**Description**
31:0	RCW	A5A5A5A5h	**SemaVal: Semaphore Value** Can be written to any value. Value is cleared when there is a read.

3.8.6 Control and Interrupt Registers

3.8.6.1 PROCENABLE: Processor Enable Global Control

The two FSBEN bits are used to enable or disable frontside bus arbitration. When frontside bus arbitration is disabled the processor is effectively disabled.

Device: 16
Function: 0
Offset: F0h

Bit	Attr	Default	Description
31:5	RV	3fAh	*Reserved.*
4:3	RWST	11	**FSBEN: FSB1 and FSB0 Enable** The field is defined as the following: 00: reserved 01: FSB1 is disabled. FSB0 is enabled. 10: FSB1 is enabled. FSB0 is disabled. 11: FSB1 is enabled. FSB0 is enabled. (default) Hard-reset is needed after changing value in this register.
2	RWST	0	**SFBYPASS: Snoop Filter Bypass** **0: SF is enabled** **1: SF is disabled** Note: The output of the fuse “SF CHOP” is gated appropriately with this register field viz. SFBYPASS for further internal decoding by Intel 5000X Chipset MCH. The fuse has overriding effect.
1:0	RV	0h	*Reserved.*

3.8.6.2 FSBS[1:0] - Processor Bus Status Register

This register holds status from the Processor Busses.

Device: 16
Function: 0
Offset: 7Ch, 74h

Bit	Attr	Default	Description
31:2	RV	0h	*Reserved.*
1	RO	0	**2SOCKET: 2 Sockets present on this FSB** Set when Intel 5000X Chipset MCH has seen Ab[22] asserted, indicating there are more than 1 processors present on this FSB.
0	RO	0	**2CORE: 2 Cores present** Set when Intel 5000X Chipset MCH has seen Ab[30] asserted, indicating there is more than 1 core in a processor. Note: Mixing single core with dual-core processors will be recognized as dual-core processor on this FSB.

3.8.6.3 XTPR[7:0] - External Task Priority Register

These registers control redirectable interrupt priority for xAPIC agents connected to the MCH. Up to four agents on each bus are supported. These agents may be two dual core processors each with two threads or four single core processors. The xAPIC architecture provides for lowest priority delivery through interrupt redirection by the MCH. If the redirectable “hint bit” is set in the xAPIC message, the chipset may redirect the interrupt to another agent. Redirection of interrupts can be applied to both I/O interrupts and IPIs.

Each register contains the following fields:

1. Agent priority (Task Priority)
2. APIC enable bit (TPR Enable)
3. Logical APIC ID (LOGID)
4. Processor physical APIC ID (PHYSID)

The XTPR registers are modified by a front side bus xTPR_Update transaction. In addition, the XTPR registers can be modified by software.

Table 3-31. XTPR Index

Index	Value
3	0 for FSB0, 1 for FSB1
2	Ab[29]
1	Ab[30] OR Ab[22]
0	Ab[21]

These registers are used for lowest priority delivery through interrupt redirection by the chipset.

Device: 16
Function: 0
Offset: BCh, B8h, B4h, B0h, ACh, A8h, A4h, A0h, 9Ch, 98h, 94h, 90h, 8Ch, 88h, 84h, 80h

Bit	Attr	Default	Description
31	if (XTPR0) {RW} else {RV} endif	0	**CLUSTER: Global Cluster Mode (XTPR[0] only)** Used in interrupt redirection for lowest priority delivery. Updated by every xTPR_Update transaction on either bus (Aa[3]). 0: flat Note: Cluster Mode not Supported
30:24	RV	00h	*Reserved.*
23	RW	0	**TPREN: TPR Enable** This bit reflects the value of Ab[31]#. When Ab[31]# is asserted, the value of this bit will be 0.
22:20	RV	0h	*Reserved.*
19:16	RW	0h	**PRIORITY: Task Priority** The processor with the lowest enabled value will be assigned the redirectable interrupt. This field is updated with Ab[27:24] of the xTPR_Update transaction.
15:8	RW	0h	**PHYSID: Physical APIC ID** The physical ID of the APIC agent associated with the XTPR entry. This field is updated with Aa[19:12] of the xTPR_Update transaction.
7:0	RW	0h	**LOGID: Logical APIC ID** The logical ID of the APIC agent associated with the XTPR entry. This field is updated with Aa[11:4] of the xTPR_Update transaction.

3.8.7 PCI Express Device Configuration Registers

This section describes the registers associated with the PCI Express Interface.

The PCI Express register structure is exposed to the operating system and requires a separate device per port. Ports 2-7 will be assigned devices 2 through 7 while Port 0 is the ESI interconnect to the Intel 631xESB/632xESB I/O Controller Hub. The PCI Express ports determine at reset the maximum width of the devices to which they are connected through link training. All ports will be made visible to OS even if

unconnected. If Ports are combined to form larger widths (for example, x8 or x16 from a x4 link), then the unused ports will Master Abort (reads return all ones, writes dropped) any accesses to it. Note that configuration accesses to the unconnected port will still be allowed to permit device remapping, Hot-Plug and so forth.

Table 3-32. When will a Intel 5000X Chipset PCI Express* Device be Accessible?

PCI Express Port	Device	x16	Registers may be accessed if:
7	7	High Performance Graphics Port	
6	6		
5	5		
4	4		Port 4 is connected to x16 device
3	3	Possible Combination	Port3 is connected to a 4x device
2	2		Port2 is connected to a x4 or x8 device
0	0	ESI - Not combinable	Port0 is connected to a x4 ESB2 port through ESI and cannot be combined with any other port

Figure 3-4 illustrates how each PCI Express port's configuration space appears to software. Each PCI Express port's configuration space has four regions:

- **Standard PCI Header** - This region closely resembles a standard PCI-to-PCI bridge header.
- **PCI Device Dependent Region** - The region is also part of standard PCI configuration space and contains the PCI capability structures. For the Intel 5000X Chipset MCH, the supported capabilities are:
  - Message Signalled Interrupts
  - Hot-Plug
  - PCI Express Capability
- **PCI Express Extended Configuration Space** - This space is an enhancement beyond standard PCI and only accessible with PCI Express aware software. The MCH supports the Enhanced Error Signalling capability.
- **Capability Working Register Sets** - These ranges are indirectly accessed through Data and Select registers in the capability structures. For the MCH, working register sets exist for the Standard Hot-Plug Controller and Power Management capabilities.

Figure 3-4. PCI Express Configuration Space

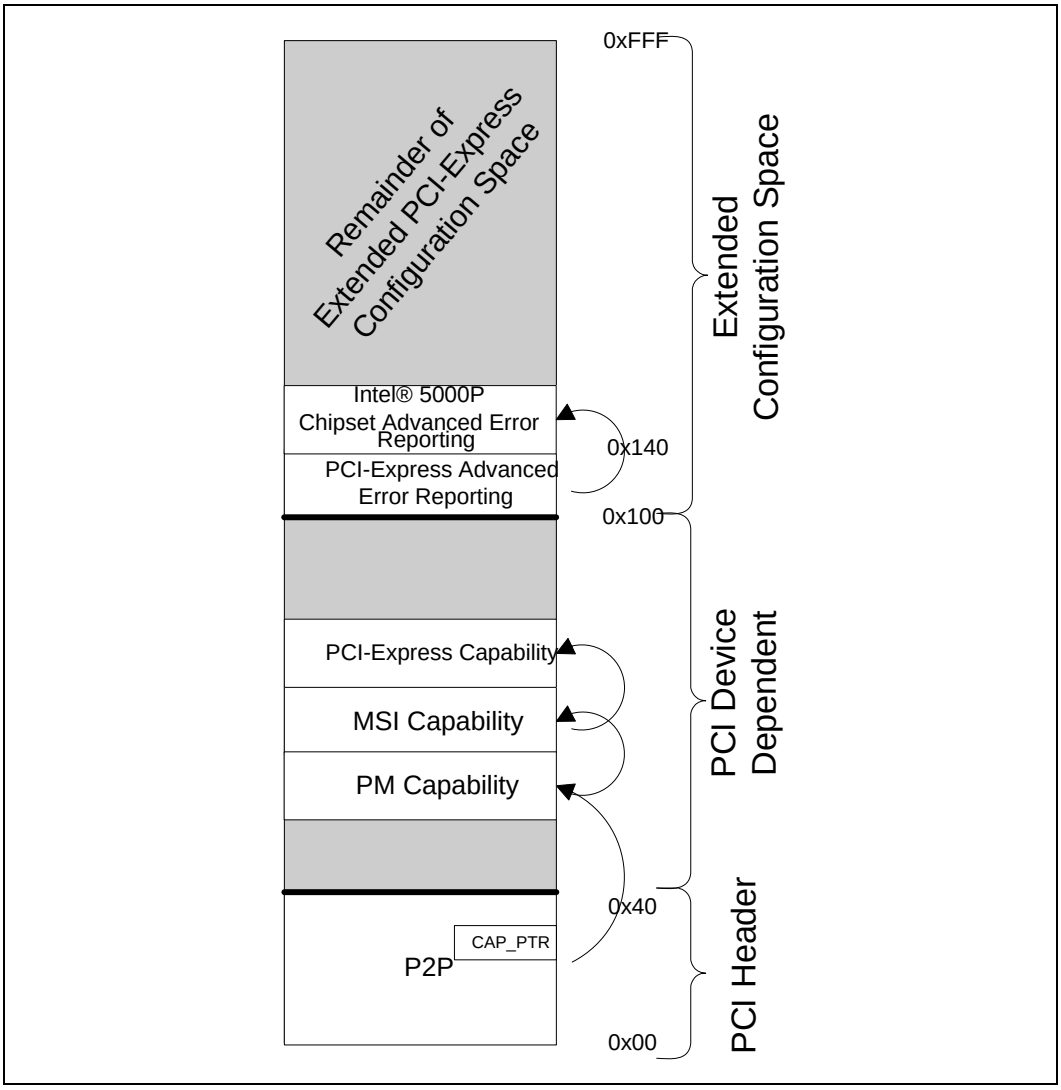


Figure 3-3 shows the configuration register offset addresses for each of the PCI Express ports as defined in the *PCI Express Base Specification*, Revision 1.0a. It is also compatible with the standard PCI 2.3 capability structure and comprises of a linked list where each capability has a pointer to the next capability in the list. For PCI Express extended capabilities, the first structure is required to start at 0x100 offset.

3.8.8 PCI Express Header

The following registers define the standard PCI 2.3 compatible and extended PCI Express configuration space for each of the PCI Express x4 links in the MCH. Unless otherwise specified, the registers are enumerated as a vector [2:7] mapping to each of the six PCI Express ports uniquely while the ESI port is referred by index 0.

3.8.8.1 PCICMD[7:2, 0]- Command Register

This register defines the PCI 2.3 compatible command register values applicable to PCI Express space.

Device: 0, 2-7
Function: 0
Offset: 04h

Bit	Attr	Default	Description
15:11	RV	0h	*Reserved.* (by PCI SIG)
10	RW	0	**INTxDisable: Interrupt Disable** Controls the ability of the PCI Express port to generate INTx messages. This bit does not affect the ability of the Intel 5000X Chipset to route interrupt messages received at the PCI Express port. However, this bit controls the generation of legacy interrupts to the DMI for PCI Express errors detected internally in this port (for example, Malformed TLP, CRC error, completion time out, and so forth) or when receiving root port error messages or interrupts due to HP/PM events generated in legacy mode within the Intel 5000X Chipset MCH. Refer to the INTP register in Section 3.8.8.27, "INTP[7:2,0] - Interrupt Pin Register" on page 111 for interrupt routing to DMI. 1: Legacy Interrupt mode is disabled 0: Legacy Interrupt mode is enabled
9	RO	0	**FB2B: Fast Back-to-Back Enable** Not applicable to PCI Express and is hardwired to 0
8	RW	0	**SERRE: SERR Message Enable** his field handles the reporting of fatal and non-fatal errors by enabling the error pins ERR[2:0]. 1: The Intel 5000X Chipset is enabled to send fatal/non-fatal errors. 0: The Intel 5000X Chipset is disabled from generating fatal/non-fatal errors. The errors are also enabled by the PEXDEVCTRL register in Section 3.8.11.4 In addition, for Type 1 configuration space header devices, for example, Virtual P2P bridge), this bit, when set, enables transmission of ERR_NONFATAL and ERR_FATAL error messages[a] forwarded from the PCI Express interface. This bit does not affect the transmission of forwarded ERR_COR messages. Refer to the Intel 5000X Chipset MCH RAS Error Model.
7	RO	0	IDSELWCC: IDSEL Stepping/Wait Cycle Control Not applicable to PCI Express. Hardwired to 0.
6	RW	0	**PERRE: Parity Error Response Enable** When set, this field enables parity checking.
5	RO	0	**VGAPSE: VGA palette snoop Enable** Not applicable to PCI Express. Hardwired to 0.
4	RO	0	**MWIEN: Memory Write and Invalidate Enable** Not applicable to PCI Express. Hardwired to 0.
3	RO	0	SCE: Special Cycle Enable Not applicable to PCI Express. Hardwired to 0.

Device: 0, 2-7 Function: 0 Offset: 04h			
Bit	**Attr**	**Default**	**Description**
2	RW	0	**BME: Bus Master Enable** Controls the ability of the PCI Express port to forward memory or I/O transactions. 1: Enables the PCI Express port to successfully complete the memory or I/O read/write requests. 0: The Bus Master is disabled. The MCH will treat upstream memory writes/reads, I/O writes/reads, and MSIs as illegal cycles and return Unsupported Request Status (equivalent to Master abort) in PCI Express When the BME is disabled, the MCH will treat upstream memory writes/reads, I/O writes/reads, and MSIs as illegal cycles and return Unsupported Request Status (equivalent to Master abort) in PCI Express Requests other than inbound memory or I/O (e.g configuration, outbound) are not controlled by this bit. The BME is typically used by the system software for operations such as Hot-Plug, device configuration. When the CPURESET# signal is asserted during a power good or hard reset and after the DMI completes its training, the LPC device in the Intel 631xESB/632xESB I/O Controller Hub (or other NIC/SIO4 cards could potentially send inbound requests even before the CPURESET# is deassserted. This corner case is handled by the BME filtration in the Intel 5000X Chipset MCH's PCI Express port using the above rules since BME is reset. However, in general, it is illegal for a an I/O device to issue inbound requests until the CPURESET# has been deasserted to prevent any possible malfunction in the Intel 5000X Chipset MCH logic.
1	if (port 7-2) {RW} elseif (port 0) {RO} endif	0	**MSE: Memory Space Enable** Controls the bridge's response as a target to memory accesses on the primary interface that address a device that resides behind the bridge in both the non-prefetchable and prefetchable memory ranges (high/low) or targets a memory-mapped location within the bridge itself 1: Enables the Memory and Prefetchable memory address ranges (MMIO) defined in the **MBASE/MLIM, PMBASE/PMLIM, PMBU/PMLU** registers. 0: Disables the entire memory space seen by the PCI Express port on the primary side (MCH). Requests will then be subtractively claimed by Intel 631xESB/632xESB I/O Controller Hub. For port 0, this bit is hardwired to 0 since the ESI is not a P2P bridge.
0	if (port 7-2) {RW} elseif (port 0) {RO} endif	0	**IOAE: Access Enable** 1: Enables the I/O address range defined in the IOBASE and IOLIM registers. 0: Disables the entire I/O space seen by the PCI Express port on the primary. Requests will be then be subtractively claimed by Intel 631xESB/632xESB I/O Controller Hub For port 0, this bit is hardwired to 0 since the ESI is not a P2P bridge.

Notes:

a. In addition, BCCTRL.BCSERRE also gates the transmission of ERR_FATAL, NON_FATAL and ERR_COR messages received from the PCI Express interface. See Section 3.8.8.28

3.8.8.2 PCISTS[7:2, 0] - Status Register

The PCISTS is a 16-bit status register that reports the occurrence of error conditions associated with the primary side of the “virtual” PCI-PCI bridge embedded in the selected PCI Express cluster of the MCH.

Device: 0, 2-7
Function: 0
Offset: 06h

Bit	Attr	Default	Description
15	RWC	0	**DPE: Detected Parity Error** This bit is set when the PCI Express port receives an uncorrectable data error or Address/Control parity errors regardless of the Parity Error Response Enable bit (PERRE). This applies only to parity errors that target the PCI Express port interface (inbound/outbound direction). The detected parity error maps to B1, F6, M2 and M4 (uncorrectable data error from FSB, Memory or internal sources) of the Intel 5000X Chipset MCH.
14	RWC	0	**SSE: Signaled System Error** 1: The PCI Express port generated internal FATAL/NON FATAL errors (IO0-IO17) through the ERR[2:0] pins with SERRE bit enabled. Software clears this bit by writing a ‘1’ to it. 0: No internal PCI Express port errors are signaled.
13	RWC	0	**RMA: Received Master Abort** This bit is set when a requestor (primary side for Type 1 header configuration space header device) receives a completion with Unsupported Request Completion Status. 1. Assert this RMA bit when the primary side performs operations for an unsupported transaction. These apply to inbound configs, I/O accesses, locks, bogus memory reads and any other request that is master aborted internally. These are terminated on the PCI Express link with a UR completion status, but only if a completion is required. Software clears this bit by writing a 1 to it. PEXDEVSTS.URD is set and UNCERRSTS[20].IO2Err is set in addition. 0: No Master Abort is generated
12	RWC	0	**RTA: Received Target Abort** This bit is set when a requestor (primary side for Type 1 header configuration space header device) receives a completion with Completer Abort Completion Status. For example, for supported requests that cannot be completed because of address decoding problems or other errors. These are terminated on the PCI Express link with a CA completion status, but only if a completion is required. Software clears this bit by writing a 1 to it.
11	RO	0	**STA: Signaled Target Abort** Target Abort does not exist on the primary side of the PCI Express port. Hardwired to 0.
10:9	RO	0h	**DEVSELT: DEVSEL# Timing** Not applicable to PCI Express. Hardwired to 0.
8	RWC	0	**MDPERR: Master Data Parity Error** This bit is set by the PCI Express port if the Parity Error Response Enable bit (PERRE) is set and it receives error B1, F2, F6, M2 and M4 (uncorrectable data error or Address/Control parity errors or an internal failure). If the Parity Error Enable bit (PERRE) is cleared, this bit is never set.
7	RO	0	**FB2B: Fast Back-to-Back** Not applicable to PCI Express. Hardwired to 0.
6	RV	0	*Reserved.* (by PCI SIG)
5	RO	0	**66MHZCAP: 66 MHz capable.** Not applicable to PCI Express. Hardwired to 0.
4	RO	1	**CAPL: Capabilities List** This bit indicates the presence of PCI Express capabilities list structure in the PCI Express port. Hardwired to 1. (Mandatory)

Device: 0, 2-7 Function: 0 Offset: 06h			
Bit	**Attr**	**Default**	**Description**
3	RO	0	**INTxSTAT: INTx Status** Indicates that an INTx interrupt message is pending internally in the PCI Express port. The INTx status bit should be rescinded when all the relevant events via RAS errors/HP/PM internal to the port that requires legacy interrupts are cleared by software.
2:0	RV	0h	*Reserved.* (by PCI SIG)

3.8.8.3 CLS[7:2, 0] - Cache Line Size

This register contains the Cache Line Size and is set by BIOS/Operating system. It does not affect the PCI Express port functionality in the MCH.

Device: 0, 2-7 Function: 0 Offset: 0Ch			
Bit	**Attr**	**Default**	**Description**
7:0	RW	00h	**CLS: Cache Line Size** This is an 8-bit value that indicates the size of the cache line and is specified in DWORDs. It does not affect the MCH.

3.8.8.4 PRI_LT[7:2, 0] - Primary Latency Timer

This register denotes the maximum time slice for a burst transaction in legacy PCI 2.3 on the primary interface. It does not affect/influence PCI Express functionality.

Device: 0, 2-7 Function: 0 Offset: 0Dh			
Bit	**Attr**	**Default**	**Description**
7:0	RO	00h	**Prim_Lat_timer: Primary Latency Timer** Not applicable to PCI Express. Hardwired to 00h.

3.8.8.5 BIST[7:2,0] - Built-In Self Test

This register is used for reporting control and status information of BIST checks within a PCI Express port. It is not supported in the Intel 5000X Chipset MCH.

Device: 0, 2-7 Function: 0 Offset: 0Fh			
Bit	**Attr**	**Default**	**Description**
7:0	RO	00h	**BIST_TST: BIST Tests** Not supported. Hardwired to 00h

3.8.8.6 BAR0[7:2,0] - Base Address Register 0

Base address registers are used for mapping internal registers to an MMIO or I/O space. It does not affect the MCH. The base address register 0 is not supported/defined in the PCI Express port of the MCH.

3.8.8.7 BAR1[7:2,0] - Base Address Register 1

The base address register 1 is not supported/defined in the MCH.

3.8.8.8 EXP_ROM[0]: Expansion ROM Registers

The ESI port (device 0, function 0) does not implement any Base address registers in the Intel 5000X Chipset MCH from offset 10h to 24h. Similarly no Expansion ROM base address register is defined in offset 30h. Also no Cardbus CIS pointer is defined in offset 28h. The MIN_GNT (offset 3Eh) and MAX_LAT (3Fh) registers are also not implemented as they are not applicable to the ESI interface.

3.8.8.9 PBUSN[7:2] - Primary Bus Number

This register identifies the bus number on the on the primary side (MCH) of the PCI Express port.

Device: 2-7
Function: 0
Offset: 18h

Bit	Attr	Default	Description
7:0	RO	00h	**PBUBSNUM: Primary Bus Number** Configuration software typically programs this field with the number of the bus on the primary side of the bridge. Since the PCI Express virtual PCI-PCI bridge is an internal device and its primary bus is consistently 0, these bits are read only and are hardwired to 0.

3.8.8.10 SBUSN[7:2] - Secondary Bus Number

This register identifies the bus number assigned to the secondary side (PCI Express) of the “virtual” PCI-PCI bridge. This number is programmed by the PCI configuration software to allow mapping of configuration cycles to devices connected to PCI Express.

Device:2-7
Function: 0
Offset: 19h

Bit	Attr	Default	Description
7:0	RW	00h	**SECBUSNUM: Secondary Bus Number** This field is programmed by configuration software with the lowest bus number of the busses connected to PCI Express. Since both bus 0, device 1 and the PCI to PCI bridge on the other end are considered by configuration software to be PCI-PCI bridges, this bus number will consistently correspond to the bus number assigned to the PCI Express port.

3.8.8.11 SUBUSN[7:2] - Subordinate Bus Number

This register identifies the subordinate bus (if any) that resides at the level below the secondary PCI Express interface. This number is programmed by the PCI configuration software to allow mapping of configuration cycles to devices subordinate to the secondary PCI Express port.

Device: 2-7
Function: 0
Offset: 1Ah

Bit	Attr	Default	Description
7:0	RW	00h	**SUBBUSNUM: Subordinate Bus Number** This register is programmed by configuration software with the number of the highest subordinate bus that is behind the PCI Express port.

3.8.8.12 SEC_LT[7:2] - Secondary Latency Timer

This register denotes the maximum time slice for a burst transaction in legacy PCI 2.3 on the secondary interface. It does not affect/influence PCI Express functionality.

Device: 2-7
Function: 0
Offset: 1Bh

Bit	Attr	Default	Description
7:0	RO	00h	**Slat_tmr: Secondary Latency Timer** Not applicable to PCI Express. Hardwired to 00h.

3.8.8.13 IOBASE[7:2] - I/O Base Register

The I/O Base and I/O Limit registers (see Section 3.8.8.14) define an address range that is used by the PCI Express port to determine when to forward I/O transactions from one interface to the other using the following formula:

IO_BASE <= A[15:12]<=IO_LIMIT

Only the upper 4 bits are programmable. For the purpose of address decode, address bits A[11:0] are treated as 0. The bottom of the defined I/O address range will be aligned to a 4 KB boundary while the top of the region specified by IO_LIMIT will be one less than a 4 KB multiple. Refer to Section 4.5.1 and Section 4.5.3 in the Intel 5000X Chipset Platform Specification.

Device: 2-7
Function: 0
Offset: 1Ch

Bit	Attr	Default	Description
7:4	RW	0h	**IOBASE: I/O Base Address** Corresponds to A[15:12] of the I/O addresses at the PCI Express port.
3:0	RO	0h	**IOCAP: I/O Address capability** 0h – 16 bit I/O addressing, (supported) 1h – 32 bit I/O addressing, others - Reserved. The MCH does not support 32 bit addressing, so these bits are hardwired to 0.

3.8.8.14 IOLIM[7:2] - I/O Limit Register

The I/O Base and I/O Limit registers define an address range that is used by the PCI Express bridge to determine when to forward I/O transactions from one interface to the other using the following formula:

IO_BASE <= A[15:12] <=IO_LIMIT

Only the upper 4 bits of this register are programmable. For the purpose of address decode, address bits A[11:0] of the I/O limit register is treated as FFFh.

Device: 2-7
Function: 0
Offset: 1Dh

Bit	Attr	Default	Description
7:4	RW	0h	**IOLIMIT: I/O Address Limit** Corresponds to A[15:12] of the I/O addresses at the PCI Express port.
3:0	RO	0h	**IOLCAP: I/O Address Limit Capability** 0h – 16 bit I/O addressing, (supported) 1h – 32 bit I/O addressing, others - Reserved. The MCH does not support 32 bit I/O addressing, so these bits are hardwired to 0.

3.8.8.15 SECSTS[7:2] - Secondary Status

SECSTS is a 16-bit status register that reports the occurrence of error conditions associated with secondary side (that is, PCI Express side) of the “virtual” PCI-PCI bridge embedded within MCH.

Device: 2-7
Function: 0
Offset: 1Eh

Bit	Attr	Default	Description
15	RWC	0	**SDPE: Detected Parity Error** This bit is set by the Intel 5000X Chipset MCH whenever it receives a poisoned TLP in the PCI Express port regardless of the state the Parity Error Response bit (**in the BCTRL.PRSPEN register)**. **BCTRL.PRSPEN register)**. This corresponds to IO4 as defined in Table 5-38, “Intel 5000X Chipset Error List” on page 453
14	RWC	0	**SRSE: Received System Error** This bit is set by the MCH when it receives a ERR_FATAL or ERR_NONFATAL message. Section 3.8.8.28. (Note that BCTRL.BCSERRE is not a gating item for the recording of this error on the secondary side).
13	RWC	0	S**RMAS: Received Master Abort Status** This bit is set when the PCI Express port receives a Completion with “Unsupported Request Completion” Status.
12	RWC	0	**SRTAS: Received Target Abort Status** This bit is set when the PCI Express port receives a Completion with “Completer Abort” Status.
11	RWC	0	**SSTAS: Signaled Target Abort** This bit is set when the PCI Express port completes a request with “Completer Abort” Status when the PEXSTS.RTA is set since the MCH acts as a virtual PCI bridge and passes the completion abort from the primary to the secondary side. Note however that the MCH will not set the SSTAS field directly on the secondary side since all requests are passed upstream through the primary side to the internal core logic for decoding.

Device: 2-7 Function: 0 Offset: 1Eh			
Bit	**Attr**	**Default**	**Description**
10:9	RO	00	**SDEVT: DEVSEL# Timing** Not applicable to PCI Express. Hardwired to 0
8	RWC	0	**SMDPERR: Master Data Parity Error** This bit is set by the PCI Express port on the secondary side (PCI Express link) if the Parity Error Response Enable bit (PRSPEN) in the Section 3.8.8.28 is set and either of the following two conditions occurs: •The PCI Express port receives a Completion marked poisoned •The PCI Express port poisons a write Request If the Parity Error Response Enable bit is cleared, this bit is never set. Refer to Table 3-33 for details on the data parity error handling matrix in the Intel 5000X Chipset MCH.
7	RO	0	**SFB2BTC: Fast Back-to-Back Transactions Capable** Not applicable to PCI Express. Hardwired to 0.
6	RV	0	*Reserved.* (by PCI SIG)
5	RO	0	**S66MHCAP: 66 MHz capability** Not applicable to PCI Express. Hardwired to 0.
4:0	RV	0h	*Reserved.* (by PCI SIG)

Table 3-33. Intel 5000X Chipset MCH PCISTS and SECSTS Master/Data Parity Error RAS Handling

Register Name	OB Post	OB Compl	IN Post	IB Compl
PCISTS[15].DPE[a]	yes	yes	no	no
PCISTS[8].MDPERR	no	yes	no	no
SECSTS[15].SDPE	no	no	yes	yes
SECSTS[8].SMDPERR	no	no	no	yes

Notes:

a. In general, the DPE field is the superset of the MDPERR from a virtual PCI-PCI bridge perspective but there may be cases where a PCISTS[8].MDPERR may not be logged in the PCISTS[15].DPE field in the Intel 5000X Chipset MCH on the primary side.

3.8.8.16 MBASE[7:2] - Memory Base

The Memory Base and Memory Limit registers define a memory mapped I/O non-prefetchable address range (32-bit addresses) and the MCH directs accesses in this range to the PCI Express port based on the following formula:

MEMORY_BASE <= A[31:20] <= MEMORY_LIMIT

The upper 12 bits of both the Memory Base and Memory Limit registers are read/write and corresponds to the upper 12 address bits, AD[31:20], of 32-bit addresses. For the purpose of address decoding, the bridge assumes that the lower 20 address bits, AD[19:0], of the memory base address are zero. Similarly, the bridge assumes that the lower 20 address bits, AD[19:0], of the memory limit address (not implemented in the Memory Limit register) are FFFFFh. Thus, the bottom of the defined memory address range will be aligned to a 1 MB boundary and the top of the defined memory address range will be one less than a 1 MB boundary. Refer to Section 4.3.9, Section 4.4.2 and Section 4.4.3 in the Intel 5000X Chipset programmer's guide for further details on address mapping.

Device: 2-7 Function: 0 Offset: 20h			
Bit	**Attr**	**Default**	**Description**
15:4	RW	0h	**MBASE: Memory Base Address** Corresponds to A[31:20] of the memory address on the PCI Express port.
3:0	RO	0h	*Reserved.* (by PCI SIG)

3.8.8.17 MLIM[7:2]: Memory Limit

This register controls the processor to PCI Express non-prefetchable memory access routing based on the following formula as described above:

MEMORY_BASE <= A[31:20] <= MEMORY_LIMIT

The upper 12 bits of the register are read/write and correspond to the upper 12 address bits A[31:20] of the 32 bit address. The bottom 4 bits of this register are read-only and return zeroes when read. This register must be initialized by the configuration software. For the purpose of address decode address bits A[19:0] are assumed to be FFFFFh.

Memory range covered by MBASE and MLIM registers, are used to map non-prefetchable PCI Express address ranges (typically where control/status memory-mapped I/O data structures reside) and PMBASE and PMLIM are used to map prefetchable address ranges. This segregation allows application of USWC space attribute to be performed in a true plug-and-play manner to the prefetchable address range for improved PCI Express memory access performance.

Note also that configuration software is responsible for programming all address range registers such as MIR, MLIM, MBASE, IOLIM, IOBASE, PMBASE, PMLIM, PMBU, PMLU (coherent, MMIO, prefetchable, non-prefetchable, I/O) with the values that provide exclusive address ranges that is, prevent overlap with each other and/or with the ranges covered with the main memory. There is no provision in the MCH hardware to enforce prevention of overlap and operations of the system in the case of overlap are not guaranteed.

Device: 2-7 Function: 0 Offset: 22h			
Bit	**Attr**	**Default**	**Description**
15:4	RW	0h	**MLIMIT: Memory Limit Address** Corresponds to A[31:20] of the memory address that corresponds to the upper limit of the range of memory accesses that will be passed by the PCI Express bridge
3:0	RO	0h	*Reserved.* (by PCI SIG)

3.8.8.18 PMBASE[7:2] - Prefetchable Memory Base

The Prefetchable Memory Base and Memory Limit registers define a memory mapped I/O prefetchable address range (32-bit addresses) which is used by the PCI Express bridge to determine when to forward memory transactions based on the following formula:

PREFETCH_MEMORY_BASE <= A[31:20] <= PREFETCH_MEMORY_LIMIT

The upper 12 bits of both the Prefetchable Memory Base and Memory Limit registers are read/write and corresponds to the upper 12 address bits, A[31:20], of 32-bit addresses. For the purpose of address decoding, the bridge assumes that the lower 20 address bits, A[19:0], of the memory base address are zero. Similarly, the bridge assumes that the lower 20 address bits, A[19:0], of the memory limit address (not implemented in the Memory Limit register) are F FFFFh. Thus, the bottom of the defined memory address range will be aligned to a 1 MB boundary and the top of the defined memory address range will be one less than a 1 MB boundary.

Device: 2-7
Function: 0
Offset: 24h

Bit	Attr	Default	Description
15:4	RW	0h	**PMBASE: Prefetchable Memory Base Address** Corresponds to A[31:20] of the prefetchable memory address on the PCI Express port.
3:0	RO	1h	**PMBASE_CAP: Prefetchable Memory Base Address Capability** 0h – 32 bit Prefetchable Memory addressing 1h – 64bit Prefetchable Memory addressing, others - Reserved.

The bottom 4 bits of both the Prefetchable Memory Base and Prefetchable Memory Limit registers are read-only, contain the same value, and encode whether or not the bridge supports 64-bit addresses. If these four bits have the value 0h, then the bridge supports only 32 bit addresses. If these four bits have the value 01h, then the bridge supports 64-bit addresses and the Prefetchable Base Upper 32 Bits and Prefetchable Limit Upper 32 Bits registers hold the rest of the 64-bit prefetchable base and limit addresses respectively.

3.8.8.19 PMLIM[7:2] - Prefetchable Memory Limit

This register controls the processor to PCI Express prefetchable memory access routing based on the following formula as described above:

PREFETCH_MEMORY_BASE <= A[31:20] <= PREFTCH_MEMORY_LIMIT

The upper 12 bits of the register are read/write and correspond to the upper 12 address bits A[31:20] of the 32 bit address. This register must be initialized by the configuration software. For the purpose of address decode address bits A[19:0] are assumed to be F FFFFh.

Device: 2-7
Function: 0
Offset: 26h

Bit	Attr	Default	Description
15:4	RW	0h	**PMLIMIT: Prefetchable Memory Limit Address** Corresponds to A[31:20] of the memory address on the PCI Express bridge
3:0	RO	1h	**PMLIMIT_CAP: Prefetchable Memory Limit Address Capability** 0h – 32 bit Prefetchable Memory addressing 1h – 64 bit Prefetchable Memory addressing, others - Reserved.

3.8.8.20 PMBU[7:2] - Prefetchable Memory Base (Upper 32 bits)

The Prefetchable Base Upper 32 Bits and Prefetchable Limit Upper 32 Bits registers are extensions to the Prefetchable Memory Base and Prefetchable Memory Limit registers. If the Prefetchable Memory Base and Prefetchable Memory Limit registers indicate support for 32-bit addressing, then the Prefetchable Base Upper 32 Bits and Prefetchable Limit Upper 32 Bits registers should return zero when read. If the Prefetchable Memory Base and Prefetchable Memory Limit registers indicate support for 64-bit addressing, then the Prefetchable Base Upper 32 Bits and Prefetchable Limit Upper 32 Bits registers are implemented as read/write registers.

If a 64-bit prefetchable memory address range is supported, the Prefetchable Base Upper 32 Bits and Prefetchable Limit Upper 32 Bits registers specify the upper 32-bits, corresponding to A[63:32], of the 64-bit base and limit addresses which specify the prefetchable memory address range.

Device: 2-7
Function: 0
Offset: 28h

Bit	Attr	Default	Description
31:0	RW	0h	**PUMBASE: Prefetchable Upper 32-bit Memory Base Address** Corresponds to A[63:32] of the memory address that maps to the upper base of the prefetchable range of memory accesses that will be passed by the PCI Express bridge. OS should program these bits based on the available physical limits of the system.

3.8.8.21 PMLU[7:2] - Prefetchable Memory Limit (Upper 32 bits)

Device: 2-7
Function: 0
Offset: 2Ch

Bit	Attr	Default	Description
31:0	RW	0h	**PUMLIM: Prefetchable Upper 32-bit Memory Limit Address** Corresponds to A[63:32] of the memory address that maps to the upper limit of the prefetchable range of memory accesses that will be passed by the PCI Express bridge. OS should program these bits based on the available physical limits of the system.

3.8.8.22 IOB[7:2] - I/O Base Register (Upper 16 bits)

Not used since MCH does not support upper 16-bit I/O addressing.

3.8.8.23 IOL[7:2] - I/O Limit Register (Upper 16 bits)

Not used since MCH does not support upper 16-bit I/O addressing.

3.8.8.24 CAPPTR[7:2, 0]- Capability Pointer

The CAPPTR is used to point to a linked list of additional capabilities implemented by this device.

It provides the offset to the first set of capabilities registers located in the PCI compatible space from 40h. Currently the first structure is located 50h to provide room for other registers.

Device: 0, 2-7 Function: 0 Offset: 34h			
Bit	**Attr**	**Default**	**Description**
7:0	RO	50h	**CAPPTR: Capability Pointer** Points to the first capability structure (PM) in PCI 2.3 compatible space at 50h

3.8.8.25 RBAR[7:2] - ROM Base Address Register

Not implemented in MCH, since the MCH is a virtual PCI-PCI bridge.

3.8.8.26 INTL[7:2,0] - Interrupt Line Register

The Interrupt Line register is used to communicate interrupt line routing information between the initialization code and the device driver. The MCH does not have a dedicated interrupt line. This register RO and is provided for backwards compatibility.

Device: 0, 2-7 Function: 0 Offset: 3Ch			
Bit	**Attr**	**Default**	**Description**
7:0	RO	00h	**INTL: Interrupt Line** BIOS writes the interrupt routing information to this register to indicate which input of the interrupt controller this PCI Express Port is connected to. Not used in MCH since the PCI Express port does not have interrupt lines.

3.8.8.27 INTP[7:2,0] - Interrupt Pin Register

The INTP register identifies legacy interrupts for INTA, INTB, INTC and INTD as determined by BIOS/ firmware. These are emulated over the ESI port using the appropriate Assert_Intx commands.

Device: 0, 2-7 Function: 0 Offset: 3Dh			
Bit	**Attr**	**Default**	**Description**
7:0	RWO	01h	**INTP: Interrupt Pin** This field defines the type of interrupt to generate for the PCI Express port. 001: Generate INTA 010: Generate INTB 011: Generate INTC 100: Generate INTD Others: Reserved BIOS/configuration Software has the ability to program this register once during boot to set up the correct interrupt for the port.

3.8.8.28 BCTRL[7:2] - Bridge Control Register

This register provides extensions to the PCICMD register that are specific to PCI-PCI bridges. The BCTRL provides additional control for the secondary interface (that is, PCI Express) as well as some bits that affect the overall behavior of the “virtual” PCI-PCI bridge embedded within the MCH, for example, VGA compatible address range mapping.

Device: 2-7
Function: 0
Offset: 3Eh

Bit	Attr	Default	Description
15:12	RV	0h	*Reserved.* (by PCI SIG)
11	RO	0	**DTSS: Discard Timer SERR Status** Not applicable to PCI Express. This bit is hardwired to 0.
10	RO	0	**DTS: Discard Timer Status** Not applicable to PCI Express. This bit is hardwired to 0.
9	RO	0	**SDT: Secondary Discard Timer** Not applicable to PCI Express. This bit is hardwired to 0.
8	RO	0	**PDT: Primary Discard Timer** Not applicable to PCI Express. This bit is hardwired to 0.
7	RO	0	**FB2BEN: Fast Back-to-Back Enable** Not applicable to PCI Express. This bit is hardwired to 0.
6	RW	0	**SBUSRESET: Secondary Bus Reset** 1: Setting this bit causes a hot reset on the link for the corresponding PCI Express port and the PCI Express hierarchy domain subordinate to the port. This sends the LTSSM into the Hot-Reset state, which necessarily implies a reset to the downstream device and all subordinate devices. The mechanism to reset the downstream device is utilizing the TS1/TS2 “link reset” bit (bit number 0 of symbol 5). It is recommended for software/BIOS that the SBUSRESET field be held asserted for a minimum of 2 ms to ensure that the Link enters the Hot-Reset state from L0 or L1/L2. Software can also poll the PEXLNKSTS.LNKTRG bit for a deasserted condition to determine if the hot-reset state has been entered at which point it can clear the SBUSRESET field to train the link. When this SBUSRESET bit is cleared after the MCH enters the “hot-reset” state, the Intel 5000X Chipset MCH will initiate operations to move to “detect” state and then train the link (polling, configuration, L0 (link-up)) after sending at least 2 TS1 and receiving 1 TS1 with the HotReset bit set in the training control field of TS1 and waiting for 2 ms in the Hot-reset state. The 2ms stay in the Hot-reset state is enforced by the chipset LTSSM for the PCI Express hierarchy to reset. If the SBUSRESET is held asserted even after the 2 ms time-out has expired, the Intel 5000X Chipset MCH will continue to maintain the hot-reset state. Hence it is necessary for software to clear this register appropriately to bring the link back in training. Note also that a secondary bus reset will not in general reset the primary side configuration registers of the targeted PCI Express port. This is necessary to allow software to specify special training configuration, such as entry into loopback mode. 0: No reset happens on the PCI Express port.
5	RO	0	**MAMODE: Master Abort Mode** Not applicable to PCI Express. This bit is hardwired to 0.
4	RW	0	**VGA16bdecode: VGA 16-bit decode** This bit enables the virtual PCI-to-PCI bridge to provide 16-bit decoding of VGA I/O address precluding the decoding of alias addresses every 1 KB. The I/O addresses decoded is in the range of 03B0h to 03BBh or 03C0h to 03DFh within the first 1 KB I/O space. 0: execute 10-bit address decodes on VGA I/O accesses. 1: execute 16-bit address decodes on VGA I/O accesses. This bit only has meaning if bit 3 (VGAEN) of this register is also set to 1, enabling VGA I/O decoding and forwarding by the bridge. This read/write bit enables system configuration software to select between 10- and 16-bit I/O address decoding for all VGA I/O register accesses that are forwarded from the primary to secondary whenever the VGAEN is set to 1.

Device: 2-7 Function: 0 Offset: 3Eh			
Bit	**Attr**	**Default**	**Description**
3	RW	0	**VGAEN: VGA Enable** Controls the routing of CPU initiated transactions targeting VGA compatible I/O and memory address ranges. This bit may only be set for one PCI Express port.
2	RW	0	**ISAEN: ISA Enable** Modifies the response by the Intel 5000X Chipset MCH to an I/O access issued by the CPU that target ISA I/O addresses. This applies only to I/O addresses that are enabled by the IOBASE and IOLIM registers. 1: The Intel 5000X Chipset MCH will not forward to PCI Express any I/O transactions addressing the last 768 bytes in each 1KB block even if the addresses are within the range defined by the IOBASE and IOLIM registers. See Section 4.5.2. Instead of going to PCI Express these cycles will be forwarded to ESI where they can be subtractively or positively claimed by the ISA bridge. 0: All addresses defined by the IOBASE and IOLIM for CPU I/O transactions will be mapped to PCI Express.
1	RW	0	**BCSERRE: SERR Enable** This bit controls forwarding of ERR_COR, ERR_NONFATAL and ERR_FATAL messages from the PCI Express port to the primary side. 1: Enables forwarding of ERR_COR, ERR_NONFATAL and ERR_FATAL messages. 0: Disables forwarding of ERR_COR, ERR_NONFATAL and ERR_FATAL. Note that BCSERRE is no longer a gating item for the recording of the SESCSTS.SRSE error.
0	RW	0	**PRSPEN: Parity Error Response Enable** This bit controls the response to poisoned TLPs in the PCI Express port 1: Enables reporting of poisoned TLP errors. 0: Disables reporting of poisoned TLP errors

3.8.8.29 PEXLWSTPCTRL: PCI Express Link Width Strap Control Register

This register provides the ability to change the PCI Express link width through software control. Normally, the Intel 5000X Chipset MCH will use the PEX_LWSTP[3:0] pins to train the links. However, if BIOS needs the ability to circumvent the pin strappings and enforce a specific setting for a given platform, it must perform a soft initialization sequence through the following actions in this register:

1. Set PEXLWSTPCTRL.LWOEN to '1'.
2. Write the desired link width to PEXLWSTPCTRL.GPMNXT0(1) fields for IOU0 and IOU1 clusters.
3. Perform a hard reset to the Intel 5000X Chipset MCH.

The chipset will then use the values initialized in the PEXLWSTPCTRL.GPMNXT0(1) fields and train the links appropriately following the hard reset. The Intel 5000X Chipset MCH will also provide status information to the software as to what link width it is currently using to train the link via PEXLWSTPCTRL.GPMCUR0(1) fields and the appropriate training mode, PEXLWSTPCTRL.LWTM. (pins strap vs. software enabled mode).

Device: 0 Function: 0 Offset: 40h			
Bit	**Attr**	**Default**	**Description**
15:14	RV	0h	**Reserved**
13:11	RO	000	**GPMCUR1: IOU1 max width Current Configuration Now (ports 4-7)** This field is updated by the hardware to indicate the current link width of IOU1 ports that is used for training. This field is set before training gets underway. 000: x4,x4,x4,x4 001: x8,--,x4,x4 010: x4,x4, x8,-- 011: x8,--,x8,-- 100: x16,--,--,--,-- 111: Auto negotiation others: Reserved
10:8	RO	000	**GPMCUR0: IOU0 max width Current Configuration (ports 2-3 only, port 0, ESI, is always x4)** This field is updated by the hardware to indicate the current link width of IOU1 ports that is used for training. This field is set before training gets underway. 000: x4,x4 001: Reserved 010: x8,-- 111: Auto Negotiation Others: Reserved
7	RO	0	**LWTM: Link Width Training Mode** This field is updated by the hardware to provide feedback to software on the training mode it is using following reset. that is, link strap or soft initialization of link widths. 0: IOU clusters trained the links using the PEX_LWSTP[3:0] pins (external strapping) [default] 1: IOU clusters trained the links using the soft initialization mechanism in this register viz. GPMNXT1 and GPMNXT0 following a hard reset.
6:4	RWST	000	**GPMNXT1: IOU1 max width Configuration Next (ports 4-7)** The IOU1 cluster will use this field to train the link after a hard reset provided LWOEN is set. Refer to Table , "" on page 117
3:1	RWST	000	**GPMNXT0: IOU0 max width Configuration Next (ports 2-3)** The IOU0 cluster will use this field to train the links after a hard reset provided LWOEN is set. Refer to Table , "" on page 117
0	RWST	0	**LWOEN: Link Width override Enable** 0: Disables software from setting the PCI Express link width through this register and the Link width is controlled by the external pins PEX_LWSTP[3:0]. (default). 1. Enables BIOS/Software to set the required link width through this register. When this bit is set, the IOU cluster will ignore the external pin strap (PEX_LWSTP[3:0] and use the described table for configuring the link width. The values will take effect after a hard reset.

Table 3-1. GIO Port Mode Selection

GIO Port (IOU0)					GIO Port (IOU1)				
GPMNXT1 [2:0] (IOU0)	**Port0 (ESI)**	**Port1 (RSVD)**	**Port2**	**Port3**	**GPMNXT0 [2:0] (IOU1)**	**Port4**	**Port5**	**Port6**	**Port7**
3'b000	x4	RSVD	x4	x4	3'b000	x4	x4	x4	x4
3'b001	invalid				3'b001	x8	N/A	x4	x4
3'b010	x4	RSVD	x8	N/A	3'b010	x4	x4	x8	N/A

GIO Port (IOU0)					GIO Port (IOU1)				
GPMNXT1 [2:0] (IOU0)	Port0 (ESI)	Port1 (RSVD)	Port2	Port3	GPMNXT0 [2:0] (IOU1)	Port4	Port5	Port6	Port7
3'b011	invalid				3'b011	x8	N/A	x8	N/A
3'b100	invalid				3'b100	x16	N/A	N/A	N/A
3'b101	invalid				3'b101	invalid			
3'b110	invalid				3'b110	invalid			
3'b111	x4	RSVD	Auto Negotiation		3'b111	Auto Negotiation			

3.8.8.30 CBPRES: DMA Engine Present Control Register

This register provides control for suppressing access to the configuration space of selected devices within the Intel 5000X Chipset MCH. Specifically, the BIOS can enable/disable DMA Engine configuration and memory mapped operations to device 8, function 0 and device 8, function 1 respectively. This is a special register intended to suppress the “yellow-bang” warning for the DMA Engine device for Intel 5000X Chipset customers who install non-standard Operating systems without associated drivers. It can also be used as a defeature mode to block DMA Engine technology from being used.

Device: 0
Function: 0
Offset: 44h

Bit	Attr	Default	Description
15:1	RV	0	Reserved
0	RWO	0	**CB_CFG_ENABLE: DMA Engine Configuration Enable** 1: Enable Configuration/Memory mapped accesses to the DMA Engine configuration space located in Device 8, Fn 0 and Fn 1.
			0: Disable DMA Engine configuration accesses to Device 8, Fn 0 and Fn 1. The Intel 5000X Chipset MCH will master abort requests to DMA Engine configuration/memory mapped space.

3.8.8.31 SSCTRL[7:2,0]: Stop and Scream Control Register

This register provides the ability for the Intel 5000X Chipset MCH to filter poisoned TLP in the outbound direction (writes, read completions) and prevent the data from corrupting the end point.

Device: 7-2, 0
Function: 0
Offset: 47h

Bit	Attr	Default	Description
7:1	RV	0h	***Reserved***
0	RW	0	**SSEN: Stop and Scream Enable** 0: Disable Stop and Scream (normal/default operation) 1: Enables the link to perform Stop and Scream functionality. When this bit is set, if the Intel 5000X Chipset MCH encounters a poisoned data traversing the outbound path towards the PCI Express/DMI Port, the IOU cluster stops the data, sets the “link_down condition” internally and places the link in “hot_reset”. Inbound/outbound paths are blocked and all outstanding requests will be master aborted including Memory, I/O and configuration requests. **Note:** The Intel 5000X Chipset MCH does not support this feature and therefore it is required to keep this bit disabled.

3.8.8.32 PEXCTRL[7,2:0]: PCI EXPRESS Control Register

Device: 7-2,0
Function: 0
Offset: 48h

Bit	Attr	Default	Description
31:26	RW	0h	***Reserved***
25:24	RW	00	**COALESCE_MODE**: Used to increase the amount of combining for completions. 00: No restriction on coalescing_hint. The IOU will try to maximize completion combining. Since Intel 5000X Chipset MCH issues requests in order, it does not make sense to restrict the coalesce hint because there are few resources available at the time of fetch. By the time the hint is used, resources could be freed up and reused for the following requests. Note: This mode of "00" is the preferred setting for Intel 5000X Chipset MCH if COALESCE_EN=1 for software/BIOS 01: #CPL_ENTRIES_FREE will restrict coalesce_hint 10: if set then #PF_PEND will restrict coalesce hint 11: Minimum of coalesce_hint obtained from settings "01" and "10"
23	RW	0	**TIMEOUT_ENABLE_CFG: Timeout enable for configuration transactions** 1: Config transactions can time out. 0: Config transactions cannot time out. Suggested value: 0 ***Note:*** In general, configuration timeouts on the PCI-Express port should not be enabled. This is necessary to permit slow devices nested deep in the PCI hierarchy that may take longer to complete requests than the maximum timeout specified in the Intel 5000P. Software/BIOS should set this field based on the context and usage/platform configuration. For example, compliance testing with a known broken card should have this field set. ***Note:*** For the configuration timeout to take effect, (due to Intel 5000X RTL implementation) the PEXCTRL.TIMEOUT_ENABLE (bit 22) has to be set. ***Note:*** Due to recently discovered RTL bug in B3 and later stepping, the IOU will log a completion error (IO6) for any outstanding configuration transaction that crosses the counter limit even if this register field is cleared or bit 22 of this register is cleared (Example, either timeout is disabled. However, it does not affect the functionality and the config transaction will be outstanding indefinitely until the completion is returned except for the unnecessary error log. Software should be aware of this limitation when the field is cleared.)
22	RW	0	**TIMEOUT_ENABLE: Timeout enable for non-configuration transactions** 1: Non config transactions can time out. 0: Non config transactions cannot time out. Suggested value: 1 ***Note:*** When both TIMEOUT_ENABLE_CFG and TIMEOUT_ENABLE fields are set to 0, the Intel 5000X will assume an infinite completion time for the respective transactions. Hence the system is dependent on the end device returning the completion response at some point in time, else it will result in a hang. ***Note:*** Due to recently discovered RTL bug in B3 and later stepping, the IOU will log a completion error (IO6) for any outstanding configuration transaction that crosses the counter limit even if this register field is cleared or bit 22 of this register is cleared (Example, either timeout is disabled. However, it does not affect the functionality and the config transaction will be outstanding indefinitely until the completion is returned except for the unnecessary error log. Software should be aware of this limitation when the field is cleared.)

Device: 7-2,0
Function: 0
Offset: 48h

Bit	Attr	Default	Description
21	RW	0	MALTLP_EN: 1: Check for certain malformed TLP types. 0: Do not check for certain malformed TLP types. Suggested value: 1 When this bit is set, it enables the following conditions to mark a packet as malformed: • 4DW header MEM_RD or MEM_WR and the address is less than 32 bits (address[39:32] = 0) • Byte enable check for mem/io/cfg requests. Length > 1 DW and (first dword byte enables = 0 or last dword byte enables = 0) Length = 1 DW and last dword byte enables != 0 • IO{rd,wr}/cfg{rd,wr}{0,1} and (traffic class != 0 or attributes != 0 or length != 1) • A configuration retry completion response (CRS) received for a non-cfg outbound request
20:13	RV	0h	***Reserved***
12	RW	0	**Max_rdcmp_lmt_EN**: Maximum Read completion combining limit Enable 1: Up to 256B return and **COALESCE_EN** = 1. 0: Up to 128B return if COALESCE_EN = 1 Note: It is strongly recommended that this field should not be set to 1 (256 B completion combining) due an MCH B2 silicon issue, especially when MPS is configured to 256 B.
11	RW	0	**COALESCE_FORCE**: Force coalescing of accesses. When 1, forces Intel 5000X Chipset MCH to wait for all coalescable data before sending the transaction as opposed to forwarding as much as possible. 0: Normal operation 1: wait to coalesce data **Note:** It is strongly recommended that COALESCE_FORCE should not be set to '1' due to an MCH B2 silicon erratum.
10	RW	0	**COALESCE_EN**: Read completion coalescing enable When 1, enables read return of >64B. 1: Returns of >64B enabled. (See **Max_rdcmp_lmt_EN above)**. 0: Returns are 64B or less. **NOTE:** For optimal read completion combining, this field should be set to '1' along with **Max_rdcmp_lmt_EN** as '0' for 128 B completion combining.
9	RW	0	**PMEGPEEN**: PME GPE Enable 1: Enables "assert_pmegpe" (deassert_pmegpe) messages to be sent over the DMI from the root complex for PM interrupts. 0: Disables "assert_pmegpe" (deassert_pmegpe) messages for PM events to the root complex. This has an overriding effect to generate ACPI PM interrupts over traditional interrupts (MSI/intx).
8	RW	0	**HPGPEEN: Hotplug GPE Enable** 1: Enables "assert_hpgpe" (deassert_hpgpe) messages to be sent from the root complex for Hot-plug events. 0: Disables "assert_hpgpe" (deassert_hpgpe) messages for Hot-plug events from the root complex. This has an overriding effect to generate ACPI HP events over traditional interrupts.
7	RV	1	*Reserved*

Device: 7-2,0 Function: 0 Offset: 48h			
Bit	**Attr**	**Default**	**Description**
6:3	RW	0000	**VPP: Virtual Pin Port** [6:4] = SMBus Address, [3] =IO Port defines the 8-bit IO port that is used for routing power, attention, Hotplug, presence, MRL and other events defined in Section 3.8.11.10.
2	RW	1	**DIS_VPP: Disable VPP** The Intel 5000X Chipset MCH will use this bit to decide whether the VPP is valid or not for the given PCI Express port as set by configuration software. For example, to distinguish HP events for a legacy card or PCI Express port module, this bit can be used. 1: VPP is disabled for this PCI Express port. 0: VPP is enabled for this PCI Express port. Default value is to disable vpp for the PCI Express port
1	RW	0	**DIS_APIC_EOI; Disable APIC EOI** The Intel 5000X Chipset MCH will use this bit to decide whether end of interrupts (EOI) need to be sent to an APIC controller/bridge (for example, Intel 6700PXH 64 bit PCI Hub) through this PCI Express device. 1: No EOIs are sent (disabled). 0: EOIs are dispatched to the APIC Controller. ***Note:*** In the case of slave (secondary) ports, the BIOS has to disable EOI for that port by setting this register field. for example, x8 device connected on port 2-3 should have the PEXCTRL.DIS_APIC_EOI of the slave port (viz. #3) set to prevent EOIs from causing deadlocks. This is a micro-architectural requirement due to the internal handshake between IOU-CE for EOI slave handling.
0	if (port 7-2) {RWO} elsif (port 0) {RV} endif	0	**DEVHIDE: Device_hide** The device hide bit is used to enable the Intel 5000X Chipset MCH to hide the PCI Express device from the Operating system and is applicable only to ports 7-2. Typically, an external I/O processor acts as its proxy by configuring it and claiming resources on behalf of it and then unhides. The hiding is done by changing the class code (CCR register) for this port to 0x0600. This will prevent the OS from attempting to probe or modify anything related to this device. 1: The PCI Express port CCR register has a value of 0600. 0: The PCI Express port CCR register has a value of 0604 (bridge) The default value is '0' (to make the device a bridge). The device hide bit does not apply to the DMI interface (port 0) and has no effect on its operation

This 32-bit register implements chipset specific operations for general control/ accessibility such as device hiding, selective configuration cycles and interrupt signaling

3.8.8.33 PEXCTRL2[7:2,0]: PCI Express Control Register 2

This is an auxiliary control register for PCI Express port specific debug/defeature operations.

Device: 0, 2-7 Function: 0 Offset: 4Ch			
Bit	**Attr**	**Default**	**Description**
7:1	RV	0	*Reserved.*
0	RW	0	**NO_COMPLIANCE**: Set by software to enable link operation in the presence of single wire failures on the link. If clear, then specified link behavior in the presence of a wire failure will be Polling.Compliance.

3.8.8.34 PEXCTRL3[7:2,0] - PCI Express Control Register 3

Device: 0, 2-7
Function: 0
Offset: 4Dh

Bit	Attr	Default	Description
7:5	RV	0	*Reserved.*
4	RWO	1	**PORTENABLE: PCI Express port enable control** 1: The PCI Express port can be enabled by software and is available for use 0: The PCI Express port is disabled and not available. This setting disables the underlying port logic and associated PCI Express x4 lanes, completely removing the port from register configuration space.
3	RV	Xb	*Reserved*
2	RWO	if (port 7-2) {0} else {1} endif	**DIS_CB_BAR: Disable access to CB_BAR Memory mapped space** This bit is used for controlling CB_BAR MMIO accesses from a PCI Express port in the Intel 5000X Chipset MCH (top level). 1: Disables CB_BAR Memory mapped access and all requests to CB_BAR will be Master aborted on the PCI Express interface. 0: Enables accesses to CB_BAR region. This register bit is of type “write once” and is controlled by BIOS/special initialization firmware. The default value is “0” for ports 2 and 3 (that is, CB_Bar access is allowed) while it is disabled for the rest of the ports. If this bit is enabled for the rest of the ports other than 2 and 3, then partial access to manipulate DMA Engine resources (DMA only) is available to the end device/card connected to the enabled PCI Express port. Pl. note CB_BAR access from the non DMA Engine enabled port 0 is not a Intel 5000X Chipset MCH POR product feature and is added for internal debug/control and future expansion/market SKU variations if required.
1:0	RV	0	***Reserved***

This is an additional control register for PCI Express port specific debug/defeature operations for RAS.

3.8.8.35 CBCTRL[3:2] - PCI Express Control Register

This 8-bit register implements chipset specific operations for the ports 2 and 3 in IOU0 cluster that work with DMA Engine Technology such as disabling fast path access and write combining enable.

Device: 2-3
Function: 0
Offset: 4Eh

Bit	Attr	Default	Description
7:2	RV	0h	*Reserved.*
1	RW	0	**DIS_FP: Disable Fast Path for CB_BAR Memory mapped space** The DIS_FP is a defeature bit to debug IOU decoding for CB_BAR MMIO access through the fast path. By default, it is cleared to enable CB ports 2 and 3 to have faster (latency saving) path to the CB_BAR memory space. 1: Disables the IOU cluster from using the Fast path for CB_BAR memory access. Requests will be routed to config ring for disposition. 0: Enables IOU cluster to process memory reads/writes to CB_BAR space using the fast path that bypasses the config ring and optimizes latency. Note that DIS_CB_BAR register bit defined in Section 3.8.8.34 is the overriding bit and if DIS_CB_BAR is set for ports 2 or 3, then requests to CB_BAR will be master aborted irrespective of the value of DIS_FP bit in this register.

Device: 2-3 Function: 0 Offset: 4Eh			
Bit	**Attr**	**Default**	**Description**
0	RW	0	**WCEN: WCEnable** 1: Enables Write combining in chipset (CSWC) to proceed in the PCI Express port. 0: Disables Write combining in the given PCI Express port. Note WCEN is a feature bit that controls write combining for the port irrespective of the settings in the WCNUM, WCSIZE registers. **Note**: The default feature is to disable the WC logic as it is no longer a MCH feature.

3.8.8.36 PEXGCTRL - PCI Express Global Control Register

This 32-bit global register in the MCH implements chipset specific operations for generalized control of all PCI Express events and activity such as Power Management, hot-plug. There is only one register for all PCI Express ports and DMA Engine device that controls related I/O operations.

Device:19 Function:0 Offset:17Ch Version:Intel 5000P Chipset, Intel 5000V Chipset, Intel 5000Z Chipset			
Bit	**Attr**	**Default**	**Description**
31:18	RV	3FFFh	**Timeout: Completion Time out** Internal timer for handling Outbound NP completion timeouts. This varies based on the core clock frequency and the time at which the completion structure is loaded relative to the timeout timer which is free-running. The bounds of this roll over can be approximated as a Minimum of 6 or Max of 7 ± few cycles) since there is a 3 bit counter whose roll over is tied to the timeout timer For 333 Mhz, the granularity of this timer viz. each increment is in the range (9216 ns, 10,752 ns) giving a min/max value for a full face value of this register field as (150.99 ms, 176.15 ms) For 266 Mhz, the granularity of this timer viz. each increment is in the range (11520 ns, 13440 ns) giving a min/max value for a full face value of this register field as (188.73 ms, 220.19 ms) BIOS/Software needs to set this field as appropriate for handling various timeout conditions required by the system. ***Note:*** For example with BNB running at 333 Mhz, for SMBUS protocols, the maximum value recommended for this field is 0x744 (or 1860 decimal) to achieve a 20 ms timeout threshold (that is, 20 ms =~ 10,752 * 1860) such that it provides headroom to the chipset for the global SMBUS timeout of 25ms. Example: With 0x744 as default and 333 MHz core clock, 1. Max timeout value: If bits 31:28 were set to 0x744 (1860d), the timeout delay is calculated as follows: 1860*7 (for the rollovers)*512(lower 9 bits)*3.0ns (for 333 MHz) = 1860*107542=19.998ms=~20 ms 2. Min timeout value: If bits 31:28 were set to 0x744 (1860d), the delay calculation would be like this: 1860*6 (too close to the limit, so missed full count for one rollover)*512 (lower 9 bits)*3.0 ns (for 333 MHz)= 17.141 ms=~17 ms
17:2	RV	1385	*Reserved.*
1	RWST	0	PME_TURN_OFF: Send PME Turn Off Message When set, the Intel 5000P Chipset MCH will issue a PME Turn Off Message to all enabled PCI Express ports excluding the ESI port. The Intel 5000P Chipset MCH will clear this bit once the Message is sent.

Device:19 Function:0 Offset:17Ch Version:Intel 5000P Chipset, Intel 5000V Chipset, Intel 5000Z Chipset			
Bit	**Attr**	**Default**	**Description**
0	RWC	0	PME_TO_ACK: Received PME Time Out Acknowledge Message The Intel 5000P Chipset MCH sets this bit when it receives a PME_TO_ACK Message from all enabled PCI Express ports excluding the ESI port. Software will clear this bit when it handles the Acknowledge. Note that the ESI2 will not generate a PME_TO_Ack based on the flow described in the ESI spec. However, if a PME_TO_Ack is received at the Intel 5000P Chipset MCH ESI port, it will be Master Aborted.

3.8.8.37 INTXSWZCTRL[7:2,0]: PCI Express Interrupt Swizzle Control Register

Device: 7-2,0 Function: 0 Offset: 4Fh			
Bit	**Attr**	**Default**	**Description**
7:2	RO	0h	***Reserved***
1:0	RWO	00	**INTxSWZ: INTx Swizzle** The encoding below defines the target Intx type to which the incoming Intx message is mapped to for that port. (4 combinations using the Barber-pole slide mechanism) 00: INTA=>INTA, INTB=>INTB, INTC=>INTC, INTD=>INTD (default 1:1) 01: INTA=>INTB, INTB=>INTC, INTC=>INTD, INTD=>INTA 10: INTA=>INTC, INTB=>INTD, INTC=>INTA, INTD=>INTB 11: INTA=>INTD, INTB=>INTA, INTC=>INTB, INTD=>INTC

This register provides software the ability to swizzle the legacy interrupts (intx) from each port and remap them to a different interrupt type (IntA,B,C,D) for the purposes of interrupt rebalancing to optimize system performance. This swizzling only applies to inbound intx messages that arrive at the various ports (including ESI). The default setting is to have one-to-one map of the same interrupt types i.e (INTA => INTA, and so forth). BIOS can program this register during boot time (before enabling interrupts) to swizzle the intx types for the various ports within the combinations described in this register. MCH will use the transformed intx messages from the various ports and track them using the bit vector as a wired-or logic for sending assert/deassert intx messages on the ESI.

3.8.9 PCI Express Power Management Capability Structure

The Intel 5000X Chipset MCH PCI Express port provides basic power management capabilities to handle PM events for compatibility. The PCI Express ports can be placed in a pseudo D3 hot state but it does have real power savings and works as if it were in the D0 mode.

3.8.9.1 PMCAP[7:2,0] - Power Management Capabilities Register

The PM Capabilities Register defines the capability ID, next pointer and other power management related support. The following PM registers /capabilities are added for software compliance.

Device: 0, 2-7
Function: 0
Offset: 50h

Bit	Attr	Default	Description
31:27	RO	11001	**PMES: PME Support** Identifies power states in the Intel 5000X Chipset MCH which can send an "Assert_PMEGPE/Deassert PMEGPE" message. Bits 31, 30 and 27 must be set to '1' for PCI-PCI bridge structures representing ports on root complexes. The definition of these bits is taken from the PCI Bus Power Management Interface Specification Revision 1.1. XXXX1b - Assert_PMEGPE/Deassert PMEGPE can be sent from D0 XXX1Xb - Assert_PMEGPE/Deassert PMEGPE can be sent from D1 (**Not supported by Intel 5000X Chipset MCH**) XX1XXb - Assert_PMEGPE/Deassert PMEGPE can be sent from D2 (**Not supported by Intel 5000X Chipset MCH**) X1XXXb - Assert_PMEGPE/Deassert PMEGPE can be sent from D3 hot (**Supported by Intel 5000X Chipset MCH**) 1XXXXb - Assert_PMEGPE/Deassert PMEGPE can be sent from D3 cold (**Not supported by Intel 5000X Chipset MCH**)
26	RO	0	**D2S: D2 Support** Intel 5000X Chipset MCH does not support power management state D2.
25	RO	0	**D1S: D1 Support** Intel 5000X Chipset MCH does not support power management state D1.
24:22	RO	0h	**AUXCUR: AUX Current**
21	RO	0	**DSI: Device Specific Initialization**
20	RV	0	*Reserved.*
19	RO	0	**PMECLK: PME Clock** This field is hardwired to 0h as it does not apply to PCI Express.
18:16	RO	010	**VER: Version** This field is set to 2h as version number from the *PCI Express Base Specification*, Revision 1.0a specification.
15:8	RO	58h	**NXTCAPPTR: Next Capability Pointer** This field is set to offset 58h for the next capability structure (MSI) in the PCI 2.3 compatible space.
7:0	RO	01h	**CAPID: Capability ID** Provides the PM capability ID assigned by PCI-SIG.

3.8.9.2 PMCSR[7:2, 0] - Power Management Control and Status Register

This register provides status and control information for PM events in the PCI Express port of the MCH.

Device: 0, 2-7
Function: 0
Offset: 54h

Bit	Attr	Default	Description
31:24	RO	00h	**Data: Data** Data read out based on data select (DSEL). Refer to section 3.2.6 of PCI PM specification for details. This is not implemented in the Power Management capability for Intel 5000X Chipset MCH and is hardwired to 0h.
23	RO	0h	**BPCCEN: Bus Power/Clock Control Enable** This field is hardwired to 0h as it does not apply to PCI Express.
22	RO	0h	**B2B3S: B2/B3 Support** This field is hardwired to 0h as it does not apply to PCI Express.
21:16	RV	0h	*Reserved.*
15	RWCST	0h	**PMESTS: PME Status** This PME Status is a sticky bit. When set, the PCI Express port generates a PME internally independent of the PMEEN bit defined below. Software clears this bit by writing a '1' when it has been completed. As a root port, the Intel 5000X Chipset MCH will never set this bit, because it never generates a PME internally independent of the PMEEN bit.
14:13	RO	0h	**DSCL: Data Scale** This 2-bit field indicates the scaling factor to be used while interpreting the "data_scale" field.
12:9	RO	0h	**DSEL: Data Select** This 4-bit field is used to select which data is to reported through the "data" and the "Data Scale" fields.
8	RWST	0h	**PMEEN: PME Enable** This field is a sticky bit and when set enables PMEs generated internally to appear at the Intel 631xESB/632xESB I/O Controller Hub through the "Assert(Deassert)_PMEGPE"message. This has no effect on the Intel 5000X Chipset MCH since it does not generate PME events internally
7:2	RV	0h	*Reserved.*
1:0	RW	0h	**PS: Power State** This 2-bit field is used to determine the current power state of the function and to set a new power state as well. 00: D0 01: D1 (reserved) 10: D2 (reserved) 11: D3_hot If Software sets this to D1 or D2, then the power state will default to D0.

3.8.10 PCI Express Message Signaled Interrupts (MSI) Capability Structure

Message Signaled Interrupts (MSI) is an optional feature that enables a device to request service by writing a system-specified message to a system-specified address in the form of an interrupt message. The transaction address (for example, FEEx_xxxxh) specifies the message destination and the transaction data specifies the message. The MSI mechanism is supported by the following registers: the MSICAPID, MSINXPTR, MSICTRL, MSIAR and MSIDR register described below.

3.8.10.1 MSICAPID[7:2, 0] - MSI Capability ID

Device: 0, 2-7
Function: 0
Offset: 58h

Bit	Attr	Default	Description
7:0	RO	05h	**CAPID: Capability ID** Assigned by PCI-SIG for message signaling capability.

3.8.10.2 MSINXPTR[7:2, 0]- MSI Next Pointer

Device: 0, 2-7
Function: 0
Offset: 59h

Bit	Attr	Default	Description
7:0	RO	6Ch	**NXTPTR: Next Pointer** This field is set to 6 Ch for the next capability list (PCI Express capability structure - PEXCAP) in the chain.

3.8.10.3 MSICTRL[7:2, 0] - Message Control Register

Device: 0, 2-7
Function: 0
Offset: 5Ah

Bit	Attr	Default	Description
15:8	RV	00h	*Reserved.*
7	RO	0	**AD64CAP: 64-bit Address Capable** This field is hardwired to 0h since the message writes addresses are only 32-bit addresses (for example, FEEx_xxxxh).
6:4	RW	000	**MMEN: Multiple Message Enable** Software writes to this field to indicate the number of allocated messages which is aligned to a power of two. When MSI is enabled, the software will allocate at least one message to the device. See below for discussion on how the interrupts are handled if N is the number of messages by software. If software writes a value greater than the limit specified by the MMCAP field in the MMEN field, it is considered as a programming error. The Intel 5000X Chipset MCH GNB will only use the LSB of the MMEN (as a power of 2) to decode up to 2 messages.
3:1	RO	001	**MMCAP: Multiple Message Capable** Software reads this field to determine the number of requested messages. which is aligned to a power of two. It is set to 2 messages (encoding of 001). The Intel 5000X Chipset MCH is designed to handle MSIs for different events HP/PM events RAS Error events
0	RW	0	**MSIEN: MSI Enable** The software sets this bit to select legacy interrupts or transmit MSI messages. 0: Disables MSI from being generated. 1: Enables the Intel 5000X Chipset MCH to use MSI messages to request context specific service through register bits defined in the Section 3.8.8.34 for events such as Hot-Plug, PM, RAS. Refer to the *Intel 5000X Chipset Programming Guide* for details on the legacy, ACPI and interrupt generation events.

3.8.10.4 MSIAR[7:2, 0] - MSI Address Register

The MSI Address Register (MSIAR) contains the system specific address information to route MSI interrupts and is broken into its constituent fields.

Device: 0, 2-7
Function: 0
Offset: 5Ch

Bit	Attr	Default	Description
31:20	RO	FEEh	**AMSB: Address MSB** This field specifies the 12 most significant bits of the 32-bit MSI address.
19:12	RW	00h	**ADSTID: Address Destination ID** This field is initialized by software for routing the interrupts to the appropriate destination.
11:4	RW	00h	**AEXDSTID: Address Extended Destination ID** This field is not used by IA32 processor.
3	RW	0h	**ARDHINT: Address Redirection Hint** 0: directed 1: redirectable
2	RW	0h	**ADM: Address Destination Mode** 0: physical 1: logical
1:0	RV	0h	*Reserved.* Not used since the memory write is D-word aligned

3.8.10.5 MSIDR[7:2, 0] - MSI Data Register

The MSI Data Register (MSIDR) contains all the data (interrupt vector) related information to route MSI interrupts.

Device: 0, 2-7
Function: 0
Offset: 60h

Bit	Attr	Default	Description
31:16	RV	0000h	*Reserved.*
15	RW	0h	**TM: Trigger Mode** This field Specifies the type of trigger operation 0: Edge 1: level
14	RW	0h	**LVL: Level** If TM is 0h, then this field is a don't care. Edge triggered messages are consistently treated as assert messages. For level triggered interrupts, this bit reflects the state of the interrupt input. If TM is 1h, then 0: Deassert Messages 1: Assert Messages
13:11	RW	0h	These bits are don't care in IOxAPIC interrupt message data field specification.

Device: 0, 2-7 Function: 0 Offset: 60h			
Bit	**Attr**	**Default**	**Description**
10:8	RW	0h	**DM: Delivery Mode** 000: Fixed 001: Lowest Priority 010: SMI/HMI 011: *Reserved* 100: NMI 101: INIT 110: *Reserved* 111: ExtINT
7:0	RW	0h	**IV: Interrupt Vector** The interrupt vector (LSB) will be modified by the Intel 5000X Chipset MCH to provide context sensitive interrupt information for different events that require attention from the processor. For example, Hot-Plug, Power Management and RAS error events. Depending on the number of Messages enabled by the processor in Section 3.8.10.3, and Table 3-34 illustrates the breakdown.

Table 3-34. IV Handling and Processing by MCH

Number of Messages enabled by Software (MSICTRL.MMEN)	Events	IV[7:0]
1	All	xxxxxxxx[a]
2	HP, PM	xxxxxxx0
	RAS errors	xxxxxxx1

Notes:

a. The term “xxxxxx” in the Interrupt vector denotes that software/BIOS initializes them and the MCH will not modify any of the “x” bits except the LSB as indicated in the table as a function of MMEN.

3.8.11 PCI Express Capability Structure

The PCI Express capability structure describes PCI Express related functionality, identification and other information such as control/status associated with the port. It is located in the PCI 2.3 compatible space and supports legacy operating system by enabling PCI software transparent features.

3.8.11.1 PEXCAPL[7:2, 0]- PCI Express Capability List Register

The PCI Express Capability List register enumerates the PCI Express Capability structure in the PCI 2.3 configuration space.

Device: 0, 2-7
Function: 0
Offset: 6Ch

Bit	Attr	Default	Description
15:8	RO	00h	**NXTPTR: Next Ptr** This field is set to NULL pointer to terminate the PCI capability list.
7:0	RO	10h	**CAPID: Capability ID** Provides the PCI Express capability ID assigned by PCI-SIG.

3.8.11.2 PEXCAP[7:2, 0] - PCI Express Capabilities Register

The PCI Express Capabilities register identifies the PCI Express device type and associated capabilities.

Device: 0, 2-7
Function: 0
Offset: 6Eh

Bit	Attr	Default	Description
15:14	RV	0h	*Reserved.*
13:9	RO	00h	**IMN: Interrupt Message Number** This field indicates the interrupt message number that is generated from the PCI Express port. When there are more than one MSI interrupt Number, this register field is required to contain the offset between the base Message Data and the MSI Message that is generated when the status bits in the slot status register or root port status registers are set. The chipset is required to update the field if the number of MSI messages changes.
8	if (port 7-2) {RWO} elsif (port 0) {RO} endif	0	**SLOT_Impl: Slot Implemented** 1: indicates that the PCI Express link associated with the port is connected to a slot. 0: indicates no slot is connected to this port. This register bit is of type “write once” and is controlled by BIOS/special initialization firmware. For the DMI port, this value should always be 0b since it is not Hot-Pluggable and it is required for boot. Rest of the PCI_Express ports which are slotted/Hot-Pluggable, BIOS or Software can set this field to enable the slots.
7:4	RO	0100	**DPT: Device/Port Type** This field identifies the type of device. It is set to 0100 as defined in the spec since the PCI Express port is a “root port” in the Intel 5000X Chipset MCH.
3:0	RO	0001	**VERS: Capability Version** This field identifies the version of the PCI Express capability structure. Set to 0001 by PCI SIG.

3.8.11.3 PEXDEVCAP[7:2, 0] - PCI Express Device Capabilities Register

The PCI Express Device Capabilities register identifies device specific information for the port.

Device: 0, 2-7
Function: 0
Offset: 70h

Bit	Attr	Default	Description
31:28	RV	0h	*Reserved.*
27:26	RO	0h	**CSPLS: Captured Slot Power Limit Scale** Specifies the scale used for the Captured Slot Power Limit Value. It does not apply to Intel 5000X Chipset MCH as it is a Root complex. Hardwired to 0h.
25:18	RO	00h	**CSPLV: Captured Slot Power Limit Value** This field specifies upper limit on power supplied by a slot in an upstream port. It does not apply to Intel 5000X Chipset MCH as it is a Root complex. Hardwired to 00h.
17:15	RV	0h	*Reserved*
14	RO	0	**PIPD: Power Indicator Present on Device** This bit when set indicates that a Power Indicator is implemented. 0: PIPD is disabled in Intel 5000X Chipset MCH 1: *Reserved*
13	RO	0	**AIPD: Attention Indicator Present** This bit when set indicates that an Attention Indicator is implemented. 0: AIPD is disabled in Intel 5000X Chipset MCH 1: *Reserved*
12	RO	0	**ABPD: Attention Button Present** This bit when set indicates that an Attention Button is implemented. 0: ABPD is disabled in Intel 5000X Chipset MCH 1: *Reserved*
11:9	RO	111	**EPL1AL: Endpoint L1 Acceptable Latency** This field indicates the acceptable latency that an Endpoint can withstand due to the transition from L1 state to the L0 state. 000: Less than 1µs 001: 1 µs to less than 2 µs 010: 2 µs to less than 4 µs 011: 4 µs to less than 8 µs 100: 8 µs to less than 16 µs 101: 16 µs to less than 32 µs 110: 32 µs to 64 µs 111: More than 64 µs The Intel 5000X Chipset MCH does not support EndpointL1 acceptable latency and is set to the maximum value for safety.
8:6	RO	111	**EPL0AL: Endpoints L0s Acceptable Latency** This field indicates the acceptable latency that an Endpoint can withstand due to the transition from L0s state to the L0 state. 000: Less than 64 ns 001: 64 ns to less than 128 ns 010: 128 ns to less than 256 ns 011: 256 ns to less than 512 ns 100: 512 ns to less than 1 µs 101: 1 µs to less than 2 µs 110: 2 µs to 4 µs 111: More than 4 µs Note that Intel 5000X Chipset MCH does not support L0s implementation and for backup, this field is set to the maximum value.

Device: 0, 2-7 Function: 0 Offset: 70h			
Bit	**Attr**	**Default**	**Description**
5	RO	0	**ETFS: Extended Tag Field Supported** This field indicates the maximum supported size of the Tag field. 0: In the Intel 5000X Chipset MCH, only 5-bit Tag field is supported
4:3	RO	0h	**PFS: Phantom Functions Supported** This field indicates the number of most significant bits of the function number portion of Requester ID in a TLP that are logically combined with the Tag identifier. 0: For root ports, no function number bits for phantom functions are supported
2:0	RO	001	**MPLSS: Max Payload Size Supported** This field indicates the maximum payload size that the PCI Express port can support for TLPs. 001: 256B max payload size Others - *Reserved* Note that the Intel 5000X Chipset MCH only supports up to a maximum of 256B payload (for example, writes, read completions) for each TLP and violations will be flagged as PCI Express errors

3.8.11.4 PEXDEVCTRL[7:2, 0] - PCI Express Device Control Register

The PCI Express Device Control register controls PCI Express specific capabilities parameters associated with this port.

Device: 0, 2-7 Function: 0 Offset: 74h			
Bit	**Attr**	**Default**	**Description**
15	RV	0h	*Reserved.*
14:12	RW	101	**MRRS: Max_Read_Request_Size** This field sets maximum Read Request size generated by the Intel 5000X Chipset MCH. The PCI Express port must not generate read requests with size exceeding the set value. 000: 128 B max read request size 001: 256 B max read request size 010: 512 B max read request size 011: 1024 B max read request size 100: 2048 B max read request size 101: 4096 B max read request size 110: Reserved 111: Reserved The MCH will not generate read requests larger than 64B in general on the outbound side due to the internal Micro-architecture (CPU initiated, DMA or Peer to Peer). Hence the field is set to 000b encoding.
11	RW	1	**ENNOSNP: Enable No Snoop** When set, the PCI Express port is permitted to set the “No Snoop bit” in the Requester Attributes of transactions it initiates that do not require hardware enforced cache coherency. Typically the “No Snoop bit” is set by an originating PCI Express device down in the hierarchy. The Intel 5000X Chipset MCH never sets or modifies the “No snoop bit” in the received TLP even if ENNOSNP is enabled. For outbound traffic, the Intel 5000X Chipset MCH does not need to snoop.

Device: 0, 2-7
Function: 0
Offset: 74h

Bit	Attr	Default	Description
10	RWST	0	**APPME: Auxiliary Power Management Enable** 1: Enables the PCI Express port to draw AUX power independent of PME AUX power. 0: Disables the PCI Express port to draw AUX power independent of PME AUX power. Devices that require AUX power on legacy operating systems should continue to indicate PME AUX power requirements. AUX power is allocated as requested in the AUX_Current field on the Power Management Capabilities Register (PMC), independent of the PMEEN bit in the Power Management Control & Status Register (PMCSR) defined in Section 3.8.9.2.
9	RO	0	**PFEN: Phantom Functions Enable** This bit enables the PCI Express port to use unclaimed functions as Phantom Functions for extending the number of outstanding transaction identifiers. Intel 5000X Chipset MCH does not implement this bit (Root complex) and is hardwired to 0
8	RO	0h	**ETFEN: Extended Tag Field Enable** This bit enables the PCI Express port to use an 8-bit Tag field as a requester. The Intel 5000X Chipset MCH does not use this field (Root complex) and is hardwired to 0.
7:5	RW	001	**MPS: Max Payload Size** This field is set by configuration software for the maximum TLP payload size for the PCI Express port. As a receiver, the Intel 5000X Chipset MCH must handle TLPs as large as the set value. As a transmitter, it must not generate TLPs exceeding the set value. Permissible values that can be programmed are indicated by the Max_Payload_Size_Supported in the Device Capabilities register: 000: 128 B max payload size 001: 256 B max payload size 010: 512 B max payload size 011: 1024 B max payload size 100: 2048 B max payload size 101: 4096 B max payload size others: Reserved Note: The MCH supports max payload sizes only up to 256 B. If Software programs a value that exceeds 256 B for the MPS field, then it will be considered as an error. For receive TLPs, it will be flagged as “unsupported request” and for transmit TLPs, it will be recorded as a Malformed TLP. **Note:** Due to erratum 501664, read completion coalescing cannot be used if MPS=256 B is set by software. Read completion combining up to 128 B would work only if the MPS is set by software. Read completion combining up to 128B would work only if the MPS is set to 128B. See PEXCTRL.COALESCE_EN field.
4	RO	0	**ENRORD: Enable Relaxed Ordering** Intel 5000X Chipset MCH enforces only strict ordering only and hence this bit is initialized to ‘0’
3	RW	0	**URREN: Unsupported Request Reporting Enable** This bit controls the reporting of unsupported requests to the MCH in the PCI Express port. 0: Unsupported request reporting is disabled 1: Unsupported request reporting is enabled Note that the reporting of error messages (such as ERR_CORR, ERR_NONFATAL, ERR_FATAL) received by PCI Express port is controlled exclusively by the PCI Express Root Control register (PEXRTCTRL) described in Section 3.8.11.12.
2	RW	0	**FERE: Fatal Error Reporting Enable** This bit controls the reporting of fatal errors internal to the MCH in the PCI Express port. 0: Fatal error reporting is disabled 1: Fatal error reporting is enabled

Device: 0, 2-7 Function: 0 Offset: 74h			
Bit	**Attr**	**Default**	**Description**
1	RW	0	**NFERE: Non Fatal Error Reporting Enable** This bit controls the reporting of non fatal errors internal to the MCH in the PCI Express port. 0: Non Fatal error reporting is disabled 1: Non Fatal error reporting is enabled
0	RW	0	**CERE: Correctable Error Reporting Enable** This bit controls the reporting of correctable errors internal to the MCH in the PCI Express port. 0: Correctable error reporting is disabled 1: Correctable Fatal error reporting is enabled

3.8.11.5 PEXDEVSTS[7:2, 0] - PCI Express Device Status Register

The PCI Express Device Status register provides information about PCI Express device specific parameters associated with this port.

Device: 0, 2-7 Function: 0 Offset: 76h			
Bit	**Attr**	**Default**	**Description**
15:6	RV	000h	*Reserved.*
5	RO	0h	**TP: Transactions Pending** 1: Indicates that the PCI Express port has issued Non-Posted Requests which have not been completed. 0: A device reports this bit cleared only when all Completions for any outstanding Non-Posted Requests have been received. Since the MCH Root port that do not issue Non-Posted Requests on their own behalf, it is hardwired to 0b.
4	RO	0	**APD: AUX Power Detected** 1- AUX power is detected by the PCI Express port. 0: No AUX power is detected
3	RWC	0	**URD: Unsupported Request Detected** This bit indicates that the device received an Unsupported Request in the PCI Express port. Errors are logged in this register regardless of whether error reporting is enabled or not in the Device Control Register. 1: Unsupported Request detected at the port This records the detection of receiving an unsupported request, error IO2.
2	RWC	0	**FED: Fatal Error Detected** This bit indicates that status of a fatal (uncorrectable) error detected in the PCI Express port. Errors are logged in this register regardless of whether error reporting is enabled or not in the Device Control register. 1: Fatal errors detected 0: No Fatal errors detected
1	RWC	0	**NFED: Non Fatal Error Detected** This bit indicates status of non-fatal errors detected. This bit gets set if a non-fatal uncorrectable error is detected in the PCI Express port. Errors are logged in this register regardless of whether error reporting is enabled or not in the Device Control register. 1: Non Fatal errors detected 0: No non-Fatal Errors detected

Device: 0, 2-7 Function: 0 Offset: 76h			
Bit	**Attr**	**Default**	**Description**
0	RWC	0	**CED: Correctable Error Detected** This bit indicates status of correctable errors detected. This bit gets set if a correctable error is detected in the PCI Express port. Errors are logged in this register regardless of whether error reporting is enabled or not in the PCI Express Device Control register. 1: Correctable errors detected 0: No correctable errors detected

3.8.11.6 PEXLNKCAP[7:2,0] - PCI Express Link Capabilities Register

The Link Capabilities register identifies the PCI Express specific link capabilities.

Device: 0, 2-7 Function: 0 Offset: 78h			
Bit	**Attr**	**Default**	**Description**
31:24	RWO	if (port 0) {0h} elsif (port 2) {02h} elsif (port 3) {03h} elsif (port 4) {04h} elsif (port 5) {05h} elsif (port 6) {06h} elsif (port 7) {07h} endif	**PN: Port Number** This field indicates the PCI Express port number for the link and is initialized by software/BIOS. This will correspond to the device number for each port. port 0- device number of 0 (ESI) port 2 - device number of 2 port 3 - device number of 3 port 4 - device number of 4 port 5- device number of 5 port 6- device number of 6 port 7- device number of 7
23:18	RV	0h	*Reserved.*
17:15	RO	7h	**L1EL: L1 Exit Latency** This field indicates the L1 exit latency for the given PCI Express port. It indicates the length of time this port requires to complete transition from L1 to L0. 000: Less than 1µs 001: 1 µs to less than 2 µs 010: 2 µs to less than 4 µs 011: 4 µs to less than 8 µs 100: 8 µs to less than 16 µs 101: 16 µs to less than 32 µs 110: 32 µs to 64 µs 111: More than 64us The Intel 5000X Chipset MCH does not support L1 acceptable latency and is set to the maximum value for safety

Device: 0, 2-7 Function: 0 Offset: 78h			
Bit	**Attr**	**Default**	**Description**
14:12	RO	7h	**L0sEL: L0s Exit Latency** This field indicates the L0s exit latency (i.e L0s to L0) for the PCI Express port. 000: Less than 64 ns 001: 64 ns to less than 128 ns 010: 128 ns to less than 256 ns 011: 256 ns to less than 512 ns 100: 512 ns to less than 1 µs 101: 1 µs to less than 2 µs 110: 2 µs to 4 µs 111: More than 4 µs Note that Intel 5000X Chipset MCH does not support L0s exit latency implementation and for safety, this field is set to the maximum value.
11:10	RO	01	**ACTPMS: Active State Link PM Support** This field indicates the level of active state power management supported on the given PCI Express port. 00: Disabled 01: L0s Entry Supported 10: Reserved 11: L0s and L1 Supported The Intel 5000X Chipset MCH does not initiate L0s active state Power Management but it does permit a downstream device from placing the link in L0s
9:4	RO	if (port 0,1,3,5,7) {x4} elseif (port 2,6) {x8} elseif (port 4) {x16} endif	**MLW: Maximum Link Width** This field indicates the maximum width of the given PCI Express Link attached to the port. 000001: x1 000100: x4 001000: x8 010000: x16 Others - Reserved See Table 3-35.
3:0	RO	0001	**MLS: Maximum Link Speed** This field indicates the maximum Link speed of the given PCI Express port. 0001: 2.5 Gb/s Others - *Reserved*

Table 3-35. Maximum Link Width Default Value for Different PCI Express Ports

Device / Port	Maximum Link Width	Value
0,3,5,7	x4	000100
2,6	x8	001000
4	x16	010000
	x8	001000

Table 3-35 shows various combining options for PCI Express ports. When ports combine, the control registers for the combined port revert to the lower numbered port. Thus when ports 2 and 3 are combined, the combined x8 port is accessed through port 2 control registers.

3.8.11.7 PEXLNKCTRL[7:2, 0] - PCI Express Link Control Register

The PCI Express Link Control register controls the PCI Express Link specific parameters.

Device: 0, 2-7
Function: 0
Offset: 7Ch

Bit	Attr	Default	Description
15:8	RV	00h	*Reserved.*
7	RW	0	**Ext_Synch: Extended Synch** This bit when set forces the transmission of 4096 FTS ordered sets in the L0s state followed by a single SKP ordered set prior to entering the L0 state, and the transmission of 1024 TS1 ordered sets in the L1 state prior to entering the Recovery state. This mode provides external devices (for example, logic analyzers) monitoring the Link time to achieve bit and Symbol lock before the Link enters the L0 or Recovery states and resumes communication.
6	RW	0	**CCCON**: **Common Clock Configuration** 0: indicates that this PCI Express port and its counterpart at the opposite end of the Link are operating with an asynchronous reference clock. 1: indicates that this PCI Express port and its counterpart at the opposite end of the Link are operating with a distributed common reference clock. Components utilize this common clock configuration information to report the correct L0s and L1 Exit Latencies.
5	WO	0	**RLNK: Retrain Link** This bit, when set, initiates link retraining in the given PCI Express port. It consistently returns 0 when read.
4	RW	0	**LNKDIS: Link Disable** This field indicates whether the link associated with the PCI Express port is enabled or disabled. 0: Enables the link associated with the PCI Express port 1: Disables the link associated with the PCI Express port Software should wait a minimum of 2 ms to make sure the link has entered the electrical idle state before clearing this bit.
3	RO	0	**RCB: Read Completion Boundary** This field defines the read completion boundary for the PCI Express port. Defined encodings for RCB capabilities are: 0: 64 byte 1: 128 byte The Intel 5000X Chipset MCH supports only 64B read completion boundary and is hardwired to 0.
2	RV	0	*Reserved.*
1:0	RW	01	**ASTPMCTRL: Active State Link PM Control** This field controls the level of active state power management supported on the given PCI Express port. 00: Disabled 01: L0s Entry Supported 10: Reserved 11: L0s and L1 Supported Note: This has no effect on the Intel 5000X Chipset MCH.

3.8.11.8 PEXLNKSTS[7:2, 0] - PCI Express Link Status Register

The PCI Express Link Status register provides information on the status of the PCI Express Link such as negotiated width, training, and so forth.

Device: 0, 2-7 Function: 0 Offset: 7Eh			
Bit	**Attr**	**Default**	**Description**
15:13	RV	0h	*Reserved.*
12	RWO	1	**SCCON: Slot Clock Configuration** This bit indicates that the component uses the same physical reference clock that the platform provides on the connector. If the device uses an independent clock irrespective of the presence of a reference on the connector, this bit must be clear. 1: indicates same physical clock in the PCI Express connector as in the platform 0: indicates independent clock on the PCI Express connector from that of the platform. The Intel 5000X Chipset MCH initializes this bit to '1' because the expected state of the platform is to have one clock source shared between the Intel 5000X Chipset MCH component and any down-devices or slot connectors. It is the responsibility of BIOS to be aware of the real platform configuration, and clear this bit if the reference clocks differ.
11	RO	0	**LNKTRG: Link Training** This field indicates the status of an ongoing link training session in the current PCI Express port and is controlled by the Hardware. 0: indicates that the LTSSM is neither in “Configuration” nor “Recovery” states. 1: indicates Link training in progress (Physical Layer LTSSM is in Configuration or Recovery state or the RLNK (retrain link) was set in Section 3.8.11.7 but training has not yet begun. Also refer to the BCTRL.SBUSRESET for details on how the Link training bit can be used for sensing Hot-reset states.
10	RO	0	**TERR: Training Error** This field indicates the occurrence of a Link training error. 0: indicates no Link training error occurred. 1: indicates Link training error occurred.
9:4	RO	000100	**NLNKWD: Negotiated Link Width**[a] This field indicates the negotiated width of the given PCI Express link after training is completed. Only x1, x2, x4, x8, and x16 link width negotiations are possible in the Intel 5000X Chipset MCH. Refer to Table 3-36 for the port and link width assignment after training is completed.
3:0	RO	1h	**LNKSPD: Link Speed** This field indicates the negotiated Link speed of the given PCI Express Link: 0001- 2.5 Gb/s PCI Express link Others - *Reserved*

Notes:

a. The NLNKWD field is set to a default value corresponding to x4 internally within the Intel 5000X Chipset MCH. Note that this field is a don’t care until training is completed for the link. Software should not use this field to determine whether a link is up (enabled) or not.

Table 3-36. Negotiated Link Width For Different PCI Express Ports After Training

Device / Port	Negotiated Link Width	Value
2,3,4,5,6,7	x1	000001
2,3,4,5,6,7	x2	000010
0,2,3,4,5,6,7	x4	000100
2,4,6	x8	001000[a]
4	x16	010000[b]

Notes:

a. Ports 3, 5, and 7 report 000000 as appropriate.

b. Ports 5, 6, and 7 report 000000 as appropriate.

3.8.11.9 PEXSLOTCAP[7:2, 0] - PCI Express Slot Capabilities Register

The Slot Capabilities register identifies the PCI Express specific slot capabilities.

Device: 0, 2-7
Function: 0
Offset: 80h

Bit	Attr	Default	Description
31:19	RWO	0h	**PSN: Physical Slot Number** This field indicates the physical slot number connected to the PCI Express port. It should be initialized to 0 for ports connected to devices that are either integrated on the system board or integrated within the same silicon such as the Root port in Intel 5000X Chipset MCH.
18:17	RV	0h	*Reserved.*
16:15	RWO	0h	**SPLS: Slot Power Limit Scale** This field specifies the scale used for the Slot Power Limit Value. Range of Values: 00: 1.0x 01: 0.1x 10: 0.01x 11: 0.001x
14:7	RWO	00h	**SPLV: Slot Power Limit Value** This field specifies the upper limit on power supplied by slot in conjunction with the Slot Power Limit Scale value defined previously Power limit (in Watts) = SPLS x SPLV
6	RWO	0h	**HPC: Hot-Plug Capable** This field defines hot-plug support capabilities for the PCI Express port 0: indicates that this slot is not capable of supporting Hot-Plug operations. 1: indicates that this slot is capable of supporting Hot-Plug operations
5	RO	0h	**HPS: Hot-Plug Surprise** This field indicates that a device in this slot may be removed from the system without prior notification. 0: Indicates that Hot-Plug surprise is not supported 1: Indicates that Hot-Plug surprise is supported The Intel 5000X Chipset MCH does not support **Hot-Plug surprise feature.**
4	RWO	0h	**PIP: Power Indicator Present** This bit indicates that a Power Indicator is implemented on the chassis for this slot. 0: Indicates that Power Indicator is not present 1: Indicates that Power Indicator is present
3	RWO	0h	**AIP: Attention Indicator Present** This bit indicates that an Attention Indicator is implemented on the chassis for this slot. 0: Indicates that an Attention Indicator is not present 1: Indicates that an Attention Indicator is present
2	RWO	0h	**MRLSP: MRL Sensor Present** This bit indicates that an MRL Sensor is implemented on the chassis for this slot. 0: Indicates that an MRL Sensor is not present 1: Indicates that an MRL Sensor is present
1	RWO	0h	**PCP: Power Controller Present** This bit indicates that a Power Controller is implemented on the chassis for this slot. 0: Indicates that a Power Controller is not present 1: Indicates that a Power Controller is present

Device: 0, 2-7 Function: 0 Offset: 80h			
Bit	**Attr**	**Default**	**Description**
0	RWO	0h	**ABP: Attention Button Present** This bit indicates that an Attention Button is implemented on the chassis for this slot. 0: Indicates that an Attention Button is not present 1: Indicates that an Attention Button is present

3.8.11.10 PEXSLOTCTRL[7:2, 0] - PCI Express Slot Control Register

The Slot Control register identifies the PCI Express specific slot control specific parameters for operations such as Hot-Plug and Power Management. Software issues a command to a Hot-Plug capable Port by issuing a write transaction that targets Slot Control Register fields viz, PWRCTRL, PWRLED, ATNLED described below. A single write to the Slot Control register is considered to be a single command, even if the write affects more than one field in the Slot Control register. In response to this transaction, the port must carry out the requested actions and then set the associated status field (PEXSLOTS.CMDCMP) for the command completed event. The PEXSLOTSTS.CMDCMP bit will be set only when there is a unique change to the state of the PWRCTRL, PWRLED, ATNLED in this register.

Device: 0, 2-7 Function: 0 Offset: 84h			
Bit	**Attr**	**Default**	**Description**
15:11	RV	0h	*Reserved.*
10	RW	0h	**PWRCTRL:** Power Controller Control This bit indicates the current state of the Power applied to the slot of the PCI Express port. 0: Power On 1: Power Off
9:8	RW	0h	**PWRLED: Power Indicator Control** This bit indicates the current state of the Power Indicator of the PCI Express port 00: Reserved. 01: On 10: Blink (The Intel 5000X Chipset MCH drives 1.5 Hz square wave for Chassis mounted LEDs in the case of legacy card form factor for PCI Express devices) 11: Off Default is set to 11b (OFF) When this field is written, the Intel 5000X Chipset MCH sends appropriate POWER_INDICATOR messages through the PCI Express port. For legacy card based PCI Express devices, the event is signaled via the virtual pins1 of the Intel 5000X Chipset MCH, in addition. For PCI Express modules with advanced form factor that incorporate LEDs and onboard decoding logic, the PCI Express messages are interpreted directly (No virtual pins).

Device: 0, 2-7 Function: 0 Offset: 84h			
Bit	**Attr**	**Default**	**Description**
7:6	RW	0h	**ATNLED: Attention Indicator Control** This bit indicates the current state of the Attention Indicator of the PCI Express port 00: *Reserved*. 01: On 10: Blink (The Intel 5000X Chipset MCH drives 1.5 Hz square wave) 11: Off Default is set to 11b (OFF) When this field is written, the Intel 5000X Chipset MCH sends appropriate ATTENTION_INDICATOR messages through the PCI Express port. For legacy card based PCI Express devices, the event is signaled via the virtual pins of the Intel 5000X Chipset MCH, in addition. For PCI Express modules with advanced form factor that incorporate LEDs and onboard decoding logic, the PCI Express messages are interpreted directly (No virtual pins).
5	RW	0h	**HPINTEN: Hot-Plug Interrupt Enable** This field enables the generation of Hot-Plug interrupts and events in the PCI Express port. 0: disables Hot-Plug events and interrupts 1: enables Hot-Plug events and interrupts
4	RW	0h	**CCIEN: Command Completed Interrupt Enable** This field enables the generation of Hot-Plug interrupts when a command is completed by the Hot-Plug controller connected to the PCI Express port 0: Disables Hot-Plug interrupts on a command completion by a Hot-Plug Controller 1: Enables Hot-Plug interrupts on a command completion by a Hot-Plug Controller
3	RW	0h	**PRSINTEN: Presence Detect Changed Enable** This bit enables the generation of Hot-Plug interrupts or wake messages via a presence detect changed event. 0: Disables generation of Hot-Plug interrupts or wake messages when a presence detect changed event happens. 1- Enables generation of Hot-Plug interrupts or wake messages when a presence detect changed event happens.
2	RW	0h	**MRLINTEN: MRL Sensor Changed Enable** This bit enables the generation of Hot-Plug interrupts or wake messages via a MRL Sensor changed event. 0: Disables generation of Hot-Plug interrupts or wake messages when an MRL Sensor changed event happens. 1: Enables generation of Hot-Plug interrupts or wake messages when an MRL Sensor changed event happens.
1	RW	0h	**PWRINTEN:** Power Fault Detected Enable This bit enables the generation of Hot-Plug interrupts or wake messages via a power fault event. 0: Disables generation of Hot-Plug interrupts or wake messages when a power fault event happens. 1: Enables generation of Hot-Plug interrupts or wake messages when a power fault event happens.
0	RW	0h	**ATNINTEN: Attention Button Pressed Enable** This bit enables the generation of Hot-Plug interrupts or wake messages via an attention button pressed event. 0: Disables generation of Hot-Plug interrupts or wake messages when the attention button is pressed. 1: Enables generation of Hot-Plug interrupts or wake messages when the attention button is pressed.

3.8.11.11 PEXSLOTSTS[7:2, 0] - PCI Express Slot Status Register

The PCI Express Slot Status register defines important status information for operations such as Hot-Plug and Power Management.

Device: 0, 2-7
Function: 0
Offset: 86h

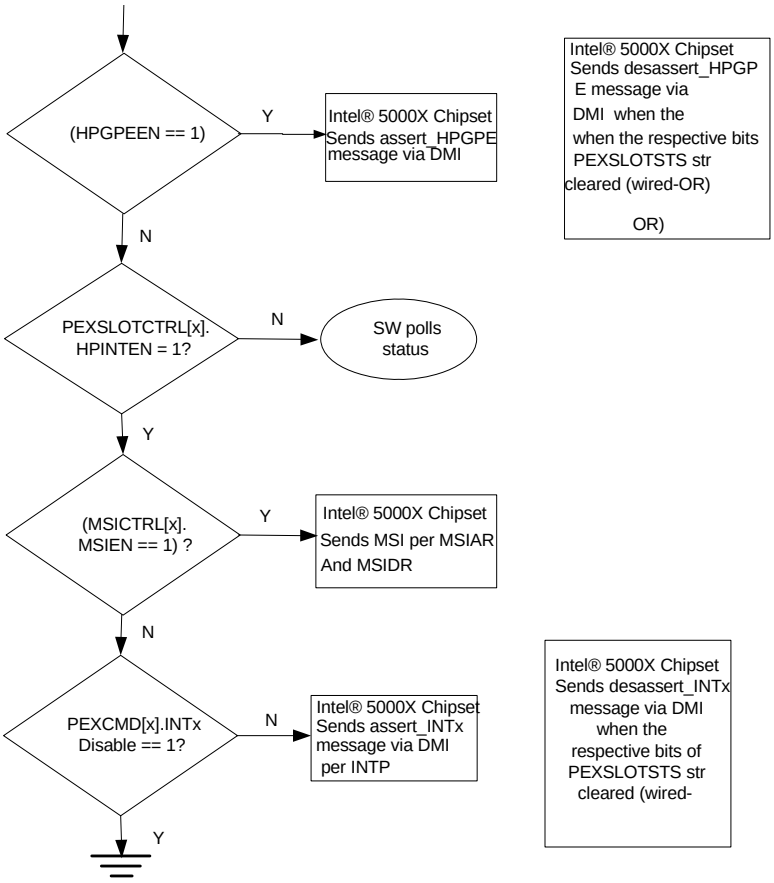
Bit	Attr	Default	Description
15:7	RV	0h	*Reserved.*
6	RO	1h	**PDS: Presence Detect State** This field conveys the Presence Detect status determined via an in-band mechanism or through the Present Detect pins and shows the presence of a card in the slot. 0: Slot Empty 1: Card Present in slot
5	RO	0h	**MRLSS: MRL Sensor State** This bit reports the status of an MRL sensor if it is implemented. 0: MRL Closed 1: MRL Open
4	RWC	0h	**CMDCOMP: Command Completed** This bit is set by the Intel 5000X Chipset MCH when the Hot-Plug controller completes an issued command and is ready to accept a new command. It is subsequently cleared by software after the field has been read and processed.
3	RWC	0h	**PRSINT: Presence Detect Changed** This bit is set by the Intel 5000X Chipset MCH when a Presence Detect Changed event is detected. It is subsequently cleared by software after the field has been read and processed.
2	RWC	0h	**MRLSC: MRL Sensor Changed** This bit is set by the Intel 5000X Chipset MCH when an MRL Sensor Changed event is detected. It is subsequently cleared by software after the field has been read and processed.
1	RWC	0h	**PWRINT: Power Fault Detected** This bit is set by the Intel 5000X Chipset MCH when a power fault event is detected by the power controller. It is subsequently cleared by software after the field has been read and processed.
0	RWC	0h	**ABP: Attention Button Pressed** This bit is set by the Intel 5000X Chipset MCH when the attention button is pressed. It is subsequently cleared by software after the field has been read and processed.

Note that the Assert_intx/Assert_HPGPE message is sent to ESI port when any of the events defined in bits[4:0] (CMDCOMP,PRSINT, MRLSC, PWRINT, ABP) of the PEXSLOTSTS register are set provided the corresponding events in bits [4:0] of the Section 3.8.11.10 and HPINTEN are enabled. Software writes to clear these bits and MCH will send a Deassert_HPGPE message to ESI port (wired-OR).

For the case when MSI is enabled, any new event that sets these bits (for example, ABP, PRSINT, and so forth) will cause an MSI message to be sent to the FSB for each occurrence. that is, each bit is considered unique.

Whereas in the case of Legacy interrupts, a wired-OR approach is used to mimic the level sensitive behavior and only one assert_intx/assert_GPE (deassert_intx/ deassert_GPE) is sent even when multiple interrupt generating bits of the register get set. Refer to Figure 3-5.

Figure 3-5. PCI Express Hot-Plug Interrupt Flow



HPGPEEN	HPINTEN	MSIEN	INTx Disable	Output
1	x	x	x	assert_hpgpe
0	1	1	x	MSI
0	1	0	0	assert_intx
0	1	0	1	--
0	0	x	x	--

3.8.11.12 PEXRTCTRL[7:2, 0] - PCI Express Root Control Register

The PCI Express Root Control register specifies parameters specific to the root complex port.

Device: 0, 2-7
Function: 0
Offset: 88h

Bit	Attr	Default	Description
15:4	RV	0h	*Reserved.*
3	RW	0h	**PMEINTEN: PME Interrupt Enable** This field controls the generation of interrupts for PME messages. 1: Enables interrupt generation upon receipt of a PME message as reflected in the PME Status bit defined in the PEXRTSTS register. A PME interrupt is generated if the PMESTATUS register bit defined in Section 3.8.11.13, is set when this bit is set from a cleared state. 0: Disables interrupt generation for PME messages.
2	RW	0h	**SEFEEN: System Error on Fatal Error Enable** This field controls generation of system errors in the PCI Express port hierarchy for fatal errors. 1: Indicates that a System Error should be generated if a fatal error (ERR_FATAL) is reported by any of the devices in the hierarchy associated with and including this PCI Express port. 0: No System Error should be generated on a fatal error (ERR_FATAL) reported by any of the devices in the hierarchy.
1	RW	0h	**SENFEEN: System Error on Non-Fatal Error Enable** This field controls generation of system errors in the PCI Express port hierarchy for non-fatal errors. 1: Indicates that a System Error should be generated if a non-fatal error (ERR_NONFATAL) is reported by any of the devices in the hierarchy associated with and including this PCI Express port. 0: No System Error should be generated on a non-fatal error (ERR_NONFATAL) reported by any of the devices in the hierarchy.
0	RW	0h	**SECEEN: System Error on Correctable Error Enable** This field controls generation of system errors in the PCI Express port hierarchy for correctable errors. 1: Indicates that a System Error should be generated if a correctable error (ERR_COR) is reported by any of the devices in the hierarchy associated with and including this PCI Express port 0: No System Error should be generated on a correctable error (ERR_COR) reported by any of the devices in the hierarchy associated with and including this PCI Express port.

3.8.11.13 PEXRTSTS[7:2, 0] - PCI Express Root Status Register

The PCI Express Root Status register specifies parameters specific to the root complex port.

Device: 0, 2-7
Function: 0
Offset: 8Ch

Bit	Attr	Default	Description
31:18	RV	0h	*Reserved.*
17	RO	0h	**PMEPEND: PME Pending** This field indicates that another PME is pending when the PME Status bit is set. When the PME Status bit is cleared by software; the pending PME is delivered by hardware by setting the PME Status bit again and updating the Requestor ID appropriately. The PME pending bit is cleared by hardware if no more PMEs are pending. ***Note:*** The Intel 5000X Chipset MCH can handle two outstanding PM_PME messages in its internal queues of the Power Management controller per port. If the downstream device issues more than 2 PM_PME messages successively, it will be dropped.
16	RWC	0h	**PMESTATUS: PME Status**[a] This field indicates status of a PME that is underway in the PCI Express port. 1: PME was asserted by a requester as indicated by the PMEREQID field This bit is cleared by software by writing a '1'. Subsequent PMEs are kept pending until the PME Status is cleared.
15:0	RO	0000h	**PMEREQID: PME Requester ID** This field indicates the PCI requester ID of the last PME requestor.

Notes:

a. PMEINTEN defined in PEXRTCTRL has to be set for PM interrupts to be generated. For non-MSI PM interrupts, the PMESTATUS bit in each of the PEXRTSTS[2:7] registers are wired OR together and when set, the MCH will send the "Assert_PMEGPE" message to the Intel 631xESB/632xESB I/O Controller Hub for power management. When all the bits are clear, it will send the "Deassert_PMEGPE" message. PMEINTEN defined in PEXRTCTRL has to be set for PM interrupts to be generated. PM_PME events that generate MSI will depend on the MSIEN field in Section 3.8.10.3. Refer to the PM interrupt flow in Power Management Chapter.

3.8.11.14 ESICTRL[0] - ESI Control Register

The ESICTRL register holds control information and defeature bits pertaining to the ESI interface for power management.

Device: 0
Function: 0
Offset: D4h

Bit	Attr	Default	Description
31:15	RV	0h	*Reserved.*
14	RW	1	**DL23R: Override L23 Ready - Recommend setting this bit to 1.** **0:** Wait for PME_Enter_L23 on all PCIe ports **1:** Do not wait for PME_Enter_L23 on all PCIe ports
13:12	RV	0	***Reserved***
11	RWC	0	**PTE: PME_TO_Ack Time Expired** 0: Default mode, “PME_TO_Ack” message received on all PCI Express ports before timeout. 1: Signal that time expiration has occurred when the PTOV field described below crosses the threshold in the Intel 5000X Chipset MCH.
10:9	RW	0h	**PTOV: PME_TO_Ack Time Out Value** 00: 1 ms (default) 01: 10 ms 10: 50 ms 11: *Reserved* This register field provides the timer limit for the Intel 5000X Chipset MCH to keep track of the elapsed time from sending “PME_Turn_off” to receiving a “PME_TO_Ack”.
8:4	RV	0h	*Reserved.*
3:0	RW	0h	**SAC: STOPGRANT ACK COUNT** This field tracks the number of Stop Grant acks received from the FSBs. The MCH will forward the last StopGrantAck received from the FSB to the Intel 631xESB/632xESB I/O Controller Hub using the “Req_C2” command. Software is expected to set this field to “THREADs-1” where the variable “THREAD” is the total number of logical threads present in the system (currently can handle up to 16). Typically each CPU thread will issue a StopGrantAck in response to a STPCLK# assertion from the Intel 631xESB/ 632xESB I/O Controller Hub. When the final StopGrantAck is received from the FSB and the internal counter hits the value of SAC+1 (which is equal to THREAD), the MCH will initiate the “Req_C2” command on the DMI. It is illegal for the CPU to send more Stop Grant Acks than that specified in the “THREAD” variable. ***Note:*** For Sx Power management in H/W or S/W mode

3.8.12 PCI Express Advanced Error Reporting Capability

3.8.12.1 PEXENHCAP[7:2, 0] - PCI Express Enhanced Capability Header

This register identifies the capability structure and points to the next structure.

Device: 0, 2-7
Function: 0
Offset: 100h

Bit	Attr	Default	Description
31:20	RO	140h	**NCAPOFF: Next Capability Offset** This field points to the next Capability in extended configuration space.

Device: 0, 2-7 Function: 0 Offset: 100h			
Bit	**Attr**	**Default**	**Description**
19:16	RO	1h	**CV: Capability Version** Set to 1h for this version of the PCI Express logic
15:0	RO	0001h	**PEXCAPID: PCI Express Extended CAP_ID** Assigned for advanced error reporting

3.8.12.2 UNCERRSTS[7:2] - Uncorrectable Error Status

This register identifies uncorrectable errors detected for the PCI Express Port. If an error occurs and is unmasked in the detect register (EMSAK_UNCOR_PEX), the appropriate error bit will be recorded in this register. If an error is recorded in the UNCERRSTS register and the appropriate bit (along with the severity bit of the UNCERRSEV register) determines which bit in the PEX_FAT_FERR, PEX_NF_COR_FERR, PEX_FAT_NERR, PEX_NF_COR_NERR register gets recorded.These error log registers are described starting from Section 3.8.12.24.

Device: 2-7 Function: 0 Offset: 104h			
Bit	**Attr**	**Default**	**Description**
31:21	RV	0h	Reserved
20	RWCST	0	**IO2Err: Received an Unsupported Request**
19	RV	0	Reserved
18	RWCST	0	**IO9Err: Malformed TLP Status**
17	RWCST	0	**IO10Err: Receiver Buffer Overflow Status**
16	RWCST	0	**IO8Err: Unexpected Completion Status**
15	RWCST	0	**IO7Err: Completer Abort Status**
14	RWCST	0	**IO6Err: Completion Time-out Status**
13	RWCST	0	**IO5Err: Flow Control Protocol Error Status**
12	RWCST	0	**IO4Err: Poisoned TLP Status**
11:6	RV	0h	Reserved
5	RWST	0	**IO19Err: Surprise Link Down Error Status**
4	RWCST	0	**IO0Err: Data Link Protocol Error Status**
3:1	RV	0h	Reserved
0	RWCST	0	**IO3Err:Training Error Status** This field should not be used for obtaining Training error status due to a recent *PCI Express Base Specification*, Revision 1.0a Errata Dec 2003 to remove training error.

3.8.12.3 UNCERRSTS[0] - Uncorrectable Error Status For ESI Port

This register identifies uncorrectable errors detected on ESI Port. If an error occurs and is unmasked in the detect register (EMASK_UNCOR_PEX), the appropriate error bit will be recorded in this register. If an error is recorded in the UNCERRSTS register and the appropriate bit (along with the severity bit of the UNCERRSEV register) determines which bit in the PEX_FAT_FERR, PEX_NF_COR_FERR, PEX_FAT_NERR, PEX_NF_COR_NERR registers get recorded. These error log registers are described starting from Section 3.8.12.24.

Device: 0 Function: 0 Offset: 104h			
Bit	**Attr**	**Default**	**Description**
31:22	RV	0h	Reserved
21	RWCST	0	**IO18Err: ESI Reset time-out**
20	RWCST	0	**IO2Err: Received an Unsupported Request**
19	RV	0	Reserved
18	RWCST	0	**IO9Err: Malformed TLP Status**
17	RWCST	0	**IO10Err: Receiver Buffer Overflow Status**
16	RWCST	0	**IO8Err: Unexpected Completion Status**
15	RWCST	0	**IO7Err: Completer Abort Status**
14	RWCST	0	**IO6Err: Completion Time-out Status**
13	RWCST	0	**IO5Err: Flow Control Protocol Error Status**
12	RWCST	0	**IO4Err: Poisoned TLP Status**
11:6	RV	0h	Reserved
5	RWST	0	**IOErr: Surprise Link Down Error Status**
4	RWCST	0	**IO0Err: Data Link Protocol Error Status**
3:1	RV	0h	Reserved
0	RWCST	0	**IO3Err:Training Error Status** ***Note:*** This field should not be used for obtaining Training error status due to a recent *PCI Express Base Specification*, Revision 1.0a Errata Dec 2003 to remove training error. Hardware behavior is undefined.

3.8.12.4 UNCERRMSK[7:2] - Uncorrectable Error Mask

This register masks uncorrectable errors from the UNCERRSTS[2:7] register from being signaled.

Device: 2-7 Function: 0 Offset: 108h			
Bit	**Attr**	**Default**	**Description**
31:21	RV	0h	Reserved
20	RWST	0	**IO2Msk: Received an Unsupported Request**
19	RV	0	Reserved
18	RWST	0	**IO9Msk: Malformed TLP Status**
17	RWST	0	**IO10Msk: Receiver Buffer Overflow Mask**
16	RWST	0	**IO8Msk: Unexpected Completion Mask**
15	RWST	0	**IO7Msk: Completer Abort Status**
14	RWST	0	**IO6Msk: Completion Time-out Mask**
13	RWST	0	**IO5Msk: Flow Control Protocol Error Mask**
12	RWST	0	**IO4Msk: Poisoned TLP Mask**
11:6	RV	0h	Reserved
5	RWST	0	**IO19Msk: Surprise Link Down Error Mask**

Device: 2-7 Function: 0 Offset: 108h			
Bit	**Attr**	**Default**	**Description**
4	RWST	0	**IO0Msk: Data Link Layer Protocol Error Mask**
3:1	RV	000	Reserved
0	RWST	0	**IO3Msk:Training Error Mask** ***Note:*** This field should not be used for setting Training error Mask due to a recent *PCI Express Base Specification*, Revision 1.0a Errata Dec 2003 to remove training error. Hardware behavior is undefined.

3.8.12.5 UNCERRMSK[0] - Uncorrectable Error Mask For ESI Port

This register masks uncorrectable errors from the UNCERRSTS[0] register (ESI port) from being signaled.

Device: 0 Function: 0 Offset: 108h			
Bit	**Attr**	**Default**	**Description**
31:22	RV	0h	Reserved
21	RWST	0	**IO18Msk: ESI Reset time-out**
20	RWST	0	**IO2Msk: Received an Unsupported Request**
19	RV	0	Reserved
18	RWST	0	**IO9Msk: Malformed TLP Status**
17	RWST	0	**IO10Msk: Receiver Buffer Overflow Mask**
16	RWST	0	**IO8Msk: Unexpected Completion Mask**
15	RWST	0	**IO7Msk: Completer Abort Status**
14	RWST	0	**IO6Msk: Completion Time-out Mask**
13	RWST	0	**IO5Msk: Flow Control Protocol Error Mask**
12	RWST	0	**IO4Msk: Poisoned TLP Mask**
11:5	RV	0h	Reserved
5	RWST	0	**IO19Msk: Surprise Link Down Error Mask**
4	RWST	0	**IO0Msk: Data Link Layer Protocol Error Mask**
3:1	RV	000	Reserved
0	RWST	0	**IO3Msk:Training Error Mask** This field should not be used for setting Training error Mask due to a recent *PCI Express Base Specification*, Revision 1.0a Errata Dec 2003 to remove training error. Hardware behavior is undefined.

3.8.12.6 UNCERRSEV[0] - Uncorrectable Error Severity For ESI Port

This register indicates the severity of the uncorrectable errors for the ESI port. An error is reported as fatal when the corresponding error bit in the severity register is set. If the bit is cleared, the corresponding error is considered non-fatal. If an error is recorded in the UNCERRSTS register, the corresponding bit of UNCERRSEV determines if the error gets reflected as a device fatal or nonfatal error in the PEX_FAT_FERR, PEX_NF_COR_FERR, PEX_FAT_NERR, PEX_NF_COR_NERR registers.

Device: 0 Function: 0 Offset: 10Ch			
Bit	**Attr**	**Default**	**Description**
31:22	RV	0h	Reserved
21	RWST	1	**IO18Severity: ESI Reset time-out**
20	RWST	0	**IO2Severity: Received an Unsupported Request**
19	RV	0	Reserved
18	RWST	1	**IO9Severity: Malformed TLP Severity**
17	RWST	1	**IO10Severity: Receiver Buffer Overflow Severity**
16	RWST	0	**IO8Severity: Unexpected Completion Severity**
15	RWST	0	**IO7Severity: Completer Abort Status**
14	RWST	0	**IO6Severity: Completion Time-out Severity**
13	RWST	1	**IO5Severity: Flow Control Protocol Error Severity**
12	RWST	0	**IO4Severity: Poisoned TLP Severity**
11:6	RV	0h	Reserved
5	RWST	0	**IO19 Severity: Surprise Link Down Severity**
4	RWST	1	**IO0Severity: Data Link Protocol Error Severity** (See Figure 3-17 in *PCI Express Base Specification*, Revision 1.0a)
3:1	RV	000	Reserved
0	RWST	1	**IO3Severity:Training Error Severity** This field should not be used for setting Training error severity due to a recent *PCI Express Base Specification*, Revision 1.0a Errata Dec 2003 to remove training error. Hardware behavior is undefined.

3.8.12.7 UNCERRSEV[7:2] - Uncorrectable Error Severity

This register indicates the severity of the uncorrectable errors. An error is reported as fatal when the corresponding error bit in the severity register is set. If the bit is cleared, the corresponding error is considered non-fatal. If an error is recorded in the UNCERRSTS register, the appropriate bit of UNCERRSEV determines if the error gets reflected as a device fatal or nonfatal error in the PEX_FAT_FERR, PEX_NF_COR_FERR, PEX_FAT_NERR, PEX_NF_COR_NERR registers.

Device: 2-7 Function: 0 Offset: 10Ch			
Bit	**Attr**	**Default**	**Description**
31:21	RV	0h	Reserved
20	RWST	0	**IO2Severity: Received an Unsupported Request**
19	RV	0	Reserved
18	RWST	1	**IO9Severity: Malformed TLP Severity**
17	RWST	1	**IO10Severity: Receiver Buffer Overflow Severity**
16	RWST	0	**IO8Severity: Unexpected Completion Severity**
15	RWST	0	**IO7Severity: Completer Abort Status**
14	RWST	0	**IO6Severity: Completion Time-out Severity**
13	RWST	1	**IO5Severity: Flow Control Protocol Error Severity**

Device: 2-7 Function: 0 Offset: 10Ch			
Bit	**Attr**	**Default**	**Description**
12	RWST	0	**IO4Severity: Poisoned TLP Severity**
11:6	RV	0h	Reserved
5	RWST	0	**IO19Severity: Surprise Link Down Severity**
4	RWST	1	**IO0Severity: Data Link Protocol Error Severity** (See Figure 3-17 in *PCI Express Base Specification*, Revision 1.0a)
3:1	RV	000	Reserved
0	RWST	1	**IO3Severity:Training Error Severity** This field should not be used for setting Training error severity due to a recent *PCI Express Base Specification*, Revision 1.0a Errata Dec 2003 to remove training error. Hardware behavior is undefined.

3.8.12.8 CORERRSTS[7:2, 0] - Correctable Error Status

This register identifies which unmasked correctable error has been detected. The error is directed to the respective device correctable error bit in the PEX_NF_COR_FERR, PEX_NF_COR_NERR registers (If the error is unmasked in the CORERRMSK register defined in Section 3.8.12.9). These registers are discussed starting from Section 3.8.12.25.

Device: 0, 2-7 Function: 0 Offset: 110h			
Bit	**Attr**	**Default**	**Description**
31:13	RV	0h	Reserved
12	RWCST	0	**IO16Err: Replay Timer Time-out Status**
11:9	RV	0h	Reserved
8	RWCST	0	**IO15Err: Replay_Num Rollover Status**
7	RWCST	0	**IO14Err: Bad DLLP Status**
6	RWCST	0	**IO13Err: Bad TLP Status**
5:1	RV	0h	Reserved
0	RWCST	0	**IO12Err: Receiver Error Status**

3.8.12.9 CORERRMSK[7:2, 0] - Correctable Error Mask

This register masks correctable errors from being signalled. They are still logged in the CORERRSTS register.

Device: 0, 2-7 Function: 0 Offset: 114h			
Bit	**Attr**	**Default**	**Description**
31:13	RV	0h	Reserved
12	RWST	0	**IO16Msk: Replay Timer Time-out Mask**
11:9	RV	0h	Reserved
8	RWST	0	**IO15Msk: Replay_Num Rollover Mask**

Device: 0, 2-7 Function: 0 Offset: 114h			
Bit	**Attr**	**Default**	**Description**
7	RWST	0	**IO14Msk: Bad DLLP Mask**
6	RWST	0	**IO13Msk: Bad TLP Mask**
5:1	RV	0h	Reserved
0	RWST	0	**IO12Msk: Receiver Error Mask**

3.8.12.10 AERRCAPCTRL[7:2, 0] - Advanced Error Capabilities and Control Register

This register identifies the capability structure and points to the next structure.

Device: 0, 2-7 Function: 0 Offset: 118h			
Bit	**Attr**	**Default**	**Description**
31:9	RV	0h	Reserved
8	RO	0	**ECRCCHKEN: ECRC Check Enable** This bit when set enables ECRC checking.
7	RO	0	**ECRCCHKCAP: ECRC Check Capable** Intel 5000X Chipset MCH does not support ECRC.
6	RO	0	**ECRCGENEN: ECRC Generation Enable** Intel 5000X Chipset MCH does not generate ECRC.
5	RO	0	**ECRCGENCAP: ECRC Generation Capable** Intel 5000X Chipset MCH does not generate ECRC.
4:0	ROST	0h	**FERRPTR: First error pointer** The First Error Pointer is a read-only register that identifies the bit position of the first error reported in the Uncorrectable Error status register. Left most error bit if multiple bits occurred simultaneously.

3.8.12.11 HDRLOG0[7:2, 0] - Header Log 0

This register contains the first 32 bits of the header log locked down when the first uncorrectable error occurs. Headers of the subsequent errors are not logged.

Device: 0, 2-7 Function: 0 Offset: 11Ch			
Bit	**Attr**	**Default**	**Description**
31:0	ROST	0h	**HDRLOGDW0:** Header of TLP (DWORD 0) associated with first uncorrectable error

3.8.12.12 HDRLOG1[7:2, 0] - Header Log 1

This register contains the second 32 bits of the header log.

Device: 0, 2-7 Function: 0 Offset: 120h			
Bit	**Attr**	**Default**	**Description**
31:0	ROST	0h	**HDRLOGDW1:** Header of TLP (DWORD 1) associated with error

3.8.12.13 HDRLOG2[7:2, 0] - Header Log 2

This register contains the third 32 bits of the header log.

Device: 0, 2-7 Function: 0 Offset: 124h			
Bit	**Attr**	**Default**	**Description**
31:0	ROST	0h	**HDRLOGDW2:** Header of TLP (DWORD 2) associated with error

3.8.12.14 HDRLOG3[7:2, 0] - Header Log 3

This register contains the fourth 32 bits of the header log.

Device: 0, 2-7 Function: 0 Offset: 128h			
Bit	**Attr**	**Default**	**Description**
31:0	ROST	0h	**HDRLOGDW3:** Header of TLP (DWORD 3) associated with error

3.8.12.15 RPERRCMD[7:2, 0] - Root Port Error Command

This register controls behavior upon detection of errors.

Device: 0, 2-7 Function: 0 Offset: 12Ch			
Bit	**Attr**	**Default**	**Description**
31:8	RV	0h	Reserved
7:3	**RV**	**0h**	Reserved
2	**RW**	**0**	**EN_FAT_ERR: FATAL Error Reporting Enable** Enable interrupt on fatal errors when set.
1	**RW**	**0**	**EN_NONFAT_ERR: Non-FATAL Error Reporting Enable** Enable interrupt on a non-fatal (uncorrectable) error when set
0	**RW**	**0**	**EN_CORR_ERR: Correctable Error Reporting Enable** Enable interrupt on correctable errors when set

3.8.12.16 RPERRSTS[7:2, 0] - Root Error Status Register

The Root Error Status register reports status of error messages (ERR_COR, ERR_NONFATAL, and ERR_FATAL) received by the Root Complex in the MCH, and errors detected by the Root Port itself (which are treated conceptually as if the Root Port had sent an error message to itself). The ERR_NONFATAL and ERR_FATAL messages are grouped together as uncorrectable. Each correctable and uncorrectable (Non-fatal and

Fatal) error source has a first error bit and a next error bit associated with it respectively. When an error is received by a Root Complex, the respective first error bit is set and the Requestor ID is logged in the Error Source Identification register. A set individual error status bit indicates that a particular error category occurred; software may clear an error status by writing a 1 to the respective bit. If software does not clear the first reported error before another error message is received of the same category (correctable or uncorrectable), the corresponding next error status bit will be set but the Requestor ID of the subsequent error message is discarded. The next error status bit may be cleared by software by writing a 1 to the respective bit as well. This register is updated regardless of the settings of the Root Control register in Section 3.8.11.12 and the Root Error Command register defined in Section 3.8.12.15.

Device: 0, 2-7
Function: 0
Offset: 130h

Bit	Attr	Default	Description
31:27	RO	0h	**ADVERR_INT_MSG_NUM:** Advanced Error Interrupt Message Number Advanced Error Interrupt Message Number offset between base message data an the MSI message if assigned more than one message number to be used of any status in this capability.
26:7	RV	0h	Reserved
6	RWCST	0	**FAT_ERR_Rcvd:** Fatal Error Messages Received Set when one or more Fatal Uncorrectable error Messages[a] have been received.
5	RWCST	0	**NFAT_ERR_Rcvd**: Non-Fatal Error Messages Received Set when one or more Non-Fatal Uncorrectable error Messages have been received.
4	RWCST	0	**FRST_UNCOR_FATAL:** First Uncorrectable Fatal Set when the first Uncorrectable error message (which is FATAL) is received.
3	RWCST	0	**MULT_ERR_NOFAT_ERR:** Multiple ERR_FATAL NO FATAL_Received Set when either a fatal or a non-fatal error message is received and ERR_FAT_NONFAT_RCVD is already set, that is, log from the 2nd Fatal or No fatal error message onwards
2	RWCST	0	**ERR_FAT_NOFAT_RCVD:** ERROR FATAL NOFATAL Received Set when either a fatal or a non-fatal error message is received and this bit is already not set. that is, log the first error message
1	RWCST	0	**MULT_ERR_COR_RCVD:** Multiple Correctable Error Received Set when either a correctable error message is received and ERR_CORR_RCVD is already set, that is, log from the 2nd Correctable error message onwards.
0	RWCST	0	**ERR_CORR_RCVD:** First Correctable Error Received Set when a correctable error message is received and this bit is already not set. that is, log the first error message

Notes:

a. This applies to both internal generated Root port errors and those messages received from an external source.

3.8.12.17 RPERRSID[7:2, 0] - Error Source Identification Register

The Error Source Identification register identifies the source (Requestor ID) of first correctable and uncorrectable (Non-fatal/Fatal) errors reported in the Root Error Status register defined in Section 3.8.12.16. This register is updated regardless of the settings of the Root Control register defined in Section 3.8.11.12 and the Root Error Command register defined in.Section 3.8.12.15.

Device: 0, 2-7 Function: 0 Offset: 134h			
Bit	**Attr**	**Default**	**Description**
31:16	ROST	0h	**ERR_FAT_NOFAT_SID: Fatal No Fatal Error Source ID** Requestor ID of the source when an Fatal or No Fatal error is received and the **ERR_FAT_NOFAT_RCVD bit** is not already set. That is, log ID of the first Fatal or Non Fatal error.
15:0	ROST	0h	**ERR_CORR_SID:** Correctable Error Source ID Requestor ID of the source when a correctable error is received and the **ERR_CORR_RCVD** is not already set. That is, log ID of the first correctable error.

3.8.12.18 Intel 5000X Chipset MCH SPCAPID[7:2, 0] - MCH Specific Capability ID

This register identifies the capability structure and points to the next structure.

Device: 0, 2-7 Function: 0 Offset: 140h			
Bit	**Attr**	**Default**	**Description**
31:20	RO	0h	**NXTCAPOFF:** Next Capability Offset This field points to the next Capability in extended configuration space. It is set 000h since this is the final structure in the chain.
19:16	RO	0h	**VN:** Version Number Version number for this capability structure
15:0	RO	0h	**EXTCAPID: Extended CAP_ID**

3.8.12.19 PEX_ERR_DOCMD[7:2, 0] - PCI Express Error Do Command Register

Link Error Commands for doing the various signaling: ERR[2:0] and MCERR.

Device: 0, 2-7 Function: 0 Offset: 144h			
Bit	**Attr**	**Default**	**Description**
31:8	RV	0h	*Reserved*.
7:6	RW	00	**PEX_RP_FAT_MAP: Root Port steering for fatal errors** 00: ERR[0] 01: ERR[1] 10: ERR[2] 11: MCERR The Root Port Fatal errors are routed to one of the ERR[2:0] pins or MCERR.
5:4	RW	00	**PEX_RP_NF_MAP: Root Port steering for non-fatal errors** 00: ERR[0], 01: ERR[1] 10: ERR[2] 11: MCERR The Root Port Non Fatal (uncorrectable) errors are routed to one of the ERR[2:0] pins or MCERR.

Device: 0, 2-7 Function: 0 Offset: 144h			
Bit	**Attr**	**Default**	**Description**
3:2	RW	00	**PEX_RP_CORR_MAP: Root Port steering for correctable errors** 00: ERR[0], 01: ERR[1] 10: ERR[2] 11: MCERR The Root Port correctable errors are routed to one of the ERR[2:0] pins or MCERR.
1:0	RW	00	**PEX_DEV_UNSUP_MAP:** Report steering for unsupported request errors (master aborts) for legacy devices 00: ERR[0] 01: ERR[1] 10: ERR[2] 11: MCERR Unsupported request error report enable is in the Device control register. This is Error IO2.

3.8.12.20 EMASK_UNCOR_PEX[0] - Uncorrectable Error Detect Mask For ESI

This register masks (blocks) the detection of the selected error bits for the ESI port. When a specific error is blocked, it does NOT get reported or logged.

Device: 0 Function: 0 Offset: 148h			
Bit	**Attr**	**Default**	**Description**
31:22	RV	0h	Reserved
21	RW	0	**IO18DMIRstDetMsk:** ESI Reset time-out
20	RW	0	**IO2DetMsk:** Received an Unsupported Request
19	RV	0	Reserved
18	RW	0	**IO9DetMsk:** Malformed TLP Status
17	RW	0	**IO10DetMsk:** Receiver Buffer Overflow Status
16	RW	0	**IO8DetMsk:** Unexpected Completion Status
15	RW	0	**IO7DetMsk:** Completer Abort Status
14	RW	0	**IO6DetMsk:** Completion Time-out Status
13	RW	0	**IO5DetMsk:** Flow Control Protocol Error Status
12	RW	0	**IO4DetMsk:** Poisoned TLP Status
11:5	RV	0h	Reserved
4	RW	0	**IO0DetMsk:** Data Link Protocol Error Status
3:1	RV	0h	Reserved
0	RW	0	**IO3DetMsk:**Training Error Status This field should not be used for setting Training error severity due to a recent PCI-SIG ECN (Jan 22, 04) to remove training error. Hardware behavior is undefined.

3.8.12.21 EMASK_UNCOR_PEX[7:2] - Uncorrectable Error Detect Mask

This register masks (blocks) the detection of the selected error bits. When a specific error is blocked, it does NOT get reported or logged.

Device: 2-7
Function: 0
Offset: 148h

Bit	Attr	Default	Description
31:21	RV	0h	Reserved
20	RW	0	**IO2DetMsk:** Received an Unsupported Request
19	RV	0	Reserved
18	RW	0	**IO9DetMsk:** Malformed TLP Status
17	RW	0	**IO10DetMsk:** Receiver Buffer Overflow Status
16	RW	0	**IO8DetMsk**: Unexpected Completion Status
15	RW	0	**IO7DetMsk:** Completer Abort Status
14	RW	0	**IO6DetMsk**: Completion Time-out Status
13	RW	0	**IO5DetMsk:** Flow Control Protocol Error Status
12	RW	0	**IO4DetMsk:** Poisoned TLP Status
11:6	RV	0h	Reserved
5	RW	0	IO19DetMsk: Surprise Link Down Mask
4	RW	0	**IO0DetMsk:** Data Link Protocol Error Status
3:1	RV	0h	Reserved
0	RW	0	**IO3DetMsk:**Training Error Status This field should not be used for setting Training error severity due to a recent PCI-SIG ECN (Jan 22, 04) to remove training error. Hardware behavior is undefined.

3.8.12.22 EMASK_COR_PEX[7:2, 0] - Correctable Error Detect Mask

This register masks (blocks) the detection of the selected bits. Normally all are detected. But software can choose to disable detecting any of the error bits.

Device: 0, 2-7
Function: 0
Offset: 14Ch

Bit	Attr	Default	Description
31:13	RV	0h	Reserved
12	RW	0	**IO16DetMsk:** Replay Timer Time-out Mask
11:9	RV	0h	Reserved
8	RW	0	**IO15DetMsk:** Replay_Num Rollover Mask
7	RW	0	**IO14DetMsk:** Bad DLLP Mask
6	RW	0	**IO13DetMsk:** Bad TLP Mask
5:1	RV	0h	Reserved
0	RW	0	**IO12DetMsk:** Receiver Error Mask

3.8.12.23 EMASK_RP_PEX[7:2, 0] - Root Port Error Detect Mask

This register masks (blocks) the detection of the selected bits associated with the root port errors. Normally, all are detected.

Device: 0, 2-7
Function: 0
Offset: 150h

Bit	Attr	Default	Description
31:3	RV	0h	Reserved
2	RW	0	**IO1DetMsk:** Fatal Message Detect Mask
1	RW	0	**IO11DetMsk:** Uncorrectable Message Detect Mask
0	RW	0	**IO17DetMsk:** Correctable Message Detect Mask

3.8.12.24 PEX_FAT_FERR[7:2, 0] - PCI Express First Fatal Error Register

This register records the occurrence of the first unmasked PCI Express FATAL errors and written by the MCH if the respective bits are not set prior. The classification of uncorrectable errors into FATAL is based on the severity level of the UNCERRSEV register described in Section 3.8.12.7.

Device: 0, 2-7
Function: 0
Offset: 154h

Bit	Attr	Default	Description
31:13	RV	0h	Reserved
12	RWCST	0	FIRST_FAT_Err_IO19: Surprise Link Down Status
11	RWCST	0	**First_FAT_Err_IO18**: ESI Reset time-out
10	RWCST	0	**First_FAT_Err_IO9:** PEX - Malformed TLP
9	RWCST	0	**First_FAT_Err_IO10:** PEX - Receive Buffer Overflow Error
8	RWCST	0	**First_FAT_Err_IO8:** PEX - Unexpected Completion Error

Device: 0, 2-7 Function: 0 Offset: 154h			
Bit	**Attr**	**Default**	**Description**
7	RWCST	0	**First_FAT_Err_IO7:** PEX - Completer Abort
6	RWCST	0	**First_FAT_Err_IO6:** PEX - Completion Timeout
5	RWCST	0	**First_FAT_Err_IO5:** PEX - Flow Control Protocol Error
4	RWCST	0	**First_FAT_Err_IO4:** PEX - Poisoned TLP
3	RWCST	0	**First_FAT_Err_IO3:** PEX - Training Error This field should not be used for setting Training error severity due to a recent PCI-SIG ECN (Jan 22, 04) to remove training error. Hardware behavior is undefined.
2	RWCST	0	**First_FAT_Err_IO2:** PEX - Received Unsupported Request
1	RWCST	0	**First_FAT_Err_IO1:** PEX - Received Fatal Error Message
0	RWCST	0	**First_FAT_Err_IO0:** PEX - Data Link Layer Protocol Error

3.8.12.25 PEX_NF_COR_FERR[7:2, 0] - PCI Express First Non-Fatal or Correctable Error Register

This register records the occurrence of the first unmasked PCI Express NON-FATAL (Uncorrectable) and CORRECTABLE errors. These errors are written by the MCH if the respective bits are not set prior. The classification of uncorrectable errors into FATAL or Non-Fatal is based on the UNCERRSEV register described in Section 3.8.12.7.

Device: 0, 2-7 Function: 0 Offset: 158h			
Bit	**Attr**	**Default**	**Description**
31:18	RV	0h	Reserved
17	RWCST	0	First_NFAT_Corr_Err_IO19: Surprise Link Down (uncorrectable)
16	RWCST	0	**First_NFAT_COR_Err_IO17:** PEX - Received Correctable Error Message
15	RWCST	0	**First_NFAT_COR_Err_IO16:** PEX - Replay Timer Timeout (correctable)
14	RWCST	0	**First_NFAT_COR_Err_IO15:** PEX - Replay_Num Rollover (correctable)
13	RWCST	0	**First_NFAT_COR_Err_IO14:** PEX - BAD DLLP Error (correctable)
12	RWCST	0	**First_NFAT_COR_Err_IO13:** PEX - Bad TLP Error (correctable)
11	RWCST	0	**First_NFAT_COR_Err_IO12:** PEX - Receiver Error (correctable)
10	RWCST	0	**First_NFAT_COR_Err_IO11:** PEX - Received Non Fatal (uncorrectable) Error Message
9	RWCST	0	**First_NFAT_COR_Err_IO10:** PEX - Receive Buffer Overflow Error (uncorrectable)
8	RWCST	0	**First_NFAT_COR_Err_IO9:** PEX -Malformed TLP (uncorrectable)
7	RWCST	0	**First_NFAT_COR_Err_IO8:** PEX - Unexpected Completion Error (uncorrectable)
6	RWCST	0	**First_NFAT_COR_Err_IO7:** PEX - Completer Abort (uncorrectable)
5	RWCST	0	**First_NFAT_COR_Err_IO6:** PEX - Completion Timeout (uncorrectable)
4	RWCST	0	**First_NFAT_COR_Err_IO5:** PEX - Flow Control Protocol Error (uncorrectable)
3	RWCST	0	**First_NFAT_COR_Err_IO4:** PEX - Poisoned TLP (uncorrectable)

Device: 0, 2-7 Function: 0 Offset: 158h			
Bit	**Attr**	**Default**	**Description**
2	RWCST	0	**First_NFAT_COR_Err_IO3:** PEX - Training Error (uncorrectable) This field should not be used for setting Training error severity due to a recent PCI-SIG ECN (Jan 22, 04) to remove training error. Hardware behavior is undefined.
1	RWCST	0	**First_NFAT_COR_Err_IO2:** PEX - Received Unsupported Request (uncorrectable)
0	RWCST	0	**First_NFAT_COR_Err_IO0:** PEX - Data Link Layer Protocol Error (uncorrectable)

3.8.12.26 PEX_FAT_NERR[7:2, 0] - PCI Express Next Fatal Error Register

This register records the subsequent occurrences after the first unmasked PCI Express FATAL errors and written by the MCH if the respective bits are set in the PEX_FERR_FAT register.

Device: 0, 2-7 Function: 0 Offset: 15Ch			
Bit	**Attr**	**Default**	**Description**
31:13	RV	0h	Reserved
12	RWCST	0	Next_FAT_Err_IO19: Surprise Link Down
11	RWCST	0	**Next_FAT_Err_IO18:** ESI Reset time-out
10	RWCST	0	**Next_FAT_Err_IO9:** PEX - Malformed TLP
9	RWCST	0	**Next_FAT_Err_IO10:** PEX - Receive Buffer Overflow Error
8	RWCST	0	**Next_FAT_Err_IO8**: PEX - Unexpected Completion Error
7	RWCST	0	**Next_FAT_Err_IO7:** PEX - Completer Abort
6	RWCST	0	**Next_FAT_Err_IO6:** PEX - Completion Timeout
5	RWCST	0	**Next_FAT_Err_IO5**: PEX - Flow Control Protocol Error
4	RWCST	0	**Next_FAT_Err_IO4:** PEX - Poisoned TLP
3	RWCST	0	**Next_FAT_Err_IO3**: PEX - Training Error This field should not be used for setting Training error severity due to a recent PCI-SIG ECN (Jan 22, 04) to remove training error. Hardware behavior is undefined.
2	RWCST	0	**Next_FAT_Err_IO2:** PEX - Received Unsupported Request
1	RWCST	0	**Next_FAT_Err_IO1**: PEX - Received Fatal Error Message
0	RWCST	0	**Next_FAT_Err_IO0:** PEX - Data Link Layer Protocol Error

3.8.12.27 PEX_NF_COR_NERR[7:2, 0] - PCI Express Non Fatal or Correctable Next Error Register

These errors are written by the MCH if the respective bits are set in PEX_NF_COR_FERR register. This register records the subsequent occurrences of unmasked PCI Express NON-FATAL (Uncorrectable) and CORRECTABLE errors.

Device: 0, 2-7
Function: 0
Offset: 160h

Bit	Attr	Default	Description
31:18	RV	0h	Reserved
17	RWST	0	Next_NFAT_Corr_Err_IO19: Surprise Link Down
16	RWCST	0	**Next_NFAT_COR_Err_IO17:** PEX - Received Correctable Error Message
15	RWCST	0	**Next_NFAT_COR_Err_IO16:** PEX - Replay Timer Timeout (correctable)
14	RWCST	0	**Next_NFAT_COR_Err_IO15:** PEX - Replay_Num Rollover (correctable)
13	RWCST	0	**Next_NFAT_COR_Err_IO14:** PEX - BAD DLLP Error (correctable)
12	RWCST	0	**Next_NFAT_COR_Err_IO13:** PEX - Bad TLP Error (correctable)
11	RWCST	0	**Next_NFAT_COR_Err_IO12:** PEX - Receiver Error (correctable)
10	RWCST	0	**Next_NFAT_COR_Err_IO11:** PEX - Received Non Fatal (uncorrectable) Error Message
9	RWCST	0	**Next_NFAT_COR_Err_IO10:** PEX - Receive Buffer Overflow Error (uncorrectable)
8	RWCST	0	**Next_NFAT_COR_Err_IO9:** PEX -Malformed TLP (uncorrectable)
7	RWCST	0	**Next_NFAT_COR_Err_IO8:** PEX - Unexpected Completion Error (uncorrectable)
6	RWCST	0	**Next_NFAT_COR_Err_IO7:** PEX - Completer Abort (uncorrectable)
5	RWCST	0	**Next_NFAT_COR_Err_IO6:** PEX - Completion Timeout (uncorrectable)
4	RWCST	0	**Next_NFAT_COR_Err_IO5:** PEX - Flow Control Protocol Error (uncorrectable)
3	RWCST	0	**Next_NFAT_COR_Err_IO4:** PEX - Poisoned TLP (uncorrectable)
2	RWCST	0	**Next_NFAT_COR_Err_IO3:** PEX - Training Error (uncorrectable) This field should not be used for setting Training error severity due to a recent PCI-SIG ECN (Jan 22, 04) to remove training error. Hardware behavior is undefined.
1	RWCST	0	**Next_NFAT_COR_Err_IO2:** PEX - Received Unsupported Request (uncorrectable)
0	RWCST	0	**Next_NFAT_COR_Err_IO0:** PEX - Data Link Layer Protocol Error (uncorrectable)

3.8.12.28 PEX_UNIT_FERR[7:2, 0] - PCI Express First Unit Error Register

This register records the occurrence of the first unit errors that are specific to this PCI Express port caused by external activities. For example, VPP error due to a malfunctioning port on the SMBUS that did not receive acknowledge due to a PCI Express Hot-Plug event. The unit errors are sent to the Coherency Engine to classify as to which port cluster it came from ports 2-3 or ports 4-7 and the errors are recorded in Coherency Engine and appropriate interrupts generated through ERR pins.

Device: 2-7 Function: 0 Offset: 168h			
Bit	**Attr**	**Default**	**Description**
31:1	RV	0h	Reserved
0	RWCST	0	**First_FAT_VPP_Err:** VPP Error for PCI Express port Records the occurrence of the first VPP error if this bit is not set prior. Software clears this when the error has been serviced.

3.8.12.29 PEX_UNIT_NERR[7:2] - PCI Express Next Unit Error Register

This register records the occurrence of subsequent unit errors that are specific to this PCI Express port caused by external activities. For example, VPP error due to a malfunctioning port on the SMBUS that did not receive acknowledge due to a PCI Express Hot-Plug event. The next unit errors are sent to the Coherency Engine where the errors are further recorded and appropriate interrupts are generated through ERR pins.

Device: 2-7 Function: 0 Offset: 16Ch			
Bit	**Attr**	**Default**	**Description**
31:1	RV	0h	Reserved
0	RWCST	0	**Next_FAT_VPP_Err:** VPP Error for PCI Express port Records the occurrence of subsequent VPP errors after the PEX_UNIT_FERR.First_FAT_VP_ERR is set. Software clears this when the error has been serviced.

3.8.12.30 PEX_SSERR[7:2,0]: PCI Express Stop and Scream Error Register

This register records the occurrence of stop and scream error due to data poisoning.

Device: 7-2, 0 Function: 0 Offset: 170h			
Bit	**Attr**	**Default**	**Description**
7:1	RV	0h	*Reserved*
0	RWCST	0	**SSErr: Stop and Scream Error for PCI Express port** Records the occurrence of the first stop and scream error on the PCI Express port if this bit is clear.

3.8.13 Error Registers

This section describes the registers that record the first and next errors, logging, detection masks, signalling masks, and error injection control. The FERR_GLOBAL (first error register) is used to record the first error condition. The NERR_GLOBAL register is used to record subsequent errors.

The contents of FERR_GLOBAL and NERR_GLOBAL are “sticky” across a reset (while PWRGOOD remains asserted). This provides the ability for firmware to perform diagnostics across reboots. Note that only the contents of FERR_GLOBAL affects the update of the any error log registers.

3.8.13.1 FERR_GLOBAL - Global First Error Register

The first fatal and/or first non-fatal errors are flagged in the FERR_GLOBAL register, subsequent errors are indicated in the NERR_GLOBAL register.

Device: 16
Function: 2
Offset: 40h

Bit	Attr	Default	Description
31	RWCST	0	**Global_FERR_31:** Internal Intel 5000X Chipset MCH Fatal Error
30	RWCST	0	**Global_FERR_30:** DMA Engine Device Fatal Error
29	RWCST	0	**Global_FERR_29:** FSB1 Fatal Error
28	RWCST	0	**Global_FERR_28:** FSB 0 Fatal Error
27	RWCST	0	**Global_FERR_27:** FB-DIMM Channel 3 Fatal Error
26	RWCST	0	**Global_FERR_26:** FB-DIMM Channel 2 Fatal Error **NOTE**: Intel 5000X Chipset MC can only decode ECC failures to a specific rank. Hence if such an error is detected, it will log errors on the respective branch in the lower channel that is, an ECC fail in branch 1 would be logged in Channel 2
25	RWCST	0	**Global_FERR_25:** FB-DIMM Channel 1 Fatal Error
24	RWCST	0	**Global_FERR_24:** FB-DIMM Channel 0 Fatal Error **NOTE**: Intel 5000X Chipset MCH can only decode ECC failures to a specific rank. Hence if such an error is detected, it will log errors on the respective branch in the lower channel that is, an ECC fail in branch 0 would be logged in Channel 0‘
23	RWCST	0	**Global_FERR_23:** PCI Express Device 7 Fatal Error
22	RWCST	0	**Global_FERR_22:** PCI Express Device 6 Fatal Error
21	RWCST	0	**Global_FERR_21:** PCI Express Device 5 Fatal Error
20	RWCST	0	**Global_FERR_20:** PCI Express Device 4 Fatal Error
19	RWCST	0	**Global_FERR_19:** PCI Express Device 3 Fatal Error
18	RWCST	0	**Global_FERR_18:** PCI Express Device 2 Fatal Error
17	RV	0	Reserved
16	RWCST	0	**Global_FERR_16:** ESI Fatal Error
15	RWCST	0	**Global_FERR_15:** Internal Intel 5000X Chipset MCH Non-Fatal Error
14	RWCST	0	**Global_FERR_14:** DMA Engine Device Non Fatal Error

Device: 16 Function: 2 Offset: 40h			
Bit	**Attr**	**Default**	**Description**
13	RWCST	0	**Global_FERR_13:** FSB1 Non-Fatal Error
12	RWCST	0	**Global_FERR_12:** FSB 0 Non-Fatal Error
11	RWCST	0	**Global_FERR_11:** FB-DIMM Channel 3 Non-Fatal Error
10	RWCST	0	**Global_FERR_10:** FB-DIMM Channel 2 Non-Fatal Error
9	RWCST	0	**Global_FERR_09:** FB-DIMM Channel 1 Non-Fatal Error
8	RWCST	0	**Global_FERR_08:** FB-DIMM Channel 0 Non-Fatal Error
7	RWCST	0	**Global_FERR_07:** PCI Express Device 7Non-Fatal Error
6	RWCST	0	**Global_FERR_06:** PCI Express Device 6 Non-Fatal Error
5	RWCST	0	**Global_FERR_05:** PCI Express Device 5 Non-Fatal Error
4	RWCST	0	**Global_FERR_04:** PCI Express Device 4 Non-Fatal Error
3	RWCST	0	**Global_FERR_03:** PCI Express Device 3 Non-Fatal Error
2	RWCST	0	**Global_FERR_02:** PCI Express Device 2 Non-Fatal Error
1	RV	0	Reserved
0	RWCST	0	**Global_FERR_00:** ESI Non-Fatal Error

3.8.13.2 NERR_GLOBAL - Global Next Error Register

Once an error has been logged in the FERR_GLOBAL, subsequent errors are logged in the NERR_GLOBAL register.

Device: 16 Function: 2 Offset: 44h			
Bit	**Attr**	**Default**	**Description**
31	RWCST	0	**Global_NERR_31:** Internal Fatal Error
30	RWCST	0	**Global_NERR_30:** DMA Engine Device Fatal Error
29	RWCST	0	**Global_NERR_29:** FSB1 Fatal Error
28	RWCST	0	**Global_NERR_28:** FSB 0 Fatal Error
27:25	RV	0	Reserved

Device: 16
Function: 2
Offset: 44h

Bit	Attr	Default	Description
24	RWCST	0	**Global_NERR_24:** FB-DIMM Channel 0,1,2 or 3 Fatal Error
23	RWCST	0	**Global_NERR_23:** PCI Express Device 7 Fatal Error
22	RWCST	0	**Global_NERR_22:** PCI Express Device 6 Fatal Error
21	RWCST	0	**Global_NERR_21:** PCI Express Device 5 Fatal Error
20	RWCST	0	**Global_NERR_20:** PCI Express Device 4 Fatal Error
19	RWCST	0	**Global_NERR_19:** PCI Express Device 3 Fatal Error
18	RWCST	0	**Global_NERR_18:** PCI Express Device 2 Fatal Error
17	RV	0	Reserved
16	RWCST	0	**Global_NERR_16:** ESI Fatal Error
15	RWCST	0	**Global_NERR_15:** Internal Intel 5000X Chipset MCH Non-Fatal Error
14	RWCST	0	**Global_NERR_14:** DMA Engine Device Non Fatal Error
13	RWCST	0	**Global_NERR_13:** FSB1 Non-Fatal Error
12	RWCST	0	**Global_NERR_12:** FSB 0 Non-Fatal Error
11:9	RV	0h	Reserved
8	RWCST	0	**Global_NERR_08:** FB-DIMM Channel 0,1, 2 or 3 Non-Fatal Error
7	RWCST	0	**Global_NERR_07:** PCI Express Device 7 Non-Fatal Error
6	RWCST	0	**Global_NERR_06:** PCI Express Device 6 Non-Fatal Error
5	RWCST	0	**Global_NERR_05:** PCI Express Device 5 Non-Fatal Error
4	RWCST	0	**Global_NERR_04:** PCI Express Device 4 Non-Fatal Error
3	RWCST	0	**Global_NERR_03:** PCI Express Device 3 Non-Fatal Error
2	RWCST	0	**Global_NERR_02:** PCI Express Device 2 Non-Fatal Error
1	RV	0	Reserved
0	RWCST	0	**Global_NERR_00:** ESI Non-Fatal Error

3.8.13.3 FERR_FAT_FSB[1:0]: FSB First Fatal Error Register

Device: 16
Function: 0
Offset: 480h, 180h

Bit	Attr	Default	Description
7:6	RV	00	Reserved
5	RWCST	0	**F9Err:** FSB protocol Error
4	RV	0h	Reserved
3	RWCST	0	**F2Err:** Unsupported Processor Bus Transaction
2:1	RV	0h	Reserved
0	RWCST	0	**F1Err:** Request/Address Parity Error

3.8.13.4 FERR_NF_FSB[1:0]: FSB First Non-Fatal Error Register

Device: 16
Function: 0
Offset: 481h, 181h

Bit	Attr	Default	Description
7:3	RV	00000	Reserved
2	RWCST	0	**F7Err:** Detected MCERR from a processor
1	RWCST	0	**F8Err:** Detected BINIT from a processor
0	RWCST	0	**F6Err:** Parity Error in Data from FSB Interface

3.8.13.5 NERR_FAT_FSB[1:0]: FSB Next Fatal Error Register

This register logs all FSB subsequent errors after the FERR_FAT_FSB has logged the 1st fatal error.

Device: 16
Function: 0
Offset: 482h, 182h

Bit	Attr	Default	Description
7:6	RV	00	Reserved
5	RWCST	0	**F9Err:** FSB protocol Error
4	RV	0h	Reserved
3	RWCST	0	**F2Err:** Unsupported Processor Bus Transaction
2:1	RV	0h	Reserved
0	RWCST	0	**F1Err:** Request/Address Parity Error

3.8.13.6 NERR_NF_FSB[1:0]: FSB Next Non-Fatal Error Register

This register logs all FSB subsequent errors after the FERR_NF_FSB has logged the 1st fatal error.

Device: 16 Function: 0 Offset: 483h, 183h			
Bit	**Attr**	**Default**	**Description**
7:3	RV	00000	Reserved
2	RWCST	0	**F7Err:** Detected MCERR from a processor
1	RWCST	0	**F8Err:** Detected BINIT from a processor
0	RWCST	0	**F6Err:** Parity Error in Data from FSB Interface

3.8.13.7 NRECFSB[1:0]: Non Recoverable FSB Error Log Register

FSB Log registers for non recoverable errors when a fatal error is logged in its corresponding FERR_FAT_FSB Register.

Device: 16 Function: 0 Offset: 484h, 184h			
Bit	**Attr**	**Default**	**Description**
31:29	RV	000	Reserved
28:24	ROST	00000	**REQA:** REQa[4:0] fields of the FSB
23:21	ROST	000	**REQB:** REQb[2:0] fields of the FSB
20:16	ROST	00000	**EXF:** EXF[4:0] fields of the FSB
15:8	ROST	00h	**ATTR:** ATTR[7:0] fields of the FSB
7:0	ROST	00h	**DID:** DID[7:0] fields of the FSB

3.8.13.8 RECFSB[1:0]: Recoverable FSB Error Log Register

The following error log registers captures the FSB fields on the logging of an error in the corresponding FERR_NF_FSB Register.

Device: 16 Function: 0 Offset: 488h, 188h			
Bit	**Attr**	**Default**	**Description**
31:29	RV	000	Reserved
28:24	ROST	00000	**REQA:** REQa[4:0] fields of the FSB
23:21	ROST	000	**REQB:** REQb[2:0] fields of the FSB
20:16	ROST	00000	**EXF:** EXF[4:0] fields of the FSB
15:8	ROST	00h	**ATTR**: ATTR[7:0] fields of the FSB
7:0	ROST	00h	**DID:** DID[7:0] fields of the FSB

3.8.13.9 NRECADDRL[1:0]: Non Recoverable FSB Address Low Error Log Register

This register captures the lower 32 bits of the FSB address for non recoverable errors when a fatal error is logged in its corresponding FERR_FAT_FSB Register. This register is only valid for Request FSB Errors.

Device: 16
Function: 0
Offset: 48Ch, 18Ch

Bit	Attr	Default	Description
31:4	ROST	0h	**A31DT4:** FSB Address [31:4]
3	ROST	0	**A3:** FSB Address [3]
2:0	RV	000	Reserved

3.8.13.10 NRECADDRH[1:0]: Non Recoverable FSB Address High Error Log Register

This register captures the upper 8 bits of the FSB address for non recoverable errors when a fatal error is logged in its corresponding FERR_FAT_FSB Register. This register is only valid for Request FSB Errors.

Device: 16
Function: 0
Offset: 490h, 190h

Bit	Attr	Default	Description
7:0	ROST	00h	**A39DT32:** FSB Address [39:32]

3.8.13.11 EMASK_FSB[1:0]: FSB Error Mask Register

A ‘0’ in any field enables that error.

Device: 16
Function: 0
Offset: 492h, 192h

Bit	Attr	Default	Description
15:9	RV	0h	Reserved
8	RWST	1	**F9Msk:** FSB Protocol Error
7	RWST	1	**F8Msk:** B-INIT
6	RWST	1	**F7Msk:** Detected MCERR
5	RWST	1	**F6Msk:** Data Parity Error
4	RV	0h	Reserved
3	RV	0h	Reserved
2	RV	0h	Reserved
1	RWST	1	**F2Msk:** Unsupported Processor Bus Transaction
0	RWST	1	**F1Msk:** Request/Address Parity Error

3.8.13.12 ERR2_FSB[1:0]: FSB Error 2 Mask Register

This register enables the signaling of Err[2] when an error flag is set. Note that one and only one error signal should be enabled ERR2_FSB, ERR1_FSB, ERR0_FSB, and MCERR_FSB for each of the corresponding bits.

Device: 16
Function: 0
Offset: 498h, 198h

Bit	Attr	Default	Description
15:9	RV	0h	Reserved
8	RWST	1	**F9Msk:** FSB Protocol Error
7	RWST	1	**F8Msk:** B-INIT
6	RWST	1	**F7Msk:** Detected MCERR
5	RWST	1	**F6Msk:** Data Parity Error
4	RV	0	Reserved
3	RV	0h	Reserved
2	RV	0h	Reserved
1	RWST	1	**F2Msk:** Unsupported Processor Bus Transaction
0	RWST	1	**F1Msk:** Request/Address Parity Error

3.8.13.13 ERR1_FSB[1:0]: FSB Error 1 Mask Register

This register enables the signaling of Err[1] when an error flag is set. Note that one and only one error signal should be enabled ERR2_FSB, ERR1_FSB, ERR0_FSB, and MCERR_FSB for each of the corresponding bits.

Device: 16
Function: 0
Offset: 496h, 196h

Bit	Attr	Default	Description
15:9	RV	0h	Reserved
8	RWST	1	**F9Msk:** FSB Protocol Error
7	RWST	1	**F8Msk:** B-INIT
6	RWST	1	**F7Msk:** Detected MCERR
5	RWST	1	**F6Msk:** Data Parity Error
4	RV	0h	Reserved
3	RV	0h	Reserved
2	RV	0h	Reserved
1	RWST	1	**F2Msk:** Unsupported Processor Bus Transaction
0	RWST	1	**F1Msk:** Request/Address Parity Error

3.8.13.14 ERR0_FSB[1:0]: FSB Error 0 Mask Register

This register enables the signaling of Err[0] when an error flag is set. Note that one and only one error signal should be enabled ERR2_FSB, ERR1_FSB, ERR0_FSB, and MCERR_FSB for each of the corresponding bits.

Device: 16 Function: 0 Offset: 494h, 194h			
Bit	**Attr**	**Default**	**Description**
15:9	RV	0h	Reserved
8	RWST	1	**F9Msk:** FSB Protocol Error
7	RWST	1	**F8Msk:** B-INIT
6	RWST	1	**F7Msk:** Detected MCERR
5	RWST	1	**F6Msk:** Data Parity Error
4	RV	0h	Reserved
3	RV	0h	Reserved
2	RV	0h	Reserved
1	RWST	1	**F2Msk:** Unsupported Processor Bus Transaction
0	RWST	1	**F1Msk:** Request/Address Parity Error

3.8.13.15 MCERR_FSB[1:0]: FSB MCERR Mask Register

This register enables the signaling of MCERR when an error flag is set. Note that one and only one error signal should be enabled ERR2_FSB, ERR1_FSB, ERR0_FSB, and MCERR_FSB for each of the corresponding bits.

Device: 16 Function: 0 Offset: 49Ah, 19Ah			
Bit	**Attr**	**Default**	**Description**
15:9	RV	0h	Reserved
8	RWST	1	**F9Msk:** FSB Protocol Error
7	RWST	1	**F8Msk:** B-INIT
6	RWST	1	**F7Msk:** Detected MCERR
5	RWST	1	**F6Msk:** Data Parity Error
4	RV	0h	Reserved
3	RV	0h	Reserved
2	RV	0h	Reserved
1	RWST	1	**F2Msk:** Unsupported Processor Bus Transaction
0	RWST	1	**F1Msk:** Request/Address Parity Error

3.8.13.16 NRECSF - Non-Recoverable Error Control Information of Snoop Filter

Device: 16 Function: 2 Offset: B0h			
Bit	**Attr**	**Default**	**Description**
63:38	RV	0h	Reserved
37	ROST	0	**Hit(1), Miss(0)**
36:16	ROST	0h	**Tag(A[39:19])**
15:4	ROST	000h	**Set(A[18:7])**

Device: 16 Function: 2 Offset: B0h			
Bit	**Attr**	**Default**	**Description**
3	ROST	0	**Interleave(A[6])**
2	ROST	0	**State**
1:0	ROST	00	**Presence Vector**

3.8.13.17 RECSF - Recoverable Error Control Information of Snoop Filter

Device: 16 Function: 2 Offset: B8h			
Bit	**Attr**	**Default**	**Description**
63:38	RV	0h	Reserved
37	ROST	0	**Hit(1), Miss(0)**
36:16	ROST	0h	**Tag(A[39:19])**
15:4	ROST	000h	**Set(A[18:7])**
3	ROST	0	**Interleave(A[6])**
2	ROST	0	**State**
1:0	ROST	00	**Presence Vector**

3.8.13.18 FERR_FAT_INT - Internal First Fatal Error Register

FERR_FAT _INT latches the first MCH internal fatal error. All subsequent errors get logged in the NERR_FAT_INT.

Device: 16 Function: 2 Offset: C0h			
Bit	**Attr**	**Default**	**Description**
7:5	RV	000	Reserved
4	RWCST	0	**B7Err:** Multiple ECC error in any of the ways during SF lookup
3	RV	0	**Reserved**
2	RWCST	0	**B3Err:** Coherency Violation Error for WEWB
1	RWCST	0	**B2Err:** Multi-Tag Hit SF
0	RWCST	0	**B1Err:** DM Parity Error

3.8.13.19 FERR_NF_INT - Internal First Non-Fatal Error Register

Device: 16 Function: 2 Offset: C1h			
Bit	**Attr**	**Default**	**Description**
7:3	RV	0h	Reserved
2	RWCST	0	**B8Err:** SF Coherency Error for BIL (SF)
1	RWCST	0	**B6Err:** Single ECC error on SF lookup (SF)

Device: 16 Function: 2 Offset: C1h			
Bit	**Attr**	**Default**	**Description**
0	RWCST	0	**B5Err:** Single Address Map Error (COH)

3.8.13.20 NERR_FAT_INT - Internal Next Fatal Error Register

Device: 16 Function: 2 Offset: C2h			
Bit	**Attr**	**Default**	**Description**
7:5	RV	000	Reserved
4	RWCST	0	**B7Err:** Multiple ECC error in any of the ways during SF lookup (SF)
3	RV	0	**Reserved**
2	RWCST	0	**B3Err:** Coherency Violation Error (COH) for EWB
1	RWCST	0	**B2Err:** Multi-Tag Hit SF (SF)
0	RWCST	0	**B1Err:** DM Parity Error (DM)

3.8.13.21 NERR_NF_INT - Internal Next Non-Fatal Error Register

Device: 16 Function: 2 Offset: C3h			
Bit	**Attr**	**Default**	**Description**
7:3	RV	0h	Reserved
2	RWCST	0	**B8Err:** SF Coherency Error for BIL (SF)
1	RWCST	0	**B6Err:** Single ECC error on SF lookup (SF)
0	RWCST	0	**B5Err:** Address Map Error (COH)

3.8.13.22 NRECINT - Non Recoverable Internal MCH Error Log Register

This register will log non-recoverable errors (Fatal and Non Fatal) based on the internal MCH errors that originate from the FERR_FAT_INT, FERR_NF_INT described starting from Section 3.8.13.18. For debugging VPP errors in this register, for example, if VPP_PEX_PORT2-3 is set, then software can scan the PCI Express configuration space for unit errors logged in the device 2, 3 for PEX_UNIT_FERR/NERR register as defined in Section 3.8.12.28 to determine the failing port. The same can be repeated for the FB-DIMM Channels when VPP_FBD is set.

Device: 16 Function: 2 Offset: C4h			
Bit	**Attr**	**Default**	**Description**
31:21	RV	0h	Reserved
20:13	ROST	0h	**DM entry**
12:11	RV	00	Reserved

Device: 16 Function: 2 Offset: C4h			
Bit	**Attr**	**Default**	**Description**
10:8	ROST	00000	**Internal Block that detected the Failure** 001: VPP_PEX_PORT2-3 010: VPP_PEX_PORT4-7 011: VPP_FBD 101: COH 101: DM Others: *Reserved*
7	RV	0	Reserved
6:0	ROST	0h	**COH Entry of Failed Location**

3.8.13.23 RECINT - Recoverable Internal MCH Data Log Register

This register is not currently used as there are no correctable errors with in the internal data path of the MCH.

Device: 16 Function: 2 Offset: C8h			
Bit	**Attr**	**Default**	**Description**
31:21	RV	0h	Reserved
20:13	ROST	0h	**DM entry**
12:11	RV	00	Reserved
10:8	ROST	000	**Internal Block that detected the Failure** 001: VPP_PEX_PORT2-3 010: VPP_PEX_PORT4-7 011: VPP_FBD 101: COH 101: DM Others: *Reserved*
7	RV	0	Reserved
6:0	ROST	00h	**COH Entry of Failed Location**

3.8.13.24 EMASK_INT - Internal Error Mask Register

A ‘0’ in any bit position enables the corresponding error.

Device: 16
Function: 2
Offset: CCh

Bit	Attr	Default	Description
7	RV	1	**B8Msk:** SF Coherency Error for BIL
6	RWST	1	**B7Msk:** Multiple ECC error in any of the ways during SF lookup
5	RWST	1	**B6Msk:** Single ECC error on SF lookup
4	RWST	1	**B5Msk:** Address Map Error
3	RWST	1	**B4Msk:** Virtual Pin Port Error
2	RWST	1	**B3Msk:** Coherency Violation Error for EWB
1	RWST	1	**B2Msk:** Multi-Tag Hit SF
0	RWST	1	**B1Msk:** DM Parity Error

3.8.13.25 ERR2_INT - Internal Error 2 Mask Register

This register enables the signaling of Err[2] when an error flag is set. Note that one and only one error signal should be enabled in the ERR2_INT, ERR1_INT, ERR0_INT, and MCERR_INT for each of the corresponding bits.

Device: 16
Function: 2
Offset: D2h

Bit	Attr	Default	Description
7	RWST	1	**B8Err2Msk:** SF Coherency Error for BIL
6	RWST	1	**B7Err2Msk:** Multiple ECC error in any of the ways during SF lookup
5	RWST	1	**B6Err2Msk:** Single ECC error on SF lookup
4	RWST	1	**B5Err2Msk:** Address Map Error
3	RWST	1	**B4Err2Msk:** SMBus Virtual Pin Error
2	RWST	1	**B3Err2Msk:** Coherency Violation Error for EWB
1	RWST	1	**B2Err2Msk:** Multi-Tag Hit SF
0	RWST	1	**B1Err2Msk:** DM Parity Error

3.8.13.26 ERR1_INT - Internal Error 1 Mask Register

This register enables the signaling of Err[1] when an error flag is set. Note that one and only one error signal should be enabled in the ERR2_INT, ERR1_INT, ERR0_INT, and MCERR_INT for each of the corresponding bits.

Device: 16
Function: 2
Offset: D1h

Bit	Attr	Default	Description
7	RWST	1	**B8Err1Msk:** SF Coherency Error for BIL
6	RWST	1	**B7Err1Msk:** Multiple ECC error in any of the ways during SF lookup
5	RWST	1	**B6Err1Msk:** Single ECC error on SF lookup
4	RWST	1	**B5Err1Msk:** Address Map Error
3	RWST	1	**B4Err1Msk:** SMBus Virtual Pin Error
2	RWST	1	**B3Err1Msk:** Coherency Violation Error
1	RWST	1	**B2Err1Msk:** Multi-Tag Hit SF
0	RWST	1	**B1Err1Msk:** DM Parity Error

3.8.13.27 ERR0_INT - Internal Error 0 Mask Register

This register enables the signaling of Err[0] when an error flag is set. Note that one and only one error signal should be enabled in the ERR2_INT, ERR1_INT, ERR0_INT, and MCERR_INT for each of the corresponding bits.

Device: 16
Function: 2
Offset: D0h

Bit	Attr	Default	Description
7	RWST	1	**B8Err0Msk:** SF Coherency Error for BIL
6	RWST	1	**B7Err0Msk:** Multiple ECC error in any of the ways during SF lookup
5	RWST	1	**B6Err0Msk:** Single ECC error on SF lookup
4	RWST	1	**B5Err0Msk:** Address Map Error
3	RWST	1	**B4Err0Msk:** SMBus Virtual Pin Error
2	RWST	1	**B3Err0Msk:** Coherency Violation Error for EWB
1	RWST	1	**B2Err0Msk:** Multi-Tag Hit SF
0	RWST	1	**B1Err0Msk:** DM Parity Error

3.8.13.28 MCERR_INT - Internal MCERR Mask Register

This register enables the signaling of MCERR when an error flag is set. Note that one and only one error signal should be enabled. Note that one and only one error signal should be enabled in the ERR2_INT, ERR1_INT, ERR0_INT, and MCERR_INT for each of the corresponding bits.

Device: 16
Function: 2
Offset: D3h

Bit	Attr	Default	Description
7	RWST	1	**B8McErrMsk:** SF Coherency Error for BIL
6	RWST	1	**B7McErrMsk:** Multiple ECC error in any of the ways during SF lookup
5	RWST	1	**B6McErrMsk:** Single ECC error on SF lookup
4	RWST	1	**B5McErrMsk:** Address Map Error
3	RWST	1	**B4McErrMsk:** SMBus Virtual Pin Error
2	RWST	1	**B3McErrMsk:** Coherency Violation Error for EWB
1	RWST	1	**B2McErrMsk:** Multi-Tag Hit SF
0	RWST	1	**B1McErrMsk:** DM Parity Error

3.8.13.29 THRMSR_OP: Thermal Sensor Output Register

Device: 16
Function: 2
Offset: E0h

Bit	Attr	Default	Description
15:8	RV	0h	Reserved
7:0	RO	0h	**THRMSR_OP_VAL: Thermal Sensor Output Value** The field contains the temperature read out from the 8-bit A/D convertor in the thermal sensor. The output of the thermal sensor is latched by the Intel 5000X Chipset MCH in this register based on the settings of the THRMSR_CTRL. register. This register has a granularity of 0.5°C. Hence the temperature range corresponds to (0,127.5°C) for the numerical range (0,255).

3.8.13.30 THRMSR_THRLD_LO: Thermal Sensor Threshold Low Register

Device: 16
Function: 2
Offset: E2h

Bit	Attr	Default	Description
7:0	RW	B4h	**THRMSR_THTRLD_LOLM: Thermal Sensor Threshold Low Limit** The field is initialized by software to set the “low” threshold mark for the Thermal sensor logic in the Intel 5000X Chipset MCH. (Tsr_lo). Resolution of this register is 0.5’C. Default value is 90°C. That is, B4h (180d) See Section 5.4.2 for details on the interrupt generation

3.8.13.31 THRMSR_THRLD_HI: Thermal Sensor Threshold High Register

Device: 16 Function: 2 Offset: E3h			
Bit	**Attr**	**Default**	**Description**
7:0	RW	C8h	**THRMSR_THTRLD_HILM: Thermal Sensor Threshold High Limit** The field is initialized by software to set the "high" threshold for the Thermal sensor logic in the Intel 5000X Chipset MCH. (Tsr_hi) Resolution of this register is 0.5°C. Default value is 100°C. that is, C8h (200d) See Section 5.4.2 for details on the interrupt generation

3.8.13.32 THRMSR_STS_LO: Thermal Sensor Status Low Register

The thermal sensor status Low register provides information on the thermal event that triggered the interrupts on low. This includes the status bit, temperature read out when the low threshold was crossed.

Device: 16 Function: 2 Offset: E4h			
Bit	**Attr**	**Default**	**Description**
15:9	RV	0h	Reserved
8	RWCST	0	**THRMSR_STSEV_LO: Thermal Sensor Status Event Low** 0: Thermal sensor event/interrupt is not generated by the Intel 5000X Chipset MCH (default) 1: Thermal sensor event/interrupts is set by the Intel 5000X Chipset MCH when the Thermal sensor low threshold trip point is crossed. Refer to Section 5.4.2 for details. This field is sticky through reset. Software clears this field by writing a 1b. When this field is set, the THRMSR_STS.THRMSR_TEMP_LO field below is updated with the value from the THRMSR_OP register.
7:0	RWCST	0h	**THRMSR_TEMP_LO: Thermal Sensor Temperature Low** The temperature output from the THRMSR_OP register is recorded in this field by the Intel 5000X Chipset MCH when the Thermal sensor low threshold trip point is crossed and THRMSR_STS.THRMSR_STSEV_LO is set. Resolution of this register field is 0.5°C. This field is sticky through reset. Software clears this field by writing a 1b. Note that the chipset will record (overwrite) new temperature values if a subsequent threshold crossing has occurred before the software could clear it.

3.8.13.33 THRMSR_STS_HI: Thermal Sensor Status High Register

The thermal sensor status register provides information on the thermal event that triggered the interrupts. This includes the status bit, temperature read out when the high threshold was crossed.

Device: 16
Function: 2
Offset: E6h

Bit	Attr	Default	Description
15:9	RV	0h	Reserved
8	RWCST	0	**THRMSR_STSEV_HI: Thermal Sensor Status Event High** 0: Thermal sensor event/interrupt is not generated by the Intel 5000X Chipset MCH (default) 1: Thermal sensor event/interrupts is set by the Intel 5000X Chipset MCH when the Thermal sensor high threshold trip point is crossed. Refer to Section 5.4.2 for details. This field is sticky through reset. Software clears this field by writing a 1b. When this field is set, the THRMSR_STS.THRMSR_TEMP_HI field below is updated with the value from the THRMSR_OP register.
7:0	RWCST	0h	**THRMSR_TEMP_HI: Thermal Sensor Temperature High** The temperature output from the THRMSR_OP register is recorded in this field by the Intel 5000X Chipset MCH when the thermal sensor high threshold trip point is crossed and THRMSR_STS.THRMSR_STSEV_HI is set. Resolution of this register is 0.5°C. This field is sticky through reset. Software clears this field by writing a 1b. Note that the chipset will record (overwrite) new temperature values if a subsequent threshold crossing has occurred before the software could clear it.

3.8.13.34 THRMSR_MSK: Thermal Sensor Mask Register

This register provides control to block thermal interrupts high/low to ERR[2:0]# pins.

Device: 16
Function: 2
Offset: E8h

Bit	Attr	Default	Description
7:2	RV	0h	*Reserved*.
1	RW	0	**THRMSR_INTLO_MSK:** Thermal Sensor Interrupt Low Event Mask 0: Enables Thermal sensor interrupts on Low to be signaled. 1: Disables (blocks) thermal interrupts on Low from being signaled. This bit does not inhibit the functioning of the THRMSR_STS register.
0	RW	0	**THRMSR_INTHI_MSK:** Thermal Sensor Interrupt High Event Mask 0: Enables Thermal sensor interrupts on high to be signaled. 1: Disables (blocks) thermal interrupts on high from being signaled. This bit does not inhibit the functioning of the THRMSR_STS register.

3.8.13.35 THRMSR_DOCMD: Thermal Sensor Do Command Register

This register provides the routing information to ERR[2:0] pins for thermal sensor event signaling.

Device: 16
Function: 2
Offset: E9h

Bit	Attr	Default	Description
7:2	RV	0h	*Reserved*.
1:0	RW	00	**THRMSR_EVS:** Thermal Sensor Interrupt Event steering for interrupts 00: ERR[0] 01: ERR[1] 10: ERR[2] 11: Reserved Software sets this field to route the thermal sensor event to one of the ERR[2:0] pins. Note that both interrupts high and low are routed through the same pins.

3.8.13.36 THRMSR_CTRL: Thermal Sensor Control Register

Device: 16
Function: 2
Offset: EAh

Bit	Attr	Default	Description
1	RWST	0	**THRMSR_EN: Thermal Sensor Enable** This register field controls the functionality of the thermal sensor and the latching capability of the interrupt generation logic. 0: Disables the thermal sensor A/D convertor logic from calculating the on-die temperature. In addition, it will disable temperature latching in the THRMSROP register. 1: Enables the thermal sensor A/D convertor circuitry to perform its operations for reading the temperature and also allows the THRMSROP register to latch the temperature readout from the sensor every clock. Default value is 0b and the field is sticky through hard reset. **Usage**: When BIOS/SW enables the THRMSR_EN field by writing a '1', the thermal sensor logic will be enabled for 4096 clocks during the first 125ms interval, and then subsequently, it will use a 64 clock window to enable the thermal sensor every 125ms. This will provide 8 samples in a 1 second window and deliver the required accuracy both for temperature gradient change and measured value.
0	RW	0	**THRMSR_EVLOG_EN: Thermal Sensor Event logic Enable** 0: Thermal sensor status event/interrupt generation logic is disabled 1: Thermal sensor status event/interrupts are signaled when the required threshold is crossed. See Section 5.4.2 for details on the interrupt generation. Default value is set to 0b to disable the thermal event generation logic. For this bit to function correctly, THRMSR_CTRL.THRMSR_LATCH_EN must also be set.

3.8.13.37 SFCMD - SF Access Command Register

This register is the command register for SF configuration access.

Device: 17
Function: 0
Offset: 80

Bit	Attr	Default	Description
63:0	WO	0h	**SFCmdEntry:** SF Command Register Quad-Word.

3.8.13.38 SFRESP - SF Access Response Register

This register is the command register for SF configuration access.

Device: 17
Function: 0
Offset: 88h

Bit	Attr	Default	Description
63:0	RO	0h	**SFResponse:** SF Response Register Quad-Word.

3.9 Memory Control Registers

3.9.1 MC - Memory Control Settings

Miscellaneous controls not implemented in other registers.

Device: 16
Function: 1
Offset: 40h

Bit	Attr	Default	Description
31	RV	0	Reserved
30	RW	0	**RETRY: Retry Enable** '1' = enables retry. '0' = disables retry.
29	RV	0	Reserved
28:25	RW	0h	**BADRAMTH: BADRAM Threshold** Number of consecutive instances of adjacent symbol errors required to mark a bad device in a rank. Number of patrol scrub cycles required to decrement a non-saturated BADCNT. If Software desires to enable the "enhanced mode" and use the BADRAMTH, it needs to set a non-zero value to this register field prior. Otherwise, a value of 0 is considered illegal and memory RAS operations may lead to indeterministic behavior.
24:22	RV	0	Reserved
21	RW	0	**INITDONE: Initialization Complete.** This scratch bit communicates software state from Intel 5000X Chipset MCH to BIOS. BIOS sets this bit to 1 after initialization of the DRAM memory array is complete. This bit has no effect on Intel 5000X Chipset MCH operation.
20	RW	0	**FSMEN: FSM enable.** '1' = Enables operation of DDR protocol. This can be used as a synchronous reset to the FSM. (normal) '0' = Inhibits processing of enqueued transactions. Disables all DRAM accesses which means that the FB-DIMM link comes up, trains, goes to L0, sends NOPs, does alerts, syncs, fast resets, AMB configurations, and so forth, but does not perform: a) Memory reads b) Memory writes c) Refreshes Not preserved by SAVCFG bit in the SYRE register.

Device: 16
Function: 1
Offset: 40h

Bit	Attr	Default	Description
19:18	RW	0h	**ETHROT: DIMM Electrical Throttling Limit** Electrical throttling is required to prevent data corruption by limiting the number of activates within a specific time interval and is enabled by MTR.ETHROTTLE register bit. For each rank in the DIMM, Maximum number of activates is four per sliding electrical throttle window as defined below: 00: 10 clocks(DDR533) 01: 13 clocks(DDR667) 10: 15 clocks(DDR800) 11: 20 clocks(safe/conservative setting) The Memory controller should stop sending more than 4 activates for each sliding electrical throttle window. When the sliding window boundary crosses, the counter is reset and the process repeats.
17	RW	0	**GTW_MODE: Global Throttling window mode** This register field is used to reduce the Global throttling window size for the purposes of debug/validation. 0: Global/open-loop throttling window of 16384*1344 (default, normal working mode). If global throttling is enabled in this normal window, it will be held active for 16 global throttling windows without any DIMM exceeding GBLACT. 1: Global/open-loop throttling window of 4*1344 (debug, validation). If global throttling is enabled in this debug window, it will be held active for 2 global throttling windows without any DIMM exceeding GBLACT.
16	RW	0	**MIRROR: Mirror mode enable** '1' = mirroring enabled '0' = mirroring disabled. FBDHPC.NEXTSTATE defines other characteristics of mirrored mode. The Intel 5000X Chipset MCH does not support mirroring while sparing is enabled: this bit should not be set if SPCPC.SPAREN is set. The Intel 5000X Chipset MCH does not support mirroring with demand scrub: this bit should not be set if DEMSEN is set. Note: When MIRROR mode is enabled, both WAY0 and WAY1 of MIR register should be set to 1. Otherwise, it is a programming error.
15:9	RV	0h	Reserved
8	RW	0	**SCRBALGO: Scrub Algorithm for x8 uncorrectable error detection** 0: Normal mode 1: Enhanced mode
7	RW	0	**SCRBEN: Patrol Scrub Enable** 1: Enables patrol scrubbing. 0: Disables patrol scrubbing The scrub engine will start the scrub operations from the beginning to the end of the memory each time the SCRBEN register bit is set. Note that SCRBEN should be disabled during MIR updates.
6	RW	0	**DEMSEN: Demand Scrub Enable** Enables demand scrubbing. This bit must not be set when MIRROR is set.
5	RW	0	**ERRDETEN: Error Detection Enable** '1' = Northbound CRC/ECC checking enabled. '0' = Northbound CRC/ECC checking disabled FB-DIMM "Alert" detection is disabled, status packets are ignored, northbound error logging and data poisoning are disabled when Northbound CRC/ECC checking is disabled.
4	RWC	0	**SCRBDONE: Scrub Complete** The scrub unit will set this bit to '1' when it has completed scrubbing the entire memory. Software should poll this bit after setting the Scrub Enable (SCRBEN) bit to determine when the operation has completed. If the Scrub enable bit is cleared midway during the scrub cycle, then the SCRBDONE bit will not be set and the Intel 5000X Chipset MCH will stop the scrub cycle immediately.
3:0	RV	0h	Reserved

3.9.2 GBLACT - Global Activation Throttle Register

This register contains the hostel limit for Global Activation throttle control.

Device: 16 Function: 1 Offset: 60h			
Bit	**Attr**	**Default**	**Description**
7:0	RW	0	**GBLACTLM: Global Activation Throttle Limit** This field controls the activation of Global throttling based on the number of activates sampled per DIMM pair on each branch. If the number of activates in the global throttling window[a] exceeds the number indicated by the GBLACTLM filed in this register, then global throttling is started by setting the THRSTS[1:0].GBLTHRT bit for the respective branch and the Global activation throttling logic to use the THRTMID register for throttling. The granularity of this field is 65536 activations. Refer to Table 3-37. If Software sets this value greater than 168, the chipset will cap the GBLACTLM field to 168.

Notes:

a. If (MC.GTW_MODE==1), then the global throttling window is 4*1344 cycles (debug, validation). Else if (MC.GTW_MODE==0), then the window is set to 16384*1344 cycles (normal).

Table 3-37. Global Activation Throttling as a Function of Global Activation Throttling Limit (GBLACTM) and Global Throttling Window Mode (GTW_MODE) Register Fields

GBLACT.GBLACTM Range (0.168)	Number of Activations	
	MC.GTW_MODE=0 (16384*1344 window)	MC.GTW_MODE=1 (4*1344 window)
0	No Throttling (unlimited activations)	No Throttling (unlimited activations)
1	65536	16
2	131072	32
16	1048576	256
32	2097152	512
64	4194304	1024
96	6291456	1536
100	6553600	1600
128	8388608	2048
150	9830400	2400
168	11010048 (100% BW)	2688 (100% BW)

3.9.3 THRTSTS[1:0] - Thermal Throttling Status Register

Device: 16 Function: 1 Offset: 6Ah, 68h			
Bit	**Attr**	**Default**	**Description**
15:9	RV	0h	Reserved

Device: 16 Function: 1 Offset: 6Ah, 68h			
Bit	**Attr**	**Default**	**Description**
8	RO	0h	**GBLTHRT: Global Activation Throttle**[a] This field is set by the Intel 5000X Chipset MCH to indicate the start of the Global Activation throttling based on the number of activates sampled in the global window. If the number of activates in the global window (16384*1344 cycles) exceeds the number indicated by the GBLACTLM field in this register, then THRTSTS.GBLTHRT bit is set to enable the Global activation throttling logic. Global activation throttling logic will remain active until 16 (or 2) global throttling[b] windows in a row have gone by without any DIMM exceeding the GBLACT.GBLACTLM register at which point this register field will be reset.
7:0	RO	0h	**THRMTHRT: Thermal Throttle Value** This field holds the current activation throttling value based on the Intel 5000X Chipset MCH/FB-DIMM throttling algorithm. 0: No throttling (unlimited activation) 1: 4 activations per activation window 2: 8 activations per activation window 168: 672 activations per activation window This field will be set by the Intel 5000X Chipset MCH and the value of this field will vary between THRTLOW and THRTHI registers based on the throttling.

Notes:

a. The Intel 5000X Chipset MCH will use an internal signal called GBLTHRT* from its combinatorial cluster for controlling open loop throttling.

b. If MC.GTW_MODE=1, then the debug mode is enabled and the Intel 5000X Chipset MCH will use 2 windows for global activation logic to be valid.

3.9.4 THRTLOW - Thermal Throttling Low Register

Device: 16 Function: 1 Offset: 64h			
Bit	**Attr**	**Default**	**Description**
7:0	RW	0h	**THRTLOWLM: Thermal Throttle Low Limit** A base throttling level that is applied when the temperature is in the low range (below Tlow) and THRTCTRL.THRMHUNT is set and the THRTSTS.GBLTHRT* bit is not set by the Global Throttling Window logic. ***Note:*** The GBLTHRT* is an internal signal from the Intel 5000X Chipset MCH open loop combinatorial cluster before it is latched in the THRSTS.GLTHRT register. This will prevent any stale/delayed information from being used for the open loop throttling logic. This base throttling is also enabled If THRTCTRL.THRMHUNT = 0 and THRTSTS.GBLTHRT* bit =0. The maximum value this field can be initialized by software is 168 (decimal). This corresponds to 672 activations per activation (throttling) window and gives 100% BW. The granularity of this field is 4 activations. 0: No throttling (unlimited activation) 1: 4 activations per activation window 2: 8 activations per activation window 168: 672 activations per activation window If Software sets this value greater than 168, the chipset will cap the THRTLOWLM field to 168.

3.9.5 THRTMID - Thermal Throttle Mid Register

Device: 16
Function: 1
Offset: 65h

Bit	Attr	Default	Description
7:0	RW	0h	**THRTMIDLM: Thermal Throttle Middle Limit** A mid level throttling level that is applied when the temperature is in the middle range (above Tlow but below Tmid) and THRTCTRL.THRMHUNT is set or the THRTSTS.GBLTHRT* bit is set by the Global Throttling Window logic in the Intel 5000X Chipset MCH. The maximum value this field can be initialized by software is 168 (decimal). This corresponds to 672 activations per activation window and gives 100% BW. The granularity of this field is 4 activations. 0: No throttling (unlimited activation) 1: 4 activations per activation window 2: 8 activations per activation window 168: 672 activations per activation window If Software sets this value greater than 168, the chipset will cap the THRTMIDLM field to 168. This field should be less than or equal to the THRTLOW.THRTLOWLM.

3.9.6 THRTHI - Thermal Throttle High Register

Device: 16
Function: 1
Offset: 66h

Bit	Attr	Default	Description
7:0	RW	0h	**THRTHILM: Thermal Throttle High Limit** The highest level of throttling (that is, minimum number of activations). When THRTCTRL.THRMODE=1, this level is applied whenever the temperature is above Tmid. When THRTCTRL.THRMODE=0, this level is the ceiling of the hunting algorithm of the closed loop throttling. The temperature being above Tmid has priority over the Global Throttling Window enabling throttling (the higher throttling level takes precedence). This throttling will be enabled if THRTCTRL.THRMHUNT is set. The maximum value this field can be initialized by software is 168 (decimal). This corresponds to 672 activations per activation window and gives 100% BW. The granularity of this field is 4 activations. 0: No throttling (unlimited activation) 1: 4 activations per activation window 2: 8 activations per activation window 168: 672 activations per activation window If Software sets this value greater than 168, the chipset will cap the THRTHILM field to 168. This field should be less than or equal to the THRTMID.THRTMIDLM.

3.9.7 THRTCTRL - Thermal Throttling Control Register

Device: 16
Function: 1
Offset: 67h

Bit	Attr	Default	Description
7:2	RV	0h	Reserved
1	RW	0	**THRMODE: Thermal Throttle Mode** 0: THRTSTS.THRMTHRT register is initialized by the Intel 5000X Chipset MCH such that they vary in range between THRTMID and THRTHI above Tmid (staircase) 1: THRTSTS.THRMTHRT register field is “slammed” to THRTHI above Tmid.

Device: 16 Function: 1 Offset: 67h			
Bit	**Attr**	**Default**	**Description**
0	RW	0	**THRMHUNT: Intelligent Thermal Throttle Enable** 0: THRTSTS.THRMTHRT register is not enabled 1: THRTSTS.THRMTHRT register is enabled for the temperature to have any influence on the throttle parameters. If THRMHUNT=0 only the GBLTHRT bit from the Global Throttle Window can change the THRMTHRT register field.

3.9.8 MCA - Memory Control Settings A

Additional miscellaneous control not reflected in other registers.

Device: 16 Function: 1 Offset: 58h			
Bit	**Attr**	**Default**	**Description**
31:28	RW	0h	**TO: Starvation Timeout** A value of zero represents eight cycles. Each increment adds eight cycles. Maximum is 128 cycles.
27:15	RV	0h	Reserved
14	RW	0	**SCHDIMM:** Single Channel DIMM Operation 0: The MC assumes that the Intel 5000X Chipset MCH is operating normally, that is, MC is not operating with only one FB-DIMM channel as in single channel mode. 1: In this mode, the Intel 5000X Chipset MCH MC will operate such that only 1 channel (that is, branch 0, channel 0) is active and there can be one or more DIMMS present in Channel 0.
13:0	RV	0h	*Reserved*.

3.9.9 DDRFRQ - DDR Frequency Ratio

This register specifies the CORE:DDR frequency ratio.

Device: 16 Function: 1 Offset: 56h			
Bit	**Attr**	**Default**	**Description**
7:6	RV	0000h	Reserved1
5:4	RO	00h	**NOW: Present CORE:DDR Frequency Ratio** '00' = **1:1**. for example, if BUSCLK=333 MHz, then DDR=667 MHz. '01' = Reserved '10' = **4:5**. for example, if BUSCLK=266 MHz, then DDR=667 MHz. '11' = **5:4**. for example, if BUSCLK=333 MHz, then DDR=533 MHz. This field will only specify the relationship between the CORE-domain and FB-DIMM-domain clocks. This field does not indicate the frequency of the FB-DIMM SCiD link... that is entirely determined by the frequency of the FBDCLK reference clocks. To achieve successful FB-DIMM channel initialization, the frequency of the FBDCLK reference clock must match the frequency of the FB-DIMM-domain clock. For example, if the BUSCLK=333 MHz and the NOW field specifies a ratio of 1:1, then FB-DIMM channel initialization will succeed with an FBDCLK frequency of 333 MHz.
3:2	RV	00h	Reserved

Device: 16 Function: 1 Offset: 56h			
Bit	**Attr**	**Default**	**Description**
1:0	RWST	00	**NEXT: Future CORE:DDR Frequency Ratio** This frequency ratio will take effect and transfer to the "NOW" field after the next Intel 5000X Chipset MCH hard reset. '00' = **1:1**. for example, if BUSCLK=333 MHz, then DDR=667 MHz. '01' = Reserved '10' = **4:5**. for example, if BUSCLK=266 MHz, then DDR=667 MHz. '11' = **5:4**. for example, if BUSCLK=333 MHz, then DDR=533 MHz. This field will only set the relationship between the CORE-domain and FB-DIMM-domain clocks. This field will not set the frequency of the FB-DIMM SCiD link... that is entirely determined by the frequency of the FBDCLK reference clocks. To achieve successful FB-DIMM channel initialization, the frequency of the FBDCLK reference clock must match the frequency of the FB-DIMM-domain clock. For example, if the BUSCLK=333 MHz and the NEXT field specifies a ratio of 1:1, then after the next Intel 5000X Chipset MCH hard reset, FB-DIMM channel initialization will succeed with an FBDCLK frequency of 333 MHz.

3.9.10 FBDTOHOSTGRCFG0: FB-DIMM to Host Gear Ratio Configuration 0

This register consists of 8 nibbles of mux select data for the proper selection of gearing behavior on the FB-DIMM. This is the first of two registers to control the behavior for the FB-DIMM to host (north bound) data flow.

Device: 16 Function: 1 Offset: 160h			
Bit	**Attr**	**Default**	**Description**
31:0	RWST	11111111h	FBDHSTGRMUX: FB-DIMM to Host Clock Gearing mux selector. Eight nibbles of mux select for memory/DDR2 to FSB/core geared clock boundary crossing phase enables. Refer to Table 3-38 for the programming details.

Table 3-38. FB-DIMM to Host Gear Ratio Mux

FSB:Memory Frequency	Gear Ratio[a]	Value
333:333 267:267 400:400	1:1	11111111h
333:267	5:4	00023230h
267:333	4:5 (conservative)	00004323h
267:333	4:5 (aggressive)	00002323h

Notes:

a. For 4:5 gear ratio, software should use either conservative or aggressive mode for all the respective memory gearing registers (no mix and match).

3.9.11 FBDTOHOSTGRCFG1: FB-DIMM to Host Gear Ratio Configuration 1

This register consists of eight nibbles of mux select data for the proper selection of gearing behavior on the FB-DIMM for the 1:1 and 4:5 modes.This is the second register for FB-DIMM to Host gearing control.

Device: 16 Function: 1 Offset: 164h			
Bit	**Attr**	**Default**	**Description**
31:0	RWST	00000000h	FBDHSTGRMUX: FB-DIMM to Host Clock Gearing mux selector. Eight nibbles of mux select for memory/DDR2 to FSB/core geared clock boundary crossing phase enables. Refer to Table 3-39 for the programming details.

Table 3-39. FB-DIMM to Host Gear Ratio Mux

FSB:Memory Frequency	Gear Ratio[a]	Value
333:333 267:267 400:400	1:1	00000000h
333:267	5:4	00000000h[b]
267:333	4:5 (conservative)	00002000h
267:333	4:5 (aggressive)	00000400h

Notes:

a. For 4:5 gear ratio, software should use either conservative or aggressive mode for all the respective memory gearing registers (no mix and match).
b. Ignored by MGr registers in the 5:4 mode.

3.9.12 HOSTTOFBDGRCFG: Host to FB-DIMM Gear Ratio Configuration

This register consists of eight nibbles of mux select data for the proper selection of gearing behavior on the Host to FB-DIMM path (south bound).

Device: 16 Function: 1 Offset: 168h			
Bit	**Attr**	**Default**	**Description**
31:0	RWST	11111111h	HSTFBDGRMUX: Host to FB-DIMM Clock Gearing mux selector. Eight nibbles of mux select for FSB/core to memory/DDR2 geared clock boundary crossing phase enables. Refer to Table 3-40 for the programming details.

Table 3-40. Host to FB-DIMM Gear Ratio Mux Select

FSB:Memory Frequency	Gear Ratio[a]	Value
333:333 267:267 400:400	1:1	11111111h
333:267	5:4	00004323h
267:333	4:5 (conservative)	00023230h
267:333	4:5 (aggressive)	00023023h

Notes:

a. For 4:5 gear ratio, software should use either conservative or aggressive mode for all the respective memory gearing registers (no mix and match).

3.9.13 GRFBDVLDCFG: FB-DIMM Valid Configuration

This register provides valid signals to assert data in the FB-DIMM side for various gearing ratios. It primarily affects the southbound data path for 4:5 gearing and determines when a NOP packet is to be inserted into the FB-DIMM.

Device: 16
Function: 1
Offset: 16Ch

Bit	Attr	Default	Description
7:0	RWST	0h	FBDVLDMUX: FB-DIMM Data Valid Mux selector. Determines which valid host cycle to insert NOP. Refer toTable 3-41 for the programming details. This primarily affects the 4:5 gearing ratio.

Table 3-41. FB-DIMM Host Data Cycle Valid Mux Select

FSB:Memory Frequency	Gear Ratio[a]	Value
333:333 267:267 400:400	1:1	00h
333:267	5:4	00h[b]
267:333	4:5 (conservative)	01h
267:333	4:5 (aggressive)	04h

Notes:

a. For 4:5 gear ratio, software should use either conservative or aggressive mode for all the respective memory gearing registers (no mix and match).
b. Ignored by Mgr registers in the 5:4 mode.

3.9.14 GRHOSTFULLCFG: Host Full Flow Control Configuration

This register configures flow control when the host is full. It primarily effects the Southbound data path and determines when the flow control signal to the core is asserted.

Device: 16
Function: 1
Offset: 16Dh

Bit	Attr	Default	Description
7:0	RWST	0h	FCMUX: Flow Control Mux Selector Configures Flow control on the host according to Table 3-42. This primarily affect the 5:4 gearing ratio.

Table 3-42. FB-DIMM to Host Flow Control Mux Select

FSB:Memory Frequency	Gear Ratio[a]	Value
333:333 267:267 400:400	1:1	00h
333:267	5:4	02h
267:333	4:5 (conservative)	02h
267:333	4:5 (aggressive)	08h

Notes:

a. For 4:5 gear ratio, software should use either conservative or aggressive mode for all the respective memory gearing registers (no mix and match).

3.9.15 GRBUBBLECFG: FB-DIMM Host Bubble Configuration

This register provides valid signals to assert data in the FB-DIMM side for various gearing ratios. This primarily affects the Northbound data path for the 5:4 configuration and determines when a bubble is inserted when gearing up.

Device: 16
Function: 1
Offset: 16Eh

Bit	Attr	Default	Description
7:0	RWST	0h	FBDBBLMUX: FB-DIMM Data Bubble Mux selector. Configures bubbles in the host according to Table 3-43. This primarily affect the 5:4 gearing ratio.

Table 3-43. FB-DIMM Bubble Mux Select

FSB:Memory Frequency	Gear Ratio	Value
333:333 267:267 400:400	1:1	00h
333:267	5:4	04h
267:333	4:5	00h[a]

Notes:

a. Ignored by Mgr registers in 4:5 mode.

3.9.16 GRFBDTOHOSTDBLCFG: FB-DIMM To Host Double Configuration

This register provides valid signals to assert data in the FB-DIMM side for various gearing ratios. This primarily affects the Northbound data path of the 4:5 config and determines when both the lanes in core contain valid FB-DIMM data.

Device: 16
Function: 1
Offset: 16Fh

Bit	Attr	Default	Description
7:0	RWST	0h	FBDHSTDBLMUX: FB-DIMM to Host Double Mux Selector Configures when both data lines are valid according to Table 3-44. This primarily affect the 4:5 gearing ratio.

Table 3-44. FB-DIMM to Host Double Config Mux Select

FSB:Memory Frequency	Gear Ratio[a]	Value
333:333 267:267 400:400	1:1	00h
333:267	5:4	00h[b]
267:333	4:5 (conservative)	08h
267:333	4:5 (aggressive)	04h

Notes:

a. For 4:5 gear ratio, software should use either conservative or aggressive mode for all the respective memory gearing registers (no mix and match).
b. Ignored by MGr registers in the 5:4 mode.

3.9.17 Summary of Memory Gearing Register operating modes

- FBDTOHOSTGRCFG1, GRFBDVLDCFG, and GRFBDTOHOSTDBLCFG are used only in 4:5 mode.
- GRBUBBLECFG is only used in 5:4 mode.
- GRHOSTFULLCFG is used in both 4:5 and 5:4 modes.
- FBDTOHOSTGRCFG0 and HOSTTOFBDGRCFG are used in 4:5, 5:4, AND 1:1 modes.

3.9.18 DRTA - DRAM Timing Register A

This register defines timing parameters for all DDR2 SDRAMs in the memory subsystem. The parameters for these devices are obtained by serial presence detect. This register must be set to provide timings that satisfy the specifications of all DRAMs detected. For example, if DRAMs present have different TRCs, the maximum should be used to program this register. Consult the JEDEC DDR2 DRAM specifications for the technology of the devices in use.

3.9.19 DRTB - DDR Timing Register B

Device: 16
Function: 1
Offset: 48h

Bit	Attr	Default	Description
31	RV	0	Reserved
30:28	RW	0h	**TAL: Additive Latency for Posted CAS** This parameter is the posted-CAS "t_{AL}" DDR2 timing parameter. It must match the value written to the EMRS register in the DRAM.
27:22	RW	0h	**TWRC: Activate command to activate command delay following a DDR write** This parameter is the minimum delay from an activate command followed by a write with page-close to another activate command on the same bank. This parameter prevents bank activation protocol violations in the DRAM's. This parameter is defined as follows: tRCD + (tCL – 1) + BL/2 + tWR + tRP where t_{RCD} is the DDR ras-to-cas delay, t_{CL} is the cas-to-first-read-data latency, BL is the burst length, t_{WR} is the write recovery time, and tRP is the precharge time.This parameter is defined in core cycles. This parameter is set to greater than or equal to the largest TWRC of any DIMM on the branch.

Device: 16 Function: 1 Offset: 48h			
Bit	**Attr**	**Default**	**Description**
21:16	RW	0h	**TRC: Activate command to activate command delay (same bank)** This parameter is the minimum delay from an activate command to another activate or refresh command to the same bank. This parameter ensures that the page of the bank that was opened by the first activate command is closed before the next activate command is issued. This parameter is defined in core cycles. This parameter is set to greater than or equal to the largest TRC of any DIMM on the branch.
15:8	RW	00h	**TRFC: Refresh command to activate command delay** This parameter is the minimum delay from a refresh command to another activate or refresh command. This parameter ensures that the banks that were opened by the refresh command are closed before the next activate command is issued. This parameter is defined in core cycles. This parameter is set to greater than or equal to the largest TRFC of any DIMM on the branch.
7:4	RW	0h	**TRRD: Activate command to activate command delay (different banks)** This parameter is the minimum delay from an activate command to another activate or refresh command to a different bank on the same rank. This parameter ensures that the electrical disturbance to the SDRAM die caused by the first activate has attenuated sufficiently before the next activate is applied. This parameter is defined in core cycles. This parameter is set to greater than or equal to the largest TRRD of any DIMM on the branch.
3:0	RW	0h	**TREF: Refresh command to Refresh command delay** This parameter is the maximum delay from a refresh command to another refresh command to the same rank. This parameter ensures that a sufficient number of refreshes per time interval are issued to each rank. This parameter is defined as an integral multiple of FBD super frames. An FBD super frame is 42 FBD packets (1:1, 5:4) or 40 FBD packets for 4:5 gear ratios in the Intel 5000X Chipset. This parameter is set to less than or equal to the smallest TREF of any DIMM on the memory sub-system. The refresh interval is typically 7.80 us for a DDRII DIMM rank. Refer to Table 3-45, “Optimal TREF Values for Various DIMM Configurations and Gear Ratios”. The refresh period is calculated as follows: DIMM refresh period = TREF * Super_Frame_size * 8 * FBD clock period where the number “8” is a constant denoting the maximum number of ranks supported by the Intel 5000X Chipset. As an example, the refresh period is given as 7 * 42 * 8 * 3ns = 7056ns for a DDRII667 system with an FSB to FBD ratio of 1:1. A value of zero disables refresh and clears the refresh counter, allowing a test program to align refresh events with the test and thus improve failure repeatability.

Table 3-45. Optimal TREF Values for Various DIMM Configurations and Gear Ratios

DIMM	Optimum TREF values as a function of core: FBD gear ratios (in FBD Super frames)		
	1:1	5:4	4:5
DDRII533	6	6	N/A
DDRII667	7	N/A	8

3.9.20 DRTB - DDR Timing Register B

This register defines timing parameters that work with all DDR ports in the memory subsystem. This register must be set to provide timings that satisfy the specifications of all detected DDR ports. For example, if DDR ports have different TR2Ws, the maximum should be used to program this register.

Device: 16
Function: 1
Offset: 4Ch

Bit	Attr	Default	Description
31:19	RV	000h	Reserved
18:16	RW	0h	**TW2RDR: Write command to read command delay, different rank** This parameter is the minimum delay from a write command to a read command on different ranks of the same DIMM. This parameter prevents data strobe protocol violations on the DIMMs DDR data bus. This parameter is defined in core cycles. This parameter is dependent on cache line size. The formula for this value is BL/2 + t_{FRR} - 1. t_{FRR} is the turnaround time required to read from different ranks on the DIMM.
15:12	RW	0h	**TR2W: Read command to write command delay** This parameter is the minimum delay from a read command to a write command on the same DIMM. This parameter prevents data strobe protocol violations on the DIMMs DDR data bus. This parameter is defined in core cycles. This parameter is dependent on cache line size. The formula for this value is BL/2 + t_{FRR} + 1. t_{FRR} is the turnaround time from read to write on the DIMM. This value applies to a DIMM-hit in the conflict checking unit.
11:8	RW	0h	**TW2R: Write command to read command delay, same rank** This parameter is the minimum delay from a write command to a read command on the same rank. This parameter prevents data strobe protocol violations on the DIMMs DDR data bus. This parameter is defined in core cycles. This parameter is dependent on cache line size. The formula for this value is t_{CL} - 1 + BL/2 + t_{WTR}.
7:4	RW	0h	**TR2R: Read command to read command delay** This parameter is the minimum delay from a read command to another read command on a different rank of the same DIMM. This parameter prevents data strobe protocol violations on the DIMMs DDR data bus. This parameter is defined in core cycles. The formula for this value is BL/2 + t_{FRR}. t_{FRR} is the turnaround time required to read from different ranks on the DIMM.
3:0	RW	0h	**TW2W: Write command to write command delay** This parameter is the minimum delay from a write command to another write command on the same DIMM. This parameter prevents data strobe protocol violations on the DIMMs DDR data bus. This parameter is defined in core cycles. This parameter is dependent on cache line size.The formula for this value is BL/2.

3.9.21 ERRPER - Error Period

Non-zero UERRCNT and CERRCNT counts are decremented when the error period counter reaches this threshold. The error period counter is cleared on reset or when it reaches this threshold. The error period counter increments every 32,768 cycles. Table 3-46 indicates the timing characteristics of this register:

Table 3-46. Timing Characteristics of ERRPER

Core Frequency	Per Increment	Maximum Period
333 MHz	98.304us	4 days, 21 hours, 16 minutes, 52.465056596 seconds
266 MHz	122.880us	6 days, 2 hours, 36 minutes, 5.581331712 seconds

Device: 16
Function: 1
Offset: 50h

Bit	Attr	Default	Description
31:0	RW	0h	**THRESH:** UERRCNT / CERRCNT decrement threshold.

3.9.22 Memory Map Registers

3.9.22.1 TOLM - Top Of Low Memory

This register defines the low MMIO gap below 4 GB. See Section 3.9.22.2.

Whereas the MIR.LIMITs are adjustable, TOLM establishes the maximum address below 4 GB that should be treated as a memory access. TOLM is defined in a 256 MB boundary.

This register must not be modified while servicing memory requests.

Device: 16
Function: 1
Offset: 6Ch

Bit	Attr	Default	Description
15:12	RW	1h	**TOLM: Top Of Low Memory** This register defines the maximum DRAM memory address that lies below 4 GB. Addresses **equal to** or greater than the TOLM, and less than 4 G, are decoded as low MMIO, MMCFG (if map within this range by HECBASE), chipset, interrupt/SMM and firmware as described in the address mapping chapter. All accesses less than the TOLM are treated as DRAM accesses (except for the VGA region when enabled and PAM gaps). Configuration software should set this field either to maximize the amount of memory in the system (same as the top MIR.LIMIT), or to minimize the allocated space for the lower PCI memory (low MMIO) plus 32 MB (chipset/ interrupt/SMM and firmware) at a 256 MB boundary. This field must be set to at least 1h, for a minimum of 256 MB of DRAM. There is also a minimum of 256 MB between TOLM and 4 GB (for low MMIO, MMCFG, chipset, interrupt/SMM and firmware) since TOLM is on a 256 MB boundary. This field corresponds to A[31:28]. Setting of “1111” corresponds to 3.75 GB DRAM, and so on down to “0001” corresponds to 0.25 GB DRAM. “0000” setting is illegal and a programming error.
11:0	RV	000h	Reserved

3.9.22.2 MIR[2:0] - Memory Interleave Range

These registers define each memory branch's interleave participation in processor-physical (A) space.

Each register defines a range. If the processor-physical address falls in the range defined by an MIR, the "way" fields in that MIR defines branch participation in the interleave. The way-sensitive address bit is A[6]. For a MIR to be effective, WAY0 and WAY1 fields can not be set to 00b. In mirror mode, the WAY0 and WAY1 fields should be set to 11 b. Matching addresses participate in the corresponding ways.

Compensation for a non-4 GB MMIO gap size is performed by adjusting the limit of each range upward if it is above TOLM as shown in Table 3-47.

MIR updates can only occur in the RESET, READY, FAULT, DISABLED, RECOVERYRESET, RECOVERYFAULT, and RECOVERYREADY states.

Table 3-47. Interleaving of an address is governed by MIR[i]

Limit with respect to TOLM	Match MIR[i]
if MIR[i].LIMIT[7:0] <= TOLM	then MIR[i].LIMIT[7:0] > A[35:28] >= MIR[i-1].LIMIT[7:0]
if MIR[i].LIMIT[7:0] > TOLM > MIR[i-1].LIMIT[7:0]	then MIR[i].LIMIT[7:0] + (10H - TOLM) > A[35:28] >= MIR[i-1][a].LIMIT[7:0]
if MIR[i].LIMIT[7:0] > MIR[i-1].LIMIT[7:0] >= TOLM	then MIR[i].LIMIT[7:0] + (10H - TOLM) > A[35:28] >= MIR[i-1].LIMIT[7:0] + (10H - TOLM)

Notes:

a. For MIR[0], MIR[i-1] is defined to be 0.

Device: 16
Function: 1
Offset: 88h, 84h, 80h

Bit	Attr	Default	Description
15:4	RW	000h	**LIMIT** This field defines the highest address in the range A[39:28] prior to modification by the TOLM register. Since MIRs only comprehend 64GB - (TOLM.TOLM * 256MB) of address space, LIMIT[11:8] (bits [15:12] of this register) are ignored, and the largest legal value is (64GB - (TOLM.TOLM * 256MB)) / 2^{28}.
3:2	RV	00	Reserved
1	RW	0	**WAY1** Branch 1 participate in this MIR range if this bit is set AND (the way-sensitive bit is 1b OR WAY0 of this MIR is 0b).
0	RW	0	**WAY0** Branch 0 participate in this MIR range if this bit is set AND (the way-sensitive bit is 1b OR WAY1 of this MIR is 0b).

3.9.22.3 AMIR[2:0] - Adjusted Memory Interleave Range

For the convenience of software which is trying to determine the physical location to which a processor bus address is sent, 16 scratch bits are associated with each MIR.

Device: 16 Function: 1 Offset: 94h, 90h, 8Ch			
Bit	**Attr**	**Default**	**Description**
15:0	RW	0000h	**ADJLIMIT:** Adjusted MIR Limit

3.9.23 FB-DIMM Error Registers

3.9.23.1 FERR_FAT_FBD - FB-DIMM First Fatal Errors

The first fatal error for an FB-DIMM branch is flagged in these registers. Only one flag is ever set. Lower-numbered branches have higher priority than higher-numbered branches. Lower-numbered channels have higher priority than higher-numbered channels. Higher-order error bits within a register have higher priority than lower-order bits. The FBDChan_Indx field is not an error. This register will display invalid index channel data until an error has occurred.

Device: 16 Function: 1 Offset: 98h			
Bit	**Attr**	**Default**	**Description**
31:30	RV	00	Reserved
29:28	RV	00	**FBDChan_Indx:** Logs channel in which the error occurred
27:3	RV	0000000h	Reserved
2	RWCST	0	**M3Err:** >Tmid Thermal event with intelligent throttling disabled
1	RWCST	0	**M2Err:** Northbound CRC error on non-redundant retry
0	RWCST	0	**M1Err:** Alert on non-redundant retry or fast reset timeout

3.9.23.2 NERR_FAT_FBD - FB-DIMM Next Fatal Errors

If an error is already flagged in FERR_FAT_FBD, subsequent and lower-priority fatal errors are logged in NERR_FAT_FBD.

Device: 16 Function: 1 Offset: 9Ch			
Bit	**Attr**	**Default**	**Description**
31:3	RV	0h	Reserved
2	RWCST	0	**M3Err:** >Tmid Thermal event with intelligent throttling disabled
1	RWCST	0	**M2Err:** Northbound CRC error on non-redundant retry
0	RWCST	0	**M1Err:** Alert on non-redundant retry or fast reset timeout

3.9.23.3 FERR_NF_FBD - FB-DIMM First Non-Fatal Errors

The first non-fatal error for a FB-DIMM branch is flagged in these registers. Only one flag is ever set. Lower-numbered branches have higher priority than higher-numbered branches. Lower-numbered channels have higher priority than higher-numbered

channels. Higher-order error bits within a register have higher priority than lower-order bits. The FBDChan_Indx field is not an error. This register will display invalid index channel data until an error has occurred.

Device: 16
Function: 1
Offset: A0h

Bit	Attr	Default	Description
31:30	RV	00	Reserved
29:28	RWCST	00	**FBDChan_Indx: Logs channel in which the error occurred** The least-significant-bit of this field has no significance for M4Err through M12Err and M17Err through M20Err. The least-significant-bit of this field only bears significance for M13Err through M15Err and M21Err and higher.
27:25	RV	0h	Reserved
24	RWCST	0	**M28Err:** DIMM-Spare Copy Completed
23	RWCST	0	**M27Err:** DIMM-Spare Copy Started
22	RWCST	0	**M26Err:** <Tmid Thermal Event with intelligent throttling enabled
21	RWCST	0	**M25Err:** <Tlow Thermal Event with intelligent throttling enabled
20	RWCST	0	**M24Err:** >Tmid Thermal Event with intelligent throttling enabled
19	RWCST	0	**M23Err:** >Tlow Thermal Event with intelligent throttling enabled
18	RWCST	0	**M22Err:** SPD protocol Error
17	RWCST	0	**M21Err:** FBD Northbound CRC error on FBD Sync Status
16	RWCST	0	**M20Err:** Correctable Patrol Data ECC
15	RWCST	0	**M19Err:** Correctable Spare-Copy Data ECC
14	RWCST	0	**M18Err:** Correctable Mirrored Demand Data ECC
13	RWCST	0	**M17Err:** Correctable Non-Mirrored Demand Data ECC
12	RV	0	Reserved
11	RWCST	0	**M15Err:** Non-Retry or Redundant Retry FBD Northbound CRC error on read data
10	RWCST	0	**M14Err:** Non-Retry or Redundant Retry FBD Configuration Alert
9	RWCST	0	**M13Err:** Non-Retry or Redundant Retry FBD Memory Alert or Redundant Fast Reset Timeout
8	RWCST	0	**M12Err:** Non-Aliased Uncorrectable Patrol Data ECC
7	RWCST	0	**M11Err:** Non-Aliased Uncorrectable Spare-Copy Data ECC
6	RWCST	0	**M10Err:** Non-Aliased Uncorrectable Mirrored Demand Data ECC
5	RWCST	0	**M9Err:** Non-Aliased Uncorrectable Non-Mirrored Demand Data ECC
4	RWCST	0	**M8Err:** Aliased Uncorrectable Patrol Data ECC
3	RWCST	0	**M7Err:** Aliased Uncorrectable Spare-Copy Data ECC
2	RWCST	0	**M6Err:** Aliased Uncorrectable Mirrored Demand Data ECC
1	RWCST	0	**M5Err:** Aliased Uncorrectable Non-Mirrored Demand Data ECC
0	RWCST	0	**M4Err:** Uncorrectable Data ECC on Replay

3.9.23.4 NERR_NF_FBD - FB-DIMM Next Fatal Errors

If an error is already flagged in FERR_NF_FBD, subsequent and lower-priority non-fatal errors are logged in NERR_NF_FBD.

Device: 16 Function: 1 Offset: A4h			
Bit	**Attr**	**Default**	**Description**
31:25	RV	00h	Reserved
24	RWCST	0	**M28Err:** DIMM-Spare Copy Completed
23	RWCST	0	**M27Err:** DIMM-Spare Copy Started
22	RWCST	0	**M26Err:** <Tmid Thermal Event with intelligent throttling enabled
21	RWCST	0	**M25Err:** <Tlow Thermal Event with intelligent throttling enabled
20	RWCST	0	**M24Err:** >Tmid Thermal Event with intelligent throttling enabled
19	RWCST	0	**M23Err:** >Tlow Thermal Event with intelligent throttling enabled
18	RWCST	0	**M22Err:** SPD protocol Error
17	RWCST	0	**M21Err:** FBD Northbound CRC error on FBD Sync Status
16	RWCST	0	**M20Err:** Correctable Patrol Data ECC
15	RWCST	0	**M19Err:** Correctable Spare-Copy Data ECC
14	RWCST	0	**M18Err:** Correctable Mirrored Demand Data ECC
13	RWCST	0	**M17Err:** Correctable Non-Mirrored Demand Data ECC
12	RV	0	Reserved
11	RWCST	0	**M15Err:** Non-Retry or Redundant Retry FBD Northbound CRC error on read data
10	RWCST	0	**M14Err:** Non-Retry or Redundant Retry FBD Configuration Alert
9	RWCST	0	**M13Err:** Non-Retry or Redundant Retry FBD Memory Alert or Redundant Fast Reset Timeout
8	RWCST	0	**M12Err:** Non-Aliased Uncorrectable Patrol Data ECC
7	RWCST	0	**M11Err:** Non-Aliased Uncorrectable Spare-Copy Data ECC
6	RWCST	0	**M10Err:** Non-Aliased Uncorrectable Mirrored Demand Data ECC
5	RWCST	0	**M9Err:** Non-Aliased Uncorrectable Non-Mirrored Demand Data ECC
4	RWCST	0	**M8Err:** Aliased Uncorrectable Patrol Data ECC
3	RWCST	0	**M7Err:** Aliased Uncorrectable Spare-Copy Data ECC
2	RWCST	0	**M6Err:** Aliased Uncorrectable Mirrored Demand Data ECC
1	RWCST	0	**M5Err:** Aliased Uncorrectable Non-Mirrored Demand Data ECC
0	RWCST	0	**M4Err:** Uncorrectable Data ECC on Replay

3.9.23.5 EMASK_FBD - FB-DIMM Error Mask Register

A ‘0’ in any field enables that error.

Device: 16 Function: 1 Offset: A8h			
Bit	**Attr**	**Default**	**Description**
31:28	RV	00h	Reserved
27	RWST	1	**M28Err:** DIMM-Spare Copy Completed
26	RWST	1	**M27Err:** DIMM-Spare Copy Started
25	RWST	1	**M26Err:** <Tmid Thermal Event with intelligent throttling enabled

Device: 16
Function: 1
Offset: A8h

Bit	Attr	Default	Description
24	RWST	1	**M25Err:** <Tlow Thermal Event with intelligent throttling enabled
23	RWST	1	**M24Err:** >Tmid Thermal Event with intelligent throttling enabled
22	RWST	1	**M23Err:** >Tlow Thermal Event with intelligent throttling enabled
21	RWST	1	**M22Err:** SPD protocol Error
20	RWST	1	**M21Err:** FBD Northbound CRC error on FBD Sync Status
19	RWST	1	**M20Err:** Correctable Patrol Data ECC
18	RWST	1	**M19Err:** Correctable Spare-Copy Data ECC
17	RWST	1	**M18Err:** Correctable Mirrored Demand Data ECC
16	RWST	1	**M17Err:** Correctable Non-Mirrored Demand Data ECC
15	RV	0	Reserved
14	RWST	1	**M15Err:** Non-Retry or Redundant Retry FBD Northbound CRC error on read data
13	RWST	1	**M14Err:** Non-Retry or Redundant Retry FBD Configuration Alert
12	RWST	1	**M13Err:** Non-Retry or Redundant Retry FBD Memory Alert or Redundant Fast Reset Timeout
11	RWST	1	**M12Err:** Non-Aliased Uncorrectable Patrol Data ECC
10	RWST	1	**M11Err:** Non-Aliased Uncorrectable Spare-Copy Data ECC
9	RWST	1	**M10Err:** Non-Aliased Uncorrectable Mirrored Demand Data ECC
8	RWST	1	M9Err: Non-Aliased Uncorrectable Non-Mirrored Demand Data ECC
7	RWST	1	**M8Err:** Aliased Uncorrectable Patrol Data ECC
6	RWST	1	**M7Err:** Aliased Uncorrectable Spare-Copy Data ECC
5	RWST	1	**M6Err:** Aliased Uncorrectable Mirrored Demand Data ECC
4	RWST	1	**M5Err:** Aliased Uncorrectable Non-Mirrored Demand Data ECC
3	RWST	1	**M4Err:** Uncorrectable Data ECC on Replay
2	RWST	1	**M3Err:** >Tmid Thermal event with intelligent throttling disabled
1	RWST	1	**M2Err:** Northbound CRC error on retry
0	RWST	1	**M1Err:** Alert on non-redundant retry or fast reset timeout

3.9.23.6 ERR0_FBD: FB-DIMM Error 0 Mask Register

A '0' in any field enables that error. This register enables the signaling of Err[0] when an error flag is set.

Device: 16
Function: 1
Offset: ACh

Bit	Attr	Default	Description
31:28	RV	0h	Reserved
27	RWCST	1	**M28Err:** DIMM-Spare Copy Completed
26	RWCST	1	**M27Err:** DIMM-Spare Copy Started
25	RWCST	1	**M26Err:** <Tmid Thermal Event with intelligent throttling enabled
24	RWCST	1	**M25Err:** <Tlow Thermal Event with intelligent throttling enabled
23	RWCST	1	**M24Err:** >Tmid Thermal Event with intelligent throttling enabled
22	RWCST	1	**M23Err:** >Tlow Thermal Event with intelligent throttling enabled
21	RWCST	1	**M22Err:** SPD protocol Error
20	RWCST	1	**M21Err:** FBD Northbound CRC error on FBD Sync Status
19	RWCST	1	**M20Err:** Correctable Patrol Data ECC
18	RWCST	1	**M19Err:** Correctable Spare-Copy Data ECC
17	RWCST	1	**M18Err:** Correctable Mirrored Demand Data ECC
16	RWCST	1	**M17Err:** Correctable Non-Mirrored Demand Data ECC
15	RV	0	Reserved
14	RWCST	1	**M15Err:** Non-Retry or Redundant Retry FBD Northbound CRC error on read data
13	RWCST	1	**M14Err:** Non-Retry or Redundant Retry FBD Configuration Alert
12	RWCST	1	**M13Err:** Non-Retry or Redundant Retry FBD Memory Alert or Redundant Fast Reset Timeout
11	RWCST	1	**M12Err:** Non-Aliased Uncorrectable Patrol Data ECC
10	RWCST	1	**M11Err:** Non-Aliased Uncorrectable Spare-Copy Data ECC
9	RWCST	1	**M10Err:** Non-Aliased Uncorrectable Mirrored Demand Data ECC
8	RWCST	1	**M9Err**: Non-Aliased Uncorrectable Non-Mirrored Demand Data ECC
7	RWCST	1	**M8Err:** Aliased Uncorrectable Patrol Data ECC
6	RWCST	1	**M7Err:** Aliased Uncorrectable Spare-Copy Data ECC
5	RWCST	1	**M6Err:** Aliased Uncorrectable Mirrored Demand Data ECC
4	RWCST	1	**M5Err:** Aliased Uncorrectable Non-Mirrored Demand Data ECC
3	RWCST	1	**M4Err:** Uncorrectable Data ECC on Replay
2	RWCST	1	**M3Err:** >Tmid Thermal event with intelligent throttling disabled
1	RWCST	1	**M2Err:** Northbound CRC error on retry
0	RWCST	1	**M1Err:** Alert on non-redundant retry or fast reset timeout

3.9.23.7 ERR1_FBD: FB-DIMM Error 1 Mask Register

A '0' in any field enables that error. This register enables the signaling of Err[1] when an error flag is set.

Device: 16 Function: 1 Offset: B0h			
Bit	**Attr**	**Default**	**Description**
31:28	RV	0h	Reserved
27	RWCST	1	**M28Err:** DIMM-Spare Copy Completed
26	RWCST	1	**M27Err:** DIMM-Spare Copy Started
25	RWCST	1	**M26Err:** <Tmid Thermal Event with intelligent throttling enabled
24	RWCST	1	**M25Err:** <Tlow Thermal Event with intelligent throttling enabled
23	RWCST	1	**M24Err:** >Tmid Thermal Event with intelligent throttling enabled
22	RWCST	1	**M23Err:** >Tlow Thermal Event with intelligent throttling enabled
21	RWCST	1	**M22Err:** SPD protocol Error
20	RWCST	1	**M21Err:** FBD Northbound CRC error on FBD Sync Status
19	RWCST	1	**M20Err:** Correctable Patrol Data ECC
18	RWCST	1	**M19Err:** Correctable Spare-Copy Data ECC
17	RWCST	1	**M18Err:** Correctable Mirrored Demand Data ECC
16	RWCST	1	**M17Err:** Correctable Non-Mirrored Demand Data ECC
15	RV	0	Reserved
14	RWCST	1	**M15Err:** Non-Retry or Redundant Retry FBD Northbound CRC error on read data
13	RWCST	1	**M14Err:** Non-Retry or Redundant Retry FBD Configuration Alert
12	RWCST	1	**M13Err:** Non-Retry or Redundant Retry FBD Memory Alert or Redundant Fast Reset Timeout
11	RWCST	1	**M12Err:** Non-Aliased Uncorrectable Patrol Data ECC
10	RWCST	1	**M11Err:** Non-Aliased Uncorrectable Spare-Copy Data ECC
9	RWCST	1	**M10Err:** Non-Aliased Uncorrectable Mirrored Demand Data ECC
8	RWCST	1	**M9Err**: Non-Aliased Uncorrectable Non-Mirrored Demand Data ECC
7	RWCST	1	**M8Err:** Aliased Uncorrectable Patrol Data ECC
6	RWCST	1	**M7Err:** Aliased Uncorrectable Spare-Copy Data ECC
5	RWCST	1	**M6Err:** Aliased Uncorrectable Mirrored Demand Data ECC
4	RWCST	1	**M5Err:** Aliased Uncorrectable Non-Mirrored Demand Data ECC
3	RWCST	1	**M4Err:** Uncorrectable Data ECC on Replay
2	RWCST	1	**M3Err:** >Tmid Thermal event with intelligent throttling disabled
1	RWCST	1	**M2Err:** Northbound CRC error on retry
0	RWCST	1	**M1Err:** Alert on non-redundant retry or fast reset timeout

3.9.23.8 ERR2_FBD: FB-DIMM Error 2 Mask Register

A ‘0’ in any field enables that error. This register enables the signaling of Err[2] when an error flag is set.

Device: 16
Function: 1
Offset: B4h

Bit	Attr	Default	Description
31:28	RV	0h	Reserved
27	RWCST	1	**M28Err:** DIMM-Spare Copy Completed
26	RWCST	1	**M27Err:** DIMM-Spare Copy Started
25	RWCST	1	**M26Err:** <Tmid Thermal Event with intelligent throttling enabled
24	RWCST	1	**M25Err:** <Tlow Thermal Event with intelligent throttling enabled
23	RWCST	1	**M24Err:** >Tmid Thermal Event with intelligent throttling enabled
22	RWCST	1	**M23Err:** >Tlow Thermal Event with intelligent throttling enabled
21	RWCST	1	**M22Err:** SPD protocol Error
20	RWCST	1	**M21Err:** FBD Northbound CRC error on FBD Sync Status
19	RWCST	1	**M20Err:** Correctable Patrol Data ECC
18	RWCST	1	**M19Err:** Correctable Spare-Copy Data ECC
17	RWCST	1	**M18Err:** Correctable Mirrored Demand Data ECC
16	RWCST	1	**M17Err:** Correctable Non-Mirrored Demand Data ECC
15	RV	0	Reserved
14	RWCST	1	**M15Err:** Non-Retry or Redundant Retry FBD Northbound CRC error on read data
13	RWCST	1	**M14Err:** Non-Retry or Redundant Retry FBD Configuration Alert
12	RWCST	1	**M13Err:** Non-Retry or Redundant Retry FBD Memory Alert or Redundant Fast Reset Timeout
11	RWCST	1	**M12Err:** Non-Aliased Uncorrectable Patrol Data ECC
10	RWCST	1	**M11Err:** Non-Aliased Uncorrectable Spare-Copy Data ECC
9	RWCST	1	**M10Err:** Non-Aliased Uncorrectable Mirrored Demand Data ECC
8	RWCST	1	**M9Err**: Non-Aliased Uncorrectable Non-Mirrored Demand Data ECC
7	RWCST	1	**M8Err:** Aliased Uncorrectable Patrol Data ECC
6	RWCST	1	**M7Err:** Aliased Uncorrectable Spare-Copy Data ECC
5	RWCST	1	**M6Err:** Aliased Uncorrectable Mirrored Demand Data ECC
4	RWCST	1	**M5Err:** Aliased Uncorrectable Non-Mirrored Demand Data ECC
3	RWCST	1	**M4Err:** Uncorrectable Data ECC on Replay
2	RWCST	1	**M3Err:** >Tmid Thermal event with intelligent throttling disabled
1	RWCST	1	**M2Err:** Northbound CRC error on retry
0	RWCST	1	**M1Err:** Alert on non-redundant retry or fast reset timeout

3.9.23.9 MCERR_FBD - FB-DIMM MCERR Mask Register

A ‘0’ in any field enables that error. This register enables the signaling of MCERR when an error flag is set.

Device: 16 Function: 1 Offset: B8h			
Bit	**Attr**	**Default**	**Description**
31:28	RV	0h	Reserved
27	RWCST	1	**M28Err:** DIMM-Spare Copy Completed
26	RWCST	1	**M27Err:** DIMM-Spare Copy Started
25	RWCST	1	**M26Err:** <Tmid Thermal Event with intelligent throttling enabled
24	RWCST	1	**M25Err:** <Tlow Thermal Event with intelligent throttling enabled
23	RWCST	1	**M24Err:** >Tmid Thermal Event with intelligent throttling enabled
22	RWCST	1	**M23Err:** >Tlow Thermal Event with intelligent throttling enabled
21	RWCST	1	**M22Err:** SPD protocol Error
20	RWCST	1	**M21Err:** FBD Northbound CRC error on FBD Sync Status
19	RWCST	1	**M20Err:** Correctable Patrol Data ECC
18	RWCST	1	**M19Err:** Correctable Spare-Copy Data ECC
17	RWCST	1	**M18Err:** Correctable Mirrored Demand Data ECC
16	RWCST	1	**M17Err:** Correctable Non-Mirrored Demand Data ECC
15	RV	0	Reserved
14	RWCST	1	**M15Err:** Non-Retry or Redundant Retry FBD Northbound CRC error on read data
13	RWCST	1	**M14Err:** Non-Retry or Redundant Retry FBD Configuration Alert
12	RWCST	1	**M13Err:** Non-Retry or Redundant Retry FBD Memory Alert or Redundant Fast Reset Timeout
11	RWCST	1	**M12Err:** Non-Aliased Uncorrectable Patrol Data ECC
10	RWCST	1	**M11Err:** Non-Aliased Uncorrectable Spare-Copy Data ECC
9	RWCST	1	**M10Err:** Non-Aliased Uncorrectable Mirrored Demand Data ECC
8	RWCST	1	**M9Err:** Non-Aliased Uncorrectable Non-Mirrored Demand Data ECC
7	RWCST	1	**M8Err:** Aliased Uncorrectable Patrol Data ECC
6	RWCST	1	**M7Err**: Aliased Uncorrectable Spare-Copy Data ECC
5	RWCST	1	**M6Err:** Aliased Uncorrectable Mirrored Demand Data ECC
4	RWCST	1	**M5Err:** Aliased Uncorrectable Non-Mirrored Demand Data ECC
3	RWCST	1	**M4Err**: Uncorrectable Data ECC on Replay
2	RWCST	1	**M3Err**: >Tmid Thermal event with intelligent throttling disabled
1	RWCST	1	**M2Err:** Northbound CRC error on retry
0	RWCST	1	**M1Err:** Alert on non-redundant retry or fast reset timeout

3.9.23.10 NRECMEMA - Non-Recoverable Memory Error Log Register A

This register latches information on the first detected fatal memory error.

Device: 16 Function: 1 Offset: BEh			
Bit	**Attr**	**Default**	**Description**
15	RV	0	Reserved

Device: 16 Function: 1 Offset: BEh			
Bit	**Attr**	**Default**	**Description**
14:12	ROST	0h	**BANK:** Bank of the failed request
11	ROST	0	**RDWR** '0' = Read '1' = Write
10:8	ROST	0h	**RANK:** Rank of the failed request
7:0	ROST	00h	**REC_FBD_DM_BUF_ID:** DM Buffer ID of the failed request

3.9.23.11 NRECMEMB - Non-Recoverable Memory Error Log Register B

This register latches information on the first detected fatal memory error.

Device: 16 Function: 1 Offset: C0h			
Bit	**Attr**	**Default**	**Description**
31:28	RV	0h	Reserved
27:16	ROST	000h	**CAS:** CAS address of the failed request
15	RV	0	Reserved
14:0	ROST	0h	**RAS:** RAS address of the failed request

3.9.23.12 NRECFGLOG - Non-Recoverable DIMM Configuration Access Error Log Register

This register latches information on the first detected non-fatal DIMM configuration register access.

Device: 16 Function: 1 Offset: C4h			
Bit	**Attr**	**Default**	**Description**
31:28	RV	0h	Reserved
27:24	ROST	0h	**BE:** Byte Enables of the failed request
23:16	ROST	00h	**REG:** Register Address of the failed request
15:12	RV	0h	Reserved
11	ROST	0	**RDWR** '0' = Read '1' = Write
10:8	ROST	0h	**FUNCTION:** Function Number of the failed request
7:0	ROST	00h	**CFG_FBD_DM_BUF_ID:** DM Buffer ID of the failed request

3.9.23.13 NRECFBDA: Non-Recoverable FB-DIMM Error Log Register A

The NRECFBD registers defined below (A through E) have the following mapping:

Table 3-48. NRECFBD Mapping Information

Bits	Description
155:144	CRC
143:128	ECC
127:0	DATA

This register latches information on the first detected fatal northbound CRC error.

Device: 16 Function: 1 Offset: C8h			
Bit	**Attr**	**Default**	**Description**
31:0	ROST	0h	**BITS:** Bits [31:0] of the packet

3.9.23.14 NRECFBDB - Non-Recoverable FB-DIMM Error Log Register B

This register latches information on the first detected fatal northbound CRC error.

Device: 16 Function: 1 Offset: CCh			
Bit	**Attr**	**Default**	**Description**
31:0	ROST	0h	**BITS:** Bits [63:32] of the packet

0.0.0.1 NRECFBDC - Non-Recoverable FB-DIMM Error Log Register C

This register latches information on the first detected fatal northbound CRC error.

Device: 16 Function: 1 Offset: D0h			
Bit	**Attr**	**Default**	**Description**
31:0	ROST	0h	**BITS:** Bits [95:64] of the packet

3.9.23.15 NRECFBDD - Non-Recoverable FB-DIMM Error Log Register D

This register latches information on the first detected fatal northbound CRC error.

Device: 16 Function: 1 Offset: D4h			
Bit	**Attr**	**Default**	**Description**
31:0	ROST	0h	**BITS:** Bits [127:96] of the packet

3.9.23.16 NRECFBDE - Non-Recoverable FB-DIMM Error Log Register E

This register latches information on the first detected fatal northbound CRC error.

Device: 16 Function: 1 Offset: D8h			
Bit	**Attr**	**Default**	**Description**
31:28	RV	0h	Reserved
27:0	ROST	0h	**BITS:** Bits [155:128] of the packet

3.9.23.17 REDMEMB: Recoverable Memory Data Error Log Register B

This register latches information on the first detected correctable ECC error.

Device: 16 Function: 1 Offset: 7Ch			
Bit	**Attr**	**Default**	**Description**
31:18	RV	0	*Reserved*
17:0	ROST	0h	**ECC _Locator:** identifies the adjacent symbol pair in error for correctable errors according to Table 3-49, Figure 5-2 and Figure 5-4.

Table 3-49. ECC Locator Mapping Information

Symbols	Locator Bit
DS[1:0]	0
DS[3:2]	1
DS[5:4]	2
DS[7:6]	3
DS[9:8]	4
DS[11:10]	5
DS[13:12]	6
DS[15:14]	7
CS[1:0]	8
DS[17:16]	9
DS[19:18]	10
DS[21:20]	11
DS[23:22]	12
DS[25:24]	13
DS[27:26]	14
DS[29:28]	15
DS[31:30]	16
CS[3:2]	17

3.9.23.18 RECMEMA - Recoverable Memory Error Log Register A

This register latches information on the first detected non-fatal memory error.

Device: 16
Function: 1
Offset: E2h

Bit	Attr	Default	Description
15	RV	0	Reserved
14:12	ROST	0h	**BANK:** Bank of the failed request
11	ROST	0	RDWR '0' = Read '1' = Write
10:8	ROST	0h	**RANK:** Rank of the failed request
7:0	ROST	00h	**REC_FBD_DM_BUF_ID:** DM Buffer ID of the failed request

3.9.23.19 RECMEMB - Recoverable Memory Error Log Register B

This register latches information on the first detected non-fatal memory error.

Device: 16
Function: 1
Offset: E4h

Bit	Attr	Default	Description
31:28	RV	0h	Reserved
27:16	ROST	000h	**CAS:** CAS address of the failed request
15	RV	0	Reserved
14:0	ROST	0h	**RAS:** RAS address of the failed request

3.9.23.20 RECFGLOG - Recoverable DIMM Configuration Access Error Log Register

This register latches information on the first detected fatal DIMM configuration register access.

Device: 16
Function: 1
Offset: E8h

Bit	Attr	Default	Description
31:28	RV	0h	Reserved
27:24	ROST	0h	**BE:** Byte Enables of the failed request
23:16	ROST	00h	**REG:** Register Address of the failed request
15:12	RV	0h	Reserved
11	ROST	0	**RDWR** '0' = Read '1' = Write
10:8	ROST	0h	**FUNCTION:** Function Number of the failed request
7:0	ROST	00h	**CFG_FBD_CE_BUF_ID:** DM Buffer ID of the failed request

3.9.23.21 RECFBDA - Recoverable FB-DIMM Error Log Register A

This register latches information on the first northbound CRC error.

Device: 16
Function: 1
Offset: ECh

Bit	Attr	Default	Description
31:0	ROST	0h	**BITS:** Bits [31:0] of the packet

3.9.23.22 RECFBDB - Recoverable FB-DIMM Error Log Register B

This register latches information on the first detected non-fatal northbound CRC error.

Device: 16
Function: 1
Offset: F0h

Bit	Attr	Default	Description
31:0	ROST	0h	**BITS:** Bits [63:32] of the packet

3.9.23.23 RECFBDC - Recoverable FB-DIMM Error Log Register C

This register latches information on the first detected non-fatal northbound CRC error.

Device: 16
Function: 1
Offset: F4h

Bit	Attr	Default	Description
31:0	ROST	0h	**BITS:** Bits [95:64] of the packet

3.9.23.24 RECFBDD - Recoverable FB-DIMM Error Log Register D

This register latches information on the first detected non-fatal northbound CRC error.

Device: 16
Function: 1
Offset: F8h

Bit	Attr	Default	Description
31:0	ROST	0h	**BITS:** Bits [127:96] of the packet

3.9.23.25 RECFBDE - Recoverable FB-DIMM Error Log Register E

This register latches information on the first detected non-fatal northbound CRC error.

Device: 16
Function: 1
Offset: FCh

Bit	Attr	Default	Description
31:28	RV	0h	*Reserved*
27:0	ROST	0h	**BITS:** Bits [155:128] of the packet

3.9.24 FB-DIMM Branch Registers

There are two sets of the following registers, one set for each FB-DIMM branch. They each appear in function 0 of different devices as shown in Table 3-3.

3.9.24.1 FBDLVL[1:0][1:0] - FB-DIMM Packet Levelization

This register controls the FB-DIMM channel delays.

Device: 21, 22
Function: 0
Offset: 45h, 44h

Bit	Attr	Default	Description
7:6	RV	00	Reserved
5:0	RO	0h	**TRRL:** Read Round-Trip Latency Measured from issue of the FB-DIMM channel's southbound TS2 packet header to the arrival of its northbound response header.

3.9.24.2 FBDHPC[1:0]: FBD State Control

This register controls the FBD channel for Initialization and Mirroring Recovery. It consists of a next State field.

The index in FBDHPC[index] associates the FBDHPC with branch[index]. FBDHPC[0] is associated with FBD branch 0, FBDHPC[1] is associated with FBD branch 1.

When software writes to FBDHPC[x].NEXTSTATE, the transition will take effect on one or both channels within the branch depending on whether the branch is operating is single- or dual-channel mode.

When BNB hardware transitions FBDST.STATE with the following encodings: 1) disabled, 2) redundant, 3) recovery failed, 4) redundancy loss, and 5) reset, it will transition states of one or both channels within the same branch depending on whether the branch is operating in single- or dual-channel mode.

Device[a]:22, 21
Function:0
Offset:4Fh

Bit	Attr	Default	Description
7:0	RW	00h	NEXTSTATE: FBD Branch State Control field This field is written by software to change the branch state. It returns the last value written when read. Some states can only be entered under hardware control and should not be written by software. 00h: Reset 10h: Init 20h: **Ready**[b] 30h: **Active**[b] 40h: **Redundant**[b] 50h: Disabled 60h: **Redundancy Loss**[b] - may not be written 70h: Recovery Reset - (should only be selected when MC.MIRROR is set) 80h: Recovery Init - (should only be selected when MC.MIRROR is set) 90h: **Recovery Ready**[b] - (should only be selected when MC.MIRROR is set) A0h: **Resilver**[b] - (should only be selected when MC.MIRROR is set) (should not be written while FBDST.STATE=Resilver) B0h: Recovery Fault C0h: Recovery Failed D0h: Fault

Notes:

a. The nomenclature is Device 22 (branch 1), 21 (branch 0)
b. Both sync and refresh packets are sent during this mode.

3.9.24.3 FBDST[1:0] - FB-DIMM Status

These registers are inspected by software to determine the current FB-DIMM branch state. This register contains Mirroring recovery state, and Initialization state.

The indexing scheme is the same as in FBDHPC registers. The current FB-DIMM branch state field indicates state for one or both channels within the same branch depending on whether the branch is operating in single- or dual-channel mode.

Device: 21, 22
Function: 0
Offset: 4Bh

Bit	Attr	Default	Description
7:0	ROST	00h	**STATE:** FBD Branch State This field describes the current state of the FB-DIMM branch. It can be read by software to determine which FB-DIMM branch is being sequenced through recovery, and how far the FB-DIMM branch has progressed. 00h: Reset 10h: Init 20h: Ready 30h: Active 40h: Redundant 50h: Disabled 60h: Redundancy Loss - may not be written 70h: Recovery Reset - (should only be selected when MC.MIRROR is set) 80h: Recovery Init - (should only be selected when MC.MIRROR is set) 90h: Recovery Ready - (should only be selected when MC.MIRROR is set) A0h: Reserved B0h: Recovery Fault C0h: Recovery Failed D0h: Fault This field is only sticky through hard reset when SYRE.S3 is set. This field is not sticky through hard reset when SYRE.S3 is cleared.

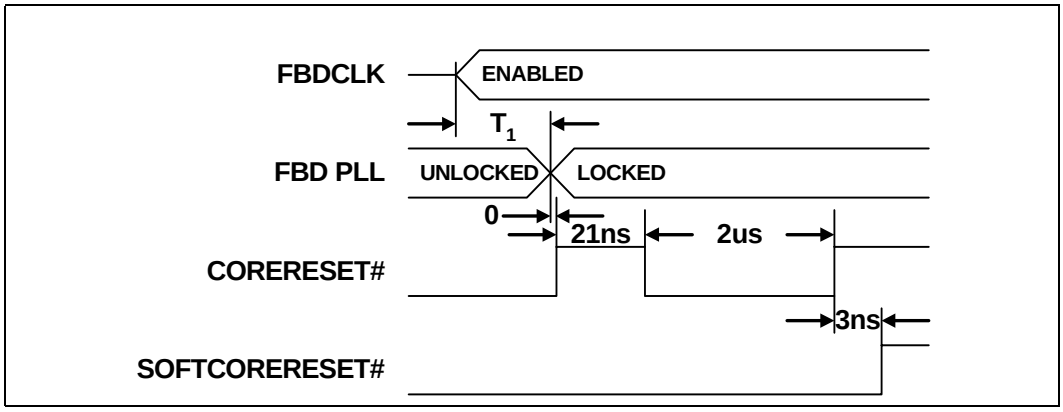
3.9.24.4 FBDRST[1:0] - FB-DIMM Reset

The FB-DIMM I/O blocks are reset separately from the rest of the Intel 5000X Chipset MCH. These blocks, composed of FAST-clocked (GHz unit-interval clocked) logic, are supplied by a PLL whose FBDCLK is not available when PWRGOOD is asserted. After FBDCLK is enabled and the FB-DIMM PLL has acquired lock, CORERESET# is deasserted for a minimum of 21 ns, then asserted for a minimum of 2 us. After the 2 us assertion, CORERESET# is deasserted followed by a minimum delay of 3ns at which time SOFTCORERESET# is deasserted. If the platform removes FBDCLK on a hot-remove of the branch, CORERESET# and SOFTCORERESET# must be asserted prior to loss of FBDCLK.

A “disabled” (not enabled) FBDCLK is floated at the source, and pulled to ground through the termination near the receiver in the Intel 5000X Chipset MCH.

All timing specifications in Figure 3-6 are minimums. After the sequence in Figure 3-6 has been executed, the FB-DIMM branch is ready for initialization.

Figure 3-6. FB-DIMM Reset Timing



Device: 21, 22
Function: 0
Offset: 53h

Bit	Attr	Default	Description
7:3	RV	00h	*Reserved*
2	RWST	0	**BRSELCMPRESET: Branch Select for Compensation Reset** 0: COMPreset is tied to CORERESET# from branch 0 1: COMPreset is tied to CORERESET# from branch 1 For Branch 1 to be selected for reset, this field has to be a '1' for both branch instances.
1	RWST	0	**SOFTCORERESET#: Soft Core Reset** See Timing diagram Figure 3-6. 0: Soft Core Reset Asserted 1: Soft Core Reset De-Asserted
0	RWST	0	**CORERESET#: Core Reset** See Timing diagram Figure 3-6. 0: Core Reset Asserted 1: Core Reset De-Asserted

3.9.24.5 SPCPC[1:0] - Spare Copy Control

These controls set up sparing for each branch. Branch zero (device 21) takes precedence over branch one (device 22): if both spare-control-enabled branches' spare error thresholds trigger in the same cycle, sparing will only commence on branch zero. Sparing will not commence on a competing branch until its in-progress competitor's spare control enable is cleared and it's UERRCNT/CERRCNT criteria is still met.

Device: 21, 22
Function: 0
Offset: 40h

Bit	Attr	Default	Description
7:4	RW	0h	**SETH: Spare Error Threshold** A spare fail-over operation will commence when the SPAREN bit is set and a UERRCNT.RANK[i] and/or CERRCNT.RANK[i] count for one and only one rank hits this threshold.

Device: 21, 22 Function: 0 Offset: 40h			
Bit	**Attr**	**Default**	**Description**
3:1	RWL	0h	**SPRANK: Spare Rank** Target of the spare copy operation. This rank should not initially appear in a DMIR.RANK field. After the spare copy, Intel 5000X Chipset MCH will update the failed DMIR.RANK fields with this value. Enabled by SPAREN. Changes to this register will not be acknowledged by the hardware while SPCPS.DSCIP is set.
0	RW	0	**SPAREN: Spare Control Enable** '1' enables sparing, '0' disables sparing. The SPRANK field defines other characteristics of the sparing operation. The Intel 5000X Chipset MCH does not support sparing in mirrored mode: this bit should not be set if MC.MIRROR is set. If this bit is cleared before SPCPS.SFO is set, then if this bit is subsequently set while the spare trigger is still valid, then the spare copy operation will not resume from where it left off, but will instead restart from the beginning.

3.9.24.6 SPCPS[1:0] - Spare Copy Status

Device: 21, 22 Function: 0 Offset: 41h			
Bit	**Attr**	**Default**	**Description**
7:6	RV	00	Reserved
5	RO	0	**LBTHR:** Leaky Bucket Threshold Reached '0' = Leaky-bucket threshold not reached '1' = Leaky-bucket count matches SPCPC.SETH. Generates error M27. Cleared by reducing the offending count(s) in the UERRCNT/CERRCNT registers.
4	RO	0	**DSCIP: DIMM Sparing Copy In Progress** '0' = DIMM sparing copy not in progress. '1' = DIMM sparing copy in progress. Set when SPCPC.SPAREN is set, and only one rank in UERRCNT/CERRCNT is at threshold. This bit remains set until SFO is set. This bit is cleared when SFO is set. Error M27 is set when this bit transitions from '0' to '1'.
3:1	RO	000	**FR: Failed Rank** Rank that was spared. Updated with the UERRCNT/CERRCNT rank that has reached threshold when DSCIP is set.
0	RO	0	**SFO: Spare Fail-Over** '0' = Spare has not been substituted for failing DIMM rank. '1' = Spare has been substituted for failing DIMM rank. Generates error M28. Cleared when SPCPC.SPAREN is cleared.

3.9.24.7 MTR[1:0][3:0] - Memory Technology Registers

These registers define the organization of the DIMM's. There is one MTR for each pair of slots comprising either one or two ranks. The parameters for these devices can be obtained by serial presence detect.

MTR[3:0] defines slot-pairs [3:0] on branch[0]. MTR[7:4] defines slot-pairs [3:0] on branch[1].
MTR[3:0] in Table 3-24 is MTR[3:0] for Device 21 which is MTR[3:0] for this Section 3.9.24.7.
MTR[3:0] in Table 3-24 is MTR[3:0] for Device 22 which is MTR[7:4] for this Section 3.9.24.7.

This register must not be modified while servicing memory requests.

Device: 21, 22 Function: 0 Offset: 8Ch, 88h, 84h, 80h			
Bit	**Attr**	**Default**	**Description**
15:9	RV	00h	Reserved
8	RW	0	**PRESENT: DIMMs are present** This bit is set if both DIMMs are present and their technologies are compatible.
7	RW	0	**ETHROTTLE: Technology - Electrical Throttle** Defines the electrical throttling level for these DIMMs: '0' = Electrical Throttling is disabled '1' = Electrical Throttling is enabled using the throttling level defined by the MC.ETHROT configuration field.
6	RW	0	**WIDTH: Technology - Width** Defines the data width of the SDRAMs used on these DIMMs '0' = x4 (4 bits wide) '1' = x8 (8 bits wide)
5	RW	0	**NUMBANK: Technology - Number of Banks** Defines the number of (real, not shadow) banks on these DIMMs '0' = four-banked '1' = eight-banked
4	RW	0	**NUMRANK: Technology - Number of Ranks** Defines the number of ranks on these DIMMs. '0' = single ranked '1' = double ranked
3:2	RW	00	**NUMROW: Technology - Number of Rows** Defines the number of rows within these DIMMs. "00"= 8,192, 13 rows "01"= 16,384, 14 rows "10"= 32,768, 15 rows "11"= Reserved
1:0	RW	00	**NUMCOL: Technology - Number of Columns** Defines the number of columns within these DIMMs "00"= 1,024, 10 columns "01"= 2,048, 11 columns "10"= 4,096, 12 columns "11"= Reserved

3.9.24.8 DMIR[1:0][4:0] - DIMM Interleave Range

These registers define rank participation in various DIMM interleaves.

Each register defines a range. If the Memory (M) address falls in the range defined by an adjacent pair of DMIR.LIMIT's, the rank fields in the upper DMIR define the number and interleave position of ranks' way participation. Matching addresses participate in the corresponding ways. The combination of two equal ranks with three unequal ranks is illegal.

When a DMIR is programmed for a 2-way interleave, RANK0/RANK2 should be with the same rank number and RANK1/RANK3 should be another rank number.

This register must not be modified while servicing memory requests.

Device: 21, 22 Function: 0 Offset: A0h, 9Ch, 98h, 94h, 90h			
Bit	**Attr**	**Default**	**Description**
31:24	RV	000h	Reserved

Device: 21, 22 Function: 0 Offset: A0h, 9Ch, 98h, 94h, 90h			
Bit	**Attr**	**Default**	**Description**
23:16	RW	00h	**LIMIT** This field defines the highest address in the range. Memory requests participate in this DMIR range if LIMIT[i] > M[34:28] >= LIMIT[i-1]. For i = 0, LIMIT[i-1]=0 (M[35] is considered as zero for the purpose of this comparison).
15:12	RV	0h	Reserved
11:9	RW	000	**RANK3** Defines which rank participates in WAY3.
8:6	RW	000	**RANK2** Defines which rank participates in WAY2.
5:3	RW	000	**RANK1** Defines which rank participates in WAY1.
2:0	RW	000	**RANK0** Defines which rank participates in WAY0.

3.9.24.9 FBDICMD[1:0][1:0] - FB-DIMM Initialization Command

These registers define channel behavior during the “Init”, “Recovery Init”, “Reset”, and “Recovery Reset” Hot-Plug states. The “AMBID” field for the even-numbered channel also defines branch behavior during fast reset.

Device: 21, 22 Function: 0 Offset: 47h, 46h			
Bit	**Attr**	**Default**	**Description**
7	RW	0	**EN: Enable** ‘0’ = Drive electrical idle on the channel. ‘1’ = Drive INITPAT on the channel. This field is not used during fast reset.
6:4	RW	00	**INITPAT:** Initialization pattern “000”=TS0: Training Sequence 0 to last AMB (not valid in “Reset”) “001”=TS1: Training Sequence 1 to last AMB (not valid in “Reset”) “010”=TS2: Training Sequence 2 to last AMB (not valid in “Reset”) “011”=TS3: Training Sequence 3 to last AMB (not valid in “Reset”) “100”=*reserved* “101”=TS2: Training Sequence 2 not to last AMB with NB Merge disabled (not valid in “Reset”) “110”=TS2: Training Sequence 2 not to last AMB with NB Merge enabled (not valid in “Reset”) “111”=All Ones (valid only in “Reset”) This pattern is superseded by the “EN” bit. This field is not used during fast reset. The note ‘(not valid in “Reset”)’ indicates that is not valid when FBDST.STATE=”Reset” or “Recovery Reset” and EN=’1’. The note ‘(valid only in “Reset”)’ indicates that this is valid only when FBDST.STATE=”Reset” or “Recovery Reset”.
3:0	RW	0h	**AMBID: Advanced Memory Buffer IDentifier** Driven during the training sequences. This field is also used during fast reset to identify the last (southernmost) DIMM.

3.9.24.10 FBDISTS[1:0][1:0] - FB-DIMM Initialization Status

The contents of this register are valid only during “Initialization” states. The thirteen bits [12:0] correspond to the northbound bit-lanes.

Device:	**21**
Function:	**0**
Offset:	**5Ah, 58h**
Version:	**Intel 5000P Chipset, Intel 5000V Chipset, Intel 5000Z chipset, Intel 5000X Chipset**
Device:	**22**
Function:	**0**
Offset:	**5Ah, 58h**
Version:	**Intel 5000P Chipset, Intel 5000X Chipset**

Bit	Attr	Default	Description
15:13	RV	000	Reserved
12:0	RO	0000h	**PATDET: Pattern Detection** "1" = Pattern recognized. "0" = Pattern not recognized. Bit-Lane Status is evaluated at the end of each instance of the pattern specified by the FBDICMD.EN and FBDICMD.INITPAT fields. Bit-Lane status is evaluated on each change to the FBDICMD.EN and FBDICMD.INITPAT.Only bits [2:0] are valid during electrical idle, and only after the FBDRST reset sequence has been executed. A recognizable training sequence must contain the FBDICMD.AMBID. TS1 detection is qualified by test patterns specified in section 4.3 of *rev. 0.75 of FBD DFx specification*, which defines the "SB/NB_Mapping" (1 bit), the "Test Parameters" (24 bits), and the "Electrical Stress Pattern".

3.9.24.11 AMBPRESENT[1:0][1:0] - FB-DIMM AMB Slot Present Register

These registers control configuration transaction routing to AMB slots on a per FB-DIMM channel basis. This includes both accesses through memory mapped region (based on AMBASE register, see Section 3.8.3.1) and AMBSELECT (for SMBus/JTAG access only, access via device 9, function 0. See Section 3.8.3.3). Software needs to program this register after SPD discovery process. Intel 5000X Chipset MCH will check this register before it sends actual FB-DIMM AMB configuration transaction.

Device:	**21, 22**
Function:	**0**
Offset:	**66h, 64h**

Bit	Attr	Default	Description
15:0	RWO	0h	**AMBSP: Slot [bit_position] present in the FBD channel** 1: Indicates AMB slot addressed by DS[3:0] in decimal = [bit_position] is present; configuration transaction will be routed to FB-DIMM channel. Bit 15 controls DS[3:0] = 1111b, bit 14 controls DS[3:0] = 1110b,..., bit 0 controls DS[3:0] = 0000b. 0: AMB slot addressed by DS[3:0] in decimal = [bit_position] is not populated; no configuration transaction will be sent to FB-DIMM channel.

3.9.25 FB-DIMM RAS Registers

There are two sets of the following registers, one set for each FB-DIMM branch. They each appear in function 0 of different devices as shown in Table 3-3.

3.9.25.1 UERRCNT[1:0] - Uncorrectable Error Count

This register implements the “leaky-bucket” counters for uncorrectable errors for each rank. Each field “limits” at a value of “15” (“1111”). Non-zero counts are decremented when the ERRPER threshold is reached by the error period counter. Counts are frozen at the threshold defined by SPCPC.SETH and set the SPCPS.LBTHR bit. Writing a value of “1111” clears and thaws the count. Changing SPCPC.SETH has no effect upon a frozen count.

Note: Aliased uncorrectable errors are NOT counted as uncorrectable errors in the implementation of this register. They are treated as correctable errors and logged in the CERRCNT register.

Device: 21, 22
Function: 0
Offset: A4h

Bit	Attr	Default	Description
31:28	RWCST	0h	**RANK7:** Error Count for Rank 7
27:24	RWCST	0h	**RANK6**: Error Count for Rank 6
23:20	RWCST	0h	**RANK5:** Error Count for Rank 5
19:16	RWCST	0h	**RANK4:** Error Count for Rank 4
15:12	RWCST	0h	**RANK3:** Error Count for Rank 3
11:8	RWCST	0h	**RANK2:** Error Count for Rank 2
7:4	RWCST	0h	**RANK1:** Error Count for Rank 1
3:0	RWCST	0h	**RANK0:** Error Count for Rank 0

3.9.25.2 CERRCNT[1:0] - Correctable Error Count

This register implements the “leaky-bucket” counters for correctable errors for each rank. Each field “limits” at a value of “15” (“1111”). Non-zero counts are decremented when the ERRPER threshold is reached by the error period counter. Counts are frozen at the threshold defined by SPCPC.SETH and set the SPCPS.LBTHR bit. Writing a value of “1111” clears and thaws the count. Changing SPCPC.SETH has no effect upon a frozen count.

Note: Aliased uncorrectable errors are counted as correctable errors in the implementation of this register.

Device: 21, 22
Function: 0
Offset: A8h

Bit	Attr	Default	Description
31:28	RWCST	0h	**RANK7:** Error Count for Rank 7
27:24	RWCST	0h	**RANK6:** Error Count for Rank 6
23:20	RWCST	0h	**RANK5:** Error Count for Rank 5
19:16	RWCST	0h	**RANK4:** Error Count for Rank 4

Device: 21, 22 Function: 0 Offset: A8h			
Bit	**Attr**	**Default**	**Description**
15:12	RWCST	0h	**RANK3:** Error Count for Rank 3
11:8	RWCST	0h	**RANK2:** Error Count for Rank 2
7:4	RWCST	0h	**RANK1:** Error Count for Rank 1
3:0	RWCST	0h	**RANK0:** Error Count for Rank 0

3.9.25.3 BADRAMA[1:0] - Bad DRAM Marker A

This register implements “failed-device” markers for the enhanced demand scrub algorithm. Hardware “marks” bad devices. The “mark” is a number between 1 and 18 inclusive. A value of “0 0000h” indicates an “un-marked” rank: all RAM’s are presumed “good”. Only ranks containing x8 DRAM are “marked”.

Device: 21, 22 Function: 0 Offset: ACh Version: Intel 5000P chipset, Intel 5000V chipset, Intel 5000Z chipset, Intel 5000X Chipset Device: 22 Function: 0 Offset: ACh Version: Intel 5000P chipset, Intel 5000X Chipset			
Bit	**Attr**	**Default**	**Description**
31:30	RV	00	Reserved
29:25	RWCST	00h	**RANK5:** Bad device in Rank 5
24:20	RWCST	00h	**RANK4:** Bad device in Rank 4
19:15	RWCST	00h	**RANK3:** Bad device in Rank 3
14:10	RWCST	00h	**RANK2:** Bad device in Rank 2
9:5	RWCST	00h	**RANK1:** Bad device in Rank 1
4:0	RWCST	00h	**RANK0:** Bad device in Rank 0

3.9.25.4 BADRAMB[1:0] - Bad DRAM Marker B

This register implements “failed-device” markers for the enhanced demand scrub algorithm. Hardware “marks” bad devices. The “mark” is a number between 1 and 18 inclusive. A value of “0_0000” indicates an “un-marked” rank: all DRAM’s are presumed “good”. Only ranks containing x8 DRAM are “marked”.

Device: 21, 22 Function: 0 Offset: B0h			
Bit	**Attr**	**Default**	**Description**
15:10	RV	00h	Reserved
9:5	RWCST	00h	**RANK7:** Bad device in Rank 7
4:0	RWCST	00h	**RANK6:** Bad device in Rank 6

3.9.25.5 BADCNT[1:0] - Bad DRAM Counter

This register implements “failing-device” counters for the aliased uncorrectable error identification algorithm. “Count” double-adjacent symbol errors within x8 devices. “Drip” each counter after “MC.BADRAMTH” patrol scrub cycles through all of memory. Values of “MC.BADRAMTH” and “0” cannot be “dripped”. A value of “MC.BADRAMTH” cannot be incremented. “Mark” the BADRAM(A/B) register when a count reaches “MC.BADRAMTH”.

Device: 21, 22
Function: 0
Offset: B4h

Bit	Attr	Default	Description
31:28	RWCST	0000	**RANK7:** Adjacent x8 symbol error count in Rank 7
27:24	RWCST	0000	**RANK6:** Adjacent x8 symbol error count in Rank 6
23:20	RWCST	0000	**RANK5:** Adjacent x8 symbol error count in Rank 5
19:16	RWCST	0000	**RANK4:** Adjacent x8 symbol error count in Rank 4
15:12	RWCST	0000	**RANK3:** Adjacent x8 symbol error count in Rank 3
11:8	RWCST	0000	**RANK2:** Adjacent x8 symbol error count in Rank 2
7:4	RWCST	0000	**RANK1:** Adjacent x8 symbol error count in Rank 1
3:0	RWCST	0000	**RANK0:** Adjacent x8 symbol error count in Rank 0

3.9.25.6 FBDSBTXCFG[1:0][1:0]: FB-DIMM Southbound Transmit Configuration Register

This register controls the FB-DIMM Southbound I/O Transmit configuration during normal operation. This value is programmed by BIOS on per channel basis.

Device: 21, 22
Function: 0
Offset: C1h, C0h

Bit	Attr	Default	Description
7:4	RV	0h	*Reserved.*
3:2	RWST	01	**SBTXDRVCUR:** South Bound Tx drive Current 00: 120% current 01: 100% current 10: 80% current 11: 60% current
1:0	RWST	00	**SBTXDEEMP:** South Bound Tx De-emphasis With De-emphasis, the Tx differential p-p swing (eye height) is maintained at nominal during data transitions, but drops down to the de-emphasized value when there is no transition between the previous bit and current bit 00: No De-emphasis 01: -3.5dB 10: -6dB 11: -9.5 dB

3.9.26 FB-DIMM Intel IBIST Registers

3.9.26.1 FBD[3:2]IBPORTCTL: FB-DIMM Intel IBIST Port Control Register

This register contains bits to control the operation of the Intel IBIST DFT feature.

Device: 22
Function: 0
Offset: 280h, 180h

Bit	Attr	Default	Description
31:26	RV	0h	*Reserved*
25	RWST	0	**RXINVSWPMD:** Rx Inversion Sweep Mode 0: Match Sweep according to the SB-to-NB_Mapping field in the TS1 training sequence. The default setting forces the RX inversion pointers to follow the unique northbound inversion across the port width. It is based on a Modulo 5 of Intel 5000X Chipset MCHMAP bit setting. If e lanes Example; If Intel 5000X Chipset MCHMAP = 0 then Lanes [4:0] are used as the reference for checking Lanes[13:10], [9:5], and [4:0]. If Intel 5000X Chipset MCHMAP = 1 then Lanes [9:5] are used as the reference for checking Lanes[13:10], [9:5], and [4:0]. For Intel 5000X Chipset MCH lane [13] does not exist but it does participate in rotate-left-shift operations. 1: Enable full inversion sweep across the entire port. When enabled the RX inversion pointers become a single entity. Lanes [13:10] rotate left-shift completely across the width of the port. Even though Lane[13] is a DFT lane it will be “shifted through” to make the logic design easier. 0->1->2->3->4->5->6->7->8->9->10->11->12->13->0.
24	RW	1	**RXAUTOINVSWPEN:** Auto-inversion sweep enable This bit enable the inversion shift register to continuously rotate the pattern in the FIBRXSHFT register. This register enables the inversion pattern to the lane at the bit position indicated by a logic 1. 0: Disable Auto-inversion 1: Enable Auto-inversion
23	RW	0	**Intel 5000X Chipset MCHMAP:** Southbound to northbound mapping for loopback testing This bit indicates which set of lanes are replicated onto the northbound lanes. 0: Lower SB lanes 1: Upper SB lanes
22	RW	0	**CMMSTR:** Compliance Measurement Mode This bit forces the component into link reset then transmits the default Intel IBIST pattern set of a fixed binary “1100” pattern continuously (depending on implementation) on all Tx lanes until this bit is cleared. If the Intel IBIST engine is used for CMM then the standard initialization sequence is follow with TS0, TS1 training set prior to entry into Intel IBIST. 0: Disable CMM 1: Enable CMM. This feature requires the Intel IBIST start bit to be set before the mode is enabled.
21:12	RWST	000h	**ERRCNT:** Error Counter [9:0] Total number of errors encountered in this port. Errors are accumulated per lane. If several errors occurred in one phit time then a binary encoded value of the number of errors is added to the error count.
11:8	ROST	00h	**ERRLNNUM:** Error Lane Number [3:0] This points to the first lane that encountered an error. If more than one lane reports an error in a cycle, the most significant lane number that reported the error will be logged.
7:6	RWCST	0	**ERRSTAT:** Port Error Status [1:0] When Intel IBIST is started, status goes to 01 until first start delimiter is received and then goes to 00 until the end or to10/11 as appropriate. 00: No error. 01: Did not receive first start delimiter. 10: Transmission error (first error). 11: Reserved.

Device: 22
Function: 0
Offset: 280h, 180h

Bit	Attr	Default	Description
5	RW	1	**AUTOINVSWPEN:** Auto-inversion sweep enable This bit enable the inversion shift register to continuously rotate the pattern in the FIBTXSHFT and FIBRXSHFT registers. These registers enable the inversion pattern to the lane at the bit position indicated by a logic 1. 0: Disable Auto-inversion 1: Enable Auto-inversion
4	RW	1	**STOPONERR:** Stop Intel IBIST on Error 0: Do not stop on error, only update error counter 1: Stop on error
3	RW	0	**LOOPCON:** Loop forever 0: No looping 1: Loop forever
2	RWCST	0	**IBDONE:** IBIST done flag 0: Not done 1: Done
1	RWST	1	**MSTRMD:** Master Mode Enable When this bit is set the next TS1 training set that has the loopback bit set will cause the transmitter to operate as a master. Even though the Intel IBIST is in the loopback state it is not in loopback. 0: Disable Master mode. This component will not enter into master when a TS1 training set with loopback bit set. 1: Enable Master Mode on the next TS1 training with loopback bit set.
0	RWST	0	**IBSTR:** IBIST Start When set, it enables receiver logic to look for start delimiters during TS1 training set. If the MSTRMD bit is set, the start bit enables the transmit state machine to start transmitting patterns during the TS1 training set. The receiver is enable in both cases. For master-slave mode, the pattern will be looped back as defined in the FB-DIMM spec. In master-master mode, the Intel IBIST controller will originate patterns and also check the incoming pattern for errors. 0: Stop IBIST transmitter 1: Start IBIST transmitter

3.9.26.2 FBD[1:0]IBPORTCTL: FB-DIMM Intel IBIST Port Control Register

This register contains bits to control the operation of the Intel IBIST DFT feature.

Device: 21
Function: 0
Offset: 280h, 180h

Bit	Attr	Default	Description
31:26	RV	0h	**Reserved**
25	RWST	0	**RXINVSWPMD:** Rx Inversion Sweep Mode 0: Match Sweep according to the SB-to-NB_Mapping field in the TS1 training sequence. The default setting forces the RX inversion pointers to follow the unique northbound inversion across the port width. It is based on a Modulo 5 of Intel 5000X Chipset MCHMAP bit setting. If e lanes Example; If Intel 5000X Chipset MCHMAP = 0 then Lanes [4:0] are used as the reference for checking Lanes[13:10], [9:5], and [4:0]. If Intel 5000X Chipset MCHMAP = 1 then Lanes [9:5] are used as the reference for checking Lanes[13:10], [9:5], and [4:0]. For Intel 5000X Chipset MCH lane [13] does not exist but it does participate in rotate-left-shift operations. 1: Enable full inversion sweep across the entire port. When enabled the RX inversion pointers become a single entity. Lanes [13:10] rotate left-shift completely across the width of the port. Even though Lane[13] is a DFT lane it will be “shifted through” to make the logic design easier. 0->1->2->3->4->5->6->7->8->9->10->11->12->13->0.
24	RW	1	**RXAUTOINVSWPEN:** Auto-inversion sweep enable This bit enable the inversion shift register to continuously rotate the pattern in the FIBRXSHFT register. 0: Disable Auto-inversion 1: Enable Auto-inversion
23	RW	0	**Intel 5000X Chipset MCHMAP:** Southbound to northbound mapping for loopback testing This bit indicates which set of lanes are replicated onto the northbound lanes. 0: Lower SB lanes 1: Upper SB lanes
22	RW	0	**CMMSTR:** Compliance Measurement Mode This bit forces the component into link reset then transmits the contents of the default Intel IBIST pattern set continuously (depending on implementation) on all Tx lanes until this bit is cleared and the IBSTR bit is cleared. If the Intel IBIST engine is used for CMM then the standard initialization sequence is follow with TS0, TS1 training set prior to entry into Intel IBIST. 0: Disable CMM 1: Enable CMM. This feature requires the Intel IBIST start bit to be set before the mode is enabled.
21:12	RWST	000h	**ERRCNT:** Error Counter [9:0] Total number of errors encountered in this port. Errors are accumulated per lane. If several errors occurred in one phit time then a binary encoded value of the number of errors is added to the error count.
11:8	ROST	00h	**ERRLNNUM:** Error Lane Number [3:0] This points to the first lane that encountered an error. If more than one lane reports an error in a cycle, the most significant lane number that reported the error will be logged.
7:6	RWCST	0	**ERRSTAT:** Port Error Status [1:0] When Intel IBIST is started, status goes to 01 until first start delimiter is received and then goes to 00 until the end or to10/11 as appropriate. 00: No error. 01: Did not receive first start delimiter. 10: Transmission error (first error). 11: Reserved.

Device: 21
Function: 0
Offset: 280h, 180h

Bit	Attr	Default	Description
5	RW	1	**TXAUTOINVSWPEN:** Auto-inversion sweep enable This bit enable the inversion shift register to continuously rotate the pattern in the FIBTXSHFT register. 0: Disable Auto-inversion 1: Enable Auto-inversion
4	RW	1	**STOPONERR: Stop IBIST on Error** 0: Do not stop on error, only update error counter 1: Stop on error
3	RW	0	**LOOPCON: Loop continuously** Enable Intel IBIST operations to loop continuously. The Intel IBIST pattern generator executes the each pattern loop for the counts specified in the bit fields but the overall loop runs continuously. This bit should be protected (gated) by the component's security mechanisms. 0: No continuous operation 1: Loop continuously
2	RWCST	0	**IBDONE:** IBIST done flag 0: Not done 1: Done
1	RWST	1	**MSTRMD: Master Mode Enable** When this bit is set the next TS1 training set that has the loopback bit set will cause the transmitter to operate as a master. Even though the Intel IBIST is in the loopback state it is not in loopback. 0: Disable Master mode. This component will not enter into master when a TS1 training set with loopback bit set. 1: Enable Master Mode on the next TS1 training with loopback bit set.
0	RWST	0	**IBSTR: IBIST Start** When set, it enables receiver logic to look for start delimiters during TS1 training set. If the MSTRMD bit is set, the start bit enables the transmit state machine to start transmitting patterns during the TS1 training set. The receiver is enable in both cases. For master-slave mode, the pattern will be looped back as defined in the FB-DIMM spec. In master-master mode, the Intel IBIST controller will originate patterns and also check the incoming pattern for errors. 0: Stop IBIST transmitter 1: Start IBIST transmitter

3.9.26.3 FBD[3:2]IBTXPGCTL: FB-DIMM Intel IBIST Pattern Generator Control Register

This register contains bits to control the operation of the Intel IBIST pattern generator.

Device: 22
Function: 0
Offset: 284h, 184h

Bit	Attr	Default	Description
31:26	RWST	04h	**OVRLOPCNT:** Overall Loop Count[5:0] 0h: Send no Intel IBIST data in payload 1h-3Fh: The number of times to loop through all the patterns
25:21	RWST	0h	**CNSTGENCNT:** Constant Generator Loop Counter[4:0] 00h: Disable constant generator output 01h: 1Fh The number of times the Modulo-N counter should be repeated before going to the next pattern type. Each buffer transfer is composed of two frames (loop counts of 24-bits each).
20	RWST	0	**CNSTGENSET:** Constant Generator Setting 0: Generate 0 1: Generate 1
19:13	RWST	19h	**MODLOPCNT:** Modulo-N Loop Counter [7:0] Each count represents 24-bits of the pattern specified by the MODPERIOD bit field. 00h: Disable Pattern Output 01h: 7Fh The number of times the Pattern Buffer should loop before going to the next
12:10	RWST	1h	**MODPERIOD:** Period of the Modulo-N counter 001: L/2 0101_0101_0101_0101_0101_0101 010: L/4 0011_0011_0011_0011_0011_0011 011: L/6 0001_1100_0111_0001_1100_0111 100: L/8 0000_1111_0000_1111_0000_1111 110: L/12 0000_0000_0000_1111_1111_1111
9:3	RWST	19h	**PATTLOPCNT:** Pattern Buffer Loop Counter[6:0] 00h: Disable Pattern Output 01h-3Fh: The number of times the Pattern Buffer should be repeated before going to the next pattern type. Each buffer transfer is composed of two frames (loop counts of 24-bits each).
2:0	RWST	000	**PTGENORD:** Pattern Generation Order 000: Pattern Store + Modulo N Cntr + Constant Generator 001: Pattern Store + Constant Generator + Modulo N Cntr 010: Modulo N Cntr + Pattern Store + Constant Generator 011: Modulo N Cntr + Constant Generator + Pattern Store 100: Constant Generator + Pattern Store + Modulo N Cntr 101: Constant Generator + Modulo N Cntr + Pattern Store 110: Reserved 111: Reserved

3.9.26.4 FBD[1:0]IBTXPGCTL: FB-DIMM Intel IBIST Pattern Generator Control Register

This register contains bits to control the operation of the Intel IBIST pattern generator.

Device: 21
Function: 0
Offset: 284h, 184h

Bit	Attr	Default	Description
31:26	RWST	04h	**OVRLOPCNT:** Overall Loop Count[5:0] 0h: Send no IBIST data in payload 1h-3Fh: The number of times to loop through all the patterns
25:21	RWST	0h	**CNSTGENCNT:** Constant Generator Loop Counter[4:0] 00h: Disable constant generator output 01h: 1Fh The number of times the Modulo-N counter should be repeated before going to the next pattern type. Each buffer transfer is composed of two frames (loop counts of 24-bits each).
20	RWST	0	**CNSTGENSET:** Constant Generator Setting 0: Generate 0 1: Generate 1
19:13	RWST	19h	**MODLOPCNT:** Modulo-N Loop Counter Each count represents 24-bits of the pattern specified by the MODPERIOD bit field. 00h: Disable Pattern Output 01h: 7Fh The number of times the Pattern Buffer should loop before going to the next
12:10	RWST	1h	**MODPERIOD:** Period of the Modulo-N counter 001: L/2 0101_0101_0101_0101_0101_0101 010: L/4 0011_0011_0011_0011_0011_0011 011: L/6 0001_1100_0111_0001_1100_0111 100: L/8 0000_1111_0000_1111_0000_1111 110: L/12 0000_0000_0000_1111_1111_1111
9:3	RWST	19h	**PATTLOPCNT:** Pattern Buffer Loop Counter[6:0] 00h: Disable Pattern Output 01h-3Fh: The number of times the Pattern Buffer should be repeated before going to the next pattern type. Each buffer transfer is composed of two frames (loop counts of 24-bits each).
2:0	RWST	000	**PTGENORD:** Pattern Generation Order 000: Pattern Store + Modulo N Cntr + Constant Generator 001: Pattern Store + Constant Generator + Modulo N Cntr 010: Modulo N Cntr + Pattern Store + Constant Generator 011: Modulo N Cntr + Constant Generator + Pattern Store 100: Constant Generator + Pattern Store + Modulo N Cntr 101: Constant Generator + Modulo N Cntr + Pattern Store 110: Reserved 111: Reserved

3.9.26.5 FBD[3:2]IBPATBUF: FB-DIMM Intel IBIST Pattern Buffer Register

This register contains the pattern bits used in Intel IBIST operations.

Device: 22
Function: 0
Offset: 288h, 188h

Bit	Attr	Default	Description
31:24	RV	0	*Reserved*
23:0	RWST	02CCFDh	**IBPATBUF:** IBIST Pattern Buffer Pattern buffer storing the default and the user programmable pattern. Default: 0000_0010_1100_1100_1111_1101

3.9.26.6 FBD[1:0]IBPATBUF: FB-DIMM Intel IBIST Pattern Buffer Register

This register contains the pattern bits used in Intel IBIST operations.

Device: 21
Function: 0
Offset: 288h, 188h

Bit	Attr	Default	Description
31:24	RV	0	*Reserved*
23:0	RWST	02CCFDh	**IBPATBUF:** IBIST Pattern Buffer Pattern buffer storing the default and the user programmable pattern. Default: 0000_0010_1100_1100_1111_1101

3.9.26.7 FBD[3:2]IBTXMSK: Intel IBIST Transmitter Mask

This register determines which lanes are enabled for Intel IBIST operations. These bits also control the power saving features of each lane. If a particular lane is masked off, the power to that lane is reduced as much as possible.

Device: 22
Function: 0
Offset: 28Ch, 18Ch

Bit	Attr	Default	Description
31:14	RV	0h	Reserved
13:10	RWST	0h	**txmaskhvm:** Transmit Mask extra DFT pins for HVM symmetry Selects which lanes to enable for testing. A lane that is *not* selected remains in electrical idle.
9:0	RWST	3FFh	**txmask:** Transmit Mask Selects which lanes to enable for testing. A lane that is *not* selected remains in electrical idle.

3.9.26.8 FBD[1:0]IBTXMSK: Intel IBIST Transmitter Mask

This register determines which lanes are enabled for Intel IBIST operations. These bits also control the power saving features of each lane. If a particular lane is masked off, the power to that lane is reduced as much as possible.

Device: 21
Function: 0
Offset: 28Ch, 18Ch

Bit	Attr	Default	Description
31:14	RV	0h	*Reserved*

Device: 21 Function: 0 Offset: 28Ch, 18Ch			
Bit	**Attr**	**Default**	**Description**
13:10	RWST	0h	**txmaskhvm:** Transmit Mask extra DFT pins for HVM symmetry Selects which lanes to enable for testing. A lane that is *not* selected remains in electrical idle.
9:0	RWST	3FFh	**txmask:** Transmit Mask Selects which lanes to enable for testing. A lane that is *not* selected remains in electrical idle.

3.9.26.9 FBD[3:2]IBRXMSK: Intel IBIST Receiver Mask

This register determines which lanes are enabled for Intel IBIST operations. These bits also control the power saving features of each lane. If a particular lane is masked off, the power to that lane is reduced as much as possible.

Device: 22 Function: 0 Offset: 290h, 190h			
Bit	**Attr**	**Default**	**Description**
31:14	RV	0h	*Reserved*
13:0	RWST	1FFFh	**rxmask:** Receive Mask Selects which lanes to enable for testing. An Rx lane that is not selected is not included in Rx channel training does not contribute to the accumulation of error counts.

3.9.26.10 FBD[1:0]IBRXMSK: Intel IBIST Receiver Mask

This register determines which lanes are enabled for Intel IBIST operations. These bits also control the power saving features of each lane. If a particular lane is masked off, the power to that lane is reduced as much as possible.

Device: 21 Function: 0 Offset: 290h, 190h			
Bit	**Attr**	**Default**	**Description**
31:14	RV	0h	Reserved
13:0	RWST	1FFFh	**rxmask:** Receive Mask Selects which lanes to enable for testing. An Rx lane that is not selected is not included in Rx channel training does not contribute to the accumulation of error counts.

3.9.26.11 FBD[3:2]IBTXSHFT: Intel IBIST Transmit Shift Inversion Register

This register indicates which channel is currently inverting the pattern to create cross talk conditions on the port.

Device: 22 Function: 0 Offset: 294h, 194h			
Bit	**Attr**	**Default**	**Description**
31:14	RV	0h	*Reserved*

Device: 22 Function: 0 Offset: 294h, 194h			
Bit	**Attr**	**Default**	**Description**
13:10	RWST	0h	**txinvshfthvm:** Transmit Inversion shift register extra DFT pins for HVM symmetry The pattern loaded in this register indicates which lanes are used for inversion. A logic 1 enables the lane connected to a particular bit position to invert the pattern that is being transmitted. Because this is a shift register the initial value will be left-shifted at the end of the loop count during Intel IBIST operations.
9:0	RWST	001h	**txinvshft:** Transmitter Inversion Shift Register The pattern loaded in this register indicates which lanes are used for inversion. A logic 1 enables the lane connected to a particular bit position to invert the pattern that is being transmitted. Because this is a shift register the initial value will be left-shifted at the end of the loop count during Intel IBIST operations.

3.9.26.12 FBD[1:0]IBTXSHFT: Intel IBIST Transmit Shift Inversion Register

This register indicates which channel is currently inverting the pattern to create cross talk conditions on the port.

Device: 21 Function: 0 Offset: 294h, 194h			
Bit	**Attr**	**Default**	**Description**
31:14	RV	0h	Reserved
13:10	RWST	0h	**txinvshfthvm:** Transmit Inversion shift register extra DFT pins for HVM symmetry The pattern loaded in this register indicates which lanes are used for inversion. A logic 1 enables the lane connected to a particular bit position to invert the pattern that is being transmitted. Because this is a shift register the initial value will be left-shifted at the end of the loop count during Intel IBIST operations.
9:0	RWST	001h	**txinvshft:** Transmitter Inversion Shift Register The pattern loaded in this register indicates which lanes are used for inversion. A logic 1 enables the lane connected to a particular bit position to invert the pattern that is being transmitted. Because this is a shift register the initial value will be left-shifted at the end of the loop count during Intel IBIST operations.

3.9.26.13 FBD[3:2]IBRXSHFT: Intel IBIST Receive Shift Inversion Register

This register indicates which channel is currently inverting the pattern to create cross talk conditions on the port.

Device: 22 Function: 0 Offset: 298h, 198h			
Bit	**Attr**	**Default**	**Description**
31:14	RV	0h	*Reserved*
13	RWST	0	**rxinvshfthi:** Receiver Inversion Shift Register for DFT The pattern loaded in this bit field indicates which lanes are used for inversion. A logic 1 enables the lane connected to a particular bit position to invert the pattern that is being transmitted. This bit location will experience rotate-left-shift operation with bits[12:0].

Device: 22 Function: 0 Offset: 298h, 198h			
Bit	**Attr**	**Default**	**Description**
12:0	RWST	0001h	**rxinvshft:** Receiver Inversion Shift Register The pattern loaded in this register indicates which lanes are used for inversion. A logic 1 enables the lane connected to a particular bit position to invert the pattern that is being transmitted. This register acts as a rotate-left shift register regardless of the setting of **RXINVSWPMD** bit. The Modulo-5 value is used to compare each sub-section of the northbound lanes for error checking.

3.9.26.14 FBD[1:0]IBRXSHFT: Intel IBIST Receive Shift Inversion Register

This register indicates which channel is currently inverting the pattern to create cross talk conditions on the port.

Device: 21 Function: 0 Offset: 298h, 198h			
Bit	**Attr**	**Default**	**Description**
31:14	RV	0h	*Reserved*
13	RWST	0	**rxinvshfthi:** Receiver Inversion Shift Register for DFT The pattern loaded in this bit field indicates which lanes are used for inversion. A logic 1 enables the lane connected to a particular bit position to invert the pattern that is being transmitted. This bit location will experience rotate-left-shift operation with bits[12:0].
12:0	RWST	0001h	**rxinvshft:** Receiver Inversion Shift Register The pattern loaded in this register indicates which lanes are used for inversion. A logic 1 enables the lane connected to a particular bit position to invert the pattern that is being transmitted. This register acts as a rotate-left shift register regardless of the setting of **RXINVSWPMD** bit. The Modulo-5 value is used to compare each sub-section of the northbound lanes for error checking.

3.9.26.15 FBD[3:2]LNERR: Intel IBIST Receive Lane Error Register

This register enables Intel IBIST operations for individual lanes.

Device: 22 Function: 0 Offset: 29Ch, 19Ch			
Bit	**Attr**	**Default**	**Description**
31:14	RV	0h	*Reserved*
13	ROST	0	**rxerrstat:** Receive error lane status for DFT. This register records the error from lane 13 of this port.
12:0	ROST	0	**rxerrstat:** Receive error lane status. This register records the errors from all lanes of this port.

3.9.26.16 FBD[1:0]LNERR: Intel IBIST Receive Lane Error Register

This register enables Intel IBIST operations for individual lanes.

Device: 21 Function: 0 Offset: 29Ch, 19Ch			
Bit	**Attr**	**Default**	**Description**
31:14	RV	0h	*Reserved*

Device: 21 Function: 0 Offset: 29Ch, 19Ch			
Bit	**Attr**	**Default**	**Description**
13	ROST	0	**rxerrstat:** Receive error lane status for DFT. This register records the error from lane 13 of this port.
12:0	ROST	0	**rxerrstat:** Receive error lane status. This register records the errors from all lanes of this port.

3.9.26.17 FBD[3:2]IBRXPGCTL: FB-DIMM Intel IBIST Rx Pattern Generator Control Register

This register contains bits to control the operation of the Rx pattern generator.

Device: 22 Function: 0 Offset: 2A0h, 1A0h			
Bit	**Attr**	**Default**	**Description**
31:26	RWST	04h	**OVRLOPCNT:** Overall Loop Count[5:0] 0h: Send no Intel IBIST data in payload 1h-3Fh: The number of times to loop through all the patterns
25:21	RWST	0h	**CNSTGENCNT:** Constant Generator Loop Counter[4:0] 00h: Disable constant generator output 01h: 1Fh The number of times the Modulo-N counter should be repeated before going to the next pattern type. Each buffer transfer is composed of two frames (loop counts of 24-bits each).
20	RWST	0	**CNSTGENSET:** Constant Generator Setting 0: Generate 0 1: Generate 1
19:13	RWST	19h	**MODLOPCNT:** Modulo-N Loop Counter [7:0] Each count represents 24-bits of the pattern specified by the MODPERIOD bit field. 00h: Disable Pattern Output 01h: 7Fh The number of times the Pattern Buffer should loop before going to the next
12:10	RWST	1h	**MODPERIOD:** Period of the Modulo-N counter 001: L/2 0101_0101_0101_0101_0101_0101 010: L/4 0011_0011_0011_0011_0011_0011 011: L/6 0001_1100_0111_0001_1100_0111 100: L/8 0000_1111_0000_1111_0000_1111 110: L/12 0000_0000_0000_1111_1111_1111
9:3	RWST	19h	**PATTLOPCNT:** Pattern Buffer Loop Counter[6:0] 00h: Disable Pattern Output 01h-3Fh: The number of times the Pattern Buffer should be repeated before going to the next pattern type. Each buffer transfer is composed of two frames (loop counts of 24-bits each).
2:0	RWST	000	**PTGENORD:** Pattern Generation Order 000: Pattern Store + Modulo N Cntr + Constant Generator 001: Pattern Store + Constant Generator + Modulo N Cntr 010: Modulo N Cntr + Pattern Store + Constant Generator 011: Modulo N Cntr + Constant Generator + Pattern Store 100: Constant Generator + Pattern Store + Modulo N Cntr 101: Constant Generator + Modulo N Cntr + Pattern Store 110: Reserved 111: Reserved

3.9.26.18 FBD[1:0]IBRXPGCTL: FB-DIMM Intel IBIST Rx Pattern Generator Control Register

This register contains bits to control the operation of the RX pattern generator.

Device: 21
Function: 0
Offset: 2A0h, 1A0h

Bit	Attr	Default	Description
31:26	RWST	04h	**OVRLOPCNT:** Overall Loop Count[5:0] 0h: Send no IBIST data in payload 1h-3Fh: The number of times to loop through all the patterns
25:21	RWST	0h	**CNSTGENCNT:** Constant Generator Loop Counter[4:0] 00h: Disable constant generator output 01h: 1Fh The number of times the Modulo-N counter should be repeated before going to the next pattern type. Each buffer transfer is composed of two frames (loop counts of 24-bits each).
20	RWST	0	**CNSTGENSET:** Constant Generator Setting 0: Generate 0 1: Generate 1
19:13	RWST	19h	**MODLOPCNT:** Modulo-N Loop Counter Each count represents 24-bits of the pattern specified by the MODPERIOD bit field. 00h: Disable Pattern Output 01h: 7Fh The number of times the Pattern Buffer should loop before going to the next
12:10	RWST	1h	**MODPERIOD:** Period of the Modulo-N counter 001: L/2 0101_0101_0101_0101_0101_0101 010: L/4 0011_0011_0011_0011_0011_0011 011: L/6 0001_1100_0111_0001_1100_0111 100: L/8 0000_1111_0000_1111_0000_1111 110: L/12 0000_0000_0000_1111_1111_1111
9:3	RWST	19h	**PATTLOPCNT:** Pattern Buffer Loop Counter[6:0] 00h: Disable Pattern Output 01h-3Fh: The number of times the Pattern Buffer should be repeated before going to the next pattern type. Each buffer transfer is composed of two frames (loop counts of 24-bits each).
2:0	RWST	000	**PTGENORD:** Pattern Generation Order 000: Pattern Store + Modulo N Cntr + Constant Generator 001: Pattern Store + Constant Generator + Modulo N Cntr 010: Modulo N Cntr + Pattern Store + Constant Generator 011: Modulo N Cntr + Constant Generator + Pattern Store 100: Constant Generator + Pattern Store + Modulo N Cntr 101: Constant Generator + Modulo N Cntr + Pattern Store 110: Reserved 111: Reserved

3.9.26.19 FBD[3:2]IBPATBUF2: FB-DIMM Intel IBIST Pattern Buffer 2 Register

This register contains the pattern bits used in Intel IBIST operations.

Device: 22
Function: 0
Offset: 2A4h, 1A4h

Bit	Attr	Default	Description
31:24	RV	0	*Reserved*
23:0	RWST	02CCFDh	**IBPATBUF:** IBIST Pattern Buffer Pattern buffer storing the default and the user programmable pattern. Default: 0000_0010_1100_1100_1111_1101

3.9.26.20 FBD[1:0]IBPATBUF2: FB-DIMM IBIST Pattern Buffer 2 Register

This register contains the pattern bits used in IBIST operations.

Device: 21
Function: 0
Offset: 2A4h, 1A4h

Bit	Attr	Default	Description
31:24	RV	0	*Reserved*
23:0	RWST	02CCFDh	**IBPATBUF:** IBIST Pattern Buffer Pattern buffer storing the default and the user programmable pattern. Default: 0000_0010_1100_1100_1111_1101

3.9.26.21 FBD[3:2]IBTXPAT2EN: Intel IBIST TX Pattern Buffer 2 Enable

This register enables which channels are inverted when Intel IBIST operations are activated.

Device: 22
Function: 0
Offset: 2A8h, 1A8h

Bit	Attr	Default	Description
31:14	RV	0h	*Reserved*
13:10	RWST	Fh	**txpatt2hvmen:** receiver Pattern Buffer 2 Enable for the HVM lanes Selects which channels to enable the second pattern buffer.
9:0	RWST	3FFh	**txpatt2en:** receiver Pattern Buffer 2 Enable Selects which channels to enable the second pattern buffer.

3.9.26.22 FBD[1:0]IBTXPAT2EN: IBIST TX Pattern Buffer 2 Enable

This register enables which channels are inverted when IBIST operations are activated.

Device: 21
Function: 0
Offset: 2A8h, 1A8h

Bit	Attr	Default	Description
31:14	RV	0h	*Reserved*
13:10	RWST	Fh	**txpatt2hvmen:** receiver Pattern Buffer 2 Enable for the HVM lanes Selects which channels to enable the second pattern buffer.
9:0	RWST	3FFh	**txpatt2en:** receiver Pattern Buffer 2 Enable Selects which channels to enable the second pattern buffer.

3.9.26.23 FBD[3:2]IBRXPAT2EN: Intel IBIST RX Pattern Buffer 2 Enable

This register enables inversion pattern testing on individual lanes.

Device: 22
Function: 0
Offset: 2ACh, 1ACh

Bit	Attr	Default	Description
31:14	RV	0h	*Reserved*
13:0	RWST	3FFFh	**rxpatt2en:** Receiver Pattern Buffer 2 Enable Selects which channels to enable the second pattern buffer.

3.9.26.24 FBD[1:0]IBRXPAT2EN: Intel IBIST RX Pattern Buffer 2 Enable

This register enables inversion pattern testing on individual lanes.

Device: 21
Function: 0
Offset: 2ACh, 1ACh

Bit	Attr	Default	Description
31:14	RV	0h	*Reserved*
13:0	RWST	3FFFh	**rxpatt2en:** Receiver Pattern Buffer 2 Enable Selects which channels to enable the second pattern buffer.

3.9.27 FB-DIMM and Memory Control Debug Registers

These registers are used to inject bit errors into memory arrays for memory error detection testing. These registers allow the user to corrupt individual memory bits or set individual FB-DIMM lanes to a stuck at “1” condition.

3.9.27.1 FBD[1:0]PLLCTRL: FB-DIMM PLL Control Register

This register defines the ability to control the PLL (on/off) for FB-DIMM branches 0 and 1. This is an additional feature besides JTAG providing BIOS an extra degree of control on the FB-DIMM branches.

Device: 22, 21
Function: 0
Offset: 1C0h

Bit	Attr	Default	Description
7:1	RV	0	**Reserved**
0	RWST	0h	DISPLL: Disable PLL for the FDB Branch 0: Normal operation i.e PLL is enabled and FB-DIMM memory can be accessed. 1: Disable PLL for the FB-DIMM Branch. This implies that the selected FBD branch is disabled and no operations to memory are possible

3.9.27.2 FBD[3:2]EINJCTL: FB-DIMM Outbound Error Injection Control Register

This register contains the error injection mask register to determine which bits get corrupted for error detection testing.

Device: 22 Function: 0 Offset: 20Ch, 10Ch			
Bit	**Attr**	**Default**	**Description**
31:24	RV	0	**Reserved**
23:22	RW	0	**EINJEN: Error injection enable** 00: No error injection 01: Flit transfer error injection 10: Stuck at 1 error injection 11: Reserved
21	RW	0	**EINJSEL: Error Injection Select** 0: Select DINJ0 response function. 1: Select DINJ1 response function.
20:18	RW	0	**DDRCMD: Select the DDR command types** For selecting which transaction to corrupt. FB-DIMM Command encoding bit field [20:18] 000: FBD command op code. Use FBDCHCMD bits [17:14] below. 001: Precharge all 010: Read 011: Write 100-111: reserved
17:14	RW	0	**FBDCHCMD: Select channel command to corrupt OP[3:0]**
13:10	RW	0h	**FTRANS: Flit transfer number to corrupt** This field indicates which flit contains the corrupted bit. A maximum depth of 12 is where the corruption could occur. 0-Bh: Flit number C-Fh: Reserved
9:0	RW	0h	**EBITPOS: Error injection bit position** Bit field can contain more than one bit for multi-bit error injection.

3.9.27.3 FBD[1:0]EINJCTL: FB-DIMM Outbound Error Injection Control Register

This register contains the error injection mask register to determine which bits get corrupted for error detection testing.

Device: 21 Function: 0 Offset: 20Ch, 10Ch			
Bit	**Attr**	**Default**	**Description**
31:24	RV	0	**Reserved**
23:22	RW	0	**EINJEN: Error injection enable** 00: No error injection 01: Flit transfer error injection 10: Stuck at 1 error injection 11: Reserved
21	RW	0	**EINJSEL: Error Injection Select** 0: Select DINJ0 response function. 1: Select DINJ1 response function.

Device: 21 Function: 0 Offset: 20Ch, 10Ch			
Bit	**Attr**	**Default**	**Description**
20:18	RW	0	**DDRCMD: Select the DDR command types** For selecting which transaction to corrupt. FB-DIMM Command encoding bit field [20:18] 000: FBD command op code. Use FBDCHCMD bits [17:14] below. 001: Precharge all 010: Read 011: Write 100-111: Reserved
17:14	RW	0	**FBDCHCMD: Select channel command to corrupt OP[3:0]**
13:10	RW	0h	**FTRANS:Flit transfer number to corrupt** This field indicates which flit contains the corrupted bit. A maximum depth of 12 is where the corruption could occur. 0-Bh: Flit number C-Fh: Reserved
9:0	RW	0h	**EBITPOS: Error injection bit position** Bit field can contain more than one bit for multi-bit error injection.

3.9.27.4 FBD[3:2]STUCKL: Stuck “ON” FB-DIMM Lanes

This register selects FB-DIMM Lanes to be stuck at “1” when error injection is enabled and Stuck on Lanes is selected.

Device: 22 Function: 0 Offset: 211h, 111h			
Bit	**Attr**	**Default**	**Description**
7:4	RV	0	**Reserved**
3:0	RW	Fh	**SBSTUCK:** SB Lane to be driven to “1” 0h: lane 0 9h: lane 9 and >9h: NOP

3.9.27.5 FBD[1:0]STUCKL: Stuck “ON” FB-DIMM Lanes

This register selects FB-DIMM Lanes to be stuck at “1” when error injection is enabled and Stuck on Lanes is selected.

Device: 21 Function: 0 Offset: 211h, 111h			
Bit	**Attr**	**Default**	**Description**
7:4	RV	0	**Reserved**
3:0	RW	Fh	**SBSTUCK:** SB Lane to be driven to “1” 0h: lane 0 9h: lane 9 and >9h: NOP

3.9.27.6 MEM[1:0]EINJMSK0: Branch Memory Error Injection Mask0 Register

This register contains the error injection mask register to determine which bits get corrupted for error detection testing.

Notes for register usage:

1. In mirror mode both sets of registers MEM[1:0]EINJMSK0 and MEM[1:0]EINJMSK1 must be programmed with the same values.
2. When the First SRAM location and the second SRAM location are the same vale then only the first causes an ECC mask corruption. The second will have no effect.
3. After the error injection features is activated, the masks for both First and Second corruption are consistently active. In other words, there isn't an enable bit for each of them. The bit field for zero is a valid selection. For these features to not have an effect the user is required to make the mask bit field as zero.

Device: 16
Function: 1
Offset: 108h, 100h

Bit	Attr	Default	Description
31:30	RV	00	**Reserved**
22:29	RW	01	**HLINESEL: Half cache line section** **00: Reserved** **01: Selects lower half cache line for error injection on transfer 0 and 1 using First and/or Second device pointers.** **10: Selects upper half cache line for error injection on transfer 2 and 3 using First and/or Second device pointers.** 11:Select both upper and lower cache lines to inject errors based on the First and Second device pointers. The same masks are applied to both halves.
27	RW	0	**EINJEN: Error injection enable** 0: Disable error injection 1: Enable error injection
26	RW	0	**EINJFUNCTSEL: Error Injection Function Select** 0: Select DINJ0 response function. 1: Select DINJ1 response function.
25:10	RW	0h	**XORMSK: XOR MaskBit for first x8 location** [25:18]: 2nd phase [17:10]: 1st phase
9:5	RW	0h	**SEC2RAM: Second x8 SRAM Location (0 to 17)**.
4:0	RW	0h	**FIR2RAM: First x8 SRAM Location (0 to 17)**.

3.9.27.7 MEM[1:0]EINJMSK1: Branch Memory Error Injection Mask1 Register

This register contains the error injection mask register to determine which bits get corrupted for error detection testing.

Device: 16
Function: 1
Offset: 106h, 104h

Bit	Attr	Default	Description
15:0	RW	0h	**XORMSK: XOR MaskBit for second x8 location** [15:8]: 2nd phase [7:0]: 1st phase

3.9.28 Serial Presence Detect Registers

There are two sets of the following registers, one set for each FB-DIMM branch. They each appear in function 0 of different devices as shown in Table 3-3.

3.9.28.1 SPD[1:0][1:0] - Serial Presence Detect Status Register

This register provides the interface to the SPD bus (SCL and SDA signals) that is used to access the Serial Presence Detect EEPROM that defines the technology, configuration, and speed of the DIMM's controlled by the MCH.

Device: 21, 22
Function: 0
Offset: 76h, 74h

Bit	Attr	Default	Description
15	RO	0	**RDO:** Read Data Valid. This bit is set by the AMB when the Data field of this register receives read data from the SPD EEPROM after successful completion of an SPDR command. It is cleared by the Intel 5000X Chipset MCH when a subsequent SPDR command is issued.
14	RO	0	**WOD:** Write Operation Done. This bit is set by the Intel 5000X Chipset MCH when a SPDW command has been completed on the SPD bus. It is cleared by the Intel 5000X Chipset MCH when a subsequent SPDW command is issued.
13	RO	0	**SBE:** SPD Bus Error. This bit is set by the Intel 5000X Chipset MCH if it initiates an SPD bus transaction that does not complete successfully. It is cleared by the AMB when an SPDR or SPDW command is issued.
12	RO	0	**BUSY:** Busy state. This bit is set by the Intel 5000X Chipset MCH while an SPD command is executing.
11:8	RV	0h	*Reserved.*
7:0	RO	00h	**DATA:** Data. Holds data read from SPDR commands.

3.9.28.2 SPDCMD[1:0][1:0] - Serial Presence Detect Command Register

A write to this register initiates a DIMM EEPROM access through the SPD bus.

Device: 21, 22
Function: 0
Offset: 7Ch, 78h

Bit	Attr	Default	Description
31:28	RWST	1010	**DTI:** Device Type Identifier. This field specifies the device type identifier. Only devices with this device-type will respond to commands. "1010" specifies EEPROM's. "0110" specifies a write-protect operation for an EEPROM. Other identifiers can be specified to target non-EEPROM devices on the SPD bus.
27	RWST	1	**CKOVRD:** Clock Override. '0' = Clock signal is driven low, overriding writing a '1' to CMD. '1' = Clock signal is released high, allowing normal operation of CMD. Toggling this bit can be used to "move" the port out of a "stuck" state.
26:24	RWST	000	**SA:** Slave Address. This field identifies the DIMM EEPROM to be accessed through the SPD register.
23:16	RWST	00h	**BA:** Byte Address. This field identifies the byte address to be accessed through the SPD register.

Device: 21, 22 Function: 0 Offset: 7Ch, 78h			
Bit	**Attr**	**Default**	**Description**
15:8	RWST	00h	**DATA:** Data. Holds data to be written by SPDW commands.
7:1	RV	0h	*Reserved*
0	RWST	0	**CMD:** Command. Writing a '0' to this bit initiates an SPDR command. Writing a '1' to this bit initiates an SPDW command.

3.10 DMA Engine Configuration Registers

Table 3-50. Device 8, Function 0, DMA Engine Configuration Map (Sheet 1 of 2)

DID		VID	00h	FERR_CHAN CMD		FERR_CHANERR	80h
PCISTS		PCICMD	04h	FERR_CHANSTS			84h
CCR		RID	08h				88h
	HDR		0Ch	FERR_DESC_CTRL			8Ch
CB_BAR			10h	FERR_SADDR			90h
			14h				94h
			18h	FERR_DADDR			98h
			1Ch				9Ch
			20h	FERR_TRANSFER_SIZE			A0h
			24h	FERR_NADDR			A4h
			28h				A8h
SID		SVID	2Ch	FERR_CHANCMPFERR_CHANERR			ACh
			30h				B0h
		CAPPTR	34h				B4h
			38h				B8h
	INTP	INTL	3Ch			NERR_CHANERR	BCh
			40h				C0h
			44h				C4h
DMACTRL			48h				C8h
			4Ch				CCh
PMCAP			50h				D0h
PMCSR			54h				D4h
MSICTRL	MSINXPTR	MSICAPID	58h				D8h
MSIAR			5Ch				DCh
MSIDR			60h				E0h
			64h				E4h
			68h				E8h
PEXCAP	PEXNXPTR	PEXCAPID	6Ch				ECh
PEXDEVCAP			70h				F0h

Table 3-50. Device 8, Function 0, DMA Engine Configuration Map (Sheet 2 of 2)

PEXDEVSTS	PEXDEVCTRL	74h		F4h
		78h		F8h
CB_ERR_DOCMD		7Ch		FCh

The DID and VID of the DMA Engine are defined in Table 3-3 of the Configuration Register Chapter. The RID is defined in Section 3.8.1.3 while the HDR register appears in Section 3.8.1.5. The SVID and SID can be found in Section 3.8.1.6.

3.11 CB_BAR MMIO Registers

These MMIO registers are placed in the configuration ring for CPU MMIO accesses but the TOE device will be able to read/write to this space using the fast, bypass inbound access method. Note that this MMIO space is not accessible by MMCFG or CFC/CF8 from the FSB and the mapping to the configuration space is an internal MCH feature to suit the microarchitecture.

Table 3-51. Device 8, Function 1, DMA Engine DMABAR MMIO Registers (General, DMA Channel 0) Mapped through Configuration (Sheet 1 of 2)

INTRCTRL	GENCTRL	XFERCAP	CHANCNT	00h	DMA_COMP0	CHANCTRL0	80h
ATTNSTATUS				04h	CHANSTS0		84h
				08h			88h
PERPORT_OFFSET			CBVER	0Ch	CHAINADDR0		8Ch
CS_STATUS		INTRDELAY		10h			90h
				14h		CHANCMD0	94h
				18h	CHANCMP0		98h
				1Ch			9Ch
				20h	CDAR0		A0h
				24h			A4h
				28h	CHANERR0		A8h
				2Ch	CHANERRMSK0		ACh
				30h			B0h
				34h			B4h
				38h			B8h
				3Ch			BCh
CHAN_SYSERR_MSK0				40h	CHANXFERSIZE0		C0h
CHAN_SYSERR_MSK1				44h	CHANDSCCTL0		C4h
CHAN_SYSERR_MSK2				48h	CHANDSCSRCADDRL0		C8h
CHAN_SYSERR_MSK3				4Ch	CHANDSCSRCADDRH0		CCh
				50h	CHANDSCDSTADDRL0		D0h
				54h	CHANDSCDSTADDRH0		D4h
				58h	CHANNXTDSCADDRL0		D8h
				5Ch	CHANNXTDSCADDRH0		DCh
				60h	CHANSRCADDRPFL0		E0h
				64h	CHDSCSTATE 0	CHANSRCLENREMPF0	CHANSRCADD RPFH0	E4h
				68h	CHANDSTADDRPFL0		E8h

Table 3-51. Device 8, Function 1, DMA Engine DMABAR MMIO Registers (General, DMA Channel 0) Mapped through Configuration (Sheet 2 of 2)

	6Ch	CHANSTATE0	CHANDSTLENREMPF0	CHANDSTADDRPFH0 ECh
	70h	CHANSRCADDRFL0		F0h
	74h	FETCHSTATE0	CHANSRCLENREMF0	CHANSRCADDRFH0 F4h
	78h	CHANDSTADDRFL0		F8h
	7Ch	ERRNOTIFYSTATE0	CHANDSTLENREMF0	CHANDSTADDRFH0 FCh

Table 3-52. Device 8, Function 1: DMABAR MMIO Channel 2 and 3 Registers

DMA_COMP1	CHANCTRL1		100h	DMA_COMP2	CHANCTRL2		180h
CHANSTS1			104h	CHANSTS2			184h
			108h				188h
CHAINADDR1			10Ch	CHAINADDR2			18Ch
			110h				190h
		CHANCMD1	114h			CHANCMD2	194h
CHANCMP1			118h	CHANCMP2			198h
			11Ch				19Ch
CDAR1			120h	CDAR2			1A0h
			124h				1A4h
CHANERR1			128h	CHANERR2			1A8h
CHANERRMSK1			12Ch	CHANERRMSK2			1ACh
			130h				1B0h
			134h				1B4h
			138h				1B8h
			13Ch				1BCh
CHANXFERSIZE1			140h	CHANXFERSIZE2			1C0h
CHANDSCCTL1			144h	CHANDSCCTL2			1C4h
CHANDSCSRCADDRL1			148h	CHANDSCSRCADDRL2			1C8h
CHANDSCSRCADDRH1			14Ch	CHANDSCSRCADDRH2			1CCh
CHANDSCDSTADDRL1			150h	CHANDSCDSTADDRL2			1D0h
CHANDSCDSTADDRH1			154h	CHANDSCDSTADDRH2			1D4h
CHANNXTDSCADDRL1			158h	CHANNXTDSCADDRL2			1D8h
CHANNXTDSCADDRH1			15Ch	CHANNXTDSCADDRH2			1DCh
CHANSRCADDRPFL1			160h	CHANSRCADDRPFL2			1E0h
CHDSCSTATE1	CHANSRCLENREMPF1	CHANSRCADDRPFH1	164h	CHDSCSTATE2	CHANSRCLENREMPF2	CHANSRCADDRPFH2	1E4h
CHANDSTADDRPFL1			168h	CHANDSTADDRPFL2			1E8h
CHANSTATE1	CHANDSTLENREMPF1	CHANDSTADDRPFH1	16Ch	CHANSTATE2	CHANDSTLENREMPF2	CHANDSTADDRPFH2	1ECh
CHANSRCADDRFL1			170h	CHANSRCADDRFL2			1F0h
FETCHSTATE1	CHANSRCLENREMF1	CHANSRCADDRFH1	174h	FETCHSTATE2	CHANSRCLENREMF2	CHANSRCADDRFH2	1F4h
CHANDSTADDRFL1			178h	CHANDSTADDRFL2			1F8h
ERRNOTIFYSTATE1	CHANDSTLENREMF1	CHANDSTADDRFH1	17Ch	ERRNOTIFYSTATE2	CHANDSTLENREMF2	CHANDSTADDRFH2	1FCh

Table 3-53. Device 8, Function 1: DMABAR MMIO Channel 3 Registers

DMA_COMP3	CHANCTRL3		200h	280h
CHANSTS3			204h	284h
			208h	288h
CHAINADDR3			20Ch	28Ch
			210h	290h
		CHANCMD3	214h	294h
CHANCMP3			218h	298h
			21Ch	29Ch
CDAR3			220h	2A0h
			224h	2A4h
CHANERR3			228h	2A8h
CHANERRMSK3			22Ch	2ACh
			230h	2B0h
			234h	2B4h
			238h	2B8h
			23Ch	2BCh
CHANXFERSIZE3			240h	2C0h
CHANDSCCTL3			244h	2C4h
CHANDSCSRCADDRL3			248h	2C8h
CHANDSCSRCADDRH3			24Ch	2CCh
CHANDSCDSTADDRL3			250h	2D0h
CHANDSCDSTADDRH3			254h	2D4h
CHANNXTDSCADDRL3			258h	2D8h
CHANNXTDSCADDRH3			25Ch	2DCh
CHANSRCADDRPFL3			260h	2E0h
CHDSCSTATE3	CHANSRCLENREMPF3	CHANSRCADDRPFH2	264h	2E4h
CHANDSTADDRPFL3			268h	2E8h
CHANSTATE3	CHANDSTLENREMPF3	CHANDSTADDRPFH3	26Ch	2ECh
CHANSRCADDRFL3			270h	2F0h
FETCHSTATE3	CHANSRCLENREMF3	CHANSRCADDRFH3	274h	2F4h
CHANDSTADDRFL3			278h	2F8h
ERRNOTIFYSTATE3	CHANDSTLENREMF3	CHANDSTADDRFH3	27Ch	2FCh

Table 3-54. Device 8, Function 1: Per Port Specific Registers for Ports 2 and 3

STRMPRI2	PPRSETLEN2	NXTPPRSET2		300h	STRMPRI3	PPRSETLEN3	NXTPPRSET3		380h
STRMCAP2	STRMIDFMT2	REQID2		304h	STRMCAP2	STRMIDFMT2	REQID3		384h
WCCTRL2		WCSIZE2	WCNUM2	308h	WCCTRL3		WCSIZE3	WCNUM3	388h
BRIDGE_ID2		WCWINDOW2	WCCAP2	30Ch	BRIDGE_ID3		WCWINDOW3	WCCAP3	38Ch
WCBASE2				310h	WCBASE3				390h
				314h					394h
	PORTPRI2	STRMMAP_OFFSET2		318h		PORTPRI3	STRMMAP_OFFSET3		398h
WC_COMP2		STRM_COMP2		31Ch	WC_COMP3		STRM_COMP3		39Ch
BR_MEM_LIMIT2		BR_MEM_BASE2		320h	BR_MEM_LIMIT3		BR_MEM_BASE3		3A0h
BR_PMEM_LIMIT2		BR_PMEM_BASE2		324h	BR_PMEM_LIMIT3		BR_PMEM_BASE3		3A4h
BR_PBASE_UPPER32_P2				328h	BR_PBASE_UPPER32_P3				3A8h
BR_PLIMIT_UPPER32_P2				32Ch	BR_PLIMIT_UPPER32_P3				3ACh
				330h					3B0h
				334h					3B4h
				338h					3B8h
				33Ch					3BCh
DOORBELL_SNOOP_ADDR0_P2				340h	DOORBELL_SNOOP_ADDR0_P3				3C0h
				344h					3C4h
DOORBELL_SNOOP_ADDR1_P2				348h	DOORBELL_SNOOP_ADDR1_P3				3C8h
				34Ch					3CCh
DOORBELL_SNOOP_ADDR2_P2				350h	DOORBELL_SNOOP_ADDR2_P3				3D0h
				354h					3D4h
DOORBELL_SNOOP_ADDR3_P2				358h	DOORBELL_SNOOP_ADDR3_P3				3D8h
				35Ch					3DCh
				360h					3E0h
				364h					3E4h
			WCERRLOG0_P2	368h				WCERRLOG0_P3	3E8h
			WCERRLOG1_P2	36Ch				WCERRLOG1_P3	3ECh
			WCERRLOG2_P2	370h				WCERRLOG2_P3	3F0h
			WCERRLOG3_P2	374h				WCERRLOG3_P3	3F4h
				378h					3F8h
				37Ch					3FCh

3.11.1 PCICMD: PCI Command Register

Device: 8
Function: 0
Offset: 04h

Bit	Attr	Default	Description
15:11	RV	0h	*Reserved*
10	RW	0	**INTxDisable:** Interrupt Disable This bit controls the ability of the DMA engine device to assert a legacy PCI interrupt during DMA completions or DMA errors. 1: Legacy Interrupt mode is disabled 0: Legacy Interrupt mode is enabled
9	RO	0	**FB2B:** Fast Back-to-Back Enable This bit does not apply to the DMA engine Device and hardwired to 0.
8	RW	0	**SERRE:** SERR Message Enable This bit indicates whether the DMA Engine device is allowed to signal a SERR condition. This field handles the reporting of fatal and non-fatal errors by enabling the error pins ERR[2:0]. 1: The DMA engine device is enabled to send fatal/non-fatal errors. 0: The DMA engine device is disabled from generating fatal/non-fatal errors.
7	RV	0	*Reserved*
6	RW	0	**PERRRSP:** Parity Error Response Controls the response when a parity error is detected in the DMA engine 1: The device can report Parity errors 0: Parity errors can be ignored by the device.
5:4	RV	00	*Reserved*
3	RO	0	**SPCEN:** Special Cycle Enable This bit does not apply to the DMA Engine Device.
2	RW	0	**BME:** Bus Master Enable Controls the ability for the DMA engine device to initiate transactions to memory including MMIO 1: Enables the DMA engine device to successfully complete memory read/write requests. 0: Disables upstream memory writes/reads If this bit is not set and the DMA engine is programmed by software to process descriptors, the chipset will flag read(write) errors (*DMA8/*DMA9) and also record the errors in the CHANERR registers when it attempts to issue cacheline requests to memory.
1	RW	0	**MAEN:** Memory Access Enable Controls the ability for the DMA Engine Device to respond to memory mapped I/O transactions initiated in the Intel 5000X Chipset MCH in its range. 1: Allow MMIO accesses in the DMA Engine 0: Disable MMIO accesses in DMA Engine This only applies to access CB_BAR space in Device 8, fn 1 where the MMIO space resides (Requests from both fast/slow paths will be master-aborted)
0	RO	0	**IOAEN:** I/O Access Enable Controls the ability for the DMA Engine Device to respond to legacy I/O transactions. The DMA Engine Device does not support/allow legacy I/O cycles.

The PCI Command register follows a subset of the *PCI Local Bus Specification,* Revision 2.3 specification. This register provides the basic control of the ability of the DMA engine device to initiate and respond to transactions sent to it and maintains compatibility with PCI configuration space.

3.11.2 PCISTS: PCI Status Register

Device: 8
Function: 0
Offset: 06h

Bit	Attr	Default	Description
15	RWC	0	**DPE:** Detected Parity Error This bit is set when the DMA engine device receives an uncorrectable data error or Address/Control parity errors regardless of the Parity Error Enable bit (PERRE). This applies only to parity errors that target the DMA engine device (inbound/outbound direction). The detected parity error maps to B1, F6, M2 and M4 (uncorrectable data error from FSB, Memory or internal sources). The DMA engine also records the data parity error in bit[6] (**Cdata_par_err**)of the CHANERR register.
14	RWC	0	**SSE:** Signalled System Error 1: The DMA engine device reported internal FATAL/NON FATAL errors (DMA0-15) through the ERR[2:0] pins with SERRE bit enabled. Software clears this bit by writing a '1' to it. 0: No internal DMA engine device port errors are signaled.
13	RO	0	**RMA:** Received Master Abort Status This field is hardwired to 0 as there is no Master Abort for the DMA operations
12	RWC	0	**RTA:** Received Target Abort Status This field is hardwired to 0 as there is no Target Abort for the DMA operations
11	RWC	0	**STA: Signalled Target Abort Status**: This field is hardwired to 0
10:9	RO	00	**DEVSELT: DEVSEL# Timing**: This bit does not apply to the DMA Engine Device.
8	RWC	0	**MDIERR:** Master Data Integrity Error This bit is set by the DMA engine device if the Parity Error Enable bit (PERRE) is set and it receives error B1, F2, F6, M2 and M4 (uncorrectable data error or Address/Control parity errors or an internal failure). If the PERRRSP bit in the Section 3.11.1 is cleared, this bit is never set.
7	RO	0	**FB2B:** Fast Back-to-Back Capable Not applicable to DMA Engine. Hardwired to 0.
6	RV	0	*Reserved*
5	RO	0	**66MHZCAP: 66 MHz capable**. Not applicable to DMA Engine. Hardwired to 0.
4	RO	1	**CAPL: Capability List Implemented**: This bit indicated that the DMA Engine device implements a PCI Capability list. See CAPPTR at offset 34h
3	RO	0	**INTxST: I**NTx State This bit is set by the hardware when the DMA engine device issues a legacy INTx (pending) and is reset when the Intx is deasserted. The intx status bit should be deasserted when all the relevant status bits/events viz DMA errors/completions that require legacy interrupts are cleared by software.
2:0	RV	000	*Reserved*

The PCI Status register follows a subset of the *PCI Local Bus Specification,* Revision 2.3 specification. This register maintains compatibility with PCI configuration space. Since this register is part of the standard PCI header, there is a PCISTS register per PCI function.

3.11.3 CCR: Class Code Register

Device: 8 Function: 0 Offset: 09h			
Bit	**Attr**	**Default**	**Description**
23:16	RWO	08h	**Base Class Code**: A 08H code indicates that the DMA engine device is a peripheral device[a]. A 06H code is used to indicate a Host bridge device. Default: 08h
15:8	RWO	80h	**Sub-Class Code**: An 80H code indicates that the DMA engine device is a non-specific peripheral device. A 00H code is used to indicate a Host bridge device. Default: 80h
7:0	RWO	0h	**Register-Level Programming Interface**: This field identifies a default value for non-specific programming requirements.

Notes:

a. A peripheral device in this case denotes an integrated device in the root complex.

The bits in this register are writable once by BIOS in order to allow the device to be programmable either as an OS-visible device [088000h](implementing a driver) or a chipset host bridge device [060000h] (relying on BIOS code and/or pure hardware control for programming the DMA engine registers). The default value of the CCR is set to 088000h (corresponding to an integrated device in the root port).

3.11.4 CB_BAR: DMA Engine Base Address Register

Device: 8 Function: 0 Offset: 10h			
Bit	**Attr**	**Default**	**Description**
63:40	RO	0h	**CB_BASE_Win_Upper: Upper DMABase Window**: The upper bits of the 64-bit addressable space are initialized to 0 as default and is unusable in Intel 5000 Series Chipset.
39:10	RW	003F9C00h	**CB_BASE_WIN:** DMABase Window This marks the 1KB memory-mapped registers used for the chipset DMA and can be placed in any MMIO region (low/high) within the physical limits of the system. For instance the Intel 5000X Chipset MCH uses only 40-bit addressable space. Hence bits 39:10 are assumed to be valid and also contains the default value of the CB_BAR in the FE70_0000h to FE70_03FFh range.
9:4	RV	0h	*Reserved*
3	RO	0	**Pref:** Prefetchable The DMA registers are not prefetchable.
2:1	RO	10	**Type:** Type The DMA registers is 64-bit address space and can be placed anywhere within the addressable region of the Intel 5000 Series Chipset (up to 40-bits).
0	RO	0	**Mem_space:** Memory Space This Base Address Register indicates memory space.

This DMA Engine base address register marks the memory-mapped registers used for the DMA functionality.

3.11.5 CAPPTR: Capability Pointer Register

Device: 8 Function: 0 Offset: 34h			
Bit	**Attr**	**Default**	**Description**
7:0	RO	50h	**CAPPTR:** Capability Pointer This register field points to the first capability. PM structure in the DMA Engine device.

3.11.6 INTP: Interrupt Pin Register

The INTP register identifies legacy interrupts for INTA, INTB, INTC and INTD as determined by BIOS/firmware. These are emulated over the ESI port using the Assert_Intx commands as appropriate.

Device: 8 Function: 0 Offset: 3Dh			
Bit	**Attr**	**Default**	**Description**
7:0	RWO	01h	**INTP:** Interrupt Pin This field defines the type of interrupt to generate for the PCI Express port. 001: Generate INTA 010: Generate INTB 011: Generate INTC 100: Generate INTD Others: Reserved

3.11.7 DMACTRL: DMA Control Register

This 32-bit register implements DMA specific operations for general control/accessibility such as defeaturing, arbitration.

Device: 8 Function: 0 Offset: 48h			
Bit	**Attr**	**Default**	**Description**
31:6	RV	0h	*Reserved.*

Device: 8
Function: 0
Offset: 48h

Bit	Attr	Default	Description
5	RW	0	**DMA_OO_MMIO_FETCH_CMP_ENABLE:** DMA Out of Order Fetch completion enable for MMIO writes This field controls the enforcement of a weak/strongly ordered model for pushing MMIO writes to the destination NIC. The weakly ordered model allows for writes to be completed out of order with increased performance since the DMA does not have to wait for the completions (in linear order) from the CE and the overall throughput is higher. 0: Disable out-of-order write fetches and enforce serialization of write requests to the destination (default). 1: Enable DMA engine to complete out-of-order write fetches to the destination (MMIO) for maximizing performance. It is the responsibility of the BIOS/Software to program this register field appropriately based on the usage/Producer-consumer requirements. ***Note:*** In either mode setting, all writes are completed before a status write or interrupt is generated and then the next descriptor (if any) is read from memory by the DMA engine for further processing. ***Note:*** In the case of memory to memory transfers, the Intel 5000X Chipset MCH CE acks the fetch completion immediately (when there is no conflict) and since the DMA engine issues fetches in order, the fetch completions also follow suit and are therefore amenable for pipelining without any reordering penalty for the general case.
4	RW	0h	**DMA_CONC_FETCH_DISABLE_:** DMA Concurrent Fetch Disable This field provides defeature control for enabling/disabling the concurrent fetch request optimization to maximize data throughput of the DMA engine. 0: Enable concurrent fetch and data transfer (i.e overlapped read/write during fetch phase) for performance (default) 1: Disable concurrent fetch operation. that is, Read and Write fetch requests will be serialized.
3:2	RW	0h	**NUM_DMA_PREF:** Number of outstanding DMA Prefetches This field controls the total number of DMA prefetches that are outstanding for both reads and writes issued by the DMA engine across all four channels 00: 24 (12 Reads + 12 writes) (default) 01: 20 (10 Reads + 10 writes)[a] 10: 16 (8 Reads + 8 writes) 11: Reserved The default value is to enable the DMA engine to prefetch up to 24 Cache lines (Reads and writes) for maximum performance. For system debug or for any issues leading to starvation of transaction IDs or bandwidth problems, this register field can be manipulated to reduce the DMA engine's prefetch limit.
1	RW	0	**MSICBEN:** MSI DMA Engine Interrupt Enable 1: Enables MSI messages (errors or Channel completions) to be sent to the root complex for DMA Engine interrupts on the DMA engine 0: Disables sending of MSI messages for DMA Engine Interrupts to the root complex. Note that for MSI DMA Engine DMA interrupt messages to be sent, both MSICBEN and MSICTRL.MSIEN bits defined in Section 3.8.10.3 have to be set.
0	RW	1	**DMAEN:** DMA Enable 1: Enables DMA engine to perform DMA Engine related data transfers (M2M or M2MMIO) 0: Disables DMA engine from performing DMA Engine related data transfers (M2M or M2MMIO). Read accesses to CB_BAR MMIO space (up to 300h in Device 8, function 1) will return 0s and writes will have no effect. The DMAEN is a feature bit that controls DMA transfers for all the supported channels of the DMA engine and must be enabled for normal operation.

Notes:

a. In the degraded mode, when the total prefetches are set to 20 (that is, 10 + 10) in the 2 channel case for example, then one channel may issue 6+6 prefetches while the other can only reach 4+4 requests. It is the expectation that this asymmetry will balance over time.

3.11.8 Power Management Capability Structure

The DMA engine integrated device within the MCH incorporates power management capability with D0 (working) and a pseudo D3 hot/cold states (sleep) that can be controlled independently through software. From a software perspective, the D3 states convey information to the power controller that the device is in the sleep mode though the physical entity inside the chipset may be fully powered. During transition[1] from D0 to D3, it will ensure that all pending DMA Channels are completed in full.

3.11.8.1 PMCAP - Power Management Capabilities Register

The PM Capabilities Register defines the capability ID, next pointer and other power management related support. The following PM registers /capabilities are added for software compliance.

Device: 8
Function: 0
Offset: 50h

Bit	Attr	Default	Description
31:27	RO	11001	**PMES:** PME Support Identifies power states which assert PMEOUT. Bits 31, 30 and 27 must be set to '1' for PCI-PCI bridge structures representing ports on root complexes. The definition of these bits is taken from the PCI Bus Power Management Interface Specification Revision 1.1. XXXX1b - PMEOUT can be asserted from D0 XXX1Xb - PMEOUT can be asserted from D1 (Not supported by Intel 5000X Chipset MCH) XX1XXb - PMEOUT can be asserted from D2 (Not supported by Intel 5000X Chipset MCH) X1XXXb - PMEOUT can be asserted from D3 hot 1XXXXb - PMEOUT can be asserted from D3 cold
26	RO	0	**D2S:** D2 Support The Intel 5000X Chipset MCH does not support power management state D2.
25	RO	0	**D1S:** D1 Support The Intel 5000X Chipset MCH does not support power management state D1.
24:22	RO	0h	**AUXCUR:** AUX Current
21	RO	0	**DSI:** Device Specific Initialization
20	RV	0	*Reserved.*
19	RO	0	**PMECLK:** PME Clock This field is hardwired to 0h.
18:16	RO	010	**VER:** Version This field is set to 2h as version number from the PCI Express 1.0 specification.
15:8	RO	58h	**NXTCAPPTR:** Next Capability Pointer This field is set to offset 58h for the next capability structure (MSI) in the PCI 2.3 compatible space.
7:0	RO	01h	**CAPID:** Capability ID Provides the PM capability ID assigned by PCI-SIG.

1. When software initiates an S0 => S3 transition, it should make the DMA engine device to enter D3 before completing the power management handshake with the MCH.

3.11.8.2 PMCSR - Power Management Control and Status Register

This register provides status and control information for PM events in the PCI Express port of the DMA Engine Device.

Device: 8
Function: 0
Offset: 54h

Bit	Attr	Default	Description
31:24	RO	0h	**Data:** Data Data read out based on data select (DSEL). Refer to section 3.2.6 of PCI PM specification for details. This is not implemented in the e Power Management capability for Intel 5000X Chipset MCH and is hardwired to 0h.
23	RO	0h	**BPCCEN:** Bus Power/Clock Control Enable This field is hardwired to 0h.
22	RO	0h	**B2B3S:** B2/B3 Support This field is hardwired to 0h.
21:16	RV	0h	*Reserved.*
15	RWCST	0h	**PMESTS:** PME Status This PME Status is a sticky bit. When set, the device generates a PME internally independent of the PMEEN bit defined below. Software clears this bit by writing a '1'. As an integrated device within the root complex, the Intel 5000X Chipset MCH will never set this bit, because it never generates a PME internally independent of the PMEEN bit.
14:13	RO	0h	**DSCL:** Data Scale This 2-bit field indicates the scaling factor to be used while interpreting the "data_scale" field.
12:9	RO	0h	**DSEL:** Data Select This 4-bit field is used to select which data is to reported through the "data" and the "Data Scale" fields.
8	RWST	0h	**PMEEN:** PME Enable This field is a sticky bit and when set enables PMEs generated internally to appear at the Intel 631xESB/632xESB I/O Controller Hub through the "Assert(Deassert)_PMEGPE"message. This has no effect on the Intel 5000X Chipset MCH since it does not generate PME events internally.
7:2	RV	0h	*Reserved.*
1:0	RW	0h	**PS:** Power State This 2-bit field is used to determine the current power state of the function and to set a new power state as well. 00: D0 01: D1 (reserved) 10: D2 (reserved) 11: D3_hot

3.11.8.3 MSICAPID - Message Signalled Interrupt Capability ID Register

Device: 8
Function: 0
Offset: 58h

Bit	Attr	Default	Description
7:0	RO	05h	**CAPID:** MSI Capability ID This code denotes the standard MSI capability assigned by PCI-SIG

3.11.8.4 MSINXPTR - Message Signalled Interrupt Next Pointer Register

Device: 8
Function: 0
Offset: 59h

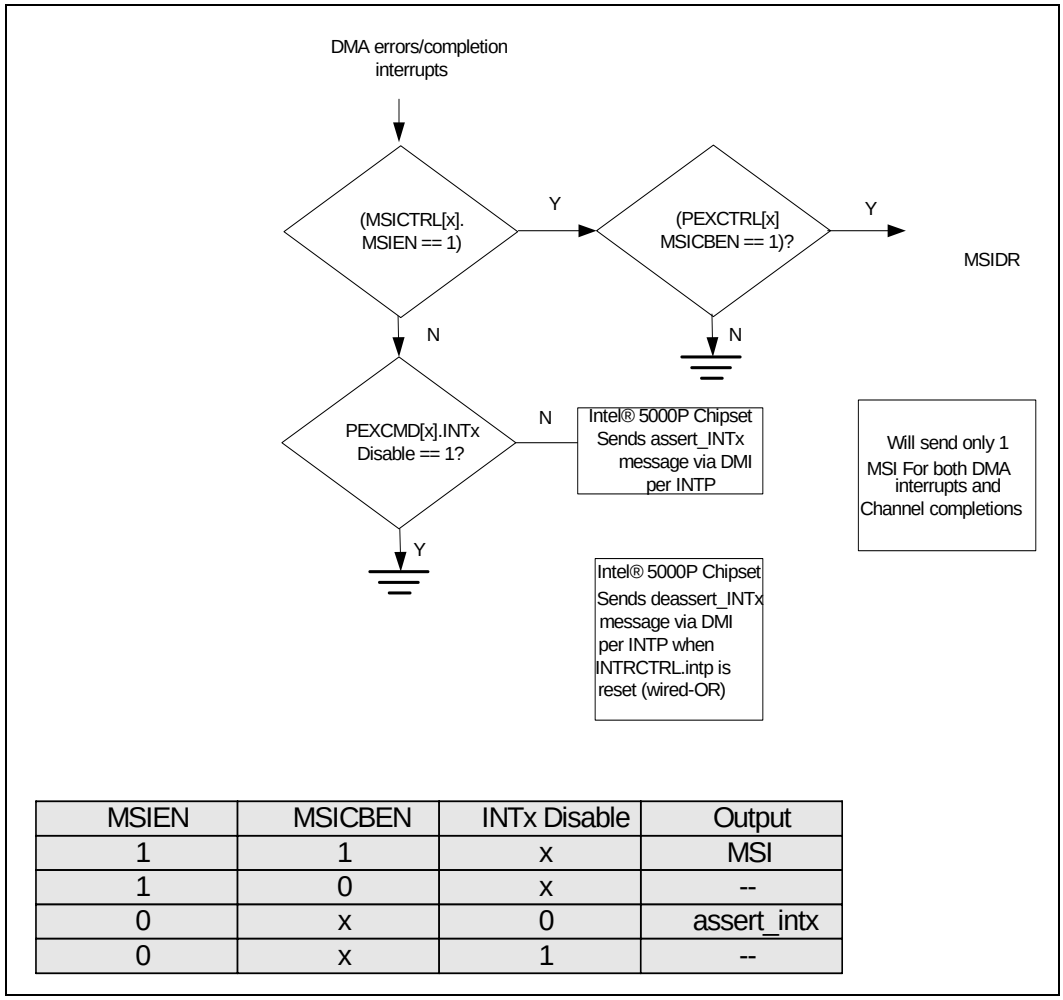
Bit	Attr	Default	Description
7:0	RO	6Ch	**NXTPTR: MSI Next Pointer**: The DMA Engine device is implemented as a PCI Express device and this points to the PCI Express capability structure.

3.11.8.5 MSICTRL - Message Signalled Interrupt Control Register

Device: 8
Function: 0
Offset: 5Ah

Bit	Attr	Default	Description
15:8	RO	0h	*Reserved*
7	RO	0	**AD64CAP:** 64-bit Address Capable All processors used with the GNC MCH do not support 64-bit addressing, hence this is hardwired to 0
6:4	RW	000	**MMEN:** Multiple Message Enable Software initializes this to indicate the number of allocate messages which is aligned to a power of two. When MSI is enabled, the software will allocate at least one message to the device. See Section 3.11.8.7 below for discussion on how the interrupts are handled.
3:1	RO	0h	**MMCAP:** Multiple Message Capable The Intel 5000X Chipset MCH DMA Engine supports only one interrupt message (power of two) for handling • DMA errors • DMA completions
0	RW	0	**MSIEN:** MSI Enable This bit enables MSI as the interrupt mode of operation instead of the legacy interrupt mechanism. 0: Disables MSI from being generated. 1: Enables MSI messages to be generated for DMA related interrupts. An extract of the flowchart of the DMA Engine error handling is given in Figure 3-7

Figure 3-7. Intel 5000X Chipset DMA Error/Channel Completion Interrupt Handling Flow



MSIEN	MSICBEN	INTx Disable	Output
1	1	x	MSI
1	0	x	--
0	x	0	assert_intx
0	x	1	--

3.11.8.6 MSIAR: Message Signalled Interrupt Address Register

Device: 8
Function: 0
Offset: 5Ch

Bit	Attr	Default	Description
31:20	RO	FEEh	**AMSB:** Address MSB This field specifies the 12 most significant bits of the 32-bit MSI address.
19:12	RW	0h	**ADSTID:** Address Destination ID This field is initialized by software for routing the interrupts to the appropriate destination.
11:4	RW	0h	**AEXDSTID:** Address Extended Destination ID This field is not used by IA32 processor.
3	RW	0	**ARDHINT:** Address Redirection Hint 0: directed 1: redirectable
2	RW	0	**ADM:** Address Destination Mode 0: physical 1: logical
1:0	RV	00	*Reserved*. Not used since the memory write is D-word aligned

3.11.8.7 MSIDR: Message Signalled Interrupt Data Register

Device: 8
Function: 0
Offset: 60h

Bit	Attr	Default	Description
31:16	RV	0000h	*Reserved.*
15	RW	0h	**TM:** Trigger Mode This field Specifies the type of trigger operation 0: Edge 1: level
14	RW	0h	**LVL:** Level if TM is 0h, then this field is a don't care. Edge triggered messages are consistently treated as assert messages. For level triggered interrupts, this bit reflects the state of the interrupt input if TM is 1h, then: 0: Deassert Messages 1: Assert Messages
13:11	RW	0h	These bits are don't care in IOxAPIC interrupt message data field specification.
10:8	RW	0h	**DM:** Delivery Mode 000: Fixed 001: Lowest Priority 010: SMI/HMI 011: *Reserved* 100: NMI 101: INIT 110: *Reserved* 111: ExtINT

Device: 8 Function: 0 Offset: 60h			
Bit	**Attr**	**Default**	**Description**
7:0	RW	0h	**IV:** Interrupt Vector The interrupt vector as programmed by BIOS/Software will be used by the Intel 5000X Chipset MCH to provide context sensitive interrupt information for different events such as DMA Errors, DMA completions that require attention from the processor. See Table 3-55 for IV handling for DMA.

Table 3-55. IV Vector Table for DMA Errors and Interrupts

Number of Messages enabled by Software (MMEN)	Events	IV[7:0]
1	All (DMA completions/errors)	xxxxxxxx[a]

Notes:

a. The term "xxxxxx" in the Interrupt vector denotes that software/BIOS initializes them and the MCH will not modify any of the "x" bits since it handles only 1 message vector that is common to all events.

3.11.8.8 PEXCAPID: PCI Express Capability ID Register

Device: 8 Function: 0 Offset: 6Ch			
Bit	**Attr**	**Default**	**Description**
7:0	RO	10h	**CAPID:** PCI Express Capability ID This code denotes the standard PCI Express capability.

3.11.8.9 PEXNPTR: PCI Express Next Pointer Register

Device: 8 Function: 0 Offset: 6Dh			
Bit	**Attr**	**Default**	**Description**
7:0	RO	00h	**NXTPTR:** PCI Express Next Pointer The PCI Express capability structure is the last capability in the linked list and set to NULL.

3.11.8.10 PEXCAPS - PCI Express Capabilities Register

Device: 8 Function: 0 Offset: 6Eh			
Bit	**Attr**	**Default**	**Description**
15:14	RV	0h	*Reserved*

Device: 8 Function: 0 Offset: 6Eh			
Bit	**Attr**	**Default**	**Description**
13:9	RO	0h	**IMN:** Interrupt Message Number: This field indicates the interrupt message number that is generated from the DMA Engine device. When there are more than one MSI interrupt Number, this register field is required to contain the offset between the base Message Data and the MSI Message that is generated when the status bits in the slot status register or root port status registers are set.
8	RO	0	**Slot_Impl:** Slot Implemented: DMA Engine is an integrated device and therefore a slot is never implemented.
7:4	RO	0000	**DPT:** Device/Port Type: DMA Engine device represents a PCI Express Endpoint.
3:0	RO	0001	**VERS:** Capability Version: DMA Engine supports Revision 1 of the PCI Express specification.

3.11.8.11 PEXDEVCAP - Device Capabilities Register

Device: 8 Function: 0 Offset: 70h			
Bit	**Attr**	**Default**	**Description**
31:28	RV	0h	Reserved
27:26	RO	00	**CSPLS:** Captured Slot Power Limit Scale This field applies only to upstream ports. Hardwired to 0h
25:18	RO	00h	**CSPLV:** Captured Slot Power Limit Value This field applies only to upstream ports. Hardwired to 0h
17:15	RV	0h	*Reserved*
14	RO	0	**PIPD: Power Indicator Present** The DMA Engine is an integrated device and therefore, an Power Indicator does not exist. Hardwired to 0h
13	RO	0	**AIPD:** Attention Indicator Present The DMA Engine is an integrated device and therefore, an Attention Indicator does not exist. Hardwired to 0h
12	RO	0	**ABPD:** Attention Button Present The DMA Engine is an integrated device and therefore, an Attention Button does not exist. Hardwired to 0h
11:9	RO	000	**EPL1AL:** Endpoint L1 Acceptable Latency The DMA Engine device is not implemented on a physical PCI Express link and therefore, this value is irrelevant. Hardwired to 0h
8:6	RO	000	**EPL0AL:** Endpoint L0s Acceptable Latency The DMA Engine device is not implemented on a physical PCI Express link and therefore, this value is irrelevant. Hardwired to 0h
5	RO	0	**ETFS:** Extended Tag Field Supported The DMA Engine device does not support extended tags. Hardwired to 0h
4:3	RO	00	**PFS:** Phantom Functions Supported The DMA Engine device does not support Phantom Functions. Hardwired to 0h
2:0	RO	000	**MPLSS:** Max_Payload_Size Supported This field indicates the maximum payload size that the CB integrated device can support. 000: 128 B max payload size others- *Reserved*

3.11.8.12 PEXDEVCTRL - Device Control Register

Device: 8
Function: 0
Offset: 74h

Bit	Attr	Default	Description
15	RV	0	Reserved
14:12	RO	000	**MRRS:** Max_Read_Request_Size Since the DMA Engine device does not issue read requests on a PCI Express interface, this field is irrelevant. Hardwired to 0h
11	RW	1	**ENNOSNP:** Enable No Snoop 1: Setting this bit enables the DMA Engine device to issue requests with the No Snoop attribute. 0: Clearing this bit behaves as a global disable when the corresponding capability is enabled for source/destination snoop control in the DMA's descriptor's Desc_Control field.
10	RO	0	**APPME:** Auxiliary Power PM Enable The DMA Engine device does not implement auxiliary power so setting this bit has no effect. Hardwired to 0h
9	RO	0	**PFEN:** Phantom Functions Enable The DMA Engine device does not implement phantom functions so setting this bit has no effect. Hardwired to 0h
8	RO	0	**ETFEN: Extended Tag Field Enable**: The DMA Engine device does not implement extended tags so setting this bit has no effect.
7:5	RW	000	**MPS: Max_Payload_Size**: The DMA Engine device must not generate packets on any PCI Express interface which exceeds the length allowed with this field. 000: 128 B max payload size 001: 256 B max payload size 010: 512 B max payload size 011: 1024 B max payload size 100: 2048 B max payload size 101: 4096 B max payload size ***Note:*** This field has no impact internally to Intel 5000X Chipset MCH and the maximum payload size of the TLPs that appear on the PCI Express port is governed by the PEXDEVCTRL.MPS for that port defined in Table 3.8.11.4.
4	RO	0	**ENRORD: Enable Relaxed Ordering** No relaxed ordering is supported by Intel 5000X Chipset MCH. Hardwired to 0h.
3	RO	0	**URREN:** Unsupported Request Reporting Enable For an integrated DMA Engine device, this bit is irrelevant. Hardwired to 0h
2	RW	0	**FERE: Fatal Error Reporting Enable**: This bit controls the reporting of fatal errors internal to the DMA Engine device 0: Fatal error reporting is disabled 1: Fatal error reporting is enabled
1	RW	0	**NFERE:** Non-Fatal Error Reporting Enable This bit controls the reporting of non fatal errors internal to the DMA Engine device in the PCI Express port. 0: Non Fatal error reporting is disabled 1: Non Fatal error reporting is enabled This has no effect on the Intel 5000X Chipset MCH DMA Engine device as it does not report any non-fatal errors.

Device: 8 Function: 0 Offset: 74h			
Bit	**Attr**	**Default**	**Description**
0	RW	0	**CERE:** Correctable Error Reporting Enable This bit controls the reporting of correctable errors internal to the DMA Engine device in the PCI Express port. 0: Correctable error reporting is disabled 1: Correctable Fatal error reporting is enabled This has no effect on the Intel 5000X Chipset MCH DMA Engine device as it does not report any correctable errors.

3.11.8.13 PEXDEVSTS - PCI Express Device Status Register

Device: 8 Function: 0 Offset: 76h			
Bit	**Attr**	**Default**	**Description**
15:6	RV	0h	*Reserved*
5	RO	0	**TP:** Transactions Pending This bit indicates that the DMA Engine device has issued non-posted PCI Express transactions which have not yet completed. Note the Intel 5000X Chipset MCH DMA Engine device does not issue any NP transactions and hence this is hardwired to zero.
4	RO	0	**APD:** AUX Power Detected The DMA Engine device does not support AUX power. Hardwired to 0h.
3	RO	0	**URD:** Unsupported Request Detected This does not apply to DMA Engine in Intel® 5000 Series Chipset as there are no messages for the DMA engine. Hardwired to 0h
2	RWC	0	**FED:** Fatal Error Detected This bit gets set if a fatal uncorrectable error is detected. Errors are logged in this register regardless of whether error reporting is enabled or not in the Device Control register (See FERE in Section 3.11.8.12) 1: Fatal errors detected 0: No Fatal errors detected
1	RWC	0	**NFED:** Non-Fatal Error Detected This bit gets set if a non-fatal uncorrectable error is detected. Errors are logged in this register regardless of whether error reporting is enabled or not in the Device Control register. (See NFERE in Section 3.11.8.12) 1: Non Fatal errors detected 0: No non-Fatal Errors detected
0	RWC	0	**CED:** Correctable Error Detected This bit gets set if a correctable error is detected. Errors are logged in this register regardless of whether error reporting is enabled or not in the Device Control register. (See CERE in Section 3.11.8.12) 1: correctable errors detected 0: No correctable errors detected

3.11.9 DMA Error Logging

The following error registers record the occurrence of the first and next errors of the DMA engine along with the necessary debug information based on the context. Refer to the RAS Chapter for further details.

3.11.9.1 CB_ERR_DOCMD - DMA Engine Error Do Command Register

Device: 8
Function: 0
Offset: 7Ch

Bit	Attr	Default	Description
31:2	RV	0h	*Reserved*.
1:0	RWST	00	**CB_FAT_MAP:** DMA Engine steering for fatal errors 00: ERR[0] 01: ERR[1] 10: ERR[2] 11: MCERR The CB Fatal errors are routed to one of the ERR[2:0] pins or MCERR.

3.11.9.2 FERR_CHANERR - First Error Channel Error Register

The Channel Error Register records the first error conditions occurring within a given DMA channel. The channel number in error is recorded in the FERR_CHANSTS register defined in Section 3.11.9.4.

Device: 8
Function: 0
Offset: 80h

Bit	Attr	Default	Description
15	RO	0	**FERR_Unaffiliated_Error_DMA15: FERR_Unaffiliated_Error_DMA15** The Intel 5000X Chipset MCH does not detect unaffiliated errors
14	RO	0	**FERR_Soft_Error_DMA14**: **FERR_Soft_Error_DMA14** The Intel 5000X Chipset MCH DMA engine does not record/detect any soft errors.
13	RWCST	0	**FERR_Interrupt_Configuration Error_DMA13**: **FERR_Interrupt_Configuration Error_DMA13** The DMA channel sets this bit indicating that the interrupt registers were not configured properly when the DMA channel attempted to generate an interrupt.
12	RWCST	0	**FERR_Completion_Address Error_DMA12**: **FERR_Completion_Address Error_DMA12** The DMA channel sets this bit indicating that the completion address register was configured to an illegal address[a] or has not been configured. This address will be checked and set by the DMA engine during execution. that is, when the completion address is fetched for status update.
11	RWCST	0	**FERR_Descriptor_Length Error_DMA11**: **FERR_Descriptor_Length Error_DMA11** The DMA channel sets this bit indicating that the current transfer has an illegal length field value (either zero or exceeded the maximum length allowed in the XFERCAP.trans_cap field in Section 3.11.10.2
10	RWCST	0	**FERR_Descriptor_Control_Error_DMA10**: **FERR_Descriptor_Control_Error_DMA10** The DMA channel sets this bit indicating that the current descriptor has an illegal control field value in the “desc_control” field.
9	RWCST	0	**FERR_Write_Data_Error_DMA9**: **FERR_Write_Data_Error_DMA9** The DMA channel sets this bit indicating that the current transfer has encountered an error while writing the destination data (for example, no space available in DM). When this bit has been set, the address of the failed descriptor is in the Channel Status register.
8	RWCST	0	**FERR_Read_Data_Error_DMA8**: **FERR_Read_Data_Error_DMA8** The DMA channel sets this bit indicating that the current transfer has encountered an error while accessing the source data. (for example, starvation). When this bit has been set, the address of the failed descriptor is in the Channel Status register.

Device: 8 Function: 0 Offset: 80h			
Bit	**Attr**	**Default**	**Description**
7	RO	0	**FERR_DMA_Data_Parity Error_DMA7**: **FERR_DMA_Data_Parity Error_DMA7** The DMA engine has no internal parity checking and is hard-wired to 0.
6	RWCST	0	**FERR_Chipset_Data_Parity_Error_DMA6**: **FERR_Chipset_Data_Parity_Error_DMA6** The DMA channel sets this bit indicating that the current transfer has encountered a parity error reported by the chipset. When this bit has been set, the address of the failed descriptor is in the Channel Status register. In the case of Source data error, FERR*DMA8 is also set. In the case of Destination data error, FERR*DMA9 is also set
5	RWCST	0	**FERR_CHANCMD_Error_DMA5**: **FERR_CHANCMD_Error_DMA5** The DMA channel sets this bit indicating that a write to the CHANCMD register contained an invalid value (for example, more than one command bit set).
4	RWCST	0	**FERR_Chain Address_Value_Error_DMA4**: **FERR_Chain Address_Value_Error_DMA4** The DMA channel sets this bit indicating that the CHAINADDR register has an illegal address including an alignment error (not on a 64-byte boundary). This address will be checked and set by the DMA engine during execution. that is, when the initial descriptor is fetched
3	RWCST	0	**FERR_Descriptor_Error_DMA3**: **FERR_Descriptor_Error_DMA3** The DMA channel sets this bit indicating that the current descriptor has encountered an error when executing a DMA descriptor that is not otherwise related to other error bits. For example, an illegal next descriptor address flagged by the system Address decoder, which the DMA engine encounters in the current descriptor after having successfully completed the data transfer for the current descriptor including any associated completions/interrupts.
2	RWCST	0	**FERR_Next Descriptor_ Address_Error_DMA2**: **FERR_Next Descriptor_ Address_Error_DMA2** The DMA channel sets this bit indicating that the current descriptor has an illegal next descriptor address (for example, > 40-bits) including next descriptor alignment error (not on a 64-byte boundary). This error could be flagged when the data for the current descriptor is fetched and its constituent fields are checked.
1	RWCST	0	**FERR_DMA Transfer_Destination_Address_Error_DMA1**: **FERR_DMA Transfer_Destination_Address_Error_DMA1** The DMA channel sets this bit indicating that the current descriptor has an illegal destination address.
0	RWCST	0	**FERR_DMA Transfer_Source Address_Error_DMA0: FERR_DMA Transfer_Source Address_Error_DMA0** The DMA channel sets this bit indicating that the current descriptor has an illegal source address.

Notes:

a. An illegal address is one which is flagged by the Intel 5000X Chipset MCH system address decoder as invalid (for example, hitting a reserved space in memory) or which is decoded by the DMA engine as erroneous. e.g >40-bits in length or an alignment issue.

3.11.9.3 FERR_CHANCMD - First Error Channel Command Register

Device: 8 Function: 0 Offset: 83h			
Bit	**Attr**	**Default**	**Description**
7:5	RV	0h	Reserved
4	RWCST	0	**FERR_Resume DMA**: **FERR_Resume DMA** Records the “Resume_DMA” field during the occurrence of the first error

Device: 8 Function: 0 Offset: 83h			
Bit	**Attr**	**Default**	**Description**
3	RWCST	0	**FERR_Abort DMA**: **FERR_Abort DMA** Records the “Abort_DMA” field during the occurrence of the first error
2	RWCST	0	**FERR_Suspend DMA**: **FERR_Suspend DMA** Records the “Suspend_DMA” field during the occurrence of the first error
1	RWCST	0	**FERR_Append DMA**: **FERR_Append DMA** Records the “Append_DMA” field during the occurrence of the first error
0	RWCST	0	**FERR_Start DMA**: Records the “Start_DMA” field during the occurrence of the first error

This register records the first error of the CHANCMD register when the unaffiliated error happens (that is, more than 1 bit is set in this register).

3.11.9.4 FERR_CHANSTS - First Error Channel Status Register

Device: 8 Function: 0 Offset: 84h			
Bit	**Attr**	**Default**	**Description**
63:6	RWCST	0h	**FERR_Completed Descriptor Address**: This register stores the upper address bits of the current DMA descriptor (CHAINADDR) when an error is encountered.
5	RV	0	*Reserved*
4:3	RWCST	00	**FERR_DMA Channel number**: This field records the Channel number of the DMA engine that encountered an error.
2:0	RWCST	011	**FERR_DMA Transfer Status**: The DMA engine sets the bits indicating the state of the current DMA transfer when an error is encountered. 000 - Active 001 - Completed, DMA Transfer Done (no hard errors) 010 - Suspended 011 - Halted, operation aborted (refer to Channel Error register for further detail)

3.11.9.5 FERR_DESC_CTRL - First Error Descriptor Control Register

This register contains the desc_ctrl fields extracted from the current descriptor when an error happened.

Device: 8
Function: 0
Offset: 8Ch

Bit	Attr	Default	Description
31:0	RWCST	0h	**FERR_Desc_ctrl**: This field records the “desc_ctrl” field from the current descriptor that encountered an error.

3.11.9.6 FERR_SADDR - First Error Source Address Register

This register records the source address of the current descriptor which encountered an error.

Device: 8
Function: 0
Offset: 90h

Bit	Attr	Default	Description
63:0	RWCST	0h	**FERR_Source Address**: This 64 bit field marks the address of the source address in a given descriptor that encountered an error.

3.11.9.7 FERR_DADDR - First Error Destination Address Register

This register records the destination address of the current descriptor which encountered an error.

Device: 8
Function: 0
Offset: 98h

Bit	Attr	Default	Description
63:0	RWCST	0h	**FERR_Destination Address**: This 64 bit field marks the address of the destination address in a given descriptor that encountered an error.

3.11.9.8 FERR_TRANSFER_SIZE - First Error Transfer Size Register

This register records the transfer size of the current descriptor which encountered an error.

Device: 8
Function: 0
Offset: A0h

Bit	Attr	Default	Description
31:0	RWCST	0h	**FERR_Transfer_size**: This field records the transfer size of the descriptor that was detected in error.

3.11.9.9 FERR_NADDR - First Error Next Address Register

This register records the next address of the current descriptor chain which encountered an error.

Device: 8 Function: 0 Offset: A4h			
Bit	**Attr**	**Default**	**Description**
63:0	RWCST	0h	**FERR_Next Address**: This 64 bit field marks the address of the next address field in a given descriptor that encountered an error.

3.11.9.10 FERR_CHANCMP - First Error Channel Completion Address Register

This register specifies the address of the Channel completion address when an error happened.

Device: 8 Function: 0 Offset: ACh			
Bit	**Attr**	**Default**	**Description**
63:0	RWCST	0h	**FERR_Channel Completion Address**: This field records the upper bits of the Channel completion address when an error was encountered.

3.11.9.11 NERR_CHANERR - Next Channel Error Register

The Channel Error Register records the Next error conditions that occurs for a DMA channel. These next error bits are recorded only if the respective bits are not set prior in the FERR_CHANERR register. Also note that the NERR_CHANERR is common to all channels and only gives an indication of the type of error that follows. Further debug information can be obtained by probing the DMA channels for error, status information (See Section 3.11.11.8).

Device: 8 Function: 0 Offset: BCh			
Bit	**Attr**	**Default**	**Description**
15	RO	0	**NERR_Unaffiliated_Error_DMA15** The Intel 5000X Chipset MCH does not detect unaffiliated errors.
14	RO	0	**NERR_Soft_Error_DMA14** The Intel 5000X Chipset MCH DMA engine does not record/detect any soft errors.
13	RWCST	0	**NERR_Interrupt_Configuration Error_DMA13** The DMA channel sets this bit indicating that the interrupt registers were not configured properly when the DMA channel attempted to generate an interrupt.
12	RWCST	0	**NERR_Completion_Address Error_DMA12** The DMA channel sets this bit indicating that the completion address register was configured to an illegal address or has not been configured. This address will be checked and set by the DMA engine during execution. that is, when the completion address is fetched for status update.
11	RWCST	0	**NERR_Descriptor_Length Error_DMA11** The DMA channel sets this bit indicating that the current transfer has an illegal length field value (either zero or exceeded the maximum length allowed in the XFERCAP.trans_cap field in Section 3.11.10.2
10	RWCST	0	**NERR_Descriptor_Control_Error_DMA10** The DMA channel sets this bit indicating that the current descriptor has an illegal control field value in the “desc_control” field.

Device: 8 Function: 0 Offset: BCh			
Bit	**Attr**	**Default**	**Description**
9	RWCST	0	**NERR_Write_Data_Error_DMA9** The DMA channel sets this bit indicating that the current transfer has encountered an error while writing the destination data (for example, no space available in DM). When this bit has been set, the address of the failed descriptor is in the Channel Status register.
8	RWCST	0	**NERR_Read_Data_Error_DMA8** The DMA channel sets this bit indicating that the current transfer has encountered an error while accessing the source data. (for example, starvation) When this bit has been set, the address of the failed descriptor is in the Channel Status register.
7	RO	0	**NERR_DMA_Data_Parity Error_DMA7** The DMA engine has no internal parity checking and is hard-wired to 0.
6	RWCST	0	**NERR_Chipset_Data_Parity_Error_DMA6** The DMA channel sets this bit indicating that the current transfer has encountered a parity error reported by the chipset. When this bit has been set, the address of the failed descriptor is in the Channel Status register. In the case of Source data error, NERR*DMA8 is also set. In the case of Destination data error, NERR*DMA9 is also set
5	RWCST	0	**NERR_CHANCMD_Error_DMA5** The DMA channel sets this bit indicating that a write to the CHANCMD register contained an invalid value (for example, more than one command bit set).
4	RWCST	0	**NERR_Chain Address_Value_Error_DMA4** The DMA channel sets this bit indicating that the CHAINADDR register has an illegal address including an alignment error (not on a 64-byte boundary). This address will be checked and set by the DMA engine during execution. i.e when the initial descriptor is fetched
3	RWCST	0	**NERR_Descriptor_Error_DMA3** The DMA channel sets this bit indicating that the current descriptor has encountered an error when executing a DMA descriptor that is not otherwise related to other error bits. For example, an illegal next descriptor address flagged by the system address decoder, which the DMA engine encounters in the current descriptor after having successfully completed the data transfer for the current descriptor including any associated completions/interrupts.
2	RWCST	0	**NERR_Next Descriptor_ Address_Error_DMA2** The DMA channel sets this bit indicating that the current descriptor has an illegal next descriptor address (for example, > 40-bits) including next descriptor alignment error (not on a 64-byte boundary). This error could be flagged when the data for the current descriptor is fetched and its constituent fields are checked.
1	RWCST	0	**NERR_DMA Transfer_Destination_Address_Error_DMA1** The DMA channel sets this bit indicating that the current descriptor has an illegal destination address.
0	RWCST	0	**NERR_DMA Transfer_Source Address_Error_DMA0** The DMA channel sets this bit indicating that the current descriptor has an illegal source address.

3.11.10 DMA Registers

Table 3-56 lists the memory-mapped registers used to control the chipset DMA functionality. The DMA channel specific information is contained in four locations starting from offset 80h of the CB_BAR register. Each channel is separated by 128B from other channels. For software compatibility, the DMA Engine device is required to implement these registers at the listed memory-mapped offsets.

There is one set of general registers followed by multiple sets of per-channel registers (that is, one set for each of the 4 DMA channels of the MCH).

All of these registers are accessible from both a processor and an I/O device from the PCI Express ports.2 and 3.

Table 3-56. DMA Memory Mapped Register Set Locations

Register Set	Offset from CB_BAR
General Registers	0000h
Channel 0	0080h
Channel 1	0100h
Channel 2	0180h
Channel 3	0200h

3.11.10.1 CHANCNT - Channel Count

The Channel Count register specifies the number of channels that are implemented.

Offset from CB_BAR:
Offset: 00h

Bit	Attr	Default	Description
7:5	RV	0h	*Reserved*
4:0	RO	00100	**num_chan:** Number of channels Specifies the number of DMA channels. The Intel 5000X Chipset MCH supports 4 DMA Channels. ***Note:*** This field will be set to "00000" by the chipset when the DMA engine is disabled. Refer to Section 3.11.7 for details

3.11.10.2 XFERCAP - Transfer Capacity

The Transfer Capacity specifies the minimum of the maximum DMA transfer size supported on all channels.

Offset from CB_BAR
Offset: 01h

Bit	Attr	Default	Description
7:5	RV	0h	*Reserved*
4:0	RO	01100	**Trans_cap:** Transfer Capacity This field specifies the number of bits that may be specified in a DMA descriptor's Transfer Size field. This defines the minimum of the maximum transfer size supported by Intel 5000X Chipset MCH as a power of 2. 01100: The value of 12 indicates 4K maximum transfer in Intel 5000X Chipset MCH and is the default value Others: *Reserved*

3.11.10.3 INTRCTRL - Interrupt Control

The Interrupt Control register provides for control of DMA interrupts.

Offset from CB_BAR
Offset: 03h

Bit	Attr	Default	Description
7:3	RV	0h	*Reserved*

Offset from CB_BAR Offset: 03h			
Bit	**Attr**	**Default**	**Description**
2	RO	0	**intp:** Interrupt This bit is set whenever any bit in the Attention Status register is set and the Master Interrupt Enable bit is set. That is, it is the logical AND of Interrupt Status and Master Interrupt Enable bits of this register. This bit represents the legacy interrupt drive signal (when in legacy interrupt mode) and for MSI mode, the hardware sends an MSI interrupt each time this signal transitions from reset to set.
1	RO	0	**intp_sts:** Interrupt Status This bit is set whenever any bit in the Attention Status register is set. That is, it is the logical OR of the Channel Attention bits of the ATTNSTATUS register.
0	RW	0	**Mstr_intp_En:** Master Interrupt Enable Setting this bits enables the generation of an interrupt. This bit is automatically reset each time this register is read. When this bit is clear, the chipset will not generate an interrupt (that is, does not send MSI nor drive the legacy interrupt signal).

3.11.10.4 ATTNSTATUS - Attention Status

The Interrupt Status register identifies which channels have generated an interrupt and provides the means for the interrupt service routine to reset the interrupt. This register works in conjunction with the Interrupt Disable bit in the CHANCTRL register of the per-channel registers. (See Section 3.11.11.2.)

Offset from CB_BAR Offset: 04h			
Bit	**Attr**	**Default**	**Description**
31:4	RV	0h	*Reserved*
3:0	RC[a]	0h	**ChanAttn:** Channel Attention Each bit specifies the interrupt status of each DMA channel. Bit 0 represents channel 0, bit 1 represents channel 1, and so forth. These bits are OR'd together to provide the legacy interrupt signal. When a DMA channel generates an interrupt and its CHANCTL Interrupt Disable bit is not set, the hardware sets this bit. The software (interrupt service routine) reads these bits to learn which channels are generating the interrupt, and calls the controlling process(es). A channel that has generated an interrupt can not generate another interrupt until the controlling process clears the channel's Interrupt Disable bit in its CHANCTL register. The hardware automatically clears all ATTNSTATUS bits when software reads this register, writing this register has no effect.

Notes:

a. Reading this register clears all of the bits.

3.11.10.5 CBVER - DMA Engine Version

The DMA Engine version register field indicates the version of the DMA Engine specification that the MCH implements. The most significant 4-bits (range 7:4) are the major version number and the least significant 4-bits (range 3:0) are the minor version number. The MCH implementation for this DMA Engine version is 1.0 encoded as 0001 0000b.

Offset from CB_BAR
Offset: 08h

Bit	Attr	Default	Description
7:4	RO	1h	**MJRVER:** Major Version Specifies Major version of the DMA Engine implementation. Current value is 1h
3:0	RO	2h	**MNRVER:** Minor Version Specifies Minor version of the DMA Engine implementation. Current value is 2h

3.11.10.6 PERPORT_OFFSET - Per-port Offset

For each PCI Express port that has DMA Engine capabilities there is a block of per-port registers in the DMA Engine device's MMIO space. The per-port offset[2] indicates the location of the first set of per-port registers residing in the MMIO space. This register points to the first set and each set contains a pointer to the next set of per-port registers. This provides the means for S/W to locate all of the per-port MMIO register sets.

Offset from CB_BAR
Offset: 0Ah

Bit	Attr	Default	Description
15:2	RO	00C0h	**Fst_PPR_OFF:** Points to the first set of Per-port registers for Write combining. A value of zero means that there are no per-port CB resources. Since the per port offset is 32-bit aligned, the effective address is calculated by concatenating "00C0h" ((bits 8,9 are '1's) with "00b" giving 300h as an offset from CB_BAR and this locates the port priority registers for port 2 in the map.
1:0	RV	0h	*Reserved*

3.11.10.7 INTRDELAY - Interrupt Delay Register

Offset from CB_BAR
Offset: 0Ch

Bit	Attr	Default	Description
15	RO	0	**Interrupt Coalescing Supported** The MCH doe not support interrupt coalescing by delaying interrupt generation.
14	RO	0	*Reserved*
13:0	R/W	0	**Interrupt Delay Time** Specifies the number of microseconds that the chipset delays generation of an interrupt (legacy or MSI) from the time that interrupts are enabled (that is, Master Interrupt Enable bit in the INTRCTRL bit is set). The MCH does not support interrupt delay timer and hence hardwired to 0.

2. Offsets are calculated from the base address of the CB_BAR register defined in Section 3.11.4, "CB_BAR: DMA Engine Base Address Register" on page 241.

The Interrupt Delay Time field specifies how long the chipset will delay the interrupt signal to the CPU in order to coalesce interrupts. The setting of INTRCTRL:Interrupt Interrupt Enable to 1 causes a timer to start for period equal to Interrupt Delay Time (±5%), which masks the interrupt generation until the timer expires. When the timer expires, if any interrupts are pending (that is, INTRCTRL:Interrupt bit is 1), then the chipset generates an interrupt to the CPU. If an interrupt occurs after the time expires, then the chipset immediately generates the interrupt to the CPU. The state of the timer does not effect the value of the bits in the INTRCTRL register.

Note: A change to the Interrupt Delay Time field does not take effect until after the next time Master Interrupt Enable is set (that is, writing INTRDELAY does not modify a timing cycle in progress). When software writes INTRCTRL with Master Interrupt Enable = 1, regardless of the previous value of that bit, the chipset starts the timer (if it is not running) or restarts the timer (if it is already running) using the current value from the Interrupt Delay Time register. Thus, interrupts will be delayed for Interrupt Delay Time microseconds after each write to the INTRCTRL register.

The interrupt delay timer has no effect on the MCH and interrupts will be enabled as soon as the Master Interrupt Enable bit in the INTRCTRL bit is set and there are pending interrupts.

3.11.10.8 CS_STATUS: Chipset Status Register

The CS_STATUS register provides the means for the chipset to report conditions that might be adverse to DMA Engine operation.

Offset from CB_BAR Offset: 0Eh			
Bit	**Attr**	**Default**	**Description**
15:2	RO	0h	*Reserved*
1	RO	0	**MMIO_Rstn:** MMIO Restriction when this bit is zero, it indicates the DMA can access all MMIO space. There is no limitation in the Intel 5000X Chipset MCH address decoding logic.
0	RO	0	**Deg_Mode:** Degraded Mode The Intel 5000X Chipset MCH does not support degraded mode for DMA Engine operation and is hardwired to 0.

3.11.10.9 CHAN_SYSERR_MSK[3:0]: Channel System Error Mask Register

The Channel System Error Mask Register[3] provides selective control to mask errors that may cause SERR, SCI/NMI through BIOS control via ERR[2:0]# pins in the Intel 5000X Chipset MCH. If one of the bits in the CHAN_SYSERR_MSK register is set, then that specific error does not cause error signaling. Software decides which DMA errors are fatal[4] to the system and can set them accordingly. The default value is to disable the system error signalling.

Offset from CB_BAR Offset: 4Ch, 48h, 44h, 40h			
Bit	**Attr**	**Default**	**Description**
31:16	RV	0h	*Reserved*

3. The channel specific system error mask register has been defined in the general debug register area for lack of debug space in the channel registers.
4. An example would be an illegal write address to reserved space (firmware) that causes the system to crash (blue-screen in windows OS)

Offset from CB_BAR Offset: 4Ch, 48h, 44h, 40h			
Bit	**Attr**	**Default**	**Description**
15	RO	1	**Unaffil_err_Msk:** Unaffiliated Error Mask
14	RO	1	**Soft_err_Msk:** Soft Error Mask
13	RWST	1	**int_cfg_err_Msk:** Interrupt Configuration Error Mask
12	RWST	1	**Cmp_addr_err_Msk:** Completion Address Error Mask
11	RWST	1	**Desc_len_err_Msk:** Descriptor Length Error Mask
10	RWST	1	**Desc_ctrl_err_Msk:** Descriptor Control Error Mask
9	RWST	1	**Wr_data_err_Msk:** Write Data Error Mask
8	RWST	1	**Rd_data_err_Msk:** Read Data Error Mask
7	RO	1	**DMA_data_par_err_Mskr:** DMA Data Parity Error Mask
6	RWST	1	**Cdata_par_err_Msk:** Chipset Data Parity Error Mask
5	RWST	1	**Chancmd_err_Msk:** CHANCMD Error Mask
4	RWST	1	**Chn_addr_val_err_Msk:** Chain Address Value Error Mask
3	RWST	1	**Desc_err_Msk:** Descriptor Error Mask
2	RWST	1	**Nxt_desc_addr_err_Msk:** Next Descriptor Address Error Mask
1	RWST	1	**DMA_xfrer_daddr_err_Msk:** DMA Transfer Destination Address Error Mask
0	RWST	1	**DMA_trans_saddr_err_Msk:** DMA Transfer Source Address Error Mask

3.11.11 DMA Channel Specific Registers

As described in Table 3-51, the DMA channel specific information is contained in four locations starting from offset 80h of the CB_BAR register (see Section 3.11.4) and separated by 128B for each channel. They are located in Bus=0, Device=1, Function=1 at the offsets shown in Section 3.11.

3.11.11.1 DMA_COMP[3:0]: DMA Compatibility Register

Offset from CB_BAR Offset: 202h, 182h, 102h, 82h			
Bit	**Attr**	**Default**	**Description**
15:1	RV	0h	Reserved
0	RO	1	**v1_comp:** v1 Compatibility The DMA operation is compatible with software written for version 1 of the *DMA Engine Specification* and hence it is set to '1' by the chipset.

3.11.11.2 CHANCTRL[3:0] - Channel Control Register

The Channel Control register controls the behavior of the DMA channel when specific events occur such as completion or errors.

Offset from CB_BAR Offset: 200h, 180h, 100h, 80h			
Bit	**Attr**	**Default**	**Description**
15:9	RV	0h	Reserved

Offset from CB_BAR Offset: 200h, 180h, 100h, 80h			
Bit	**Attr**	**Default**	**Description**
8	RW	0	**In_use:** In Use This bit indicates whether the DMA channel is in use. The first time this bit is read after it has been cleared, it will return '0' and automatically transition from '0' to '1', reserving the channel for the first consumer that reads this register. All subsequent reads will return '1' indicating that the channel is in use. This bit is cleared by writing a '0' value, thus releasing the channel. A consumer uses this mechanism to atomically claim exclusive ownership of the DMA channel. This should be done before attempting to program any register in the DMA channel register set.
7:6	RV	00	Reserved
5	RW	0	**Desc_addr_snp_ctrl:** Descriptor address snoop control 1: When set, this bit indicates that the descriptors are not in coherent space and should not be snooped. 0: When cleared, the descriptors are in coherent space and each descriptor address must be snooped on the chipset.
4	RW	0	**Err_Int_En:** Error Interrupt Enable This bit enables the DMA channel to generate an interrupt (MSI or legacy) when an error occurs during the DMA transfer. If Any Error Abort Enable (see below) is not set, then unaffiliated errors do not cause an interrupt.
3	RW	0	**AnyErr_Abrt_En:** Any Error Abort Enable This bit enables an abort operation when any error is encountered during the DMA transfer. When the abort occurs, the DMA channel generates an interrupt and a completion update as per the Error Interrupt Enable and Error Completion Enable bits. When this bit is reset, only affiliated errors cause the DMA channel to abort. It has not effect on the Intel 5000X Chipset MCH since only affiliated errors are detected and they automatically cause the channel to abort.
2	RW	0	**Err_Cmp_En:** Error Completion Enable This bit enables a completion write to the address specified in the CHANCMP register upon encountering an error during the DMA transfer. If Any Error Abort is not set, then unaffiliated errors do not cause a completion write.
1	RV	0	Reserved
0	RWC	0	**Intp_Dis:** Interrupt Disable Upon completing a descriptor, if an interrupt is specified for that descriptor and this bit is reset, then the DMA channel generates an interrupt and sets this bit. The choice between MSI or legacy interrupt mode is determined with the MSICTRL register. Legacy interrupts are further gated through intxDisable bit in the PCICMD command register of Section 3.11.1. The controlling process can re-enable this channel's interrupt by writing a one to this bit, which clears the bit. Writing a zero has no effect. Thus, each time this bit is reset, it enables the DMA channel to generate one interrupt.

3.11.11.3 CHANSTS[3:0]: Channel Status Register

Offset from CB_BAR Offset: 204h, 184h, 104h, 84h			
Bit	**Attr**	**Default**	**Description**
63:6	RO	0h	**Cmp_Desc_Addr: Completed Descriptor Address** This register stores the upper address bits (64B aligned) of the last descriptor processed. The DMA channel automatically updates this register when an error or successful completion occurs. For each completion, the DMA channel over-writes the previous value regardless of whether that value has been read.
5	RV	0	Reserved

Offset from CB_BAR Offset: 204h, 184h, 104h, 84h			
Bit	**Attr**	**Default**	**Description**
4	RO	0	**Soft_err**: Soft Error The hardware sets this bit when it detects an error that it was able to correct and resets this bit immediately after it writes the Channel Status to the location specified by CHANCMP. Soft error detection is not supported by Intel 5000X Chipset MCH and is hardwired to 0.
3	RO	0	**Unaffil_err:** Unaffiliated Error The hardware sets this bit when it detects an unaffiliated error and resets this bit immediately after it writes the Channel Status to the location specified by CHANCMP. Unaffiliated error detection is not supported by Intel 5000X Chipset MCH and is hardwired to 0.
2:0	RO	011	**DMA_trans_state:** DMA Transfer Status The DMA engine sets these bits indicating the state of the current DMA transfer. The cause of an abort can be either error during the DMA transfer or invoked by the controlling process via the CHANCMD register defined in Section 3.11.11.5. 000 - Active 001 - Completed, DMA Transfer Done (no hard errors) 010 - Suspended 011 - Halted, operation aborted (refer to Channel Error register for further detail)

The Channel Status Register records the address of the last descriptor completed by the DMA channel. Note that software could read the register any time when the chipset is updating the status of the descriptor. To prevent coherency issues, when reading the mismatched lower/upper 32-bits of this register[5], the chipset must latch the upper DW of the address when the lower DW is read, such that a subsequent read of the upper DW provides the rest of the higher address returned by the prior lower DW read. The chipset will guarantee consistency during the data transfer to the latched register and synchronize concurrent events. The software should issue only 32-bit (DW) reads to the CHANSTS registers. Any other read length less than 32-bits (for example, reading just one byte) will result in indeterministic behavior. Similarly CB BAR MMIO reads of 64-bit in length will also return indeterministic data and is not preferred since the internal data path cannot handle sizes larger than one DW.

3.11.11.4 CHAINADDR[3:0] - Descriptor Chain Address Register

This register is written by the processor or I/O agent to specify the first descriptor to be fetched by the DMA channel. This address will be checked and FERR/NERR*DMA4 error will be set by the DMA engine during execution. i.e when the initial descriptor is fetched. The error will also be logged in the CHANERR.Chn_addr_val_err register field.

Offset from CB_BAR Offset: 20Ch, 18Ch, 10Ch, 8Ch			
Bit	**Attr**	**Default**	**Description**
63:0	RW	0h	**Desc_Addr:** Descriptor Address This 64 bit field marks the address of the first descriptor to be fetched by the DMA channel. The least significant 6 bits must be zero for the address to be valid. ***Note:*** A value of zero to the “Desc_Addr” field is considered illegal and will be considered as a Fatal error.

5. The lower DW could contain the address of the previous descriptor while the upper DW contains the address of the next descriptor and hence the effective address may be garbled due to the asynchronous nature between the software read and the chipset update.

3.11.11.5 CHANCMD[3:0] - DMA Channel Command Register

Offset from CB_BAR Offset: 214h, 194h, 114h, 94h			
Bit	**Attr**	**Default**	**Description**
7:6	RV	0h	Reserved
5	RW	0	**Rst_DMA:** Reset DMA Set this bit to reset the DMA channel. When this bit has been set, the DMA channel terminates pending transactions immediately and releases any buffers that have been allocated. Setting this bit is a last resort to recover the DMA channel from a programming error or other problem such as deadlocks from cache coherency protocol, misprogrammed channels or other bottlenecked traffic scenarios that will help recovery. Execution of this command does not generate an interrupt or generate status. This command causes the DMA channel to return to a known state (Halted). The CDAR register will log the current descriptor address when the Reset DMA is processed by the DMA engine. In this mode, only the CHANSTS.DMA_trans_state register field will be updated to reflect the final state of the DMA engine.
4	RW	0	**Res_DMA:** Resume DMA Set this bit to resume DMA transfer after the channel has been suspended. When this bit has been set, the DMA channel resumes operation by fetching the last descriptor (pointed to by CDAR defined in Section 3.11.11.7) and following the Next Descriptor Address.
3	RW	0	**Abrt_DMA:** Abort DMA Set this bit to abort the current DMA transfer. When this bit has been set, the DMA channel will abort the current DMA transfer if the current DMA transfer is active and sets DMA Transfer Status as “Halted” in channel status register. Otherwise, it just sets DMA Transfer Status as “Halted” in channel status register.
2	RW	0	**Susp_DMA:** Suspend DMA Set this bit to suspend the current DMA transfer. When this bit has been set, the DMA channel halts at current descriptor boundary, sets DMA Transfer Status as “Suspended”, and waits for a Resume DMA or Abort DMA command.
1	RW	0	**Appnd_DMA:** Append DMA Set this bit to append a new descriptor or a chain of descriptors. When this bit has been set, the DMA channel checks if it is at the last descriptor or finished the last descriptor. If it is, the DMA channel fetches the last descriptor. If it is not at the last descriptor, the DMA engine ignores this bit.
0	RW	0	**Strt_DMA:** Start DMA Set this bit to initiate a new DMA transfer. When this bit has been set, the DMA channel begins to fetch a new descriptor at the address of the CHAINADDR register. The processor or I/O device must update the CHAINADDR register before it sets this bit.

This register is written by the controlling process to invoke DMA operation. Setting more than one of these bits with the same write operation will result in an Fatal error (affiliated). It is also the responsibility of the software to program the correct DMA command into this register depending on the operational needs since the DMA engine will only process the last written[6] command after it has completed servicing a descriptor. The Start DMA bit will not progress if there any hard/affiliated errors (Fatal) recorded in the CHANERR register until the software clears[7] it. (See Section 3.11.11.8.) If the Start DMA is set and the CHANERR register is not cleared for affiliated errors, then the CHANERR.chancmd_error bit will be asserted and the engine will be in halted status.

6. If multiple writes happen to this register when the DMA channel is busy, prior writes will be overwritten and only the last command that remains, when the engine scans the CHANCMD register, will be serviced if it is valid. The exception to this is the Append DMA command which is processed after the earlier command is completed.
7. The erroneous CHANCMD register is first cleared by Software and then the CHANERR.chan_cmd error register. field is reset by Software.

3.11.11.6 CHANCMP[3:0]: Channel Completion Address Register

This register specifies the address where the DMA channel writes the completion status upon completion or an error condition that is, it writes the contents of the CHANSTS register to the destination as pointed by the CHANCMP register. The channel completion write is sent after the DMA engine has updated all its internal states following the transfer. The CHANCMP address will be checked and an error will be set by the DMA engine during execution if it is found illegal. The CHANCMP error will be recorded in FERR/NERR*DMA12 register fields and also in the CHANERR.Cmp_addr_err fields.

Offset from CB_BAR
Offset: 218h, 198h, 118h, 98h

Bit	Attr	Default	Description
63:3	RW	0h	**Chan_cmp_addr:** Channel Completion Address This 64-bit field specifies the address where the DMA engine writes the completion status (CHANSTS). This address can fall within system memory or memory-mapped I/O space but should be 8-byte aligned[a].
2:0	RV	0h	Reserved

Notes:

a. This prevents straddling across cache lines since the contents of the CHANSTS registers are 8-bytes wide.

3.11.11.7 CDAR[3:0]: Current Descriptor Address Register

The software should issue only 32-bit (DW) reads to the CDAR register. The upper DW is latched when the lower DW is read by software and the chipset will ensure the data consistency for the split reads. See description in Section 3.11.11.3

Offset from CB_BAR
Offset: 220h, 1A0h, 120h, A0h

Bit	Attr	Default	Description
63:0	RO	0h	**Cur_desc_addr:** Current Descriptor Address This 64-bit field denotes the address of the current Descriptor (that is, the descriptor that is currently being processed by the DMA channel).

3.11.11.8 CHANERR[3:0] - Channel Error Register

The Channel Error Register records the error conditions occurring within a given DMA channel. All errors in this register from bits 14:0 are treated as affiliated errors while bits 15:14 are read only as Intel 5000X Chipset MCH does not detect unaffiliated or soft errors. For all Fatal errors (affiliated), the DMA engine will not resume from the Halted state (or start a new operation) until the software handles the source of the error, resetting the respective registers and then clearing the associated error bits in the CHANERR register. If there are affiliated errors in CHANERR and the Start DMA is asserted, then the CHANERR.chancmd_err bit will be set.

Offset from CB_BAR
Offset: 228h, 1A8h, 128h, A8h

Bit	Attr	Default	Description
31:16	RV	0h	*Reserved*
15	RO	0	**Unaffil_err: Unaffiliated Error** The Intel 5000X Chipset MCH does not detect unaffiliated Errors.

Offset from CB_BAR Offset: 228h, 1A8h, 128h, A8h			
Bit	**Attr**	**Default**	**Description**
14	RO	0	**Soft_err: Soft Error** The Intel 5000X Chipset MCH does not implement soft error detection.
13	RWCST	0	**int_cfg_err:** Interrupt Configuration Error The DMA channel sets this bit indicating that the interrupt registers were not configured properly when the DMA channel attempted to generate an interrupt. See Section 3.11.7, Section 3.11.1 and Section 3.11.8.5. MSICTRL.MSIEN = 1, DMACTRL.MSICBEN = 0, PCICMD.INTx Disable = 1 MSICTRL.MSIEN = 0, DMACTRLMSICBEN = 1, PCICMD.INTx Disable = 1 MSICTRL.MSIEN = 0, DMACTRLMSICBEN = 0, PCICMD.INTx Disable = 1
12	RWCST	0	**Cmp_addr_err:** Completion Address Error The DMA channel sets this bit indicating that the completion address register was configured to an illegal address or has not been configured.
11	RWCST	0	**Desc_len_err:** Descriptor Length Error The DMA channel sets this bit indicating that the current transfer has an illegal length field value. When this bit has been set, the address of the failed descriptor and the channel returns to the Halted state, is recorded in the Channel Status register.
10	RWCST	0	**Desc_ctrl_err:** Descriptor Control Error The DMA channel sets this bit indicating that the current transfer has an illegal control field value. When this bit has been set, the address of the failed descriptor and the channel returns to the Halted state, is recorded in the Channel Status register.
9	RWCST	0	**Wr_data_err:** Write Data Error The DMA channel sets this bit indicating that the current transfer has encountered an error while writing the destination data. When this bit has been set, the address of the failed descriptor when the channel returns to the Halted state, is recorded in the Channel Status register.
8	RWCST	0	**Rd_data_err:** Read Data Error The DMA channel sets this bit indicating that the current transfer has encountered an error while accessing the source data. When this bit has been set, the address of the failed descriptor when the channel returns to the Halted state, is recorded in the Channel Status register.
7	RO	0	**DMA_data_par_err:** DMA Data Parity Error The DMA channel sets this bit indicating that the current transfer has encountered a parity error reported by the DMA engine. The Intel 5000X Chipset MCH does not implement DMA parity error detection.
6	RWCST	0	**Cdata_par_err:** Chipset Data Parity Error The DMA channel sets this bit indicating that the current transfer has encountered a parity error reported by the chipset. When this bit has been set, the address of the failed descriptor when the channel returns to the Halted state, is recorded in the Channel Status register. In the case of Source data (read) error, the "**Rd_data_err**" field is also set. In the case of Destination (write) data error, the "**Wr_data_err**" field is also set
5	RWCST	0	**Chancmd_err:** CHANCMD Error The DMA channel sets this bit indicating that a write to the CHANCMD register contained an invalid value (for example, more than one command bit set). Software should take necessary action to clear the CHANCMD register first before resetting this register field.
4	RWCST	0	**Chn_addr_val_err:** Chain Address Value Error The DMA channel sets this bit indicating that the CHAINADDR register has an illegal address including an alignment error (not on a 64-byte boundary). This address will be checked and set by the DMA engine during execution. i.e when the initial descriptor is fetched.

Offset from CB_BAR Offset: 228h, 1A8h, 128h, A8h			
Bit	**Attr**	**Default**	**Description**
14	RO	0	**Soft_err: Soft Error** The Intel 5000X Chipset MCH does not implement soft error detection.
13	RWCST	0	**int_cfg_err:** Interrupt Configuration Error The DMA channel sets this bit indicating that the interrupt registers were not configured properly when the DMA channel attempted to generate an interrupt. See Section 3.11.7, Section 3.11.1 and Section 3.11.8.5. MSICTRL.MSIEN = 1, DMACTRL.MSICBEN = 0, PCICMD.INTx Disable = 1 MSICTRL.MSIEN = 0, DMACTRLMSICBEN = 1, PCICMD.INTx Disable = 1 MSICTRL.MSIEN = 0, DMACTRLMSICBEN = 0, PCICMD.INTx Disable = 1
12	RWCST	0	**Cmp_addr_err:** Completion Address Error The DMA channel sets this bit indicating that the completion address register was configured to an illegal address or has not been configured.
11	RWCST	0	**Desc_len_err:** Descriptor Length Error The DMA channel sets this bit indicating that the current transfer has an illegal length field value. When this bit has been set, the address of the failed descriptor and the channel returns to the Halted state, is recorded in the Channel Status register.
10	RWCST	0	**Desc_ctrl_err:** Descriptor Control Error The DMA channel sets this bit indicating that the current transfer has an illegal control field value. When this bit has been set, the address of the failed descriptor and the channel returns to the Halted state, is recorded in the Channel Status register.
9	RWCST	0	**Wr_data_err:** Write Data Error The DMA channel sets this bit indicating that the current transfer has encountered an error while writing the destination data. When this bit has been set, the address of the failed descriptor when the channel returns to the Halted state, is recorded in the Channel Status register.
8	RWCST	0	**Rd_data_err:** Read Data Error The DMA channel sets this bit indicating that the current transfer has encountered an error while accessing the source data. When this bit has been set, the address of the failed descriptor when the channel returns to the Halted state, is recorded in the Channel Status register.
7	RO	0	**DMA_data_par_err:** DMA Data Parity Error The DMA channel sets this bit indicating that the current transfer has encountered a parity error reported by the DMA engine. The Intel 5000X Chipset MCH does not implement DMA parity error detection.
6	RWCST	0	**Cdata_par_err:** Chipset Data Parity Error The DMA channel sets this bit indicating that the current transfer has encountered a parity error reported by the chipset. When this bit has been set, the address of the failed descriptor when the channel returns to the Halted state, is recorded in the Channel Status register. In the case of Source data (read) error, the "**Rd_data_err**" field is also set. In the case of Destination (write) data error, the "**Wr_data_err**" field is also set
5	RWCST	0	**Chancmd_err:** CHANCMD Error The DMA channel sets this bit indicating that a write to the CHANCMD register contained an invalid value (for example, more than one command bit set). Software should take necessary action to clear the CHANCMD register first before resetting this register field.
4	RWCST	0	**Chn_addr_val_err:** Chain Address Value Error The DMA channel sets this bit indicating that the CHAINADDR register has an illegal address including an alignment error (not on a 64-byte boundary). This address will be checked and set by the DMA engine during execution. i.e when the initial descriptor is fetched.

Offset from CB_BAR
Offset: 228h, 1A8h, 128h, A8h

Bit	Attr	Default	Description
14	RO	0	**Soft_err: Soft Error** The Intel 5000X Chipset MCH does not implement soft error detection.
13	RWCST	0	**int_cfg_err:** Interrupt Configuration Error The DMA channel sets this bit indicating that the interrupt registers were not configured properly when the DMA channel attempted to generate an interrupt. See Section 3.11.7, Section 3.11.1 and Section 3.11.8.5. MSICTRL.MSIEN = 1, DMACTRL.MSICBEN = 0, PCICMD.INTx Disable = 1 MSICTRL.MSIEN = 0, DMACTRLMSICBEN = 1, PCICMD.INTx Disable = 1 MSICTRL.MSIEN = 0, DMACTRLMSICBEN = 0, PCICMD.INTx Disable = 1
12	RWCST	0	**Cmp_addr_err:** Completion Address Error The DMA channel sets this bit indicating that the completion address register was configured to an illegal address or has not been configured.
11	RWCST	0	**Desc_len_err:** Descriptor Length Error The DMA channel sets this bit indicating that the current transfer has an illegal length field value. When this bit has been set, the address of the failed descriptor and the channel returns to the Halted state, is recorded in the Channel Status register.
10	RWCST	0	**Desc_ctrl_err:** Descriptor Control Error The DMA channel sets this bit indicating that the current transfer has an illegal control field value. When this bit has been set, the address of the failed descriptor and the channel returns to the Halted state, is recorded in the Channel Status register.
9	RWCST	0	**Wr_data_err:** Write Data Error The DMA channel sets this bit indicating that the current transfer has encountered an error while writing the destination data. When this bit has been set, the address of the failed descriptor when the channel returns to the Halted state, is recorded in the Channel Status register.
8	RWCST	0	**Rd_data_err:** Read Data Error The DMA channel sets this bit indicating that the current transfer has encountered an error while accessing the source data. When this bit has been set, the address of the failed descriptor when the channel returns to the Halted state, is recorded in the Channel Status register.
7	RO	0	**DMA_data_par_err:** DMA Data Parity Error The DMA channel sets this bit indicating that the current transfer has encountered a parity error reported by the DMA engine. The Intel 5000X Chipset MCH does not implement DMA parity error detection.
6	RWCST	0	**Cdata_par_err:** Chipset Data Parity Error The DMA channel sets this bit indicating that the current transfer has encountered a parity error reported by the chipset. When this bit has been set, the address of the failed descriptor when the channel returns to the Halted state, is recorded in the Channel Status register. In the case of Source data (read) error, the "**Rd_data_err**" field is also set. In the case of Destination (write) data error, the "**Wr_data_err**" field is also set
5	RWCST	0	**Chancmd_err:** CHANCMD Error The DMA channel sets this bit indicating that a write to the CHANCMD register contained an invalid value (for example, more than one command bit set). Software should take necessary action to clear the CHANCMD register first before resetting this register field.
4	RWCST	0	**Chn_addr_val_err:** Chain Address Value Error The DMA channel sets this bit indicating that the CHAINADDR register has an illegal address including an alignment error (not on a 64-byte boundary). This address will be checked and set by the DMA engine during execution. i.e when the initial descriptor is fetched.

Offset from CB_BAR Offset: 228h, 1A8h, 128h, A8h			
Bit	**Attr**	**Default**	**Description**
14	RO	0	**Soft_err: Soft Error** The Intel 5000X Chipset MCH does not implement soft error detection.
13	RWCST	0	**int_cfg_err:** Interrupt Configuration Error The DMA channel sets this bit indicating that the interrupt registers were not configured properly when the DMA channel attempted to generate an interrupt. See Section 3.11.7, Section 3.11.1 and Section 3.11.8.5. MSICTRL.MSIEN = 1, DMACTRL.MSICBEN = 0, PCICMD.INTx Disable = 1 MSICTRL.MSIEN = 0, DMACTRLMSICBEN = 1, PCICMD.INTx Disable = 1 MSICTRL.MSIEN = 0, DMACTRLMSICBEN = 0, PCICMD.INTx Disable = 1
12	RWCST	0	**Cmp_addr_err:** Completion Address Error The DMA channel sets this bit indicating that the completion address register was configured to an illegal address or has not been configured.
11	RWCST	0	**Desc_len_err:** Descriptor Length Error The DMA channel sets this bit indicating that the current transfer has an illegal length field value. When this bit has been set, the address of the failed descriptor and the channel returns to the Halted state, is recorded in the Channel Status register.
10	RWCST	0	**Desc_ctrl_err:** Descriptor Control Error The DMA channel sets this bit indicating that the current transfer has an illegal control field value. When this bit has been set, the address of the failed descriptor and the channel returns to the Halted state, is recorded in the Channel Status register.
9	RWCST	0	**Wr_data_err:** Write Data Error The DMA channel sets this bit indicating that the current transfer has encountered an error while writing the destination data. When this bit has been set, the address of the failed descriptor when the channel returns to the Halted state, is recorded in the Channel Status register.
8	RWCST	0	**Rd_data_err:** Read Data Error The DMA channel sets this bit indicating that the current transfer has encountered an error while accessing the source data. When this bit has been set, the address of the failed descriptor when the channel returns to the Halted state, is recorded in the Channel Status register.
7	RO	0	**DMA_data_par_err:** DMA Data Parity Error The DMA channel sets this bit indicating that the current transfer has encountered a parity error reported by the DMA engine. The Intel 5000X Chipset MCH does not implement DMA parity error detection.
6	RWCST	0	**Cdata_par_err:** Chipset Data Parity Error The DMA channel sets this bit indicating that the current transfer has encountered a parity error reported by the chipset. When this bit has been set, the address of the failed descriptor when the channel returns to the Halted state, is recorded in the Channel Status register. In the case of Source data (read) error, the "**Rd_data_err**" field is also set. In the case of Destination (write) data error, the "**Wr_data_err**" field is also set
5	RWCST	0	**Chancmd_err:** CHANCMD Error The DMA channel sets this bit indicating that a write to the CHANCMD register contained an invalid value (for example, more than one command bit set). Software should take necessary action to clear the CHANCMD register first before resetting this register field.
4	RWCST	0	**Chn_addr_val_err:** Chain Address Value Error The DMA channel sets this bit indicating that the CHAINADDR register has an illegal address including an alignment error (not on a 64-byte boundary). This address will be checked and set by the DMA engine during execution. i.e when the initial descriptor is fetched.

Offset from CB_BAR
Offset: 228h, 1A8h, 128h, A8h

Bit	Attr	Default	Description
14	RO	0	**Soft_err: Soft Error** The Intel 5000X Chipset MCH does not implement soft error detection.
13	RWCST	0	**int_cfg_err:** Interrupt Configuration Error The DMA channel sets this bit indicating that the interrupt registers were not configured properly when the DMA channel attempted to generate an interrupt. See Section 3.11.7, Section 3.11.1 and Section 3.11.8.5. MSICTRL.MSIEN = 1, DMACTRL.MSICBEN = 0, PCICMD.INTx Disable = 1 MSICTRL.MSIEN = 0, DMACTRLMSICBEN = 1, PCICMD.INTx Disable = 1 MSICTRL.MSIEN = 0, DMACTRLMSICBEN = 0, PCICMD.INTx Disable = 1
12	RWCST	0	**Cmp_addr_err:** Completion Address Error The DMA channel sets this bit indicating that the completion address register was configured to an illegal address or has not been configured.
11	RWCST	0	**Desc_len_err:** Descriptor Length Error The DMA channel sets this bit indicating that the current transfer has an illegal length field value. When this bit has been set, the address of the failed descriptor and the channel returns to the Halted state, is recorded in the Channel Status register.
10	RWCST	0	**Desc_ctrl_err:** Descriptor Control Error The DMA channel sets this bit indicating that the current transfer has an illegal control field value. When this bit has been set, the address of the failed descriptor and the channel returns to the Halted state, is recorded in the Channel Status register.
9	RWCST	0	**Wr_data_err:** Write Data Error The DMA channel sets this bit indicating that the current transfer has encountered an error while writing the destination data. When this bit has been set, the address of the failed descriptor when the channel returns to the Halted state, is recorded in the Channel Status register.
8	RWCST	0	**Rd_data_err:** Read Data Error The DMA channel sets this bit indicating that the current transfer has encountered an error while accessing the source data. When this bit has been set, the address of the failed descriptor when the channel returns to the Halted state, is recorded in the Channel Status register.
7	RO	0	**DMA_data_par_err:** DMA Data Parity Error The DMA channel sets this bit indicating that the current transfer has encountered a parity error reported by the DMA engine. The Intel 5000X Chipset MCH does not implement DMA parity error detection.
6	RWCST	0	**Cdata_par_err:** Chipset Data Parity Error The DMA channel sets this bit indicating that the current transfer has encountered a parity error reported by the chipset. When this bit has been set, the address of the failed descriptor when the channel returns to the Halted state, is recorded in the Channel Status register. In the case of Source data (read) error, the "**Rd_data_err**" field is also set. In the case of Destination (write) data error, the "**Wr_data_err**" field is also set
5	RWCST	0	**Chancmd_err:** CHANCMD Error The DMA channel sets this bit indicating that a write to the CHANCMD register contained an invalid value (for example, more than one command bit set). Software should take necessary action to clear the CHANCMD register first before resetting this register field.
4	RWCST	0	**Chn_addr_val_err:** Chain Address Value Error The DMA channel sets this bit indicating that the CHAINADDR register has an illegal address including an alignment error (not on a 64-byte boundary). This address will be checked and set by the DMA engine during execution. i.e when the initial descriptor is fetched.

Offset from CB_BAR Offset: 228h, 1A8h, 128h, A8h			
Bit	**Attr**	**Default**	**Description**
14	RO	0	**Soft_err: Soft Error** The Intel 5000X Chipset MCH does not implement soft error detection.
13	RWCST	0	**int_cfg_err:** Interrupt Configuration Error The DMA channel sets this bit indicating that the interrupt registers were not configured properly when the DMA channel attempted to generate an interrupt. See Section 3.11.7, Section 3.11.1 and Section 3.11.8.5. MSICTRL.MSIEN = 1, DMACTRL.MSICBEN = 0, PCICMD.INTx Disable = 1 MSICTRL.MSIEN = 0, DMACTRLMSICBEN = 1, PCICMD.INTx Disable = 1 MSICTRL.MSIEN = 0, DMACTRLMSICBEN = 0, PCICMD.INTx Disable = 1
12	RWCST	0	**Cmp_addr_err:** Completion Address Error The DMA channel sets this bit indicating that the completion address register was configured to an illegal address or has not been configured.
11	RWCST	0	**Desc_len_err:** Descriptor Length Error The DMA channel sets this bit indicating that the current transfer has an illegal length field value. When this bit has been set, the address of the failed descriptor and the channel returns to the Halted state, is recorded in the Channel Status register.
10	RWCST	0	**Desc_ctrl_err:** Descriptor Control Error The DMA channel sets this bit indicating that the current transfer has an illegal control field value. When this bit has been set, the address of the failed descriptor and the channel returns to the Halted state, is recorded in the Channel Status register.
9	RWCST	0	**Wr_data_err:** Write Data Error The DMA channel sets this bit indicating that the current transfer has encountered an error while writing the destination data. When this bit has been set, the address of the failed descriptor when the channel returns to the Halted state, is recorded in the Channel Status register.
8	RWCST	0	**Rd_data_err:** Read Data Error The DMA channel sets this bit indicating that the current transfer has encountered an error while accessing the source data. When this bit has been set, the address of the failed descriptor when the channel returns to the Halted state, is recorded in the Channel Status register.
7	RO	0	**DMA_data_par_err:** DMA Data Parity Error The DMA channel sets this bit indicating that the current transfer has encountered a parity error reported by the DMA engine. The Intel 5000X Chipset MCH does not implement DMA parity error detection.
6	RWCST	0	**Cdata_par_err:** Chipset Data Parity Error The DMA channel sets this bit indicating that the current transfer has encountered a parity error reported by the chipset. When this bit has been set, the address of the failed descriptor when the channel returns to the Halted state, is recorded in the Channel Status register. In the case of Source data (read) error, the "**Rd_data_err**" field is also set. In the case of Destination (write) data error, the "**Wr_data_err**" field is also set
5	RWCST	0	**Chancmd_err:** CHANCMD Error The DMA channel sets this bit indicating that a write to the CHANCMD register contained an invalid value (for example, more than one command bit set). Software should take necessary action to clear the CHANCMD register first before resetting this register field.
4	RWCST	0	**Chn_addr_val_err:** Chain Address Value Error The DMA channel sets this bit indicating that the CHAINADDR register has an illegal address including an alignment error (not on a 64-byte boundary). This address will be checked and set by the DMA engine during execution. i.e when the initial descriptor is fetched.

Offset from CB_BAR
Offset: 228h, 1A8h, 128h, A8h

Bit	Attr	Default	Description
14	RO	0	**Soft_err: Soft Error** The Intel 5000X Chipset MCH does not implement soft error detection.
13	RWCST	0	**int_cfg_err:** Interrupt Configuration Error The DMA channel sets this bit indicating that the interrupt registers were not configured properly when the DMA channel attempted to generate an interrupt. See Section 3.11.7, Section 3.11.1 and Section 3.11.8.5. MSICTRL.MSIEN = 1, DMACTRL.MSICBEN = 0, PCICMD.INTx Disable = 1 MSICTRL.MSIEN = 0, DMACTRLMSICBEN = 1, PCICMD.INTx Disable = 1 MSICTRL.MSIEN = 0, DMACTRLMSICBEN = 0, PCICMD.INTx Disable = 1
12	RWCST	0	**Cmp_addr_err:** Completion Address Error The DMA channel sets this bit indicating that the completion address register was configured to an illegal address or has not been configured.
11	RWCST	0	**Desc_len_err:** Descriptor Length Error The DMA channel sets this bit indicating that the current transfer has an illegal length field value. When this bit has been set, the address of the failed descriptor and the channel returns to the Halted state, is recorded in the Channel Status register.
10	RWCST	0	**Desc_ctrl_err:** Descriptor Control Error The DMA channel sets this bit indicating that the current transfer has an illegal control field value. When this bit has been set, the address of the failed descriptor and the channel returns to the Halted state, is recorded in the Channel Status register.
9	RWCST	0	**Wr_data_err:** Write Data Error The DMA channel sets this bit indicating that the current transfer has encountered an error while writing the destination data. When this bit has been set, the address of the failed descriptor when the channel returns to the Halted state, is recorded in the Channel Status register.
8	RWCST	0	**Rd_data_err:** Read Data Error The DMA channel sets this bit indicating that the current transfer has encountered an error while accessing the source data. When this bit has been set, the address of the failed descriptor when the channel returns to the Halted state, is recorded in the Channel Status register.
7	RO	0	**DMA_data_par_err:** DMA Data Parity Error The DMA channel sets this bit indicating that the current transfer has encountered a parity error reported by the DMA engine. The Intel 5000X Chipset MCH does not implement DMA parity error detection.
6	RWCST	0	**Cdata_par_err:** Chipset Data Parity Error The DMA channel sets this bit indicating that the current transfer has encountered a parity error reported by the chipset. When this bit has been set, the address of the failed descriptor when the channel returns to the Halted state, is recorded in the Channel Status register. In the case of Source data (read) error, the "**Rd_data_err**" field is also set. In the case of Destination (write) data error, the "**Wr_data_err**" field is also set
5	RWCST	0	**Chancmd_err:** CHANCMD Error The DMA channel sets this bit indicating that a write to the CHANCMD register contained an invalid value (for example, more than one command bit set). Software should take necessary action to clear the CHANCMD register first before resetting this register field.
4	RWCST	0	**Chn_addr_val_err:** Chain Address Value Error The DMA channel sets this bit indicating that the CHAINADDR register has an illegal address including an alignment error (not on a 64-byte boundary). This address will be checked and set by the DMA engine during execution. i.e when the initial descriptor is fetched.

Offset from CB_BAR Offset: 228h, 1A8h, 128h, A8h			
Bit	**Attr**	**Default**	**Description**
14	RO	0	**Soft_err: Soft Error** The Intel 5000X Chipset MCH does not implement soft error detection.
13	RWCST	0	**int_cfg_err:** Interrupt Configuration Error The DMA channel sets this bit indicating that the interrupt registers were not configured properly when the DMA channel attempted to generate an interrupt. See Section 3.11.7, Section 3.11.1 and Section 3.11.8.5. MSICTRL.MSIEN = 1, DMACTRL.MSICBEN = 0, PCICMD.INTx Disable = 1 MSICTRL.MSIEN = 0, DMACTRLMSICBEN = 1, PCICMD.INTx Disable = 1 MSICTRL.MSIEN = 0, DMACTRLMSICBEN = 0, PCICMD.INTx Disable = 1
12	RWCST	0	**Cmp_addr_err:** Completion Address Error The DMA channel sets this bit indicating that the completion address register was configured to an illegal address or has not been configured.
11	RWCST	0	**Desc_len_err:** Descriptor Length Error The DMA channel sets this bit indicating that the current transfer has an illegal length field value. When this bit has been set, the address of the failed descriptor and the channel returns to the Halted state, is recorded in the Channel Status register.
10	RWCST	0	**Desc_ctrl_err:** Descriptor Control Error The DMA channel sets this bit indicating that the current transfer has an illegal control field value. When this bit has been set, the address of the failed descriptor and the channel returns to the Halted state, is recorded in the Channel Status register.
9	RWCST	0	**Wr_data_err:** Write Data Error The DMA channel sets this bit indicating that the current transfer has encountered an error while writing the destination data. When this bit has been set, the address of the failed descriptor when the channel returns to the Halted state, is recorded in the Channel Status register.
8	RWCST	0	**Rd_data_err:** Read Data Error The DMA channel sets this bit indicating that the current transfer has encountered an error while accessing the source data. When this bit has been set, the address of the failed descriptor when the channel returns to the Halted state, is recorded in the Channel Status register.
7	RO	0	**DMA_data_par_err:** DMA Data Parity Error The DMA channel sets this bit indicating that the current transfer has encountered a parity error reported by the DMA engine. The Intel 5000X Chipset MCH does not implement DMA parity error detection.
6	RWCST	0	**Cdata_par_err:** Chipset Data Parity Error The DMA channel sets this bit indicating that the current transfer has encountered a parity error reported by the chipset. When this bit has been set, the address of the failed descriptor when the channel returns to the Halted state, is recorded in the Channel Status register. In the case of Source data (read) error, the "**Rd_data_err**" field is also set. In the case of Destination (write) data error, the "**Wr_data_err**" field is also set
5	RWCST	0	**Chancmd_err:** CHANCMD Error The DMA channel sets this bit indicating that a write to the CHANCMD register contained an invalid value (for example, more than one command bit set). Software should take necessary action to clear the CHANCMD register first before resetting this register field.
4	RWCST	0	**Chn_addr_val_err:** Chain Address Value Error The DMA channel sets this bit indicating that the CHAINADDR register has an illegal address including an alignment error (not on a 64-byte boundary). This address will be checked and set by the DMA engine during execution. i.e when the initial descriptor is fetched.

Offset from CB_BAR
Offset: 228h, 1A8h, 128h, A8h

Bit	Attr	Default	Description
14	RO	0	**Soft_err: Soft Error** The Intel 5000X Chipset MCH does not implement soft error detection.
13	RWCST	0	**int_cfg_err:** Interrupt Configuration Error The DMA channel sets this bit indicating that the interrupt registers were not configured properly when the DMA channel attempted to generate an interrupt. See Section 3.11.7, Section 3.11.1 and Section 3.11.8.5. MSICTRL.MSIEN = 1, DMACTRL.MSICBEN = 0, PCICMD.INTx Disable = 1 MSICTRL.MSIEN = 0, DMACTRLMSICBEN = 1, PCICMD.INTx Disable = 1 MSICTRL.MSIEN = 0, DMACTRLMSICBEN = 0, PCICMD.INTx Disable = 1
12	RWCST	0	**Cmp_addr_err:** Completion Address Error The DMA channel sets this bit indicating that the completion address register was configured to an illegal address or has not been configured.
11	RWCST	0	**Desc_len_err:** Descriptor Length Error The DMA channel sets this bit indicating that the current transfer has an illegal length field value. When this bit has been set, the address of the failed descriptor and the channel returns to the Halted state, is recorded in the Channel Status register.
10	RWCST	0	**Desc_ctrl_err:** Descriptor Control Error The DMA channel sets this bit indicating that the current transfer has an illegal control field value. When this bit has been set, the address of the failed descriptor and the channel returns to the Halted state, is recorded in the Channel Status register.
9	RWCST	0	**Wr_data_err:** Write Data Error The DMA channel sets this bit indicating that the current transfer has encountered an error while writing the destination data. When this bit has been set, the address of the failed descriptor when the channel returns to the Halted state, is recorded in the Channel Status register.
8	RWCST	0	**Rd_data_err:** Read Data Error The DMA channel sets this bit indicating that the current transfer has encountered an error while accessing the source data. When this bit has been set, the address of the failed descriptor when the channel returns to the Halted state, is recorded in the Channel Status register.
7	RO	0	**DMA_data_par_err:** DMA Data Parity Error The DMA channel sets this bit indicating that the current transfer has encountered a parity error reported by the DMA engine. The Intel 5000X Chipset MCH does not implement DMA parity error detection.
6	RWCST	0	**Cdata_par_err:** Chipset Data Parity Error The DMA channel sets this bit indicating that the current transfer has encountered a parity error reported by the chipset. When this bit has been set, the address of the failed descriptor when the channel returns to the Halted state, is recorded in the Channel Status register. In the case of Source data (read) error, the "**Rd_data_err**" field is also set. In the case of Destination (write) data error, the "**Wr_data_err**" field is also set
5	RWCST	0	**Chancmd_err:** CHANCMD Error The DMA channel sets this bit indicating that a write to the CHANCMD register contained an invalid value (for example, more than one command bit set). Software should take necessary action to clear the CHANCMD register first before resetting this register field.
4	RWCST	0	**Chn_addr_val_err:** Chain Address Value Error The DMA channel sets this bit indicating that the CHAINADDR register has an illegal address including an alignment error (not on a 64-byte boundary). This address will be checked and set by the DMA engine during execution. i.e when the initial descriptor is fetched.

Offset from CB_BAR Offset: 228h, 1A8h, 128h, A8h			
Bit	**Attr**	**Default**	**Description**
14	RO	0	**Soft_err: Soft Error** The Intel 5000X Chipset MCH does not implement soft error detection.
13	RWCST	0	**int_cfg_err:** Interrupt Configuration Error The DMA channel sets this bit indicating that the interrupt registers were not configured properly when the DMA channel attempted to generate an interrupt. See Section 3.11.7, Section 3.11.1 and Section 3.11.8.5. MSICTRL.MSIEN = 1, DMACTRL.MSICBEN = 0, PCICMD.INTx Disable = 1 MSICTRL.MSIEN = 0, DMACTRLMSICBEN = 1, PCICMD.INTx Disable = 1 MSICTRL.MSIEN = 0, DMACTRLMSICBEN = 0, PCICMD.INTx Disable = 1
12	RWCST	0	**Cmp_addr_err:** Completion Address Error The DMA channel sets this bit indicating that the completion address register was configured to an illegal address or has not been configured.
11	RWCST	0	**Desc_len_err:** Descriptor Length Error The DMA channel sets this bit indicating that the current transfer has an illegal length field value. When this bit has been set, the address of the failed descriptor and the channel returns to the Halted state, is recorded in the Channel Status register.
10	RWCST	0	**Desc_ctrl_err:** Descriptor Control Error The DMA channel sets this bit indicating that the current transfer has an illegal control field value. When this bit has been set, the address of the failed descriptor and the channel returns to the Halted state, is recorded in the Channel Status register.
9	RWCST	0	**Wr_data_err:** Write Data Error The DMA channel sets this bit indicating that the current transfer has encountered an error while writing the destination data. When this bit has been set, the address of the failed descriptor when the channel returns to the Halted state, is recorded in the Channel Status register.
8	RWCST	0	**Rd_data_err:** Read Data Error The DMA channel sets this bit indicating that the current transfer has encountered an error while accessing the source data. When this bit has been set, the address of the failed descriptor when the channel returns to the Halted state, is recorded in the Channel Status register.
7	RO	0	**DMA_data_par_err:** DMA Data Parity Error The DMA channel sets this bit indicating that the current transfer has encountered a parity error reported by the DMA engine. The Intel 5000X Chipset MCH does not implement DMA parity error detection.
6	RWCST	0	**Cdata_par_err:** Chipset Data Parity Error The DMA channel sets this bit indicating that the current transfer has encountered a parity error reported by the chipset. When this bit has been set, the address of the failed descriptor when the channel returns to the Halted state, is recorded in the Channel Status register. In the case of Source data (read) error, the "**Rd_data_err**" field is also set. In the case of Destination (write) data error, the "**Wr_data_err**" field is also set
5	RWCST	0	**Chancmd_err:** CHANCMD Error The DMA channel sets this bit indicating that a write to the CHANCMD register contained an invalid value (for example, more than one command bit set). Software should take necessary action to clear the CHANCMD register first before resetting this register field.
4	RWCST	0	**Chn_addr_val_err:** Chain Address Value Error The DMA channel sets this bit indicating that the CHAINADDR register has an illegal address including an alignment error (not on a 64-byte boundary). This address will be checked and set by the DMA engine during execution. i.e when the initial descriptor is fetched.

Offset from CB_BAR
Offset: 228h, 1A8h, 128h, A8h

Bit	Attr	Default	Description
14	RO	0	**Soft_err: Soft Error** The Intel 5000X Chipset MCH does not implement soft error detection.
13	RWCST	0	**int_cfg_err:** Interrupt Configuration Error The DMA channel sets this bit indicating that the interrupt registers were not configured properly when the DMA channel attempted to generate an interrupt. See Section 3.11.7, Section 3.11.1 and Section 3.11.8.5. MSICTRL.MSIEN = 1, DMACTRL.MSICBEN = 0, PCICMD.INTx Disable = 1 MSICTRL.MSIEN = 0, DMACTRLMSICBEN = 1, PCICMD.INTx Disable = 1 MSICTRL.MSIEN = 0, DMACTRLMSICBEN = 0, PCICMD.INTx Disable = 1
12	RWCST	0	**Cmp_addr_err:** Completion Address Error The DMA channel sets this bit indicating that the completion address register was configured to an illegal address or has not been configured.
11	RWCST	0	**Desc_len_err:** Descriptor Length Error The DMA channel sets this bit indicating that the current transfer has an illegal length field value. When this bit has been set, the address of the failed descriptor and the channel returns to the Halted state, is recorded in the Channel Status register.
10	RWCST	0	**Desc_ctrl_err:** Descriptor Control Error The DMA channel sets this bit indicating that the current transfer has an illegal control field value. When this bit has been set, the address of the failed descriptor and the channel returns to the Halted state, is recorded in the Channel Status register.
9	RWCST	0	**Wr_data_err:** Write Data Error The DMA channel sets this bit indicating that the current transfer has encountered an error while writing the destination data. When this bit has been set, the address of the failed descriptor when the channel returns to the Halted state, is recorded in the Channel Status register.
8	RWCST	0	**Rd_data_err:** Read Data Error The DMA channel sets this bit indicating that the current transfer has encountered an error while accessing the source data. When this bit has been set, the address of the failed descriptor when the channel returns to the Halted state, is recorded in the Channel Status register.
7	RO	0	**DMA_data_par_err:** DMA Data Parity Error The DMA channel sets this bit indicating that the current transfer has encountered a parity error reported by the DMA engine. The Intel 5000X Chipset MCH does not implement DMA parity error detection.
6	RWCST	0	**Cdata_par_err:** Chipset Data Parity Error The DMA channel sets this bit indicating that the current transfer has encountered a parity error reported by the chipset. When this bit has been set, the address of the failed descriptor when the channel returns to the Halted state, is recorded in the Channel Status register. In the case of Source data (read) error, the "**Rd_data_err**" field is also set. In the case of Destination (write) data error, the "**Wr_data_err**" field is also set
5	RWCST	0	**Chancmd_err:** CHANCMD Error The DMA channel sets this bit indicating that a write to the CHANCMD register contained an invalid value (for example, more than one command bit set). Software should take necessary action to clear the CHANCMD register first before resetting this register field.
4	RWCST	0	**Chn_addr_val_err:** Chain Address Value Error The DMA channel sets this bit indicating that the CHAINADDR register has an illegal address including an alignment error (not on a 64-byte boundary). This address will be checked and set by the DMA engine during execution. i.e when the initial descriptor is fetched.

Offset from CB_BAR Offset: 228h, 1A8h, 128h, A8h			
Bit	**Attr**	**Default**	**Description**
14	RO	0	**Soft_err: Soft Error** The Intel 5000X Chipset MCH does not implement soft error detection.
13	RWCST	0	**int_cfg_err:** Interrupt Configuration Error The DMA channel sets this bit indicating that the interrupt registers were not configured properly when the DMA channel attempted to generate an interrupt. See Section 3.11.7, Section 3.11.1 and Section 3.11.8.5. MSICTRL.MSIEN = 1, DMACTRL.MSICBEN = 0, PCICMD.INTx Disable = 1 MSICTRL.MSIEN = 0, DMACTRLMSICBEN = 1, PCICMD.INTx Disable = 1 MSICTRL.MSIEN = 0, DMACTRLMSICBEN = 0, PCICMD.INTx Disable = 1
12	RWCST	0	**Cmp_addr_err:** Completion Address Error The DMA channel sets this bit indicating that the completion address register was configured to an illegal address or has not been configured.
11	RWCST	0	**Desc_len_err:** Descriptor Length Error The DMA channel sets this bit indicating that the current transfer has an illegal length field value. When this bit has been set, the address of the failed descriptor and the channel returns to the Halted state, is recorded in the Channel Status register.
10	RWCST	0	**Desc_ctrl_err:** Descriptor Control Error The DMA channel sets this bit indicating that the current transfer has an illegal control field value. When this bit has been set, the address of the failed descriptor and the channel returns to the Halted state, is recorded in the Channel Status register.
9	RWCST	0	**Wr_data_err:** Write Data Error The DMA channel sets this bit indicating that the current transfer has encountered an error while writing the destination data. When this bit has been set, the address of the failed descriptor when the channel returns to the Halted state, is recorded in the Channel Status register.
8	RWCST	0	**Rd_data_err:** Read Data Error The DMA channel sets this bit indicating that the current transfer has encountered an error while accessing the source data. When this bit has been set, the address of the failed descriptor when the channel returns to the Halted state, is recorded in the Channel Status register.
7	RO	0	**DMA_data_par_err:** DMA Data Parity Error The DMA channel sets this bit indicating that the current transfer has encountered a parity error reported by the DMA engine. The Intel 5000X Chipset MCH does not implement DMA parity error detection.
6	RWCST	0	**Cdata_par_err:** Chipset Data Parity Error The DMA channel sets this bit indicating that the current transfer has encountered a parity error reported by the chipset. When this bit has been set, the address of the failed descriptor when the channel returns to the Halted state, is recorded in the Channel Status register. In the case of Source data (read) error, the "**Rd_data_err**" field is also set. In the case of Destination (write) data error, the "**Wr_data_err**" field is also set
5	RWCST	0	**Chancmd_err:** CHANCMD Error The DMA channel sets this bit indicating that a write to the CHANCMD register contained an invalid value (for example, more than one command bit set). Software should take necessary action to clear the CHANCMD register first before resetting this register field.
4	RWCST	0	**Chn_addr_val_err:** Chain Address Value Error The DMA channel sets this bit indicating that the CHAINADDR register has an illegal address including an alignment error (not on a 64-byte boundary). This address will be checked and set by the DMA engine during execution. i.e when the initial descriptor is fetched.

Offset from CB_BAR
Offset: 228h, 1A8h, 128h, A8h

Bit	Attr	Default	Description
14	RO	0	**Soft_err: Soft Error** The Intel 5000X Chipset MCH does not implement soft error detection.
13	RWCST	0	**int_cfg_err:** Interrupt Configuration Error The DMA channel sets this bit indicating that the interrupt registers were not configured properly when the DMA channel attempted to generate an interrupt. See Section 3.11.7, Section 3.11.1 and Section 3.11.8.5. MSICTRL.MSIEN = 1, DMACTRL.MSICBEN = 0, PCICMD.INTx Disable = 1 MSICTRL.MSIEN = 0, DMACTRLMSICBEN = 1, PCICMD.INTx Disable = 1 MSICTRL.MSIEN = 0, DMACTRLMSICBEN = 0, PCICMD.INTx Disable = 1
12	RWCST	0	**Cmp_addr_err:** Completion Address Error The DMA channel sets this bit indicating that the completion address register was configured to an illegal address or has not been configured.
11	RWCST	0	**Desc_len_err:** Descriptor Length Error The DMA channel sets this bit indicating that the current transfer has an illegal length field value. When this bit has been set, the address of the failed descriptor and the channel returns to the Halted state, is recorded in the Channel Status register.
10	RWCST	0	**Desc_ctrl_err:** Descriptor Control Error The DMA channel sets this bit indicating that the current transfer has an illegal control field value. When this bit has been set, the address of the failed descriptor and the channel returns to the Halted state, is recorded in the Channel Status register.
9	RWCST	0	**Wr_data_err:** Write Data Error The DMA channel sets this bit indicating that the current transfer has encountered an error while writing the destination data. When this bit has been set, the address of the failed descriptor when the channel returns to the Halted state, is recorded in the Channel Status register.
8	RWCST	0	**Rd_data_err:** Read Data Error The DMA channel sets this bit indicating that the current transfer has encountered an error while accessing the source data. When this bit has been set, the address of the failed descriptor when the channel returns to the Halted state, is recorded in the Channel Status register.
7	RO	0	**DMA_data_par_err:** DMA Data Parity Error The DMA channel sets this bit indicating that the current transfer has encountered a parity error reported by the DMA engine. The Intel 5000X Chipset MCH does not implement DMA parity error detection.
6	RWCST	0	**Cdata_par_err:** Chipset Data Parity Error The DMA channel sets this bit indicating that the current transfer has encountered a parity error reported by the chipset. When this bit has been set, the address of the failed descriptor when the channel returns to the Halted state, is recorded in the Channel Status register. In the case of Source data (read) error, the "**Rd_data_err**" field is also set. In the case of Destination (write) data error, the "**Wr_data_err**" field is also set
5	RWCST	0	**Chancmd_err:** CHANCMD Error The DMA channel sets this bit indicating that a write to the CHANCMD register contained an invalid value (for example, more than one command bit set). Software should take necessary action to clear the CHANCMD register first before resetting this register field.
4	RWCST	0	**Chn_addr_val_err:** Chain Address Value Error The DMA channel sets this bit indicating that the CHAINADDR register has an illegal address including an alignment error (not on a 64-byte boundary). This address will be checked and set by the DMA engine during execution. i.e when the initial descriptor is fetched.

Offset from CB_BAR Offset: 228h, 1A8h, 128h, A8h			
Bit	**Attr**	**Default**	**Description**
14	RO	0	**Soft_err: Soft Error** The Intel 5000X Chipset MCH does not implement soft error detection.
13	RWCST	0	**int_cfg_err:** Interrupt Configuration Error The DMA channel sets this bit indicating that the interrupt registers were not configured properly when the DMA channel attempted to generate an interrupt. See Section 3.11.7, Section 3.11.1 and Section 3.11.8.5. MSICTRL.MSIEN = 1, DMACTRL.MSICBEN = 0, PCICMD.INTx Disable = 1 MSICTRL.MSIEN = 0, DMACTRLMSICBEN = 1, PCICMD.INTx Disable = 1 MSICTRL.MSIEN = 0, DMACTRLMSICBEN = 0, PCICMD.INTx Disable = 1
12	RWCST	0	**Cmp_addr_err:** Completion Address Error The DMA channel sets this bit indicating that the completion address register was configured to an illegal address or has not been configured.
11	RWCST	0	**Desc_len_err:** Descriptor Length Error The DMA channel sets this bit indicating that the current transfer has an illegal length field value. When this bit has been set, the address of the failed descriptor and the channel returns to the Halted state, is recorded in the Channel Status register.
10	RWCST	0	**Desc_ctrl_err:** Descriptor Control Error The DMA channel sets this bit indicating that the current transfer has an illegal control field value. When this bit has been set, the address of the failed descriptor and the channel returns to the Halted state, is recorded in the Channel Status register.
9	RWCST	0	**Wr_data_err:** Write Data Error The DMA channel sets this bit indicating that the current transfer has encountered an error while writing the destination data. When this bit has been set, the address of the failed descriptor when the channel returns to the Halted state, is recorded in the Channel Status register.
8	RWCST	0	**Rd_data_err:** Read Data Error The DMA channel sets this bit indicating that the current transfer has encountered an error while accessing the source data. When this bit has been set, the address of the failed descriptor when the channel returns to the Halted state, is recorded in the Channel Status register.
7	RO	0	**DMA_data_par_err:** DMA Data Parity Error The DMA channel sets this bit indicating that the current transfer has encountered a parity error reported by the DMA engine. The Intel 5000X Chipset MCH does not implement DMA parity error detection.
6	RWCST	0	**Cdata_par_err:** Chipset Data Parity Error The DMA channel sets this bit indicating that the current transfer has encountered a parity error reported by the chipset. When this bit has been set, the address of the failed descriptor when the channel returns to the Halted state, is recorded in the Channel Status register. In the case of Source data (read) error, the "**Rd_data_err**" field is also set. In the case of Destination (write) data error, the "**Wr_data_err**" field is also set
5	RWCST	0	**Chancmd_err:** CHANCMD Error The DMA channel sets this bit indicating that a write to the CHANCMD register contained an invalid value (for example, more than one command bit set). Software should take necessary action to clear the CHANCMD register first before resetting this register field.
4	RWCST	0	**Chn_addr_val_err:** Chain Address Value Error The DMA channel sets this bit indicating that the CHAINADDR register has an illegal address including an alignment error (not on a 64-byte boundary). This address will be checked and set by the DMA engine during execution. i.e when the initial descriptor is fetched.

Offset from CB_BAR Offset: 228h, 1A8h, 128h, A8h			
Bit	**Attr**	**Default**	**Description**
14	RO	0	**Soft_err: Soft Error** The Intel 5000X Chipset MCH does not implement soft error detection.
13	RWCST	0	**int_cfg_err:** Interrupt Configuration Error The DMA channel sets this bit indicating that the interrupt registers were not configured properly when the DMA channel attempted to generate an interrupt. See Section 3.11.7, Section 3.11.1 and Section 3.11.8.5. MSICTRL.MSIEN = 1, DMACTRL.MSICBEN = 0, PCICMD.INTx Disable = 1 MSICTRL.MSIEN = 0, DMACTRLMSICBEN = 1, PCICMD.INTx Disable = 1 MSICTRL.MSIEN = 0, DMACTRLMSICBEN = 0, PCICMD.INTx Disable = 1
12	RWCST	0	**Cmp_addr_err:** Completion Address Error The DMA channel sets this bit indicating that the completion address register was configured to an illegal address or has not been configured.
11	RWCST	0	**Desc_len_err:** Descriptor Length Error The DMA channel sets this bit indicating that the current transfer has an illegal length field value. When this bit has been set, the address of the failed descriptor and the channel returns to the Halted state, is recorded in the Channel Status register.
10	RWCST	0	**Desc_ctrl_err:** Descriptor Control Error The DMA channel sets this bit indicating that the current transfer has an illegal control field value. When this bit has been set, the address of the failed descriptor and the channel returns to the Halted state, is recorded in the Channel Status register.
9	RWCST	0	**Wr_data_err:** Write Data Error The DMA channel sets this bit indicating that the current transfer has encountered an error while writing the destination data. When this bit has been set, the address of the failed descriptor when the channel returns to the Halted state, is recorded in the Channel Status register.
8	RWCST	0	**Rd_data_err:** Read Data Error The DMA channel sets this bit indicating that the current transfer has encountered an error while accessing the source data. When this bit has been set, the address of the failed descriptor when the channel returns to the Halted state, is recorded in the Channel Status register.
7	RO	0	**DMA_data_par_err:** DMA Data Parity Error The DMA channel sets this bit indicating that the current transfer has encountered a parity error reported by the DMA engine. The Intel 5000X Chipset MCH does not implement DMA parity error detection.
6	RWCST	0	**Cdata_par_err:** Chipset Data Parity Error The DMA channel sets this bit indicating that the current transfer has encountered a parity error reported by the chipset. When this bit has been set, the address of the failed descriptor when the channel returns to the Halted state, is recorded in the Channel Status register. In the case of Source data (read) error, the "**Rd_data_err**" field is also set. In the case of Destination (write) data error, the "**Wr_data_err**" field is also set
5	RWCST	0	**Chancmd_err:** CHANCMD Error The DMA channel sets this bit indicating that a write to the CHANCMD register contained an invalid value (for example, more than one command bit set). Software should take necessary action to clear the CHANCMD register first before resetting this register field.
4	RWCST	0	**Chn_addr_val_err:** Chain Address Value Error The DMA channel sets this bit indicating that the CHAINADDR register has an illegal address including an alignment error (not on a 64-byte boundary). This address will be checked and set by the DMA engine during execution. i.e when the initial descriptor is fetched.

Offset from CB_BAR Offset: 228h, 1A8h, 128h, A8h			
Bit	**Attr**	**Default**	**Description**
14	RO	0	**Soft_err: Soft Error** The Intel 5000X Chipset MCH does not implement soft error detection.
13	RWCST	0	**int_cfg_err:** Interrupt Configuration Error The DMA channel sets this bit indicating that the interrupt registers were not configured properly when the DMA channel attempted to generate an interrupt. See Section 3.11.7, Section 3.11.1 and Section 3.11.8.5. MSICTRL.MSIEN = 1, DMACTRL.MSICBEN = 0, PCICMD.INTx Disable = 1 MSICTRL.MSIEN = 0, DMACTRLMSICBEN = 1, PCICMD.INTx Disable = 1 MSICTRL.MSIEN = 0, DMACTRLMSICBEN = 0, PCICMD.INTx Disable = 1
12	RWCST	0	**Cmp_addr_err:** Completion Address Error The DMA channel sets this bit indicating that the completion address register was configured to an illegal address or has not been configured.
11	RWCST	0	**Desc_len_err:** Descriptor Length Error The DMA channel sets this bit indicating that the current transfer has an illegal length field value. When this bit has been set, the address of the failed descriptor and the channel returns to the Halted state, is recorded in the Channel Status register.
10	RWCST	0	**Desc_ctrl_err:** Descriptor Control Error The DMA channel sets this bit indicating that the current transfer has an illegal control field value. When this bit has been set, the address of the failed descriptor and the channel returns to the Halted state, is recorded in the Channel Status register.
9	RWCST	0	**Wr_data_err:** Write Data Error The DMA channel sets this bit indicating that the current transfer has encountered an error while writing the destination data. When this bit has been set, the address of the failed descriptor when the channel returns to the Halted state, is recorded in the Channel Status register.
8	RWCST	0	**Rd_data_err:** Read Data Error The DMA channel sets this bit indicating that the current transfer has encountered an error while accessing the source data. When this bit has been set, the address of the failed descriptor when the channel returns to the Halted state, is recorded in the Channel Status register.
7	RO	0	**DMA_data_par_err:** DMA Data Parity Error The DMA channel sets this bit indicating that the current transfer has encountered a parity error reported by the DMA engine. The Intel 5000X Chipset MCH does not implement DMA parity error detection.
6	RWCST	0	**Cdata_par_err:** Chipset Data Parity Error The DMA channel sets this bit indicating that the current transfer has encountered a parity error reported by the chipset. When this bit has been set, the address of the failed descriptor when the channel returns to the Halted state, is recorded in the Channel Status register. In the case of Source data (read) error, the "**Rd_data_err**" field is also set. In the case of Destination (write) data error, the "**Wr_data_err**" field is also set
5	RWCST	0	**Chancmd_err:** CHANCMD Error The DMA channel sets this bit indicating that a write to the CHANCMD register contained an invalid value (for example, more than one command bit set). Software should take necessary action to clear the CHANCMD register first before resetting this register field.
4	RWCST	0	**Chn_addr_val_err:** Chain Address Value Error The DMA channel sets this bit indicating that the CHAINADDR register has an illegal address including an alignment error (not on a 64-byte boundary). This address will be checked and set by the DMA engine during execution. i.e when the initial descriptor is fetched.

Offset from CB_BAR Offset: 228h, 1A8h, 128h, A8h			
Bit	**Attr**	**Default**	**Description**
3	RWCST	0	**Desc_err:** Descriptor Error The DMA channel sets this bit indicating that the current descriptor has encountered an error when executing a DMA descriptor that is not otherwise related to other error bits. For example, an illegal next descriptor address flagged by the system Address decoder, which the DMA engine encounters in the current descriptor after having successfully completed the data transfer for the current descriptor including any associated completions/interrupts. When this bit has been set and the channel returns to the Halted state, the address of the failed descriptor is in the Channel Status register.
2	RWCST	0	**Nxt_desc_addr_err**: Next Descriptor Address Error The DMA channel sets this bit indicating that the current descriptor has an illegal next descriptor address (for example, >40-bits) including next descriptor alignment error (not on a 64-byte boundary). This error could be flagged when the data for the current descriptor is fetched and its constituent fields are checked.
1	RWCST	0	**DMA_xfer_daddr_err:** DMA Transfer Destination Address Error The DMA channel sets this bit indicating that the current descriptor has an illegal destination address. When this bit has been set, the address of the failed descriptor when the channel returns to the Halted state, is recorded in the Channel Status register.
0	RWCST	0	**DMA_trans_saddr_err:** DMA Transfer Source Address Error The DMA channel sets this bit indicating that the current descriptor has an illegal source address. When this bit has been set, the address of the failed descriptor when the channel returns to the Halted state, is recorded in the Channel Status register.

3.11.11.9 CHANERRMSK[3:0]: Channel Error Mask Register

The Channel Error Mask Register provides selective control to mask errors from before being signaled. They are still recorded in the CHANERR,FERR/NERR_CHAN_ERR log registers irrespective of the value in this register. If one of the bits in the CHANERRMSK register is set, then that specific error does not cause interrupt signaling.

Offset from CB_BAR Offset: 22Ch, 1ACh, 12Ch, ACh			
Bit	**Attr**	**Default**	**Description**
31:16	RV	0h	Reserved
15	RO	0	**Unaffil_err_Msk:** Unaffiliated Error Mask
14	RO	0	**Soft_err_Msk:** Soft Error Mask
13	RWST	0	**int_cfg_err_Msk:** Interrupt Configuration Error Mask
12	RWST	0	**Cmp_addr_err_Msk:** Completion Address Error Mask
11	RWST	0	**Desc_len_err_Msk:** Descriptor Length Error Mask
10	RWST	0	**Desc_ctrl_err_Msk:** Descriptor Control Error Mask
9	RWST	0	**Wr_data_err_Msk:** Write Data Error Mask
8	RWST	0	**Rd_data_err_Msk:** Read Data Error Mask
7	RWST	0	**DMA_data_par_err_Mskr:** DMA Data Parity Error Mask
6	RWST	0	**Cdata_par_err_Msk:** Chipset Data Parity Error Mask
5	RWST	0	**Chancmd_err_Msk:** CHANCMD Error Mask
4	RWST	0	**Chn_addr_val_err_Msk:** Chain Address Value Error Mask
3	RWST	0	**Desc_err_Msk:** Descriptor Error Mask
2	RWST	0	**Nxt_desc_addr_err_Msk:** Next Descriptor Address Error Mask

Offset from CB_BAR Offset: 22Ch, 1ACh, 12Ch, ACh			
Bit	**Attr**	**Default**	**Description**
1	RWST	0	**DMA_xfrer_daddr_err_Msk:** DMA Transfer Destination Address Error Mask
0	RWST	0	**DMA_trans_saddr_err_Msk:** DMA Transfer Source Address Error Mask

3.11.12 PCI Express Per-Port Registers

Table 3-54 lists the standard registers added to the DMA Engine device's MMIO address map for each port that supports DMA Engine functionality. These registers control both the PCI Express proprietary enhancements as well as the outbound write combining feature. Each port which implements DMA Engine Technology is required to have a set of these registers located in the DMA Engine device's MMIO space.

Under some existing operating systems (for example, Microsoft Windows*), a device driver is only allowed to access its own PCI configuration space, not the configuration space of any other device. This is why the per-port stream ID and write combining register sets exist in the memory mapped I/O (MMIO) space. This avoids having a separate PCI filter device driver (instance) on each of the PCI-to-PCI bridges and solves the problem of the OS remapping PCI Bridge MMIO space.

It also avoids the need for providing inbound configuration access, besides easier for resource reallocation (Hot-Plug), and provides the TOE device a direct mechanism to manipulate the resources pertaining to Write combining for that port.

3.11.12.1 NXTPPRSET2 - Next Per Port Register Set

Offset from CB_BAR Offset: 300h			
Bit	**Attr**	**Default**	**Description**
15:2	RO	00E0h	**Nxt_PP_Offset:** Next Per port offset This value added to the CB_BAR value provides the memory address of the next set of per-port registers for port 3. Since the Per port offset is 32-bit aligned, the effective address is calculated by concatenating "00E0h" (bits 7,8,9 are '1's) with "00b" giving 380h as an offset from CB_BAR and this locates the port priority registers for port 3 in the map.
1:0	RV	00	*Reserved*

The per-port offset indicates where the next set of per-port registers resides in the MMIO space. This register points to the next set of per-port registers and provides the means for S/W to locate all of the per-port MMIO register sets.

3.11.12.2 NXTPPRSET3 - Next Per Port Register Set

Offset from CB_BAR Offset: 380h			
Bit	**Attr**	**Default**	**Description**
15:2	RO	0h	**Nxt_PP_Offset:** Next Per port offset This is the last per-port register set (port 3) in the linked list and hence this value is 0h.
1:0	RV	0	*Reserved*

3.11.12.3 PPRSETLEN[3:2] - Per-Port Register Set Length Register

Offset from CB_BAR Offset: 382h, 302h			
Bit	**Attr**	**Default**	**Description**
7:0	RO	60h	**PPRSLEN:** PPR Set Length This register indicates the length of the per-port register set in bytes (including NXTPPRSET, and PPRSETLEN) and up to the end of the last DOORBELL_SNOOP_ADDR register.

3.11.12.4 STRMPRI[3:2]: Stream Priority Register

Offset from CB_BAR Offset: 383h, 303h			
Bit	**Attr**	**Default**	**Description**
7:4	RO	0000	**Highest_pri_supported: Highest Priority Supported** The Intel 5000X Chipset MCH does not support stream priority and hence this field is initialized to 0.
3:0	RW	0000	**Def_strmpri:** Default Stream Priority This field specifies the priority given to StreamIDs greater than the number of streams supported. If a stream priority greater than 0 is set in this register field, the Intel 5000X Chipset MCH will treat the stream priority for the exceeding streams to default to that of priority 0.

3.11.12.5 REQID[3:2] - Requestor ID Register

Offset from CB_BAR Offset: 384h, 304h			
Bit	**Attr**	**Default**	**Description**
15:8	RO	0h	**BN:** Bus Number This field indicates the bus number which the device implementing streams resides on. This is reflected in the Requester ID field of the PCI Express packet. The chipset checks this register against the incoming transaction's header before applying stream prioritization.
7:3	RO	0h	**DN:** Device Number This field indicates the device number which the device implementing streams resides on. This is reflected in the Requester ID field of the PCI Express packet. The chipset checks this register against the incoming transaction's header before applying stream prioritization.
2:0	RO	0h	**FN:** Function Number This field indicates the function number which the device implementing streams resides on. This is reflected in the Requester ID field of the PCI Express packet. The chipset checks this register against the incoming transaction's header before applying stream prioritization.

The Requester ID register is programmed such that the chipset can differentiate between devices which implement prioritized streams and those which do not. Since the MCH does not support stream priority, this register is RO.

3.11.12.6 STRMCAP[3:2] - Stream Capacity Register

This register defines the chipset's capacity to implement an arbitrary number of streams for DMA Engine. The MCH will implement and support 2 streams each for ports 2 and 3.

Offset from CB_BAR
Offset: 387h, 307h

Bit	Attr	Default	Description
7:6	RV	0h	*Reserved*
5:0	RO	000000	**Num_strms_supported:** Number of Streams Supported This field identifies how many Stream ID's the Intel 5000X Chipset MCH supports by indicating the highest StreamID value for which the device can specify a priority. The Intel 5000X Chipset MCH supports only 1 stream and the highest stream ID is 0.

3.11.12.7 STRMIDFMT[3:2] - Stream ID Format Register

Offset from CB_BAR
Offset: 386h, 306h

Bit	Attr	Default	Description
7	RO	0	**In_use:** In Use This bit indicates whether this Stream ID register is in use. A device uses this mechanism to atomically claim exclusive ownership of this Stream ID format register. This should be done before attempting to program it. Since Intel 5000X Chipset MCH does not support stream priority, this is a RO register field
6:5	RV	0h	*Reserved*
4	RO	0	**Ign_fn_num:** Ignore Function Number 0: When this bit is clear, the chipset only honors stream prioritization with transactions with a Function number matching the of the function number recorded in the REQID register and the chipset places transactions with a different Function number into the low priority queue (LP). 1: When this bit is set, the function number captured in the REQID register is ignored for incoming transactions. This enables the stream prioritization feature to be implemented across multiple functions in the device. The value of this bit does not effect the number of Function field bits that the chipset decodes (see Function Field Size below).
3:2	RO	0h	**Function Field Size**: This field identifies how many upper bits of the PCI Express header's Function field in the RequesterID should be decoded as the StreamID. For any values other than 00, the Phantom Function capability must be employed in the device. These bit are updated when the chipset receives a Stream Priority Message 00: No Function bits are used to identify the StreamID. 01: Function[2] is used to identify the StreamID. 10: Function[2:1] is used to identify the StreamID. 11: Function[2:0] is used to identify the StreamID.
1:0	RO	0h	**Tag Field Size**: This field identifies how many upper bits of the PCI Express header's Tag field should be decoded as the StreamID. These bit are updated when the chipset receives a Stream Priority Message. 00: No Tag bits are used to identify the StreamID. 01: Tag[7] is used to identify the StreamID. 10: Tag[7:6] is used to identify the StreamID. 11: Tag[7:5] is used to identify the StreamID.

The StreamID format register is programmed such that the chipset can decode the appropriate header bits as the StreamID for device initiated requests. A combination of the upper Tag bits and the Function field form the StreamID. This register is programmed in two ways: explicitly through BIOS or with the Stream Priority message (using the StrmFmt field in the message). Since the MCH does not support stream priority this register is RO and can not be programmed.

3.11.12.8 WCNUM[3:2] - Write Combining Number Register

Offset from CB_BAR Offset: 388h, 308h			
Bit	**Attr**	**Default**	**Description**
7:0	RW	04h	**Num_WCR:** Number of WC Ranges This specifies the number of WC regions enabled in the Intel 5000X Chipset MCH. By default, the Intel 5000X Chipset MCH supports a maximum of 4 WC Ranges. Any value greater than 4 programmed by Software will be deemed illegal and the chipset will set this field to the default maximum of 4. If Software sets a value for Num_WCR as 0, it implies that there are no write combining ranges and the chipset should not decode any requests in this WC range

The Write Combining Number register reports the number of write combining windows supported. The first window begins at WCBASE (see below) and all windows are contiguous from there.

3.11.12.9 WCSIZE[3:2] - Write Combining Size Register

Offset from CB_BAR Offset: 389h, 309h			
Bit	**Attr**	**Default**	**Description**
7:3	RV	0h	Reserved
2:0	RW	0h	**Write Combining Range Size**: This field specifies the size of a WC range which the Intel 5000X Chipset MCH chipset permits to write combine. (Note that WCNUM as defined earlier specifies the total number of ranges). Refer to the Write combining section in the PCI Express chapter for details. 000 - No region(s) 001 - 1 KB region(s) 010 - 2 KB region(s) 011 - 4 KB region(s) 100 - 8 KB region(s) Others - Invalid (no region) This value is programmed by the device driver software based on the network requirements. A value of 0 implies that there are no WC Ranges for the chipset to decode.

3.11.12.10 WCCTRL[3:2] - Write Combining Control Register

The Write Combining Control register controls features specific to the chipset write combining capabilities.

Offset from CB_BAR Offset: 38Ah, 30Ah			
Bit	**Attr**	**Default**	**Description**
15:9	RV	0h	Reserved
8	RW	0	**In Use**: This bit indicates whether write combining register set (WCNUM, WCSIZE, WCCTRL, WCBASE and WCWINDOW) is in use. The first time this bit is read when it has been cleared, it will return 0 and automatically transition from 0 to 1, reserving the channel for the first consumer that reads this register. All subsequent reads will return 1 indicating that this write combining register set is in use. This bit is cleared by writing a 0 value, thus releasing this write combining register set. A consumer uses this mechanism to atomically claim exclusive ownership of the write combining register set. This should be done before attempting to program any register in the write combining register set.
7:2	RV	0h	Reserved

Offset from CB_BAR Offset: 38Ah, 30Ah			
Bit	**Attr**	**Default**	**Description**
1	RW	0	**Write Combining Window Clear**: Setting this bit clears all chipset write combining windows by invalidating all entries (data is not flushed). The chipset hardware clears this bit once all the data is invalidated from the window.
0	RW	0	**Write Combining Window Flush**: Setting this bit flushes all chipset write combining windows. The chipset hardware clears this bit once all the pending data is flushed (posted) from the window.

3.11.12.11 WCCAP[3:2] - Write Combining Capacity Register

The Write Combining Capacity register specifies the maximum value for WCNUM.

Offset from CB_BAR Offset: 38Ch, 30Ch			
Bit	**Attr**	**Default**	**Description**
7:0	RO	04h	**NUMWCR:** Number of WC Regions This specifies the maximum number of WC regions supported by the Intel 5000X Chipset MCH chipset which is 4.

3.11.12.12 WCWINDOW[3:2] - Write Combining Window Register

The Write Combining Window register is programmed with the maximum size the chipset should write combine before evicting the data to the I/O device.

Offset from CB_BAR Offset: 38Dh, 30Dh			
Bit	**Attr**	**Default**	**Description**
7:4	RV	0h	Reserved
3:0	RO	0100	**Write Combining Window Size**: This field specifies the size of the write combining window supported by the Intel 5000X Chipset MCH. **0100** - 128 byte window size

3.11.12.13 BRIDGE_ID2 - Bridge ID Register

Offset from CB_BAR Offset: 30Eh			
Bit	**Attr**	**Default**	**Description**
15:8	RO	0	**BN:** Bus Number This field indicates the bus number of the PCI Bridge serving the port for this set of per-port DMA Engine functions.
7:3	RO	02h	**DN:** Device Number This field indicates the device number of the PCI Bridge serving the port for this set of per-port DMA Engine functions.
2:0	RO	0	**FN:** Function Number This field indicates the function number of the PCI Bridge serving the port for this set of per-port DMA Engine functions.

The Bridge ID register identifies the PCI bridge for port 2 for which this set of per-port functions are valid. S/W can query the bridge's configuration registers to determine which I/O devices are downstream of the bridge and thus can use this set of per-port DMA Engine resources. Note that only one of those down stream devices can use Write Combining or Stream Priority capability at a time (see In-Use bit).

3.11.12.14 BRIDGE_ID3 - Bridge ID Register

Offset from CB_BAR
Offset: 38Eh

Bit	Attr	Default	Description
15:8	RO	0	**BN:** Bus Number This field indicates the bus number of the PCI Bridge serving the port for this set of per-port DMA Engine functions.
7:3	RO	03h	**DN:** Device Number This field indicates the device number of the PCI Bridge serving the port for this set of per-port DMA Engine functions.
2:0	RO	0	**FN:** Function Number This field indicates the function number of the PCI Bridge serving the port for this set of per-port DMA Engine functions.

The Bridge ID register identifies the PCI bridge for port 3 for which this set of per-port functions are valid. S/W can query the bridge's configuration registers to determine which I/O devices are downstream of the bridge and thus can use this set of per-port DMA Engine resources. Note that only one of those down stream devices can use Write Combining or Stream Priority capability at a time.

3.11.12.15 WCBASE[3:2] - Write Combining Base Register

The Write Combining Base register contains the starting address which the chipset is permitted to combine. The write combining base is 4KB aligned and is used with the WCSIZE, WCBASE register to define the top of the WC region to be combined.

Offset from CB_BAR
Offset: 390h, 310h

Bit	Attr	Default	Description
63:12	RW	0h	**Write Combining Region Base Address**: This field specifies the base address of the range which the chipset permits write combining.
11:0	RV	0h	Reserved

The address decoding in Intel 5000X Chipset for write combining with WCNUM=4 is given as follows:

- WCBASE <= Address < WCBASE +WCSIZE for WC range 0
- WCBASE + WCSIZE <= Address < WCBASE+2*WCSIZE for WC range 1
- WCBASE + 2*WCSIZE <= Address < WCBASE+3*WCSIZE for WC range 2
- WCBASE + 3*WCSIZE <= Address < WCBASE+4*WCSIZE for WC range 3

The Write Combining Size register (WCSIZE) is programmed with the size of the region which the chipset is permitted to combine while the Write Combining Number Register (WCNUM) defines the number of WC ranges.

If either WCSIZE or WCNUM is 0, then it implies that there are effectively no WC Ranges and the chipset should not decode and coalesce WC write requests.

3.11.12.16 STRMMAP_OFFSET2: Stream Priority Mapping Offset Register

Offset from CB_BAR Offset: 318h			
Bit	**Attr**	**Default**	**Description**
15:2	RO	0h	**Strmid_offset: Offset of Stream ID Map** The Intel 5000X Chipset MCH does not implement stream priority and hence this field is set to a Null pointer.
1:0	RV	0h	*Reserved*

The offset indicates where the Stream ID priority Mapping register for port 2 resides in the MMIO space.

3.11.12.17 STRMMAP_OFFSET3: Stream Priority Mapping Offset Register

Offset from CB_BAR Offset: 398h			
Bit	**Attr**	**Default**	**Description**
15:2	RO	0h	**Strmid_offset: Offset of Stream ID Map** The Intel 5000X Chipset MCH does not implement stream priority and hence this field is set to a Null pointer.
1:0	RV	0h	*Reserved*

The offset indicates where the Stream ID priority Mapping register for port 3 resides in the MMIO space.

3.11.12.18 PORTPRI2: Port Priority Register

Offset from CB_BAR Offset: 31Ah			
Bit	**Attr**	**Default**	**Description**
7:4	RV	0	*Reserved*
3:0	RO	0h	**Port_pri: Port Priority** The Intel 5000 Series Chipset does not implement port priority for port 2 and this register will consistently return zero.

3.11.12.19 PORTPRI3: Port Priority Register

Offset from CB_BAR Offset: 39Ah			
Bit	**Attr**	**Default**	**Description**
7:4	RV	0	*Reserved*
3:0	RO	0h	**Port_pri: Port Priority** The Intel 5000 Series Chipset does not implement port priority for port 3 and this register will consistently return zero.

3.11.12.20 STRM_COMP[3:2] - Stream Priority Compatibility Register

Offset from CB_BAR Offset: 39Ch, 31Ch			
Bit	**Attr**	**Default**	**Description**
15:1	RV	0h	*Reserved*
0	RO	0	**v1_comp:** v1 Compatibility The Intel 5000X Chipset MCH does not support Stream Priority operation for ports 2 and 3. Hence, it is not compatible with DMA Engine software and is hardwired to 0.

3.11.12.21 WC_COMP[3:2] - Write Combining Compatibility Register

Offset from CB_BAR Offset: 39Eh, 31Eh			
Bit	**Attr**	**Default**	**Description**
15:1	RV	0h	*Reserved*
0	RO	1	**v1_comp:** v1 Compatibility The chipset supports write Combining operation that is compatible with software written for version 1 of the DMA Engine specification and is set to 1 by the hardware.

3.11.12.22 BR_MEM_BASE[3:2] - Bridge Memory Base Register

Offset from CB_BAR Offset: 3A0h, 320h			
Bit	**Attr**	**Default**	**Description**
15:0	RO	0h	**BR_MBASE:** Bridge Memory Base Register This is a copy of the Memory Base Register of the virtual P2P bridge for the PCI Express port with which it is associated (port 3 or port2). See also Section 3.8.8.16.

The BR_MEM_BASE and BR_MEM_LIMIT registers identifies the MMIO address range for PCI devices served by the PCI Express port 2 or 3. These registers are a copy of the PCI Express_configuration registers for the port for which this set of per-port functions are valid. The corresponding port's PEXCMD.MSE register field will gate the output of this register during read operations.This enables software to determine which I/O devices are downstream of this port and thus can use this set of per-port DMA Engine resources. Note that only one of those down stream devices can use Write Combining or Stream Priority capability at a time (see In-Use bit).

3.11.12.23 BR_MEM_LIMIT[3:2] - Bridge Memory Limit Register

Offset from CB_BAR Offset: 3A2h, 322h			
Bit	**Attr**	**Default**	**Description**
15:0	RO	0h	**BR_MLIM:** Bridge Memory Limit Register This is a copy of the Memory Limit Register of the virtual P2P bridge for the PCI Express port with which it is associated (port 3 or port2). Refer to Section 3.8.8.17.

3.11.12.24 BR_PMEM_BASE[3:2] - Bridge Prefetchable Memory Base Register

Offset from CB_BAR Offset: 3A4h, 324h			
Bit	**Attr**	**Default**	**Description**
15:0	RO	0h	**BR_PMBASE:** Bridge Prefetchable Memory Base Register This is a copy of the Prefetchable Memory Base Register of the virtual P2P bridge for the PCI Express port with which it is associated (port 3 or port2). Refer to Section 3.8.8.18. If PEXCMD.MSE==0 for the virtual P2P port, then this register, when read, returns a value of 0h else return contents of BR_PMBASE field.

The BR_PMEM_BASE along with BR_PMEM_LIMIT, BR_PBASE_UPPER32 and BR_PLIMIT_UPPER32 registers defined below identify the prefetchable address range for PCI Express devices served by the respective port (64-bit addressable space). These registers are a copy of the PCI Express port's configuration registers for which this set of per-port functions are valid. The corresponding port's PEXCMD.MSE register field will gate the output of this register during read operations. S/W can determine which I/O devices are downstream of this port and thus can use this set of per-port DMA Engine resources. Note that only one of those down stream devices can use Write Combining or Stream Priority capability at a time (see In-Use bit).

3.11.12.25 BR_PMEM_LIMIT[3:2] - Bridge Prefetchable Memory Limit Register

Offset from CB_BAR Offset: 3A6h, 326h			
Bit	**Attr**	**Default**	**Description**
15:0	RO	0h	**BR_PMLIM:** Bridge Prefetchable Memory Limit Register This is a copy of the Prefetchable Memory Limit Register of the virtual P2P bridge for the PCI Express port with which it is associated (port 3 or port2). Refer to Section 3.8.8.19If PEXCMD.MSE=0 for the virtual P2P port, then this register when read, returns a value of 0h when read else return contents of BR_PMLIM field.

3.11.12.26 BR_PBASE_UPPER32_P[3:2] - Bridge Prefetchable Base Upper 32 Register

Offset from CB_BAR Offset: 3A8h, 328h			
Bit	**Attr**	**Default**	**Description**
31:0	RO	0h	**BR_PMBU:** Bridge Prefetchable Base Upper 32 Register This is a copy of the Prefetchable Base Upper 32 Register of the virtual P2P bridge for the PCI Express port with which it is associated (port 3 or port2). Refer to Section 3.8.8.20If PEXCMD.MSE==0 for the virtual P2P port, then this register when read, returns a value of 0h when read else return contents of BR_PMBU field.

3.11.12.27 BR_PLIMIT_UPPER32_P[3:2] - Bridge Prefetchable Limit Upper 32 Register

Offset from CB_BAR Offset: 3ACh, 32Ch			
Bit	**Attr**	**Default**	**Description**
31:0	RO	0h	**Bridge Prefetchable Limit Upper 32 Register**: This is a copy of the Prefetchable Limit Upper 32 Register of the virtual P2P bridge (Section 3.8.8.21). If PEXCMD.MSE==0 for the virtual P2P port, then this register when read, returns a value of 0h when read else return contents of BR_PMLU field.

3.11.12.28 DOORBELL_SNOOP_ADDR0_P[3:2] - Doorbell Snoop Address Register 0

Offset from CB_BAR Offset: 3C0h, 340h			
Bit	**Attr**	**Default**	**Description**
63:40	RO	0h	**DOORBELL_SNP_Addr_Upper:** Doorbell Snoop Address Upper Address bits 63:40 are not implemented in Intel 5000X Chipset MCH and are hardwired to 0.
39:0	RW	0h	**DOORBELL_SNP_Addr:** Doorbell Snoop Address This register holds the physical address of the corresponding doorbell register on the interfaced I/O device. When software writes to this location, the chipset logic first flushes out the corresponding write combining window and subsequently, allows the doorbell write to proceed.

The MCH implements 4 Door bell snoop registers i, each of which is uniquely associated with one write combining window in WC range i, 0 <= i < WC_NUM as shown in Table 3-57.

Table 3-57. MCH WC Implementation with 4 WC (WCNUM=4) Ranges and 1 WC Window per Range

WC Range	Start of Range	End of Range	Affiliated DOORBELL SNOOP Address Register
0	WCBASE	WCBASE+WCSIZE-1	DOORBELLSNOOP0
1	WCBASE+WCSIZE	WCBASE+2*WCSIZE-1	DOORBELLSNOOP1
2	WCBASE+2*WCSIZE	WCBASE+3*WCSIZE-1	DOORBELLSNOOP2
3	WCBASE+3*WCSIZE	WCBASE+4*WCSIZE-1	DOORBELLSNOOP3

3.11.12.29 DOORBELL_SNOOP_ADDR1_P[3:2] - Doorbell Snoop Address Register 1

Offset from CB_BAR Offset: 3C8h, 348h			
Bit	**Attr**	**Default**	**Description**
63:40	RO	0h	**DOORBELL_SNP_Addr_Upper:** Doorbell Snoop Address Upper Address bits 63:40 are not implemented in Intel 5000X Chipset MCH and are hardwired to 0.
39:0	RW	0h	**DOORBELL_SNP_Addr:** Doorbell Snoop Address This register holds the physical address of the corresponding doorbell register on the interfaced I/O device. When software writes to this location, the chipset logic first flushes out the corresponding write combining window and subsequently, allows the doorbell write to proceed.

3.11.12.30 DOORBELL_SNOOP_ADDR2_P[3:2] - Doorbell Snoop Address Register 2

Offset from CB_BAR Offset: 3D0h, 350h			
Bit	**Attr**	**Default**	**Description**
63:40	RO	0h	**DOORBELL_SNP_Addr_Upper:** Doorbell Snoop Address Upper Address bits 63:40 are not implemented in Intel 5000X Chipset MCH and are hardwired to 0.
39:0	RW	0h	**DOORBELL_SNP_Addr:** Doorbell Snoop Address This register holds the physical address of the corresponding doorbell register on the interfaced I/O device. When software writes to this location, the chipset logic first flushes out the corresponding write combining window and subsequently, allows the doorbell write to proceed.

3.11.12.31 DOORBELL_SNOOP_ADDR3_P[3:2] - Doorbell Snoop Address Register 3

Offset from CB_BAR Offset: 3D8h, 358h			
Bit	**Attr**	**Default**	**Description**
63:40	RO	0h	**DOORBELL_SNP_Addr_Upper:** Doorbell Snoop Address Upper Address bits 63:40 are not implemented in Intel 5000X Chipset MCH and are hardwired to 0.
39:0	RW	0h	**DOORBELL_SNP_Addr:** Doorbell Snoop Address This register holds the physical address of the corresponding doorbell register on the interfaced I/O device. When software writes to this location, the chipset logic first flushes out the corresponding write combining window and subsequently, allows the doorbell write to proceed.

3.11.12.32 WCERRLOG0_P[3:2] - Write Combining Error Log 0

This register records errors associated with the respective WC window in the WC Range 0 for the respective port in the MCH. These errors are considered non-fatal.

Offset from CB_BAR Offset: 3E8h, 368h			
Bit	**Attr**	**Default**	**Description**
7:1	RV	0h	*Reserved*
0	RWCST	0h	**WC_Data_poison:** WC Data Poison error 1: Indicates that the WC data in WC Range 0 was poisoned due to an error in the Intel 5000X Chipset MCH or its interfaces 0: No poisoned data error has occurred.

3.11.12.33 WCERRLOG1_P[3:2] - Write Combining Error Log 1

This register records errors associated with the respective WC window in the WC Range 0 for the respective port in the MCH. These errors are considered non-fatal.

Offset from CB_BAR Offset: 3ECh, 36Ch			
Bit	**Attr**	**Default**	**Description**
7:1	RV	0h	*Reserved*
0	RWCST	0h	**WC_Data_poison:** WC Data Poison error 1: Indicates that the WC data in WC Range 1 was poisoned due to an error in the Intel 5000X Chipset MCH or its interfaces 0: No poisoned data error has occurred.

3.11.12.34 WCERRLOG2_P[3:2] - Write Combining Error Log 2

This register records errors associated with the respective WC window in the WC Range 0 for the respective port in the MCH. These errors are considered non-fatal.

Offset from CB_BAR Offset: 3F0h, 370h			
Bit	**Attr**	**Default**	**Description**
7:1	RV	0h	*Reserved*
0	RWCST	0h	**WC_Data_poison:** WC Data Poison error 1: Indicates that the WC data in WC Range 2 was poisoned due to an error in the Intel 5000X Chipset MCH or its interfaces 0: No poisoned data error has occurred.

3.11.12.35 WCERRLOG3_P[3:2] - Write Combining Error Log 3

This register records errors associated with the respective WC window in the WC Range 0 for the respective port in the MCH. These errors are considered non-fatal.

Offset from CB_BAR Offset: 3F4h, 374h			
Bit	**Attr**	**Default**	**Description**
7:1	RV	0h	*Reserved*
0	RWCST	0h	**WC_Data_poison:** WC Data Poison error 1: Indicates that the WC data in WC Range3 was poisoned due to an error in the Intel 5000X Chipset MCH or its interfaces 0: No poisoned data error has occurred.

3.11.13 PCI Express Intel IBIST Registers

3.11.13.1 DIOIBSTR: PCI Express Intel IBIST Global Start/Status Register

This register contains the global start for all the ports in the Intel 5000X Chipset MCH component simultaneously. One start bit is placed in the register for each port. Intel IBIST will start at approximately the same time on all ports written to with a 1 in the same write access.

Device: 0 Function: 0 Offset: 398h			
Bit	**Attr**	**Default**	**Description**
7	RW	0	**START7:** Writing a 1 starts IBIST on port 7.
6	RW	0	**START6:** Writing a 1 starts IBIST on port 6.
5	RW	0	**START5:** Writing a 1 starts IBIST on port 5.
4	RW	0	**START4:** Writing a 1 starts IBIST on port 4.
3	RW	0	**START3:** Writing a 1 starts IBIST on port 3.
2	RW	0	**START2:** Writing a 1 starts IBIST on port 2.
1	RV	0	*Reserved*
0	RW	0	**START0:** Writing a 1 starts IBIST on port 0.

3.11.13.2 DIO0IBSTAT: PCI Express Intel IBIST Completion Status Register

Device: 0 Function: 0 Offset: 394h			
Bit	**Attr**	**Default**	**Description**
7	RO	0	**IBSTAT7:** IBIST Status port 7 0: IBIST either has not started the first time or it is still running. 1: IBIST is done. This bit will be cleared by hardware when the start bit is asserted.
6	RO	0	**IBSTAT6:** IBIST Status port 6 0: IBIST either has not started the first time or it is still running. 1: IBIST is done. This bit will be cleared by hardware when the start bit is asserted.
5	RO	0	**IBSTAT5:** IBIST Status port 5 0: IBIST either has not started the first time or it is still running. 1: IBIST is done. This bit will be cleared by hardware when the start bit is asserted.
4	RO	0	**IBSTAT4:** IBIST Status port 4 0: IBIST either has not started the first time or it is still running. 1: IBIST is done. This bit will be cleared by hardware when the start bit is asserted.
3	RO	0	**IBSTAT3:** IBIST Status port 3 0: IBIST either has not started the first time or it is still running. 1: IBIST is done. This bit will be cleared by hardware when the start bit is asserted.
2	RO	0	**IBSTAT2:** IBIST Status port 2 0: IBIST either has not started the first time or it is still running. 1: IBIST is done. This bit will be cleared by hardware when the start bit is asserted.
1	RV	0	*Reserved*.
0	RO	0	**IBSTAT0: IBIST Status port 0** 0: IBIST either has not started the first time or it is still running. 1: IBIST is done. This bit will be cleared by hardware when the start bit is asserted.

3.11.13.3 DIO0IBERR: PCI Express Intel IBIST Error Register

Device: 0
Function: 0
Offset: 395h

Bit	Attr	Default	Description
7	RWC	0	**P7ERRDET: Error Detected on port 7**
6	RWC	0	**P6ERRDET: Error Detected on port 6**
5	RWC	0	**P5ERRDET: Error detected on port 5**
4	RWC	0	**P4ERRDET: Error detected on port 4**
3	RWC	0	**P3ERRDET: Error Detected on port 3**
2	RWC	0	**P2ERRDET: Error Detected on port 2**
1	RV	0	*Reserved*
0	RWC	0	**P0ERRDET: Error detected on port 0**

3.11.13.4 PEX[7:2,0]IBCTL: PEX Intel IBIST Control Register

This register contains the control bits and status information necessary to operate the Fixed and Open modes of the Intel IBIST logic. The default settings allow the CMM logic to operate with link width of a PEX port. Only valid PCI Express control characters/ symbols are allowed for Intel IBIST testing.

Device: 0, 2-7
Function: 0
Offset: 380h

Bit	Attr	Default	Description
31	RW	0	**SYMTYPSEL3:** Symbol[3] Type Select 1: selects Symbol [3] as a control character 0: selects Symbol [3] to a data character
30	RW	1	**SYMTYPSEL2:** Symbol[2] Type Select 1: selects Symbol [2] as a control character 0: selects Symbol [2] to a data character
29	RW	0	**SYMTYPSEL1:** Symbol[1] Type Select 1: selects Symbol [1] as a control character 0: selects Symbol [1] to a data character
28	RW	1	**SYMTYPSEL0:** Symbol[0] Type Select 1: selects Symbol [0] as a control character 0: selects Symbol [0] to a data character
27:23	RV	0	Reserved
22:14	RO	0h	**ERRVAL:** Error Value This is the raw 9-bit error value captured on the lane that asserted the Error Detected bit (ERRDET) or the Global Error status bit if this register is implemented. The value must be extracted in the datapath before the 10b/8b decoder in order to examine its contents for debugging potential link errors.
13:9	RO	0h	**ERRLNNUM:** Error Lane Number This field indicates which lane reported the error that was detected when ERRDET was asserted. Note: When the number of lanes reporting exceeds 32, this field will show an aliased error lane number and cannot be used to indicate the errant lane. Larger lane indications will require an extended register to display accurate information.

Device: 0, 2-7 Function: 0 Offset: 380h			
Bit	**Attr**	**Default**	**Description**
8	RWCST	0	**ERRDET:** Error Detected A mis-compare between the transmitted symbol and the symbol received on link indicates an error condition occurred. Refer to Error Value, Error Symbol Pointer and Error Symbol Type bit fields for further information about fault locations. This bit is cleared by writing a logic '1' and it remains asserted through reset (sticky). 0: No error detected 1: Error Detected ***Note:*** The error signal that causes this bit to be set should be made available externally to the Intel IBIST logic. It is implementation specific as to how this is accomplished. The purpose is for symbol (bit) error rate testing. It is assumed that this signal is either sent to a performance counter or an external pin for signal assertion accumulation. There is not any error counting resources available in this spec.
7	RW	0	**SUPSKP: Suppress Skips** 0: Skips are still inserted in the Intel IBIST data stream during Intel IBIST test operations. 1: Skip insertion is suppressed
6:4	RW	000	**DSYMINJLNUM:** Delay Symbol Injection Lane Number This selects the Lane number to inject the delay symbol pattern. All 8 values could be valid depending on the setting of the IBEXTCTL.LNMODUEN bit field. This is true regardless of whether this Intel IBIST engine is instantiated for a x4 or a x8 port.
3	RW	1	**AUTOSEQEN:** Automatic Sequencing Enable of Delay Symbol 0: Disable delay symbol auto-sequence. Intel IBIST does not automatically sequence the delay symbol across the width of the link. 1: Enable delay symbol auto-sequencing.
2	RV	0	*Reserved*
1	RW	0	**INITDISP:** Initial Disparity This bit sets the disparity of the first Intel IBIST data pattern symbol. The default is negative meaning that the first symbol transmitted by Tx will have a negative disparity regardless of what the running disparity is. This allows a deterministic pattern set to be transmitted on the link for every Intel IBIST run. If Intel IBIST causes a discontinuous disparity error in the receiver this error can be ignored in the reporting register. It will not affect the operation of the Intel IBIST since it is outside of its domain. Higher levels of software management must be aware that side effects from running Intel IBIST could cause other errors and should they be ignored. 0: Disparity starts as negative 1: Disparity starts as positive
0	RW	0	**IBSTR:** IBIST Start This bit is OR'ed with the global start bit. 0: Stop IBIST 1: Start IBIST

3.11.13.5 PEX[7:2,0]IBSYMBUF: PEX Intel IBIST Symbol Buffer

This register contains the character symbols that are transmitted on the link. Only valid PCI Express control characters/symbols are allowed for Intel IBIST testing.

Device: 0, 2-7 Function: 0 Offset: 384h			
Bit	**Attr**	**Default**	**Description**
31:24	RW	4Ah	**CHARSYM3:** Character Symbol [3] This character is symbol [3] of the four-symbol pattern buffer. The default value is the 8-bit encoding for D10.2.
23:16	RW	BCh	**CHARSYM2:** Character Symbol [2] This character is symbol [3] of the four-symbol pattern buffer. The default value is the 8-bit encoding for K28.5.
15:8	RW	B5h	**CHARSYM1:** Character Symbol [1] This character is symbol [3] of the four-symbol pattern buffer. The default value is the 8-bit encoding for D21.5.
7:0	RW	BCh	**CHARSYM0:** Character Symbol [0] This character is symbol [3] of the four-symbol pattern buffer. The default value is the 8-bit encoding for K28.5.

3.11.13.6 PEX[7:2,0]IBEXTCTL: PEX Intel IBIST Extended Control Register

This register extends the functionality of the Intel IBIST with pattern loop counting, skip character injection, and symbol management. A bit is provided to ignore the count value and loop continuously for port testing. Only valid PCI Express control characters/ symbols are allowed for Intel IBIST testing.

Device: 0, 2-7 Function: 0 Offset: 388h			
Bit	**Attr**	**Default**	**Description**
31:29	RV	0h	*Reserved*
28	RW	0	**FRCENT:** Forced Entry: Setting this bit forces entry into master Intel IBIST loopback state when the Start bit is asserted. No TS1s are sent when enabled. The Intel IBIST is granted direct control of the transmitted path regardless of which state the LTSSM is in. The Intel IBIST state machine sends the contents of the pattern buffer until the stop condition is reached. The receiver isn't expected to perform any error checking and must ignore input symbols. NOTE: For Intel 5000X Chipset MCH the user must reset the component to stop this function. The receiver sets the done condition and the receiver is disabled with this function. 0: Execute normally 1: Force to Loopback state as a 'master' condition.
27:26	RW	01	**LNMODUEN: Lane Modulo Enable for Delay Symbol Injection** 00: No symbols sent on lanes 01: Delay symbols sent on modulo 4 group of lanes across the width of the port. 10: Delay symbols sent on modulo 8 group of lanes across the width of the port. 11: Reserved
25	RW	0	**DISSTOP:** Disable Stop on Error 0: Enable Stop on Error 1: Disable Stop on Error. The Intel IBIST engine continues to run in its current mode in the presence of an error. If an error occurs overwrite the error status collected from a previous error event.
24	RW	0	**LPCON:** Loop Continuously 0: Use loop counter. Test terminates at the end of the global count. 1: Loop symbols continuously.

Device: 0, 2-7 Function: 0 Offset: 388h			
Bit	**Attr**	**Default**	**Description**
23:12	RW	000h	**SKPCNTINT: Skip Count Interval** This register indicates when a skip order sequence is sent on the transmitter. Upon reaching this count the transmitter sends an SOS then clears the skip counter and counting resumes until the next match on the skip count interval. 000: No Skip Ordered Sets are sent on TX. nnn: The number of 8 symbol sets transmitted before a Skip Ordered Set is sent.
11:0	RW	07Fh	**LOOPCNTLIM:** Loop Count Limit This register indicates the number of times the data symbol buffer is looped as a set of 8 symbol times. If LOOPCON is set then this count limit is ignored. 00: No symbols are sent from symbol buffer unless LOOPCON is set. If LOOPCON is cleared and this value is 000h then the transmitter immediately exits out of loopback state by sending EIOS without sending a pattern buffer payload. 01-FFF: 1 to 4095 sets of symbols from the symbol buffer. One set of symbols is defined as either two copies of the contents of the buffer or the modified delayed symbol set.

3.11.13.7 PEX[7:2,0]IBDLYSYM: PEX Intel IBIST Delay Symbol

This register stores the value of the delay symbol used in lane inversion cross-talk testing. Only valid PCI Express control characters/symbols are allowed for Intel IBIST testing.

Device: 0, 2-7 Function: 0 Offset: 38Ch			
Bit	**Attr**	**Default**	**Description**
15:9	RV	0h	*Reserved*
8:0	RW	1BCh	**DLYSYM:** Delay Symbol This is the 9-bit delay symbol value used (default is K28.5).

3.11.13.8 PEX[7:2,0]IBLOOPCNT: PEX Intel IBIST Loop Counter

This register stores the current value of the loop counter.

Device: 0, 2-7 Function: 0 Offset: 38Eh			
Bit	**Attr**	**Default**	**Description**
15:12	RV	0h	*Reserved*
11:0	RO	000h	**LOOPCNTVAL:** Loop Count Value Once the Intel IBIST is engaged, loop counts are incremented when a set of 8 symbols has been received. If an error occurs, this register reflects the loop count value of the errant Rx lane. If there is no error then this register reads 00h. Note: Since each receiver is not deskewed with respect to the Intel IBIST pattern generator we cannot have a coherent loop count value with N number of receivers and only one loop counter. It would require additional logic to select which receiver indicates the count.

3.11.13.9 PEX[7:2,0]IBLNS[3:0]: PEX Intel IBIST Lane Status

This register contains the control bits and status information necessary to perate the Fixed and Open modes of the Intel IBIST logic. The default settings allow the CMM logic to operate with link width of a PEX port. Only valid PCI Express control characters/ symbols are allowed for Intel IBIST testing.

Device: 0, 2-7
Function: 0
Offset: 393h, 392h, 391h, 390h

Bit	Attr	Default	Description
7	RO	0	**ERRPTRTYP:** Error Symbol Pointer Type This bit indicates whether or not the errant symbol pointer was a delay symbol set. If an Intel IBIST engine is implemented with the MISR compare method then this field is reserved. 0: Errant symbol pointer was a DATA symbol set 1: Errant symbol pointer was a DELAY symbol set
6:4	RO	0h	**ERRPTR:** Error Symbol Pointer This value indicates which symbol of the 8 possible symbols sent on the lane, as a set of characters, failed. The value corresponds to position of the set of 8 symbols. If an Intel IBIST engine is implemented with the MISR compare method then this field is reserved.
3	RWC	0	**IBLOOPSTAT:** IBIST Loopback State Status: This bit is set when the Rx received a TS1 with the loopback bit set. Write a logic '1' to clear. 0: IBIST did not receive a TS1 with loopback bit set. 1: IBIST received a TS1 with loopback bit set.
2	RV	0	*Reserved.*
1	RWCST	0	**ERRLNSTAT:** Error Lane Status Error assertion for this lane. Writing a logic '1' will clear this bit. This bit is sticky. 0: No error on this lane 1: Error has occurred on this lane
0	RW	1	**LNSTREN:** Lane Start Enable When the lane is disabled, no electrical transmissions may occur on the Tx driver and the receiver's (Rx) Intel IBIST error reporting is suppressed. This allows the pattern generator and receiver checking logic to function normally if required for design simplicity. But it forces a quite Tx lane for adjacent lane testing. 0: This lane is disabled from Intel IBIST testing, no Tx transmissions and Rx error reporting is suppressed. 1: Lane enabled. Allows the port start bit to begin Intel IBIST symbol operations on this lane.

3.11.13.10 DIO[1:0]SQUELCH_CNT: PCIe Cluster Squelch Count

Device: 4, 0
Function: 0
Offset: 396h

Bit	Attr	Default	Description
15:12	RV	1h	***Reserved***
11	RWST	1h	**Dis_Rx_L1L0s_idle**: Disable automatic shutoff of receivers during L1.Idle/ L0s.Idle power states 1: Disable automatic shut off of receivers during L1 or L0s idle power state entry. (default). 0: Enables Rx shut off. This bit when clear forces the hardware to shut off the Receiver side in BNB during the L0s/L1 power states. ***Note:*** This bit is functional in MCH steppings B3 and newer. Refer to erratum 19 (501621).
10	RWST	0h	**DIS_LANE_LANE_DESKEW: Disable Lane to Lane Deskew** 1: Lane to lane deskew is disabled. Should be set before Intel IBIST is started and cleared after Intel IBIST is stopped 0: Normal link operation. i.e Lane to lane deskew is enabled (default)
9:0	RV	164h	***Reserved***

§

4 System Address Map

The Intel 5000X Chipset supports 36 bits of memory address space. Internally Intel 5000X Chipset carries 40 bits of address into various memory controller components. The processors designed for Intel 5000X Chipset, support only 36 bits of memory addressing and 16 bits of addressable I/O space. However internally the MCH supports 40 bits and several of the MCH memory configuration registers require 40 bit address programming.

There is a legacy (compatibility) memory address space under the 1 MB region that is divided into regions that can be individually controlled with programmable attributes (for example, disable, read/write, write only, or read only). Attribute programming is described in Chapter 3. The Intel 5000X chipset supports several fixed address ranges in addition to the compatibility range. These are:

- Compatibility area below 1 MB
- Interrupt delivery region
- System region in 32 MB just below 4 GB

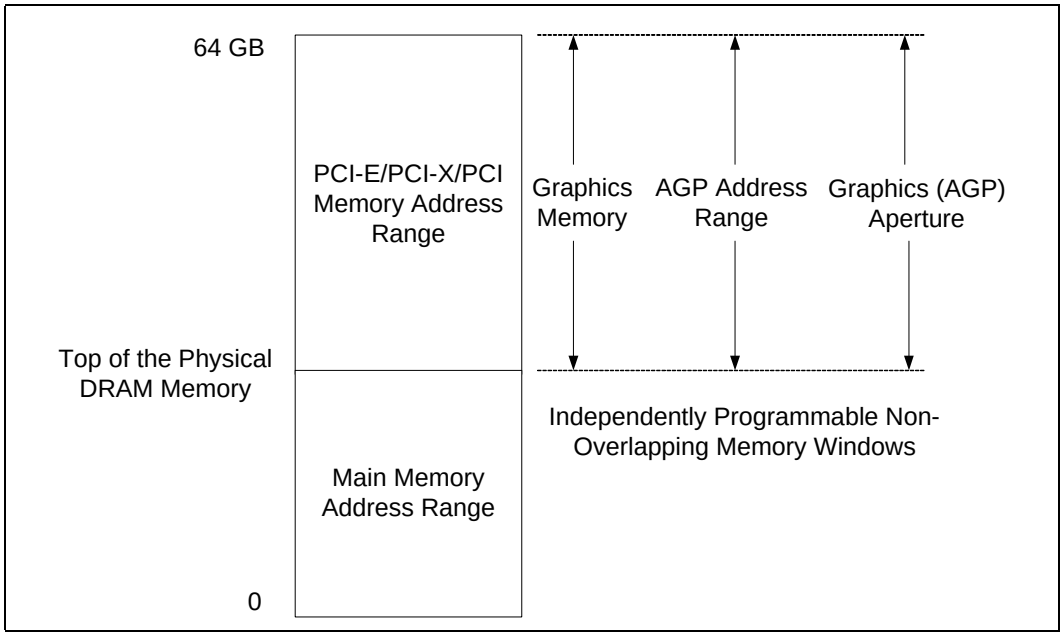
There are several relocatable regions such as the memory mapped I/O region.These regions are controlled by various programmable registers covered in Chapter 3.

This chapter focuses on how the memory space is partitioned and the uses of the separate memory regions.

In the following sections, it is assumed that all of the compatibility memory ranges reside on the ESI/PCI Express/PCI interfaces. VGA address ranges are mapped to PCI Express address space as well. In the absence of more specific references, cycle descriptions referencing PCI should be interpreted as the ESI/PCI interface.

The Intel 5000X chipset memory map includes a number of programmable ranges. All of these ranges must be unique and non-overlapping as shown in Figure 4-1. There are no hardware interlocks to prevent problems in the case of overlapping ranges. Accesses to overlapped ranges may produce indeterminate results. For example, setting HECBASE to all zeros will overlap the MMCFG region and the compatibility region resulting in unpredictable results.

Figure 4-1. System Memory Address Map



4.1 System Memory Address Ranges

The Intel 5000X Chipset platform supports 36 bits (64 GB) of physical memory Or maximum of 40-bit system address with up to 64 GB or physical memory support.

Other address spaces supported by the Intel 5000X Chipset are:

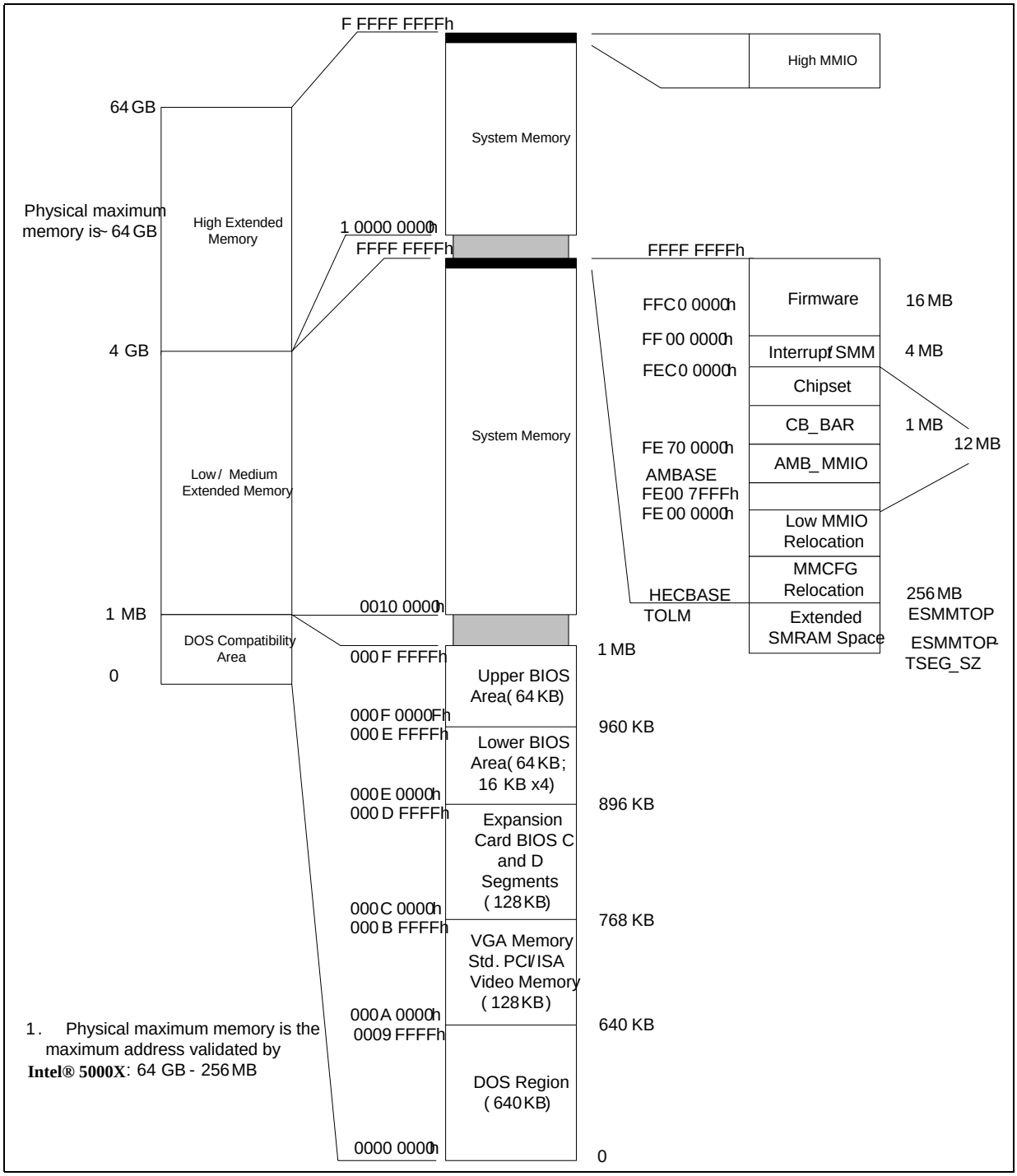
- 36-bit local address supported over the FB-DIMM channels for physical memory space.
- 32 and 64 bit address bit formats supported for PCI Express interfaces.

The chipset treats accesses to various address ranges in different ways. There are fixed ranges like the compatibility region below 1 MB, and variable ranges like the memory mapped I/O range. The locations of these ranges in the memory map are illustrated in Figure 4-2.

4.1.1 32/64-bit addressing

For inbound and outbound writes and reads, the Intel 5000X Chipset MCH supports 64-bit address format. If an outbound transaction's address is a 32-bit address, the Intel 5000X Chipset MCH will issue the transaction with a 32-bit addressing format on PCI Express. Only when the address requires more than 32 bits will the Intel 5000X Chipset MCH initiate transactions with 64-bit address format. It is the responsibility of the software to ensure that the relevant bits are programmed for 64-bits based on the OS limits. (for example, 40-bits for Intel 5000X Chipset MCH).

Figure 4-2. Detailed Memory System Address Map



4.2 Compatibility Area

This is the range from 0 - 1 MB (0 0000h to F FFFFh). Requests to the compatibility area are directed to main memory, the Compatibility Bus (ESI), or the VGA device. Any physical DRAM addresses that would be addressed by requests in this region that are mapped to the Compatibility Bus (ESI) and are not recovered.

DRAM that has a physical address between 0-1 MB must not be recovered or relocated or reflected. This range must always be available to the OS as DRAM, even if at times addresses in this range are sent to the compatibility bus or VGA or other non-DRAM areas.

Addresses below 1 M that are mapped to memory are accessible by the processors and by any I/O bus. The address range below 1 M is divided into five address regions. These regions are:

- 0 - 640 KB MS-DOS Area.
- 640 – 768 KB Video Buffer Area.
- 768 – 896 KB in 16-KB sections (total of eight sections) - Expansion Card BIOS, Segments C and D.
- 896 - 960 KB in 16-KB sections (total of four sections) - Lower Extended System BIOS, Segment E.
- 960 KB–1 MB memory (BIOS Area) - Upper System BIOS, Segment F.

There are fifteen memory segments in the compatibility area. Thirteen of the memory ranges can be enabled or disabled independently for both read and write cycles.

Table 4-1. Memory Segments and Their Attributes

Memory Segments	Attributes	Comments
000000h–09FFFFh	Fixed: mapped to main DRAM	0 to 640 KB – DOS Region
0A0000h–0BFFFFh	Mapped to ESI, x16 graphics port	Video Buffer (physical DRAM configurable as SMM space)
0C0000h–0C3FFFh	WE RE	Add-on BIOS
0C4000h–0C7FFFh	WE RE	Add-on BIOS
0C8000h–0CBFFFh	WE RE	Add-on BIOS
0CC000h–0CFFFFh	WE RE	Add-on BIOS
0D0000h–0D3FFFh	WE RE	Add-on BIOS
0D4000h–0D7FFFh	WE RE	Add-on BIOS
0D8000h–0DBFFFh	WE RE	Add-on BIOS
0DC000h–0DFFFFh	WE RE	Add-on BIOS
0E0000h–0E3FFFh	WE RE	BIOS Extension
0E4000h–0E7FFFh	WE RE	BIOS Extension
0E8000h–0EBFFFh	WE RE	BIOS Extension
0EC000h–0EFFFFh	WE RE	BIOS Extension
0F0000h–0FFFFFh	WE RE	BIOS Area

4.2.1 MS-DOS Area (0 0000h–9 FFFFh)

The MS-DOS area is 640 KB in size and is mapped to main memory controlled by the MCH.

4.2.2 Legacy VGA Ranges (A 0000h–B FFFFh)

The 128 KB Video Graphics Adapter Memory range (A 0000h to B FFFFh) can be mapped to the VGA device which may be on any PCI Express or ESI port, or optionally it can be mapped to main memory (it must be mapped to SMM space). Mapping of this region is controlled by the VGA steering bits. At power on this space is mapped to the ESI port.

Priority for VGA mapping is constant in that the MCH consistently decodes internally mapped devices first. The MCH positively decodes internally mapped devices. This region can be redirected by BIOS to point to any bus which has a VGA card. If the VGAEN bit is set in one of the Intel 5000X Chipset MCH.BCTRL configuration registers associated with the PCI Express port, then transactions in this space are sent to that PCI Express port.

The VGAEN bit can only be set in one and only one of the Intel 5000X Chipset MCH.BCTRL registers. If any VGAEN bits are set, all the ISAEN bits must be set. If the VGAEN bit of a PCI Express port x in the Intel 5000X Chipset MCH is set and BCTRL[x].VGA16bdecode is set to zero, then ISAEN bits of all peer PCI Express ports with valid I/O range (PCICMD.IOAE = 1, IOLIMIT >= IOBASE) in the MCH must be set by software. Otherwise, it is a programming error due to the resulting routing conflict.

If the VGAEN bit of a PCI Express port x in the MCH is set, and BCTRL[x].VGA16bdecode is set to one, and if there is another PCI Express port y (x != y) with valid I/O range including the lowest 4K I/O addresses (PCICMD[y].IOAE = 1, IOLIMIT[y] >= IOBASE[y] = 0000h), BCTRL[y].ISAEN bit must be set to one by software. Otherwise, it is a programming error.

This region is non-cacheable.

Compatible SMRAM Address Range (A 0000h–B FFFFh)

The legacy VGA range may also be used for mapping SMM space. The SMM range (128 KB) can overlay the VGA range in the A and B segments. If the SMM range overlaps an enabled VGA range then the state of the SMMEM# signal determines where accesses to the SMM Range are directed. SMMEM# is a special FSB message bit that uses multiplexed address bit FSBxA[7]#. SMMEM# is valid during the second half of the FSB request phase clock. (the clock in which FSBxADS# is driven asserted).

SMMEM# asserted directs the accesses to the memory and SMMEM# deasserted directs the access to the PCI Express bus where VGA has been mapped.

When compatible SMM space is enabled, SMM-mode processor accesses to this range are routed to physical system DRAM at this address. Non-SMM-mode processor accesses to this range are considered to be to the video buffer area as described above. Graphics port and ESI originated cycles to enabled SMM space are not allowed and are considered to be to the Video Buffer Area.

Monochrome Adapter (MDA) Range (B 0000h–B 7FFFh)

The Intel 5000X Chipset does not support this range.

4.2.3 Expansion Card BIOS Area (C 0000h–D FFFFh)

This 128-KB ISA Expansion Card BIOS covers segments C and D. This region is further divided into eight, 16-KB segments. Each segment can be assigned one of four read/write states: read only, write only, read/write, or disabled. Typically, these blocks are mapped through the MCH and are subtractively decoded to ISA space. Memory that is disabled is not remapped.

Read and write transactions may be directed to different destinations with in the range C 0000h to D FFFFh. Historically, these blocks were used to shadow ISA device BIOS code. For the Intel 5000X Chipset, these regions are used to provide address space to PCI devices requiring memory space below 1 MB. The range is divided into 8 sub-ranges. These ranges are defined by Intel 5000X Chipset MCH.PAM registers. There is a PAM register for each sub-range that defines the routing of reads and writes.

Table 4-2. PAM Settings

PAM [5:4]/1:0]	Write Destination	Read Destination	Result
00	ESI	ESI	Mapped to ESI Port
01	ESI	Main Memory	Memory Write Protect
10	Main Memory	ESI	In-Line Shadowed
11	Main Memory	Main Memory	Mapped to main memory

The power-on default for these segments is mapped read/write to the ESI port (Intel 631xESB/632xESB I/O Controller Hub). Software should not set cacheable memory attributes for any of these ranges, unless both reads and writes are mapped to main memory. Chipset functionality is not guaranteed if this region is cached in any mode other than both reads and writes being mapped to main memory.

For locks to this region, the Intel 5000X Chipset will complete, but does not guarantee the atomicity of locked access to this range when writes and reads are mapped to separate destinations. If inbound accesses are expected, the C and D segments MUST be programmed to send accesses to DRAM.

4.2.4 Lower System BIOS Area (E 0000h–E FFFFh)

This 64-KB area, from E 0000h to E FFFFh, is divided into four, 16-KB segments. Each segment can be assigned independent read and write attributes through the Intel 5000X Chipset MCH.PAM registers. This area can be mapped either the ESI port (Intel 631xESB/632xESB I/O Controller Hub) or to main memory. Historically this area was used for BIOS ROM. Memory segments that are disabled are not remapped elsewhere.

The power-on default for these segments is to map them to the ESI port (Intel 631xESB/632xESB I/O Controller Hub). Software should not set cacheable memory attributes for any of these ranges unless both read and write transactions are mapped to main memory. Chipset functionality is not guaranteed if this region is cached.

For locks to this region, the Intel 5000X Chipset will complete them, but does not guarantee the atomicity of locked access to this range when writes and reads are mapped to separate destinations. If inbound transactions are expected, the E segment MUST be programmed to send these transactions to DRAM.

4.2.5 Upper System BIOS Area (F 0000h–F FFFFh)

This area is a single, 64-KB segment, from E 0000h - F FFFFh. This segment can be assigned read and write attributes through the Intel 5000X Chipset MCH.PAM registers. The power-on default is set to read/write disabled with transactions forwarded to the ESI port (Intel 631xESB/632xESB I/O Controller Hub). By manipulating the read/write attributes, the MCH can “shadow” BIOS into the main system memory. When disabled, this segment is not remapped.

For locks to this region, the Intel 5000X Chipset will complete them, but does not guarantee the atomicity of locked access to this range when writes and reads are mapped to separate destinations. If inbound transactions are expected, the F segment MUST be programmed to send these transactions to DRAM.

4.3 System Memory Area

The low/medium memory regions range from 1 MB to 4 GB. It consists of sub-regions for Firmware, Processor memory mapped functions, and Intel 5000X Chipset-specific registers.

The Extended Memory Area covers from 10 0000h (1 MB) to FFFF FFFFh (4 GB-1) address range and it is divided into the following regions:

- Main System Memory from 1 MB to the Top of Memory; 4-GB system memory.
- PCI Memory space from the Top of Memory to 4 GB with two specific ranges:
- APIC Configuration Space from FEC0 0000h (4 GB–20 MB) to FECF FFFFh and FEE0 0000h to FEEF FFFFh
- High BIOS area is from 16MB to 4 GB - 1

Main System DRAM Address Range (0010 0000h to Top of System Memory)

The address range from 1 MB to the top of system memory is mapped to system memory address range controlled by the MCH. The Top of Main Memory (TOLM) is limited to 4-GB DRAM. All accesses to addresses within this range will be forwarded by the MCH to the system memory.

The MCH provides a maximum system memory address decode space of 4 GB. The MCH does not remap APIC memory space. The MCH does not limit system memory address space in hardware.

4.3.1 System Memory

See Section 4.3.9.

4.3.2 15 MB - 16 MB Window (ISA Hole)

The Intel 5000X Chipset does not support the legacy ISA hole between addresses F0 0000h - FF FFFFh. All transactions to this address range are treated as system memory.

4.3.3 Extended SMRAM Space (TSEG)

SMM space allows system management software to partition a region in main memory to be used by system management software. This region is protected for access by software other than system management software. When the SMM range is enabled, memory in this range is not exposed to the Operating System. The Intel 5000X Chipset

allows accesses to this range only when the SMMEM# signal on the processor bus is asserted with the request. If SMMEM# is deasserted, accesses to the SMM Range are master aborted. If SMMEM# is asserted the access is routed to main memory. Intel 5000X Chipset uses the SMM enable and range registers to determine where to route the access.

Extended SMRAM Space is different than the SMM space defined with in the VGA address space, A 0000h - B FFFFh. This region is controlled by the Intel 5000X Chipset registers Intel 5000X Chipset MCH.EXSMRC.TSEG_SZ and Intel 5000X Chipset MCH.EXSMRTOP.ESMMTOP. The TSEG SMM space starts at ESMMTOP - TSEG_SZ and ends at ESMMTOP. This region may be 512 KB, 1 MB, 2 MB, or 4 MB in size, depending on the TSEG_SZ field. ESMMTOP is relocatable to accommodate software that wishes to configure the TSEG SMM space before MMIO space is known. The ESMMTOP will default to the same default value as Top Of Low Memory (TOLM), defined by the TOLM register.

Intel 5000X Chipset will not support a locked access that crosses an SMM boundary. Firmware should not create data structures that span this boundary. SMM main memory is protected from Inbound accesses.

In order to make cacheable SMM possible, the chipset must accept EWB's and must absorb IWB data regardless of the condition of the SMMEM# pin. The Intel 5000X Chipset MCH will not set the error bit EXSMRAMC.E_SMERR in this case. Because of this, care must be used when attempting to cache SMM space. The chipset/platform cannot protect against processors who attempt to illegally access SMM space that is modified in another processor's cache. Any software that creates such a condition (for example, by corrupting the page table) will jeopardize the protective properties of SMM.

4.3.4 Memory Mapped Configuration (MMCFG) Region

There is one relocatable memory mapped configuration region in the Intel 5000X Chipset MCH. The processor bus address defines the particular configuration register to be accessed. This configuration mechanism is atomic.

The memory mapped configuration region is compatible with the PCI Express enhanced configuration mechanism. The MMCFG region is a 256 MB window that maps to PCI Express registers on both the Intel 5000X Chipset and the south bridge.

The location of this MMCFG window is defined by the Intel 5000X Chipset MCH.HECBASE register. The HECBASE register could also be accessed through a fixed location. The default value of Intel 5000X Chipset MCH.HECBASE maps this region such that there will be no wasted memory that is lost behind it. The default value for the PCI Express registers is the same as the default value of TOLM. If this range is moved, the following recommendations will enable reclaiming the memory that is lost to MMCFG accesses.

1. MMCFG range is mapped to a legal location within the range between TOLM and 4 GB. Since ranges must not overlap other legal ranges, it is safest to put this range between TOLM and the lowest real MMIO range. (The current default is in these ranges) OR
2. Put the region above 4 GB Low/Medium Memory limit and not overlapping above 4 GB MMIO space.

BIOS/software must ensure there are no outstanding configuration accesses or memory accesses to the old and new MMCFG range addresses when relocating this range.

Note: An SMM program can address up to 4 GB of memory. SMM is similar to read-address mode in that there are no privileges or address mapping. The Intel 5000X Chipset MCH allows the relocation of HECBASE above 4GB. However, SMM code cannot access extended configuration space if HECBASE is relocated above 4 GB. This is a CPU limitation. Page Size Extension (PSE) is supported in SMM but Page Address Extension (PAE) support in SMM is currently not in Intel® Xeon processors. Refer to the *IA-32 Intel® Architecture Software Developer's Manual, Vol. 3, Sect. 13.1.*

For more information on the memory mapped configuration mechanism described here, please see the Configuration Map and Access Chapter.

4.3.5 Low Memory Mapped I/O (MMIO)

This is the first of two Intel 5000X Chipset memory mapped I/O ranges. The low memory mapped I/O range is defined to be between Top Of Low Memory, (TOLM) and FE00 0000h. This low MMIO region is further subdivided between the PCI Express and ESI ports. The following table shows the registers used to define the MMIO ranges for each PCI Express/ESI device. These registers are compatible with PCI Express and the PCI to PCI bridge specifications. Note that all subranges must be contained in the low memory mapped I/O range (between TOLM and FE00 0000). In other words, the lowest base address must be above TOLM and the highest LIMIT register must be below FE00_0000. Subranges must also not overlap each other.

Table 4-3. Low Memory Mapped I/O[1]

I/O Port	MCH Base	MCH Limit
ESI	N/A[2]	N/A[2]
PEX2 Memory	MBASE2	MLIMIT2
PEX2 Prefetchable Memory	PMBASE2	PMLIMIT2
PEX3 Memory	MBASE3	MLIMIT3
PEX3 Prefetchable Memory	PMBASE3	PMLIMIT3
PEX4 Memory	MBASE4	MLIMIT4
PEX4 Prefetchable Memory	PMBASE4	PMLIMIT4
PEX5 Memory	MBASE5	MLIMIT5
PEX5 Prefetchable Memory	PMBASE5	PMLIMIT5
PEX6 Memory	MBASE6	MLIMIT6
PEX6 Prefetchable Memory	PMBASE6	PMLIMIT6
PEX7 Memory	MBASE7	MLIMIT7
PEX7 Prefetchable Memory	PMBASE7	PMLIMIT7

Notes:
1. This table assumes Intel 5000X Chipset MCH.PMLU and Intel 5000X Chipset MCH.PMBU are 0's. Otherwise, the prefetchable memory space will be located in high MMIO space.
2. MCH does not need base/limit for Intel 631xESB/632xESB I/O Controller Hub because subtractive decoding will send the accesses to the Intel 631xESB/632xESB I/O Controller Hub. This is OK for software also, since the Intel 631xESB/632xESB I/O Controller Hub is considered part of the same bus as the MCH.

The Intel 5000X Chipset MCH will decode addresses in this range and route them to the appropriate ESI or PCI Express port. If the address is in the low MMIO range, but is not contained in any of the PCI Express base and limit ranges, it will be routed to the ESI.

If the Intel 5000X Chipset MCH.PMLU and Intel 5000X Chipset MCH.PMBU registers are greater than 0, then the corresponding prefetchable region will be located in the high MMIO range instead.

4.3.6 Chipset Specific Range

The address range FE00 0000h - FEBF FFFFh region is reserved for chipset specific functions.

FE00 0000h - FE00 8000h: This range (with size of 128 KB for four FB-DIMM channels; 16 Advanced Memory Buffer (AMB) per channel, 2 KB per AMB), is used for accessing AMB registers. These registers can only be accessed through memory mapped register access mechanism as MMIO. Notice that they are not accessible through CF8/CFC or MMCFG which are used for PCI/PCI Express configuration space registers. This range could be relocated by programming AMBASE register. The AMBASE register could also be accessed through a fixed location.

FE60 0000h - FE6F FFFFh: This range is used for fixed memory mapped Intel 5000X Chipset registers. They are accessible only from the processor bus. These registers are fixed since they are needed early during the boot process. The registers include:

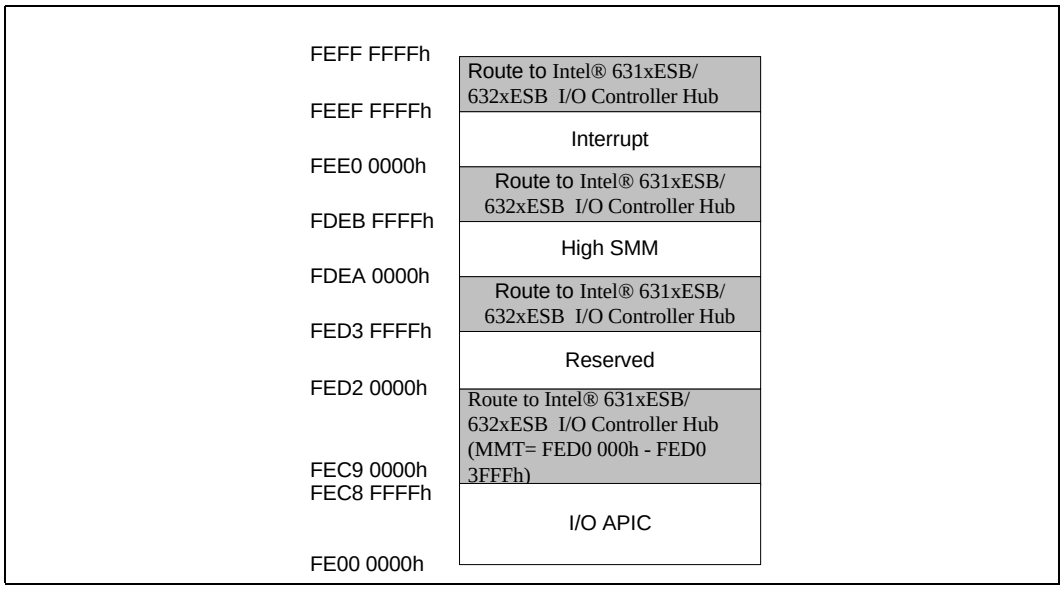
- Four Scratch Pad Registers
- Four Sticky Scratch Pad Registers
- Four Boot flag registers
- HECBASE register for MMCFG
- AMBASE register for AMB memory mapped registers

These registers are described in the Intel 5000X Chipset MCH Configuration Register, Chapter 3. The Intel 5000X Chipset MCH will master abort requests to the remainder of this region unless they map into one of the relocatable regions such as MMCFG. The mechanism for this range can be the same as it is for the memory mapped configuration accesses.

4.3.7 Interrupt/SMM Region

This 4 MB range is used for processor specific applications. This region lies between FEC0 0000h and FEFF FFFFh and is split into four 1 MB segments.

Figure 4-3. Interrupt /SMM Region



This region is used to support various processor and system functions. These functions include I/O APIC control range which is used to communicate with I/O APIC controllers located on Intel® 6700PXH 64-bit PCI Hub and Intel 631xESB/632xESB I/O Controller Hub devices. The high SMM range is enabled under register control. Transactions directed to this range are redirected to physical memory located in the compatible (legacy) SMM space; 0A 0000h - 0B FFFFh. The interrupt range is used to deliver interrupts. Memory read or write transactions from the processor are illegal.

4.3.7.1 I/O APIC Controller Range

This address range FEC0 0000h to FEC8 FFFF is used to communicate with the IOAPIC controllers in the Intel 6700PXH 64-bit PCI Hub or Intel 631xESB/632xESB I/O Controller Hub devices.

The APIC ranges are hard coded. Reads and writes to each IOAPIC region should be sent to the appropriate ESI or PCI Express port as indicated below.

Table 4-4. I/O APIC Address Mapping

IOAPIC0 (ESI)	FEC0 0000h to FEC7 FFFFh
IOAPIC1 (PEX2)	FEC8 0000h to FEC8 0FFFh
IOAPIC2 (PEX3)	FEC8 1000h to FEC8 1FFFh
IOAPIC3 (PEX4)	FEC8 2000h to FEC8 2FFFh
IOAPIC4 (PEX5)	FEC8 3000h to FEC8 3FFFh
IOAPIC5 (PEX6)	FEC8 4000h to FEC8 4FFFh
IOAPIC6 (PEX7)	FEC8 5000h to FEC8 5FFFh
Reserved (Intel 631xESB/632xESB I/O Controller Hub for master abort)	FEC0 6000h to FEC8 FFFFh

For Hot-Plug I/O APIC support, it is recommended that software use the standard MMIO range to communicate with the Intel 6700PXH 64 bit PCI Hub. To accomplish this, the Intel 6700PXH 64-bit PCI Hub.MBAR and/or Intel 6700PXH 64 bit PCI Hub.XAPIC_BASE_ADDRESS_REG must be programmed within the PCI Express device MMIO region.

Inbound accesses to this memory range should also be routed to the I/O APIC controllers. This could happen if software configures MSI devices to send MSIs to an I/O APIC controller.

4.3.7.2 High SMM Range

If high SMM space is enabled by EXSMRC.H_SMRAME, then requests to the address range from FEDA 0000h to FEDB FFFFh will be aliased down to the physical address of A 0000h to B FFFFh. The HIGHSMM space allows cacheable accesses to the compatible (legacy) SMM space. In this range, the chipset will accept EWBs (BWLs) regardless of the SMMEM# pin. Also, if there is an implicit write back (HITM with data), the chipset will update memory with the new data (regardless of the SMMEM# pin). Note that if the HIGHSMM space is enabled, the aliased SMM space of 0A 0000h - 0B FFFFh will be disabled.

Note: In order to make cacheable SMM possible, the chipset must accept EWBs (BWLs) and must absorb IWB (HITM) data regardless of the condition of the SMMEM# pin. Because of this, care must be used when attempting to cache SMM space. The chipset/platform **cannot protect** against processors who attempt to illegally access SMM space that is modified in another processor's cache. Any software that creates such a condition (for

example, by corrupting the page table) will jeopardize the protective properties of SMM.

4.3.7.3 Interrupt Range

Requests to the address range FEE0 0000h to FEEF FFFFh are used to deliver interrupts. Memory reads or write transactions to this range are illegal from the processor. The processor issues interrupt transactions to this range. Inbound interrupt requests from the PCI Express devices in the form of memory writes are converted by the MCH to processor bus interrupt requests.

4.3.7.4 Reserved Ranges

The Intel 5000X Chipset MCH will master abort requests to the addresses in the interrupt/reserved range (FEC0 0000h - FEFF FFFFh) which are not specified. This can be done by sending the request to the compatibility bus (ESI) to be master aborted.

4.3.7.5 Firmware Range

The Intel 5000X Chipset platform allocates 16 MB of firmware space from FF00 0000h to FFFF FFFFh. Requests in this range are directed to the Compatibility Bus. The Intel 631xESB/632xESB I/O Controller Hub will route these to its FWH interface. This range is accessible from any processor bus.

4.3.8 High Extended Memory

This is the range above 4 GB. The range from 4 GB to Intel 5000X Chipset MCH.MIR[2].LIMIT is mapped to system memory. There can also be a memory mapped I/O region that is located at the top of the address space. (Just below 1 TB).

4.3.8.1 System Memory

See Section 4.3.9

4.3.8.2 High MMIO

The high memory mapped I/O region is located above the top of memory as defined by Intel 5000X Chipset MCH.MIR[2].LIMIT. These Intel 5000X Chipset MCH.PMBU and Intel 5000X Chipset MCH.PMLU registers in each PCI Express configuration device determine whether there is memory mapped I/O space above the top of memory. If an access is above MIR[2].LIMIT and it falls within the Intel 5000X Chipset MCH.PMBU+PMBASE and Intel 5000X Chipset MCH.PMLU+PMLIMIT range, it should be routed to the appropriate PCI Express port. For accesses above MIR[2].LIMIT (and above 4 GB) that are not in a high MMIO region, they should be master aborted.

4.3.8.3 CB_BAR MMIO

The integrated DMA device has a 1 KB MMIO space with a default range from FE70 0000h to FE70 03FFh. This range can be relocated by programming CB_BAR register. This range could be used as a private MMIO space by software.

4.3.8.4 Extended Memory

The range of memory just below 4 GB from TOLM to 4 GB (Low MMIO, Chipset, Interrupt/SMM/LT) does not map to memory. If the DRAM memory, behind the TOLM to 4 GB range, is not relocated, it will be unused.

The Intel 5000X Chipset MCH uses MIR[2].LIMIT to indicate the top of usable memory. Note that ESMMTOP cannot be greater than TOLM.

4.3.9 Main Memory Region

4.3.9.1 Application of Coherency Protocol

The Intel 5000X Chipset MCH applies the coherency protocol to all accesses to main memory. Application of the coherency protocol includes snooping the other processor bus.

Two exceptions to this rule are the Expansion Card BIOS area, 0C 0000h - 0F FFFFh and the legacy SMM, 0A 0000h - 0B FFFFh, range. The Expansion Card BIOS area 0C 0000h - 0F FFFFh may not necessarily route both reads and writes to memory, the legacy SMM range, 0A 0000h - 0B FFFFh, may target non-memory when not in SMM mode. The coherency protocol is not applied to these two exceptions.

4.3.9.2 Routing Memory Requests

When a request appears on the processor bus, ESI port, or PCI Express link, and it does not fall in any of the previously mentioned regions, it is compared against the MIR.LIMIT registers in the MCH.

The Intel 5000X Chipset MCH.MIR.LIMIT registers will decode an access into a specific interleaving range. Within the interleaving range, the Intel 5000X Chipset MCH.MIR.LIMIT register indicates which FB-DIMM memory branch the address is associated with. In the event that a mirroring event is occurring, memory writes are associated with both FB-DIMM branches.

4.4 Memory Address Disposition

The following section presents a summary of address dispositions for the Intel 5000X Chipset MCH.

4.4.1 Registers Used for Address Routing

Table 4-5 is a summary of the registers used to control memory address disposition. These registers are described in detail in Section 3.

Table 4-5. Intel 5000X Chipset MCH Memory Mapping Registers (Sheet 1 of 2)

Name	Function
MIR[2:0]	Memory Interleaving Registers (FB-DIMM Branch Interleaving)
AMIR[2:0]	Scratch pad register for software to use related to memory interleaving. For example, software can write MMIO gap adjusted limits here to aid in subsequent memory RAS operations.
PAM[6:0]	Defines attributes for ranges in the C and D segments. Supports shadowing by routing reads and writes to memory of I/O
SMRAMC	SMM Control
EXSMRC, EXSMRAMC	Extended SMM Control
EXSMRTOP	Top of extended SMM memory
BCTRL	Contains VGAEN and ISAEN for each PCI Express.
TOLM	Top of low memory. Everything between TOLM and 4 GB will not be sent to memory.

Table 4-5. Intel 5000X Chipset MCH Memory Mapping Registers (Sheet 2 of 2)

Name	Function
HECBASE	Base of the memory mapped configuration region that maps to all PCI Express registers
MBASE (dev 2-7)	Base address for memory mapped I/O to PCI Express ports 2 - 7
MLIMIT (dev 2-7)	Limit address for memory mapped I/O to PCI Express ports 2 - 7
PMBASE (dev 2-7)	Base address for memory mapped I/O to prefetchable memory of PCI Express ports 2-7[a]
PMLIMIT (dev 2-7)	Limit address for memory mapped I/O to prefetchable memory of PCI Express ports 2-7
PMBU (dev 2-7)	Prefetchable Memory Base (Upper 32 bits) - Upper address bits to the base address of prefetchable memory space. If the prefetchable memory is below 4GB, this register will be set to all 0's.
PMLU (dev 2-7)	Prefetchable Memory Limit (Upper 32 bits) - Upper address bits to the limit address of prefetchable memory space. If the prefetchable memory is below 4GB, this register will be set to all 0's.
PCICMD (dev 2-7)	MSE (Memory Space Enable) bit enables the memory and prefetchable ranges.

Notes:

a. The chipset treats memory and prefetchable memory the same. These are just considered 2 apertures to the PCI Express port.

4.4.2 Address Disposition for Processor

The following tables define the address disposition for the Intel 5000X Chipset MCH. Table 4-6 defines the disposition of outbound requests entering the Intel 5000X Chipset MCH on the processor bus. Table 4-10 defines the disposition of inbound requests entering the Intel 5000X Chipset MCH on an I/O bus. For address dispositions of PCI Express/ESI devices, please refer to the respective product specifications for the Intel 6700PXH 64-bit PCI Hub or Intel 631xESB/632xESB I/O Controller Hub.

Table 4-6. Address Disposition for Processor (Sheet 1 of 2)

Address Range	Conditions	BNB Behavior
DOS	0 to 09FFFFh	Coherent Request to Main Memory. Route to main memory according to Intel 5000X Chipset MCH.MIR registers. Apply Coherence Protocol.
SMM/VGA	0A0000h to 0BFFFFh	see Table 4-8 and Table 4-9.
C and D BIOS segments	0C0000h to 0DFFFFh and PAM=11	Non-coherent request to main memory. Rout to appropriate FB-DIMM device according to Intel 5000X Chipset MCH.MIR registers.
	Write to 0C0000h to 0DFFFFh and PAM=10	
	Read to 0C0000h to 0DFFFFh and PAM=01	
	Read to 0C0000h to 0DFFFFh and PAM=10	Issue request to ESI.
	Write to 0C0000h to 0DFFFFh and PAM=01	
	0C0000h to 0DFFFFh and PAM=00	

Table 4-6. Address Disposition for Processor (Sheet 2 of 2)

Address Range	Conditions	BNB Behavior
E and F BIOS segments	0E0000h to 0FFFFFh and PAM=11	Non-coherent request to main memory. Rout to appropriate FB-DIMM device according to Intel 5000X Chipset MCH.MIR registers.
	Write to 0E0000h to 0FFFFFh and PAM=10	
	Read to 0E0000h to 0FFFFFh and PAM=01	
	Read to 0E0000h to 0FFFFFh and PAM=10	Issue request to ESI.
	Write to 0E0000h to 0FFFFFh and PAM=01	
	0E0000h to 0FFFFFh and PAM=00	
Low/Medium Memory	10_0000 <= Addr < TOLM	Coherent request to main memory. Route to main memory according to Intel 5000X Chipset MCH.MIR registers. Coherence protocol is applied. Note: the extended SMRAM space is within this range.
Extended SMRAM Space	ESMMTOP-TSEG_SZ <= Addr < ESMMTOP	see Table 4-8 and Table 4-9.
Low MMIO	TOLM <= Addr < FE00_0000 and falls into a legal BASE/LIMIT range	Request to PCI Express based on <MBASE/MLIMIT and PMBASE/PMLIMIT> registers.
	TOLM <= Addr < FE00_0000 and not in a legal BASE/LIMIT range	Send to ESI to be master aborted.
PCI Express MMCFG	HECBASE <= Addr < HECBASE+256MB	Convert to a configuration access and route according to the Configuration Access Disposition.
Intel® 5000X specific	FE00_0000h to FEBF_FFFFh AND valid BNB memory mapped register address plus AMB targeted addresses	Issue configuration access to memory mapped register inside BNB or to the FB-DIMM based on the context.
	FE00_0000h to FEBF_FFFFh AND (NOT a valid BNB memory mapped register address or NOT a valid AMB targeted address)	Send to ESI to be master aborted.
I/O APIC registers	FEC0_0000 to FEC8_FFFFh	Non-coherent request to PCI Express or ESI based on Table 4-4.
Intel 631xESB/ 632xESB I/O Controller Hub / Intel 631xESB/ 632xESB I/O Controller Hub timers	FEC9_0000h to FED1_FFFF	Issue request to ESI.
High SMM	FEDA_0000h to FEDB_FFFF	see Table 4-8 and Table 4-9.
Interrupt	interrupt transaction to FEE0_0000h to FEEF_FFFFh (not really memory space)	Route to appropriate FSB(s).
	memory transaction to FEE0_0000h to FEEF_FFFFh	Send to ESI to be master aborted.
Firmware	FF00_0000h to FFFF_FFFFh	Issue request to ESI.
High Memory	1_0000_0000 to MIR[2].LIMIT (max FF_FFFF_FFFF)	Coherent request to main memory. Route to main memory according to Intel 5000X Chipset MCH.MIR registers. Coherence protocol is applied.
High MMIO	PMBU+PMBASE <= Addr <= PMLU+PMLIMIT	Route request to appropriate PCI Express port
All others	All Others (subtractive decoding)	Issue request to ESI.

4.4.2.1 Access to SMM Space (Processor Only)

Accesses to SMM space are restricted to processors, inbound transactions are prohibited. Inbound transactions to enabled SMM space are not allowed and Intel 5000X Chipset MCH will set Intel 5000X Chipset MCH.EXSMRAMC.E_SMERR bit. The following table defines when a SMM range is enabled. All the enable bits: G_SMRAME, H_SMRAM_EN, and TSEG_EN are located in the Intel 5000X Chipset MCH.EXSMRC register.

Table 4-7. Enabled SMM Ranges

Global Enable G_SMRAME	High SMM Enable H_SMRAM_EN	TSEG Enable TSEG_EN	Legacy SMM Enabled?	HIGH SMM Enabled?	Extended SMRAM Space (TSEG) Enabled?
0	X	X	No	No	No
1	0	0	Yes	No	No
1	0	1	Yes	No	Yes
1	1	0	No	Yes	No
1	1	1	No	Yes	Yes

The processor bus has a SMMEM# signal that qualifies the request asserted as having access to a system management memory. The SMM register defines SMM space that may fall in one of three ranges: legacy SMRAM, Extended SMRAM Space (TSEG), or High SMRAM Space (H_SMM). Table 4-8 defines the access control of SMM memory regions from processors.

Table 4-8. SMM Memory Region Access Control from Processor

G_SMRAME	D_LCK	D_CLS	D_OPEN	SMMEM#	Code Access to SMM Memory[a]	Data Access to SMM Memory[b]
0[c]	x	x	x	x	no	no
1	0	x	0	0	no	no
1	0	0	0	1	yes	yes
1	0	0	1	x	yes	yes
1	0	1	0	1	yes	no (legacy SMM) yes (H_SMM, TSEG)
1[d]	x	1	1	x	illegal settings	illegal settings
1	1	0	x	0	no	no
1	1	1	0	0	no	no
1	1	0	x	1	yes	yes
1	1	1	0	1	yes	no (legacy SMM) yes (H_SMM, TSEG)

Notes:

a. BRLC
b. Data access transaction other than BRLC
c. For access to TSEG region (address range between ESMMTOP - TSEG_SZ and ESMMTOP), Intel 5000X Chipset MCH will route to identical system memory by definition (as TSEG is not enabled).
d. It is a programming error if D_CLS and D_OPEN are both set to 1, Intel 5000X Chipset MCH's behavior is undefined. Intel 5000X Chipset MCH could master abort SMM access.

The BNB prevents illegal processor access to SMM memory. This is accomplished by routing memory requests from processors as a function of transaction request address, code or data access, the SMMEM# signal accompanying request and the settings of the Intel 5000X Chipset MCH.SMRAMC, Intel 5000X Chipset MCH.EXSMRC, and Intel 5000X Chipset MCH.BCTRL registers. Table 4-9 defines Intel 5000X Chipset MCH's routing for each case. Illegal accesses are either routed to the ESI bus where they are Master Aborted or are blocked with error flagging. SMMEM# only affects BNB behavior if it falls in an enabled SMM space. Note that the D_CLS only applies to the legacy (A_0000-B_FFFF) SMM region. The bold values indicate the reason SMM access was granted or denied.

Note: If a spurious inbound access targets the enabled SMM range (viz., legacy, High SMM Memory and Extended SMRAM (T-segment)), then it will be Master-aborted. The EXSMRAMC.E_SMERR register field (Invalid SMRAM) is set for accesses to the High SMM Memory and Extended SMRAM (T-segment)). Refer to Table 4-10.

Table 4-9. Decoding Processor Requests to SMM and VGA Spaces

SMM region	Transaction Address Range	SMM Memory Address Range	SMM Access Control [a]	G_SMRAME	H_SMRAME	T_EN	EWB/IWB	Routing
Legacy VGA/SMM [b]	A_0000h to B_FFFFh	A_0000h to B_FFFFh	x	0	x	x	x	to the VGA-enabled port (in BCTRL); otherwise, ESI[c]
			yes	1	**1**	x	x	
			no	1	x	x	x	
			yes	1	0	x	x	to SMM memory
Extended SMRAM (TSEG)	ESMMTOP -TSEG_SZ to ESMMTOP	ESMMTOP -TSEG_SZ to ESMMTOP	x	0	x	x	x	to identical system memory by definition
			x	1	x	0	x	
			yes	1	x	1	x	to SMM memory
			no	1	x	1	1	
			no	1	x	1	0	block access: master abort set EXSMRAMC.E_SMERR
High SMM	FEDA_0000h to FEDB_FFFFh	A_0000h to B_FFFFh	x	0	x	x	x	to ESI (where access will be master aborted)
			x	1	**0**	x	x	
			yes	1	1	x	x	to SMM memory[d]
			no	1	1	x	1	
			no	1	1	x	0	block access: master abort set EXSMRAMC.E_SMERR

Notes:

a. SMM memory access control, see Table 4-8.
b. Software must not cache this region.
c. One and only one BCTRL can set the VGAEN; otherwise, send to ESI.
d. Notice this range is mapped into legacy SMM range (A_0000h to B_FFFFh).

4.4.3 Inbound Transactions

In general, inbound I/O transactions are decoded and dispositioned similarly to processor transactions. The key differences are in SMM space, memory mapped configuration space, and interrupts. Inbound transaction targeting at itself will be master aborted.

Note that inbound accesses to the SMM region must be handled in such a way that FSB snooping and associated potential implicit writebacks are avoided. This is necessary to prevent compromising SMM data by returning real content to the I/O subsystem. Note also that DMA engine is treated as an I/O device, thus accesses initiated by the DMA engine are considered as inbound accesses.

For all table entries where an access is forwarded to ESI to be master aborted, if an access comes from ESI, the Intel 5000X Chipset MCH ESI may master abort a transaction without forwarding it back to the ESI.

Table 4-10. Address Disposition for Inbound Transactions (Sheet 1 of 2)

Address Range	Conditions	Intel 5000X Chipset Behavior
DOS	0 to 09FFFFh	Coherent Request to Main Memory. Route to main memory according to Intel 5000X Chipset MCH.MIR registers. Apply Coherence Protocol.
SMM/VGA	0A0000h to 0BFFFFh, and VGAEN=0	Send to ESI to be master aborted. Set EXSMRAMC.E_SMERR
	0A0000h to 0BFFFFh and VGAEN=1	Non-coherent read/write request to the decoded PCI Express or to ESI based on BCTRL[a]
C, D, E, and F BIOS segments	0C0000h to 0FFFFFh and PAM=11[b]	Non-coherent request to main memory. (Coherency does not need to be guaranteed. Coherency protocol can be followed if it simplifies implementation.) Route to appropriate FB-DIMM according to Intel 5000X Chipset MCH.MIR registers.
Low/Medium Memory	10_0000 <= Addr < ESMMTOP - TSEG_SZ	Coherent Request to Main Memory. Route to main memory according to Intel 5000X Chipset MCH.MIR registers. Apply Coherence Protocol.
Extended SMRAM Space	ESMMTOP -TSEG_SZ <= Addr < ESMMTOP	Send to system memory if G_SMRAME = 0 or (G_SMRAME = 1 and T_EN = 0); otherwise Send to ESI to be master aborted. Set EXSMRAMC.E_SMERR bit
Low MMIO	TOLM <= Addr < FE00_0000 and falls into a legal BASE/LIMIT range	Request to PCI Express based on <MBASE/MLIMIT and PMBASE/PMLIMIT> registers.
	TOLM <= Addr < FE00_0000 and not in a legal BASE/LIMIT range	Send to ESI to be master aborted.
PCI Express MMCFG	HECBASE <= Addr < HECBASE+256MB	Inbound MMCFG access is not allowed and will be aborted.
Intel 5000X specific	FE00_0000h to FEBF_FFFFh AND valid BNB memory mapped register address	Inbound MMCFG access is not allowed and will be aborted.
	FE00_0000h to FEBF_FFFFh AND NOT a valid BNB memory mapped register address	Send to ESI to be master aborted.
I/O APIC registers	FEC0_0000 to FEC8_FFFFh	Non-coherent request to PCI Express or ESI based on Table 4-4
Intel 631xESB/ 632xESB I/O Controller Hub / Intel 631xESB/ 632xESB I/O Controller Hub timers	FEC9_0000h to FED1_FFFF	Issue request to ESI.
High SMM	FEDA_0000h to FEDB_FFFF	Send to ESI to be master aborted. Set EXSMRAMC.E_SMERR bit
Interrupt	Inbound write to FEE0_0000h - FEEF_FFFFh	Route to appropriate FSB(s). See Interrupt Chapter for details on interrupt routing.
	memory transaction (other than write) to FEE0_0000h - FEEF_FFFFh	Send to ESI to be master aborted.
Firmware	FF00_0000h to FFFF_FFFFh	Master abort

Table 4-10. Address Disposition for Inbound Transactions (Sheet 2 of 2)

Address Range	Conditions	Intel 5000X Chipset Behavior
High Memory	1_0000_0000 to MIR[2].LIMIT (max FF_FFFF_FFFF)	Coherent Request to Main Memory. Route to main memory according to Intel 5000X Chipset MCH.MIR registers. Apply Coherence Protocol.
High MMIO	PMBU+PMBASE <= Addr <= PMLU+PMLIMIT	Route request to appropriate PCI Express port
All others	All Others (subtractive decoding)	Issue request to ESI.

Notes:

a. One and only one BCTRL can set the VGAEN; otherwise, send to ESI for master abort.
b. Other combinations of PAM's are not allowed if inbound accesses to this region can occur. Just like Cayuse, chipset functionality is not guaranteed.

4.5 I/O Address Map

The I/O address map is separate from the memory map and is primarily used to support legacy code/drivers that use I/O mapped accesses rather than memory mapped I/O accesses. Except for the special addresses listed in Section 4.5.1, I/O accesses are decoded by range and sent to the appropriate ESI/PCI-Express port, which will route the I/O access to the appropriate device.

4.5.1 Special I/O Addresses

There are two classes of I/O addresses that are specifically decoded by the Intel 5000X Chipset MCH:

- I/O addresses used for VGA controllers.
- I/O addresses used for the PCI Configuration Space Enable (CSE) protocol. The I/O addresses 0CF8h and 0CFCh are specifically decoded as part of the CSE protocol.

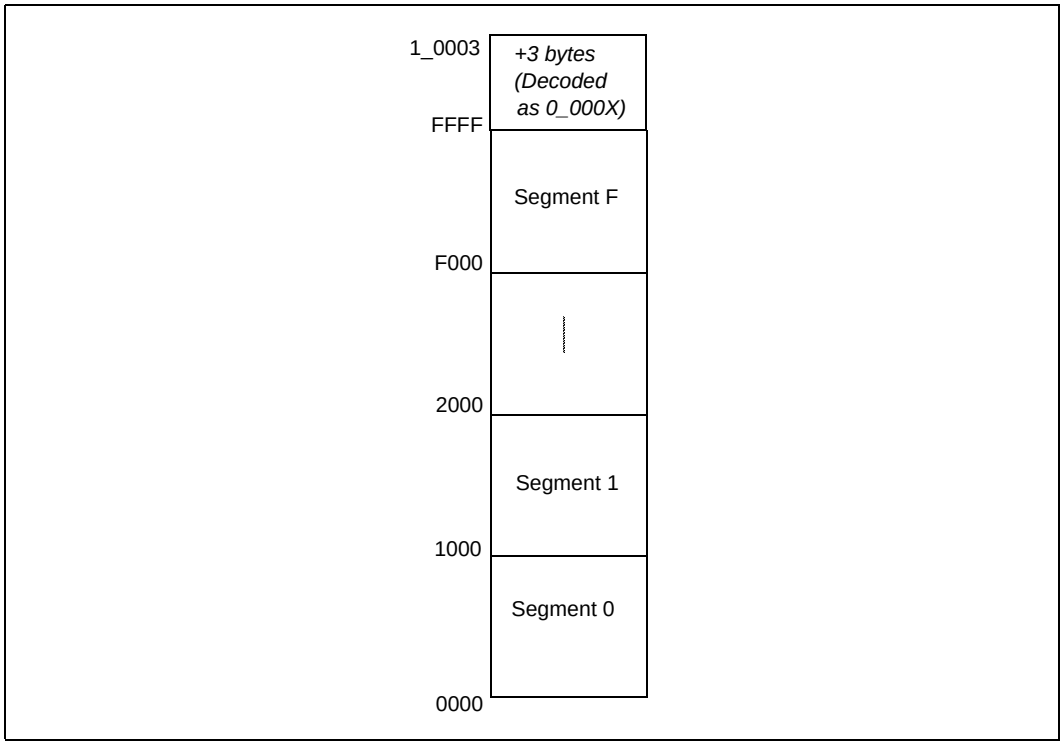
Historically, the 64 K I/O space actually was 64 K+3 bytes. For the extra three bytes, A#[16] is asserted on FSB. The Product Name decodes only A#[15:3] when the request encoding indicates an I/O cycle. Therefore first three byte I/O accesses with A#[16] asserted are decoded as if they were accesses to the first three bytes starting from I/O addresses 0 (wrap-around the 64 KB line). A[16] is not forwarded by Intel 5000X Chipset MCH.

At power-on, all I/O accesses are mapped to the ESI.

4.5.2 Outbound I/O Access

The Intel 5000X Chipset MCH chipset allows I/O addresses to be mapped to resources supported on the I/O buses underneath the BNB. This I/O space is partitioned into 16 4 KB segments. Each of PCI-Express port can have from 1 to 16 consecutive segments mapped to it by programming its IOBASE and IOLIM registers. Each PCI-Express port must be assigned contiguous segments. The lowest segment, from 0 to 0FFFh, should be programmed to send to the ESI for compatibility.

Figure 4-4. System I/O Address Space



4.5.2.1 Outbound I/O Accesses Routing

The BNB applies these routing rules **in the following order**:
(A[2:0] for the following is not physically present on the processor bus, but are calculated from BE[7:0]).

1. I/O addresses used for VGA controllers on PCI-Express:
   If PCICMD[y].IOAE and BCTRL[y].VGAEN of PCI-Express port y are set to 1 and BCTRL[y].VGA16bdecode = **0**, then I/O accesses with the following VGA addresses will be forwarded to PCI-Express port y: A[9:0] (A[15:10] are ignored for this decode since BCTRL[y].VGA16bdecode is set to 0) = 3B0h - 3BBh, 3C0h - 3DFh if every addressed byte is within these two ranges. For example, a two byte read starting at X3BBh includes X3BB -X3BCh. (X can be any hex number since A[15:10] are ignored) Since the second byte with A[9:0] = 3BCh is not within these ranges, the access is not routed to port y.

   If PCICMD[y].IOAE and BCTRL[y].VGAEN of port y are set to 1 and BCTRL[y].VGA16bdecode = **1**, then I/O accesses with the following VGA addresses will be forwarded to PCI-Express port y: A[**15**:0] = **0**3B0h - **0**3BBh, **0**3C0h - **0**3DFh if every addressed byte is within these two ranges. For example, a four byte I/O read starting at F3B0h includes F3B0 - F3B3h are not within these ranges, the access is not routed to port y.

   Note that software should program PEXCMDs and BCTRLs to ensure that at most only one port is allowed to forward these accesses with VGA addresses. It is a programming error if more than one port are programmed to forward accesses with VGA addresses.

2. Configuration accesses: If a request is a DW accesses to 0CF8h (See CFGADR register) or
   1-4 B accesses to 0CFCh (See CFGDAT register) with configuration space enabled (See CFGE bit, bit 31, of CFGADR register), the request is considered a

configuration access. Configuration accesses are routed based on the bus and device numbers as programmed by software.

3. ISA Aliases: If the PCICMD[y].IOAE and BCTRL[y].ISAEN are set to 1 for a PCI-Express port y and the I/O address falls within (IOBASE[y], IOLIMIT[y]) and if the addresses are X100-X3FFh, X500-X7FFh, X900-XBFF, and XD00-XFFFh (X can be any hex number) will result in the access being sent out to the ESI (Intel 631xESB/632xESB I/O Controller Hub). This is the top 768 B in each 1 KB block.
4. I/O defined by IOBASE/IOLIMIT: If PCICMD[y].IOAE is set for a given PCI-Express port and the I/O address falls in this range: (IOBASE[y] <= address <= IOLIMIT[y]) for that port, then the access will be routed to the PCI-Express port y.
5. Otherwise, the I/O Read/Write is sent to ESI (Intel 631xESB/632xESB I/O Controller Hub).

4.5.3 Inbound I/O Access

Inbound I/Os are supported only for peer to peer accesses and are decoded the same as processor initiated I/Os. Inbound I/O transaction for configuration access to the Intel 5000X Chipset MCH are not supported and will be routed as normal inbound I/O transaction.

Inbound requests (memory, I/O) received from a PCI Express port and which target the same port as destination will be Master Aborted.

4.6 Configuration Space

All chipset registers are represented in the memory address map. In addition, some registers are also mapped as PCI registers in PCI configuration space.These adhere to the *PCI Local Bus Specification*, Revision 2.2 .

The memory mapped configuration space is described in Section 4.3.4. Individual register maps are in the registers chapters of the Intel 5000X Chipset MCH Component Specifications.

If a CPU issues a zero length configuration cycle accessing the Intel 5000X Chipset MCH's internal configuration space registers or the CB_BAR/AMB Memory mapped area, then it will be completed on the FSB "in order" with no data.

4.7 I/O Address Map

The I/O address map is separate from the memory map and is primarily used to support legacy code/drivers that use I/O mapped accesses rather than memory mapped I/O accesses. Except for the special addresses listed in "Special I/O addresses" on page 246, I/O accesses are decoded by range and sent to the appropriate ESI/PCI Express port, which will route the I/O access to the appropriate device.

4.7.1 Special I/O Addresses

There are two classes of I/O addresses that are specifically decoded by the Intel 5000X Chipset MCH:

- I/O addresses used for VGA controllers.
- I/O addresses used for the PCI Configuration Space Enable (CSE) protocol. The I/O addresses 0CF8h and 0CFCh are specifically decoded as part of the CSE protocol.

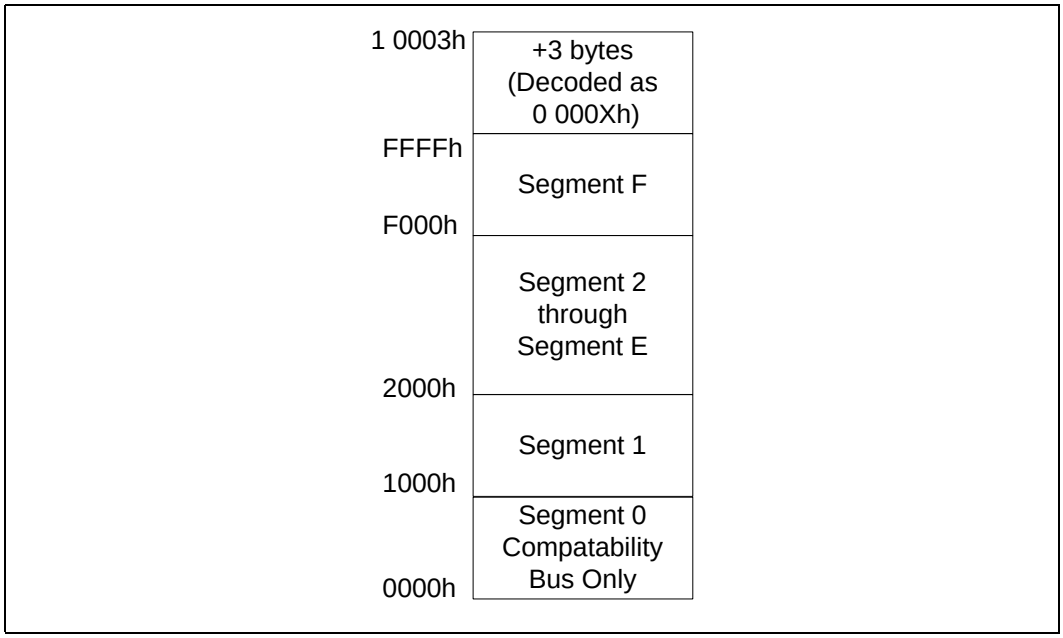
Historically, the 64 K I/O space actually was 64 K+3 bytes. For the extra 3 bytes, A#[16] is asserted. The Intel 5000X Chipset decodes only A#[15:3] when the request encoding indicates an I/O cycle. Therefore accesses with A#[16] asserted are decoded as if they were accesses to address 0 and are forwarded to the Compatibility Bus.

At power-on, all I/O accesses are mapped to the Compatibility Bus.

4.7.2 Outbound I/O Access

The Intel 5000X Chipset allows I/O addresses to be mapped to resources supported on the I/O buses underneath the MCH. This I/O space is partitioned into 16 4 KB segments. Each of the I/O buses can have from 1 to 15 segments mapped to it by programming its IOBASE and IOLIM registers. Each PCI bus must be assigned contiguous segments. The lowest segment, from 0 to 0FFFh, is sent to the ESI.

Figure 4-5. System I/O Address Space



4.7.3 Inbound I/O Access

Inbound I/Os are supported only for peer to peer accesses and are decoded the same as processor initiated I/Os. Inbound I/O accesses to the MCH (that is, CF8/CFC) are not supported and will receive a Master Abort response.

4.8 Configuration Space

All chipset registers are represented in the memory address map. In addition, some registers are also mapped as PCI registers in PCI configuration space.These adhere to the *PCI Local Bus Specification*, Revision 2.2.

§

5 Functional Description

This chapter describes each of the MCH interfaces and functional units including the Dual Independent Bus (DIB), processor Frontside Bus (FSB) interface, the PCI Express ports, system memory controller, power management, and clocking.

5.1 Processor Front Side Buses

The MCH supports two Dual-Core Intel® Xeon® 5000 Sequence processors on a 65 nanometer process in a 771-land, FC-LGA4 package. Dual-Core Intel Xeon 5000 Sequence is a fourth generation 32-bit Intel® Xeon® processor supporting Intel® Extended Memory 64 Technology (Intel® EM64T) based on Intel NetBurst® microarchitecture.

The MCH supports 1066 MHz FSB which is a quad-pumped bus running off a 266 MHz system clock, and a point to point DIB processor system bus interface. Each processor FSB supports peak address generation rates of 533 Million Addresses/second. Both FSB data buses are quad pumped 64-bits which allows peak bandwidths of 8.5 GB/s (1066 MT/s). The MCH supports 36-bit host addressing, decoding up to 64 GB of the processor's memory address space. Host-initiated I/O cycles are decoded to AGP/PCI, PCI Express, ESI interface or MCH configuration space. Host-initiated memory cycles are decoded to AGP/PCI, PCI Express, ESI or system memory.

5.1.1 FSB Overview

The MCH is the only priority agent for two point to point, independent, processor front side buses (FSB). These two buses are referred to as Dual Independent Buses (DIB). The MCH maintains coherency across these two buses. The MCH may complete deferrable transactions with either defer-replies or in-order responses. Intel 5000X Chipset contains an internal Snoop-Filter to remove unnecessary snoops on the remote FSB, and to be able to complete transactions in-order without deferring for transactions that do not need to have a remote snoop. Data transactions on the FSBs are optimized to support 64 byte cache lines.

Each processor FSB contains a 36 bit address bus, a 64 bit data bus, and associated control signals. The FSB utilizes a split-transaction, deferred reply protocol. The FSB uses source-synchronous transfer of address and data to improve performance. The FSB address bus is double pumped (2X) with ADS being sourced every other clock. The address bus generates a maximum bandwidth of 133 Million Addresses/second (MA/s).

The FSB data bus is quad pumped (4X) and supports peak bandwidths of 8.5 GB/s (1066 MT/s). Parity protection is applied to the data bus. This yields a combined bandwidth of 17 GB/s for both FSBs.

Interrupts are also delivered via the FSB.

5.1.2 FSB Dynamic Bus Inversion

The MCH supports Dynamic Bus Inversion (DBI) when driving and when receiving data from the processor. DBI limits the number of data signals that are driven to a low voltage on each quad pumped data phase. This decreases the worst-case power consumption of the MCH. The DBI[3:0]# signals indicate if the corresponding 16 bits of data are inverted on the bus for each quad pumped data phase.

Table 5-1. DBI[3:0]# / Data Bit Correspondence

DBI[3:0]#	Data Bits
DBI0#	D[15:0]#
DBI1#	D[31:16]#
DBI2#	D[47:32]#
DBI3#	D[63:48]#

When the processor or the MCH drives data, each 16-bit segment is analyzed. If more than 8 of the 16 signals would normally be driven low on the bus, the corresponding DBI# signal will be asserted and the data will be inverted prior to being driven on the bus. When the processor or the MCH receives data, it monitors DBI[3:0]# to determine if the corresponding data segment should be inverted.

5.1.3 FSB Interrupt Overview

The Dual-Core Intel Xeon 5000 Sequence processor supports FSB interrupt delivery. The legacy APIC serial bus interrupt delivery mechanism is not supported. Interrupt-related messages are encoded on the FSB as “Interrupt Message Transactions.” In the Intel 5000X Chipset platform, FSB interrupts may originate from the processor on the system bus, or from a downstream device on the Enterprise South Bridge Interface (ESI) or AGP. In the later case, the MCH drives the Interrupt Message Transaction onto the system bus.

In the Intel 5000X Chipset the Intel 631xESB/632xESB I/O Controller Hub contains IOxAPICs, and its interrupts are generated as upstream ESI memory writes. Furthermore, PCI 2.3 defines Message Signaled Interrupts (MSI) that are also in the form of memory writes. A PCI 2.3 device may generate an interrupt as an MSI cycle on its PCI bus instead of asserting a hardware signal to the IOxAPIC. The MSI may be directed to the IOxAPIC which in turn generates an interrupt as an upstream ESI memory write. Alternatively, the MSI may be directed directly to the FSB. The target of an MSI is dependent on the address of the interrupt memory write. The MCH forwards inbound ESI and AGP/PCI (PCI semantic only) memory writes to address 0FEEx_xxxxh to the FSB as Interrupt Message Transactions.

5.1.3.1 Upstream Interrupt Messages

The MCH accepts message-based interrupts from PCI (PCI semantics only) or ESI and forwards them to the FSB as Interrupt Message Transactions. The interrupt messages presented to the MCH are in the form of memory writes to address 0FEEx xxxxh. At the ESI or PCI interface, the memory write interrupt message is treated like any other memory write; it is either posted into the inbound data buffer (if space is available) or retried (if data buffer space is not immediately available). Once posted, the memory write from PCI or ESI to address 0FEEx xxxxh is decoded as a cycle that needs to be propagated by the MCH to the FSB as an Interrupt Message Transaction.

5.2 Snoop Filter

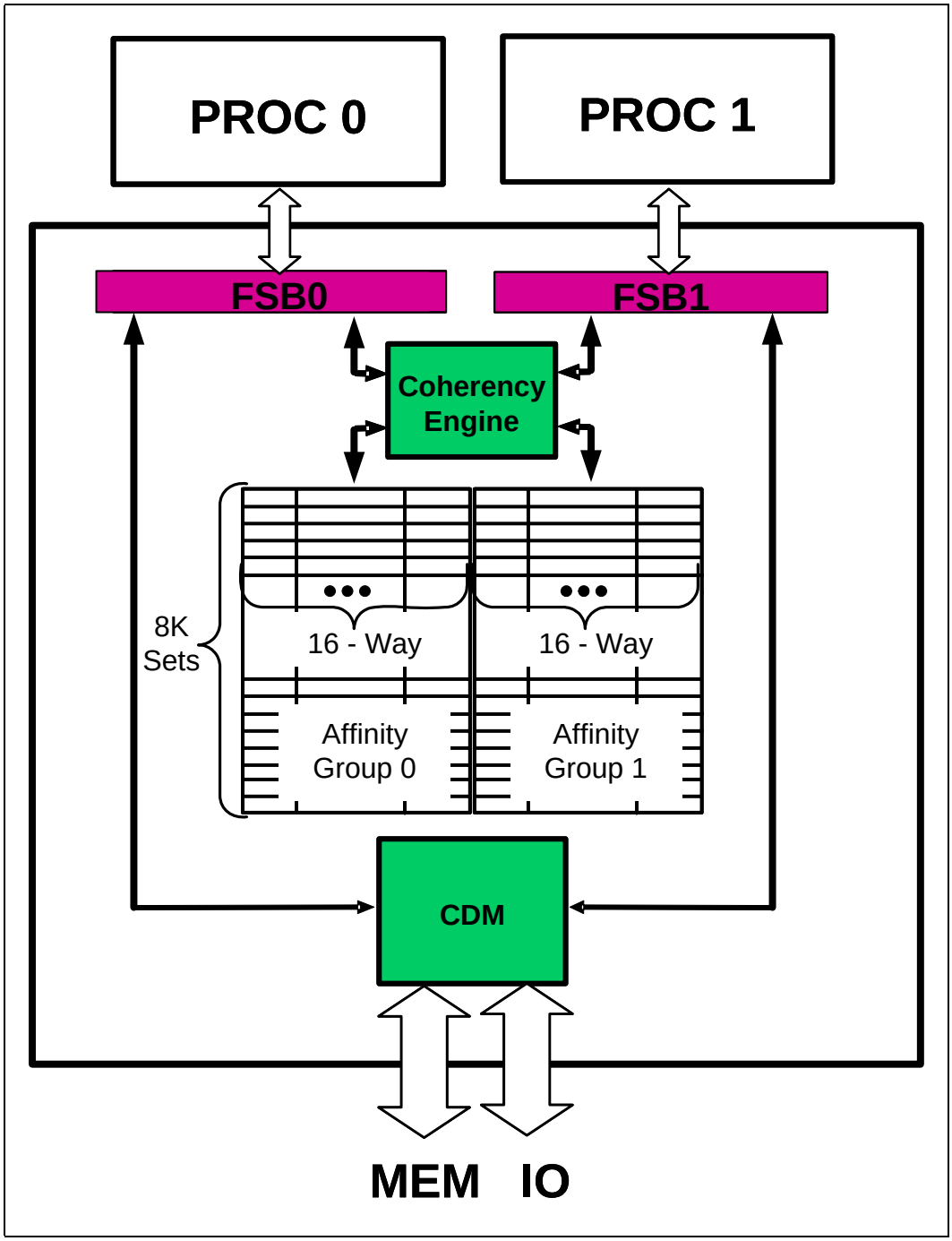
The Snoop Filter (SF) offers significant performance enhancements on several workstation benchmarks by eliminating traffic on the snooped frontside bus of the processor being snooped. By removing snoops from the snooped bus, the full bandwidth is available for other transactions. Supporting concurrent snoops effectively reduces performance degradation attributable to multiple snoop stalls. See Figure 5-1.

The SF is composed of two affinity groups each containing 8K sets of x16-way associative entries. The overall SF size is 16MB in size. Each affinity group supports a pseudo-LRU replacement algorithm. Lookups are done on a full 32-way per set for hit/miss checks.

As shown in Figure 5-1 the snoop filter is organized in two halves referred to as the Affinity Group 1 and Affinity Group 0 or the odd and even snoop filters respectively. As shown in Figure 5-1 Affinity Group 1 is associated with processor 1 and Affinity Group 0 is associated with processor 0. Under normal conditions a snoop is competed with a 1 snoop stall penalty. When the processors request simultaneous snoops the first snoop is completed with a one snoop stall penalty, the second snoop requires a 2 snoop stall penalty.

For the purposes of simultaneous SF access arbitration, processor 0 is given priority over processor 1. Thus simultaneous snoops are resolved with a 1 snoop stall penalty for processor 0 and a 2 snoop stall penalty for processor 2.

Figure 5-1. Snoop Filter



The SF stores the tags and coherency state information for all cache lines in the system. The SF is used to determine if a cache line associated with an address is cached in the system and where. The coherency protocol engine (CE) accesses the SF to look-up an entry, update/add an entry, or invalidate an entry in the snoop filter.

The SF has the following features:

- Snoop Filter tracks total of 16 MB of cachelines (2^{18} L2 lines).
- 8K sets organized as one interleave via a 2 x 16 Affinity Set-Associativity array. There are a total of 8 K x 2 x 16 = 256 K Lines (2^{18}).
- 2 x 16 Affinity Set-Associativity will allocate/evict entries within the 16-way corresponding to the assigned affinity group if the SF look up is a miss. Each SF look up will be based on 32-way (2x16 ways) look up.
- The array size of the snoop filter RAM is equivalent to 1 MB plus 0.03 MB of Pseudo-Least-Recently-Used (pLRU) RAM.

  Tag array size = 8192 sets * 4 bytes/set/group * (2 groups* 16 ways) = 1048576B =1 MB

  pLRU array size = 8192 sets * 15 bits/set/group * (2 groups) = 30720B = ~0.03 MB
- The Snoop Filter is operated at 2x of Intel 5000X Chipset core frequency, that is, 533 MHz to provide 267 MLUU/s (where a Look-Up-Update operation is a read followed by a write operation to the tag and pLRU arrays).
  - The maximum lookup and update bandwidth of the Snoop Filter is equal to the max request bandwidth from both FSB's. The lookup and update bandwidth from I/O coherent transactions have to share the bandwidth with both FSBs per request weighted-round-robin arbitration.
  - The SF lookup latency is four SF-clocks or two Intel 5000X Chipset core clocks to support single snoop stall in idle condition (single request issued from either bus). If both bus are making requests simultaneously, the snoop-filter will always select bus 0 first. In such scenario, bus 0 request will have one snoop-stall and bus 1 request will have two snoop-stalls.
- Pseudo-Least-Recently-Used (pLRU) replacement algorithm, with updates on lookups, and invalidates.
- Tag entries supporting a 40-bit internal physical address space. The MCH external address space is 36 bits.
- Stores coherency state (EM) and Bus[1:0] for each valid cache line in the system. The tracking algorithm utilizes conservative tracking (super-set tracking). The processor can silently down grade a line state from E to S/I or S to I without any action appearing on the FSB. Therefore, a line appearing in the SF as E states may actually missed in the corresponding processor caches. Conversely a SF S-line will never be found in E/M state in a processors L2 cache, or a SF miss will never be found in M/E/S state in a processors L2 cache. The following is the summery of the snoop-filter state definitions:
  - Coherency state: the cache line is in E/M state if the bit is set; else, the line is in share state
  - If Bus[1:0]=00, the entry is invalid.
  - If Bus[1:0]=01, the FSB0 processor(s) has ownership of the line.
  - If Bus[1:0]=10, the FSB1 processor(s) has ownership of the line.
  - If Bus[1:0]=11, both buses have ownership and the line must be shared by both FSB processors (EM must be 0).
  - EM||Bus[1:0] =111 is a reserved definition.
- ECC coverage, with correction of single bit errors, detection of double bit errors (SEC-DED).
  - pLRU array does not need ECC protection. Bit failure will result in selecting different entry than the pLRU selection and may affect the performance. There are no correctness issue.

- Snoop-Filter Fast array initialization and/or self test through configuration register access.

5.2.1 Snoop Filter Address Bit Mapping

The SF supports a 40-bit physical address. Table 5-1 shows the partitioning of the address for indexing into the SF array.

Table 5-1. Snoop Filter Physical Address Partitioning

Tag (21b)	Set (13b)	Byte Offset (6b)
A[39:19]	A[18:6]	A[5:0]

5.2.2 Operations and Interfaces

The following table shows the snoop-filter look up qualifier for coherent transaction issued from processors, that is, ADS# assertion driven from processors.

Table 5-2. FSB Transaction Encoding Qualification for SF Look Up

SF look up transactions from FSB: The following REQa[2:0] encoding with ADS# assertion from processor qualification.			
Request Names	REQa[2:0]		
	REQa[2]	REQa[1]	REQa[0]
BRIL/BIL	0	1	0
BRLC	1	0	0
BRLD/BLR	1	1	0
BWL	1	0	1
BWIL/BLW	1	1	1

Table 5-1 shows the organization of each snoop filter entry, and interpretation of the contents.

Table 5-3. Snoop Filter Entry (Sheet 1 of 2)

Bits	Value
[31]	**Redundant bit.**
[30:24]	**ECC check bits**

Table 5-3. Snoop Filter Entry (Sheet 2 of 2)

Bits	Value
[23]	**State of the cache line** 1 The cache line is in E/M state, that is, the line is either exclusive (but clean) or modified (dirty) state. 0 The cache line is in non-E/M state, that is, S state if Bus presence vector is non-zero or I state if Bus presence vector is zero.
[22:21]	**Bus presence vector** [00] The entry is invalid [xx] The entry is present in any of the processor L2 in the corresponding FSB. Bus0 is the least significant bit. Bus0 corresponds to FSB0 on the Intel® 5000P chipset/Intel 5000X Chipset MCH. Bus1 corresponding to FSB1
[20:0]	**Tag portion of the address**

The snoop filter supports the following key operations during normal operation. Due to timing constraints, these lookup and update commands have been removed for the SF configuration access.

- SF_Lookup with pLRU status update by MRU operation

  On a lookup, the SF uses the tag and set portion of the input address to determine if the entry is in the SF. The SF asserts a hit if there is a match and provides the contents, and way information for the matched entry. If the lookup is a miss, the SF provides the contents of the victim entry, set and way information of the victim. The pLRU vector is updated according to a Most-Recently-Used (MRU) algorithm. The SF indicates if a single or double bit error was detected. Single-bit errors are corrected (but the array is not updated). Hit/miss calculation is performed after the ECC logic. All FSB request to memory will use this command for SF lookup.

- SF_Lookup with no pLRU status update operation

  The SF uses the tag and set portion of the input address to determine if the entry is in the SF. The SF asserts a hit if there is a match and provides the contents, and way information for the matched entry. If the lookup is a miss, the SF provides the contents of the victim entry, set and way information of the victim. The pLRU vector is not updated. The SF indicates if a single or double bit error was detected. Single-bit errors are corrected (but the array is not updated). Hit/miss calculation is performed after the ECC logic. Inbound memory accesses will use this command for SF lookup.

- SF_Update with pLRU status update by LRU operation

  Set and way are provided for write operations. Writes can either be updates or invalidates. On SF-invalidations, the pLRU array is updated using a Least-Recently-Used (LRU) entry tracking algorithm. The SF update during inbound write will also use the “SF_Update with pLRU status update by LRU op” if there is a hit during SF lookup.

- SF_Update with no pLRU status update operation

  Set and way are provided for write operations. The pLRU array is not updated. This command is used during non-SF-entry-invalidation operations.

5.3 System Memory Controller

The MCH masters four Fully-Buffered DIMM (FB-DIMM) memory channels. Up to four DIMMs can be connected to each FB-DIMM channel (up to sixteen DIMMs for the entire array). FB-DIMM memory utilizes a narrow high speed frame oriented interface referred to as a channel.

The four FB-DIMM channels are organized into two branches of two channels per branch. Each branch is supported by a separate Memory Controller (MC). The two channels on each branch operate in lock step to increase FB-DIMM bandwidth. A branch transfers 16 bytes of payload/frame on Southbound lanes and 32 bytes of payload/ frame on Northbound lanes.

The two branches may be operated in mirrored (RAID 1) or non-mirrored mode. When operating in mirrored mode, 64 GB of memory will produce an effective 32 GB memory space.

The key features of the FB-DIMM memory interface are summarized in the following list.

- Four Fully Buffered DDR (FB-DIMM) memory channels.
- Branch channels are paired together in lock step to match FSB bandwidth requirement.
- Each FB-DIMM Channel can link up to four Fully Buffered - DDR DIMMs (FB-DIMM).
- Supports up to 16 dual-ranked FB-DDR2 4GB DIMMs, that is, 64GB[8]of physical memory in non-mirrored configuration or 32GB of physical memory in mirrored configuration.
- The FB-DIMM link speed is at 6x the DDR data transfer speed. A 3.2 GHz FB-DIMM link supports DDR2-533 (FSB@1067 MT/s).
- The MCH will comply with the FB-DIMM specification definition of a host and will be compatible with any FB-DIMM-compliant DIMM.
- Special single channel, single DIMM operation mode (Branch 0, Channel 0, Slot 0 position only).
- All memory devices must be DDR2.

Table 5-2 and Figure 5-4 present system memory capacity as a function of DRAM device capacity and MCH operating mode.

Table 5-2. Minimum System Memory Configurations and Upgrade Increments

DRAM Technology	Smallest System Configuration - One DIMM	Smallest Upgrade Increment - Two DIMM
256 Mbit	256 MB	512 MB
512 Mbit	512 MB	1024 MB
1024 Mbit	1024 MB	2048 MB
2048 Mbit	2048 MB	4096 MB

The *Smallest System Configuration - One DIMM* column represents the smallest possible single DIMM capacity for a given technology (MCH operating in single channel, single DIMM mode with x8 single rank (x8SR) DIMM populated). The *Smallest Upgrade Increment - Two DIMMs* column represents the smallest possible memory upgrade capacity for a given technology using two x8 single rank DIMMs.

8. User can only access up to 63.5 GB of memory due to minimum 256 MB MMIO/TOLM and limited address decoding above 64 GB.

Table 5-3. Maximum 16 DIMM System Memory Configurations

DRAM Technology x8 Single Rank	Maximum Capacity Mirrored Mode	Maximum Capacity Non-Mirrored Mode
256 Mbit	2 GB	4 GB
512 Mbit	4 GB	8 GB
1024 Mbit	8 GB	16 GB
2048 Mbit	16 GB	32 GB

Note: The *Maximum Capacity Mirrored Mode* and *Maximum Capacity Non-Mirrored Mode* columns represent the system memory available when all DIMM slots are populated with identical x8 Single Rank (x8DR) DIMMs using the DRAM Technology indicated.

Table 5-4. Maximum 16 DIMM System Memory Configurations

DRAM Technology x4 Dual Rank	Maximum Capacity Mirrored Mode	Maximum Capacity Non-Mirrored Mode
256 Mbit	8 GB	16 GB
512 Mbit	16 GB	32 GB
1024 Mbit	32 GB	64 GB
2048 Mbit	32 GB	64 GB

Note: The *Maximum Capacity Mirrored Mode* and *Maximum Capacity Non-Mirrored Mode* columns represent the system memory available when all DIMM slots are populated with identical x4 Double Rank (x4DR) DIMMs using the DRAM Technology indicated.

5.3.1 Memory Population Rules

DIMM population rules depend on the operating mode of the MC. When operating in non-mirrored mode the minimum memory upgrade increment is two identical DIMMs per branch (DIMMs must be identical with respect to size, speed, and organization). Non-mirrored mode has an exceptional mode that operates with a single DIMM which is discussed in the following section. When operating in mirrored mode the minimum upgrade increment is four identical DIMMs

5.3.1.1 Non-Mirrored Mode Memory Upgrades

The minimum memory upgrade increment is two DIMMs per branch. The DIMMs must cover the same slot position on both channels. DIMMs that cover a slot position must be identical with respect to size, speed, and organization. DIMMs that cover adjacent slot positions need not be identical.

Within a branch memory DIMMs must be populated in slot order; slot 0 is populated first, slot 1 second, slot 2 third, and slot 3 last. Slot 0 is closest to the MCH.

Section 5-2 depicts the minimum two DIMM configuration. The populated DIMMs are depicted in gray (Slot 0 of Branch 0 populated).

Figure 5-2. Minimum Two DIMM Configuration

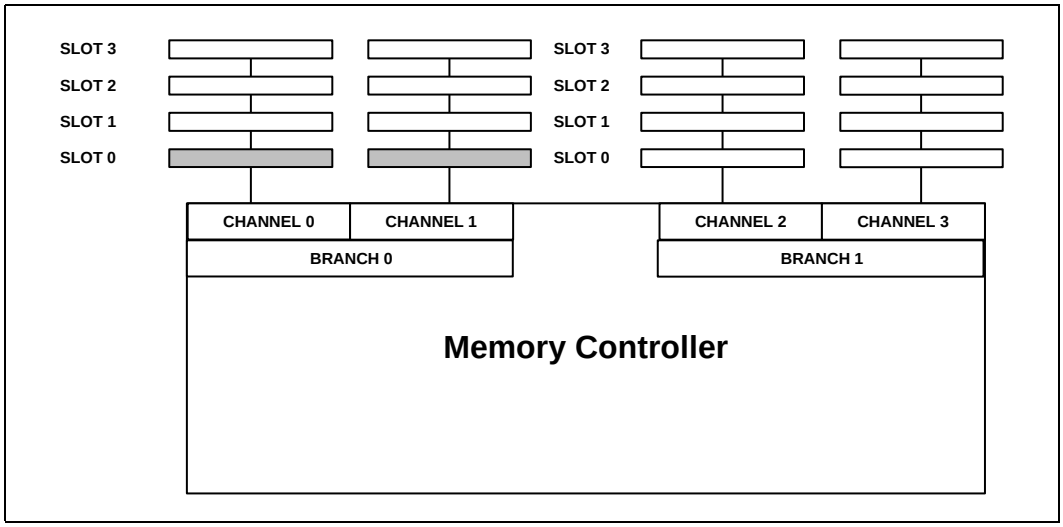


Figure 5-3 depicts the next two positions where DIMMs may be added. These positions are depicted in dark gray. The two upgrade positions are Branch 0, Slot 1 and Branch 1, Slot 0. Of these Branch 1, Slot 0 is the preferred upgrade because it allows both branches to operate independently and simultaneously. FB-DIMM memory bandwidth is doubled when both branches operate in parallel.

While it is possible to completely populate one branch before populating the second branch, it is not desirable to do so from a performance standpoint. In general memory upgrades should be balanced with respect to both branches to optimize FB-DIMM performance.

Figure 5-3. Next Two DIMM Upgrade Positions

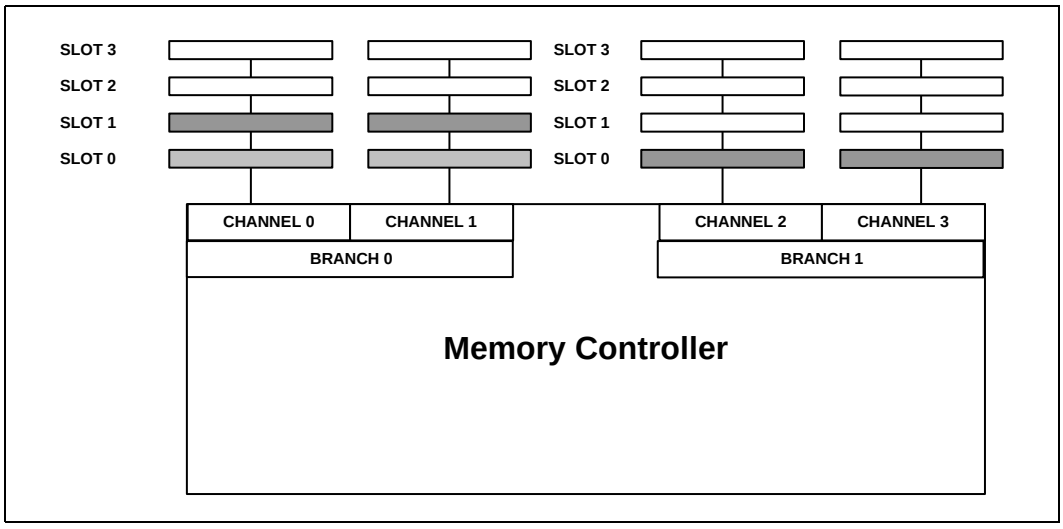
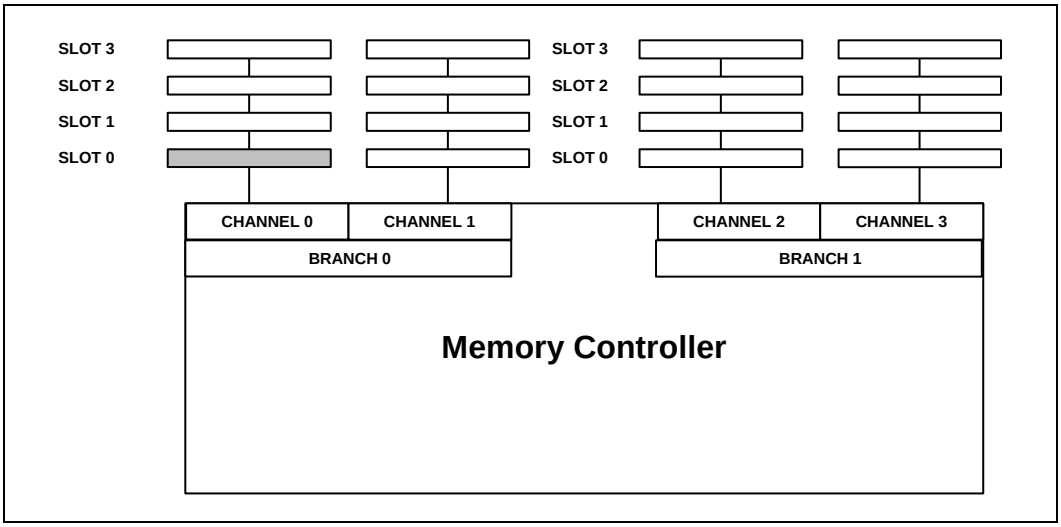


Figure 5-4 depicts a special single DIMM non-mirrored operation mode. This mode requires that the DIMM be placed in Branch 0, Channel 0, Slot 0. When upgrading from this mode the normal two DIMM memory upgrade rules are followed.

Figure 5-4. Single DIMM Operation Mode



5.3.1.2 Mirrored Mode Memory Upgrades

When operating in mirrored mode both branches operate in lock step. In mirrored mode Branch 1 contains a replicate copy of the data in Branch 0. For this reason the minimum memory upgrade increment, for mirrored mode, is four DIMMs across all branches. The DIMMs must cover the same slot position on both branches. DIMMs that cover a slot position must be identical with respect to size, speed, and organization. DIMMs within a slot position must match each other, but aren't required to match adjacent slot positions.

Figure 5-5 shows the minimum memory configuration required to operate in mirrored mode.

Figure 5-5. Minimum Mirrored Mode Memory Configuration

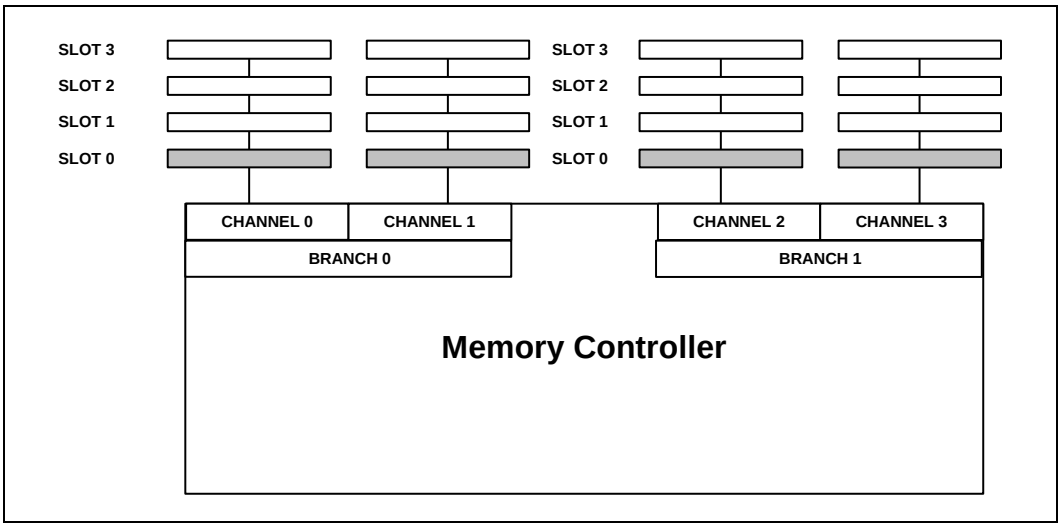
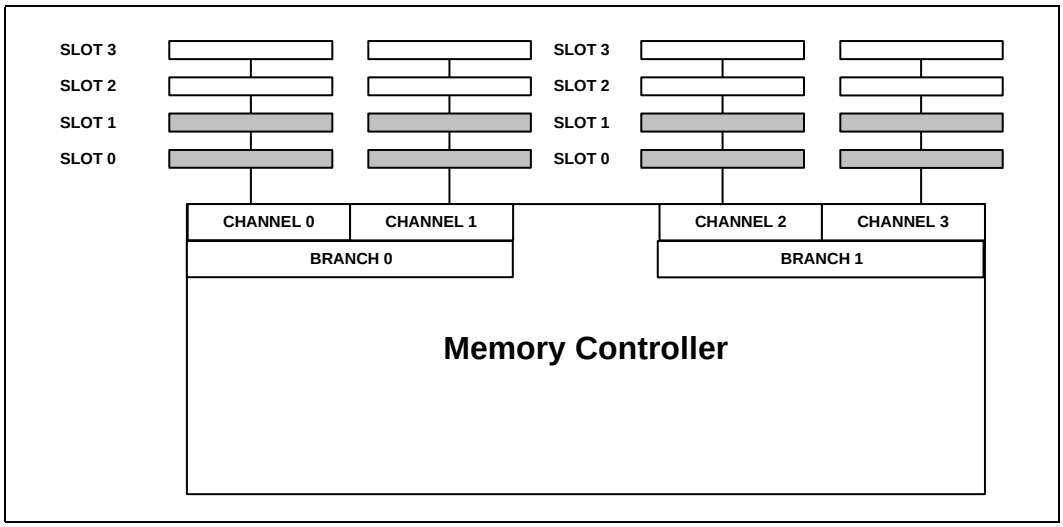


Figure 5-6 shows the positions of the next four DIMM upgrade. Like non-mirrored mode upgrade DIMMs must be added in slot order, starting from the slot closest to the MCH. DIMMs in a slot position must be identical with respect to size, and organization. Speed should be matched but is not required. The MCH will adjust to the lowest speed DIMM. DIMMs in adjacent slots need not be identical.

Figure 5-6. Mirrored Mode Next Upgrade



5.3.2 Fully Buffered DIMM Technology and Organization

Fully Buffered DIMM technology was developed to address the higher performance needs of server and workstation platforms. FB-DIMM addresses the dual needs for higher bandwidth and larger memory sizes.

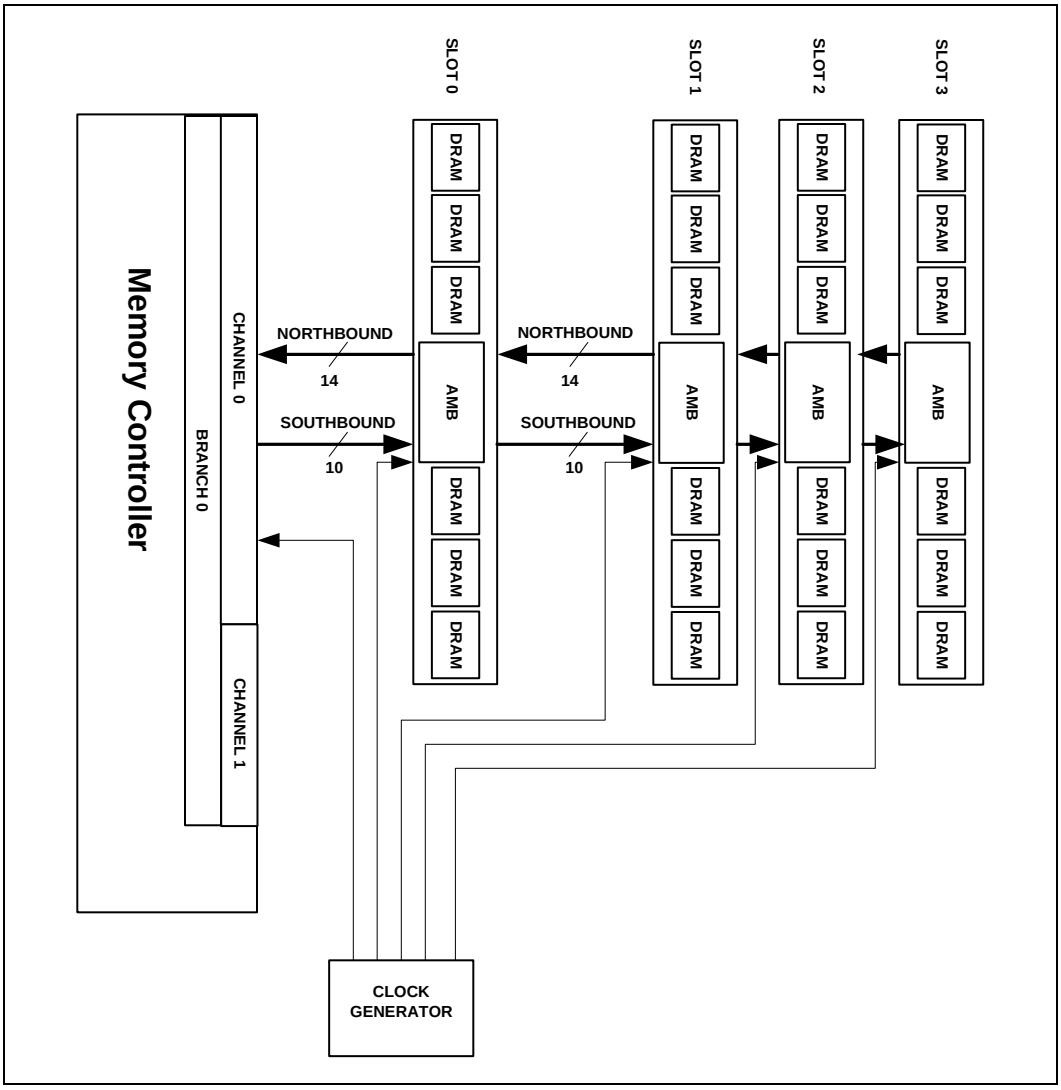
FB-DIMM memory DIMMs contain an Advanced Memory Buffer (AMB) device that serves as an interface between the point to point FB-DIMM Channel links and the DDR2 DRAM devices. Each AMB is capable of buffering up to two ranks of DRAM devices. Each AMB supports two complete FB-DIMM channel interfaces. The first FB-DIMM interface is the incoming interface between the AMB and its proceeding device. The second interface is the outgoing interface and is between the AMB and its succeeding device. The point to point FB-DIMM links are terminated by the last AMB in a chain. The outgoing interface of the last AMB requires no external termination.

There are three major components of the FB-DIMM channel interface:

- 14 Differential Northbound Signal pairs
- 10 Differential Southbound Signal pairs
- 1 Differential Clock Signal pair

Figure 5-7 depicts a single FB-DIMM channel with these three signal groups.

Figure 5-7. FB-DIMM Channel Schematic



A FB-DIMM channel consists of 14 unidirectional differential signal pairs referred to as the Northbound path, 10 unidirectional differential signal pairs referred to as the Southbound path, and a differential reference clock.

NOTE: The northbound signal pairs are enumerated from 0 to 13. Signal pair 13 is not active but must be connected to properly terminate the FB-DIMM channel. See sections, Figure 2.2.1 and Figure 2.2.2.

The southbound path is used to convey DIMM commands and write data to the addressed DIMMs. The northbound path returns read data and status from the addressed DIMM.

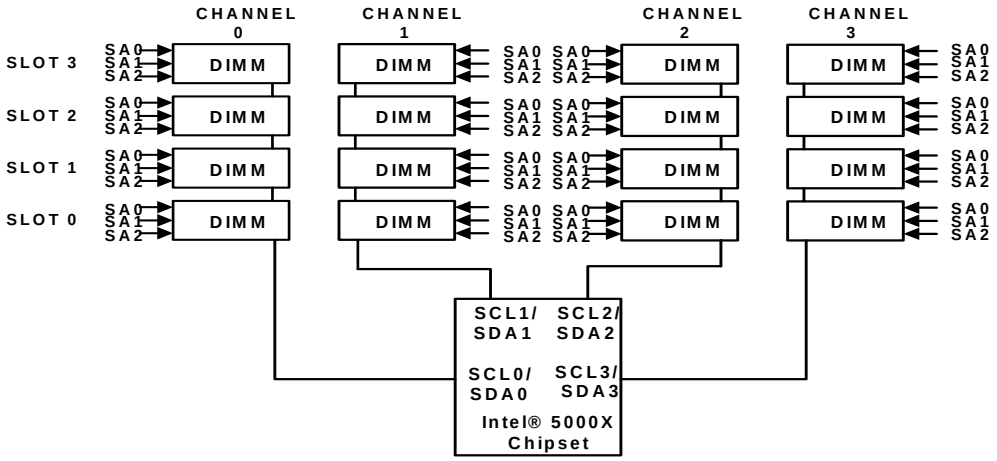
The northbound and southbound paths are used to convey FB-DIMM frames that are synchronized to the reference clock. Each frame consists of 12 data transfers. Southbound frames contain a payload of 8 bytes per frame per channel. Northbound frames contain a payload of 16 bytes per frame per channel.

5.3.3 FB-DIMM Memory Operating Modes

The MCH supports two major modes of operation, mirrored and non-mirrored.

5.3.3.1 Non-Mirrored Mode Operation

When operating in non-mirrored mode the MCH operates the two branches independently. In non-mirrored mode the full MCH address space of 64GB is available.



Normally when operating in non-mirrored mode both channels on a branch are operated in lock step, referred to as dual-channel mode. There is a single DIMM, single channel mode of operation referred to as single-channel mode.

5.3.3.1.1 Non-Mirrored Mode ECC

ECC is supported differently for each of these single- and dual-channel modes:

Dual-Channel Mode:

When branches operate in dual-channel mode, the MCH supports the 18 device DRAM failure correction code option for FB-DIMM. As applied by Intel 5000X Chipset, this code has the following properties:

- Correction of any x4 or x8 DRAM device failure
- Detection of 99.986% of all single bit failures that occur in addition to a x8 DRAM failure. The MCH will detect a series of failures on a specific DRAM and use this information in addition to the information provided by the code to achieve 100% detection of these cases.
- Detection of all two wire faults on the DIMMs. This includes any pair of single bit errors.
- Detection of all permutations of two x4 DRAM failures.

Single-Channel Mode:

When the branch operates in single-channel/single-DIMM mode, the MCH employs x8 SDDC as in the dual channel case. However, in this case, the ECC RAS feature set is limited for the single DIMM memory subsystem. In the single DIMM mode (for example, nine x8 devices), the SDDC cannot correct single wire fault (stuck-at) errors or permanent full device errors. This is because the error correction capability in the SDDC is limited to adjacent symbol errors on a 16-bit boundary and in the single DIMM mode with a Burst Length of 8, there are 4 transfers of 8 B to form a 32 B codeword.

Hence a single wire failure in the same device is replicated across all 4 symbols hampering the error correction. The SDDC can detect most x4/x8 DRAM failures but it can only correct adjacent symbol errors that occur within a 16-bit boundary of each codeword.

5.3.3.2 Mirrored Mode Operation

Memory Mirroring MCH is a user-selectable feature. Mirrored mode provides for complete recovery from a DIMM device failure. The mirroring feature is fundamentally a way for hardware to maintain two copies of all data in the memory subsystem, such that a hardware failure or uncorrectable error is no longer fatal to the system. When an uncorrectable error is encountered during normal operation, hardware simply retrieves the “mirror” copy of the corrupted data, and no system failure will occur unless both primary and mirror copies of the same data are corrupt simultaneously (statistically very unlikely).

When operating in mirrored mode FB-DIMM Branch 0 (Channels 0/1) and Branch 1 (Channels 2/3) contain replicate copies of data (are mirrored images). Since Branch 1 contains a replicate copy of Branch 0’s data, the maximum addressable memory is reduced to 32 GB.

Mirrored mode must be selected at configuration time by enabling mirrored operation.

The general flows for mirroring are as follows:

- The same Write (in the non-failed case) is issued to both branches in the same cycle (which is complete when both branches acknowledge).
- Different Reads (in the non-failed case) are issued to each branch in the same cycle. Read returns from Branch 1 are delayed two cycles from read returns from Branch 0.
- Corrected data will be forwarded to the requester.
- Uncorrectable errors will be retried from the other image. If the other image is off-line, uncorrectable errors will be retried from the same image.
- Software can temporarily degrade operation to one memory branch and then resume operation with both memory branches.
- To recover from a failed DIMM:
  - DIMM failure is detected
  - The defective branch is shut down
  - The system is gracefully shut down by operator and the defective DIMM is replaced
  - The system is repowered up
  - Normal processing is restored

5.3.3.3 Mirrored Mode ECC

When operating in non-mirrored mode the MCH operates the two branches in lock step (one branch mirroring the contents of the other). In mirrored mode the maximum address space is reduced to 32GB because the two channels are mirroring each other. When operating in mirrored mode both channels on a branch are operated in lock step, referred to as dual-channel mode.

ECC is calculated using the dual-channel method defined in Section 5.3.3.1.1.

5.3.3.4 Memory Sparing

At configuration time, a DIMM rank is set aside to replace a defective DIMM rank. When the error rate for a failing DIMM rank reaches a pre-determined threshold, the SPCPS.LBTHRconfiguration bit will issue an interrupt and initiate a spare copy. While the copy engine is automatically reading locations from the failing DIMM rank and writing them to the spare (see Section 3.9.23.4 and Section 3.9.23.5, “Spare Copy Status & Spare Copy Control”, system reads will be serviced from the failing DIMM rank, and system writes will be written to both the failing DIMM rank and the spare DIMM rank. At the completion of the copy, the failing DIMM rank is disabled and the “spared” DIMM rank will be used in its place. The MCH will change the rank numbers in the DMIRs from the failing rank to the spare rank. DMIR.LIMIT’s are not updated.

This mechanism requires no software support once it has been enabled by designating the spare rank through the SPCPC.SPRANK configuration register field and enabling sparing by setting the SPCPC.SPAREN configuration bit. Hardware will detect the threshold-initiated fail, accomplish the copy, and off-line the “failed” DIMM rank once the copy has completed. This is accomplished autonomously by the memory control subsystem. The SPCPS.SFO configuration bit is set and an interrupt is issued indicating that a sparing event has completed.

Sparing cannot be invoked while operating a mirrored memory configuration. Sparing to a smaller DIMM is not supported.

Note: DIMM sparing is not validated in the Single Channel Mode when Intel 5000X Chipset.MCA.SCHDIMM is set.

5.3.4 Data Poisoning in Memory

Data Poisoning in memory is defined as all zeroes in the code word (32B0 except for the least significant bytes being 0xFF00FF. The Intel 5000X Chipset MCH poisons a memory location based on the events described in Table 5-5.

Table 5-5. Memory Poisoning Table

Event	Correctable Error	UnCorrectable Error
Normal Memory Read	Correct Data to be given register Intel 5000X Chipset MCH logs M17 error. (Correctable Non-Mirrored demand data ECC Error) Correct Data to be written back to memory	Detects an Uncorrectable and logs a M9 error (Non-aliased uncorrectable non-mirrored demand data ECC error) Re-Issue Read to Memory If error persistent 1. Poison the response to requester and log. 2. Leave data untouched in memory location
Patrol Scrub	Correct Data to be written back to memory and log M20 error. (Correctable patrolled data ECC error)	1. Log and Signal M12 Error (Non-Aliased uncorrectable patrol data ECC error). 2: Leave data untouched in memory location.
DIMM Spare Copy	Correct Data to be written back to memory and log M19 error. (Correctable re-silver or spare copy data ECC error)	If error persistent 1. Log and Signal M11 Error (Non Aliased uncorrectable re-silver or spare copy data ECC error). 2: Poison Location in DIMM Spare
Mirror Copy	Correct Data to be written to new memory and log M19 error. (Correctable re-silver or spare copy data ECC error)	Re-use Read to memory and signal a M11 error. (Non-aliased uncorrectable re-silver or spare copy data ECC error). If error persistent 1. Poison the new memory image.

5.3.5 Patrol Scrubbing

To enable this function, the MC.SCRBEN configuration bit must be set. The scrub unit starts at DIMM Rank 0 / Address 0 upon reset. Every 16 k core cycles the unit will scrub one cache line and then increment the address one cache line provided that back pressure or other internal dependencies (queueing, conflicts, and so forth) do not prolong the issuing of these transactions to FB-DIMM. Using this method, roughly 6 GBytes of memory behind the Intel 5000X Chipset MCH can be completely scrubbed every day (estimate). Error logs include RAS/CAS/BANK/RANK. Patrol scrub writes hit both branches in mirrored mode (when MC.MIRROR is set). Normally, one branch is scrubbed in entirety before proceeding to the other branch. In the instance of a fail-down to non-redundant operation that off-lines the branch that was being scrubbed, the scrub pointer merely migrates to the other branch without being cleared. In this unique instance, the scrub cycles for that branch is incomplete.

5.3.6 Demand Scrubbing

To enable this function, the MC.DEMSEN configuration bit must be set. Correctable read data will be corrected to the requestor and scrubbed in memory. This adds an extra cycle of latency to accomplish the correction. Error logs include RAS/CAS/BANK/RANK.

Demand scrubbing is not available in mirrored mode (when MC.MIRROR is set) to simplify the design. If a correctable error is encountered, the data is corrected and sent to the requestor at the cost of one extra cycle of latency. The probability of soft errors due to alpha rays affecting multiple x4/x8 devices is low. However, patrol scrubbing when enabled in the Intel 5000X Chipset MCH for the Mirrored mode will clean up all correctable errors in the memory running in the background and runs twice as fast. There is no incurred RAS benefit by enforcing demand scrubbing in mirrored mode with the exception of the error logging. Demand scrubbing does not help in failed ECC case (uncorrectable errors). that is, If the data read is uncorrectable from the bad branch, then the golden data needs to be retrieved from the other mirrored branch (copy) at the cost of additional FB-DIMM reset, link training and DDR protocol rules. The failed branch is offlined and needs to be replaced for mirroring to continue.

5.3.7 x8 Correction

5.3.7.1 Normal

This correction mode is in effect when the MC.SCRBALGO configuration bit is cleared. An erroneous read will be logged. If the ECC was correctable, it is corrected (scrubbed) in memory. A conflicting read or write request pending issue will be held until the scrub is either completed or aborted because it was uncorrectable.

5.3.7.2 Enhanced

This correction mode is in effect when the MC.SCRBALGO configuration bit is set and software has initialized the MC.BADRAMTH to a non-zero value.

- Maintain 4-bit saturating counters per rank in the BADCNT configuration registers.

Floor at zero. Saturate at the value of the MC.BADRAMTH configuration register field. Increment on correctable errors on both symbols of a x8 device and Northbound CRC OK. Decrement upon completion of the number of patrol scrub cycles through the entire memory specified by the MC.BADRAMTH configuration register field. A sufficient resolution of this period is three patrol scrub cycles through all memory.

- Maintain five-bit bad-device marks per rank in the BADRAM(A/B) configuration registers.

Upon incrementing BADCNT to saturation, then mark the bad devices in the BADRAM(A/B) configuration registers.

- A correctable ECC in a symbol other than that marked in the BADRAM(A/B) configuration registers is an aliased uncorrectable read.

An erroneous read will be logged. If the read was correctable, it is corrected (scrubbed) in memory. A conflicting read or write request remains pending until the scrub succeeds or is dropped. A failed scrub is replayed once, resulting in success or a drop.

5.3.8 Single Device Data Correction (SDDC) Support

The Intel 5000X Chipset employs a single device data correction (SDDC) algorithm for the memory subsystem that will recover from a x4/x8 component failure. The chip disable is a 32-byte two-phase code. SDDC is also supported for x4 devices. In addition the MCH supports demand and patrol scrubbing.

A scrub corrects a correctable error in memory. A four-byte ECC is attached to each 32-byte “payload”. An error is detected when the ECC calculated from the payload mismatches the ECC read from memory. The error is corrected by modifying either the ECC or the payload or both and writing both the ECC and payload back to memory.

Only one demand or patrol scrub can be in process at a time.

The attributes of the SDDC code are as follows:

- Two Phase Code over 32 bytes of data.
- 100% Correction for all single x4 or x8 component failures.
- 100% Detection of all double x4 component failures.
- Detection Characteristics for x8 double device errors are provided in the Table 5-6.

Table 5-6. x8 Double Device Detection Characteristics

Overall coverage - 99.986%	Device plus single - 99.99999%
Double bit errors - 100%	Device plus wire - 99.99998%
Double wire faults - 100%	Device plus equal/phase - 99.9998%
Wire plus single bit - 100%	Equal/phase plus equal/phase - 100%

To increase the detection coverage of a (x8 device failure + SBE), that is, to avoid silent data corruption in the event of a particle induced error while correcting for a failed device, the Intel 5000X Chipset MCH provides the following features:

- Each rank will have an encoded value of the “failed” x8 component or pair of x4 components.
- If for any given rank, the Intel 5000X Chipset MCH detects a correctable error with a weight >1 and the “corrected” symbol does not match the “failed” component then the Intel 5000X Chipset MCH will assume that the error is multi-bit uncorrectable error and signal a “fatal error”.

5.3.9 FB-DIMM Memory Configuration Mechanism

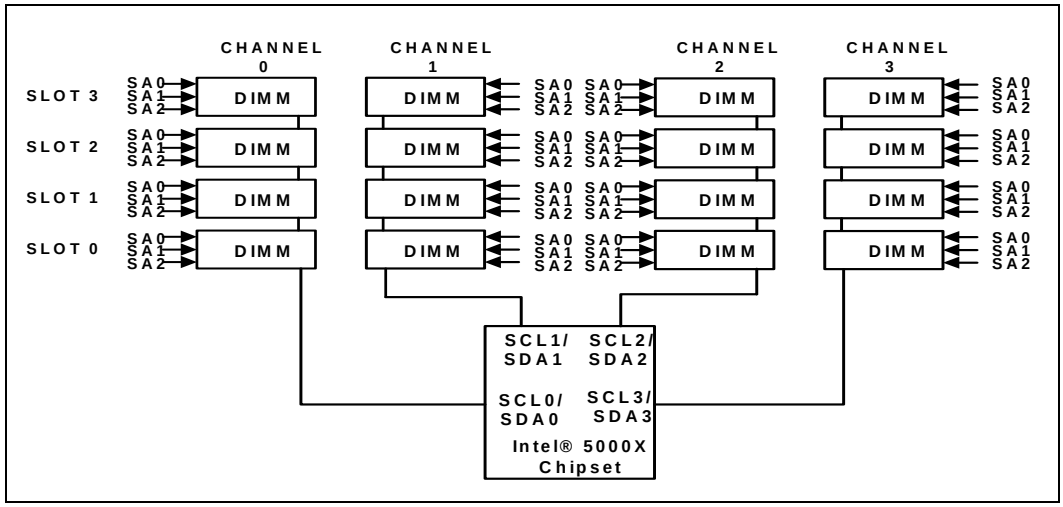
Before any cycles to the memory interface can be supported, the MCH DRAM registers must be initialized. The MCH must be configured for operation with the installed memory types. Detection of memory type and size is accomplished via the 4 Serial

Presents Detect (System Management Bus) interfaces on the MCH (SMBus 1, 2, 3 and 4). The SMBus interfaces are two-wire buses are used to extract the DRAM type and size information from the Serial Presence Detect port on the DIMMs.

FB-DIMMs contain a 6-pin Serial Presence Detect interface, which includes SCL (serial clock), SDA (serial data), and SA[3:0] (serial address). Devices on the SMBus bus have a 7-bit address. For the DIMMs, the upper three bits are fixed at 101. The lower four bits are strapped via the SA[3:0] pins. SCL and SDA are connected to the respective SPDxSMBDATA, SPDxSMBCLK pins on the MCH, see Figure 5-8.

The Intel 5000X Chipset MCH integrates a 100KHz SPD controller to access the DIMM SPD EEPROM's. There are four SPD ports. SPD0SMBDATA, and SPD0SMBCLK are defined for channel 0; SPD1SMBDATA, and SPD1SMBCLK are defined for channel 1; SPD2SMBDATA, and SPD2SMBCLK are defined for channel 2; and SPD3SMBDATA, and SPD3SMBCLK are defined for channel 3. There can be a maximum of eight SPD EEPROM's associated with each SPD bus. Therefore, the SPD interface is wired as indicated in Figure 5-8.

Figure 5-8. Connection of DIMM Serial I/O Signals



Board layout must map chip selects to SPD Slave Addresses as shown in Table 5-7. The slave address is written to the **SPDCMD** configuration register (see Section 3.9.28.2).

Table 5-7. SPD Addressing (Sheet 1 of 2)

SPD Bus	FB-DIMM Channel	SLOT	Slave Address
0	0	0	0
		1	1
		2	2
		3	3
1	1	0	0
		1	1
		2	2
		3	3

Table 5-7. SPD Addressing (Sheet 2 of 2)

SPD Bus	FB-DIMM Channel	SLOT	Slave Address
2	2	0	0
		1	1
		2	2
		3	3
3	3	0	0
		1	1
		2	2
		3	3

5.3.10 FB-DIMM Memory Failure Isolation Mechanisms

Since the Intel 5000X Chipset MCH does not operate FB-DIMM in fail-over mode, CRC accompanies Northbound data. Successful transaction completion is signalled by the absence of alerts within a read round-trip. Bad CRC accompanies alerts. Alerts preempt read data. Detection of corrupted CRC or corrupted write acknowledge (idle) will initiate an FB-DIMM fast reset followed by a retry of all commands since completion of the last successful transaction. A consecutive CRC/ack failure on the same transaction is fatal.

5.3.10.1 FB-DIMM Configuration Read Error

An erroneous configuration read return will be master aborted and return all 1's. It will not be retried.

5.3.10.2 DIMM Failure Isolation

The failing DIMM may be isolated using information contained in several registers. ECC error flag bits are recorded in register FERR_NF_FBD, Section 3.9.23.3. This register records various error sources related to FB-DIMM memory transactions. When an error occurs the channel/branch information is recorded in the FBDChan_indx field.

The FBDChan_indx is a two bit field that records branch ECC errors. ECC errors are reported on a per branch basis (the LSB of this field has no relevance for ECC errors). For ECC errors the possible values for this field are:

FBDChan_indx = 0 Branch 0 ECC error

FBDChan_indx = 2 Branch 1 ECC error

Once the branch is determined the failing DIMM is determined, the rank and DIMM is determined from the RECMEMA.RANK and REDMEMB.ECC_Locator fields. The ECC_Locator indicates which x8 SDRAM device (or pair of adjacent x4 devices) caused the error. If any of the bits [8:0] is set, a DIMM on the even channel caused the error. If any of the bits [17:9] is set, a DIMM on the odd channel caused the error. See Table 3-49.

For uncorrectable errors the NRECMEMA.RANK register is used to identify the failing DIMM pair (lockstep channels).

After a mirrored branch is taken off line, BIOS can execute MemBIST routines on the suspect DIMM-Pair to reproduce failures. This can be performed out-of-band using the SPD (SM bus) interface.

5.3.10.3 ECC Code

When branches operate in dual-channel mode, the MCH supports the 18 device DRAM failure correction code (SDDC aka SECC) option for FB-DIMM. As applied by Intel 5000X Chipset, this code has the following properties:

- Correction of any x4 or x8 DRAM device failure
- Detection of 99.986% of all single bit failures that occur in addition to a x8 DRAM failure. The Intel 5000X Chipset MCH will detect a series of failures on a specific DRAM and use this information in addition to the information provided by the code to achieve 100% detection of these cases.
- Detection of all 2 wire faults on the DIMMs. This includes any pair of single bit errors.
- Detection of all permutations of 2 x4 DRAM failures.

When the branch operates in single-channel/single-DIMM mode, the Intel 5000X Chipset MCH employs x8 SDDC as in the dual channel case. However, in this case, the ECC RAS feature set is limited for the single DIMM memory subsystem. In the single DIMM mode (for example, nine x8 devices), the SDDC cannot correct single wire fault (stuck-at) errors or permanent full device errors. This is because the error correction capability in the SDDC is limited to adjacent symbol errors on a 16-bit boundary and in the single DIMM mode with a Burst Length of 8, there are 4 transfers of 8B to form a 32Bcode word. Hence a single wire failure in the same device is replicated across all 4 symbols hampering the error correction. The SDDC can detect most x4/x8 DRAM failures but it can only correct adjacent symbol errors that occur within a 16-bit boundary of each code word.

5.3.10.4 Inbound ECC Code Layout for Dual-Channel Branches

The code is systematic: that is, the data is separated from the check-bits rather than all being encoded together. It consists of 32 eight-bit data symbols (DS31-DS0) and four eight-bit Check-bit Symbols (CS3-CS0). The code corrects any two adjacent symbols in error. The symbols are arranged so that the data from every x8 DRAM is mapped to two adjacent symbols, so any failure of the DRAM can be corrected.

Figure 5-1 illustrates the ECC code layout for branch 0. The figure shows how the symbols are mapped on the FB-DIMM branch and to DRAM bits by the DIMM for a transfer in which the critical 16 B is in the lower half of the code-word (A[4]=0). If the upper portion of the code-word were transferred first, bits[7:4] of each symbol would be transferred first on the DRAM interface and in the first six transfers on the FB-DIMM channel. The layout for branch 1 is the same.

The bits of Data Symbol 0 (DS0) are traced from DRAM to FB-DIMM Northbound. The same mapping of symbols to data and code bits applies to Southbound data. The lower nibble (DS0A) consists of DS0[3:0] the upper nibble (DS0B) consists of DS0[7:4]. On the DRAM interface, DS0 is expanded to show that it occupies 4 DRAM lines for two transfers. DS0[3:0] appears in the first transfer. DS0[7:4] appear in the second transfer. DS0 and DS1 are the adjacent symbols that protect the eight lines from the first DRAM on DIMM0. The same DS0 is shown expanded on the Northbound FB-DIMM interface where it occupies the FD0NB[P:N][0] signal. DS0 and DS1 cover all transfers on FD0NB[P:N][0] (even though FD0NB[P:N][0] does not cover all of DS1).

Figure 5-1. Code Layout for Single-Channel Branches

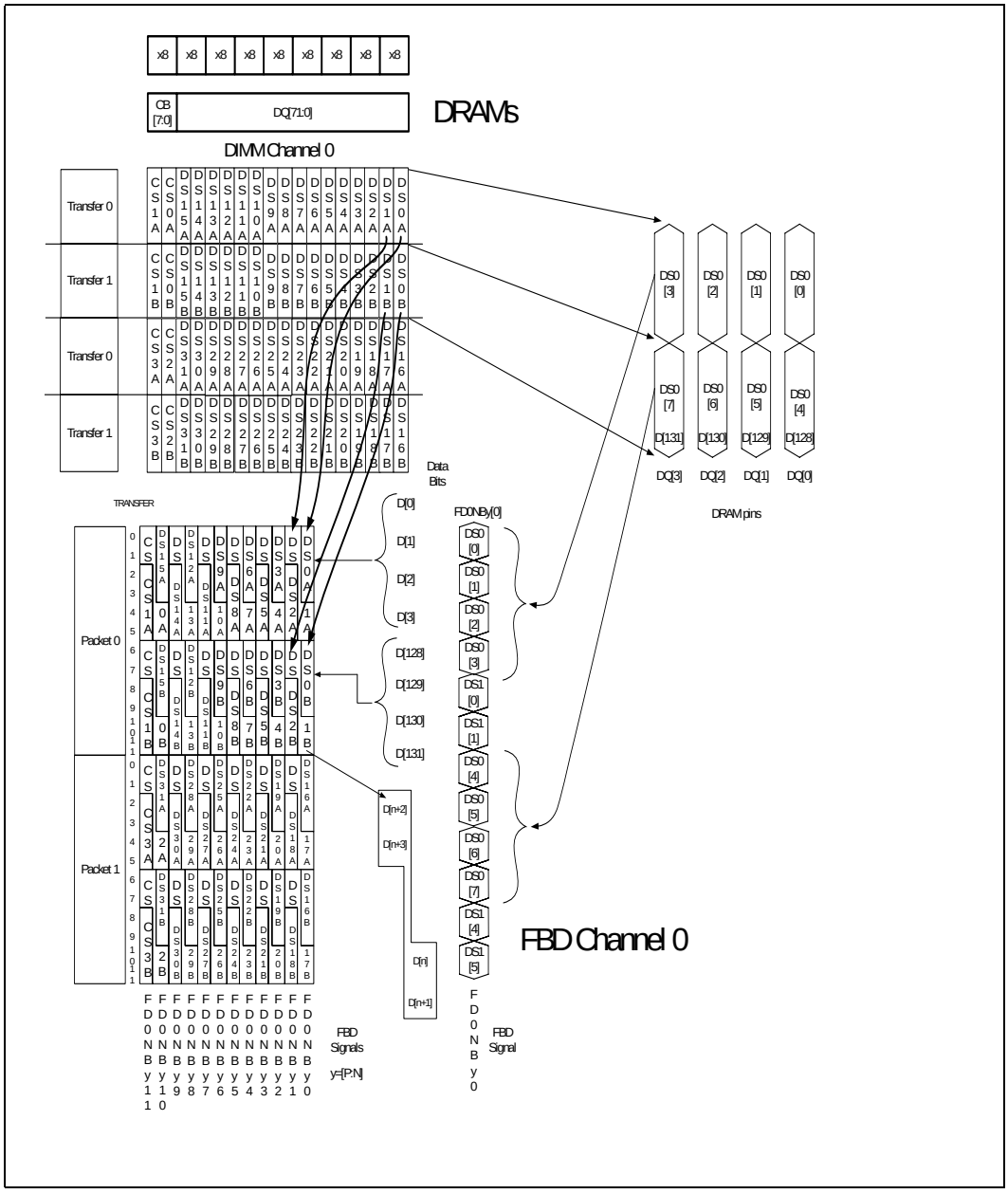
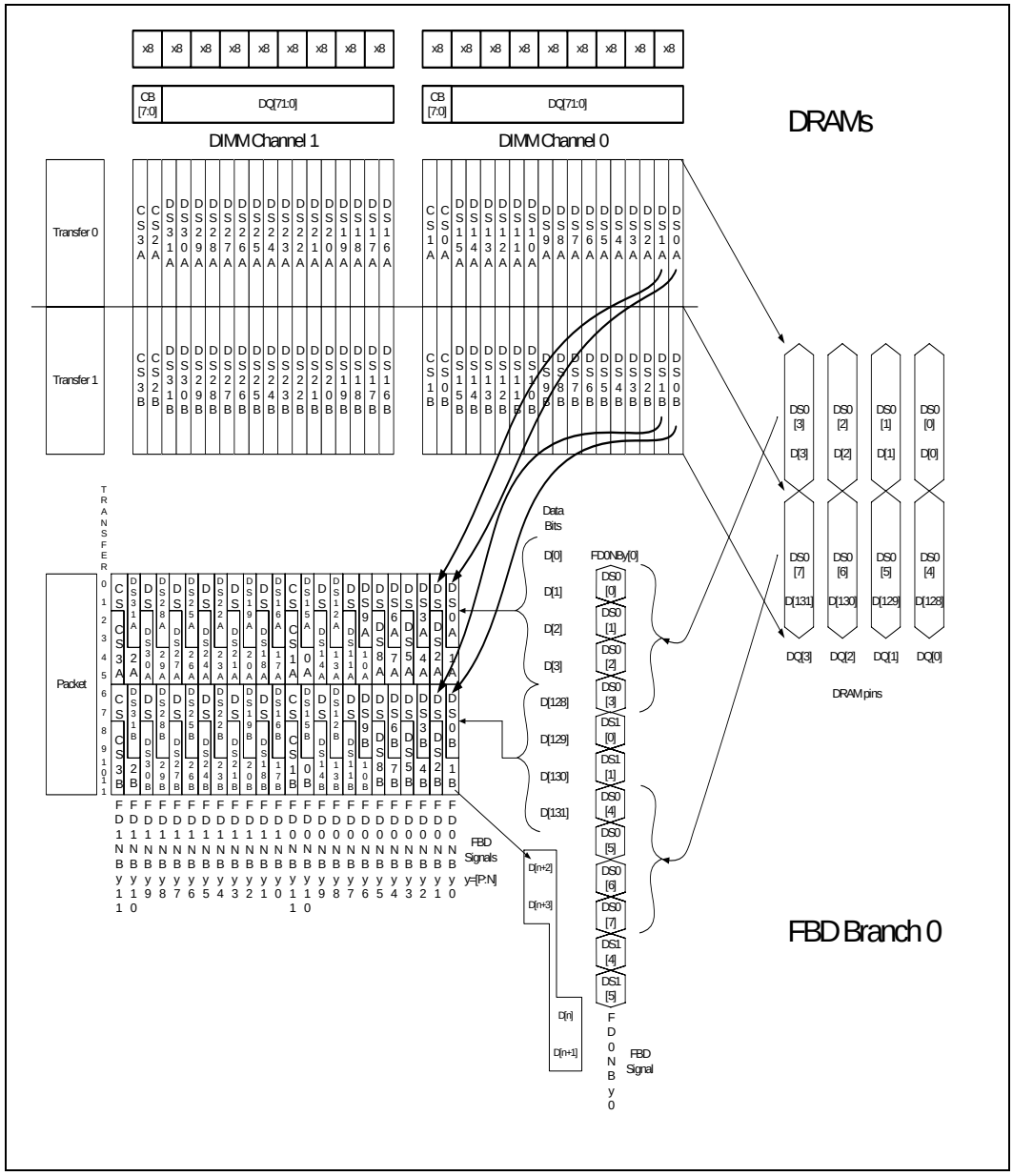


Figure 5-2. Code Layout for Dual-Channel Branches



5.3.10.5 ECC Code Layout for a Single-Channel Branch

The ninth byte of each burst on each DIMM contains the ECC bits for 8 bytes of data. These nine bytes comprise a code word. There are eight code words in a cache line.

5.3.11 DDR2 Protocol

5.3.11.1 Posted CAS

Posted CAS timing is used.

5.3.11.2 Refresh

Regardless of the number of DIMMs installed, each rank will get a minimum of one refresh every eight periods defined by the DRT.TREF configuration register field. The refreshes cycle through all eight DIMM ranks.

The DIMM enters self-refresh mode during an FB-DIMM fast reset.

5.3.11.3 Access Size

All memory accesses are 64 B.

5.3.11.4 Transfer Mode

Each DIMM is programmed to use a burst-length of 32 bytes (4 transfers) across the channel. The Mode Register of each DIMM must be programmed for a burst length of 4, and interleave mode.

5.3.11.5 Invalid and Unsupported DDR Transactions

The memory controller prevents cycle combinations leading to data interruption or early termination. The memory controller prevents combinations of DDR commands that create bus contention (that is, where multiple ranks would be required to drive data simultaneously on a DIMM). The memory controller does not interrupt writes for reads. A precharge command is provided, but early read or write termination due to precharge is not supported.

5.3.12 FB-DIMM IBIST Support

FB-DIMM IBIST support is covered in Section 6.3.2.

5.3.13 Memory Thermal Management

The Intel 5000X Chipset MCH supports FB-DIMM throttling and power management through several mechanisms. The first mechanism forces power down of unused or failed channels by setting the FB-DIMMHPC.State to Reset. This corresponds to the Disable state in the FB-DIMM specification which holds the FB-DIMM channel in reset. When in reset, the channel receivers and drivers are disabled.

The second method of power management utilizes an adaptive methodology to control the number of activations (memory requests) sent to a FB-DIMM. This methodology is composed of two components:

1. Activation throttling: This is composed of closed and open loop throttling mechanisms to control the number of activations sent to FB-DIMM devices.
   a. Closed loop thermal activation control is based on the temperature of the FB-DIMM device. This mechanism becomes active when the FB-DIMM device temperature exceeds programmed thresholds.
   b. Open loop global activate control becomes active when the number of activates exceeds a programmed number with in a long window period.
2. Electrical throttling is used to prevent silent data corruption by limiting the number of activates per rank with in a short sliding window period.

5.3.13.1 Closed Loop Thermal Activate Throttle Control

Closed loop thermal activate throttling control uses the temperature of the FB-DIMM temperature sensor located in the AMB to determine when to throttle. FB-DIMM (AMB temperature) is returned each sync packet. A thermal throttle period is defined as window consisting of 1,344 cycles (42*32). The throttling logic in the memory controller uses this information to limit the number of activates to any DIMM within a throttling window based on temperature threshold crossing algorithm described later.

Every 42 frames the host controller is required to send a sync[1] packet, which returns a status packet from the AMBs along with temperature information. The AMB component has two temperature threshold points, T_{low} (programmed into the GB.TEMPLO register, a.k.a. T1) and T_{mid} (programmed into the GB.TEMPMID register, a.k.a. T2), and the current temperature of the GB with respect to these thresholds are returned in the status packet. In addition, the sync and status packets guarantees that enough transitions occur on each lane to maintain proper bit lock.

1. The sync packet may be dispatched by the MCH at an interval less than 42 frames depending on the gear ratios, timing, circuit and other parameters. For example, in the case of the core running at 266 MHz and the DDR2 clock at 333 MHz (4:5), the MCH can send a SYNC every 32@266 MHz=40 DDR2 333 MHz clocks. This meets the minimum 42 clock requirement of the FB-DIMM protocol for sync packet generation frequency.

In each 42 frame period:

Frames 1-40 are used for normal DRAM traffic: The A slot for DRAM commands and B/C for write data, as necessary.

Frame 41 is used for configuration commands: If a configuration read, it will appear in the A slot. If a configuration write it will appear in B/C (which is the only choice).

Frame 42 is used by the Sync packet and occupies the A, B, and C slots.

FB-DIMM thermal information is returned in the Sync packet as an encoded 2 bit field. The encoding of this field described in the following table.

Table 5-8. AMB Thermal Status Bit Definitions

S[2:1]	**Thermal Trip:** This field indicates various thermal conditions of the AMB as follows:
00	Below TEMPLO
01	Above TEMPLO
10	Above TEMPMID and falling
11	Above TEMPMID and rising

This data is a duplication of the contents of the AMB.FBDS0 register discussed in the *Gold Bridge Component External Design Specification*. These 2 bits are returned for each AMB during in the Status packet.

The TEMPLO threshold is generally used to inform the host to accelerate refresh events. The TEMPMID threshold is generally used to inform the host that a thermal limit has been exceeded and that thermal throttling is needed.

There are separate counters associated with each of the 2 lockstep FB-DIMM pairs in a given branch (one counter per FB-DIMM pair per branch). When any of the counters reaches its limit (as specified by the THRTSTS.THRMTHRT register field for a given

branch), the entire branch is throttled until the end of the throttle window. No new DRAM commands are issued to any of the DIMMs on the branch until the end of the throttle window. If an activate has been issued to a bank, the follow on read or write may be issued, including an additional page hit access if applicable, to allow the page to close.

5.3.13.2 Sequence of Actions During Throttling

When throttling begins during a given throttling window, the following actions take place:

1. Stop new DRAM commands
2. Wait “X” clocks for DRAM commands to complete. Where “X” is the worst case delay as defined below
3. Assert CKE low
4. Wait for throttling window to expire
5. Just before end of activation throttle window (about 3 clocks before for the CKE setup), Assert CKE high

Once the branch has been throttled, the memory controller sends a broadcast CKE for each DIMM command to take the CKE low on all DIMMs of the branch. This command is sent after the proper time has elapsed so that the outstanding transfers complete properly on the DRAMs. When activation throttling starts, CKE must not go low on the DRAMs until the last command has completed in the DRAMs. The worst case is an activate immediately followed by a posted CAS. A fixed time from the last command is used by the Intel 5000X Chipset MCH corresponding to the worst case delay (X) defined by

$$X = \mathrm{Max}(\mathrm{worst\_case\_round\_trip\_delay},\ M \times \mathrm{TRFC})$$

$$M = \begin{cases} 1.25 & \text{if Core to FBD clock ratio is 5:4} \\ 1 & \text{Otherwise} \end{cases}$$

with a suitable guard band[9] to protect any data loss. The TRFC parameter (Refresh to Activate Command delay) is factored into the equation since a refresh could be just underway when the last activate was about to be issued. The “1.25” scaling factor is to account for the 5:4 gearing ratio required for a FSB frequency of (333MHz) and FBD/DDR clock frequency of (266MHz). The default scale factor used on platforms is 1 (FSB [266 MHz] and FBD/DDR clocks [266 MHz]).

During the time that CKE is low, no DRAM commands should be sent on the channel. However, Non-DRAM commands such as Configuration register and SYNC are required to be sent during this period.

When the throttle window is about to expire, a CKE command is sent to take all CKEs high. This must be done at least 3 clocks before the first command.

9. The worst case round trip delay is expected to be in the 10-20 clock range for a posted CAS command and the Intel 5000X Chipset MCH RTL can be microarchitected appropriately.

5.3.13.3 CKE State Near End of Activation Throttling Window

If the throttling begins very close to the end of the window, then the assertion of CKE low command would be delayed beyond the end of the throttle window. To prevent this occurrence, the memory controller logic does not observe a throttle event in the last few clocks of the window, or assert a CKE low command.

If the activation throttle is set to begin within Y clocks before the end of the window, the memory controller skips the asserting CKE low step, where Y is X + 6[10] (and the number “6” is derived from 3 clocks for the CKE low to high minimum, plus another 3 clocks for the CKE high until first command after the throttling window.

5.3.13.4 Refresh Handling During Throttling

The Intel 5000X Chipset memory controller ensures that refreshes, which are lost during the activation throttle period (possibly up to 2), are made up at the end of the period. Double refresh rates to the DIMMs should be carried out when needed regardless of the setting of the MC.THRMHUNT bit. This is particularly important for open loop throttling when the temperature could rise beyond 85°C.

5.3.13.5 Throttling Parameters for Activation Throttling.

The current throttling parameters for each branch are stored in the THRMTHRT register field defined in Section 3.9.3. All activation throttling parameters in the THRMTHRT registers are 8-bits wide, and provide increments of 4 activations per throttle window (1344 clocks). Three levels of throttling limits are defined.

- THRTLOW: A base throttling level that is applied when the temperature is in the low range (below T_{low}) and the THRTSTS.GBLTHRT*[11] bit is not set by the Global Throttling Window logic. See Section 3.9.4
- THRTMID: A mid level throttling level that is applied when the temperature is in the middle range (above T_{low} but below T_{mid}) or the THRSTS.GBLTHRT* bit is set by the Global Throttling Window logic. See Section 3.9.5
- THRTHI: The highest level of throttling. When MC.THRMODE=1, this level is applied whenever the temperature is above T_{mid}. When MC.THRMODE=0, this level is the ceiling of the hunting algorithm of the closed loop throttling. The temperature being above T_{mid} has priority over the Global Throttling Window throttling (the higher throttling level takes precedence). See Section 3.9.6

The MC.THRMHUNT bit must be enabled for the temperature to have any influence on the throttle parameters. If MC.THRMHUNT=0, only the GBLTHRT bit from the Global Throttle Window, when enabled can change the THRMTHRT register field. Refer to Figure 5-9 and Figure 5-10 for the thermal envelopes.

10. Intel 5000X Chipset MC design needs to adjust the value based on the latest JEDEC recommendation for CKE low to high transition.
11. GBLTHRT* is an internal combinatorial signal before it is latched in the THRSTS.GBLTHRT register field to enable the open loop throttling logic to use the latest value of the signal

Figure 5-9. Thermal Throttling with THRMHUNT=1

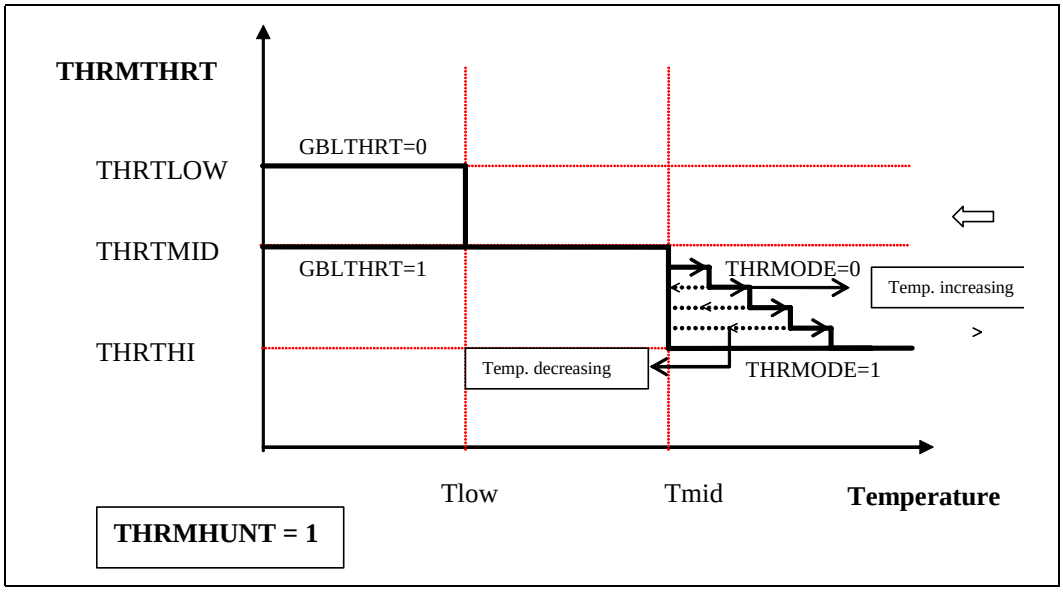
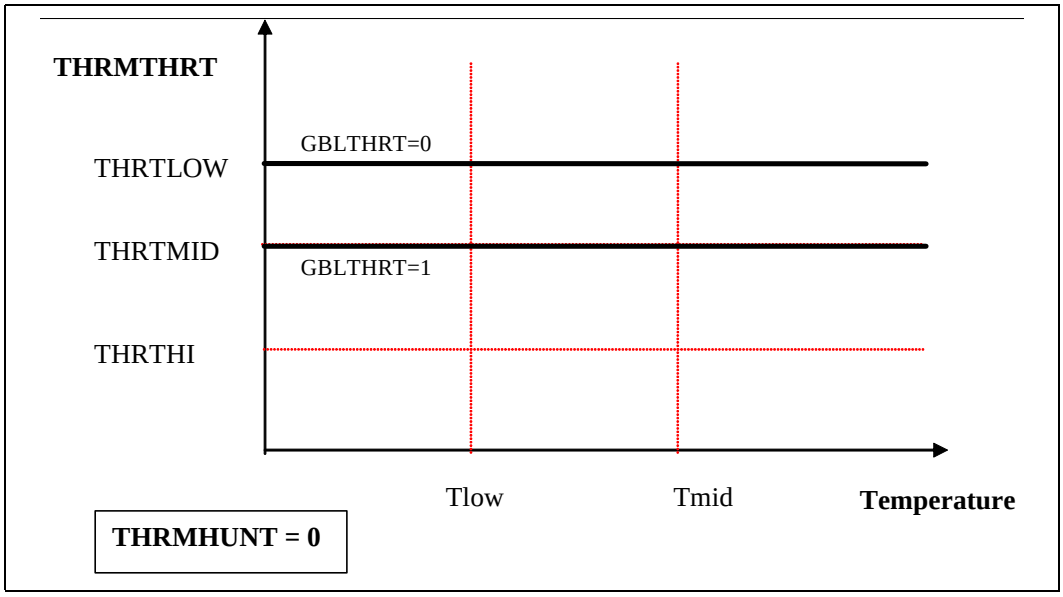


Figure 5-10. Thermal Throttling with THRMHUNT=0



5.3.13.6 Closed Loop Activation Throttling Policy

Individual DIMMs flag their thermal state in the FB-DIMM status return. When the MC.THRMHUNT configuration bit is set, memory reads and writes (summed together) will be regulated on a per-DIMM-pair basis according to the following algorithm described in Figure 5-11. Note that Intel 5000X Chipset MCH provides a greater degree of thermal throttling if there is a sudden temperature spike between from T_{low} to T_{mid} by setting the THRMTHRT register to THRTMID as a starting point when MC.THRMODE=0. Once this point is reached, if temperature increased further during the next global window, then THRMTHRT register will be adjusted by the equation THRMTHRT= MAX(THRMTHRT -2, THRTHI). See staircase effect in Figure 5-9. If temperature decreased but is still greater than T_{mid}, then the THRMTHRT will retain its

last value. This provides some degree of hysteresis control to allow the DIMMs to cool further before THRMTHRT jumps back to a larger number (that is, less throttling) at the junction when the temperature reached T_{mid}. Refer to the dotted line in Figure 5-9. This scheme helps in reducing the thermal power by limiting the number of activates. See Figure 5-11 for further details.

1. Staircase Conditioning [THRTCTRL.THRMODE=0]: This method is employed when THRTCTRL.THRMODE=0 and temperature crosses above Tmid . The THRMTHRT registries capped to THRTMID (starting point) and it uses a linearly increasing (less aggressive) throttling algorithm to reduce activations and balance performance and power envelope when temperature rises and falls around Tmid point. Once THRTMID is reached, if temperature increases further during the next global window, then THRMTHRT register will be adjusted by the equation THRMTHRT= MAX(THRMTHRT -2, THRTHI). This produces the staircase effect as shown in Figure 5-10, “Thermal Throttling with THRMHUNT=1” on page 387. If temperature decreases subsequently but is still greater than Tmid, then the THRMTHRT will retain its last value. This provides some degree of hysteresis control to allow the DIMMs to cool further before THRMTHRT jumps back to a larger number (i.e less throttling) at the junction when the temperature reached Tmid. Refer to the dotted line in Figure 5-10, “Thermal Throttling with THRMHUNT=1” on page 387. This scheme helps in reducing the thermal power by limiting the number of activates. See Figure 5-12, “Thermal throttling Activation Algorithm” on page 389 for further details.
2. Step Conditioning (brute force) [THRTCTRL.THRMODE=1]: This method is employed when THRTCTRL.THRMODE=1 and temperature crosses Tmid . The THRMTHRT register is capped to THRTHI and it provides a greater degree of throttling by allowing fewer activates to the memory allowing the DIMM to cool down quicker but at the expense of performance. This can be used to control sudden temperature surges that moves the envelope from below Tlow to above Tmid. and stays there for a long period.

During the global window, the Intel 5000X Chipset MCH will broadcast one configuration write to the DIMMs AMB.UPDATED registers. This write will not be re-played or re-sent.

Note: A channel fault could drop an AMB.UPDATED write. If the temperature increased during the previous global window, but had not actually increased during the current global window, then THRMTHRT will un-necessarily decrease. If the temperature had not increased during the previous global window, but had actually increased during the current global window, then THRMTRHT will remain unresponsive to the temperature increase for one global throttling window. The situation will rectify itself in the next global throttling window.

1. If there is a sudden temperature spike between from below Tlow to above Tmid by setting the THRMTHRT register to THRTMID as a starting point when THRTCTRL.THRMODE=0. If temperature rose from above Tlow to above Tmid, then the THRMTHRT will use THRTMID value if THRTCTRL.THRMODE=0; otherwise it will use THRTHI if THRTCTRL.THRMODE=1. See the right side of Table 5-8 on page 390 for the various modes.

Figure 5-11. Thermal Throttling Activation Algorithm

```
THRMTHRT = THRTLOW (Initialize to base-level Activations)

if (Global_Timer expires)
{
    if (MC.GTW_MODE == 1)   // Choose window size based on mode setting
    {   Global_Timer = 4*1344       // Validation & Debug Mode
     }
     else
    {   Global_Timer = 0.65625*2^25 = 16384*1344
         // make global throttling window an integral multiple of the closed loop window
     }

     if (THRTCTRL.THRMHUNT == 1)
     {
          for (each DIMM-pair [m] on each branch [n]) /* m=0..3, n=0..1 */
          {     if (temperature of any DIMM[i] >= Tmid) /* 0 <= i <= 3 */
                {     if (THRTCTRL.THRMODE == 0)
                            if (THRMTHRT > THRTMID)
                            {  /* This will cap the start point to THRTMID if there is a  */
                              /*  spike in the GB Temperature from Tlow to Tmid & beyond */
                              /*  Provides better throttling and control */
                                THRMTHRT = THRTMID
                            }
                            else      /* Staircase roll down may happen for subsequent samplings
                            {    if (the temperature of any DIMM which is above Tmid, increased)
                                 {    THRMTHRT = max(THRMTHRT - 2, THRTHI)
                                 } /* Otherwise retain last THRMTHRT value */
                            } /* end of THRMTHRT > THRTMID check */
                      }
                      else
                      {     THRMTHRT = THRTHI
                      } /* end of THRMODE==0 check */
                }
                else if ((temperature of any DIMM[i] >= Tlow[i])
                                  && (temperature of all DIMM's[i] < Tmid[i]))
                {     THRMTHRT = THRTMID
                }
                else if (temperature of all DIMM's[i] < Tlow[i]))
                {     if (GBLTHRT == 1)
                      {     THRMTHRT = THRTMID
                      }
                      else
                      {     THRMTHRT = THRTLOW
                      }
                }
          }
     }
     else
    {    if (GBLTHRT == 1)
         {     THRMTHRT = THRTMID
         }
          else
         {    THRMTHRT = THRTLOW
         }
    }
}
```

Table 5-9. FB_DIMM Bandwidth as a Function of Closed Loop Thermal Throttling

THRMTHRT Reg Value	Activates	% BW allowed	BW per DIMM GB/s	sys BW, 1 DIMM/ch	sys BW 2 DIMM/ch	sys BW 4 DIMM/ch
0	unlimited					
1	4	0.60%	0.03	0.13	0.25	0.51
2	8	1.19%	0.06	0.25	0.51	1.02
3	12	1.79%	0.10	0.38	0.76	1.52
4	16	2.38%	0.13	0.51	1.02	2.03
5	20	2.98%	0.16	0.63	1.27	2.54
6	24	3.57%	0.19	0.76	1.52	3.05
7	28	4.17%	0.22	0.89	1.78	3.56
8	32	4.76%	0.25	1.02	2.03	4.06
12	48	7.14%	0.38	1.52	3.05	6.10
16	64	9.52%	0.51	2.03	4.06	8.13
20	80	11.90%	0.63	2.54	5.08	10.16
24	96	14.29%	0.76	3.05	6.10	12.19
28	112	16.67%	0.89	3.56	7.11	14.22
32	128	19.05%	1.02	4.06	8.13	16.25
36	144	21.43%	1.14	4.57	9.14	18.29
40	160	23.81%	1.27	5.08	10.16	20.32
44	176	26.19%	1.40	5.59	11.17	
48	192	28.57%	1.52	6.10	12.19	
64	256	38.10%	2.03	8.13	16.25	
72	288	42.86%	2.29	9.14	18.29	
80	320	47.62%	2.54	10.16	20.32	
96	384	57.14%	3.05	12.19		
128	512	76.19%	4.06	16.25		
144	576	85.71%	4.57	18.29		
160	640	95.24%	5.08	20.32		
168	672	100.00%	5.33	21.33		

5.3.13.7 Open Loop Global Throttling

In the open loop global window throttling scheme, the number of activates per DIMM pair per branch is counted for a larger time period called the “Global Throttling window”. The Global throttling window is chosen as an integral multiple of the thermal throttling window of 1344 clocks for maintaining a simpler implementation. Under normal operating conditions, the Global Throttling Window is $0.65625 * 2^{25}$clocks in duration and this translates to 16384*1344 clocks (~66.06 ms) for DDR2667. However, for purposes of validation and debug, the global throttling window can be reduced to a smaller duration, 4*1344 cycles[12] (16.128 µs) for DDR2667 and this is controlled through the GTW_MODE register bit defined in Section 3.9.1. The global throttling window prevents shorts peaks in bandwidth from causing closed loop activation throttling when there has not been sufficient DRAM activity over a long period of time to warrant throttling. It is in effect a low pass filter on the closed loop activation throttling.

12. If MC.GTW_MODE=1, the Intel 5000X Chipset MCH will use the 4*1344 cycle duration for the global throttling window.

During this Global throttling window, the number of activates is counted for each DIMM pair per branch (24-bit counters are required). If the number exceeds the number indicated by the GBLACT.GBLACTLM register defined in Section 3.9.2, then the THRTSTS[1:0].GBLTHRT bit is set for the respective branch, causing the activation throttling logic to use the THRTMID register. The THRTSTS[1:0].GBLTHRT will remain active until 16 (or 2) global throttling windows in a row have gone by without any DIMM exceeding the GBLACT.

At the end of the 16 (or 2) global throttling windows, if no DIMM pair activates exceed the GBLACT.GBLACTLM value, then the MC indicates the end of the period by clearing the THRTSTS[1:0].GBLTHRT register field.

If part way through the count of 16 (or 2) global throttling windows, the GBLACT.GBLACTLM is again exceeded within one Global Throttle Window, the counter gets reset and it will once again count 16 (or 2) global throttle windows throttling at the THRTMID level.

5.3.13.8 Global Activation Throttling Software Usage

In practice, the throttle settings for THRTMID are likely to be set by software such that the memory controller throttle logic will actually prevent the GBLACT limit from being exceeded and the result will often be that such that THRTLOW is used for a Global Throttle Window, at which time, the GBLACT.GBLACTLM is exceeded, causing the MC s to use a larger throttling period THRTMID for 16 (or 2) global[13] windows. During each of those global windows, GBLACT limit is not exceeded, because the throttling will prevent it from being exceeded. After 16 (or 2) global[2] throttling windows, it switches back to THRTLOW, and on the next global window GBLACT is again exceeded, causing another 16 (or 2) windows[2]. Hence, we can get a cumulative pattern of 16,1,16,1 (or 2,1,2,1) global[2] throttling windows and this prevents excessive heat dissipation in the FB-DIMMs by prolonging the throttle period.

Note: It should be mentioned that the open and closed loop throttling control policies implemented on the Intel 5000X Chipset MCH uses the internal core clocks for the calculating the windows and not the DDR clocks. Thus any software/BIOS should take this into account for manipulating the THRMTHRT registers when dealing with different FB-DIMM technologies and speeds.

5.3.13.9 Dynamic Update of Thermal Throttling Registers

In general, the Intel 5000X Chipset registers should not updated dynamically during runtime as it may interfere with the internal state machines not designed exclusively for such changes and could result in a system hang/lock up. This requirement is relaxed (subject to validation) for the Intel 5000X Chipset thermal throttling registers where it is desirable for BIOS or special OEM software in BMC to exercise dynamic control on throttling for open/closed loop algorithm implementation. The following examples are some of the potential areas of this usage model where dynamic change is needed to balance performance and acoustic levels in the system

Fan control for CPU temperature related system acoustics or other BMC related operations

Limit hacker activity by increasing memory throttling via throttle register updates to condition the system based on some event (excessive bandwidth or CPU activity)

- Fan failure/breakdown. When this occurs, temperature conditioning can be provided by reducing the activity level in the DIMMs to a certain threshold until the failed fan can be repaired by the technician and service restored to normalcy.

13.The 2 window Global Throttling count will be chosen if MC.GTW_MODE=1.

5.3.13.10 General Software Usage Assumptions

Under normal circumstances, it is expected that there is no change of throttling values once it is configured by BIOS during boot. The external Fan control and the BIOS settings of the OEM via BMC would ensure adequate cooling and maintain the DIMMs within the prescribed tolerance limits of the TDP. However, situations such as thermal virus or fan fail down condition might warrant the BIOS/SW to take preemptive action in adjusting the throttling to say 40-70% of the normal mode before it is cleared. This means that changes to throttling registers can happen at random intervals (infrequent) and the platform should be able to tolerate any transients changes that may result when the Intel 5000X Chipset is updated with the new throttle values. These requirements are captured below.

5.3.13.11 Dynamic Change Operation Requirements for Open Loop Thermal Thottling (OLTT)

The Intel 5000X Chipset Memory throttle control register affected by OLTT include THRTMID (T2), THRTLOW (T1), GBLACT, and the THRTCTRL.THRMHUNT field. (THRMHUNT=0 selects the open loop mode).

Each update to the above mentioned throttle register takes approximately 40 core clocks in the configuration ring to complete.

Configuration register updates for throttling should be spaced out at approximately 80 core cycles apart. (2x guard band)

Only one CFC/CF8 or MMCFG configuration transaction is allowed at a time in the system.

When the number of activates exceed the GBLACT.GBLACTLM in a global throttling window, OLTT is entered and *GBLTHRT is set by the Intel 5000X Chipset for 16 consecutive global throttling windows* (irrespective of the new parameters) as described in Section 5.3.13.7. Note that OLTT is NOT history-based algorithm. Hence if software assigns new values to THRTLOW or THRMID values at some point in time, the MC cluster will update the registers and use the new values for limiting the activates immediately via THRMTHRT register for 16 consecutive global throttling windows. See also Figure 5-11.

Software can update the throttling registers as frequently as it desires provided it maintains the minimum spacing for the configuration writes and follows the other guidelines as described above. It is also software's responsibility for the fallout/ transient effect of the thermal control algorithm during such updates.

5.3.13.12 Dynamic Change Operation Requirements for Closed Loop Thermal Thottling (CLTT)

In addition to all the conditions/requirements as stipulated in Section 5.3.13.11, the closed loop throttling which uses GB temperature feedback to adjust the throttling levels requires the following:

- Intel 5000X Chipset Registers that are affected by dynamic updated include THRTMID (T2), THRTLOW (T1), THRTHI and THRTCTRL.THRMHUNT. (THRMHUNT=1 selects the closed loop mode)
- •When Temperature crosses T_{mid}, the CLTT switched to either the staircase function (if THRMCTRL.THRTMODE=0) or the single step function (THRMCTRL.THRMODE=1) as depicted in the right side of Figure 5-9. See also Figure 5-11. Note that CLTT is NOT history-based algorithm except in the staircase mode which uses the old value. In this case, the staircase function that always decrements the old value of

THRMTHRT by 2. By design THRMTHRT can never be below THRTHI. If the new THRTHI is greater than THRMTHRT, then the algorithm will reset THRMTHRT to THRTHI & the staircase function can no longer be used since the bottom (aka ceiling) of THRTHI has already been reached as defined by the following equation extracted from Figure 5-11.

THRMTHRT = max(THRMTHRT - 2, THRTHI)

5.3.13.13 Disabling Closed/Open Loop Throttling

The following registers in the Intel 5000X Chipset can be initialized to disable throttling (open/closed) if software desires to turn off throttling.

- THRTCTRL.THRMHIUNT = 0 /* This forces the Intel 5000X Chipset to ignore temperature for closed loop */
- THRTHI.THRHILM = 0 (or 168d)
- THRTHI.THRMIDLM = 0 (or 168d)
- THRTLOW.THRLOWLM = 0 (or 168d)
- GBLACT.BLACTLM = 0 (or 168d) /*Above changes force Open loop throttling to be off */

Table 5-10. Global Activation Throttling BW allocation as a function of GBLACTLM for a 163841344 window with MC.GTW_Mode=0 (normal)**

GBLACT. GBLACTLM	# of Activates	% BW allowed	BW per DIMM GB/s	sys BW, 1 DIMM/ch	sys BW 2 DIMM/ch	sys BW 4 DIMM/ch
0	unlimited					
1	65536	0.60%	0.03	0.13	0.25	0.51
2	131072	1.19%	0.06	0.25	0.51	1.02
3	196608	1.79%	0.10	0.38	0.76	1.52
4	262144	2.38%	0.13	0.51	1.02	2.03
5	327680	2.98%	0.16	0.63	1.27	2.54
6	393216	3.57%	0.19	0.76	1.52	3.05
7	458752	4.17%	0.22	0.89	1.78	3.56
8	524288	4.76%	0.25	1.02	2.03	4.06
12	786432	7.14%	0.38	1.52	3.05	6.10
16	1048576	9.52%	0.51	2.03	4.06	8.13
20	1310720	11.90%	0.63	2.54	5.08	10.16
24	1572864	14.29%	0.76	3.05	6.10	12.19
28	1835008	16.67%	0.89	3.56	7.11	14.22
32	2097152	19.05%	1.02	4.06	8.13	16.25
36	2359296	21.43%	1.14	4.57	9.14	18.29
40	2621440	23.81%	1.27	5.08	10.16	20.32
44	2883584	26.19%	1.40	5.59	11.17	
48	3145728	28.57%	1.52	6.10	12.19	
64	4194304	38.10%	2.03	8.13	16.25	
72	4718592	42.86%	2.29	9.14	18.29	
80	5242880	47.62%	2.54	10.16	20.32	
96	6291456	57.14%	3.05	12.19		
128	8388608	76.19%	4.06	16.25		
144	9437184	85.71%	4.57	18.29		
160	10485760	95.24%	5.08	20.32		
168	11010048	100.00%	5.33	21.33		

5.3.14 Electrical Throttling

Electrical throttling is a mechanism that limits the number of activates (burstiness) within a very short time interval that would otherwise cause silent data corruption on the DIMMs. Electrical throttling is enabled by setting the MTR.ETHROTTLE bit defined in Section 3.9.24.7. These bits occur on a per DIMM pair basis per branch as to whether electrical throttling should be used. It is assumed that both ranks within a DIMM would be the same technology, and therefore does need not separate enable bits.

The per rank electrical throttling for FB-DIMM is 4 activates per 37.5 ns window (JEDEC consensus) and is summarized in Figure 5-11 for various DIMM technologies.

Table 5-11. Electrical Throttle Window as a Function of DIMM Technology

DIMM Modes	Intel 5000X Chipset MCH Core: FB-DIMM clock Ratio	Electrical Throttle Window[a] (in core clocks per rank per DIMM pair per branch)
DDR533	1:1	10
	5:4	13
DDR667	1:1	13
	4:5	13 (conservative)
DDR800[b]	All	15
Conservative (safe mode)	All	20

Notes:

a. Maximum 4 activates per rank is allowed within the window.
b. This is not a supported technology/nor a POR for Intel 5000X Chipset MCH and is tabulated for information/illustrative purposes only.

The MC.ETHROT configuration register field limits the number of activations per sliding electrical throttle window. The memory controller logic can implement the sliding electrical throttle window with a 20-bit shift register per rank in each DIMM pair per branch. This register records for the last 20 clocks, whether an activate was issued or not to that rank. The number of activates can then be summed up from the state of the shift register and compared with the respective limit as shown in Figure 5-11. If the limit is reached, then further activates to the rank are blocked until the count falls below the limit. The Electrical throttling logic in the MC masks off the end bits for the DIMM technologies that require fewer clocks. As an example, if the DIMM technology used is DDR667, then it can allow 4 activates within the last 13 clocks, the remaining 7 bits are masked (forced to 0) so they do not prevent activates.

5.4 Intel 5000X Chipset MCH Thermal Sensor

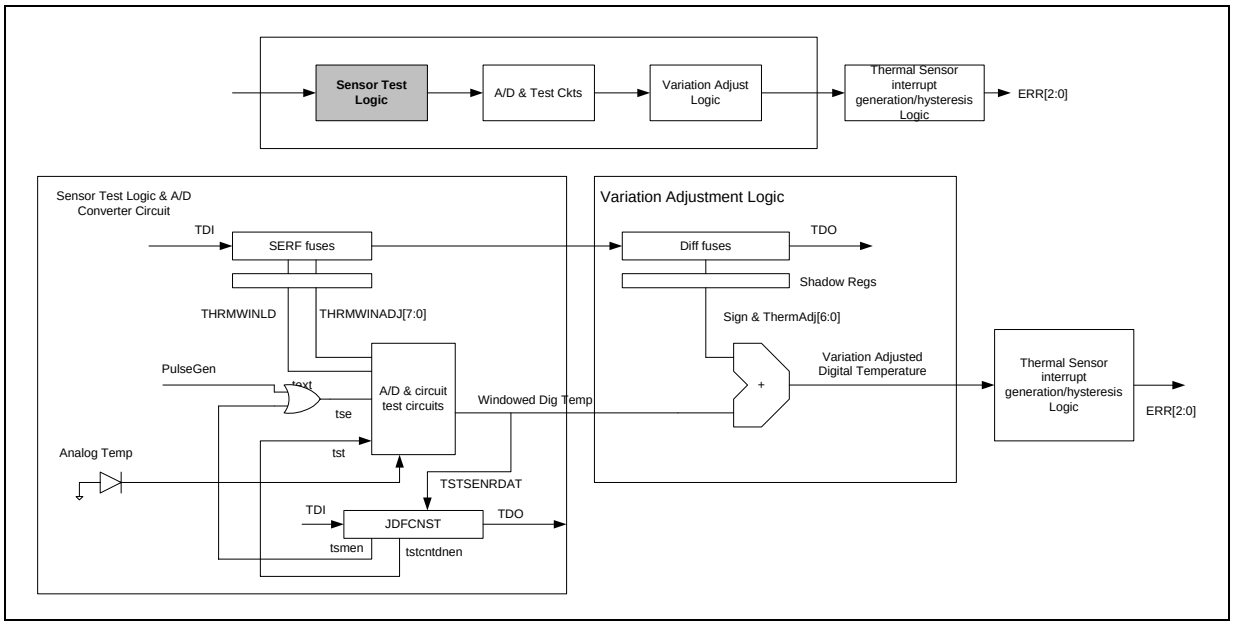
The Intel 5000X Chipset MCH provides an integrated thermal sensor to enable software to monitor the activity level in the system and throttle traffic on the various interfaces (FSB, I/O) if the temperature of the Intel 5000X Chipset MCH increased beyond a certain threshold due to events such as a power virus, hacker activity or a spike in request rate.

The thermal sensor consists of a thermal diode with an 8-bit Successive Approximation (SA) A/D converter that converts the on-die temperature to a digital value from 0 to 127.5 degrees C with a resolution of 0.5 degree C. This output is stored in the THRMSR_OP register. Refer to Section 3.8.13.29 for details. All registers are SMBUS/ JTAG accessible.

Refer to Figure 5-12.

The THRMSR_OP register (value: 0 to 255) latches the 8-bit read out from the A/D converter at an update frequency[14] specified in the THRMSR_CTRL.THRMSR_SAMP_FREQ register field (See Section 3.8.13.36). If software desires to change the THRMSR_CTRL.THRMSR_SAMP_FREQ register field, it should first disable temperature latching by clearing THRMSR_CTRL.THRMSR_LATCH_EN register before setting a new value. This action will reset the internal sampling counter in the Intel 5000X Chipset MCH and enable it for fresh start. Internal analog nodes (anode, cathode, and so forth) of this circuit will not be brought out to package pins to keep noise at a minimum.

Figure 5-12. Thermal Sensor Implementation using Sensor Test Logic, Thermal Diode, Successive Approximation 8-bit A/D Converter Circuit, Variation Adjustment Logic, and Glue Logic for Interrupt Generation

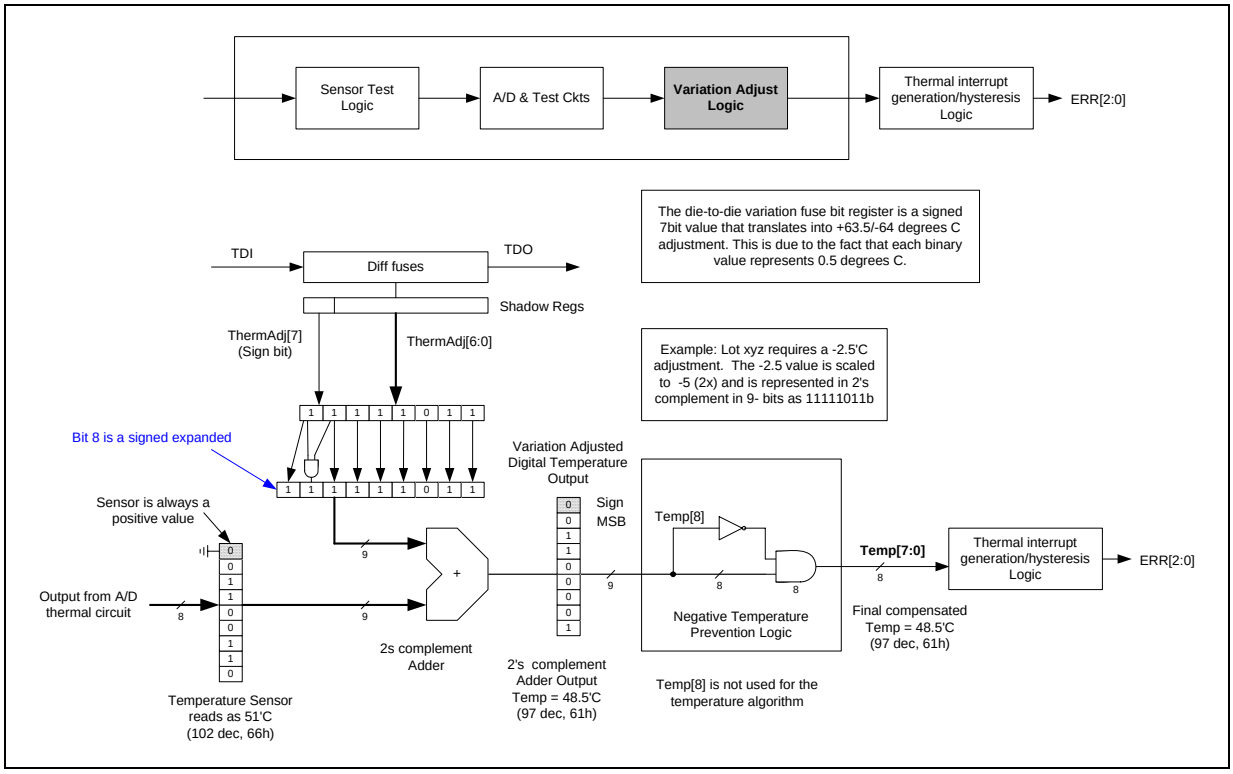


5.4.1 Thermal Sensor Calibration

Calibration of the thermal Sensor is required to improve accuracy and limit guard banding (tolerance) needed for proper operation for each component under a given operating range taking into account the process variations/known environmental conditions for each component during manufacturing test. Figure 5-13 depicts the thermal variation adjust logic functionality. Depending on the process variation/ conditions, the thermal offset (8-bits including sign) is programmed during manufacture in 2's complement notation. This entity is accumulated with the digital temperature read out of the sensor[15]using a 2's complement adder.

14.The default value is to sample temperature every 256 core clocks
15.The digital temperature from the A/D logic is also represented in 2's complemented format.

Figure 5-13. Thermal Variation Adjust Logic Illustrated with an Example



Note: For P861 process, please refer to the latest revision of the Intel 5000X Chipset Thermal/Mechanical Design Guide for more information on the Tjmax (junction temperature).

5.4.2 Thermal Sensor Interrupt/Event Signaling

Figure 5-14 captures the general thermal event signaling logic implemented in the Intel 5000X Chipset MCH along with the thermal sensor.

The chipset provides Thermal Sensor trip point registers, viz.THRMSR_HI and THRMSR_LO registers (8-bits with 0.5'C resolution), which are initialized by software. The default temperature values for T_{sr_hi} and T_{sr_lo} are 100 and 90 degree C respectively.

The outputs of the THRMSR_OP register is compared with THRMSR_LO and THRMSR_HI values along with the state of prior interrupts for signaling. Interrupts on high are generated when the temperature increases and crosses T_{sr_lo} and T_{sr_hi} trip points. Interrupts on low are generated when the temperature decreases and falls below T_{sr_hi} and T_{sr_lo} trip points.

If the sampled temperature, T_s , from the THRMSR_OP registers is such that (T_s > T_{sr_lo}) and (T_s > T_{sr_hi}) and no prior interrupt on high has been generated, then the Intel 5000X Chipset MCH will record the thermal trip event status in the THRMSR_STS_HI.THRMSR_STSEV_HI register field, store the latched temperature from the THRMSR_OP register in the THRMSR_STS_HI.THRMSR_TEMP_HI field and trigger a thermal interrupt on high by asserting the ERR[2:0]# pins. This is an indication to software that the activity level has increased beyond the maximum threshold and attention is required immediately for throttling the traffic to reduce the die temperature otherwise the chipset may fail to operate or burn out.

Similarly, if the sampled temperature, T_s , from the THRMSR_OP registers is such that ($T_s < T_{sr\_hi}$) and ($T_s < T_{sr\_lo}$) and no prior interrupt on low[1] has been generated, then the Intel 5000X Chipset MCH will record the thermal trip event status in the THRMSR_STS_LO.THRMSR_STSEV_LO register field, store the latched temperature from the THRMSR_OP register in the THRMSR_STS_LO.THRMSR_TEMP_LO field and trigger a thermal interrupt on low by asserting the ERR[2:0]# pins. This is an indication to software that the activity level has reduced to safe level and that the on-die temperature is below the T_{sr_lo} trip point.

The thermal interrupt events are masked by the THRMSR_MSK register and routed to one of the ERR[2:0]# pins based on the setting of the THRMSR_DO_CMD register.

It is the responsibility of the software to set the THRMSR_THRLD_HI(LO) registers correctly to guarantee proper event generation.

5.4.2.1 Thermal Sensor Status High/Low

The Intel 5000X Chipset MCH sets the status event fields (THRMSR_STSEV_HI(LO)) in the THRMSR_STS_HI (LO) register when the thermal trip point registers are crossed and a thermal interrupt high (low) has occurred as defined earlier. When the respective status bits are set, the Intel 5000X Chipset MCH also records the value of the sampled temperature, T_s. in the THRMSR_STS_HI(LO).THRMSR_TEMP_HI(LO) fields for logging purposes.

The THRMSR_STS_HI(LO).THRMSR_STSEV_HI(LO) register bit is cleared by software when it has completed servicing the interrupt at which point the ERR[2:0]# pins can be deasserted.

When the status event bits are set, the Intel 5000X Chipset MCH will not generate new interrupts nor record any further status events in the THRMSTS_HI(LO registers till the THRMSTS_HI.THRMSR_STSEV_HI and THRMSTS_LO.THRMSR_STSEV_LO status fields are cleared by software.

When the thermal interrupt high/low is triggered by the Chipset, software can scan the THRMSR_OP, THRMSR_STS_HI(LO) registers to determine the current temperature and status at any time and take actions to regulate (throttle) the traffic on its interfaces.

5.4.3 Thermal Conditioning Example

Figure 5-15 provides an overview of the thermal conditioning in the Intel 5000X Chipset MCH through illustration. When chipset activity increases causing the T_s to move beyond T_{sr_lo}, T_{sr_hi} trip points, an interrupt on high is generated. When software throttles activity, the sampled temperature T_s begins to fall. It crosses the T_{sr_hi} trip point and decreases gradually. When it falls below T_{sr_lo}, then an interrupt on low is generated to indicate that the temperature is under limits.

When a prior interrupt high/low is generated and subsequently if the temperature oscillates between (T_{sr_lo}, T_{sr_hi}) and either increases above T_{sr_hi} or falls below T_{sr_lo}, then no new interrupts are generated. Refer to Figure 5-15 for the points indicated when no interrupt occurs. This method provides thermal conditioning/hysteresis for noise and other external disturbances that may place the temperature above the trip points and cause inadvertent interrupts from being generated. It is the responsibility of the software to ensure sufficient throttling to bring the temperature below the threshold.

Figure 5-14. Block Diagram of Intel 5000X Chipset MCH Thermal Sensor and Event Signaling Logic

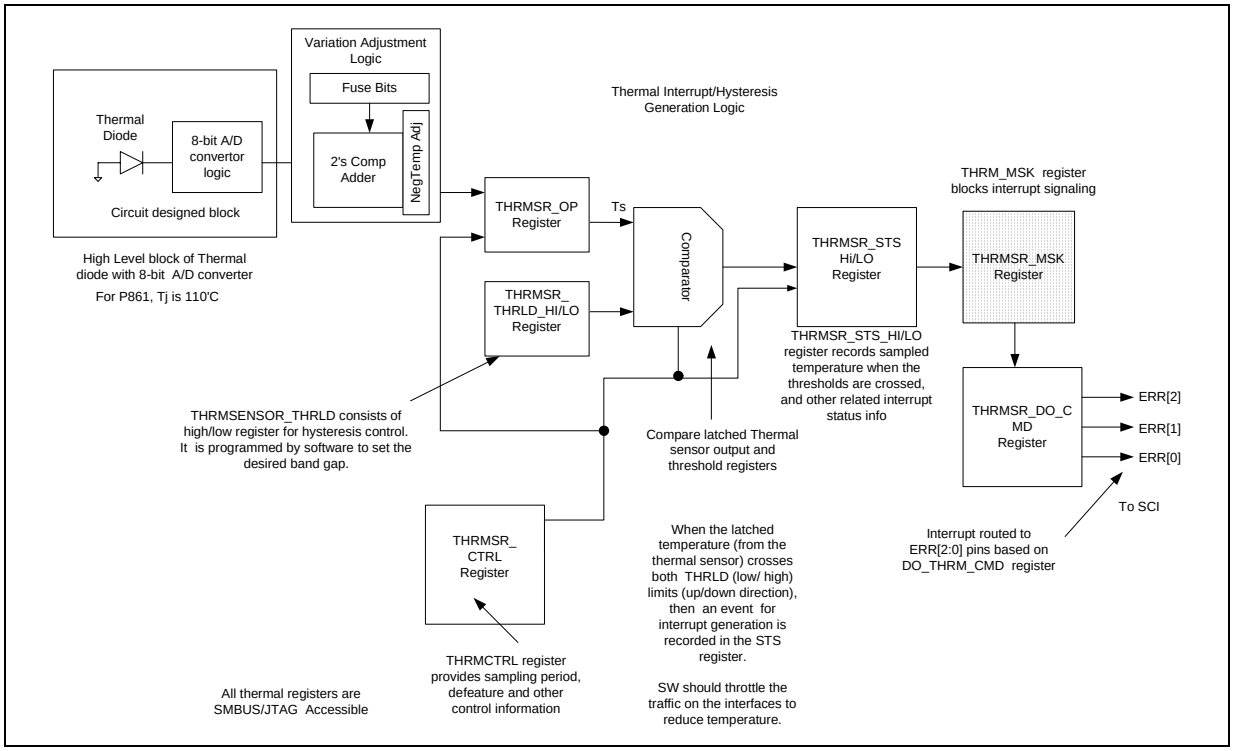
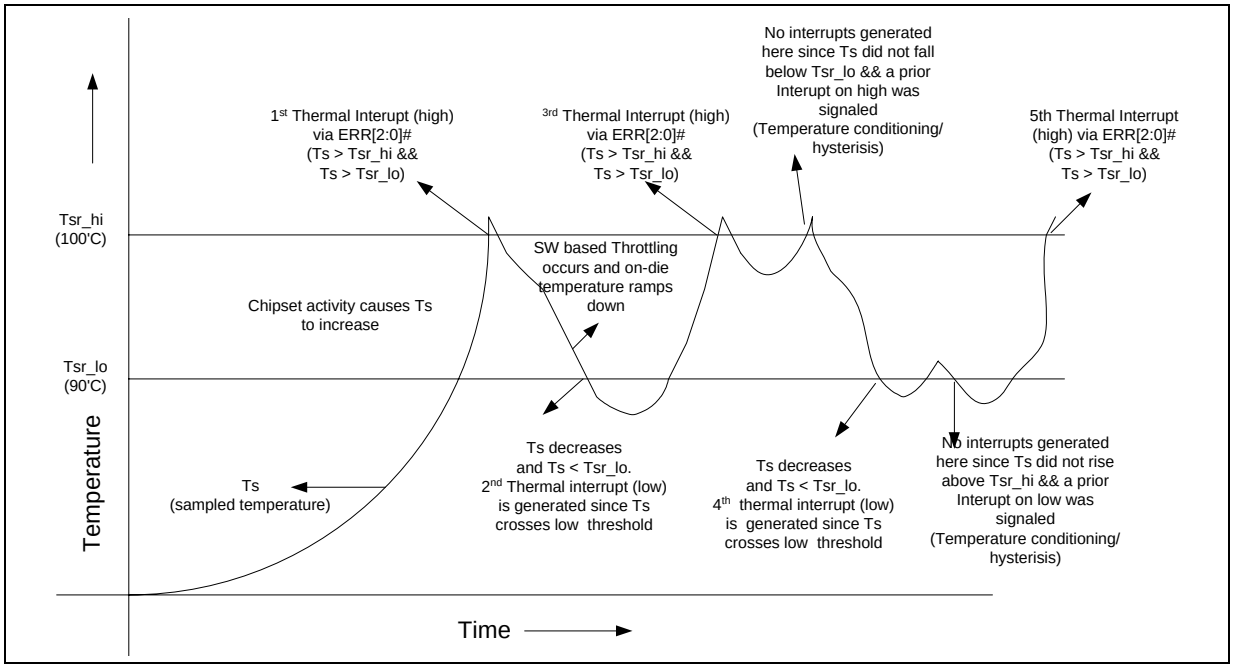


Figure 5-15. Thermal Sensor Interrupt Generation and Conditioning Example



5.5 Behavior on Overtemp State in AMB

Overtemperature occurring in an AMB may lead to data corruption in the .

- If EI is received by due to Overtemp detection in one of the AMBs, will capture random data that most likely will be interpreted as having a CRC or uncorrectable ECC error causing the link to go into a fast reset loop without data corruption.

- If the EI is interpreted as having both good CRC and good ECC, this could cause data corruption until a bad CRC/ECC frame is detected and the link enters the fast reset loop.

Note: An all 0 frame fits this case of good CRC and ECC. This is just as unlikely as any other random frame contents when interpreting EI.

5.6 Interrupts

The Intel 5000X Chipset supports both the XAPIC and traditional 8259 methods of interrupt delivery. I/O interrupts and inter processor interrupts (IPIs) appear as write or interrupt transactions in the system and are delivered to the target processor via the processor bus. This chipset does not support the three-wire sideband bus (the APIC bus) that is used by Pentium® and Pentium® Pro processors.

XAPIC interrupts that are generated from I/O will need to go through an I/O(x)APIC device unless they support Message Signalled Interrupts (MSI). In this document, I/O(x)APIC is an interrupt controller that is found in the Intel® 631xESB/632xESB I/O Controller Hub component of the chipset.

The legacy 8259 functionality is embedded in the Intel® 631xESB/632xESB I/O Controller Hub component. The Intel 5000X Chipset will support inband 8259 interrupt messages from PCI Express devices for boot. The chipset also supports the processor generated “interrupt acknowledge” (for legacy 8259 interrupts), and “end-of-interrupt” transactions (XAPIC).

Routing and delivery of interrupt messages and special transactions are described in this section.

5.7 XAPIC Interrupt Message Delivery

The XAPIC interrupt architectures deliver interrupts to the target processor core via interrupt messages presented on the front side bus. This section describes how messages are routed and delivered in a Intel 5000X Chipset system, this description includes interrupt redirection.

Interrupts can originate from I/O(x)APIC devices or processors in the system. Interrupts generated by I/O(x)APIC devices occur in the form of writes with a specific address encoding. Interrupts generated by the processor appear on the processor bus as transactions with a similar address encoding, and a specific encoding on the REQa/REQb signals (REQa=01001, REQb=11100).

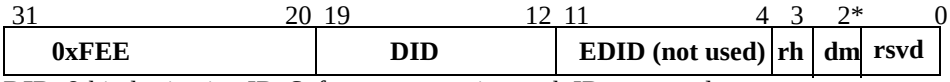
The XAPIC architecture provides for lowest priority delivery, through interrupt redirection by the chipset. If the redirectable hint bit is set in the XAPIC message, the chipset may redirect the interrupt to another processor. Note that redirection of interrupts can be to any processor on either Processor Bus ID and can be applied to

both I/O interrupts and IPIs. The redirection can be performed in logical and physical destination modes. For more details on the interrupt redirection algorithm, see Section 5.7.3.

5.7.1 XAPIC Interrupt Message Format

Interrupt messages have an address of 0x000_FEEz_zzzY. The 16-bit “zzzz” field (destination field) determines the target to which the interrupt is being sent. The Y field is mapped to A3 (redirectable interrupt) and A2 (destination mode). Figure 5-18 shows the address definition in IA32 systems (XAPIC). For each interrupt there is only one data transfer. The data associated with the interrupt message specifies the interrupt vector, destination mode, delivery status, and trigger mode. The transaction type on the processor bus is a request type of, interrupt transaction. The transaction type on the PCI Express and ESI buses is a write. The address definition of Figure 5-18 applies to both the PCI Express bus and processor bus. Note that the current assumption is that we can’t make any conclusions about which FSB an interrupt ID is associated with. At power-up, there is an association for certain types of interrupts, but the current assumption is that the OS can reprogram the interrupt ID’s. Therefore, for directed interrupts, the Intel 5000X Chipset MCH will ensure that each interrupt is seen on both FSBs.

Figure 5-16. XAPIC Address Encoding



DID: 8-bit destination ID. Software may assign each ID to any value.
EDID: Not used, is a reserved field in the Processor EHS.
rh: redirection bit (0=directed, 1=redirectable)
dm: destination mode (0=physical, 1=logical)
*****: PCI/PCI Express transaction encoding. Copied to Ab5 on processor bus

The data fields of an interrupt transaction are defined by the processor and XAPIC specifications. It is included here for reference.

Table 5-12. XAPIC Data Encoding

D[63:16]	D[15]	D[14]	D[13:11]	D[10:8]	D[7:0]
x	Trigger Mode	Delivery Status	x	Delivery Mode	Vector

5.7.2 XAPIC Destination Modes

The destination mode refers to how the processor interprets the destination field of the interrupt message. There are two types of destination modes; physical destination mode, and logical destination mode. The destination mode is selected by A[2] in PCI Express and Ab[5] on the processor bus.

5.7.2.1 Physical Destination Mode (XAPIC)

In physical mode, the APIC ID is 8 bits, supporting up to 255 agents. Each processor has a Local APIC ID Register where the lower 5 bits are initialized by hardware (Cluster ID=ID[4:3], Bus Agent ID=ID[2:1], thread ID=ID[0]). The upper 3 bits default to 0’s at system reset. These values can be modified by software. The Cluster ID is set by

address bits A[12:11] during reset. By default, the Intel 5000X Chipset will drive A[12:11] to ‘00 for FSB0, and ‘01 for FSB1. The value driven on bit A[12] during reset can be modified through the POC register on the Intel 5000X Chipset MCH.

The Intel 5000X Chipset will not rely on the cluster ID or any other fields in the APIC ID to route interrupts. The Intel 5000X Chipset will ensure the interrupt is seen on both busses and the processor with the matching APIC ID will claim the interrupt.

Physical destination mode interrupts can be directed, broadcast, or redirected. An XAPIC message with a destination field of all 1’s denotes a broadcast to all.

In a directed physical mode message the agent claims the interrupt if the upper 8 bits of the destination field (DID field) matches the Local APIC ID of the processor or the interrupt is a broadcast interrupt.

Redirected interrupts are redirected and converted to a directed interrupt by the chipset as described in Section 5.7.3.2.

5.7.2.2 Logical Destination Mode (XAPIC)

In logical destination mode, destinations are specified using an 8 bit logical ID field. Each processor contains a register called the Logical Destination Register (LDR) that holds this 8-bit logical ID. Interpretation of the LDR is determined by the contents of the processor’s Destination Format Register (DFR). Processors used with the Intel 5000X Chipset MCH operate in flat mode. Logical destination mode interrupts can be directed (fixed delivery), redirectable (lowest priority delivery), or broadcast. The LDR is initialized to flat mode (0) at reset and is programmed by firmware. The Intel 5000X Chipset also has an equivalent bit in the External Task Priority Register (XTPR0) to indicate flat or cluster mode. This is set to flat mode by reset and must not be changed, since the processors used with Intel 5000X Chipset operate in flat mode only.

The 8-bit logical ID is compared to the 8-bit destination field of the incoming interrupt message. If there is a bit-wise match, then the local XAPIC is selected as a destination of the interrupt. Each bit position in the destination field corresponds to an individual Local XAPIC Unit. The flat model supports up to 8 agents in the system. An XAPIC message where the DID (destination field) is all 1’s is a broadcast interrupt.

5.7.2.3 XAPIC Interrupt Routing

Interrupt messages that originate from I/O(x)APIC devices or from processing nodes must be routed and delivered to the target agents in the system. In general XAPIC messages are delivered to both processor busses because there is no reliable way to determine the destination processor of the message from the destination field. Interrupts originating from I/O can be generated from a PCI agent using MSI interrupts, or by an interrupt controller on a bridge chip such as the Intel® 631xESB/ 632xESB I/O Controller Hub. Table 5-13 shows the routing rules used for routing XAPIC messages in a Intel 5000X Chipset platform. This table is valid for both broadcast and non-broadcast interrupts.

Table 5-13. Intel 5000X Chipset XAPIC Interrupt Message Routing and Delivery

Source	Type	Routing
I/O	physical or logical directed	Deliver to all processor busses as an interrupt transaction.
Processor	physical or logical directed	Deliver to other processor bus as an interrupt transaction.
Any Source	logical, redirectable physical, redirectable	Redirection (see “Interrupt Redirection” on page 352) is performed by the Intel 5000X Chipset MCH and is delivered to both FSBs.

5.7.3 Interrupt Redirection

The XAPIC architecture provides for lowest priority delivery through interrupt redirection by the Intel 5000X Chipset. If the redirectable “hint bit” is set in the XAPIC message, the chipset may redirect the interrupt to another agent. Redirection of interrupts can be applied to both I/O interrupts and IPIs.

5.7.3.1 XTPR Registers

To accomplish redirection, the Intel 5000X Chipset MCH implements a set of External Task Priority registers (XTPRs), one for each logical processor (a thread is considered a logical processor). Each register contains the following fields:

1. Agent priority (Task Priority)
2. APIC enable bit (TPR Enable)
3. Logical APIC ID (LOGID)
4. Processor physical APIC ID (PHYSID)

The XTPR registers are modified by a front side bus xTPR_Update transaction. In addition, the XTPR registers can be modified by software.

In addition, XTPR0 also contains a bit for Global Cluster Mode bit used in redirection of logical destination mode messages. This bit indicates to the Intel 5000X Chipset MCH that destination field of the message is “flat” or physical (note that the XAPIC message indicates whether the destination mode is physical or logical). The default logical mode at reset is “flat” and must not be changed to “cluster” mode. Cluster mode is not supported by the Intel 5000X Chipset MCH.

More details on the Intel 5000X Chipset XTPR registers are described in the XTPR register definition in Section 3.8.6.3

The XTPR special cycle must guarantee that the XTPR register is updated for interrupt redirection in a consistent manner. For reproducibility, there needs to be an internal serialization point after which subsequent interrupts will be redirected based on the updated XTPR value.

5.7.3.2 Redirection Algorithm

Redirection is performed if an interrupt redirection hint bit (A[3]) is set. This is the algorithm used in determining the processor that the interrupt will be redirected to.

1. If A[3] =1, then this is a redirection (also known as “lowest priority”) interrupt request. Proceed to the next step.
2. FLAT: If Destination Mode = 1 (A2 for I/O, Ab5 for IPIs) is disabled (0) in the XTPR, then this is Flat-Logical Destination Mode. (Otherwise, proceed to the next step). To

select the arbitration pool, for each XTPR register: Note: Cluster Mode is not supported and should always be disabled.

If (A[19:12] (DID) AND XTPR[n].LOGID[7:0]) > 0h
AND XTPR[n].TPREN =1
then XTPR[n] is included in the arbitration pool.

3. PHYSICAL: If Destination Mode = 0 (A2 for I/O, Ab5 for IPIs), then this is Physical Destination Mode. All enabled xTPR's are included in the arbitration pool.
4. If there are no xTPR's in the arbitration pool, then forward to FSB with A[3]=0, but otherwise "without modification". Otherwise, continue to the next step.
5. XTPRs in the pool are categorized into 4 priority buckets depending on the priority. The priority bucket levels are defined by register bits BUCKET(0-2)_LIM in the REDIRCTL register.

   If (0 <= XTPR.PRIORITY < BUCKET0_LIM) then priority bucket = 0

   If (BUCKET0_LIM <= XTPR.PRIORITY < BUCKET1_LIM) then priority bucket = 1

   If (BUCKET1_LIM <= XTPR.PRIORITY < BUCKET2_LIM) then priority bucket = 2

   If (BUCKET2_LIM <= XTPR.PRIORITY < 16) then priority bucket = 3
6. All xTPR's in the arbitration pool are compared. The xTPR register with the lowest priority bucket value (0=lowest, 3=highest) is the "winner".
7. If more than one xTPR register in the arbitration pool has the same lowest priority bucket value, then LRU arbitration logic will pick an xTPR that was not recently picked.
8. The "winning" xTPR register provides the values to be substituted in the Aa[19:12]# field of the FSB Interrupt Message Transaction driven by the Intel 5000X Chipset. A[19:12]# is replaced by the logical or physical ID, depending on the type of interrupt. The interrupt is driven onto both processor buses with the redirection hint bit disabled (A3).

5.7.3.3 XTPR Update

The Intel 5000X Chipset decodes FSB XTPR_Update transactions for interrupt redirection based on lowest priority bucket. Due to specific implementation and timing issues in Intel 5000X Chipset, when an XTPR_update transaction and a pending interrupt (for example, inbound MSI from PCIExpressport) happens concurrently, the interrupt may be delivered to the prior processor thread (i.ethe CPU before XTPR_update took effect instead of the newly redirected CPU). This is not perceived to be a functional problem since the Intel 5000X Chipset will deliver the interrupt to a valid CPU based on the XTPR register values albeit it may be older. Any subsequent interrupts after the XTPR_update will not be affected and it will be dispatched to the latest CPU as indicated.Although there may be negligible performance impact, the asynchronous event may be deemed as non fatal and XTPR_updates are infrequent that this issue is not a problem with the current implementation

5.7.4 EOI

For XPF platforms using XAPIC, the EOI is a specially encoded processor bus transaction with the interrupt vector attached. Since the EOI is not directed, the Intel 5000X Chipset will broadcast the EOI transaction to all I/O(x)APIC's. The Intel 5000X Chipset MCH.PEXCTRL.DIS_APIC_EOI bit per PCI Express port can be used to determine whether an EOI needs to be sent to a specific PCI Express port. EOI usage is further described in Section 5.7.4.

Note: The Intel 5000X Chipset MCH will translate the EOI on the FSB into an EOI TLP message type on the PCI Express/ESI ports.

5.8 I/O Interrupts

For I/O interrupts from the Intel® 631xESB/632xESB I/O Controller Hub components receive interrupts with either dedicated interrupt pins or with writes to the integrated redirection table. The I/OxAPIC controller integrated within these components turns these interrupts into writes destined for the processor bus with a specific address.

Interrupts triggered from an I/O device can be triggered with either a dedicated interrupt pin or through an inbound write message from the PCI Express bus (MSI). Note that if the interrupt is triggered by a dedicated pin, the I/OxAPIC controller in the I/O bridge (Intel® 6700PXH 64 bit PCI Hub or ICH6 or ESB) turns this into an inbound write. On the processor bus, the interrupt is converted to an interrupt request. Other than a special interrupt encoding, the processor bus interrupt follows the same format as discussed in Section 5.7.1. Therefore, to all components other than the Intel® 6700PXH 64 bit PCI Hub, ICH6 or ESB, and the processor, an interrupt is an inbound write following the format mentioned in Section 5.7.1. Intel 5000X Chipset will not write combine or cache the APIC address space.

I/O(x)APIC's can be configured through two mechanisms. The traditional mechanism is the hard coded FEC0_0000 to FECF_FFFF range is used to communicate with the IOAPIC controllers in the Intel® 6700PXH 64 bit PCI Hub, ICH6 or ESB.

The second method is to use the standard MMIO range to communicate to the Intel® 6700PXH 64 bit PCI Hub. To accomplish this, the Intel® 6700PXH 64 bit PCI Hub.MBAR and/or Intel® 6700PXH 64 bit PCI Hub.XAPIC_BASE_ADDRESS_REG must be programmed within the PCI Express device MMIO region.

5.8.1 Ordering

Handling interrupts as inbound writes has inherent advantages. First, there is no need for the additional APIC bus resulting in extra pins and board routing concerns. Second, with an out-of-band APIC bus, there are ordering concerns. Any interrupt needs to be ordered correctly and all prior inbound writes must get flushed ahead of the interrupt. The *PCI Local Bus Specification*, Revision 2.2 attempts to address this by requiring all interrupt routines to first read the PCI interrupt register. Since PCI read completions are required to push all writes ahead of it, then all writes prior to the interrupt are guaranteed to be flushed. However, this assumes that all drivers perform this read.

5.8.2 Hardware IRQ IOxAPIC Interrupts

Dedicated pin interrupts may be edge or level triggered. They are routed to IRQ pins on IOxAPIC device such as the Intel® 6700PXH 64 bit PCI Hub, or Intel® 631xESB/ 632xESB I/O Controller Hub. The IOxAPIC device will convert the interrupt into either an XAPIC or 8259 interrupt.

For level-triggered interrupts, the I/OxAPIC will generate an interrupt message when any of the interrupt lines coming into it become asserted. The processor will handle the interrupt and eventually write to the initiating device that the interrupt is complete. The device will deassert the interrupt line to the I/OxAPIC. After the interrupt has been serviced, the processor sends an EOI command to inform the I/OxAPIC that the interrupt has been serviced. Since the EOI is not directed, the Intel 5000X Chipset will

broadcast the EOI transaction to all I/O(x)APIC's. If the original I/O(x)APIC sees the interrupt is still asserted, it knows there's another interrupt (shared interrupts) and will send another interrupt message.

For edge-triggered interrupts, the flow is the same except that there is no EOI message indicating that the interrupt is complete. Since the interrupt is issued whenever an edge is detected, EOIs are not necessary.

While not recommended, agents can share interrupts to better utilize each interrupt (implying level-triggered interrupts). Due to ordering constraints, agents can not use an interrupt controller that resides on a different PCI bus. Therefore either only agents on the same PCI bus can share interrupts, or the driver MUST follow the PCI requirement that interrupt routines must first read the PCI interrupt register

The Intel 5000X Chipset MCH supports the INTA (interrupt acknowledge) special bus cycle for legacy 8259 support. These are routed to the compatibility ICH6 or ESB in the system. The INTA will return data that provides the interrupt vector.

5.8.3 Message Signalled Interrupts

A second mechanism for devices to send interrupts is to issue the Message Signalled Interrupt (MSI) introduced in the *PCI Local Bus Specification*, Revision 2.2 . This appears as a 1 DWORD write on the PCI/PCI-X/PCI Express bus.

With PCI devices, there are two types of MSIs. One type is where a PCI device issues the inbound write to the interrupt range. The other type of MSI is where a PCI device issues an inbound write to the upstream APIC controller (for example, in the Intel® 6700PXH 64 bit PCI Hub) where the APIC controller converts it into an inbound write to the interrupt range. The second type of MSI can be used in the event the OS doesn't support MSIs, but the BIOS does. In either way, the interrupt will appear as an inbound write to the Intel 5000X Chipset over the PCI Express ports.

MSI is expected to be supported by the operating systems when the Intel 5000X Chipset MCH is available. An Intel 5000X Chipset platform will also feature a backup interrupt mechanism in the event that there is a short period of time when MSI is not available. This is described in the next section.

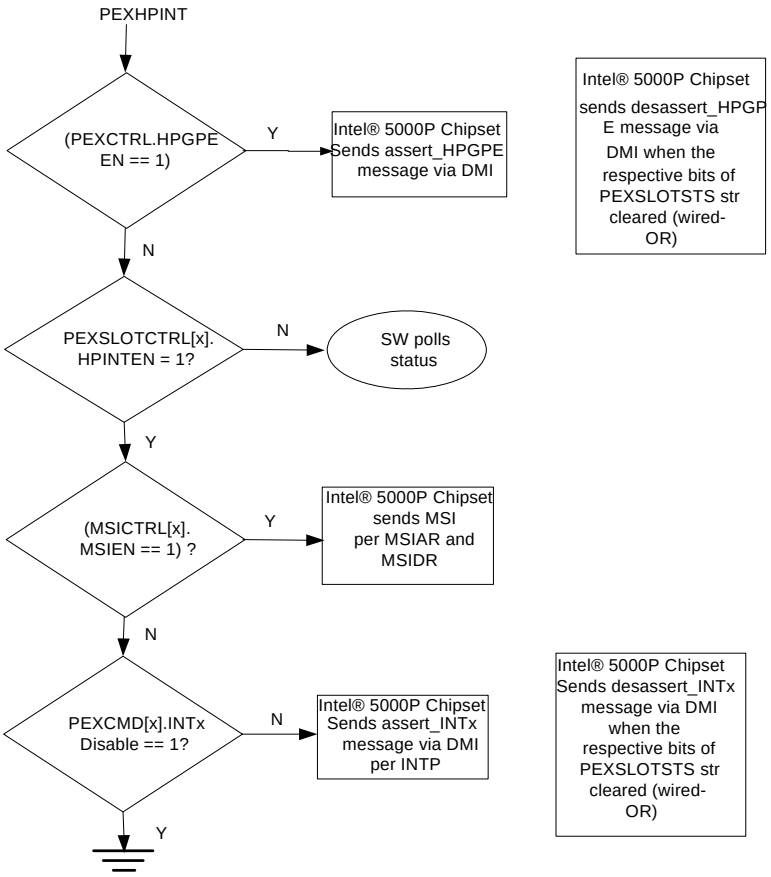
5.8.4 Non-MSI Interrupts - "Fake MSI"

For interrupts coming through the Intel® 6700PXH 64 bit PCI Hub, and Intel® 631xESB/632xESB I/O Controller Hub components, their APIC controller will convert interrupts into inbound writes, so inbound interrupts will appear in the same format as an MSI.

For interrupts that are not coming through an APIC controller, it is still required that the interrupt appear as an MSI-like interrupt. If the OS does not yet support MSI, the PCI Express device can be programmed by the BIOS to issue inbound MSI interrupts to an IOxAPIC in the system. The safest IOxAPIC to choose would be the Intel® 631xESB/632xESB I/O Controller Hub since it is always present in a system. Although the Intel 5000X Chipset supports the PCI Express "Assert_Int" and "Deassert_Int" packets for boot, the performance is not optimal and is not recommended for run time interrupts.

In this method, PCI Express devices are programmed to enable MSI functionality, and given a write path directly to the pin assertion register in a selected IOxAPIC already present in the platform. The IOxAPIC will generate an interrupt message in response,

thus providing equivalent functionality to a virtual (edge-triggered) wire between the



HPGPEEN	HPINTEN	MSIEN	INTx Disable	Output
1	x	x	x	assert_hpgpe
0	1	1	x	MSI
0	1	0	0	assert_intx
0	1	0	1	--
0	0	x	x	--

PCI Express endpoint and the I/OxAPIC. This mechanism is the same as is used in Longhorn* (XYZZY).

All PCI Express devices are strictly required to support MSI. When MSI is enabled, PCI Express devices generate a memory transaction with an address equal to the I/OxAPIC_MEM_BAR + 20 and a 32-bit data equal to the interrupt vector number corresponding to the device. This information is stored in the device's MSI address and data registers, and would be initialized by the system firmware (BIOS) prior to booting a non-MSI aware operating system. (With the theory that an MSI aware O/S would then over-write the registers to provide interrupt message delivery directly from the endpoint to the CPU complex.)

The PCI Express memory write transaction propagates to the Intel 5000X Chipset and is redirected down the appropriate PCI Express port following the Intel 5000X Chipset IOAPIC address mapping definition. The IOAPIC memory space ranges are fixed and

cannot be relocated by the OS. The assert message is indistinguishable from a memory write transaction, and is forwarded to the destination I/OxAPIC, which will then create an upstream APIC interrupt message in the form of an inbound memory write. The write nature of the message “pushes” all applicable pre-interrupt traffic through to the Intel 5000X Chipset core, and the Intel 5000X Chipset core architecture guarantees that the subsequent APIC message cannot pass any posted data already within the Intel 5000X Chipset.

5.9 Interprocessor Interrupts (IPIs)

- Previous IA-32 processors use IPIs after reset to select the boot strap processor (BSP). Recent XPF processors do not use IPIs to select the BSP. A hardware arbitration mechanism is used instead.
- IA32 processors use Startup IPIs (SIPIs) to wake up sleeping application processors (non boot strap processors) that are in “Wait for SIPI state”. These are broadcast interrupts.
- Interrupts transactions are claimed with TRDY# and No-Data Response.
- For directed XAPIC (A[3] = 0) interrupts, the Intel 5000X Chipset completes the interrupt normally and forwards the interrupt to the other bus.
- For redirectable XAPIC interrupts, the Intel 5000X Chipset will generate an interrupt message to both processor buses Intel 5000X Chipset with A[3] (redirectable hint bit) set to 0. This message will contain a processor ID based on the redirection algorithm.
- For directed XAPIC broadcast interrupts (Destination ID = 0xFF), the Intel 5000X Chipset will forward the broadcast interrupt to the other processor bus.
- Interrupts are not deferred.
- Since XAPIC directed interrupts (A[3] = 0) cannot be retried, they must be accepted. If the Intel 5000X Chipset cannot accept the interrupt, then it must assert BPRI# until resources are available.

5.9.0.1 IPI Ordering

In a system, there are ordering requirements between IPIs and other previous coherent and non-coherent accesses. The way the ordering is maintained is that it is expected that the chipset will defer the previous ordered access. The chipset will not complete the transaction until the write is “posted” or the read data is delivered. Since the processor will not issue an ordered IPI until the previous transaction has been completed, ordering is automatically maintained.

An example where the ordering must be maintained is if a processor writes data to memory and issues an IPI to indicate the data has been written, subsequent reads to the data (after the IPI) must be the updated values. (Producer consumer). For this example, assuming cacheable memory, the chipset defers the BIL/BRIL (read for ownership). Only after all other processor caches have been invalidated, and the deferred reply is returned (where the cache will be written) will the subsequent IPI be issued.

There are no ordering requirements between IPIs. There are no ordering requirements between IPIs and subsequent request. The IPIs are claimed on the FSB (front side bus) and are not deferred. Therefore, software must not rely on the ordered delivery between the IPI and subsequent transactions. If ordering is needed, it must protect any

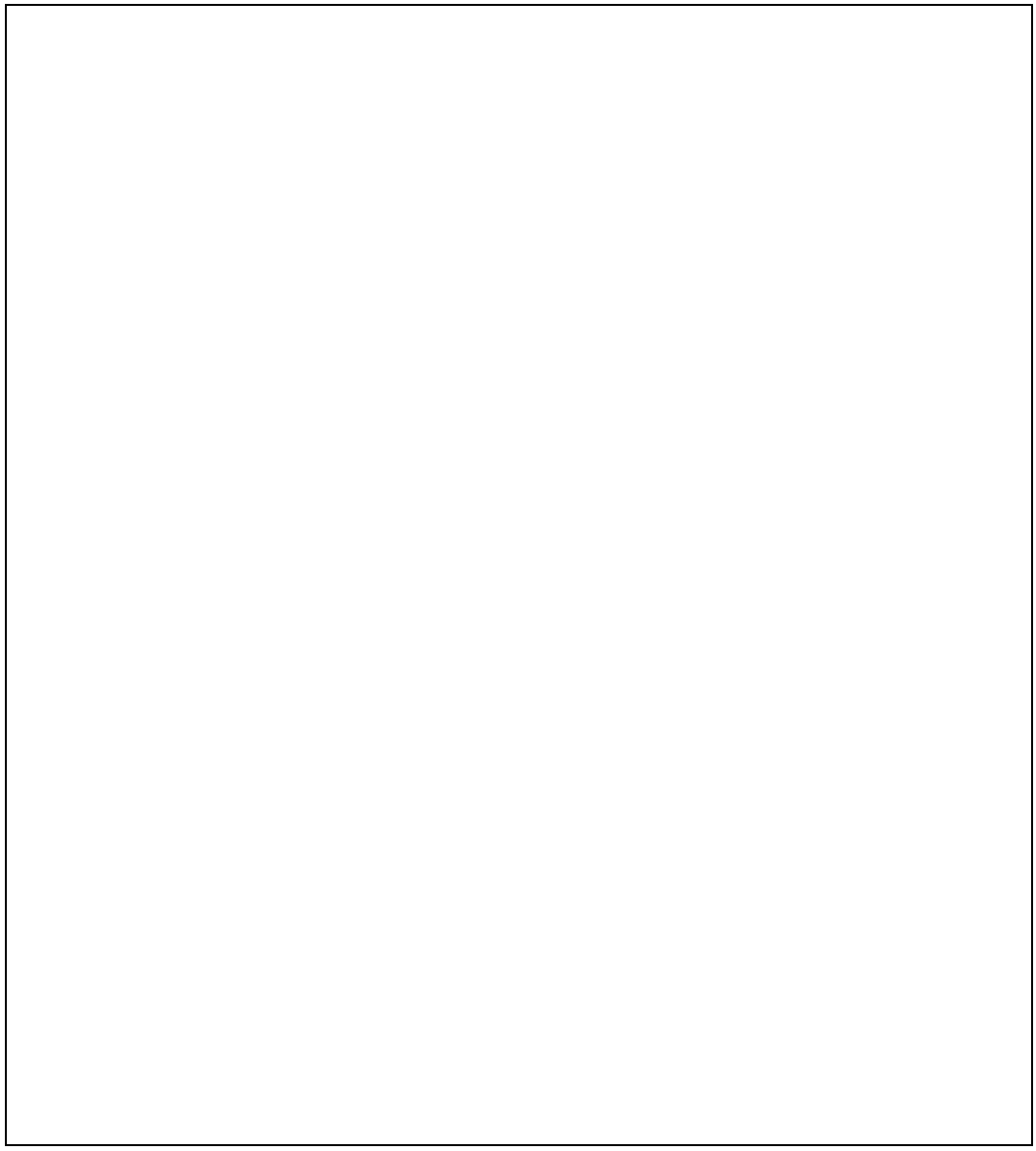
subsequent coherent and non-coherent accesses from the effects of a previous IPI using synchronization primitives. Also, software must not rely on ordered delivery of an IPI with respect to other IPI from the same processor to any target processor.

5.10 Chipset Generated Interrupts

The Intel 5000X Chipset MCH can trigger interrupts for chipset errors and for PCI Express. For these events, the chipset can be programmed to assert pins that the system can route to an APIC controller. The interrupts generated by the chipset are still being defined. The following is a preliminary list of interrupts that can be generated.

1. Chipset error - Intel 5000X Chipset MCH asserts appropriate ERR pin, depending on severity. This can be routed by the system to generate an interrupt at an interrupt controller. (Intel 5000X Chipset MCH pins ERR[2:0], MCERR, Intel® 631xESB/ 632xESB I/O Controller Hub Reset). The ERR[0] pin denotes a correctable and recoverable error. The ERR[1] pin denotes an uncorrectable error from Intel 5000X Chipset MCH. The ERR[2] pin denotes a fatal error output from Intel 5000X Chipset MCH.
2. PCI Express error - Intel 5000X Chipset MCH asserts appropriate ERR pin, depending on severity. This can be routed by the system to generate an interrupt.
   a. The Intel 5000X Chipset MCH can receive error indications from the PCI Express ports. These are in the form of inbound ERR_COR/UNC/FATAL messages. Intel 5000X Chipset MCH will assert the appropriate ERR signal just like any internal Intel 5000X Chipset MCH error as described in the RAS chapter.
3. PCI Express Hotplug - Intel 5000X Chipset MCH send Assert_HPGPE (Deassert_HPGPE) or generates an MSI or a legacy interrupt on behalf of a PCI Express Hot-Plug event.
   a. Intel 5000X Chipset MCH generated Hot-Plug event such as PresDet change, Attn button, MRL sensor changed, power fault, and so forth. Each of these events have a corresponding bit in the PCI Express Hot-Plug registers (Attention Button, Power Indicator, Power Controller, Presence Detect, MRL Sensor, Port Capabilities/Slot registers). This will generate an interrupt via the assert_HPGPE, intx, or an MSI. Refer to Figure 5-17 for the Hotplug interrupt flow priority.
   b. PCI Express Hot-Plug event from downstream. This could be either an MSI or a GPE message.
   — MSI: Handled like a normal MSI interrupt (see Figure 5.8.3)
   — GPE message: Upon receipt of a Assert_GPE message from PCI Express, Intel 5000X Chipset MCH will send assert_GPE signal to the ESI port. To generate an SCI (ACPI), this signal will be routed to the Intel® 631xESB/632xESB I/O Controller Hub appropriate GPIO pin to match the GPE0_EN register settings. When the Hot-Plug event has been serviced, Intel 5000X Chipset MCH will receive a Deassert_GPE message. At this point the Intel 5000X Chipset MCH can deassert_GPE message to ESI. There needs to be a tracking bit per PCI Express port to keep track of Assert/Deassert_GPE pairs. These tracking bits should be OR'd together to determine whether to send the assert_GPE/ Deassert_GPE message. When Intel 5000X Chipset MCH receives a matching deassert_GPE message for that port, it will clear the corresponding tracking bit. When all the tracking bits are cleared, the Intel 5000X Chipset MCH will send a Deassert_GPE message to the ESI port.
   — Sideband signals: Some systems may choose to connect the interrupt via sideband signals directly to the Intel® 631xESB/632xESB I/O Controller Hub. No action is required from the Intel 5000X Chipset MCH.

Figure 5-17. PCI Express Hot-Plug Interrupt Flow



4. PCI Hot-Plug - Chipset will receive an Assert/Deassert GPE message from the PCI Express port when a PCI Hot-Plug event is happening. Assert/Deassert GPE messages should be treated the same as Assert/Deassert GPE messages for PCI Express Hot-Plug. (Keep track of Assert/Deassert GPE messages from each port and send Assert_GPE, Deassert_GPE message to ESI appropriately)
5. PCI Express Power management - PCI Express sends a PME message. Chipset sends Assert_PMEGPE to ESI port when a power management event is detected.
    a. Upon receipt of the PME message, Intel 5000X Chipset MCH will set the PEXRTSTS.PMESTATUS bit corresponding to that port and send Assert_PMEGPE to ESI port to generate the interrupt. (Assert_PMEGPE should be sent if one or more of the PMESTATUS bits are set and enabled.) To generate an SCI (ACPI), this message will be used by the Intel® 631xESB/632xESB I/O Controller Hub to drive appropriate pin. When software has completed servicing the power management event, it will clear the

PEXRTSTS.PMESTATUS bit (by writing 1), at which point the Intel 5000X Chipset MCH can send Deassert_PMEGPE to ESI port.

The following table summarizes the different types of chipset generated interrupts that were discussed. Although the interrupt and SW mechanism is flexible and can be changed depending on how the system is hooked up, for reference this table also describes what SW mechanism is expected to be used.

Table 5-14. Chipset Generated Interrupts

Source	Signalling mechanism	Intel 5000X Chipset MCH signal method	Expected SW mechanism
Chipset Error	Intel 5000X Chipset MCH registers	ERR[2:0], MCERR, Intel® 631xESB/632xESB I/O Controller Hub Reset	Any
PCI Express Error	PCI Express ERR_COR/UNC/FATAL message	ERR[2:0], MCERR, Intel® 631xESB/632xESB I/O Controller Hub Reset	Any
PCI Express HP (PresDet chg, Attn button, and so forth.)	Intel 5000X Chipset MCH registers For card-these registers are set via the VPP/SM bus interface. For module- these registers are set by inband Hot-Plug messages.	MSI or Assert_intx, Deassert_intx, or Assert_HPGPE, Deassert_HPGPE	SCI->ACPI or MSI
PCI Express HP from downstream device	MSI	MSI interrupt (processor bus)	MSI
PCI Express HP from downstream device (non-native, Intel part)	PCI Express Assert/Deassert GPE	Assert_GPE, Deassert_GPE to ESI	SCI->ACPI
PCI Express HP from downstream device (non-native, non-Intel part)	Sideband signals directly to Intel® 631xESB/632xESB I/O Controller Hub	N/A	SCI->ACPI
Downstream PCI Hot-Plug	PCI Express Assert/Deassert GPE	Assert_GPE, Deassert_GPE to ESI	SCI->ACPI
Power Management Event (PME)	PCI Express PM_PME message	Assert_PMEGPE, Deassert_PMEGPE to ESI	SCI->ACPI

5.10.1 Intel 5000X Chipset Generation of MSIs

The Intel 5000X Chipset MCH generates MSIs on behalf of PCI Express Hot-Plug events if Intel 5000X Chipset MCH.MSICTRL.MSIEN is set. Refer to Figure 5-17. The Intel 5000X Chipset MCH will interpret PCI Express Hot-Plug events and generate an MSI interrupt based on Intel 5000X Chipset MCH.MSIAR and Intel 5000X Chipset MCH.MSIDR registers. When the Intel 5000X Chipset MCH detects any PCI Express Hot-Plug event, it will generate an interrupt transaction to both processor buses. The address will be the value in Intel 5000X Chipset MCH.MSIAR. The data value will be the value in MSIDR.

Internal to the Intel 5000X Chipset MCH, the MSI can be considered an inbound write to address MSIAR with data value of MSIDR, and can be handled the same as other inbound writes that are MSIs or APIC interrupts.

5.10.1.1 MSI Ordering in Intel 5000X Chipset MCH

Ordering issues on internally generated MSIs could manifest in the Intel 5000X Chipset MCH if software/device drivers rely on certain usage models for example, interrupt rebalancing, Hot-Plug to flush them. The producer-consumer violation may happen, if a root port has posted an MSI write internally in the MCH and the software wants to

"flush" all MSI writes from the root port that is, guarantee that all the MSI writes pending in the MCH from the root port have been delivered to the local APIC in the processor. To accomplish this flush operation, OS can perform a configuration read to, say, the VendorID/DeviceID register of the root port and the expectation is that the completion for this read will flush all the previously issued memory writes. The reason the OS wants to flush is for cases where an interrupt source (like a root port) is being retargeted to a different processor and OS needs to flush any MSI that is already pending in the fabric that is still targeting the old processor.

As a case in point, reads to Intel 5000X Chipset MCH PCI Express (internal) configuration spaces will not generally guarantee ordering of internal MSIs from a root port/DMA Engine device as required since the Intel 5000X Chipset MCH uses a configuration ring methodology which houses the registers for the various PCI Express ports, MC, DMA Engine, Dfx and so forth) and it operates independently of the MSI/ interrupt generation logic. Thus any configuration ring access targeting a PCI Express port registers will not necessarily order and align with the internal MSIs.

Solution: To mitigate this problem and enforce ordering of the MSIs, the Intel 5000X Chipset MCH will implement a "pending MSI signal" that is broadcast from the MSI/ Hotplug blocks to the coherency engine and thereby block the configuration request (non-posted) till all the MSI gets committed. Software will ensure that it will block future MSI generation for that device when it issues the configuration read for that device.

The CE will block sending any completion with the new bit-slice bit set when any of the pending MSI wires is asserted. CE will not block other transactions or completions during the block. When the pending MSI wires are deasserted, CE will be able to send the configuration completions.

The Intel 5000X Chipset MCH Coherency Engine (CE) will block processor initiated MCH configuration access completions (MMCFG or CFC/CF8) if there is a pending internally generated MSI within the Intel 5000X Chipset MCH. (MSIs could be generated from the DMA engine or the HotPlug-Pwr-Mgr-PEX Error block.

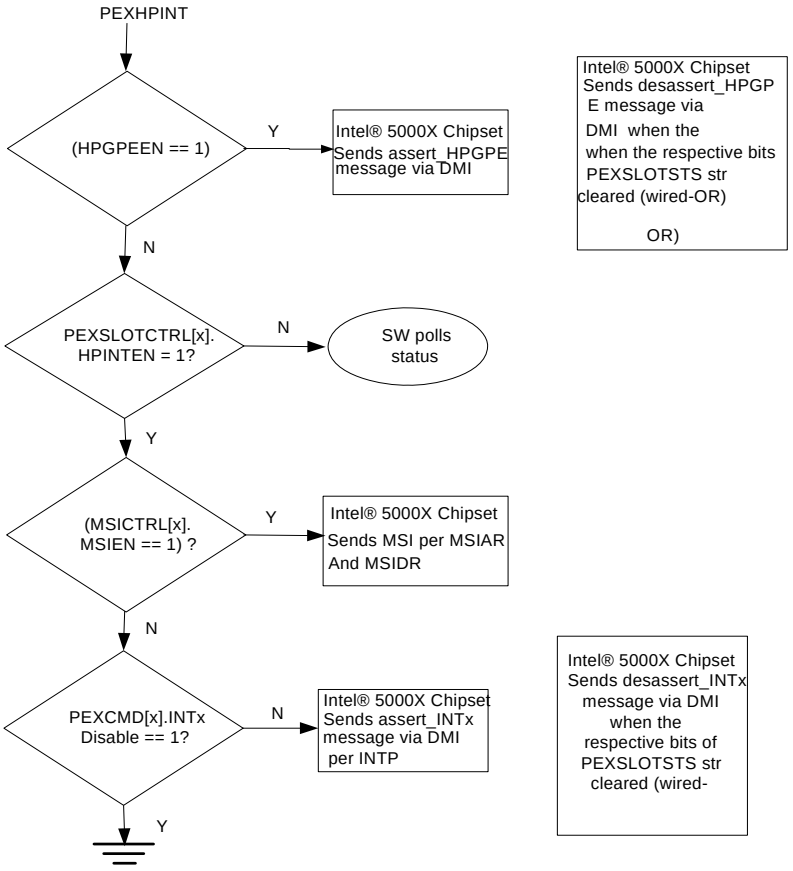
The pending MSI signal will be deasserted after fetch-completion is asserted for the MSI from CE, that is, global visibility is guaranteed on the FSB. Then release the configuration block and allow the configuration completion to flow through. This approach will order the MSI and then send the non-posted configuration for that device.

CE will add a bit-slice (one bit per table entry) to track processor initiated MCH configuration access in CE transaction table. Note: Inbound configuration access will not set this bit.

A defeature mode to control the MSI/NP_CFG ordering is defined in the COHDEF.DIS_MSI_NPCFG register field.

Note: Internal MSIs cannot be continuously generated since the corresponding status register field needs to be cleared by software through configuration access before a new MSI can be asserted.

5.11 Legacy/8259 Interrupts



HPGPEEN	HPINTEN	MSIEN	INTx Disable	Output
1	x	x	x	assert_hpgpe
0	1	1	x	MSI
0	1	0	0	assert_intx
0	1	0	1	--
0	0	x	x	--

8259 interrupt controller is supported in Intel 5000X Chipset platforms. 8259 interrupt request is delivered using the interrupt group sideband signals LINT[1:0] (a.k.a. NMI/ INTR) or through an I/O xAPIC using the message based interrupt delivery mechanism with the delivery mode set to ExtINT (111b). There can be only one active 8259 controller in the system.

The mechanism in which a PCI Express device requests an 8259 interrupt is a PCI Express inband message. (ASSERT_INTA/B/C/D, DEASSERT_INTA/B/C/D).

The target processor for the interrupt uses the interrupt acknowledge transaction to obtain the interrupt vector from the 8259 controller. The Intel 5000X Chipset forwards the interrupt acknowledge to the Intel® 631xESB/632xESB I/O Controller Hub where the active 8259 controller resides.

The Intel 5000X Chipset will support PCI Express devices that generate 8259 interrupts (for example, during boot). 8259 interrupts from PCI Express devices will be sent in-band to the Intel 5000X Chipset which will forward these interrupts to the Intel® 631xESB/632xESB I/O Controller Hub.

The Intel 5000X Chipset will have a mechanism to track inband 8259 interrupts from each PCI Express and assert virtual interrupt signals to the 8259 through the inband "Assert_(Deassert)_Intx" messages. This is done by a tracking bit per interrupt (A, B, C, D) in each PCI Express which are combined (OR'd) into virtual signals that are sent to the Intel® 631xESB/632xESB I/O Controller Hub. Each interrupt signal (A, B, C, D) from each PCI Express is OR'ed together to form virtual INT A, B, C, and D signals to the Intel® 631xESB/632xESB I/O Controller Hub (Assert_(Deassert)_IntA/B/C/D (assertion encoding)). When all of the tracking bits for a given interrupt (A, B, C, or D) are cleared from all PCI Express ports, the virtual signal A, B, C, or D is deasserted via the inband Deassert_Intx message.

For PCI Express hierarchies, interrupts will be consolidated at each level. For example, a PCI Express switch connected to a Intel 5000X Chipset PCI Express port will only send a maximum of 4 interrupts at a time, regardless of how many interrupts are issued downstream.

SMI (System Management Interrupt) interrupts are initiated by the SMI# signal in the platform. On accepting a System Management Interrupt, the processor saves the current state and enters SMM mode.

Note that the Intel 5000X Chipset core components do not interact with the LINT[1:0] and SMI signals. They are present on the Intel® 631xESB/632xESB I/O Controller Hub and the processor. Intel 5000X Chipset interrupt signals described in Section 5.10 can be routed to the Intel® 631xESB/632xESB I/O Controller Hub to generate an SMI interrupt. Similarly SCI interrupts can be generated by routing Intel 5000X Chipset interrupt signals to the appropriate Intel® 631xESB/632xESB I/O Controller Hub pin.

5.12 Interrupt Error Handling

Software must configure the system so that each interrupt has a valid recipient. In the event that an interrupt doesn't have a valid recipient, since the Intel 5000X Chipset will not necessarily know that the interrupt is targeted for a non-existing processor, will deliver the interrupt to the processor buses following the interrupt routing rules described in this chapter. If the interrupt targets a non-existing processor, it may be ignored but the transaction should still complete.

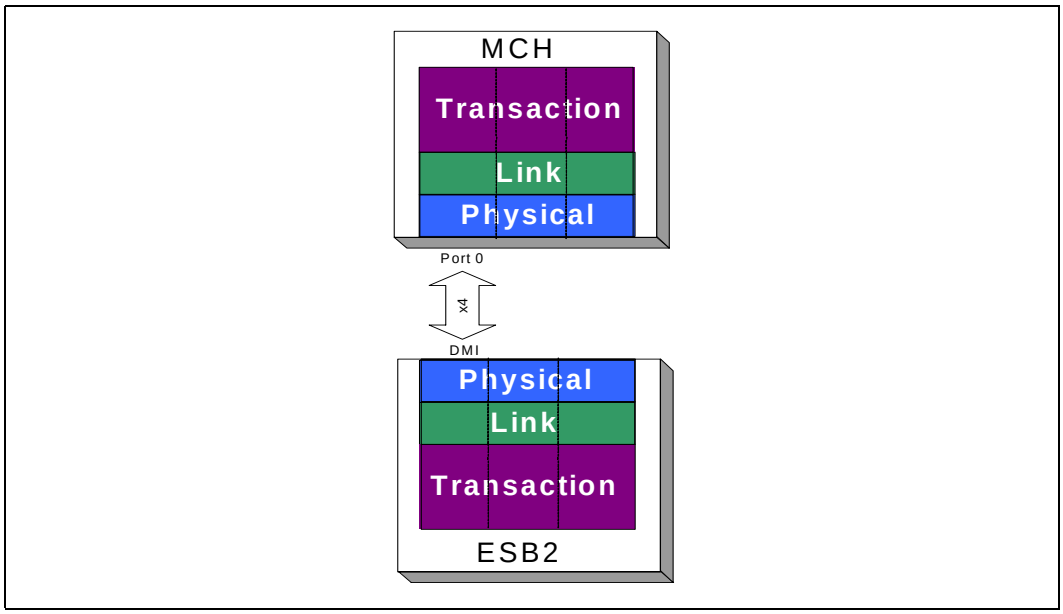
Any error in the data part of an interrupt message, interrupt acknowledge, or EOI will be treated the same way as data error with any other transaction – single bit errors will be corrected by ECC, double bit error will be treated and logged as uncorrectable. For more details on error handling, please refer to the RAS chapter.

5.13 Enterprise South Bridge Interface (ESI)

The Enterprise South Bridge Interface (ESI) in the Intel 5000X Chipset north bridge is the chip-to-chip connection to the Intel® 631xESB/632xESB I/O Controller Hub see Figure 5-18. The ESI is an extension of the standard PCI Express specification with special commands/features added to enhance the PCI Express interface for enterprise applications. This high-speed interface integrates advanced priority-based servicing allowing for concurrent traffic and true isochronous transfer capabilities. Base functionality is completely transparent permitting current and legacy software to

operate normally. For the purposes of this document, the Intel® 631xESB/632xESB I/O Controller Hub will be used as a reference point for the ESI discussion in the Intel 5000X Chipset north bridge.

Figure 5-18. MCH to Intel® 631xESB/632xESB I/O Controller Hub (ESB2) Enterprise South Bridge Interface



The ESI port in the Intel 5000X Chipset north bridge may be combined with two additional PCI Express ports to augment the available bandwidth to the Intel® 631xESB/632xESB I/O Controller Hub. When operating alone the available bi-directional bandwidth to the Intel® 631xESB/632xESB I/O Controller Hub is 2GB/s (1GB/s each direction). When the ESI is pared with 2 additional x4 PCI Express links the available bi-directional bandwidth to the Intel® 631xESB/632xESB I/O Controller Hub is increased to 6GB/s. The details of how the ESI port is combined with additional x4 PCI Express ports are covered in Figure 5.14.3

5.13.1 Peer-to-Peer Support

Peer-to-peer support is defined as transactions which initiate on one I/O interface and target another without going through main memory. The MCH ESI supports peer-to-peer transactions for memory and I/O transactions. The compatibility interface can be the destination of a peer-to-peer write or read except that peer-to-peer posted writes targeting LPC in Intel® 631xESB/632xESB I/O Controller Hub are not allowed (to prevent PHOLD deadlocks). The compatibility interface can be the source of a peer-to-peer read/write. Non-posted requests may prefetch into MMIO (with potential side effects). Peer-to-peer transactions are not observed on any interface except the target and destination (for example, no processor bus snoops).

Peer to Peer traffic from ESI to a PCI Express port should attempt to maximize the link bandwidth. Inbound coherent transactions and peer-to-peer transactions must maintain ordering rules between each other. Peer-to-peer transactions follow inbound ordering rules until they reach the head of the inbound queue. Once the transaction reaches the head of the inbound queue, the MCH routes the transaction to the next available slot in the outbound queue where PCI ordering is maintained until the end of

the transaction. The MCH does not support peer-to-peer where the source and destination is the same PCI Express interface. Note that legacy floppy drives which also use PHOLD are supported by Intel 5000X Chipset

5.13.2 Power Management Support

The Intel® 631xESB/632xESB I/O Controller Hub provides a rich set of power management capabilities for the operating system. The MCH receives PM_PME messages on its standard PCI Express port and propagates it to the Intel® 631xESB/632xESB I/O Controller Hub over the ESI as an Assert_PMEGPE message. When software clears the PEXRTSTS.PME Status register bit, in the PEXRSTSTS[7:2, 0] PCI Express Root Status Register, after it has completed the PME protocol, the MCH will generate a Deassert_PMEGPE message to the Intel® 631xESB/632xESB I/O Controller Hub. The MCH must also be able to generate the Assert_PMEGPE message when exiting S3 (after the reset). The PMGPE messages are also sent using a wired-OR approach.

5.13.2.1 Rst_Warn and Rst_Warn_Ack

The Rst_Warn message is generated by the Intel® 631xESB/632xESB I/O Controller Hub as a warning to the MCH that it wants to assert PLTRST# before sending the reset. In the past, problems have been encountered due to the effects of an asynchronous reset on the system memory states. Since memory has no reset mechanism itself other than cycling the power, it can cause problems with the memory's internal states when clocks and control signals are asynchronously tri-stated or toggled, if operations resume following this reset without power cycling. To protect against this, the Intel® 631xESB/632xESB I/O Controller Hub will send a reset warning to the MCH. The Gold Bridge (Advanced Memory Buffer) is supposed to handle putting the DIMMs into a non-lockup state in the event the link "goes down" in the middle of DDR2 protocol. The Intel 5000X Chipset MCH is NOT required to place quiesce the DRAM's prior to reset.

The MCH completes the handshake by generating the Rst_Warn_Ack message to the ICH6 at the earliest.

5.13.2.2 STPCLK Propagation

The Intel® 631xESB/632xESB I/O Controller Hub has a sideband signal called STPCLK. This signal is used to place IA32 CPUs into a low power mode. Traditionally, this signal has been routed directly from the I/O Controller Hub to the CPUs.

In future ESBx components, the plan is to rearchitect the mechanism for alerting the CPUs of a power management event. However, this chipset (using Intel® 631xESB/632xESB I/O Controller Hub) will require the same method used for past server chipsets (route STPCLK on the board as appropriate). The MCH will not provide any in-band mechanisms for STPCLK.

5.13.3 Special Interrupt Support

The Intel® 631xESB/632xESB I/O Controller Hub integrates an I/O APIC controller. This controller is capable of sending interrupts to the processors with an inbound write to a specific address range that the processors recognize as an interrupt. In general, the compatibility interface cluster treats these no differently from inbound writes to DRAM. However, there are a few notable differences listed below.

5.13.4 Inbound Interrupts

To the MCH, interrupts from the Intel® 631xESB/632xESB I/O Controller Hub are simply inbound non-coherent write commands routed to the processor buses. The MCH does not support the serial APIC bus.

5.13.5 Legacy Interrupt Messages

The ESI and PCI Express interfaces support two methods for handling interrupts: MSI and legacy interrupt messages. The interrupt messages are a mechanism for taking traditionally out-of -band interrupt signals and using in-band messages to communicate. Each PCI Express interface accepts up to four interrupts (A through D) and each interrupt has an assert/deassert message to emulate level-triggered behavior. The MCH effectively wire-ORs all the INTA messages together (INTBs are wire-ORed together, and so forth).

When the MCH accepts these PCI Express interrupt messages, it aggregates and passes the corresponding “assert_intx” messages to the Intel® 631xESB/632xESB I/O Controller Hub’s I/OAPIC with from the PCI Express ports (wired-OR output transitions from 0⇒1) mechanism. When the corresponding deassert_intx message is received at all the PCI Express ports (wired-OR output transitions from 1⇒0), the “deassert_intx” message is sent to ESI port.

5.13.6 End-of-Interrupt (EOI) Support

The EOI is a specially encoded processor bus transaction with the interrupt vector attached. Since the EOI is not directed, the MCH will broadcast the EOI transaction to all I/O(x)APICs. The MCH.PEXCTRL.DIS_APIC_EOI bit per PCI Express port can be used to determine whether an EOI needs to be sent to a specific port.

5.13.7 Error Handling

Table 5-38 describes the errors detected on ESI through the standard PCI Express and Advanced error reporting mechanism.

5.13.7.1 Inbound Errors

In general, if an inbound read transaction results in a Master Abort (unsupported request), the compatibility interface cluster returns a Master Abort completion with data as all ones. Likewise, for a Target Abort condition, the ESI cluster returns a Target Abort completion with data as all ones. If a read request results in a Master or Target Abort, the MCH returns the requested number of data phases with all ones data.

Master aborted inbound writes are dropped by the MCH, the error is logged, and the data is dropped.

If the MCH receives an inbound unsupported Special Cycle message it is ignored and the error condition is logged. If the completion required bit is set, an Unsupported Special Cycle completion is returned.

5.13.7.2 Outbound Errors

It is possible that the compatibility interface cluster will receive an error response for an outbound request. This can include a Master or Target Abort for requests that required completions. The MCH might also receive an “Unsupported Special Cycle” completion.

5.14 PCI Express Ports

The Intel 5000X Chipset MCH contains two classes of PCI Express derived ports. These are:

- Enterprise South Bridge Interface (ESI), Port 0
- DMA Engine/General purpose enabled ports, Port 2, Port 3,

Note: There is no PCI Express port designated as Port 1.

The ESI port is the primary interface to the Intel® 631xESB/632xESB I/O Controller Hub. This interface can be paired with up to two of the DMA Engine ports (Port 2 and Port 3) to increase available bandwidth to the Intel® 631xESB/632xESB I/O Controller Hub. The Intel 5000X Chipset MCH supports a high performance x16 graphics PCI Express port. This port contains several architectural enhancements to increase graphics performance.

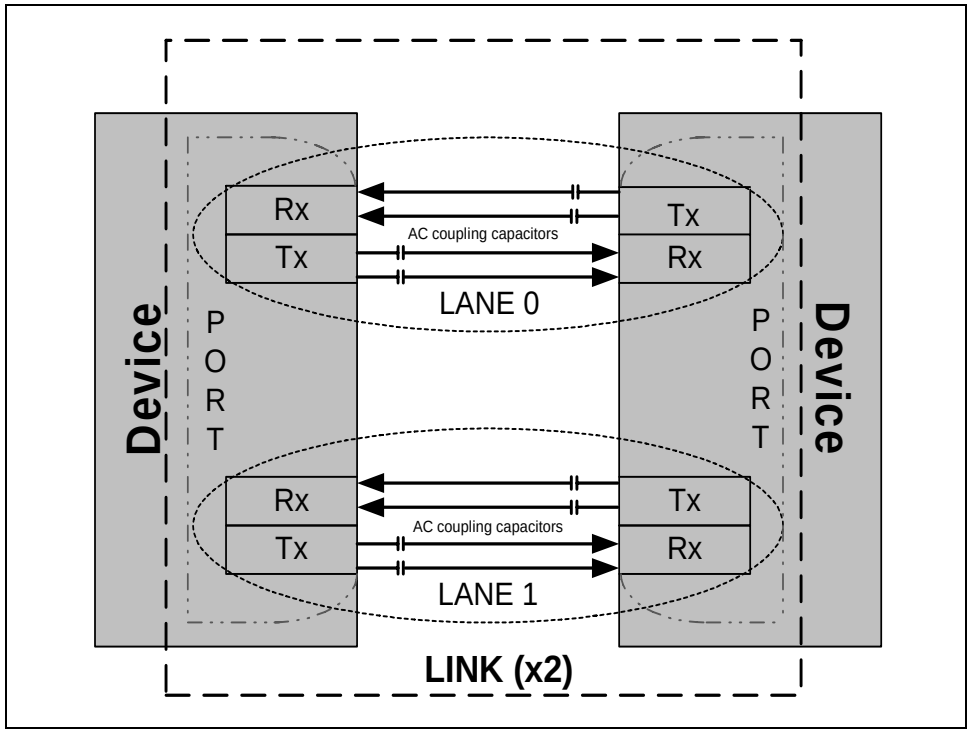
The following sections describe the characteristics of each of these port classes in detail.

5.14.1 Intel 5000X Chipset MCH PCI Express Port Overview

The Intel 5000X Chipset MCH utilizes general purpose PCI Express high speed ports to achieve superior I/O performance. The MCH PCI Express ports are compliant with the *PCI Express Interface Specification, Rev 1.0a*.

A PCI Express port is defined as a collection of bit lanes. Each bit lane consists of two differential pairs in each direction (transmit and receive) as depicted in Table 5-19.

Figure 5-19. x2 PCI Express Bit Lane



The raw bit-rate per PCI Express bit lane is 2.5 Gbit/s. This results in a real bandwidth per bit lane pair of 250 MB/s given the 8/10 encoding used to transmit data across this interface. The result is a maximum theoretical realized bandwidth on a x4 PCI Express port of 1 GB/s in each direction.

Each of the Intel 5000X Chipset MCH PCI Express port are organized as four bi-directional bit lanes, and are referred to as a x4 port.

5.14.2 Enterprise South Bridge Interface (ESI)

The ESI is the Intel® 631xESB/632xESB I/O Controller Hub to Intel 5000X Chipset MCH interface. The available bandwidth to the Intel® 631xESB/632xESB I/O Controller Hub can be increased by using the one or more of the DMA Engine enabled PCI Express ports. Figure 5-20 depicts the ESI port and the two DMA Engine PCI Express ports.

5.14.3 PCI Express DMA Engine Ports

The PCI Express DMA Engine Ports (Port2 & Port 3) feature DMA Engine DMA technology. The DMA Engine ports 2 and port 3 are general purpose x4 PCI Express ports that may be used to connect to PCI Express devices. The possible configurations of the PCI Express ports are depicted in Figure 5-20.

By configuring the DMA Engine ports with the ESI port to the Intel® 631xESB/632xESB I/O Controller Hub, bandwidth is definable from 1GB/s in each direction up to a maximum of 6GB/s bi-directional. Figure 5-21 depicts the various combinations of ESI and DMA Engine ports connecting to the Intel® 631xESB/632xESB I/O Controller Hub. The DMA Engine ports are also general purpose PCI Express ports that may be used as high performance interfaces to other PCI Express devices.

Figure 5-20. ESI and DMA Engine PCI Express Ports

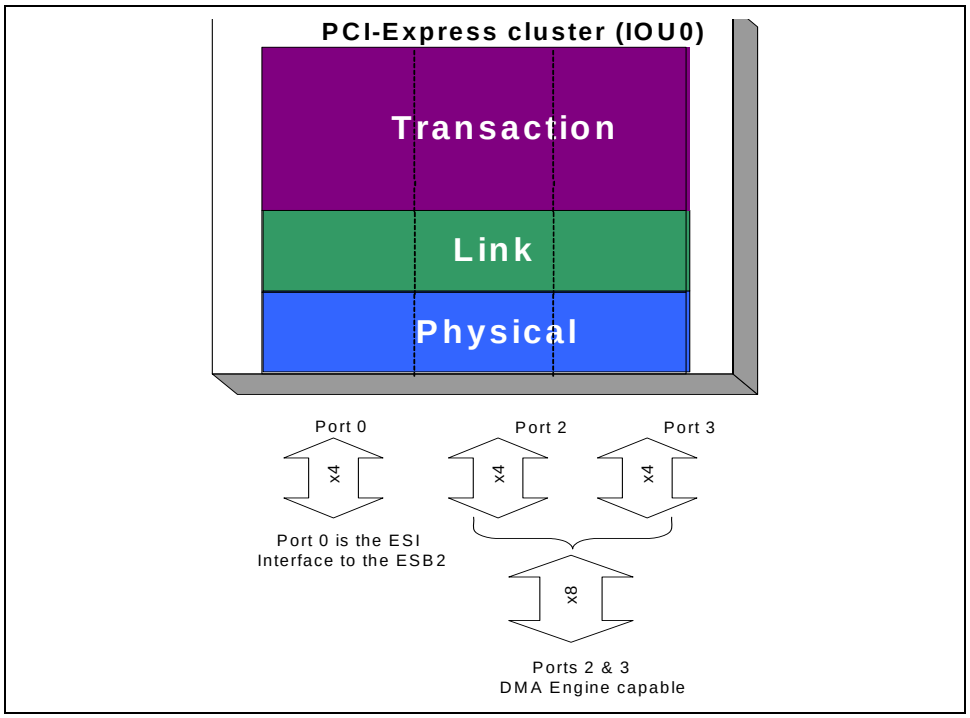
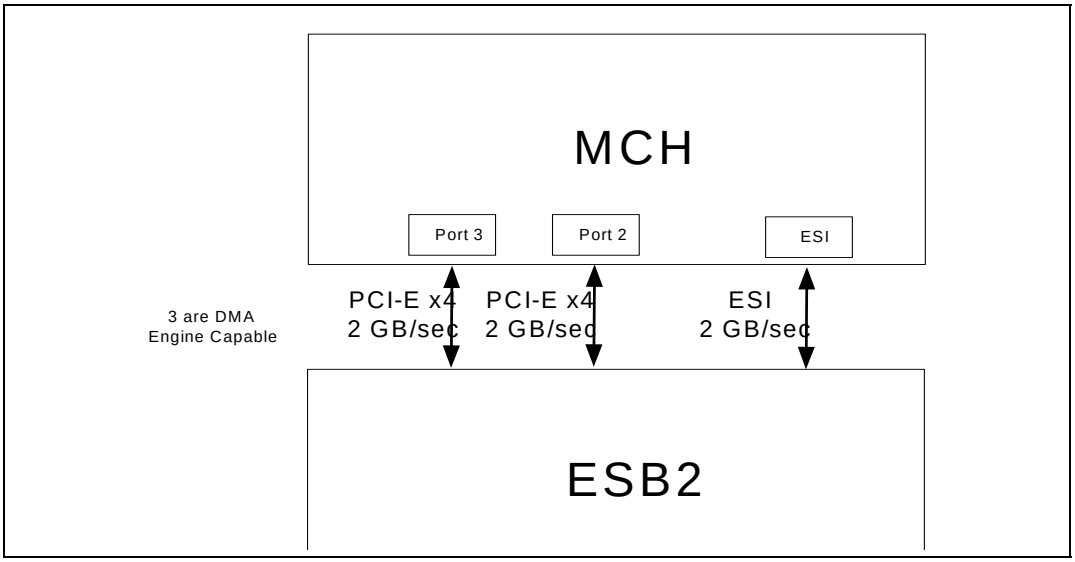


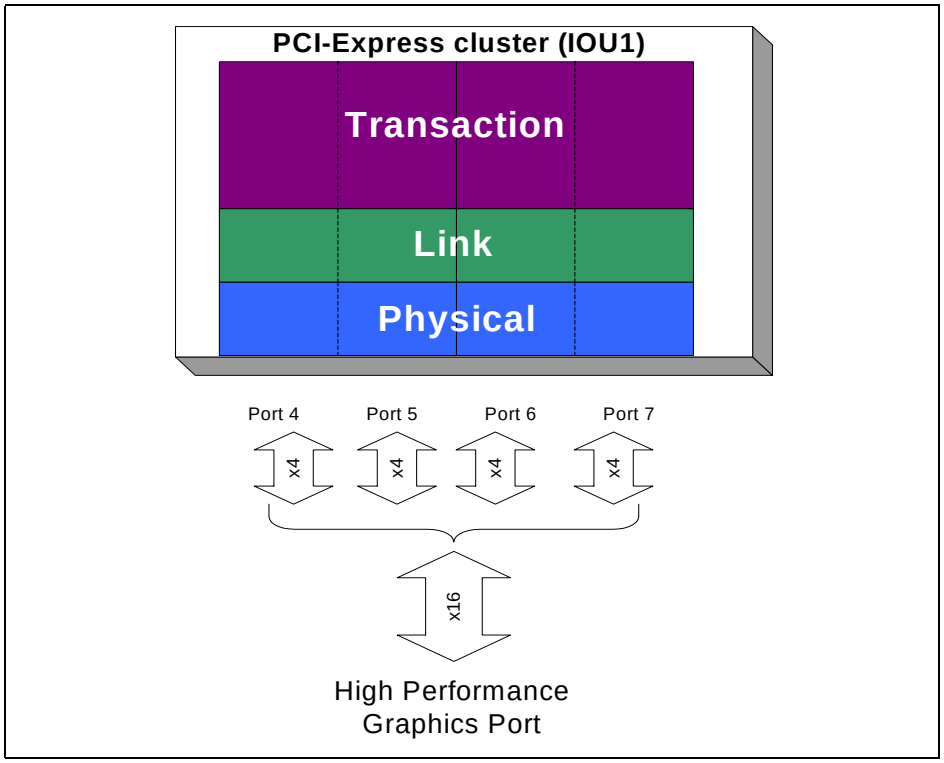
Figure 5-21. MCH to Intel® 631xESB/632xESB I/O Controller Hub (ESB2) Port Configurations



5.14.4 PCI Express General Purpose Ports

Port 4, Port 5, Port 6, and Port 7 are also DMA Engine ports and are configurable for general purpose I/O applications. The Intel 5000X Chipset MCH combines these four general purpose x4 ports into a single optimized x16 high performance graphics interface. This interface is depicted in Figure 5-12, “Intel 5000X Chipset PCI Express High Performance x16 Port” on page 370. These ports contain several architectural enhancements to improve graphics performance.

Figure 5-22. Intel 5000X Chipset PCI Express* High Performance x16 Port



5.14.5 Supported Length Width Port Partitioning

To establish a connection between PCI Express endpoints, they both participate in a sequence of steps known as training. This sequence will establish the operational width of the link as well as adjust skews of the various lanes within a link so that the data sample points can correctly take a data sample off of the link. In the case of a x8 port, the x4 link pairs will first attempt to train independently, and will collapse to a single link at the x8 width upon detection of a single device returning link ID information upstream. Once the number of links has been established, they will negotiate to train at the highest common width, and will step down in its supported link widths in order to succeed in training. The ultimate result may be that the link has trained as a x1 link. Although the bandwidth of this link size is substantially lower than a x8 link or x4 link, it will allow communication between the two devices. Software will then be able to interrogate the device at the other end of the link to determine why it failed to train at a higher width.

This autonomous capability can be overridden by the values sampled on the PEWIDTH[3:0] pins. Table 5-4 illustrates the PEWIDTH strapping options for various link widths in the PCI Express ports in the MCH.

Table 5-4. PCI Express Link Width Strapping Options for Port CPCI Configuration in MCH

PEWIDTH[3:0]	Port0 (ESI)	Port2	Port3	Port4	Port5	Port6	Port7
0000	x4	x4	x4	x4	x4	x4	x4
0001	x4	x4	x4	x4	x4	x8	
0010	x4	x4	x4	x8		x4	x4

Table 5-4. PCI Express Link Width Strapping Options for Port CPCI Configuration in MCH

PEWIDTH[3:0]	Port0 (ESI)	Port2	Port3	Port4	Port5	Port6	Port7
0011	x4	x4	x4	x8		x8	
0100	x4	x4	x4	x16			
others	Reserved						
1000	x4	x8		x4	x4	x4	x4
1001	x4	x8		x4	x4	x8	
1010	x4	x8		x8		x4	x4
1011	x4	x8		x8		x8	
1100	x4	x8		x16			
others	Reserved						
1111	x4.	All port widths determined by link negotiation.					

Note: Intel® 5000V chipset does not have PCI Express ports 4, 5, 6, and 7. So the only option is to configure ports 2 and 3 as a single x8 or two x4 ports.

Note: The *PCI Express Base Specification*, Revision 1.0a requires that a port be capable of negotiating and operating at the native width and 1x. The Intel 5000X Chipset MCH will support the following link widths for its PCI-Express ports viz., x16, x8, x4, x2 and x1. During link training, the MCH will attempt link negotiation starting from its native link width from the highest and ramp down to the nearest supported link width that passes negotiation. For example, a port strapped at 8x, will first attempt negotiation at 8x. If that attempt fails, an attempt is made at x4, then a x2 and finally a x1 link.Note that the x8, x4 and x2 link widths will only use the LSB positions from lane 0 while a x1 can be connect to any of the 4 positions (lane0,lane1, lane2, lane3) providing a higher tolerance to single point lane failures. When settling on a narrower width and the straps are used to override the auto-negotiation partitioning, the remaining links are unused. The links will use the LSB wires of the physical layer to route the packets for the negotiated width.

5.14.6 PCI Express Port Support Summary

The following table describes the options and limitations supported by the MCH PCI Express ports.

Table 5-15. Options and Limitations (Sheet 1 of 2)

Parameter	Support
Number of supported ports	The MCH will support six x4 standard PCI Express ports and an additional x4 ESI port for Intel® 631xESB/632xESB I/O Controller Hub. (Total: 6 + 1 = 7 ports)
Max payload	256B
Hot-Plug	Serial port to support pins
Virtual Channels	MCH only supports VC0
Traffic Streams	MCH supports two streams with two priority levels (High / low) for the PCI Express ports 2 and 3. Rest of the ports support only one stream and 1 priority level (including the ESI) on VC0.
Isochrony	MCH does not support isochrony
ECRC	MCH does not support ECRC
Ordering	MCH only supports strict PCI ordering

Table 5-15. Options and Limitations (Sheet 2 of 2)

Parameter	Support
No Snoop	MCH will not snoop processor caches for transactions with the No Snoop attribute
Power Management	The MCH cannot be powered down, but will forward messages, generate PME_Turn_Off and collect PME_TO_Acks. It will provide the PM Capabilities structure. The MCH does not support Active State Power Management nor the L0s state.
No Cable Support & no repeaters	Retry buffers are sized to meet the Intel 5000X Chipset platform requirements for an integrated DP chassis and which do not require cable or repeater support. Only an 8 inches of FR4 internal trace connector latency is assumed.
Poisoning	MCH will poison data that it cannot correct

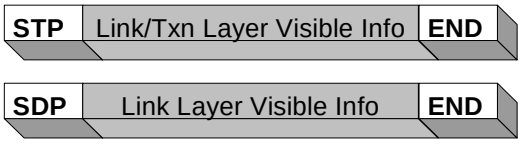
5.14.7 PCI Express Port Physical Layer Characteristics

The PCI Express physical layer implements high-speed differential serial signalling using the following techniques:

- Differential signalling (1.6V peak-to-peak)
- 2.5 GHz data rate (up to 2 GB/s/direction peak bandwidth for a x8 port)
- 8b/10b encoding for embedded clocking and packet framing
- Unidirectional data path in each direction supporting full duplex operation
- Random idle packets and spread-spectrum clocking for reduced EMI
- Loop-back mode for testability
- Lane reversal
- Polarity Inversion

Figure 5-23 illustrates the scope of the physical layer on a PCI Express packet. There are two types of packets: Link layer packets and Transaction Layer Packets. The physical layer is responsible for framing these packets with STP/END symbols (Transaction Layer Packets) and SDP/END symbols (Data Link Layer packets). The grayed out segment is not decoded by the Physical layer.

Figure 5-23. PCI Express Packet Visibility By Physical Layer



5.14.7.1 PCI Express Training

To establish a connection between PCI Express endpoints, they both participate in a sequence of steps known as training. This sequence will establish the operational width of the link as well as adjust skews of the various lanes within a link so that the data sample points can correctly take a data sample off of the link. In the case of a x8 port, the x4 link pairs will first attempt to train independently, and will collapse to a single link at the x8 width upon detection of a single device returning link ID information upstream. Once the number of links has been established, they will negotiate to train at the highest common width, and will step down in its supported link widths in order to succeed in training. The ultimate result may be that the link has trained as a x1 link. Although the bandwidth of this link size is substantially lower than a x8 link or x4 link,

it will allow communication between the two devices. Software will then be able to interrogate the device at the other end of the link to determine why it failed to train at a higher width.

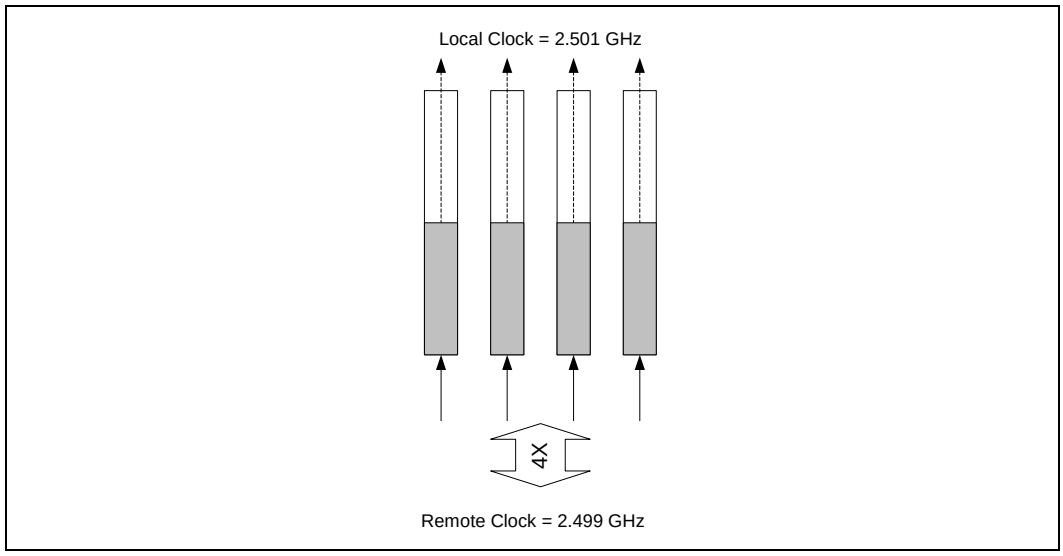
5.14.7.2 8b/10b Encoder/Decoder and Framing

As a transmitter, the physical layer is responsible for encoding each byte into a 10 bit data symbol before transmission across the link. Packet framing is accomplished by the physical layer by adding special framing symbols (STP, SDP, END). PCI Express implements the standard Ethernet and InfiniBand* 8b/10b encoding mechanism.

5.14.7.3 Elastic Buffers

Every PCI Express port implements an independent elastic buffer for each PCI Express lane. The elastic buffers are required since the Intel 5000X Chipset MCH and PCI Express endpoints could be clocked from different sources. Clocks from different sources will never be exactly the same. The outputs of the elastic buffers feed into the deskew buffer.

Figure 5-24. PCI Express Elastic Buffer (x4 Example)

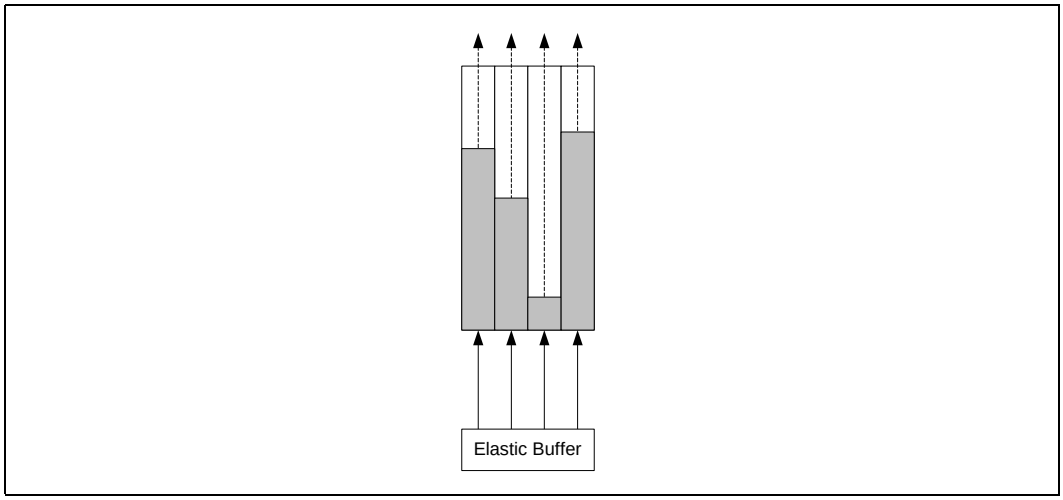


The elastic buffer is eight symbols deep. This accounts for three clocks of synchronization delay, the longest possible TLP allowed by the Intel 5000X Chipset MCH (256B), a 600ppm difference between transmitter and receiver clocks, and worst case skip ordered sequence interval of 1538, framing overheads, and a few symbols of margin.

5.14.7.4 Deskew Buffer

Every PCI Express port implements a deskew buffer. The deskew buffer compensates for the different arrival times for each of the symbols that make up a character. The outputs of the deskew buffer is the data path fed into the Link layer.

Figure 5-25. PCI Express Deskew Buffer (4X Example)



At reset, the delay of each lane in the deskew buffer is adjusted so that the symbols on each lane are aligned. The receiver must compensate for the allowable skew between lanes within a multi-lane link before delivering the data and control to the data link layer. The deskew buffer is eight symbols deep to compensate for up to 20ns of skew between lanes.

5.14.7.5 Polarity Inversion

The *PCI Express Base Specification*, Revision 1.0a defines a concept called polarity inversion. Polarity inversion allows the board designer to connect the D+ and D- lines incorrectly between devices. The Intel 5000X Chipset MCH supports polarity inversion.

5.14.8 Link Layer

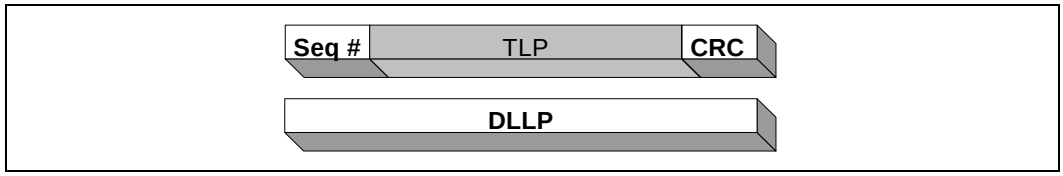
The Data Link Layer of the PCI Express protocol is primarily responsible for data integrity. This is accomplished with the following elements:

- Sequence number assignment for each packet
- ACK/NAK protocol to ensure successful transmission of every packet
- CRC protection of packets
- Time-out mechanism to detect “lost” packets
- Credit exchange

Figure 5-26 illustrates the scope of the link layer on a PCI Express packet. There are two types of packets: data link layer packets (DLLP) and Transaction Layer Packets (TLP). Data Link layer packets are sent between the Link layers of each PCI Express device and do not proceed to the Transaction Layer.

For Transaction layer packets (TLP), the link layer is responsible for prepending sequence numbers and appending 32-bit CRC. The grayed out segment is not decoded by the Data Link layer.

Figure 5-26. PCI Express Packet Visibility By Link Layer



5.14.8.1 Data Link Layer Packets (DLLP)

Refer to *PCI Express Base Specification*, Revision 1.0a for an explicit definition of all the fields in a Data Link Layer packet.

DLLPs are used to ACK or NAK packets as they are sent from the transmitter to the receiver. They are sent by the receivers of the packet to indicate to the transmitter that a packet was successfully received (ACK) or not (NAK). DLLPs are also used to exchange credit information between the transmitter and receiver.

DLLPs are protected with 16b CRC. If the CRC of a received DLLP indicates an error, the DLLP is dropped. This is safe because the PCI Express protocol supports dropping these packets and the next DLLP allows the transmitter to process successfully.

5.14.8.2 ACK / NAK

The Data Link layer is responsible for ensuring that packets are successfully transmitted between PCI Express agents. PCI Express implements an ACK/NAK protocol to accomplish this. Every packet is decoded by the physical layer and forwarded to the link layer. The CRC code appended to the packet is then checked. If this comparison fails, the packet is “retried”.

If the comparison is successful, an ACK is issued back to the transmitter and the packet is forwarded for decoding by the receiver’s Transaction layer. Typically, as each packet is successfully received by the Data Link layer, the receiver issues an ACK. However, the PCI Express protocol allows that ACKs can be combined.

5.14.8.3 Link Level Retry

The *PCI Express Base Specification*, Revision 1.0a lists all the conditions where a packet gets negative acknowledged. One example is on a CRC error. The link layer in the receiver is responsible for calculating 32b CRC (using the polynomial defined in the *PCI Express Base Specification*, Revision 1.0a) for incoming packets and comparing the calculated CRC with the received CRC. If they do not match, then the packet is retried by negative acknowledging the packet with a NAK DLLP and specifying the sequence number of the last good packet. Subsequent packets are dropped until the reattempted packet is observed again.

When the transmitter receives the NAK, it is responsible for retransmitting the packet. Furthermore, any packets sent after the last good packet will also be resent since the receiver has dropped any packets after the corrupt packet.

The transmitter keeps track of packets that have been sent but not acknowledged through the use of a retry buffer. Transactions are added to the buffer as they are on the PCI Express port. Transactions are removed from the buffer after they have been acknowledged by the receiver.

5.14.8.4 ACK Time-out

Packets can get “lost” if the packet is corrupted such that the receiver’s physical layer does not detect the framing symbols properly. Normally, lost packets are detectable with non-linearly incrementing sequence numbers. A time-out mechanism exists to detect (and bound) cases where the *last* packet sent (over a long period of time) was corrupted. A replay timer bounds the time a retry buffer entry waits for an ACK or NAK. Refer to the *PCI Express Base Specification*, Revision 1.0a for details on this mechanism for the discussion on Retry Management and the recommended timer values.

5.14.9 Flow Control

The PCI Express mechanism for flow control is credit based and only applies to TLPs. DLLP packets do not consume any credits. Through initial hardware negotiation and subsequent updates, a PCI Express transmitter is aware of the credit capabilities of the interfacing device. A PCI Express requester will never issue a transaction when there are not enough advertised credits in the other component to support that transaction. If there are not enough credits, the requester will hold off that transaction until enough credits free up to support the transaction. If the ordering rules and available credits allow other subsequent transactions to proceed, the MCH will allow those transactions.

For example, assume that there are no Non-Posted Request Header Credits (NPRH) credits remaining and a memory write is the next transaction in the queue. PCI Express ordering rules allow posted writes to pass reads. Therefore, the Intel 5000X Chipset MCH will issue the memory write. Subsequent memory reads from the source device must wait until enough NPRH credits free up.

Note: Flow control is orthogonal with packet ACKs.

The PCI Express flow control credit types are described in Table 5-16. The *PCI Express Base Specification*, Revision 1.0a defines which TLPs are covered by each flow control type.

Table 5-16. PCI Express Credit Mapping for Inbound Transactions (Sheet 1 of 2)

Flow Control Type	Definition	Initial MCH Advertisement
Inbound Posted Request Header Credits (IPRH)	Tracks the number of inbound posted requests the agent is capable of supporting. Each credit accounts for one posted request.	14 (4x) 28(8x) 56(x16)
Inbound Posted Request Data Credits (IPRD)	Tracks the number of inbound posted data the agent is capable of supporting. Each credit accounts for up to 16 bytes of data.	54 (4x) 108(8X) 216(16X)
Inbound Non-Posted Request Header Credits (INPRH)	Tracks the number of non-posted requests the agent is capable of supporting. Each credit accounts for one non-posted request.	14 (4X) 28(8X) 56(16X)

Table 5-16. PCI Express Credit Mapping for Inbound Transactions (Sheet 2 of 2)

Flow Control Type	Definition	Initial MCH Advertisement
Inbound Non-Posted Request Data Credits (INPRD)	Tracks the number of non-posted data the agent is capable of supporting. Each credit accounts for up to 16 bytes of data.	2 (4X) 4 (8X) 8 (16X)
Completion Header Credits (CPH) (outbound request completions received at the MCH)	Tracks the number of completion headers the agent is capable of supporting.[a]	0 (Infinite) (4) [x4] (8) [x8] (16) (x16)
Completion Data Credits (CPD) (outbound request completions (data) received at the MCH)	Tracks the number of completion data the agent is capable of supporting. Each credit accounts for up to 16 bytes of data.	0 (Infinite) (8) [x4] (16) [x8] (32) [x16]

Notes:

a. Root complexes and end points are permitted to advertise an infinite number of credits for completions. Though the MCH implements finite queue structures as indicated in bracket for the completions on the inbound side, by construction, it will never overflow since for each outbound request, the MCH allocates sufficient space on the inbound side. i.e guarantee by construction

Table 5-17. PCI Express Credit Mapping for Outbound Transactions

Flow Control Type	Definition	Initial MCH Advertisement
Outbound Posted Request Header Credits (OPRH)	Tracks the number of outbound posted requests the agent is capable of supporting. Each credit accounts for one posted request.	4(4x) 8(8x) 16(16)
Outbound Posted Request Data Credits (OPRD)	Tracks the number of outbound posted data the agent is capable of supporting. Each credit accounts for up to 16 bytes of data.	8(4x) 16(8X) 32(16X)
Outbound Non-Posted Request Header Credits (ONPRH)	Tracks the number of non-posted requests the agent is capable of supporting. Each credit accounts for one non-posted request.	16(4X) 32(8X) 64(16X)
Outbound Non-Posted Request Data Credits (ONPRD)	Tracks the number of non-posted data the agent is capable of supporting. Each credit accounts for up to 16 bytes of data.	16(4X) 32(8X) 64(16X)
Completion Header Credits (CPLH) (inbound request completions from MCH)	Tracks the number of completion headers the agent is capable of supporting.	2(x4) 4(x8) 8(x16)
Completion Data Credits (CPLD) (inbound request completions (data) from the MCH)	Tracks the number of completion data the agent is capable of supporting. Each credit accounts for up to 16 bytes of data.	8(x4) 16(x8) 32(x16)

The credit advertisements for the MCH are shown in Table 5-16 and Table 5-17. Every PCI Express device tracks the above six credit types (inbound) for both itself and the interfacing device. The rules governing flow control are described in the *PCI Express Base Specification*, Revision 1.0a.

5.14.9.1 Credit Update Mechanism, Flow Control Protocol (FCP)

After reset, credit information is initialized with the values indicated in Table 5-16 by following the flow control initialization protocol defined in the *PCI Express Base Specification*, Revision 1.0a. Since the MCH supports only VC0, only this channel is initialized.

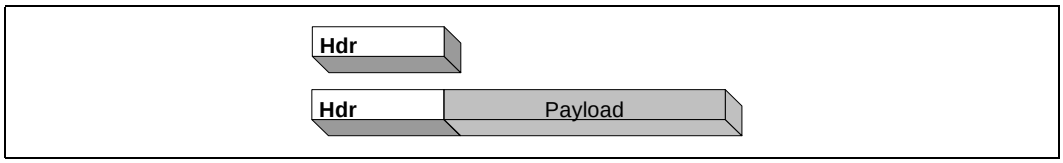
5.14.10 Transaction Layer

The PCI Express Transaction Layer is responsible for sending read and write operations between components. This is the PCI Express layer which actually moves software visible data between components. The transaction layer provides the mechanisms for:

- Software configuration of components
- Communication between the processor bus and different I/O technologies
- Communication between the memory and different I/O technologies

Figure 5-17 illustrates the scope of the transaction layer on a PCI Express packet. Some transaction layer packets have only a header (for example, read request). Some transaction layer packets have a header followed by data (for example, write requests and read completions).

Figure 5-27. PCI Express Packet Visibility By Transaction Layer



5.14.11 DMA Engine Implementation

The MCH will support the DMA Engine technology on all PCI Express Ports for IO performance acceleration, cost optimization, and reduced CPU utilization.

- Four Channel DMA engine for performing basic operations such as memory-to-memory, and memory-to-MMIO transfer with a byte aligned granularity.
- DMA Engine related configuration registers

5.14.12 DMA Engine Usage Model

DMA Engine architecture originally supported two fundamental programming models.

- A Hardware Model where this is no assistance from operating system level software and thus an I/O device driver cannot directly access DMA Engine MMIO registers (only the I/O device can access them).
- A Software Model where there is a DMA Engine driver that enables access to DMA Engine resources by I/O device drivers. Thus, the I/O device driver can access DMA Engine MMIO registers.

The HW Model is obsolete and no longer supported. The following sections describe how the SW model works and specifies requirements on BIOS, the OS, the DMA Engine device driver, and the client for initializing, configuring, managing, and programming the DMA Engine hardware.

Note: The term '*DMA Engine device driver*' refers to an OS based device driver that is loaded to control the DMA Engine device and the term '*I/O device driver*' or '*client*' refers to an

OS based device driver that is loaded to control a client I/O device (which might be a DMA Engine client, that is, the controlling process). Client access is always via the DMA Engine driver, thus stating the client writes a register implies that the client passes the request to the DMA Engine driver, which sets the register.

5.15 Using DMA Engine Technology

This section explains the mechanism by which software can request the use of and configure DMA Engine technology capabilities.

5.15.1 High Level Requirements

A generic mechanism to expose DMA Engine capabilities to clients in an operating system environment should meet the following high-level requirements:

1. It must support a single client using either a subset or all of DMA Engine features at any given time.
2. It must support multiple simultaneous clients, each using a subset or all of DMA Engine features as long as the simultaneous use is non-conflicting.
3. It must support DMA Engine capabilities to be dynamically requested and released. This can happen as client I/O devices are dynamically inserted and removed or the device driver stack for client I/O devices is dynamically loaded and unloaded.
4. It must support DMA Engine capabilities in a manner that is compatible with the host operating system. For example, the mechanism must work correctly when plug-and-play and power management events occur on operating systems that support these features.
5. It must be flexible, such that it allows different DMA Engine implementations (for example, different major or minor versions) to support different capabilities.
6. It must be able to support a DMA Engine implementation that is “asymmetric” - that is, not all DMA Engine capabilities are supported on all ports of the chipset.

Given these high-level requirements, a DMA Engine usage model must support a startup and tear-down sequence that allows these requirements to be met. Thus, any client of DMA Engine facilities must be able to:

1. Detect the presence, version, & capabilities of DMA Engine technology that exists on that platform. This includes the ability to detect and use version specific capabilities and restrictions.
2. Arbitrate for and reserve DMA Engine resources (for example, DMA channels, Write Combining, and Stream Priority) for exclusive use.
3. Release DMA Engine resources (for example, DMA channels, Write Combining, and Stream Priority) when they are no longer needed, so other potential clients can claim them.
4. Manage plug and play and power management events that cause the client I/O device (or DMA Engine) to be dynamically removed or (re)inserted and dynamically powered down and up.

5.15.2 Basic Approaches

5.15.2.1 Software Model - assistance from OS level software

The DMA Engine device is an I/O device (that is, PCI base class and subclass do not indicate a host bridge). The system BIOS is not required to provide special support for DMA Engine and DMA Engine register address space requirement is reported in the PCI configuration header. There is a DMA Engine device driver that loads under the host operating system. This driver assists in performing required startup / tear-down steps and can optionally also provide a measure of abstraction / de-coupling between the client I/O device and DMA Engine. For example, the DMA Engine driver is responsible for detecting the version, capabilities, and limitations of the current DMA Engine implementation, so that the client can get that information from the DMA Engine device driver. In addition, the DMA Engine device driver provides programming interfaces to reserve and release DMA Engine resources (for example, DMA channels). Optionally, the DMA Engine driver could provide an abstracted programming interface to use all DMA Engine capabilities (for example, to program DMA operations) so that the client does not have to be aware of the DMA Engine register layout and register formats in order to use it. Another possible usage model is one in which the host based client uses the DMA Engine driver for startup / tear-down operations but the client directly uses DMA Engine resources (for example, DMA channel) at run time once its device driver has reserved it. In this model:

1. The DMA Engine device driver must provide the clients a mechanism (programming interface) to detect DMA Engine version and capabilities. It must also provide clients a mechanism to reserve and release DMA Engine resources. Optionally, it may provide an abstracted mechanism to use / program DMA Engine capabilities, if this is desired.
2. The client must be able to communicate with the DMA Engine device driver to use its services to reserve and release DMA Engine resources.

Note that some variations in the basic usage models are possible. For example, the OS based device driver for the client I/O device can also act as the DMA Engine device driver or otherwise assist clients with DMA Engine hardware.

It is very difficult to support a model in which some client I/O devices want to use DMA Engine with OS level software assistance while other client I/O devices want to do so without any OS level software assistance. The current version of DMA Engine does not provide any additional capabilities to support such a hybrid model.

Caution: A client may want to use DMA Engine features in the pre-boot environment. This can be accomplished in multiple platform specific ways. For example, the platform BIOS could contain DMA Engine specific code similar to an operating system based DMA Engine driver. However, there are no standard interfaces (for example, INT XX calls) to invoke such DMA Engine specific software in the pre-boot environment, so the client would need to take platform specific steps to invoke it. Another possibility is that the option ROM of the client I/O device could contain DMA Engine specific code to manage DMA Engine.

5.15.3 Power Management Considerations

Chipset devices that implement DMA Engine technology as well as clients that use DMA Engine may support different device power states. At a minimum, all devices in the system must support a D0 device power state that corresponds to the “fully-on” state and a D3 device power state that corresponds to the “fully-off” state. Intermediate device power states D1 and D2 may or may not be supported. Thus, there can be multiple permutations with DMA Engine and/or its client I/O devices supporting the same or different device power states. Care must be taken to ensure that a power

management capable operating system does not put the DMA Engine device to a lower device power (D1, D2 or D3) state while its client I/O device is fully powered on (D0 state) and actively using DMA Engine. Depending on whether DMA Engine is used under an OS environment, this imposes different requirements on the device and platform implementation.

There is a DMA Engine device driver and the host OS can power manage the DMA Engine device through this driver. The software implementation must make sure that the appropriate power management dependencies between the DMA Engine device and its client I/O devices are captured and reported to the operating system. This is to ensure that the operating system does not put the DMA Engine device to a low power (D1, D2 or D3) state while any of its client I/O devices are fully powered on (D0 state) and actively using DMA Engine.

For example, the operating system might attempt to transition the DMA Engine device into the D3 device power state while putting the system into the S4 (hibernate) system power state. In that process, it must not transition the DMA Engine device into the D3 state before transitioning all client I/O devices into the D3 power state. In the same way, when the system resumes from the S4 state, the operating system must transition the DMA Engine device from D3 to the D0 state before transitioning its client I/O devices from D3 to the D0 state.

5.16 Implementation Requirements

This section specifies dependencies on the various programming components for various tasks.

5.16.1 Software Model Dependencies

The following table lists dependencies for the Software Model

Table 5-5. Software Model Dependencies

Task	Realm	Requirements
System Initialization	Chipset HW	• Initialize DMA Engine registers with specified default values
	BIOS	• Mark DMA Engine as an Integrated Device (PCI base class 0x8, subclass / interface 0x80) so that the OS will load the DMA Engine device driver. • Treat DMA Engine device as any other PCI device. • Program the APIC_ID_TAG_MAP register
	DMA Engine Device SW Driver	• none (not loaded yet)
	I/O Device HW	• Does nothing - waits for I/O SW Driver to configure and enable the I/O device.
	I/O Device SW Driver	• none (not loaded yet)
	Other Restrictions, Issues, Notes	• ·Note: DMA Engine services exposed as a Root Complex Integrated Device.

Table 5-5. Software Model Dependencies (Continued)

Task	Realm	Requirements
OS Initialization	Chipset HW	• none
	BIOS	• none
	DMA Engine Device SW Driver	• DMA Engine Device Driver is loaded and takes responsibility of all DMA Engine devices • Reads CB_BAR to discover DMA Engine Integrated Device capabilities and per Port services and resources
	I/O Device HW	• Does nothing - waits for I/O SW Driver to configure and enable the I/O device.
	I/O Device SW Driver	• Driver loads
	Other Restrictions, Issues, Notes	• DMA Engine's Root Complex Integrated Device requires Device Driver • Client Device Driver must load after DMA Engine Device Driver.
Detecting DMA Engine presence, PCI location, and register base address	Chipset HW	• N/a - the chipset still supports CB_Query[a] message and inbound MMIO reads & writes, but they are not used in the SW model.
	BIOS	• none
	DMA Engine Device SW Driver	• Exposes DMA Engine via DMA Engine Driver SW I/F • Store Client CFG info from I/O device driver • Using Client's CFG info, detect which Root Port serves Client Device.
	I/O Device HW	• Waits for its SW driver to program the I/O device and command it to use DMA Engine resources.
	I/O Device SW Driver	• Searches & detects DMA Engine Driver's SW I/F • Registers with DMA Engine driver's SW I/F • Query for DMA Engine services via DMA Engine driver's SW I/F • Passes Client's Requestor ID and CFG Information via DMA Engine driver's SW I/F (such as client I/O Device's BAR)
	Other Restrictions, Issues, Notes	• Note: Client Driver detects DMA Engine Present via DMA Engine driver's SW I/F. Load order dependence requirement. • Use Client PCI-Express Hierarchy location to determine if DMA Engine Port services available for Client.
Allocating DMA Engine Resources	Chipset HW	• None
	BIOS	• None
	DMA Engine Device SW Driver	• Programs DMA Engine registers for Client Device reserving DMA Engine resource
	I/O Device HW	• Waits for its SW driver to command it to use DMA Engine resources
	I/O Device SW Driver	• Via DMA Engine SW driver, client reserves and configures specific DMA Engine resources
	Other Restrictions, Issues, Notes	• Restriction: Each DMA Engine resource can have exactly one owner & sharing is not permitted even for WC ranges, stream ID and so forth, unless completely managed by the DMA Engine driver. • Note: After each Load, Start, or Reset - DMA Engine resources must be reserved before use.

Table 5-5. Software Model Dependencies (Continued)

Task	Realm	Requirements
De-allocate DMA Engine resources	Chipset HW	• None
	BIOS	• None
	DMA Engine Device SW Driver	• Manages client request to free DMA Engine resources and programs DMA Engine device registers to de-allocate resources
	I/O Device HW	• None
	I/O Device SW Driver	• Client, via DMA Engine SW driver, releases DMA Engine Resources
	Other Restrictions, Issues, Notes	• As part of each Client Driver Unload, Reset, or Stop, DMA Engine resources must be De-allocated
Run time management of DMA Engine resources	Chipset HW	• Support inbound MMIO reads and writes.
	BIOS	• None
	DMA Engine Device SW Driver	• Manages DMA Engine errors and so forth. • Signals Client of DMA Engine errors
	I/O Device HW	• None
	I/O Device SW Driver	• Makes necessary adjustments for any DMA Engine errors
	Other Restrictions, Issues, Notes	• None
Managing OS driven plug and play of client HW device	Chipset HW	•
	BIOS	•
	DMA Engine Device SW Driver	• Manages client request to free DMA Engine resources and programs DMA Engine device registers to free indicated resources
	I/O Device HW	•
	I/O Device SW Driver	• Client must release DMA Engine resources during PnP events such as Stop, Surprise removal, and so forth.
	Other Restrictions, Issues, Notes	• Note: Whether a STOP is treated the same as an UNLOAD (that is, free all resources on stop and reclaim them on start) is up to the Client HW/SW combination. • If DMA Engine driver is Unloaded or Stopped, Client Driver will be notified to De-allocate DMA Engine resources.
Managing OS driven power management events to the client Client HW and manage system sleep state transitions	Chipset HW	•
	BIOS	•
	DMA Engine Device SW Driver	• Must be able to take device power state transition commands from OS and notify client SW to de-allocate DMA Engine resources
	I/O Device HW	• Must be able to drain DMA Engine traffic & release DMA Engine resources when SW issues the command to move to D3 state • Must indicate to SW that it is now OK to report transition to D3 state
	I/O Device SW Driver	• Must be able to take device power state transition commands from OS and request Client HW to move to new (for example, D3) state • Must be able to drain DMA Engine traffic in hardware if any & release DMA Engine resources when OS SW issues command to move to D3 state • Must indicate to OS SW that it is now OK to report transition to D3 state
	Other Restrictions, Issues, Notes	• If DMA Engine driver is powered Down or changed to low power state, Client SW Drivers will be notified to De-allocate DMA Engine resources

Notes:

a. To assure that PCI enumeration has completed, an I/O device should not send a DMA Engine query until its own device driver has been loaded. Otherwise the DMA Engine device may not be able to respond.

5.17 Programming Flow

This section describes the steps for a client to use DMA Engine facilities.

Note: Locking of DMA Engine MMIO space is undesirable, unpredictable, and might result in deadlock. Thus, an attempt to generate a locked request to DMA Engine MMIO location is invalid.

5.17.1 General

The I/O device driver queries the DMA Engine driver to learn the location of DMA Engine registers. The DMA Engine device driver needs to know the address of the I/O device so it can determine which PCI Express port serves the device and thus select the correct set of per-port registers. Additionally, I/O device drivers will request use of DMA Engine resource via the DMA Engine driver.

5.17.2 Using DMA

In the following discussion, the term Device means I/O device and its I/O device driver.

The DMA Engine driver reads "Channel Count Register (CHANCNT)" and checks how many DMA channels are supported. A value of zero indicates no DMA support.

The client requests and the DMA Engine driver grants permission to use a particular channel. The client accesses DMA Engine hardware through the DMA Engine driver.

Note: The DMA Engine driver should not directly expose "Interrupt Control Register (INTRCTRL)" register. The I/O driver registers its interrupt handler with the DMA Engine device (see Section 5.17.3, "Interrupt Handling").

If the client successfully claims ownership of a DMA channel, then the client writes the CHANCTRL to configure the channel's DMA operation.

Client writes the "Channel Completion Address Register (CHANCMP)" to specify where the DMA channel writes the Completion Status.

The DMA channel is now operational.

Client reads the "Transfer Capacity Register (XFERCAP)" register to learn the maximum transfer capacity for the DMA channel.

To start DMA operation, the client builds DMA descriptors in main memory, writes "Descriptor Chain Address Register (CHAINADDR)" to indicate the first descriptor in the chain, and then sets the "DMA Descriptor Count Register (DMACOUNT)" to initiate DMA transfer.

Note: To avoid snooping cycles, the client can set **Destination address snoop control** and **Source address snoop control** in the Descriptor and **Descriptor address snoop control** in CHANCTRL. However, the client must make sure that processors do not access those memory locations to avoid coherency problems resulting from processor caching. Thus the I/O driver must not access that memory. Even then, when the I/O driver first allocates the memory, locations might may reside in a processor's cache due to a previous process

owning the memory. Before using the no-snoop options, the client must make sure caches are flushed. One way to flush all caches for a particular memory buffer is to program the DMA channel to move the buffer to itself (or move the first half of the buffer to the last half). This must be done without setting the 'no snoop' control bits.

For hot removal, the client must release DMA Engine resources.

To release DMA resources, the client makes sure DMA operation is halted or aborts DMA operation. The client relinquishes control via the DMA Engine driver.

5.17.3 Interrupt Handling

Note: Interrupt handling is very OS specific. This section illustrates how DMA Engine facilities are architected to be used by explaining how the various software entities can use DMA Engine facilities to process interrupts. Refer to the DMA Engine Driver specification for OS specific details and implementation requirements. Thus, this section serves as a guide for the DMA Engine Driver specification.

The DMA Engine driver provides a DMA Engine Interrupt Service Routine (ISR) that is dispatched when the DMA Engine device generates an interrupt. The ISR disables interrupts from the DMA Engine device and dispatches the DMA Engine Interrupt Handler, which calls one or more Channel Interrupt Callbacks to process the interrupt. When all interrupt conditions have been processed, interrupts are re-enabled.

For a client I/O device driver to use DMA Engine interrupts, it must register its *Channel Interrupt Callback* with the DMA Engine Driver and then enable interrupts by setting the "*Interrupt upon the completion of this descriptor*" bit in the Descriptor Control Field of descriptors for which it wants completion interrupts and may optionally set the *Error Interrupt Enable* in the CHANCTRL register.

The DMA Engine driver sets the Master Interrupt Enable bit in the INTRCTRL register to enable interrupts.

The DMA Engine ISR is invoked when any channel generates an interrupt. The ISR dispatches the DMA Engine interrupt handler that determines which channels are generating a Channel Attention and for each channel requiring attention, calls that channel's interrupt handler.

The ISR reads the Attention Status register to determine which channels generated an interrupt (the read resets the Attention Status register). When a DMA channel generates an interrupt, the chipset sets the appropriate bit in the ATTNSTATUS register and sets the Interrupt Disable bit in the CHANCTRL register. This inhibits that channel from generating another interrupt until the software clears the *Interrupt Disable* bit in the CHANCTRL register. Interrupt events that occur while Interrupt Disable is set are remembered and might cause an interrupt after Interrupt Disable is reset.

5.17.3.1 Interrupt Service Routine (ISR)

The ISR reads the INTRCTRL register to validate that there is an interrupt that needs service. Reading the INTRCTRL resets the Master Interrupt Enable, which disables the interrupt generation and clears the Interrupt bit. In a multi-processor system, this allows the first CPU that reads the INTRCTRL after an interrupt to read the Interrupt bit set and ISRs on ISR instances on other CPUs will read the bit reset and know that another instance of the ISR is processing interrupts.

- If the ISR reads the INTRCTRL register and the Master Interrupt Enable is not set, This indicates that another CPU's ISR is processing the interrupt. In this case, the ISR simply returns without dispatching the DMA Engine Interrupt Handler.

- If the ISR reads the INTRCTRL register and Master Interrupt Enable is set, but Interrupt Status is not, this indicates either a spurious interrupt or that another device is sharing the interrupt level. In either case, the ISR re-enables interrupts by setting the Master Interrupt Enable and returns without dispatching the DMA Engine Interrupt Handler.
- If the ISR reads the INTRCTRL register and Master Interrupt Enable and Interrupt Status are both set, this indicates this is the first ISR and it dispatches the DMA Engine interrupt handler, which is responsible for re-enabling interrupts by setting the master interrupt enable.

Table 5-6 illustrates the ISR's reaction to reading INTRCTRL.

Table 5-6. INTRCTRL interpretation

Master Interrupt Enable	Interrupt Status	Implied Meaning	ISR Actions
1	0	Spurious interrupt	Set Master Interrupt Enable and return (no interrupt)
1	1	Valid interrupt	Dispatch DMA Engine Interrupt Handler and return - Interrupt handler will re-enable interrupts
0	x	Interrupt already being handled	return (no interrupt)

5.17.3.2 DMA Engine Interrupt Handler

The ISR dispatches the DMA Engine interrupt handler whenever any channels need servicing. The handler performs the following steps.

1. Read the Attention Status (ATTNSTATUS) register to determine which channels require attention.
2. Call the respective channel's interrupt routine.
3. Re-read the ATTNSTATUS register to determine if any additional channels require attention.
   a. If not all zeros, go to step 2. and repeat.
   b. If all zeros, continue with next step
4. Set the master interrupt enable in INTRCTRL
5. Terminate.

The DMA Engine Interrupt handler calls each of the appropriate interrupt handlers and when they are finished, it may re-read the ATTNSTATUS (step 3.) to see if any new interrupts are pending, and if so, repeats the process. When all interrupts have been serviced, the DMA Engine interrupt handler sets the master interrupt enable and ends. Another possibility is for the DMA Engine Interrupt handler to dispatch the appropriate channel interrupt handlers, set the master interrupt enable and end (that is, skip step 3., ending as soon as it dispatches the appropriate channel handlers).

Note that the chipset automatically sets the channel's Interrupt Disable bit in the CHANCTRL register when an interrupt event sets the corresponding ATTNSTATUS bit. Thus, step 2. prevents step 3.a for the same channel, unless the channel interrupt callback had processed the interrupts, cleared the Interrupt Disable bit, and another interrupt event occurred.

5.17.3.3 Channel Interrupt Callback

The channel's interrupt handler reads the channel's status (CHANSTS), processes the completions, process errors (CHANERR), and then re-enables the channel's interrupt by clearing the channel's Interrupt Disable bit in the CHANCTRL register. After the Channel Interrupt Callback clears the Interrupt Disable bit, it may re-read the CHANSTS and CHANERR to see if any new interrupt events occurred between the time it last read them and it set the Interrupt Disable bit.

1. Read the Channel Status (CHANSTS) register to determine the current descriptor and its status.
2. Process completions for all previous descriptors (implied good status).
3. Process current descriptor.
   a. If good status, process completion.
   b. If not good status, perform error processing and reset error bits in CHANERR register.
4. Write CHANCTRL to clear Interrupt Disable.
5. Read the Channel Status (CHANSTS) register to determine if additional descriptors have completed.
   a. If not same descriptor as previous CHANSTS value, go to step 2. and repeat.
   b. If same descriptor, continue with next step.
6. Read the Channel Error (CHANERR) register to determine if additional errors have occurred.
   a. If errors, perform error processing and reset error bits in CHANERR register.
   b. If no errors, continue with next step.
7. Return.

Note that steps 5. and 6. are optional since any interrupt event that occurs after the last read of the CHANSTS register will generate another interrupt (see Section 5.17.3, "Interrupt Handling"). Additionally, the handler may choose to read the completion write memory location rather than the CHANSTS register. By not reading the CHANSTS register, any interrupt event that occurs after the one that generated the interrupt, will cause another interrupt to be generated after the handler resets Interrupt Disable. Otherwise, any interrupt event that occurs after the last read of the CHANSTS will cause another interrupt to be generated.

5.18 DMA Engine Driver

Note: This section serves as a guide for the DMA Engine Driver specification and describes the tasks of the DMA Engine driver with respect to how the DMA Engine driver interfaces with the chipset. Refer to the DMA Engine Driver specification for specific details and implementation requirements.

The DMA Engine device appears as a normal PCI Express device and thus the OS will load the DMA Engine driver by matching the Vendor and Product information in the PCI configuration registers of the DMA Engine device.

The DMA registers are at a fixed offset from CB_BAR. The per-port registers are located via the "Per-port Offset Register (PERPORT_OFFSET)" which points to the first set of per port registers. These registers form a linked list with the "Next Per-Port Register Set (NXTPPRSET)" of each set pointing to the next set and the NXTPPRSET of last set being set to zero. Thus the driver walks the chain to locate all of the ports with DMA Engine resources.

The DMA Engine driver is responsible for supplying the DMA Engine ISR and DMA Engine Interrupt Handler (see Section 5.17.3.1, "Interrupt Service Routine (ISR)").

When an I/O driver requests DMA channels, if it wants to use interrupts, it must provide the DMA Engine driver with the address of its DMA Channel Interrupt Callback routine, so the DMA Engine interrupt handler can call the Channel Interrupt Callback to process that channels interrupts.

When a client requests per-port resources, the DMA Engine driver must be able to determine which post serves the target I/O device. To do this, the requesting driver must pass the DMA Engine driver the base address of the I/O device. The DMA Engine device then walks through the per-port register sets looking at "Bridge Memory Base Register (BR_MEM_BASE)", "Bridge Memory Limit Register (BR_MEM_LIMIT)", "Bridge Prefetchable Memory Base Register (BR_PMEM_BASE)", to determine which port serves the I/O device.

If at anytime the DMA Engine device driver gets suspended, it must notify the I/O drivers and make sure that they have released their DMA Engine resources (such as clear the REQID register, WCBASE, WCSIZE, and WCNUMB registers).

5.18.1 USWC Writes

The MCH will defer all outbound writes to MMIO. To maximize performance, the processor should use USWC attribute for write combining and then evict the accumulated cache line to the chipset for further coalescing.

5.18.2 Stream/Port Arbitration

Arbitration in the PCI Express (IOU) cluster(s) occurs at several levels to dispatch transactions to/from memory. The basic methodology is to guarantee fairness, prevent starvation and provide some degree of programmability for debug/fine-tuning.

- Level 0 (lowest) - In order to provide better serviceability for high-priority streams from Ports 2 and 3, the Intel 5000X Chipset MCH employs a mixed stream/port arbitration within IOU0 cluster as follows:
  - The low-priority streams from Ports 2 and 3 are grouped along with Port 0 (ESI) and round-robin arbitration selects a request from these queues i.e low-priority (Port 2), low-priority (Port 3) and ESI (Port 0)
  - The high-priority streams from Ports 2 and 3 are grouped together and round-robin arbitration proceeds for these requests from high-priority (Port 2) and high-priority (Port 3)
- Level 1: Programmable, weighted round-robin algorithm that alternates between the high priority and low priority stream requests selected from Level 0 of IOU0 and classifies them as Posted/Non Posted.
- Level 1a: Port Arbitration: Ports 4, Port 5, Port 6 and Port 7
  - Select one of the requests from IOU1 in a round robin fashion and classify as Posted/Non-Posted
- Level 2: Simple round robin algorithm between the Posted, Non-posted queues, and completions from Level 1 (IOU0) and Level 1a (IOU1) respectively.
- Level 3 (highest): Programmable, weighted round robin algorithm that connects the IOU0, IOU1 and SDMA engine with the CDB to share the allocated 8.53/10.6 GB/s bandwidth in each direction.

5.18.2.1 Level 3 - IOU0, IOU1, DMA Arbitration

The Level 3 arbitration shares available bandwidth in the CDB port by a programmable, weighted, round robin scheme that scans for requests from the two IOUs (IOU0 & IOU1) and the DMA engine optimizing bandwidth and avoiding starvation. The CDB port has a bandwidth of 8.53 GB/s (10.6 GB/s) and is proportionately divided based on the arbitration values defined for DMA, IOU0, IOU1 in the PEXCTRL2[7:2:0], PCI Express Control Register 2.

5.18.3 Supported PCI Express Transactions

Table 5-18 lists all the transactions supported by the Intel 5000X Chipset MCH which are expected to be received from the PCI Express interface. Similarly, Table 5-20 lists all the transactions to be expected by an attached PCI Express component. Refer to the *PCI Express Base Specification*, Revision 1.0a for the specific protocol requirements of this interface.

Table 5-18. Incoming PCI Express Requests

PCI Express Transaction	Address Space or Message	Intel 5000X Chipset MCH Response
Inbound Write Requests	Memory	Forward to Main Memory or PCI Express or ESI port depending on address.
	I/O	Forward to peer PCI Express or ESI port
Inbound Read Requests	Memory	Forward to Main Memory, or PCI Express or ESI
	I/O	Forward to peer PCI Express or ESI Interface port
Inbound Message	ASSERT_INTA	Inband interrupt assertion/deassertion emulating PCI interrupts.
	DEASSERT_INTA	
	ASSERT_INTB	Inband interrupt assertion/deassertion emulating PCI interrupts.
	DEASSERT_INTB	
	ASSERT_INTC	Inband interrupt assertion/deassertion emulating PCI interrupts.
	DEASSERT_INTC	
	ASSERT_INTD	Inband interrupt assertion/deassertion emulating PCI interrupts.
	DEASSERT_INTD	
	ERR_COR	Propagate as an interrupt to system.
	ERR_UNC	Propagate as an interrupt to system.
	ERR_FATAL	Propagate as an interrupt to system.
	PM_PME	Propagate as a general purpose event to the system via the PME_OUT pin.
	PM_TO_ACK	Terminate the PME_Turn_OFF message issued from the originating PCI Express port
	PM_ENTER_L1, PM_ENTER_L23 (DLLP)	These messages are issued by downstream components that indicate their entry into L1 or L2/L3 states. The Intel 5000X Chipset MCH must block subsequent TLP issue and wait for all pending TLPs to Ack. Then, send PM_REQUEST_ACK.
	ATTENTION_BUTTON_PRESSED	Terminate the message and set the PEXSLTSTS."Attention Button Pressed". If PEXCTRL."Attention Button Pressed Enable"and "Hot-Plug Interrupt Enable" bits are set, send MSI, Assert_HPGPE, or assert_intx on the ESI. Refer to the Hotplug flow interrupt generation in the Intel® 5000 Series Chipset Programming guide.
	ASSERT_GPE	Send the received "Assert_GPE" message at the PCI Express port to the ESI port as a virtual wire using a wired-OR approach. NOTE: This is an Intel vendor-specific message.
	DEASSERT_GPE	Send the received "Deassert_GPE" message at the PCI Express port to the ESI port as a virtual wire using a wired-OR approach. NOTE: This is an Intel vendor-specific message.
	FENCE_MSG	This message is used to provide ordering between different streams in the given PCI Express port
	CB Query	This message is sent by the TOE device to request information on the DMA Engine BAR and per port offset for configuration.
	Others	Set IO2 error (unsupported request), drop transaction (master abort) and return credit.

Table 5-19. Incoming PCI Express Completions

PCI Express Transaction	Address Space or Message	Intel 5000X Chipset MCH Response
Completions for Outbound Writes	I/O or Configuration[a]	Forward to the processor bus, PCI Express or ESI from which the request originated.
Completions for Outbound Reads	Memory, I/O or Configuration	Forward to the processor bus, PCI Express or ESI from which the request originated.

Notes:

a. Outbound Memory writes are posted and have no completions

Table 5-20. Outgoing PCI Express Requests

PCI Express Transaction	Address Space or Message	Reason for Issue
Outbound Write Requests	Memory	Processor bus or peer memory-mapped I/O write targeting PCI Express device.
	I/O	Processor legacy I/O write targeting PCI Express device.
	Configuration	Processor or peer configuration write targeting PCI Express device.
Outbound Read Requests	Memory	Processor or peer memory-mapped I/O read targeting PCI Express device.
	I/O	Processor or peer I/O read targeting PCI Express device.
	Configuration	Processor or peer configuration read targeting PCI Express device.
Outbound Messages	EOI (Intel-specific)	End-of-interrupt cycle received on processor bus, Intel 5000X Chipset MCH broadcasts this message to all active PCI Express ports. Devices supporting edge triggered interrupts will ignore this cycle.
	Lock/Unlock	When a locked read or write transaction was previously issued to a PCI bridge, “Unlock” releases the PCI lock.
	PM_TURN_OFF	PEXGCTRL.PME_TURN_OFF bit was set. This message is broadcast to all enabled PCI Express ports.
	PM_REQUEST_ACK (DLLP)	Received PM_ENTER_L1 and PM_ENTER_L23. This message is continuously issued until link is idle.
	Attention_Indicator_On	PEXSLOTCTRL.”Attention Indicator Control” has been set to On.
	Attention_Indicator_Off	PEXSLOTCTRL.”Attention Indicator Control” has been set to Off.
	Attention_Indicator_Blink	PEXSLOTCTRL.”Attention Indicator Control” has been set to Blink.
	Power_Indicator_On	PEXSLOTCTRL.”Power Indicator Control” has been set to On.
	Power_Indicator_Off	PEXSLOTCTRL.”Power Indicator Control” has been set to Off.
	Power_Indicator_Blink	PEXSLOTCTRL.”Power Indicator Control” has been set to Blink.
	CB Query Response	This message is sent back by the Intel 5000X Chipset MCH in response to the DMA Engine query and contains pertinent information on DMA Engine version, BAR offset and so forth.

Table 5-21. Outgoing PCI Express Completions

PCI Express Transaction	Address Space or Message	Reason for Issue
Inbound Read Completions	Memory	Response for an inbound read to main memory or a peer I/O device.
	I/O	Response for an inbound read to a peer I/O device.

5.18.3.1 Unsupported Messages

If the Intel 5000X Chipset MCH decodes any vendor message (which is not defined in Table 5-18), the MCH will take the following actions as specified in the Vendor defined message section of the *PCI Express Base Specification*, Revision 1.0a.

- Vendor Type 0 - Unsupported Request
- Vendor Type 1 - Drop request.

5.18.3.2 32/64 Bit Addressing

For inbound and outbound writes and reads, the MCH supports 64-bit address format. If an outbound transaction's address is a 32-bit address, the MCH will issue the transaction with a 32-bit addressing format on PCI Express. Only when the address requires more than 32 bits will the MCH initiate transactions with 64-bit addressing format. It is the responsibility of the software to ensure that the relevant bits are programmed for 64 bits based on the OS limits. (e.g 36 bits for MCH)

5.18.4 Transaction Descriptor

The *PCI Express Base Specification*, Revision 1.0a defines a field in the header called the Transaction Descriptor. This descriptor comprises three sub-fields:

- Transaction ID
- Attributes
- Virtual Channel ID

5.18.4.1 Transaction ID

The Transaction ID uniquely identifies every transaction in the system. The Transaction ID comprises four sub-fields described in Table 5-22.

Table 5-22. PCI Express Transaction ID Handling

Field	Definition	MCH as Requester	MCH as Completer	Peer-to-peer Transaction
Bus Number	Specifies the bus number that the requester resides on.	The MCH sets this field to 0.	The MCH preserves this field from the request and copies it to the completion.	For peer-to-peer posted requests, the MCH preserves these fields from the source PCI Express port to the destination port. For peer-to-peer non-posted requests, the MCH preserves Requestor ID Bus, Device, and Function, but reassigns the Tag.
Device Number	Specifies the device number of the requester.	The MCH fills this field in with the content of the DID register for this port.		
Function Number	Specifies the function number of the requester.	The MCH sets this field to 0.		
Tag	Identifies a unique identifier for every transaction that requires a completion. Since the PCI Express ordering rules allow read requests to pass other read requests, this field is used to reorder separate completions if they return from the target out-of-order.	The MCH fills this field in with a value such that every pending request carries a unique Tag.		

5.18.4.2 Attributes

PCI Express supports two attribute hints described in Table 5-23.

Table 5-23. PCI Express Attribute Handling

Attribute	Definition	MCH as Requester	MCH as Completer	Peer-to-peer Transaction
Relaxed Ordering	Allows the system to relax some of the standard PCI Express ordering rules.	For outbound transactions, this bit is not applicable and set to zero.	The MCH ignores this field on inbound transactions. The MCH makes no proactive attempts to reorder differently based on the value in this field.	For requests and completions, preserve this field from the source PCI Express port to the destination port.
Snoop Not Required	This attribute is set when an I/O device controls coherency through software mechanisms. This attribute is an optimization designed to preserve processor bus bandwidth.		If this attribute is set for inbound transactions, the MCH will not snoop the transaction on the processor buses.	

5.18.4.3 Traffic Class

The MCH does not support any PCI Express virtual channels other than the default channel (channel 0). Therefore, the MCH effectively ignores the traffic class field for inbound requests. For inbound completions, the MCH will copy the TC value received into the completion. For peer-to-peer completions, the TC value of the request is preserved.

For outbound requests, the MCH sets this ID to zero.

5.18.5 Transaction Behavior

This section describes the specifics of how PCI Express transactions flow through the MCH. This section covers both generic PCI Express transactions and ESI transactions.

5.18.5.1 Inbound Transactions

Inbound requests should be serviced to maximize PCI Express bandwidth for the given link without stalling. The MCH will accept the transactions listed in Table 5-22. This section describes handling that is specific to the MCH for transactions that target the MCH or main memory. Ordering rules for inbound transactions are described in Section 5.18.6.

5.18.5.1.1 Inbound Memory Reads

Read Completion Policy

For inbound read requests, the MCH is allowed to split completions along a Read Completion Boundary (RCB) of 64B, PEXLINKCTRL[7:2, 0]: PCI Express Link Control Register. For MCH, the maximum size of a read completion is specified with Max_Payload_Size field as 256B, PEXDEVCTRL[7:2, 0], PCI Express Device Control Register. If the PCI Express interface is idle, the MCH will return a completion for that read starting at the initial address up to the next cache line boundary.

If a PCI Express interface is busy sending an outbound packet, the MCH will opportunistically combine subsequent inbound read completions up to Max_Payload_Size or until the initial request length is satisfied. Note that completion combining helps increase bus efficiency due to reduced header overhead on the PCI Express port.

5.18.5.2 Inbound Read/Write Streaming

The MCH IOU cluster implements extensive pipe-lining and non-speculative prefetching to maximize throughput and utilization of the various PCI Express interconnects. The IOU uses two phases to handle a transaction. A transaction cycle starts with a “prefetch” followed by a “fetch” cycle and terminates with a fetch completion.

1. Prefetch Phase: The IOU launches multiple cache lines as non-speculative prefetches for one or more enqueued requests. These prefetch commands are routed to the Coherency Engine (CE)/ Data Manager (DM) for decoding and then sent to memory/FSB (for snoops) if required.
2. Fetch Phase: Request data from cache lines in the CE/DM for sending to destination port. A Fetch completion is sent to terminate the cycle and remove buffer entries. In some cases, a fetch request can be issued without a prefetch for optimization (for example, an initial read to memory when completion buffers are empty).

In the prefetch phase, the streaming logic in the IOU breaks inbound transactions (in the order received) into one or more cache lines and pipelines them to the CE. It should send enough requests up to the total number of outstanding cache lines that the MCH can handle to maximize bandwidth on the PCI Express port. When acknowledgement for the individual cache lines in the prefetch phase is returned, the fetch phase begins immediately and internal commands are pipelined to the CE/DM to obtain the data as fast as possible. Finally the data is packaged into appropriate TLPs (for example, read completions) and returned to the PCI Express interface based on the request arrival order and the appropriate completion combining conditions prevailing for that port.

5.18.5.3 Zero-Length Reads/Writes

The *PCI Express Base Specification*, Revision 1.0a describes that a zero-length memory read must be supported and may be used by devices as a queue flushing mechanism. With the PCI Express ordering rules, a device can issue such a read to push ahead all previously issued writes.

When the MCH receives a zero-length read, data is not actually read from anywhere. Rather, the read is completed by the MCH after all writes previously posted on that inbound port are considered to be globally visible by the system. At that point, the MCH will return one DW of zeroes to the requesting PCI Express port.

The *PCI Express Base Specification*, Revision 1.0a also does not preclude the arrival of zero length writes on any of the PCI Express ports. For coherence compatibility and general software usage expectation, it is required to perform an RFO (Request For Ownership) for the cache line involved in the zero length write and then commit the unmodified cache line to memory. Similarly on the outbound path, the MCH will forward zero length reads and writes to the respective destinations.

5.18.5.4 Inbound Write Transactions

Inbound coherent write transactions actually comprise two operations: request for ownership, and the cache line write (mark to modified state). The PCI Express unit will enqueue each inbound write as a single atomic instruction. As the PCI Express unit enqueues the write, it will also bypass all queues by sending a request-for-ownership command (RFO) directly to the processor buses requesting for line ownership. The RFO commands are allowed to be issued in any order.

If the MCH owns the line after all inbound ordering rules have been met, the write command proceeds and the line is modified. If the line is not owned by the MCH when after all inbound ordering rules have been met, the write is temporarily stalled until

ownership is acquired and requests will continue to fill the inbound queue. When the request for ownership completes, the write command is forwarded where the line is marked in the modified state.

Note: Because of the PCI Express ordering rules, transactions after an inbound write must wait until after the write is globally visible. The inbound queue must be deep enough such that continuos inbound traffic is not stalled waiting for the above write sequence even under heavily leaded conditions.

For write transactions with the Don't Snoop attribute, the PCI Express unit (following all inbound ordering rules) will simply issue a write command to memory without snooping the processor buses. Note that ordering is required between normal and "Don't Snoop" transactions.Inbound

5.18.5.5 Inbound Write Combining

The MCH performs write combining opportunistically in the on-chip coherent data buffer.

5.18.5.6 Interrupt Handling

A PCI Express device represents interrupts with either MSI or inbound interrupt messages (ASSERT_INTx/DEASSERT_INTx).

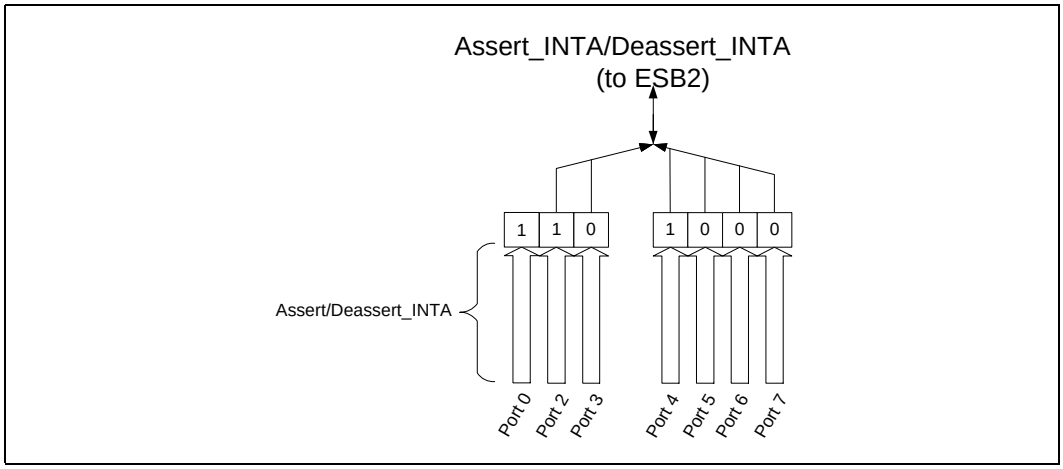
Each PCI Express port of the MCH is responsible for tracking assert/deassert messages for each of the four interrupts (INTA, INTB, INTC, INTD) and representing them with four output "virtual (ASSERT_INTA, ASSERT_INTB, ASSERT_INTC, and ASSERT_INTD) messages to the Intel® 631xESB/632xESB I/O Controller Hub.

Figure 5-28 illustrates how the PCI Express interrupts are routed to the Intel® 631xESB/632xESB I/O Controller Hub (ESB2). The example shown represents Interrupt A and this logic is replicated for the four interrupts. The bits depicted are software visible.

When a PCI Express assert message is received for a specific interrupt, another assert message will not arrive until after a deassert message has arrived for that interrupt first.

When MSI interrupts are used, the MCH treats these writes as any other inbound write. The difference is that MSI writes are detected as a write to addresses in the range FEE0 0000h - FEDF FFFFh. If the write falls within this range, the MCH issues the write to both processor buses where it will be claimed by the targeted CPU.

Figure 5-28. Legacy Interrupt Routing (INTA Example)



5.18.5.7 Error Messages

PCI Express reports many error conditions through explicit error messages: ERR_COR, ERR_UNC, ERR_FATAL.

5.18.5.8 Inbound Vendor-Specific Messages

ASSERT_GPE / DEASSERT_GPE

ASSERT_GPE and DEASSERT_GPE form a virtual wire which is sent by a PCI Express device. The Intel® 631xESB/632xESB I/O Controller Hub component issues these messages as a response to its integrated PCI Express Hot-Plug controllers.

5.18.5.9 Outbound Transactions

The MCH will generate the outbound transactions listed in Table 5-20. This section describes handling that is specific to the MCH for transactions that target a PCI Express interface. Ordering rules for outbound transactions are described in Table 5.18.6.2.

5.18.5.10 Outbound Non-Posted Transactions

Non-posted transactions that the MCH supports includes memory reads, I/O reads and writes, and configuration reads and writes. When a non-posted transaction is issued from the MCH, the PCI Express device will respond with a completion.

Each PCI Express interface supports up to four outstanding non-posted transactions comprising transactions issued by the processors or a peer PCI Express device.

5.18.5.10.1 Stalled Non-Posted Requests

Non-posted requests are non-blocking transactions. In other words, while a non-posted request is pending, subsequent transactions are required to bypass the transactions which are waiting for a completion.

5.18.5.10.2 Outbound Posted Transactions

Once a posted request (memory mapped I/O write) is issued from the PCI Express's transaction layer, the request is considered to be complete and the transaction is removed from the outbound queue. For posted requests, the acknowledge has already

been sent to the initiating interface (processor bus or alternate PCI Express inbound queue) when the write was enqueued in the outbound PCI Express unit so proper ordering is guaranteed.

5.18.5.11 Outbound Vendor-Specific Messages

The MCH supports only vendor-specific EOI messages outbound.

5.18.5.11.1 EOI Message

EOI messages will be broadcast to all the PCI Express interfaces and require posted transaction ordering. This ensures that the appropriate interrupt controller receives the end-of-interrupt. Depending on outbound traffic patterns, the EOIs will often be delivered on the PCI Express ports at different times.

EOI is a message required for I/OAPICs which support XAPIC. Since EOI is Intel-specific, this PCI Express message can only be forwarded to Intel devices that support an integrated I/OAPIC supporting level-sensitive interrupts (Intel® 631xESB/632xESB I/O Controller Hub).

5.18.5.12 Lock Support

For legacy PCI functionality, the MCH supports bus locks through an explicit sequence of events. The MCH can receive a locked transaction sequence (Read-Write or Read-Read-Write-Write) on a processor interface directed to a PCI Express-to-PCI bridge.

Note that native PCI Express devices are prohibited from supporting bus locks according to the *PCI Express Base Specification*, Revision 1.0a.

The PCI Express interface cluster must support the following capabilities:

- Block all transactions on the PCI Express ports (when the lock targets another port)
- Generate a locked read request to the target PCI Express port
- Unlock the locked PCI Express port

The Intel 5000X Chipset MCH lock flow will not complete if a memory read is sent to a native PCI Express endpoint, which will return an unsupported request (UR) status response. Operation is not guaranteed, if a lock to a native PCI Express device is issued. Locks to PCI through PCI Express will still be supported.

5.18.5.13 Peer-to-Peer Transactions

Peer-to-peer support is defined as transactions which initiate on one I/O interface and target another without going through main memory. The MCH supports peer-to-peer transactions only for memory and I/O transactions. Any PCI Express interface can be the source or destination of a peer-to-peer transaction. Peer-to-peer transactions are not observed on any interface except the target and destination (they are not snooped by the processors).

Peer-to-peer traffic between the PCI Express ports must sustain the full write bandwidth (depending on the PCI Express port configuration). For Peer-to-Peer reads, the delivered bandwidth is a function of the round trip read completion latency.

Inbound coherent transactions and peer-to-peer transactions must maintain PCI Express ordering rules between each other. Peer-to-peer transactions follow inbound ordering rules until they reach the head of the inbound queue. Once the transaction reaches the head of the inbound queue, the MCH routes the transaction to the next

available slot in the outbound queue where PCI ordering is maintained until the end of the transaction. The MCH does not support peer-to-peer transactions where the source and destination are on the same PCI Express interface.

5.18.5.14 Peer-to-Peer Configuration Cycles

The MCH supports peer-to-peer PCI Express configuration cycles through Type0 and Type1 configuration requests. For this feature to be enabled, the configuration register bit PEXCTRL.DIS_INB_P2PCFG, PEXCTRL[7:2, 0], PCI Express Control Register, for the respective ports has to be cleared. If an inbound peer-to-peer configuration cycle is received by the MCH, the bus number, device number, function number and register number are compared.

If the resulting comparison is such that the bus number does not fall in the range of the primary to subordinate bus number register, the configuration cycle is completed with an Unsupported Request completion status.

If the comparison targets the bus indicated by a secondary bus register, then the MCH translates the configuration cycle into a Type 0.

If the comparison targets the bus such that (secondary bus < bus_number <= subordinate_bus), then the MCH translates the configuration cycle into a Type 1.

5.18.6 Ordering Rules

This section describes the MCH ordering rules for transactions progressing through the PCI Express unit.

5.18.6.1 Inbound Transaction Ordering Rules

Inbound transactions originate from PCI Express and target main memory. In general, the PCI Express cluster holds inbound transactions in FIFO order. There are exceptions to this order under certain situations. For example, PCI Express requires that read completions are allowed to pass read requests. This forces any read completions to bypass any reads which might be back pressured in the queue. The PCI Express ports have no ordering relationship to each other (aside from the peer-to-peer restrictions below).

Sequential non-posted requests are not required to be completed in the order they were requested. However, if a non-posted request requires multiple sub-completions (typically due to splitting a memory read into cache line requests), then those sub-completions must be delivered in order.

Inbound writes cannot be posted beyond the PCI Express domain and outbound writes may only be posted after the write is acknowledged by the destination PCI Express cluster. The posting of writes relies on the fact that the system maintains a certain ordering relationship. Since the MCH cannot post inbound writes beyond the PCI Express cluster, the MCH must wait for snoop responses before issuing subsequent, order-dependent transactions.

5.18.6.1.1 Inbound Ordering Requirements

In general, there are no ordering requirements between transactions issued on the different PCI- Express interfaces. The following rules apply to inbound transactions issued on the same interface.

The following rules must be ensured for inbound transactions:

Rule 1. Outbound non-posted read and write completions must be allowed to progress past stalled inbound non-posted requests.

RULE3: Inbound posted write requests must be allowed to progress past stalled inbound non-posted requests.

RULE4: Inbound posted write requests, inbound read requests, outbound non-posted read and write completions cannot pass enqueued inbound posted write requests.

The Producer - Consumer model prevents read requests, write requests, and non-posted read or write completions from passing write requests. Refer to *PCI Local Bus Specification*, Revision 2.3 for details on the Producer - Consumer ordering model.

RULE5: Outbound non-posted read or write completions must push ahead *all* prior inbound posted write transactions from that PCI Express port.

RULE6: To optimize performance, Inbound, coherent, posted writes will issue ownership requests (RFO) without waiting for prior ownership requests to complete.

RULE7: Inbound messages follow the same ordering rules as inbound posted writes.

Inbound messages are listed in Table 5-18. Similarly to inbound posted writes, reads should push these commands ahead.

The above rules apply whether the transaction is coherent or non-coherent. Some regions of memory space are considered non-coherent (Don't Snoop attribute is set). The MCH PCI Express cluster will order all transactions regardless of its destination.

5.18.6.2 Outbound Transaction Ordering Rules

Outbound transactions through the MCH are memory, I/O, or configuration read/write transactions originating on a processor interface destined for a PCI Express device. Multiple transactions destined for the same PCI Express port are ordered according to the ordering rules specified in *PCI Express Base Specification*, Revision 1.0a.

5.18.6.2.1 Outbound Ordering Requirements

There are no ordering requirements between outbound transactions targeting different PCI Express interfaces. For deadlock avoidance, the following rules must be ensured for outbound transactions within the same PCI Express interface:

Rule 1. Inbound non-posted completions must be allowed to progress past stalled outbound non-posted requests.

RULE8: Outbound posted write requests must be allowed to progress past stalled outbound non-posted requests.

RULE9: Outbound non-posted requests, outbound messages, outbound write requests, and inbound completions cannot pass enqueued outbound posted write requests.

The Producer - Consumer model prevents read requests, write requests, and read completions from passing write requests. Refer to *PCI Local Bus Specification*, Revision 2.3 for details on the Producer - Consumer ordering model.

RULE10: Posted outbound messages must follow the same ordering rules as outbound posted writes.

RULE11: If a non-posted inbound request requires multiple sub-completions, then those sub-completions must be delivered in linearly increasing address order.

5.18.6.3 MCH Ordering Implementation

The following table summarizes the rules enforced on transactions from a given PCI Express port by the MCH.

Table 5-24. MCH Ordering Implementation

Transaction	Will the transaction pass a stalled Posted Request?	Will the transaction pass a stalled Non-Posted Request?	Will the transaction pass a stalled completion?
Posted requests	never	always	always
Non-Posted Requests	never	Can happen in implementation; no architectural ordering requirement is imposed	always
Completions	never	always	never

5.18.6.3.1 Peer-to-Peer Ordering

All peer-to-peer memory write transactions are treated as non-coherent memory writes by the system. Peer-to-peer memory reads are treated as non-coherent reads.

On the MCH, any peer-to-peer transaction is ordered with other inbound transactions from the same PCI Express port. This provides a serialization point for proper ordering (for example, cases where the flag and data are not in the same memory).

When the PCI Express interface receives a peer-to-peer memory write command, inbound ordering rules require that it must wait until all prior inbound writes are globally visible. The peer-to-peer write completes when the target PCI Express port receives the transaction in its ordered domain. The acknowledge must return to the source PCI Express unit quickly enough to allow the PCI Express device to post further peer-to-peer memory writes without any stalls on the interface.

Peer-to-peer memory write transactions are considered posted with regards to ordering. Peer to peer read transactions are non-posted. Peer-to-peer transactions must adhere to the ordering rules listed in Table 5.18.5.13 and Table 5.18.5.14.

5.18.6.4 Interrupt Ordering Rules

With MSI, SAPIC and Expiate, interrupts are simply inbound non-coherent writes to the processor. With legacy interrupts, the interrupts are ASSERT and DEASSERT messages (also following posted write ordering rules). This enforces that the interrupt will not be observed until all prior inbound writes are flushed to their destinations. However, the MCH does not guarantee that the interrupt will be observed by the processor before subsequent writes are visible to a processor.

5.18.6.4.1 EOI Ordering

When a processor receives an interrupt, it will process the interrupt routine. The processor will then proceed to clear the I/O card's interrupt register by writing to that I/O device. In addition, for level-triggered interrupts, the processor sends an End-of-Interrupt (EOI) special cycle (8 bit interrupt vector on D[7:0]# of the processor's data bus) to an IOAPIC controller south of MCH. This EOI cycle must be treated as an outbound write with regard to ordering rules. This ensures that the EOI will not pass any prior outbound writes. If the EOI passes the prior write to clear the register, then the IOAPIC controller could mistakenly signal a second interrupt since the register clear had not occurred yet.

5.18.7 Prefetching Policies

The MCH does not perform any speculative prefetching for PCI Express interface component reads. The PCI Express component south of the Intel 5000X Chipset MCH is solely responsible for its own prefetch algorithms since those components are best suited to know what tradeoffs are appropriate.

The MCH does not perform any outbound read prefetching.

5.18.8 PCI Express Hide Bit

To "hide" PCI Express ports from the OS, there is one PEXCTRL.CLASSCTRL bit for each port. When firmware sets the write-once PEXCTRL.CLASSCTRL bit, the PCI Express device in the MCH changes its response to a CCR read (Class Code Register, 0x0600h) so that it appears to be a host bridge. The OS will not perform configuration reads and writes to a device with its hide bit set.

5.18.9 No Isochronous Support

The Intel 5000X Chipset MCH does not support isochrony. Only the default virtual channel (channel 0) is supported on the PCI Express interfaces.

5.18.10 PCI Express Hot-Plug

PCI Express native Hot-Plug allows for higher availability and serviceability of a server. It gives the user the capability of adding, removing, or swapping out a PCI Express slot device without taking down the system. The user and system communicate through a combination of software and hardware utilizing notification through mechanical means and indicator lights.

Each Intel 5000X Chipset MCH PCI Express port supports the optional Hot-Plug capability described in *PCI Express Base Specification*, Revision 1.0a. The PCI Express Hot-Plug model implies a Hot-Plug controller per port which is identified to software as a capability of the peer-to-peer bridge configuration space.

PCI Express Hot-Plug support requires that the MCH supports a set of Hot-Plug messages, listed in Table 5-18 and Table 5-22, to manage the states between the Hot-Plug controller and the device.

The PCI Express form factor has an impact on the level of support required from the MCH. For example, some of the Hot-Plug messages are required only if the LED indicators reside on the actual card and are accessed through the endpoint device. The MCH supports all of the Hot-Plug messages.

5.18.11 PCI Express RAS Features

The PCI Express interfaces will have traditional CRC protection. The data packets will utilize a 32-bit CRC protection scheme. The smaller and less error-prone link packets will utilize a 16-bit CRC scheme. Since packets utilize 8B/10B encoding, and not all the encodings are used; this provides further data protection, as illegal codes are also detected. If errors are detected on received data packets, these data packets are retransmitted. Hardware logic will support this link-level retry without software intervention. If a link lane fails, the link can reconfigure itself during retraining to a link of smaller width to continue operation at a lower performance level.

5.18.11.1 PCI Express Retry

The PCI Express interface will incorporate a link level retry mechanism. The hardware will detect when a transmission packet is corrupted and a retry of that particular packet and all following packets will be performed. Although this will cause a temporary interruption in the delivery of packets, it does so in order to maintain the link integrity.

5.18.11.2 PCI Express Recovery

When enough errors occur, the hardware may determine that the quality of the connection is in question, and the end points can enter what amounts to a quick training sequence known as recovery. The width of the connection will not be renegotiated, but the adjustment of skew between lanes of the link may occur. This occurs without any software intervention, but the software may be notified.

5.18.11.3 PCI Express Retrain

If the hardware is unable to perform a successful recovery, then the software is notified of the link condition, and software will invoke a hardware retrain. Once software has to be involved, then data will probably be lost, and processes will needed to be restarted, but this is still better than having to take the system down, or offline for an extended period of time.

5.18.12 Unsupported Transactions and Unexpected Completions

If the MCH receives a packet that is not included in Table 5-18, then the Intel 5000X Chipset MCH treats that packet as an unsupported transaction. If the transaction is non-posted (determined after fully decoding the header), then the completion is issued by the MCH with an Unsupported Request status. If the transaction is posted, it is dropped. The following errors are also treated as Unsupported Transactions

- Lock to non-legacy device
- Illegal configuration accesses

5.18.12.1 Error Actions:

- **Receiving an UnSupported Request**
  If Non Posted Request then set UR Status in response;
  Set PEXSTS.RMA, PEXDEVSTS.URD
  Set UNCERRSTS.IO2 and assert error signal per the PEXDEVCTRL and the PEX_ERR_DOCMD register.

- **Receiving a supported request type but cannot complete**
  If Non Posted Request then send Completer Abort in response.
  Set PEXSTS.RTA
  Set UNCERRSTS.IO7 and assert error signal per the PEXDEVCTRL and the PEX_ERR_DOCMD register.

5.18.13 ECRC

The *PCI Express Base Specification*, Revision 1.0a supports an optional end-to-end CRC mechanism (ECRC). The Intel 5000X Chipset Intel 5000X Chipset MCH does not support this feature.

5.18.14 EDB

The *PCI Express Base Specification*, Revision 1.0a supports a mechanism to signal a bad packet (End Bad or EDB). The MCH can receive a packet marked with as EDB and will correctly identify the packet as “bad”, but the MCH will never generate a EDB packet.

5.18.15 Error Forwarding

PCI Express has a concept called Error Forwarding or Data Poisoning. This feature allows a PCI Express device to forward data errors across the interface without it being interpreted as an error originating on that interface.

When the MCH is the initiator of a transaction with data (write or read completion), the MCH poisons the transaction if the data contains an internal ECC error. Poisoning is accomplished by the MCH setting the EP bit in the PCI Express packet header.

When the MCH is the recipient of a transaction with data and the data is poisoned (refer to *PCI Express Base Specification*, Revision 1.0a for the various mechanisms) then the PCI Express unit poisons the data before forwarding internally to the destination interface.

5.18.16 Unconnected Ports

If a non-posted transaction targets a PCI Express interface that is not connected to any device, the MCH behaves as it would if it had received an Unsupported Request completion (master abort). The MCH should complete any reads initiated on the Intel 5000X Chipset MCH (from the processor bus, or JTAG, SMBus) to an unconnected PCI Express port with all ones in the data. If the transaction is posted, the transaction is dropped and the signal is driven.

Note: Design must comprehend the difference between L2 state and an unconnected port. In order to exit L2, a configuration cycle will target the port and must not trigger the above mechanism.

5.18.17 PCI Express Power Management Support

The *PCI Express Base Specification*, Rev 1.0a requires that root complexes enable an end point to enter low power states. PCI Express power management happens in two phases:

- The PCI Express link transitions from L0 (active state) to L1 (low power)
- The PCI Express link transitions from L1 (low power) to L2 (end point power off) through L0. that is, L0 --> L1 --> L0 --> L2 and so forth.

While it is not required that the Intel 5000X Chipset MCH enter low power states, the end point also participates in a handshake protocol to wake up a sleeping system. The Intel 5000X Chipset MCH must honor this protocol which is described further below.

Note: The Intel 5000X Chipset MCH will not initiate a PME event but will forward PME messages to wake the system due to a Hot-Plug event on the PCI Express interface, for example.

5.18.17.1 L0s

The Intel 5000X Chipset MCH does not support Active State PM transition (ASPM) into L0s state (i.e Intel 5000X Chipset MCH does not power down link into a “standby” mode and will not transmit electrical_idle ordered sets) due to latency/performance issues that degrade performance while presenting minimal power savings. However, it does tolerate an end PCI Express device from placing the link into an L0s state. The lack of L0s support in the Intel 5000X Chipset MCH is a deviant from the base *PCI Express Base Specification*, Rev 1.0a as it simplifies power control and onboard logic circuitry.

Note: The Intel 5000X Chipset MCH has recently added a mechanism through the PEXDLLPCTRL.GBLFCUT timer to prevent entry into L0s[1] by injecting FC credit DLLPs at less than 7us interval.

5.18.17.2 L1 Entry

Refer to the *PCI Express Base Specification*, Rev 1.0a for details. This section describes at a high level what is required of the Intel 5000X Chipset MCH.

The *PCI Express Base Specification*, Rev 1.0a lists two mechanisms for entering L1 state:

- Active State Power Management
- Software Initiated Power Management

Note: The Intel 5000X Chipset MCH fully supports Software Initiated Power Management but does not initiate Active State Power Management. The L1 state is initiated with a configuration write to the end device being put into the low power state. The configuration write is executed by software writing to the “Power State” bits[1:0] of the PMCSR register of the PCI-PM capability structure in the end device such that the end point goes into the D1 state. In response to this configuration write, the end point will block subsequent TLPs and wait until all pending TLPs have Acked. When this condition is met, the end point will issue a PM_ENTER_L1 DLLP to the Intel 5000X Chipset MCH to put the link in L1 mode. The Intel 5000X Chipset MCH responds to this message by blocking outbound TLPs and waiting for all “in-flight” TLPs to get acknowledged. After all pending TLPs have been ACKed, the Intel 5000X Chipset MCH issues a PM_REQUEST_ACK DLLP continuously until the link is observed to be electrically idle. The link is now in the L1 state.

5.18.17.3 L1 Exit

Exit from L1 brings the link back into the active (L0) state. When the Intel 5000X Chipset MCH receives a configuration write to a link that is in L1, the link is trained again. This is done by configuration software setting the Retrain Link bit in the PEXLNKCAP register. When the link is retrained and is reactivated in the L0 state, the Intel 5000X Chipset MCH will forward the configuration write to the end point to perform a similar link retraining.

5.18.17.4 L2/L3 Ready State Entry

The L2/L3 Ready state is entered when software wants to remove power from a device. If the end point continues to operate under Vaux power, its ending state will be L2. If the end point truly powers down, its ending state will be L3. The L2/L3 Ready state is initiated with a configuration write to PEXGCTRL.PME_Turn_Off (See Section 21.8.9.34, “PEXGCTRL: PCI Express Global Control Register” on page 654). When this bit is set, the Intel 5000X Chipset MCH broadcasts a PM_TURNOFF message to all populated PCI Express ports. The Intel 5000X Chipset MCH waits for a PM_TO_ACK message from each of the ports that were sent the PM_TURNOFF message.

After the endpoint sends a PM_TO_ACK, it will initiate L2/L3 Ready state entry with a PM_ENTER_L2 DLLP. From this point on, the link handshake follows the L1 Entry flow. Once all of the PM_TO_ACK messages are received, the Intel 5000X Chipset MCH sets the PEXGCTRL.PME_TO_Ack. Software can poll the register PEXGCTRL.PME_TO_Ack register field to determine when all the PM_TO_ACK messages have arrived, take appropriate action and reset the field. When PEXGCTRL.PME_TO_Ack is set, the Intel 5000X Chipset MCH will send an Assert_PMEGPE message to the Intel® 631xESB/632xESB I/O Controller Hub. This causes an interrupt and the software takes appropriate action to inform the power controller. When software services this interrupt it clears PEXGCTRL.PME_TO_Ack, and this causes the Intel 5000X Chipset MCH to send a deassert_PMEGPE message on the ESI port.

5.18.17.5 Wake Sequence

The *PCI Express Base Specification*, Rev 1.0a allows an endpoint to wake up a system that is asleep (in L2). This sequence is split into two elements:

- Waking the system (powering it up)
- Alerting the system that a device has awakened

To power the system and devices, Intel 5000X Chipset platforms require the side-band WAKE# signal. The inband tone mechanism is not supported. The WAKE# signal is driven by the endpoint directly to on-board logic which powers the device(s) up. The L2 state implies that power is off to the devices and therefore, exit from this state implies that power is reapplied and the links would simply retrain according to the platform reset requirements.

After the links are trained, the waking device issues a PM_PME message to the Intel 5000X Chipset MCH. This message carries the DeviceID of the initiator which is logged in the PEXRTSTS register. The Intel 5000X Chipset MCH uses the PMESTATUS bit in the PEXRTSTS register to reflect the PME state for each of the PCI Express ports. The Intel 5000X Chipset MCH OR’s all the bits together and sends the Assert_PMEGPE message to the Intel® 631xESB/632xESB I/O Controller Hub. When all the bits are clear, the Deassert_PMEGPE message is sent.

Note: The Intel 5000X Chipset MCH can handle two outstanding PM_PME messages in its internal queues of the Power Management Controller per port. If the downstream device issues more than 2 PM_PME messages successively, it will be dropped.

Note: If the Intel 5000X Chipset MCH receives a pending request to a port that is in L2 (for example, an outbound transaction), it will be a master abort and the respective error logged.

5.19 Power Management

The Intel 5000X Chipset MCH power management support includes:

- ACPI supported
- System States: S0, S1 (desktop), S3, S4, S5, C0, C1, C2 (desktop)

5.19.1 Supported ACPI States

The MCH supports the following ACPI States:

- Processor
  - C0: Full On.
  - C1: Auto Halt.
  - C2 Desktop: Stop Grant. Clock to processor still running. Clock stopped to processor core.
- System
  - G0/S0: Full On.
  - G1/S1: Stop Grant, Desktop S1, same as C2.
  - G1/S2: Not supported.
  - G1/S3: Suspend to RAM (STR). Power and context lost to chipset.
  - G1/S4: Suspend to Disk (STD). All power lost (except wake-up logic on Intel® 631xESB/632xESB I/O Controller Hub).
  - G2/S5: Soft off. Requires total system reboot.
  - G3: Mechanical Off. All power lost (except real time clock).

5.19.2 FB-DIMM Thermal Management

The Intel 5000X Chipset MCH implements the following thermal management mechanisms. These mechanisms manage the read and write cycles of the system memory interface to implement thermal throttling.

Hardware-Based Thermal Management

The number of hex-words transferred over the DRAM interface are tracked per row. The tracking mechanism takes into account that the DRAM devices consume different levels of power based on cycle type (page hit/miss/empty). If the programmed threshold is exceeded during a monitoring window, the activity on the DRAM interface is reduced. This helps in lowering the power and temperature.

Software-Based Thermal Management

This is used when the external thermal sensor in the system interrupts the processor to engage a software routine for thermal management.

5.19.3 FB-DIMM Thermal Diode Overview

The FB-DIMM Advanced Memory Buffer (AMB) contains an internal thermal diode to measure AMB / DIMM temperature. Upon detecting a thermal over temperature condition the AMB initiates a thermal throttling event. For more information see the *Gold Bridge Component External Design Specification*.

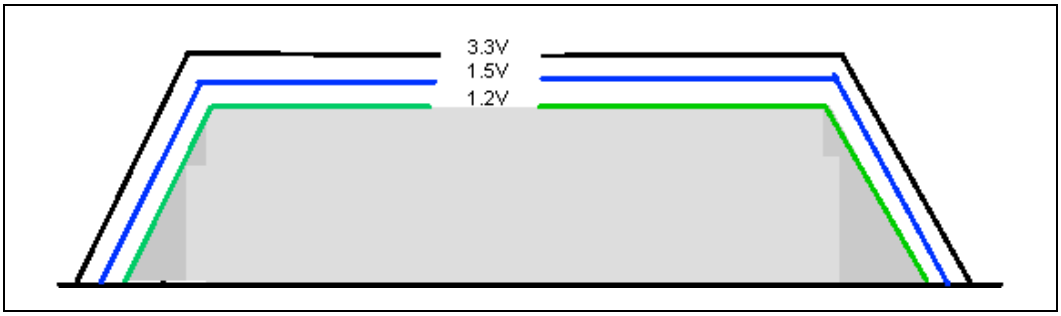
5.20 System Reset

The Intel 5000X Chipset MCH is the root of the I/O subsystem tree, and is therefore responsible for general propagation of system reset throughout the platform. The MCH must also facilitate any specialized synchronization of reset mechanisms required by the various system components.

5.20.1 MCH Power Sequencing

General power sequencing requirements for the Intel 5000X Chipset MCH are simple. In general higher voltages must come up before lower voltages. Figure 5-29 depicts the sequencing of the three main voltages powering the Intel 5000X Chipset MCH.

Figure 5-29. Intel 5000X Chipset Power Sequencing



Note: Power-up -> 3.3V must ramp ahead and stay above 1.5V, which must ramp ahead and stay above 1.2V. 3.3V must always be at least 0.7V greater than 1.5V. Duration of the power ramp must be between 0.1 ms and 100 ms.

5.20.2 MCH Reset Types

The Intel 5000X Chipset MCH differentiates among five types of reset as defined in table Table 5-25.

Table 5-25. MCH Reset Classes (Sheet 1 of 2)

Type	Mechanism	Effect / Description
Power-Good	PwrGd Input Pin	Propagated throughout the system hierarchy. Resets all logic and state machines, and initializes all registers to their default states (sticky and non-sticky). Tri-states all MCH outputs, or drives them to “safe” levels.
Hard	**RSTIN**# Input Pin, Configuration Write	Propagated throughout the system hierarchy. Resets all logic and state machines, and initializes all non-sticky registers to their default states. Tri-states all MCH outputs, or drives them to “safe” levels.
Processor-only	Configuration Write	Propagated to all processors via the FSBxRERSET# pins on the FSB. The MCH does not undergo an internal reset.

Table 5-25. MCH Reset Classes (Sheet 2 of 2)

Type	Mechanism	Effect / Description
Targeted	Configuration Write	Propagated down the targeted PCI Express port hierarchy. Treated as a "Hard" reset by all affected components, clearing all machine state and non-sticky configuration registers.
BINIT#	Internal Error Handling Propagated via FSB **BINIT**# pin	Propagated to all FSB attached components (the MCH and up to two processors). Clears the IOQ, and resets all FSB arbiters and state machines to their default states. Not recoverable.

5.20.2.1 Power-Good Mechanism

The initial boot of a Intel 5000X Chipset MCH platform is facilitated by the Power-Good mechanism. The voltage sources from all platform power supplies are routed to a system component which tracks them as they ramp-up, asserting the platform "PwrGd" signal a fixed interval (nominally 2mS) after the last voltage reference has stabilized. There are no requirements within the MCH regarding the precise sequencing of power-supply ramps, thus the platform should initialize properly regardless of the order in which supplies stabilize.

Both the Intel 5000X Chipset MCH and the Intel® 631xESB/632xESB I/O Controller Hub receive the system PwrGd signal via dedicated pins as an asynchronous input, meaning that there is no assumed relationship between the assertion or deassertion of PwrGd and any system reference clock. When PwrGd is deasserted all platform subsystems are held in their reset state. This is accomplished by various mechanisms on each of the different interfaces. The MCH will hold itself in a power-on reset state when PwrGd is deasserted. The Intel® 631xESB/632xESB I/O Controller Hub is expected to assert its PCIRST# output and maintain its assertion for 1mS after power is good. The PCIRST# output from Intel® 631xESB/632xESB I/O Controller Hub is expected to drive the RSTIN# input pin on the Intel 5000X Chipset MCH, which will in turn hold the processor complex in reset via assertion of the FSBxRESET# FSB signals.

The PCI Express attached devices and any hierarchy of components underneath them are held in reset via implicit messaging across the PCI Express interface. The MCH is the root of the hierarchy, and will not engage in link training until power is good and the internal "hard" reset has deasserted.

A PwrGd reset will clear all internal state machines and logic, and initialize all registers to their default states, including "sticky" error status bits that are persistent through all other reset classes. To eliminate potential system reliability problems, all devices are also required to either tri-state their outputs or to drive them to "safe" levels during a power-on reset.

The only system information that will "survive" a PwrGd reset is either contained in battery-backed or non-volatile storage.

5.20.2.2 Hard Reset Mechanism

Once the Intel 5000X Chipset MCH platform has been booted and configured, a full system reset may still be required to recover from system error conditions related to various device or subsystem failures. The "hard" reset mechanism is provided to accomplish this recovery without clearing the "sticky" error status bits useful to track down the cause of system reboot.

A hard reset is typically initiated by the Intel® 631xESB/632xESB I/O Controller Hub component via the PCIRST# output pin, which is commonly connected directly to the Intel 5000X Chipset MCH RSTIN# input pin. The Intel® 631xESB/632xESB I/O

Controller Hub may be caused to assert PCIRST# via both software and hardware mechanisms. The Intel 5000X Chipset MCH will recognize a hard reset any time RSTIN# is asserted while PwrGd remains asserted.

The Intel 5000X Chipset MCH will propagate a hard reset to the FSB and to all subordinate PCI Express subsystems. The FSB components are reset via the FSBxRESET# signals, while the PCI Express subsystems are reset implicitly when the root port links are taken down.

A hard reset will clear all internal state machines and logic, and initialize all “non-sticky” registers to their default states. Note that although the error registers will remain intact to facilitate root-cause of the hard reset, the Intel 5000X Chipset MCH platform in general will require a full configuration and initialization sequence to be brought back on-line.

5.20.2.3 Processor-Only Reset Mechanism

For power management and other reasons, the Intel 5000X Chipset MCH supports a targeted processor only reset semantic. This mechanism was added to the platform architecture to eliminate double-reset to the system when reset-signaled processor information (such as clock gearing selection) must be updated during initialization bringing the system back to the S0 state after power had been removed from the processor complex.

5.20.3 Targeted Reset Mechanism

The targeted reset is provided for Hot-Plug events, as well as for port-specific error handling under Machine Check Architecture (MCA) or SMI software control. The former usage model is new with PCI Express technology, and the reader is referred to the *PCI Express Interface Specification, Rev 1.0a* for a description of the Hot-Plug mechanism.

A targeted reset may be requested by setting bit 6 (Secondary Bus Reset) of the Bridge Control Register (offset 3Eh) in the target root port device. This reset will be identical to a general hard reset from the perspective of the destination PCI Express device; it will not be differentiated at the next level down the hierarchy. Sticky error status will survive in the destination device, but software will be required to fully configure the port and all attached devices once reset and error interrogation have completed. After clearing bit 6, software may determine when the downstream targeted reset has effectively completed by monitoring the state of bit 1 (Link Active) of the VS_STS1 register (offset 47h) in the target root port device. This bit will remain deasserted until the link has regained “link up” status, which implies that the downstream device has completed any internal and downstream resets, and successfully completed a full training sequence.

Under normal operating conditions it should not be necessary to initiate targeted resets to downstream devices, but the mechanism is provided to recover from combinations of fatal and uncorrectable errors which compromise continued link operation.

5.20.4 BINIT# Mechanism

The BINIT# mechanism is provided to facilitate processor handling of system errors which result in a hang on the FSB. The Machine Check Architecture (MCA) code responding to an error indication, typically IERR# or MCERR#, will cause an attempt to interrogate the MCH for error status, and if that FSB transaction fails to complete the processor will automatically time out and respond by issuing a BINIT# sequence on the FSB.

When BINIT# is asserted on the FSB, all bus agents (CPUs and MCH) are required to reset their internal FSB arbiters and all FSB tracking state machines and logic to their default states. This will effectively “un-hang” the bus to provide a path into chipset configuration space. Note that the MCH device implements “sticky” error status bits, providing the platform software architect with free choice between BINIT# and a general hard reset to recover from a hung system.

Although BINIT# will not clear any configuration status from the system, it is not a recoverable event from which the platform may continue normal execution without first running a hard reset cycle. To guarantee that the FSB is cleared of any hang condition, the MCH will clear all pending transaction states within its internal buffers. This applies to outstanding FSB cycles as required, but also to in-flight memory transactions and inbound transactions. The resulting state of the platform will be highly variable depending upon what precisely got wiped-out due to the BINIT# event, and it is not possible for hardware to guarantee that the resulting state of the machine will support continued operation. What the MCH will guarantee is that no subordinate device has been reset due to this event (PCI Express links will remain “up”), and that no internal configuration state (sticky or otherwise) has been lost. The MCH will also continue to maintain main memory via the refresh mechanism through a BINIT# event, thus machine-check software will have access not only to machine state, but also to memory state in tracking-down the source of the error.

5.20.5 Reset Sequencing

Figure 5-30, “Power-On Reset Sequence” on page 430 contains a timing diagram illustrating the progression through the power-on reset sequence. This is intended as a quick reference for system designers to clarify the requirements of the MCH.

Note the breaks in the clock waveform at the top of Figure 5-30, which are intended to illustrate further elapsed time in the interest of displaying a lengthy sequence in a single picture. Each of the delays in the reset sequence is of fixed duration, enforced by either the MCH or the Intel® 631xESB/632xESB I/O Controller Hub. In the case of a power-on sequence, the MCH internal “hard” and “core” resets deassert simultaneously. The two lines marked with names beginning “HLA” illustrate the ESI special cycle handshake between the MCH and the Intel® 631xESB/632xESB I/O Controller Hub to coordinate across the deasserting edge of the FSBxRESET# output from the MCH.

Table 5-26 summarizes the durations of the various reset stages illustrated above, and attributes the delays to the component that enforces them.

The fixed delays provide time for subordinate PLL circuitry to lock on interfaces where the clock is withheld or resynchronized during the reset sequence.

Figure 5-30. Power-On Reset Sequence

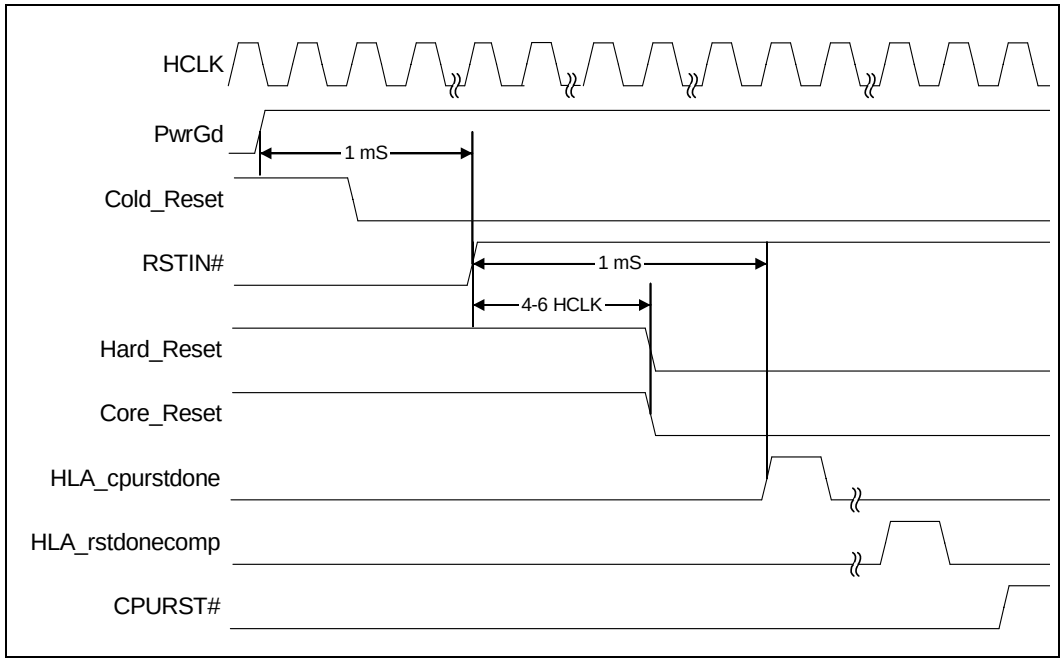


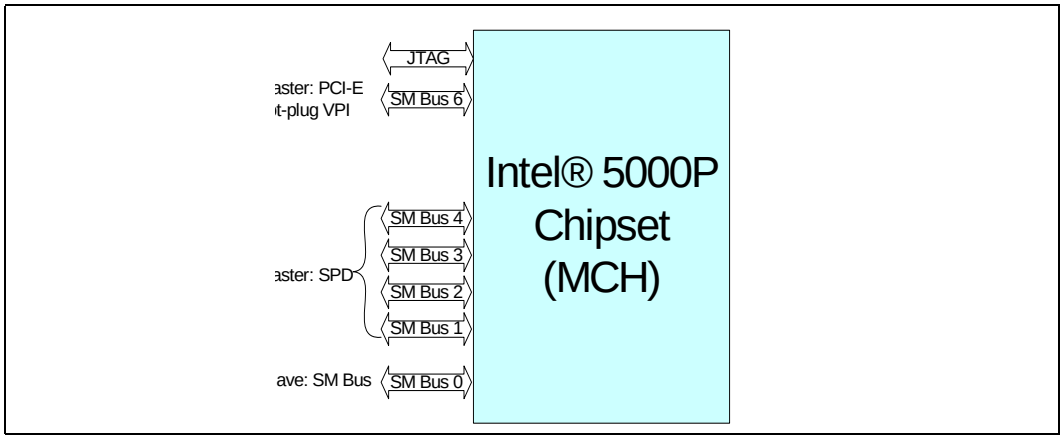
Table 5-26. Reset Sequences and Durations

From	To	Duration	Source	Comment
Power on	PwrGd	>2mS	Platform	Control logic on the platform must ensure that there are at least 2mS of stable power before PwrGd is asserted.
PwrGd	RSTIN# deassertion	1mS	Intel® 631xESB / 632xESB I/O Controller Hub	Intel® 631xESB/632xESB I/O Controller Hub enforces delay between detecting PwrGd asserted and releasing PCIRST# (note that Intel® 631xESB/ 632xESB I/O Controller Hub PCIRST# is directly connected to MCH RSTIN#).
RSTIN# deassertion	Hard/Core deassertion	4-6 HCLK	MCH	MCH waits for a common rising edge on all internal clocks, then releases core reset(s).
RSTIN# deassertion	FSBxRESET# deassertion	1mS	MCH	MCH enforces delay between RSTIN# and FSBxRESET# deassertion. ESI handshake is incremental to the timer.

5.21 SMBus Interfaces Description

The Intel 5000X Chipset MCH provides six fully functional System Management Bus (SMBus) Revision 2.0 compliant target interfaces. These interfaces are used to support platform level operations such as FB-DIMM memory Serial Presence Detect, PCI Hot-Plug, and configuration of platform devices. Each of these interfaces have dedicated uses as shown in Figure 5-31.

Figure 5-31. MCH SM Bus Interfaces



SM Buses 1, 2, 3 and 4 are dedicated to memory serial presence detect and FB-DIMM configuration. Each bus is dedicated to a single FB-DIMM channel. SM Bus 1 is assigned to FB-DIMM channel 0, SM Bus 2 is assigned to FB-DIMM channel 1, SM Bus 3 is assigned to FB-DIMM channel 2, and SM Bus 4 is assigned to FB-DIMM channel 3. SM Bus 6 is used to support PCI Express Hot-Plug.

The each SMBus interface consists of two interface pins; one a clock, and the other serial data. Multiple initiator and target devices may be electrically present on the same pair of signals. Each target recognizes a start signaling semantic, and recognizes its own 7-bit address to identify pertinent bus traffic. The MCH address is hard-coded to 01100000b (60h).

The SMBus protocol allows for traffic to stop in “mid sentence,” requiring all targets to tolerate and properly “clean up” in the event of an access sequence that is abandoned by the initiator prior to normal completion. The MCH is compliant with this requirement.

The protocol comprehends “wait states” on read and write operations, which the MCH takes advantage of to keep the bus busy during internal configuration space accesses.

5.21.1 Internal Access Mechanism

All SMBus accesses to internal register space are initiated via a write to the CMD byte. Any register writes received by the MCH while a command is already in progress will receive a NAK to prevent spurious operation. The master is no longer expected to poll the CMD byte to prevent the obliteration a command in progress prior to issuing further writes. The SMBus access will be delayed by stretching the clock until such time that the data is delivered. Note that per the *System Management Bus (SMBus) Specification, Rev 2.0*, this can not be longer than 25 ms. To set up an internal access, the four ADDR bytes are programmed followed by a command indicator to execute a read or write. Depending on the type of access, these four bytes indicate either the Bus number, Device, Function, Extended Register Offset, and Register Offset, or the memory-mapped region selected and the address within the region. The configuration type access utilizes the traditional bus number, device, function, and register offset; but in addition, also uses an extended register offset which expands the addressable register space from 256 bytes to 4 Kilobytes. The memory-mapped type access redefines these bytes to be a memory-mapped region selection byte, a filler byte which is all zeroes, and then the memory address within the region. Refer to the earlier

tables, which display this information. Note that the filler byte is not utilized, but enforces that both types of accesses have the same number of address bytes, and does allow for future expansion.

It is perfectly legal for an SMBus access to be requested while an FSB-initiated access is already in progress. The MCH supports “wait your turn” arbitration to resolve all collisions and overlaps, such that the access that reaches the configuration ring arbiter first will be serviced first while the conflicting access is held off. An absolute tie at the arbiter will be resolved in favor of the FSB. Note that SMBus accesses must be allowed to proceed even if the internal MCH transaction handling hardware and one or more of the other external MCH interfaces are hung or otherwise unresponsive.

5.21.2 SMBus Transaction Field Definitions

The SMBus target port has it’s own set of fields which the MCH sets when receiving an SMBus transaction. They are not directly accessible by any means for any device.

Table 5-27. SMBus Transaction Field Summary

Position	Mnemonic	Field Name
1	CMD	Command
2	BYTCNT	Byte Count
3	ADDR3	Bus Number (Register Mode) or Destination Memory (Memory Mapped Mode)
4	ADDR2	Device / Function Number (Register Mode) or Address Offset [23:16] (Memory Mapped Mode)
5	ADDR1	Extended Register Number (Register Mode) or Address Offset [15:8] (Memory Mapped Mode)
6	ADDR0	Register Number (Register Mode) or Address Offset [7:0] (Memory Mapped Mode)
7	DATA3	Fourth Data Byte [31:24]
8	DATA2	Third Data Byte [23:16]
9	DATA1	Second Data Byte [15:8]
10	DATA0	First Data Byte [7:0]
11	STS	Status, only for reads

Table 5-27 indicates the sequence of data as it is presented on the SMBus following the byte address of the MCH itself. Note that the fields can take on different meanings depending on whether it is a configuration or memory-mapped access type. The command indicates how to interpret the bytes.

5.21.2.1 Command Field

The command field indicates the type and size of transfer. All configuration accesses from the SMBus port are initiated by this field. While a command is in progress, all future writes or reads will be negative acknowledged (NAK) by the MCH to avoid having registers overwritten while in use. The two command size fields allows more flexibility on how the data payload is transferred, both internally and externally. The begin and end bits support the breaking of the transaction up into smaller transfers, by defining the start and finish of an overall transfer.

Position	Description
7	**Begin Transaction Indicator.** 0 = Current transaction is NOT the first of a read or write sequence. 1 = Current transaction is the first of a read or write sequence. On a single transaction sequence this bit is set along with the End Transaction Indicator.
6	End Transaction Indicator. 0 = Current transaction is NOT the last of a read or write sequence. 1 = Current transaction is the last of a read or write sequence. On a single transaction sequence this bit is set along with the Begin Transaction Indicator.
5	Address Mode. Indicates whether memory or configuration space is being accessed in this SMBus sequence. 0 = Memory Mapped Mode 1 = Configuration Register Mode
4	Packet Error Code (PEC) Enable. When set, each transaction in the sequence ends with an extra CRC byte. The MCH would check for CRC on writes and generate CRC on reads. PEC is not supported by the MCH. 0 = Disable 1 = Not Supported
3:2	Internal Command Size. All accesses are naturally aligned to the access width. This field specifies the internal command to be issued by the SMBus slave logic to the MCH core. 00 = Read Dword 01 = Write Byte 10 = Write Word 11 = Write Dword
1:0	SMBus Command Size. This field specifies the SMBus command to be issued on the SMBus. This field is used as an indication of the length of the transfer so that the slave knows when to expect the PEC packet (if enabled). 00 = Byte 01 = Word 10 = DWord 11 = Reserved

5.21.2.2 Byte Count Field

The byte count field indicates the number of bytes following the byte count field when performing a write or when setting up for a read. The byte count is also used, when returning data, to indicate the number of bytes (including the status byte) which are returned prior to the data. Note that the byte count is only transmitted for block type accesses on SMBus. SMBus word or byte accesses do not use the byte count.

Position	Description
7:0	**Byte Count.** Number of bytes following the byte count for a transaction.

5.21.2.3 Address Byte 3 Field

This field should be programmed with the bus number of the desired configuration register in the lower 5 bits for a configuration access. For a memory-mapped access, this field selects which memory-map region is being accessed. There is no status bit to poll to see if a transfer is in progress, because by definition if the transfer completed when the task is done. Clock stretch is used to guarantee the transfer is truly complete.

The MCH does not support access to other logical bus numbers via the SMBus port. All registers “attached” to the SMBus have access to all other registers that are on logical bus#0. The MCH makes use of this knowledge to implement a modified usage of the Bus Number register providing access to internal registers outside of the PCI compatible configuration window.

Position	Configuration Register Mode Description	Memory Mapped Mode Description

7:5	Ignored.	Memory map region to access. 01h = DMA 08h = DDR 09h = CHAP Others = Reserved
4:0	Bus Number. Must be zero: the SMBus port can only access devices on the MCH and all devices are bus zero.	

5.21.2.4 Address Byte 2 Field

This field indicates the Device Number and Function Number of the desired configuration register if for a configuration type access, otherwise it should be set to zero.

Position	Configuration Register Mode Description	Memory Mapped Mode Description
7:3	**Device Number.** Can only be devices on the MCH.	Zeros used for padding.
2:0	Function Number.	

5.21.2.5 Address Byte 1 Field

This field indicates the upper address bits for the 4K region specified by the register offset. Only the lower bit positions of this field are used, the upper four bits are ignored.

Position	Description
7:4	Ignored.
3:0	Extended Register Number. Upper address bits for the 4K region of register offset.

5.21.2.6 Address Byte 0 Field

This field indicates the lower eight address bits for the register with the 4K region, regardless whether it is a configuration or memory-map type of access.

Position	Description
7:0	**Register Offset.**

5.21.2.7 Data Field

This field is used to receive read data or to provide write data associated with the addressed register.

At the completion of a read command, this field will contain the data retrieved from the addressed register. All reads will return an entire aligned DWord (32 bits) of data.

For write operations, the number of byte(s) of this 32 bit field is loaded with the desired write data. For a byte write only bits 7:0 will be used, for a Word write only bits 15:0 will be used, and for a DWord write all 32 bits will be used.

Position	Description
31:24	**Byte 3 (DATA3).** Data bits [31:24] for DWord.
23:16	**Byte 2 (DATA2).** Data bits [23:16] for DWord.
15:8	**Byte 1 (DATA1).** Data bits [15:8] for DWord and Word.
7:0	**Byte 0 (DATA0).** Data bits [7:0] for DWord, Word and Byte.

5.21.2.8 Status Field

For a read cycle, the returned data is preceded by one byte of status. The following table shows how the status byte bits are defined.

Position	Description
7	**Internal Time-out.** **0 = SMBus request is completed within 2 ms internally** 1 = SMBus request is not completed in 2 ms internally.
6	Ignored.
5	**Internal Master Abort.** 0 = No Internal Master Abort Detected. 1 = Detected an Internal Master Abort.
4	**Internal Target Abort.** 0 = No Internal Target Abort Detected. 1 = Detected an Internal Target Abort.
3:1	Ignored.
0	Successful. 0 = The last SMBus transaction was not completed successfully. 1 = The last SMBus transaction was completed successfully.

5.21.2.9 Unsupported Access Addresses

It is possible for an SMBus master to program an unsupported bit combination into the ADDR registers. The MCH does not support such usage, and may not gracefully terminate such accesses.

5.21.3 SMB Transaction Pictographs

The Intel 5000X Chipset MCH SMBus target interface is targeted to enterprise domains. The enterprise domain is an extension of the original SMBus desktop domain. The following drawings are included to describe the SMBus enterprise transactions.

Figure 5-32. DWORD Configuration Read Protocol (SMBus Block Write / Block Read, PEC Disabled)

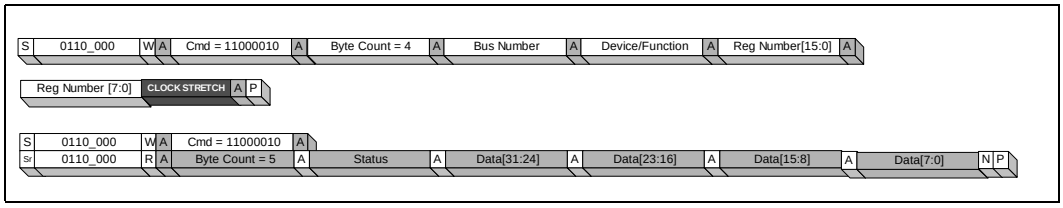


Figure 5-33. DWORD Configuration Write Protocol (SMBus Block Write, PEC Disabled)

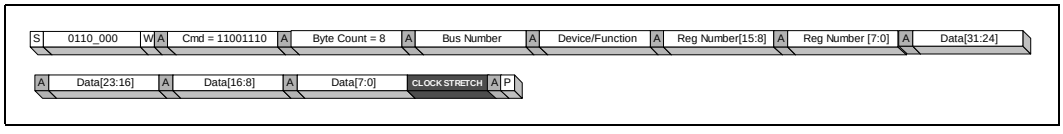


Figure 5-34. DWORD Memory Read Protocol (SMBus Block Write / Bock Read, PEC Disabled)

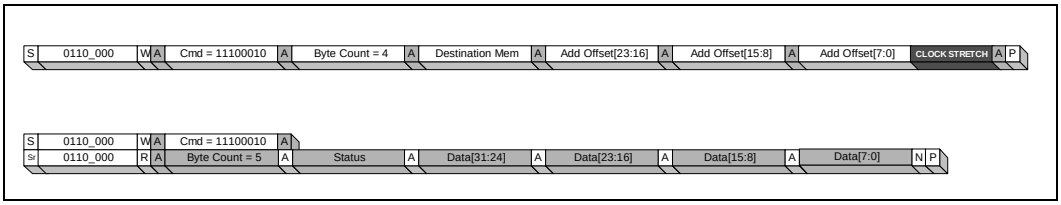


Figure 5-35. DWORD Memory Write Protocol

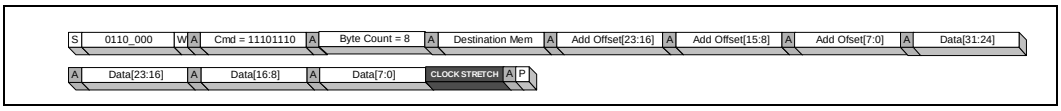


Figure 5-36. DWORD Configuration Read Protocol (SMBus Word Write / Word Read, PEC Disabled)

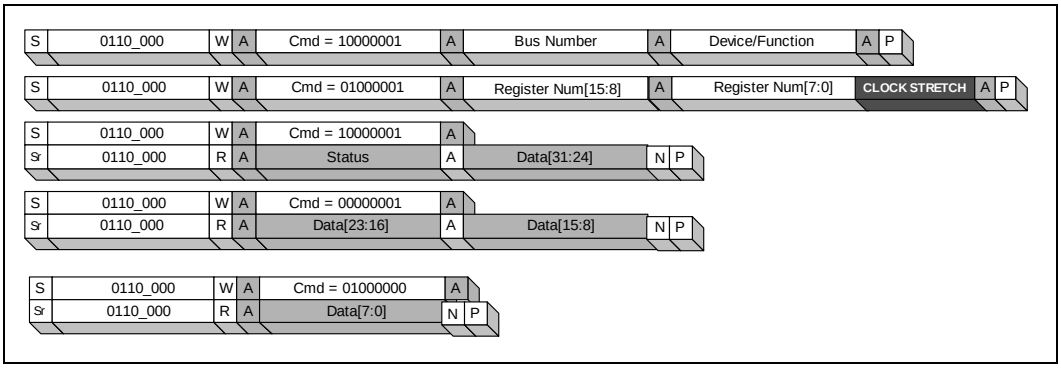


Figure 5-37. DWORD Configuration Write Protocol (SMBus Word Write, PEC Disabled)

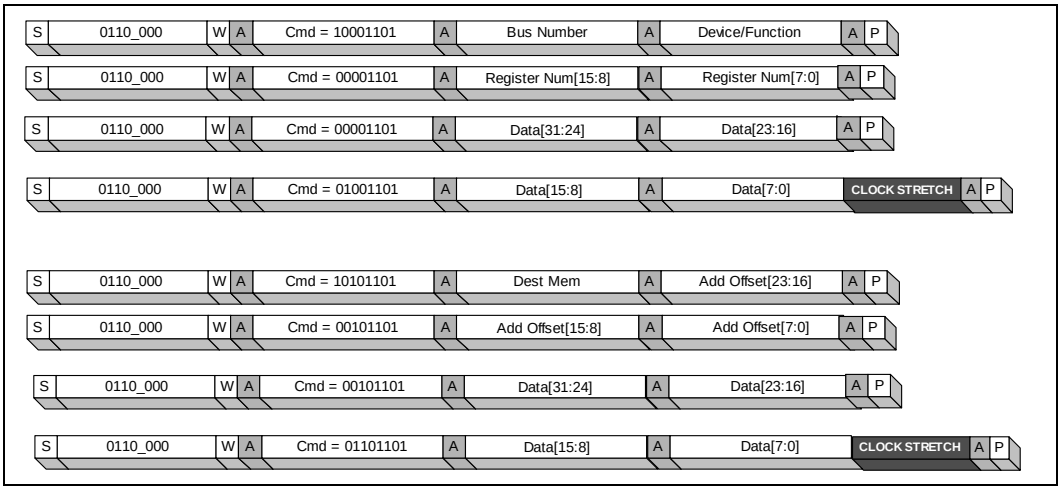


Figure 5-38. DWORD Memory Read Protocol (SMBus Word Write / Word Read, PEC Disabled)

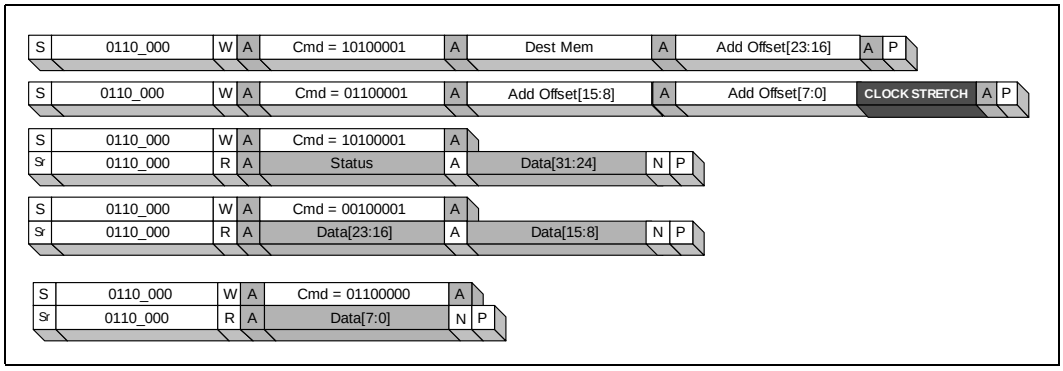
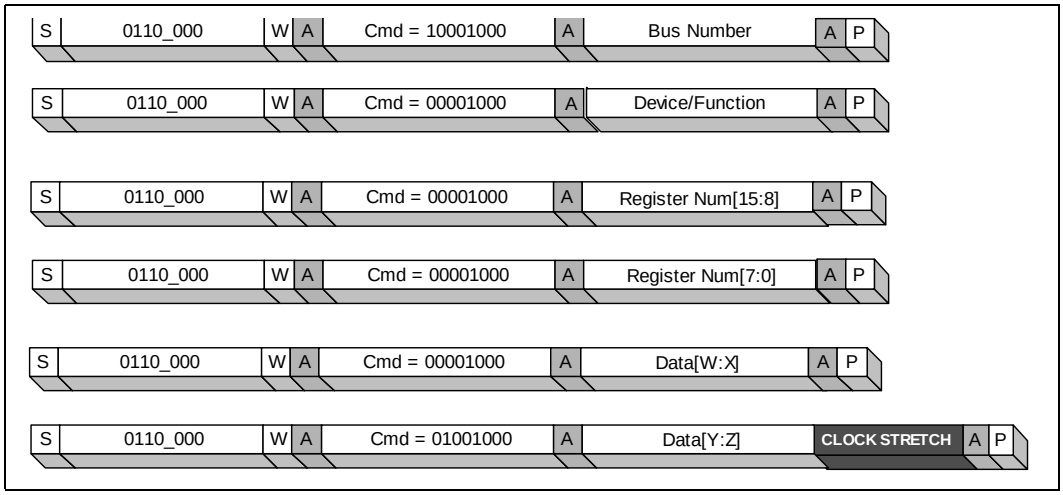


Figure 5-39. WORD Configuration Wrote Protocol (SMBus Byte Write, PEC Disabled)



5.21.4 Slave SM Bus, SM Bus 0

System Management software in a Intel 5000X Chipset platform can initiate system management accesses to the configuration registers via the Slave SM bus, SM Bus 0.

The mechanism for the Server Management (SM) software to access configuration registers is through a SMBus Specification, Revision 2.0 compliant slave port. Some Intel 5000X Chipset components contain this slave port and allow accesses to their configuration registers. The product specific details are compatible with the Intel® 631xESB/632xESB I/O Controller Hub SMBus configuration access mechanism. Most of the Intel 5000X Chipset MCH registers can be accessed through the SMBus configuration mechanism.

SMBus operations are made up of two major steps:

1. Writing information to registers within each component
2. Reading configuration registers from each component.

The following sections will describe the protocol for an SMBus master to access a Intel 5000X Chipset platform component's internal configuration registers. Refer to the SMBus Specification, Revision 2.0 for the bus protocol, timings, and waveforms.

Each component on the Intel 5000X Chipset platform must have a unique address. Intel 5000X Chipset platform component addresses are defined in the following table.

Table 5-28. SMBus Address for Product Name Platform

Component	SMBus Address (7:1)
Intel 5000X Chipset MCH	1100_000

5.21.4.1 Supported SMBus Commands

Product Name components SMBus Rev. 2.0 slave ports support the following six SMBus commands:

- Block Write
- Block Read
- Word Write
- Word Read
- Byte Write
- Byte Read

Sequencing these commands will initiate internal accesses to the component's configuration registers.

Each configuration read or write first consists of an SMBus write sequence which initializes the Bus Number, Device Number, and so forth. The term sequence is used since these variables may be written with a single block write or multiple word or byte writes. Once these parameters are initialized, the SMBus master can initiate a read sequence (which perform a configuration read) or a write sequence (which performs a configuration write).

Each SMBus transaction has an 8-bit command driven by the master. The format for this command is illustrated in Table 5-29 below.

Table 5-29. SMBus Command Encoding

7	6	5	4	3:2	1:0
Begin	End	Rsvd	PEC_en	Internal Command: 00 - Read DWord 01 - Write Byte 10 - Write Word 11 - Write DWord	SMBus Command: 00 - Byte 01 - Word 10 - Block 11 - *Rsvd*

The *Begin* bit indicates the first transaction of a read or write sequence.

The *End* bit indicates the last transaction of a read or write sequence.

The *Pecan* bit enables the 8-bit Packet Error Code (PEC) generation and checking logic.

The *Internal Command* field specifies the internal command to be issued by the SMBus slave logic. Note that the Internal Command must remain consistent during a sequence that accesses a configuration register. Operation cannot be guaranteed if it is not consistent when the command setup sequence is done.

The *SMBus Command* field specifies the SMBus command to be issued on the bus. This field is used as an indication of the length of transfer so the slave knows when to expect the Packet Error Code packet.

Reserved bits should be written to zero to preserve future compatibility.

5.21.4.2 Configuration Register Read Protocol

Configuration reads are accomplished through an SMBus write(s) and later followed by an SMBus read. The write sequence is used to initialize the Bus Number, Device, Function, and Register Number for the configuration access. The writing of this information can be accomplished through any combination of the supported SMBus write commands (Block, Word or Byte). The *Internal Command* field for each write should specify Read DWord.

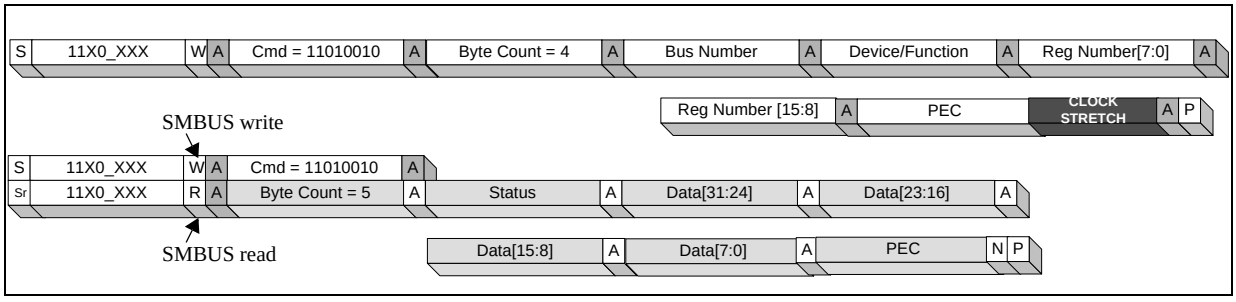
After all the information is set up, the last write (*End* bit is set) initiates an internal configuration read. If the data is not available before the slave interface acknowledges this last write command (ACK), the slave will "clock stretch" until the data returns to the SMBus interface unit. If an error occurs during the internal access, the last write command will receive a NAK. A status field indicates abnormal termination and contains status information such as target abort, master abort, and time-outs. The status field encoding is defined in the following table.

Table 5-30. Status Field Encoding for SMBus Reads

Bit	Description
7	Internal Time-out. This bit is set if an SMBus request is not completed in TBD internally (2ms?)
6	Reserved
5	Internal Master Abort
4	Internal Target Abort
3:1	Reserved
0	Successful

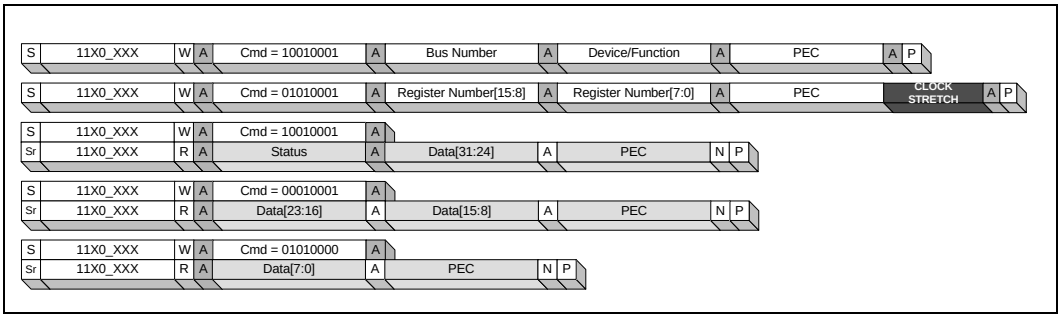
Examples of configuration reads are illustrated below. All of these examples have Packet Error Code (PEC) enabled. If the master does not support PEC, then bit 4 of the command would be cleared and there would not be a PEC phase. For the definition of the diagram conventions below, refer to the *SMBus Specification*, Revision 2.0. For SMBus read transactions, the last byte of data (or the PEC byte if enabled) is NAKed by the master to indicate the end of the transaction. For diagram compactness, "Register Number[]" is also sometimes referred to as "Reg Number" or "Reg Num".

Figure 5-40. SMBus Configuration Read (Block Write / Block Read, PEC Enabled)



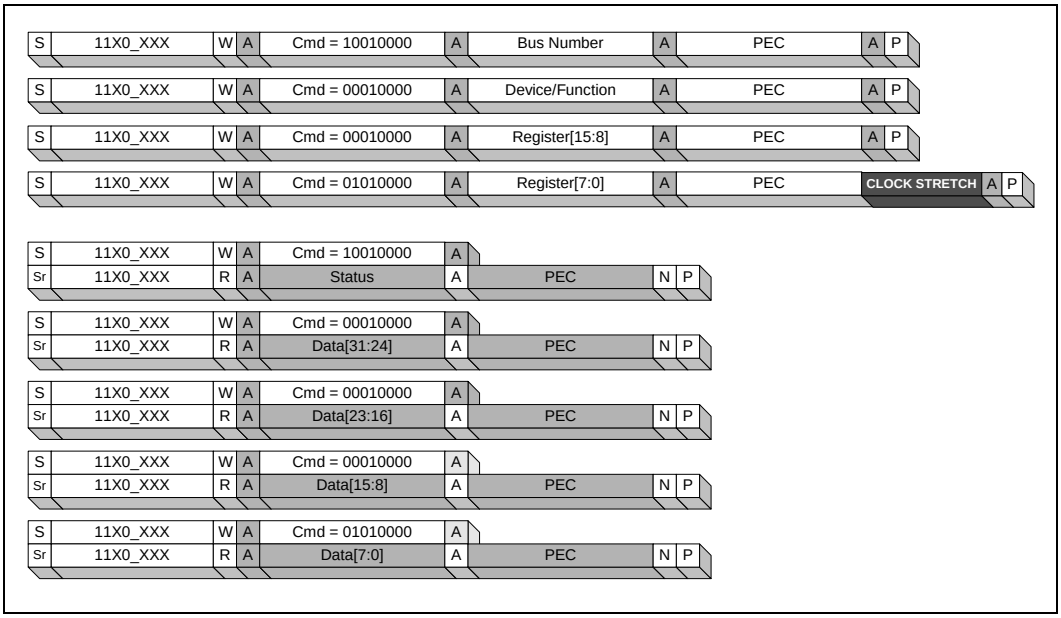
This is an example using word reads. The final data is a byte read.

Figure 5-41. SMBus Configuration Read (Word Writes / Word Reads, PEC Enabled)



The following example uses byte reads.

Figure 5-42. SMBus Configuration Read (Write Bytes / Read Bytes, PEC Enabled)



5.21.4.3 Configuration Register Write Protocol

Configuration writes are accomplished through a series of SMBus writes. As with configuration reads, a write sequence is first used to initialize the Bus Number, Device, Function, and Register Number for the configuration access. The writing of this information can be accomplished through any combination of the supported SMBus write commands (block, word or byte).

Examples of configuration writes are illustrated below. For the definition of the diagram conventions below, refer to the *SMBus Specification*, Revision 2.0.

Figure 5-43. SMBus Configuration Write (Block Write, PEC Enabled)

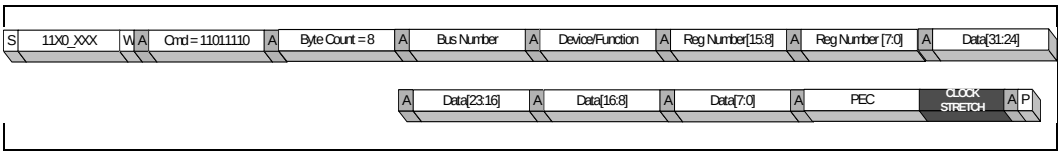


Figure 5-44. SMBus Configuration Write (Word Writes, PEC Enabled)

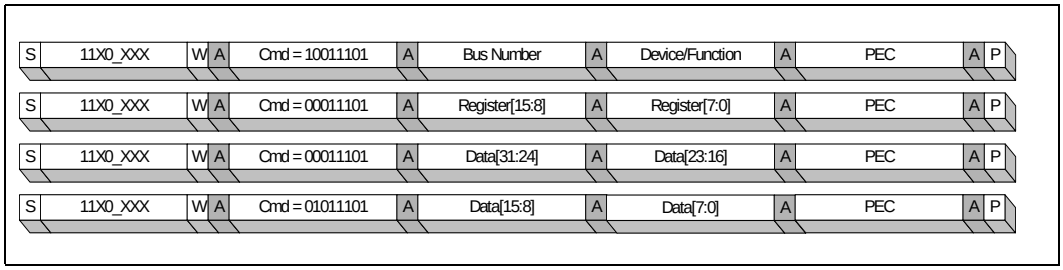
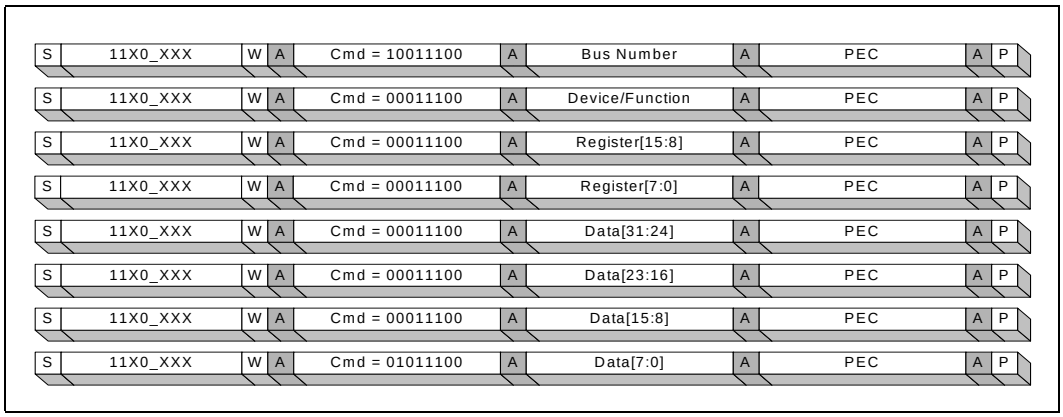


Figure 5-45. SMBus Configuration Write (Write Bytes, PEC Enabled)



5.21.4.4 SMBus Error Handling

The SMBus slave interface handles two types of errors: Internal and PEC. For example, internal errors can occur when the Intel 5000X Chipset issues a configuration read on the PCI Express port that read's terminates in error. These errors manifest as a not-acknowledge (NAK) for the read command (*End* bit is set). If an internal error occurs during a configuration write, the final write command receives a NAK just before the stop bit. If the master receives a NAK, the entire configuration transaction should be reattempted.

If the master supports Packet Error Checking (PEC) and the PEC_en bit in the command is set, then the PEC byte is checked in the slave interface. If the check indicates a failure, then the slave will NAK the PEC packet.

5.21.4.5 SMBus Interface Reset

- The slave interface state machine can be reset by the master in two ways:
- The master holds SCL low for 25ms cumulative. Cumulative in this case means that all the "low time" for SCL is counted between the Start and Stop bit. If this totals 25ms before reaching the Stop bit, the interface is reset.
- The master holds SCL continuously high for 50ms.

Note: Since the configuration registers are affected by the reset pin, SMBus masters will not be able to access the internal registers while the system is reset.

5.21.5 FB-DIMM SPD Interface, SM Buses 1, 2, 3 and 4

The MCH integrates a 100KHz SPD controller to access the FB-DIMM configuration information. SMBus 1 is dedicated to FB-DIMM branch 0, channel 0 DIMMs. SMBus 2 is dedicated to FB-DIMM branch 0, channel 1 DIMMs. SMBus 3 is dedicated to FB-DIMM branch 1, channel 0 DIMMs and SMBus 4 is dedicated to FB-DIMM branch 1, channel 1 DIMMs. There can be a maximum of four SPD EEPROM's associated with each SPD bus. The FB-DIMM SPD interfaces are wired as depicted in Figure 5-8.

Board layout must map chip selects to SPD Slave Addresses as shown in Table 5-7. The slave address is written to the SPDCMD configuration register.

5.21.5.1 SPD Asynchronous Handshake

The SPD bus is an asynchronous serial interface. Once software issues an SPD command (SPDCMD.CMD = SPDW or SPDR), software is responsible for verifying command completion before another SPD command can be issued. Software can determine the status of an SPD command by observing the SPD configuration register.

An SPD command has completed when any one command completion field (RDO, WOD, SBE) of the SPD configuration register is observed set to 1. An SPDR command has successfully completed when the RDO field is observed set to 1. An SPDW command has successfully completed when the WOD field is observed set to 1. An unsuccessful command termination is observed when the SBE field is set to 1. The MCH will clear the SPD configuration register command completion fields automatically whenever an SPDR or SPDW command is initiated. Polling may begin immediately after initiating an SPD command.

Software can determine when an SPD command is being performed by observing the BUSY field of the SPD configuration register. When this configuration bit is observed set to 1, the interface is executing a command.

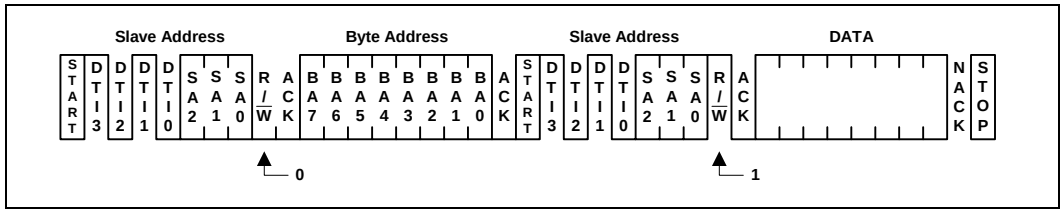
Valid SPD data is stored in the DATA field of the SPD configuration register upon successful completion of the SPDR command (indicated by 1 in the RDO field). Data to be written by an SPDW command is placed in the DATA field of the SPDCMD configuration register.

Unsuccessful command termination will occur when an EEPROM does not acknowledge a packet at any of the required ACK points, resulting in the SBE field being set to 1.

5.21.5.2 Request Packet for SPD Random Read

Upon receiving the SPDR command, the MCH generates the Random Read Register command sequence as shown in Figure 5-46. The returned data is then stored in the MCH SPD configuration register in bits [7:0], and the RDO field is set to 1 by the MCH to indicate that the data is present and that the command has completed without error.

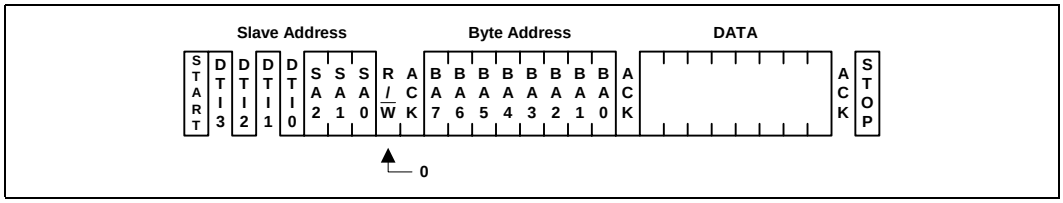
Figure 5-46. Random Byte Read Timing



5.21.5.3 Request Packet for SPD Byte Write

Upon receiving the SPDW command, the MCH generates the Byte Write Register command sequence as shown in Figure 5-47. The MCH indicates that the SIO command has completed by setting the WOD bit of the SPD configuration register to 1.

Figure 5-47. Byte Write Register Timing



5.21.5.4 SPD Protocols

The MCH supports the SPD protocols shown in Table 5-31.

Table 5-31. MCH Supported SPD Protocols

MCH Supported SPD Protocols
Random Byte Read
Byte Write

5.21.5.5 SPD Bus Time-out

If there is an error in the transaction, such that the SPD EEPROM does not signal an acknowledge, the transaction will time out. The MCH will discard the cycle and set the **SBE** bit of the **SPD** configuration register to 1 to indicate this error. The time-out counter within the MCH begins counting after the last bit of data is transferred to the DIMM, while the MCH waits for a response.

5.21.6 PCI Express Hot-Plug Support, SM Bus 6

SM Bus 6 is the PCI Express Hot-Plug port. SM Bus 6 is a Hot-Plug Virtual Pin Port (VPP) that operates using the SM Bus Masters protocol as defined in *System Management Bus Specification 2.0*.

SM Bus 6 is dedicated to support PCI Express Hot-Plug devices. Support for PCI Express is an option described in *PCI Express Base Specification*, Revision 1.0a. The PCI Express Hot-Plug model implies a hot-plug controller per port which is identified to software as a capability of the P2P Bridge configuration space.

PCI Express hot-plug support requires that the Intel 5000X Chipset MCH supports a set of hot-plug messages (listed in Figure 5-18 and Figure 5-22) to manage the states between the hot-plug controller and the device.

The PCI Express form factor has an impact to the level of support required of the MCH. For example, some of the hot-plug messages are required only if the LED indicators reside on the actual card and are accessed through the endpoint device. The Intel 5000X Chipset MCH supports all of the hot-plug messages so that the platform is not constrained to any particular form factor.

A standard hot-plug usage model is beneficial to customers who buy systems with hot-plug slots because many customers utilize hardware and software from different vendors. A standard usage model allows customers to use the PCI hot-plug slots on all of their systems without having to retrain operators.

In order to define a programming model for the PCI Standard Hot-Plug Controller (SHPC), it is necessary to make some assumptions about the interface between a human operator and a hot-plug slot. The SHPC programming model includes two indicators, one optional push button, and a sensor on the manually-operated retention latch for each supported slot.

5.21.6.1 Hot-Plug Indicators

The Standard Usage Model assumes that the platform provides two indicators per slot (the Power Indicator and the Attention Indicator). Each indicator is in one of three states: on, off, or blinking. Hot-plug system software has exclusive control of the indicator states by issuing commands to the SHPC.

The SHPC controls blink frequency, duty cycle, and phase. Blinking indicators operate at a frequency of 1.5 Hz and 50% (+/- 5%) duty cycle. Both indicators are completely under the control of system software.

5.21.6.2 Attention Button

An Attention Button is a momentary-contact push-button, located adjacent to each hot-plug slot, that is pressed by the user to initiate a hot-insertion or a hot-removal at that slot. The Power Indicator provides visual feedback to the human operator (if the system software accepts the request initiated by the Attention Button) by blinking. Once the Power Indicator begins blinking, a 5-second abort interval exists during which a second depression of the Attention Button cancels the operation. Software has the responsibility to implement this 5-second abort interval.

5.21.7 Hot-Plug Controller

PCI Express Hot-Plug requires that the Intel 5000X Chipset MCH implement a Hot-Plug controller for every Hot-Pluggable interface. The Hot-Plug controller is a capability of the bridge configuration space and the register set is accessible through the standard PCI capability mechanism defined in the *PCI Express Base Specification*, Revision 1.0a. Details on Hot-Plug operation and flow will be described in the *Intel 5000X Chipset Software Programmer's Guide*.

5.21.8 PCI Express Hot-Plug Usage Model

Not all concepts from the PCI standard hot-plug definition apply directly to PCI Express interfaces. The PCI Express specification still calls for an identical software interface in order to facilitate adoption with minimal development overhead on this aspect of the implementation. The largest variance from the old PCI hot-plug model is in control of the interface itself. PCI required arbitration support for idling already connected components, and "quick switches" to isolate the bus interface pins of a hot-plug slot. PCI Express is a point-to-point interface, making hot-plug a degenerate case of the old model that doesn't require such arbiter support. Furthermore, the PCI Express

interface is inherently tolerant of hot connect or disconnect, and does not have explicit clock or reset pins defined as a part of the bus (although they are standard pieces of some defined PCI Express connector form factors). As a result of these differences, some of the inherited hot-plug command and status codes are misleading when applied to PCI Express.

The compatible set of hot-plug registers may be accessed via memory-mapped transactions, or via the Intel 5000X Chipset MCH configuration mechanism as defined in the configuration mechanism chapter of this document. For specific information on the hot-plug register set, refer to the chapter on configuration register details.

The messages used for the hot-plug model are listed in Table 5-18, “Incoming PCI Express Requests” on page 409 and Table 5-22, “PCI Express Transaction ID Handling” on page 411 describe the behavior of the button and LEDs.

5.21.9 Virtual Pin Ports

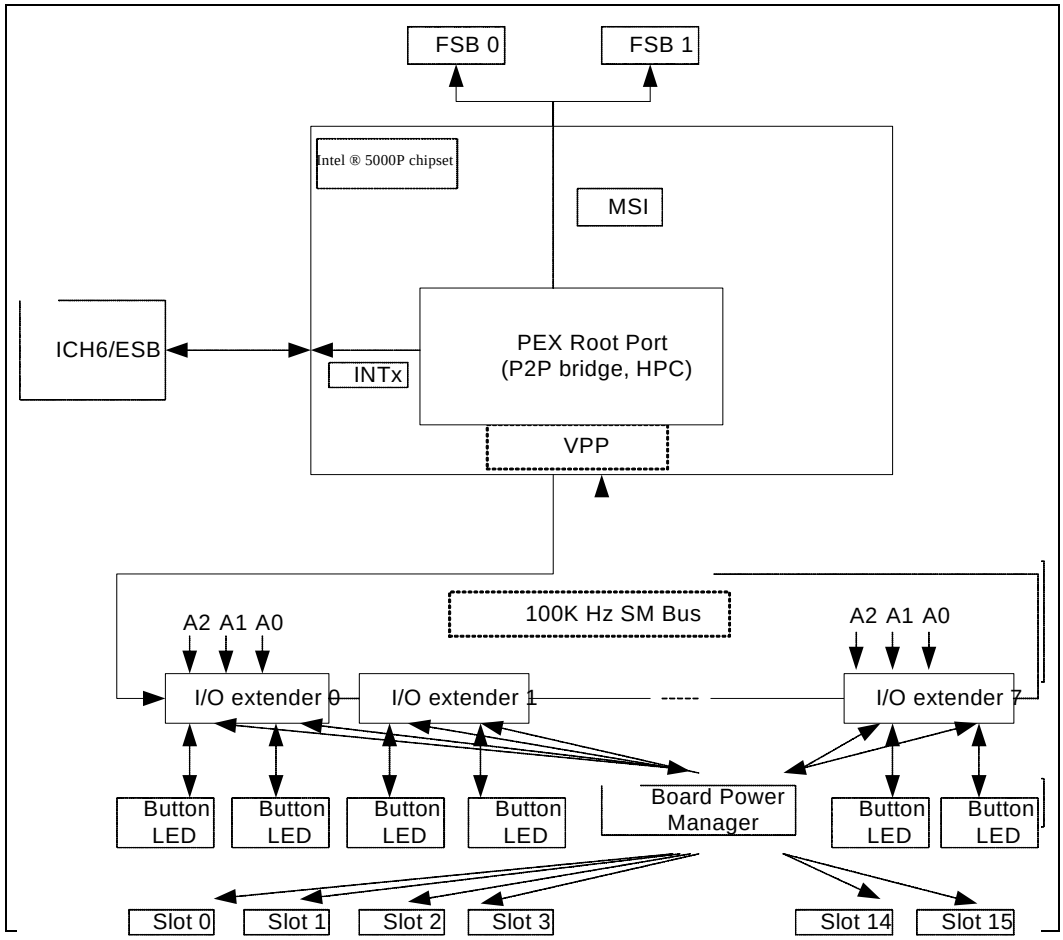
Shown in the Figure 5-1 is a high level block diagram of virtual pin ports and theoretical maximum number of PCI Express card slots that could be supported for hot-plug operations. In this VPP usage model, 16 slots (max) are shown in Figure 5-1 but for the Intel 5000X Chipset Platform only 6 PCI Express slots[16] will be used for the I/O hot-plug operations.

Note: Port 0, the ESI slot, is not hot-pluggable.

Since Intel 5000X Chipset MCH has only six PCI Express ports, only six hot-plug slots should be present in a Intel 5000X Chipset MCH platform. Intel 5000X Chipset MCH PCI Express virtual pin port will only process six hot-plug slots accordingly.

16.This does not include the ESI (port 0) which is not hot-pluggable.

Figure 5-1. PCI Express Hot-Plug/VPP Block Diagram



The Intel 5000X Chipset MCH masters a 100KHz hot-plug SMBus interface thru pins GPIOSMBCLK,and GPIOSMBDATA, for PCI Express ports that connect to a variable number of serial to parallel I/O ports such as the Phillips PCA9555[17] I/O Extender. The Intel 5000X Chipset MCH only supports SMBus devices with registers mapped as per Table 5-32. These I/O Extender components have 16 I/Os, divided into two 8-bit ports that can be configured as inputs or outputs. The Intel 5000X Chipset MCH has a crossbar which associates each PCI Express Hot-Plug Unit (HPU) slots with one of these 8-bit ports. The mapping is defined by a Virtual Pin Port register field, PEXCTRL.VPP, for each of the PCI Express HPU slots. The VPP register holds the SMBus address and port number of the IO Port associated with the PCI Express HPU. A[2:0] pins on each I/O Extender (that is, PCA9555 or compatible components) connected to the Intel 5000X Chipset MCH must strapped uniquely. Table 5-33 defines how the eight hot-plug signals are mapped to pins on the VPP.

17. Intel 5000X Chipset MCH VPP supports PCA9555 or compatible I/O Extender only.

.

Table 5-32. I/O Port Registers in I/O Extender supported by Intel 5000X Chipset MCH

Register	Name	Intel 5000X Chipset MCH Usage
0	Input Port 0	Continuously Reads Input Values
1	Input Port 1	
2	Output Port 0	Continuously Writes Output Values
3	Output Port 1	
4	Polarity Inversion Port 0	Not written by Intel 5000X Chipset MCH
5	Polarity Inversion Port 1	
6	Configuration Port 0	Direction set as per Table 5-33
7	Configuration Port 1	

5.21.9.0.1Operation

When the Intel 5000X Chipset MCH comes out of reset, the I/O ports are inactive. After a reset, the Intel 5000X Chipset MCH is not aware of how many IO Ports are connected to it, what their addresses are, nor what PCI Express ports are hot-pluggable. The Intel 5000X Chipset MCH does not master any commands on the SMBus until a hot-plug Capable bit is set.

For a PCI Express slot, an additional DIS_VPP bit is used to differentiate card or module hot-plug support, DIS_VPP bit needs to be set to 0 to enable hot-plug support for PCI Express card slot.

When BIOS sets a Hot-plug Capable bit (PEXSLOTCAP.HPC and PEXCTRL.DIS_VPP for PCI Express; HPCTL.HPC for FB-DIMM HPU), the Intel 5000X Chipset MCH initializes the associated VPP with Direction and Voltage Logic Level configuration as per Table 5-33. VPP registers for PCI Express which do not have the hot-plug capable bit set are invalid. Additionally, if the DIS_VPP bit is set to 1, then the corresponding VPP register is invalid for the PCI Express slot. This is intended for PCI Express module hot-plug which no VPP support is required. The I/O Extender's Polarity is left at its default value and never written, but the direction and voltage logic levels are written using the addresses defined in Table 5-33.

When the Intel 5000X Chipset MCH is not doing a direction write, it performs input register reads and output register writes to all valid VPPs. This sequence repeats indefinitely until a new hot-plug capability bit is set. To minimize the completion time of this sequence and minimize complexity, both ports are always read or written. For the maximum number of 6 IO Ports, and assuming no clock stretching, this sequence can take up to 51ms. If new hot-plug capability bits are not being set, this is the maximum timing uncertainty in sampling or driving these signals.

Table 5-33 describes the Hot-Plug Signals used for hot-plug.

Table 5-33. Hot-Plug Signals on a Virtual Pin Port

Bit	Direction	Voltage Logic Level	Signal	Logic True Meaning	Logic False Meaning
0	Output	High_true	ATNLED	ATTN LED is to be turned ON	ATTN LED is to be turned OFF
1	Output	High_true	PWRLED	PWR LED is to be turned ON	PWR LED is to be turned OFF
2	Input	Low_true	BUTTON#	ATTN Button is Pressed	ATTN Button is NOT Pressed
3	Input	Low_true	PWRFLT#	PWR Fault in the VRM	No PWR Fault in the VRM
4	Input	low_true	PRSNT#	Card Present in Slot	Card NOT Present in Slot
5	Output	high_true	PWREN	Power is to be enabled on the Slot	Power is NOT to be enabled on the Slot
6	Input	low_true	MRL#	MRL is open	MRL is closed
7	Input	low_true	GPI#	Power good on Slot	No Power good on Slot

The Intel 5000X Chipset MCH will send Assert_intx/Deassert_intx or Assert_HPGPE/Deassert_HPGPE messages to the ESI port as virtual pin messages to enable the Intel® 631xESB/632xESB I/O Controller Hub take the appropriate action for handling the hot-plug (legacy/ACPI interrupt mode) in non-MSI mode.

5.22 Clocking

The following section describes the Intel 5000X Chipset MCH Clocks.

5.22.1 Reference Clocks

The BUSCLK, and CORECLK (herein referred to “in aggregate” as “BUSCLK”) reference clocks, operating at 133/166/266MHz, are supplied to the Intel 5000X Chipset MCH. These are the processor bus, core, and snoop filter PLL reference clocks. This frequency is common between all processor bus agents. Phase matching between agents is required. The two processor FSBs operate in phase with the core clock.

The FB-DIMM(0/1)CLK reference clocks, (herein referred to as FBDCLK) operating at half the DDR2 frequency (operating at the SDRAM command-clock frequency, which is the FB-DIMM packet frequency), are supplied to the Intel 5000X Chipset MCH. This is the FB-DIMM PLL reference clock. This frequency is common between the Intel 5000X Chipset MCH and DIMMs. Phase matching between agents is not required (plesiochronous). The Intel 5000X Chipset MCH and DIMMs treat this frequency domain synchronously. The FB-DIMM unit-interval (UI) PLL outputs 12x the FBDCLK frequency. for example, For DDR2 667 MHz DIMMs, the FBDCLK frequency is 333MHz and the UI (link) frequency is 4.0 GHz.

The PECLK reference clock, operating at 100MHz, is supplied to the Intel 5000X Chipset MCH. This is the PCI Express PLL reference clock. The PCI Express flit PLL outputs 250 MHz. The PCI Express phit PLL outputs 2.5 GHz. The phit clock frequency must be tightly matched (mesochronous mode) between both PCI Express agents when spectrum-spreading is not employed. The phit clock frequency is common to both PCI Express agents when spectrum-spreading is employed. When the phit clock frequency is common to both PCI Express agents, no phase matching between them is required (plesiochronous mode). The Intel 5000X Chipset MCH core treats this frequency domain asynchronously.

The BUSCLK and FBDCLK reference clocks are derived from the same oscillator. The PECLK reference clock may be derived from a different oscillator.

The PCI Express interfaces operate asynchronously with respect to the core clock.

Table 5-34. Intel 5000X Chipset MCH Frequencies for Processors and Core

Core	Domain	Frequency	Reference Clock
133MHz	BUSCLK	133 MHz	
	FSB 1X		BUSCLK
	FSB 2X	266 MHz	
	FSB 4X	533 MHz	
266MHz	BUSCLK	266 MHz	
	FSB 1X		
	FSB 2X	533 MHz	
	FSB 4X	1,067 MHz	
333MHz	BUSCLK	167 MHz	
	FSB 1X		
	FSB 2X	333 MHz	
	FSB 4X	667 MHz	
333MHz	BUSCLK	333 MHz	
	FSB 1X		
	FSB 2X	667 MHz	
	FSB 4X	1333 MHz	

Table 5-35. Intel 5000X Chipset MCH Frequencies for Memory

DDR	Domain	Frequency	Reference Clock
533MHz	FBD U	3.2GHz	FBDCLK
	FBD packet	266 MHz	
	FBDCLK	133 MHz	
667MHz	FBD U	4.0 GHz	
	FBD packet	333 MHz	
	FBDCLK	167 MHz	
800MHz	FBD U	4.8GHz	
	FBD packet	400 MHz	
	FBDCLK	200 MHz	

Table 5-36. Intel 5000X Chipset MCH Frequencies for PCI Express

Domain	Frequency	Reference Clock
PCI Express phit	2.5GHz	PECLK
PCI Express flit	250 MHz	PECLK

5.22.2 JTAG

TCK is asynchronous to core clock. For private TAP register accesses, one TCK cycle is a minimum of 10 core cycles. The TCK high time is a minimum of 5 core cycles in duration. The TCK low time is a minimum of 5 core cycles in duration. The possibility of metastability during private register access is mitigated by circuit design. A metastability hardened synchronizer will guarantee an MTBF greater than 10^7 years.

For public TAP register accesses, TCK operates independently of the core clock.

5.22.3 SMBus Clock

The SMBus clock is synchronized to the core clock. Data is driven into the Intel 5000X Chipset with respect to the serial clock signal. Data received on the data signal with respect to the clock signal will be synchronized to the core using a metastability hardened synchronizer guaranteeing an MTBF greater than 10^7 years. The serial clock can not be active until 10 mS after RESETI# deassertion. When inactive, the serial clock should be deasserted (High). The serial clock frequency is 100 KHz.

5.22.4 GPIO Serial Bus Clock

The transmitted 100Khz Virtual Pin Interface (VPI) clock (one of the SCL[4:0]'s) is derived from the core clock. The PCI Express Hot-Plug signals reside on the Virtual Pin Interface.

5.22.5 Clock Pins

Table 5-37. Clock Pins (Sheet 1 of 2)

Pin Name	Pin Description
BUSCLKP	Processor bus clock
BUSCLKN	Processor bus clock (Complement)
PECLKP	PCI Express clock
PECLKN	PCI Express clock (Complement)
FBD{0/1}CLKP	FB-DIMM clocks
FBD{0/1}CLKN	FB-DIMM clocks (Complement)
PLLBYPASS	PLL Bypass mode
PRCSPEED	BUSCLK:CORECLK Bus Ratio Selector
VCC{0/1/2/3}AMP	Analog power supply for FB-DIMM PLLs
VSS{0/1/2/3}AMP	Analog ground for FB-DIMM PLLs
VCCAPB	Analog power supply for processor bus PLL
VSSAPB	Analog ground for processor bus PLL
VCCAPE	Analog power supply for PCI Express PLLs
VSSAPE	Analog ground for PCI Express PLLs
VCCACORE	Analog power supply for Core PLL
VSSACORE	Analog ground for Core PLL
TCK	TAP clock
GPIOSCL	GPIO (Virtual Pin Port) clock
SCL	SMBus clock
OCPSTBP#	Debug bus data strobe

Table 5-37. Clock Pins (Sheet 2 of 2)

Pin Name	Pin Description
OCPSTBN#	Debug bus data strobe (Complement)
PB{0/1}STBP[3:0]#	Processor bus data strobes
PB{0/1}STBN[3:0]#	Processor bus data strobes (Complements)
PB{0/1}ADSTB[1:0]#	Processor bus address strobes

5.22.6 High Frequency Clocking Support

5.22.6.1 Spread Spectrum Support

The Intel 5000X Chipset MCH PLLs will support Spread Spectrum Clocking (SSC). SSC is a frequency modulation technique for EMI reduction. Instead of maintaining a constant frequency, SSC modulates the clock frequency/period along a predetermined path, that is, the modulation profile.The Intel 5000X Chipset MCH is designed to support a nominal modulation frequency of 30 KHz with a down spread of 0.5%.

5.22.6.2 Stop Clock

PLLs in the Intel 5000X Chipset MCH cannot be stopped.

5.22.6.3 Jitter

The FB-DIMM UI clocks are produced by PLLs that multiply the FBDCLK frequency by 12. The PCI Express phit clocks are produced by PLLs that multiply the PECLK frequency by 25. These multi-GHz phit clocks require ultra-clean sources, ruling out all but specifically-crafted low-jitter clock synthesizers.

5.22.6.4 External Reference

An external crystal oscillator is the preferred source for the PLL reference clock. A spread spectrum frequency synthesizer that meets the jitter input requirements of the PLL is acceptable.

5.22.6.5 PLL Lock Time

All PLLs should be locked by PWRGOOD signal assertion. The reference clocks must be stable 1ms before the assertion of the PWRGOOD signal. The assertion of the PWRGOOD signal initiates the PLL lock process. External clocks dependent on PLLs are GPIO clock and SMBus clock. Many JTAG private registers are dependent on core PLL-generated clocks.

5.22.6.6 Other PLL Characteristics

The PLL VCOs oscillate continually from power-up. The PLL output dividers consistently track the VCO, providing pulses to the clock trees. Logic that does not receive an asynchronous reset can thus be reset “synchronously”.

A “locked” PLL will only serve to prove that the feedback loop is continuous. It will not prove that the entire clock tree is continuous.

5.22.6.7 Analog Power Supply Pins

The Intel 5000X Chipset MCH incorporates seven PLLs. Each PLL requires an Analog Vcc and Analog Vss pad and external LC filter. Therefore, there will be external LC filters for the Intel 5000X Chipset MCH. IMPORTANT: The filter is NOT to be connected to board Vss. The ground connection of the filter will be routed through the package and grounded to on-die Vss.

5.22.6.8 I/O Interface Metastability

PCI Express can be operated frequency-locked to the core. Flits are fifteen-sixteenths of the core frequency in 266MHz mode, three-quarters of the core frequency in 333MHz mode.

However, the phase between the frequency-locked domains is not controlled. This scheme results in the possibility of a metastability resonance where, for example, the commands generated by the core miss setup and hold to I/O every time. This condition can be tolerated by carefully hardened metastability design.

5.23 Error List

This section provides a summary of errors detected by the Intel 5000X Chipset. In the following table, errors are listed by the unit / interfaces. Some units / interfaces may provide additional error logging registers.

The following table provides the list of detected errors of a the MCH.

Table 5-38. Intel 5000X Chipset Error List (Sheet 1 of 9)

ERR # in MCH	Error Name	Definition	Error Type	Log Register	Cause / Actions
F1	Request/ Address Parity Error	MCH monitors the address and request parity signals on the FSB. A parity discrepancy over these fields during a valid request. MCH only detects this error caused by CPUs.	Fatal	FERR_FAT_FSB/ NERR_FAT_FSB. NRECFSB, NRECFSB_ADDRH, NRECFSB_ADDRL for FERR only.	Complete transaction on FSB with response (non-hard fail response)
F2	Unsupported Request or data size on FSB.	MCH detected an FSB Unsupported transaction. MCH only detects this error caused by CPUs.	Fatal	FERR_FAT_FSB/ NERR_FAT_FSB. NRECFSB for FERR only.	Treat as NOP. No Data Response or Retry by MCH
F5	Outstanding Deferred FSB transaction has timed out	MCH detected that a previously deferred FSB txn has not completed with Defer Reply within a specified time frame.	Fatal	FERR_FAT_FSB/ NERR_FAT_FSB. NRECFSB for FERR only	An access issued on the FSB has timed out.
F6	Data Parity Error	MCH monitors the data/ parity signals on the FSB. Set when the MCH detects an parity error during the data transfer. MCH only detects this error caused by CPUs.	UnCorr	FERR_NF_FSB/ NERR_NF_FSB. RECFSB for FERR only	Received a parity error. Poison Data and forward to the appropriate interface.
F7	Detected MCERR	MCH detected that a processor issued an MCERR.	UnCorr	FERR_NF_FSB/ NERR_NF_FSB. based on POC[5] setting	If (receive an MCERR) forward the MCERR to the other FSB bus, adhering to the MCERR protocol
F8	B-INIT	MCH detected that a processor issued an B-INIT.	UnCorr	FERR_NF_FSB/ NERR_NF_FSB. based on POC[5] setting	Do not propagate to other FSB bus, reset arb. unit, and programatically reset platform

Table 5-38. Intel 5000X Chipset Error List (Sheet 2 of 9)

ERR # in MCH	Error Name	Definition	Error Type	Log Register	Cause / Actions
F9	FSB protocol Error	BND detected FSB protocol error, for example, HitM on BIL and HitM on EWB.	Fatal	FERR_FAT_FSB/ NERR_FAT_FSB. NRECFSB, NRECFSB_ADDRH, NRECFSB_ADDRL for FERR only.	Complete transaction on FSB with response (IWB as in the example)
DMA0	Source Address Error	The DMA channel sets this bit indicating that the current descriptor has an illegal source address.	Fatal	Log FERR_CHANERR/ NERR_CHANERR,FERR_CHANSTS, FERR_CHANCMD, FERR_DESC_CTRL, FERR_SADDR, FERR_DADDR, FERR_TRANSFER_SIZE, FERR_NADDR, FERR_CHANCMP(Check Channel Completion enable) by taking a snap shot of the respective register fields when an error is met. The SADDR and DADDR will be the runtime addresses that the DMA engine is executing	Halt DMA engine
DMA1	Destination Address Error	The DMA channel sets this bit indicating that the current descriptor has an illegal destination address.	Fatal		Halt DMA engine
DMA2	Next Descriptor Address Error	The DMA channel sets this bit indicating that the next descriptor in the link list has an illegal address.	Fatal		Halt DMA engine
DMA3	Next Descriptor Alignment Error	The DMA channel sets this bit indicating that the next descriptor in the link list encountered a descriptor alignment error (not on a 64-byte boundary).	Fatal		An illegal next descriptor address flagged by the system Address decoder, which the DMA engine encounters in the current descriptor after having successfully completed the data transfer for the current descriptor including any associated completions/ interrupts Halt DMA engine
DMA4	Chain Address Value Error	The DMA channel sets this bit indicating that the register has an illegal address including an alignment error (not on a 64-byte boundary).	Fatal		Halt DMA engine
DMA5	CHANCMD Error	The DMA channel sets this bit indicating that a write to the CHANCMD register contained an invalid value (for example, more than one command bit set)	Fatal		Halt DMA engine
DMA6	Chipset Data Parity error	The DMA channel sets this bit indicating that there is a data parity error during a read/write operation of a given DMA descriptor.	Fatal		Halt DMA engine

Table 5-38. Intel 5000X Chipset Error List (Sheet 3 of 9)

ERR # in MCH	Error Name	Definition	Error Type	Log Register	Cause / Actions
DMA8	Read Data error	The DMA channel sets this bit indicating that a read could not be completed (for example, starvation).	Fatal	Log FERR_CHANERR/ NERR_CHANERR/ FERR_CHANCMD, FERR_DESC_CTRL, FERR_SADDR, FERR_DADDR, FERR_TRANSFER_SIZE, FERR_NADDR, FERR_CHANCMP NERR_CHANCMD, NERR_DESC_CTRL, NERR_SADDR, NERR_DADDR, NERR_TRANSFER_SIZE, NERR_NADDR, NERR_CHANCMP by taking a snap shot of the respective register fields when an error is met. The SADDR and DADDR will be the runtime addresses that the DMA engine is executing	Halt DMA engine
DMA9	Write Data error	The DMA channel sets this bit indicating that a write was unable to be completed at the destination (for example, no space available in DM).	Fatal		Halt DMA engine
DMA10	Descriptor Control Error	The DMA channel sets this bit indicating that the current descriptor has an illegal control field value in the “desc_control” field.	Fatal		Halt DMA engine
DMA11	Descriptor length Error	The DMA channel sets this bit indicating that the current transfer has an illegal length field value	Fatal		Halt DMA engine
DMA12	Completion Address Error	The DMA channel sets this bit indicating that the completion address register was configured to an illegal address	Fatal		Halt DMA engine
DMA13	Interrupt Configuration Error	The DMA channel sets this bit indicating that the interrupt related registers were not configured properly and an interrupt could not be generated	Fatal		Halt DMA engine
IO0	PCI Express - Data Link Layer Protocol Error	MCH detects a DL layer protocol error from the DLLP.	Default =Fatal (Check UNCERRSEV)	Log PEX_FAT_FERR/NERR or PEX_NF_COR_FERR/ NERR based on their respective Error types and Severity (UNCERRSEV) Log RPERRSTS for IO1, IO11 and IO17. Log UNCERRSTS for their respective Error Types. Log the first error pointer for UNCERRSTS in AERRCAPCTRL. Log CORRERSTS for their respective Error Types. Log PEXDEVSTS for IO12 and other I/O errors based on UNCERSEV(	Log Header of DLLP Packet. Check corresponding bit in UNCERRSEV register for severity level (Fatal or Non Fatal)
IO1	PCI Express - Received Fatal Error Message	MCH received a Fatal error message from the south bridge.	Fatal		Log header of packets with errors
IO2	PCI Express - Received Unsupported Request	Received an unsupported request, similar to master abort.	Default =UnCorr (Check UNCERRSEV)		Log header of packet Check corresponding bit in UNCERRSEV register for severity level (Fatal or Non Fatal)
IO4	PCI Express - Poisoned TLP	Received a poisoned transaction layer packet from the South Bridge.	Default =UnCorr (Check UNCERRSEV)		Log header of packets with errors Check corresponding bit in UNCERRSEV register for severity level (Fatal or Non Fatal)
IO5	PCI Express - Flow Control Protocol Error	MCH has detected a PCI Express Flow Control Protocol Error	Default =Fatal (Check UNCERRSEV)		Log header of packets with errors Check corresponding bit in UNCERRSEV register for severity level (Fatal or Non Fatal)

Table 5-38. Intel 5000X Chipset Error List (Sheet 4 of 9)

ERR # in MCH	Error Name	Definition	Error Type	Log Register	Cause / Actions
IO6	PCI Express - Completion Time-out	Pending transaction was ACKed in the data link layer but not within the time limit.	Default =UnCorr (Check UNCERRSEV)	Log PEX_FAT_FERR/NERR or PEX_NF_COR_FERR/ NERR based on their respective Error types and Severity (UNCERRSEV) Log RPERRSTS for IO1, IO11 and IO17. Log UNCERRSTS for their respective Error Types. Log the first error pointer for UNCERRSTS in AERRCAPCTRL. Log CORRERSTS for their respective Error Types. Log PEXDEVSTS for IO12 and other I/O errors based on UNCERSEV	Log header of packets with errors Check corresponding bit in UNCERRSEV register for severity level (Fatal or Non Fatal)
IO7	PCI Express - Completer Abort	Received return CA status for horrible error on the component. This is equivalent to a target abort on PCI.	Default =UnCorr (Check UNCERRSEV)		Log header of packets with errors Check corresponding bit in UNCERRSEV register for severity level (Fatal or Non Fatal)
IO8	PCI Express - Unexpected Completion Error	Received a Completion RequestorID that matches the requestor but the Tag does not match any pending entries.	Default =UnCorr (Check UNCERRSEV)		Log header of packets with errors Check corresponding bit in UNCERRSEV register for severity level (Fatal or Non Fatal)
IO9	PCI Express - Malformed TLP	Received a transaction layer packet that does not follow the TLP formation rules.	Default =UnCorr (Check UNCERRSEV)		Log header of packets with errors Check corresponding bit in UNCERRSEV register for severity level (Fatal or Non Fatal)
IO10	PCI Express - Receive Buffer Overflow Error	Receiver gets more data or transactions than credits allow.	Default =Fatal (Check UNCERRSEV)		Log header of packets with errors Check corresponding bit in UNCERRSEV register for severity level (Fatal or Non Fatal)
IO11	PCI Express - Received NonFatal Error Message	MCH received a NonFatal error message from the south bridge.	UnCorr		Log header of packets with errors
IO12	PCI Express - Receiver Error	Log header of packets with errors	Corr		Log header of packets with errors
IO13	PCI Express - Bad TLP Error	Received bad CRC or a bad sequence number in a transport layer packet.	Corr		Log header of packets with errors
IO14	PCI Express - BAD DLLP	Received bad CRC in a data link layer packet.	Corr		Log header of packets with errors
IO15	PCI Express - Replay_Num Rollover	Replay maximum count for the Retry Buffer has been exceeded.	Corr		Log header of packets with errors
IO16	PCI Express - Replay Timer Time-out	Replay timer timed out waiting for an Ack or Nak DLLP.	Corr		Log header of packets with errors
IO17	PCI Express - Received Correctable Error Message	MCH received a correctable error message from the south bridge.	Corr		Log header of packets with errors
IO18	ESI reset time-out	Did not receive ESI CPU_Reset_Done_Ack or CPU_Reset_Done_Ack_Secrets messages within T_{10max} after assertion of processor RESET# while PWRGOOD was asserted	Fatal	Log PEX_FAT_FERR/NERR	Deassert processor RESET#. Necessary to prevent processor thermal runaway.

Table 5-38. Intel 5000X Chipset Error List (Sheet 5 of 9)

ERR # in MCH	Error Name	Definition	Error Type	Log Register	Cause / Actions
B1	MCH -Parity Error from DM (Do not Include Poisoned Data)	MCH detected internal DM parity error. (This error was not generated by receiving bad data from an external interface)	Fatal	FERR_FAT_INT/ NERR_FAT_INT and NRECINT	log DM Entry on FERR.
B2	MCH -Multi-Tag Hit from snoop filter on any SF lookup port	MCH detected multiple hits in the SF lookup on any SF lookup port	Fatal	FERR_FAT_INT/ NERR_FAT_INT and NRECSF	Log, Hit/Miss, Set, Tag, State and Presence vector on FERR.
B3	MCH-Coherency Violation Error	MCH detected a cache coherency protocol error.	Fatal	FERR_FAT_INT/ NERR_FAT_INT NRECINT and NRECBF	Log CE entry on FERR
B4	Virtual Pin Interface Error	MCH detected an error on the virtual pin interface	Fatal	FERR_FAT_INT/ NERR_FAT_INT and NRECINT	
B5	MCH-Address Map Error	MCH detected address mapping error due to software programming error. The errors are described in system address map chapter.	UnCorr	FERR_NF_INT/ NERR_NF_INT and NRECINT	MCH might malfunction.
B6	Single bit ECC error on snoop filter lookup	MCH detected a hit in SF lookup and the entry has a single bit ECC error, or MCH detected a miss in SF lookup and the victim entry has a single bit error.	Corr	FERR_NF_INT/ NERR_NF_INT and RECSF	Log, Hit/Miss, Set, Tag, State and Presence vector on FERR.
B7	Multiple bit ECC error on snoop filter lookup	MCH detected a multiple ECC error in any of the ways during snoop filter lookup	Fatal	FERR_FAT_INT/ NERR_FAT_INT and NRECSF	Log, Hit/Miss, Set, Tag, State and Presence vector on FERR.
B8	Write Post Queue Parity Error	Intel 5000X Chipset MCH detected a cache coherency protocol error for a BIL. Any requestor from the bus that issued BIL not present in the SF.	Non Fatal	FERR_FAT_INT/ NERR_FAT_INT NRECINT and NRECBF	Log CE entry on FERR This applies to SF enable mode only
M1	Alert on FB-DIMM Replay or Fast Reset Time-out	The MCH detected an “Alert” on a non-redundant replay or hit a time-out on non-redundant fast reset before normal completion	Fatal	FERR_FAT_FBD NERR_FAT_FBD NRECMEM NRECFGLOG	Memory read: poison to requestor, update NRECMEM Configuration read: master-abort to requestor, update CFGLOG All others: drop.
M2	Northbound CRC error on FB-DIMM Replay	The MCH detected a northbound CRC error on a replay	Fatal	FERR_FAT_FBD NERR_FAT_FBD NRECMEM NRECFGLOG NRECFBD	Memory read: poison to requestor, update NRECMEM Configuration read: master-abort to requestor, update CFGLOG All others: drop.
M3	Tmid thermal event with intelligent throttling disabled	Intelligent throttling is disabled and the thermal sensor transitions from “below Tmid” to “above Tmid”.	Fatal	FERR_FAT_FBD NERR_FAT_FBD	
M4	Uncorrectable Data ECC Error on FB-DIMM Replay	The MCH detected an uncorrectable data ECC error during replay of the head of the FB-DIMM replay queue	Uncorr	FERR_NF_FBD NERR_NF_FBD	Poison to requestor. Don’t log error again... it was logged when the replay was launched.

Table 5-38. Intel 5000X Chipset Error List (Sheet 6 of 9)

ERR # in MCH	Error Name	Definition	Error Type	Log Register	Cause / Actions
M5	Aliased Uncorrectable Non-Mirrored Demand Data ECC Error	The MCH determined that a normally “correctable” error could be an aliased (x8 only) full device failure plus an additional single bit error.	Rec	FERR_NF_FBD NERR_NF_FBD RECMEM REDMEM UERRCNT	Re-read once. If ECC is uncorrectable with good CRC after re-read, then poison the data in memory and to the requestor. If correctable after re-read, then correct the data in memory and to the requestor.
M6	Aliased Uncorrectable Mirrored Demand Data ECC Error	In mirrored mode, the MCH determined that a normally “correctable” error could be an aliased (x8 only) full device failure plus an additional single bit error.	Rec	FERR_NF_FBD NERR_NF_FBD RECMEM REDMEM UERRCNT	First redundant read to branch X fails. MCH performs a fast reset on both branches X and Y. If both branches pass, then replay on branch Y. If branch X fails the disable branch X and replay on branch Y. If both branches fail or branch Y fails disable branch X and poison data. Under these conditions we get an M1 error. Second redundant read to branch X fails with an uncorrectable error. Perform fast reset and disable branch X and replay on branch Y.
M7	Aliased Uncorrectable Spare-Copy Data ECC Error	During a Sparing copy read from the failing DIMM the MCH determined that a normally “correctable” error could be an aliased (x8 only) full device failure plus an additional single bit error.	Rec	FERR_NF_FBD NERR_NF_FBD RECMEM REDMEM UERRCNT	Re-read once. If ECC is uncorrectable with good CRC after re-read, then poison the data in the spare DIMM or the off-line branch. If correctable after re-read, then correct the data in the spare DIMM or the off-line branch.
M8	Aliased Uncorrectable Patrol Data ECC Error	During a Patrol Scrub, the MCH determined that a normally “correctable” error could be an aliased (x8 only) full device failure plus an additional single bit error.	Rec	FERR_NF_FBD NERR_NF_FBD RECMEM REDMEM UERRCNT	The patrol read is dropped.
M9	Non-Aliased Uncorrectable Non-Mirrored Demand Data ECC Error	The MCH detected uncorrectable data with good CRC.	Rec	FERR_NF_FBD NERR_NF_FBD RECMEM UERRCNT	Re-read once. If ECC is uncorrectable with good CRC after re-read, then poison the data in memory and to the requestor. If correctable after re-read, then correct the data in memory and to the requestor. Does not include poisoned northbound data.

Table 5-38. Intel 5000X Chipset Error List (Sheet 7 of 9)

ERR # in MCH	Error Name	Definition	Error Type	Log Register	Cause / Actions
M10	Non-Aliased Uncorrectable Mirrored Demand Data ECC Error	In mirrored mode, the MCH detected uncorrectable or poisoned data with good CRC.	Rec	FERR_NF_FBD NERR_NF_FBD RECMEM UERRCNT	First redundant read to branch X fails. MCH performs a fast reset on both branches X and Y. If both branches pass, then replay on branch Y. If branch X fails the disable branch X and replay on branch Y. If both branches fail or branch Y fails disable branch X and poison data. Under these conditions we get an M1 error. Second redundant read to branch X fails with an uncorrectable error. Perform fast reset and disable branch X and replay on branch Y
M11	Non-Aliased Uncorrectable Spare-Copy Data ECC Error	The MCH detected uncorrectable data with good CRC from the failing DIMM rank during a sparing copy.	Rec	FERR_NF_FBD NERR_NF_FBD RECMEM UERRCNT	Re-read once. If ECC is uncorrectable with good CRC after re-read, then poison the data in the spare DIMM or the off-line branch. If correctable after re-read, then correct the data in the spare DIMM or the off-line branch. Does not include poisoned northbound data.
M12	Non-Aliased Uncorrectable Patrol Data ECC Error	During a patrol scrub, the MCH detected uncorrectable data with good CRC.	Rec	FERR_NF_FBD NERR_NF_FBD RECMEM UERRCNT	The patrol read is dropped.
M13	Non-Retry or redundant FB-DIMM Memory Alert, or redundant fast reset time-out	The MCH detected an “Alert” or corrupted write acknowledgement on the first attempt at an FB-DIMM memory access packet or on the replay of a redundant FB-DIMM memory access packet.	Rec	FERR_NF_FBD NERR_NF_FBD RECMEM	Non-redundant or 1st attempt: Fast reset and Initiate replay. Redundant replay or fast reset time-out: auto-degrade.
M14	Non-Retry FB-DIMM Configuration Alert	The MCH detected an “Alert” or corrupted write acknowledgement on the first attempt at an FB-DIMM configuration write command or on the replay of a redundant FB-DIMM configuration write command	Rec	FERR_NF_FBD NERR_NF_FBD CFGLOG	1st attempt: Fast reset and Initiate replay. Redundant replay: auto-degrade.
M15	Non-Retry FB-DIMM Northbound CRC error on read data	The MCH detected a northbound CRC error on the first attempt at a configuration or memory read or on the replay of a redundant configuration or memory read.	Rec	FERR_NF_FBD NERR_NF_FBD RECFBD	1st redundant memory read: re-read once from other image. Replay redundant memory read: auto-degrade All others: fast reset and initiate replay from the same image, branch, or channel.

Table 5-38. Intel 5000X Chipset Error List (Sheet 8 of 9)

ERR # in MCH	Error Name	Definition	Error Type	Log Register	Cause / Actions
M17	Correctable Non-Mirrored Demand Data ECC Error.	The MCH detected correctable data.	Corr	FERR_NF_FBD NERR_NF_FBD RECMEM REDMEM CERRCNT BADCNT BADRANK	Correct the data in memory and to the requestor.
M18	Correctable Mirrored Demand Data ECC Error	The MCH detected correctable data.	Corr	FERR_NF_FBD NERR_NF_FBD RECMEM REDMEM CERRCNT BADCNT BADRANK	Correct the data in memory and to the requestor.
M19	Correctable Spare-Copy Data ECC Error	The MCH detected correctable data from the failing DIMM rank during a sparing copy.	Corr	FERR_NF_FBD NERR_NF_FBD RECMEM REDMEM CERRCNT BADCNT BADRANK	Correct the data in the spare DIMM or the off-line branch.
M20	Correctable Patrol Data ECC Error	During a patrol scrub, the MCH detected correctable data.	Corr	FERR_NF_FBD NERR_NF_FBD RECMEM REDMEM CERRCNT BADCNT BADRANK	Correct the data in memory.
M21	FB-DIMM Northbound CRC error on FB-DIMM Sync Status	The MCH detected a northbound CRC error on a Sync Status	Corr	FERR_NF_FBD NERR_NF_FBD RECFBD	Drop. If sync was issued to prepare a fast reset for alert recovery then replay any queued configuration command destined for an alerting DIMM or a DIMM with a corrupted status CRC. **WARNING:** Possible double DIMM configuration command execution may incur undesirable side-effects.
M22	SPD protocol Error	The MCH detected an SPD interface error.	Corr	FERR_NF_FBD NERR_NF_FBD	Successive correction attempts performed by software.
~~M23~~	~~Tlow rising thermal event with intelligent throttling enabled~~	~~Intelligent throttling is enabled and the thermal sensor transitions from "below Tlow" to "above Tlow".~~	~~Corr~~	~~FERR_NF_FBD NERR_NF_FBD~~	~~Engage open-loop activation throttling~~
~~M24~~	~~Tmid rising thermal event with intelligent throttling enabled~~	~~Intelligent throttling is enabled and the thermal sensor transitions from "below Tmid" to "above Tmid".~~	~~Corr~~	~~FERR_NF_FBD NERR_NF_FBD~~	~~Engage closed-loop activation throttling~~
~~M25~~	~~Tlow falling thermal event with intelligent throttling enabled~~	~~Intelligent throttling is enabled and the thermal sensor transitions from "above Tlow" to "below Tlow".~~	~~Corr~~	~~FERR_NF_FBD NERR_NF_FBD~~	~~Disengage throttling~~
~~M26~~	~~Tmid falling thermal event with intelligent throttling enabled~~	~~Intelligent throttling is enabled and the thermal sensor transitions from "above Tmid" to "below Tmid".~~	~~Corr~~	~~FERR_NF_FBD NERR_NF_FBD~~	~~Disengage closed-loop throttling and engage open-loop throttling~~

Table 5-38. Intel 5000X Chipset Error List (Sheet 9 of 9)

ERR # in MCH	Error Name	Definition	Error Type	Log Register	Cause / Actions
M27	DIMM-Spare Copy start	Triggered DIMM-Spare copy	Corr	FERR_NF_FBD NERR_NF_FBD	Start DIMM-spare copy
M28	DIMM-Spare Copy complete	DIMM-Spare copy completed normally	Corr	FERR_NF_FBD NERR_NF_FBD	No Action

§

6 Testability

6.1 JTAG Port

Each component in the Intel 5000X Chipset includes a Test Access Port (TAP) slave which complies with the IEEE 1149.1 (JTAG) test architecture standard. Basic functionality of the 1149.1- compatible test logic is described here, but this document does not describe the IEEE 1149.1 standard in detail. For this, the reader is referred to the published standard1, and to the many books currently available on the subject.

6.1.1 JTAG Access to Configuration Space

JTAG has become a name that is synonymous with the IEEE 1149.1 test access port (TAP). Besides the boundary scan capabilities for low speed buses and pins, it provides an inexpensive serial interface port to up/download data to and from the chip. Throughout this document any reference to JTAG will imply the test access port (TAP) and the private chains that it is connected too, unless specifically mentioning the boundary scan attributes.

The feature described here is a JTAG private data chain that initiate a configuration request to the components configuration arbitration logic. During platform debug it is helpful to have a back door access to register space to determine correct configuration states. The In-Target Probe (ITP) provides an effective observation capability that links the hardware and the user together to examine and control a number of DFT and debug features.

Access to a component's configuration space must be non-blocking to a JTAG initiated configuration request to the Intel 5000X Chipset MCH's register space. Since the Intel 5000X Chipset MCH can source configuration transactions to other components and an errant configuration transaction that could potentially hang the system and prevent a JTAG access to the Intel 5000X Chipset MCH's configuration space. An additional chain is provided to ensure the ITP tool has unconditional access privilege to the Intel 5000X Chipset MCH in case there are configuration transaction hangs from another source.

6.1.2 TAP Signals

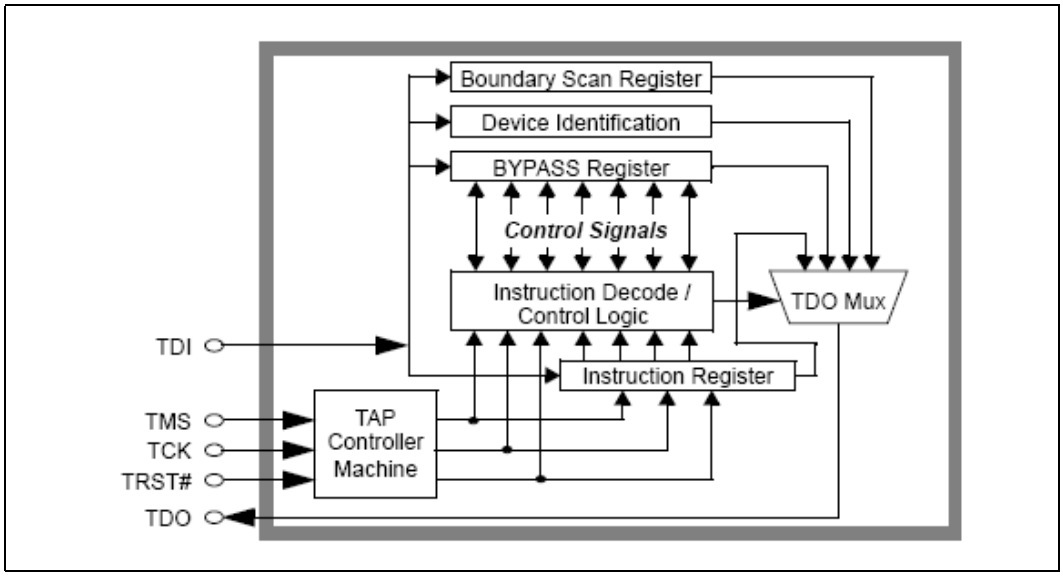
The TAP logic is accessed serially through 5 dedicated pins on each component as shown in Table 6-1.

Table 6-1. TAP Signal Definitions

TCK	TAP Clock input
TMS	Test Mode Select. Controls the TAP finite state machine.
TDI	Test Data Input. The serial input for test instructions and data.
TDO	Test Data Output. The serial output for the test data.
TRST#	Test Reset input.

TMS, TDI and TDO operate synchronously with TCK (which is independent of all other clocks). TRST# is an asynchronous reset input signal. This 5-pin interface operates as defined in the 1149.1 specification. A simplified block diagram of the TAP used in the Intel 5000X Chipset components is shown in Figure 6-1.

Figure 6-1. Simplified TAP Controller Block Diagram

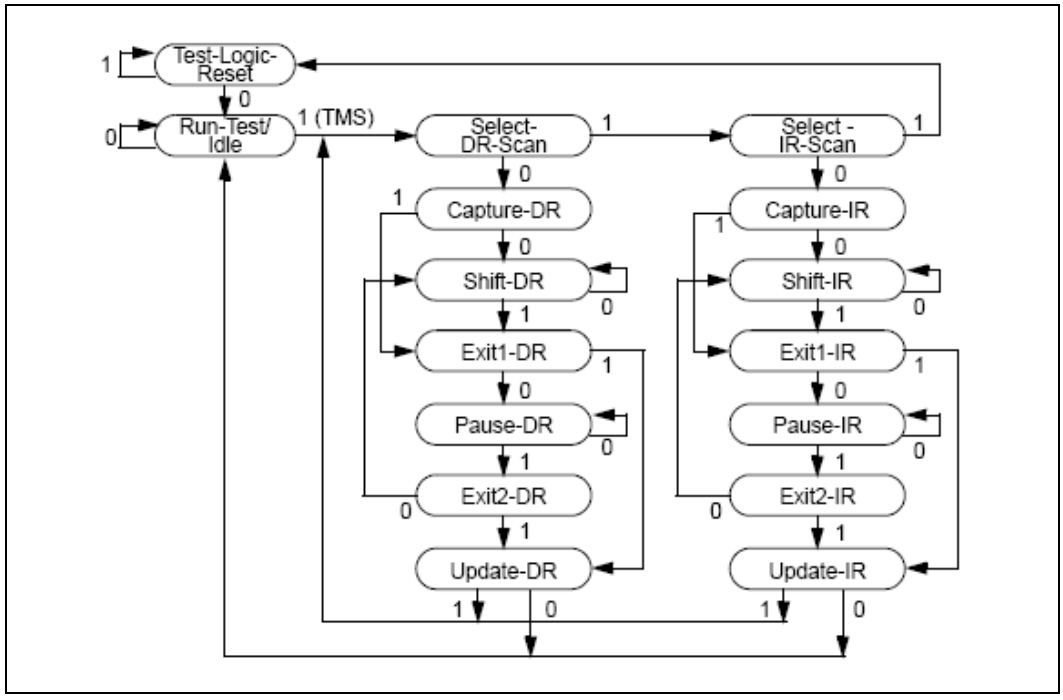


The TAP logic consists of a finite state machine controller, a serially-accessible instruction register, instruction decode logic and data registers. The set of data registers includes those described in the 1149.1 standard (the bypass register, device ID register, and so forth.), plus Intel 5000X Chipset-specific additions.

6.1.3 Accessing the TAP Logic

The TAP is accessed through an 1149.1-compliant TAP controller finite state machine, which is illustrated in Figure 6-1. The two major branches represent access to either the TAP Instruction Register or to one of the component-specific data registers. The TMS pin controls the progress through the state machine. TAP instructions and test data are loaded serially (in the Shift-IR and Shift-DR states, respectively) using the TDI pin. A brief description of the controller's states follows; refer to the IEEE 1149.1 standard for more detailed descriptions.

Figure 6-2. TAP Controller State Machine



The following list describes the behavior of each state in the TAP.

Test-Logic-Reset: In this state, the test logic is disabled so that normal operation of the device can continue unhindered. The instruction in the Instruction Register is forced to IDCODE. The controller is guaranteed to enter Test- Logic-Reset when the TMS input is held active for at least five clocks. The controller also enters this state immediately when TRST# is pulled active. The TAP controller cannot leave this state as long as TRST# is held active.

Run-Test/Idle: The TAP idle state. All test registers retain their previous values.

Capture-IR: In this state, the shift register contained in the Instruction Register loads a fixed value (of which the two least significant bits are “01”) on the rising edge of TCK. The parallel, latched output of the Instruction Register (“current instruction”) does not change.

Shift-IR: The shift register contained in the Instruction Register is connected between TDI and TDO and is shifted one stage toward its serial output on each rising edge of TCK. The output arrives at TDO on the falling edge of TCK. The current instruction does not change.

Pause-IR: Allows shifting of the Instruction Register to be temporarily halted. The current instruction does not change.

Update-IR: The instruction which has been shifted into the Instruction Register is latched onto the parallel output of the Instruction Register on the falling edge of TCK. Once the new instruction has been latched, it remains the current instruction until the next Update-IR (or until the TAP controller state machine is reset).

Capture-DR: In this state, the Data Register selected by the current instruction may capture data at its parallel inputs.

Shift-DR: The Data Register connected between TDI and TDO as a result of selection by the current instruction is shifted one stage toward its serial output on each rising edge of TCK. The output arrives at TDO on the falling edge of TCK. The parallel, latched output of the selected Data Register does not change while new data is being shifted in.

Pause-DR: Allows shifting of the selected Data Register to be temporarily halted without stopping TCK. All registers retain their previous values.

Update-DR: Data from the shift register path is loaded into the latched parallel outputs of the selected Data Register (if applicable) on the falling edge of TCK. This and Test-Logic-Reset are the only controller states in which the latched paralleled outputs of a data register can change.

All other states are temporary controller states, used to advance the controller between active states. During such temporary states, all test registers retain their prior values.

6.1.4 Reset Behavior of the TAP

The TAP and its related hardware are reset by transitioning the TAP controller finite state machine into the Test-Logic-Reset state. Once in this state, all of the reset actions listed in Figure 6-2 are performed. The TAP is completely disabled upon reset (that is, by resetting the TAP, the device will function as though the TAP did not exist).

Table 6-2. TAP Reset Actions

TAP Logic Affected	TAP Reset State Action	Related TAP Instructions (instr equivalent to reset is highlighted)
TAP instruction register	IDCODE	—
Boundary scan logic	Disabled	EXTEST
TDO pin	Tri-stated	—

The TAP can be transitioned to the Test-Logic-Reset state in one of two ways:

- Assert the TRST# pin at any time. This asynchronously resets the TAP controller.
- Hold the TMS pin high for 5 consecutive cycles of TCK. This is guaranteed to transition the TAP controller to the Test-Logic-Reset state on a rising edge of TCK.

Cycling power on a device does not ensure that the TAP is reset. System designers must utilize one of the two methods stated above to reset the TAP. The method used depends on the manufacturing and debug requirements of the system.

6.1.5 Clocking the TAP

There is no minimum frequency at which the Intel 5000X Chipset TAP will operate. Because the private chains are synchronized to the local core clock of that chain there is a maximum rate relative to the core that the interface can operate. The ratio is 12:1 providing a maximum rate of 27 MHz for a core frequency of 333 MHz.

6.1.6 Accessing the Instruction Register

Figure 6-3 shows the (simplified) physical implementation of the TAP instruction register. This register consists of a 7-bit shift register (connected between TDI and TDO), and the actual instruction register (which is loaded in parallel from the shift register). The parallel output of the TAP instruction register goes to the TAP instruction decoder.

Figure 6-3. TAP Instruction Register

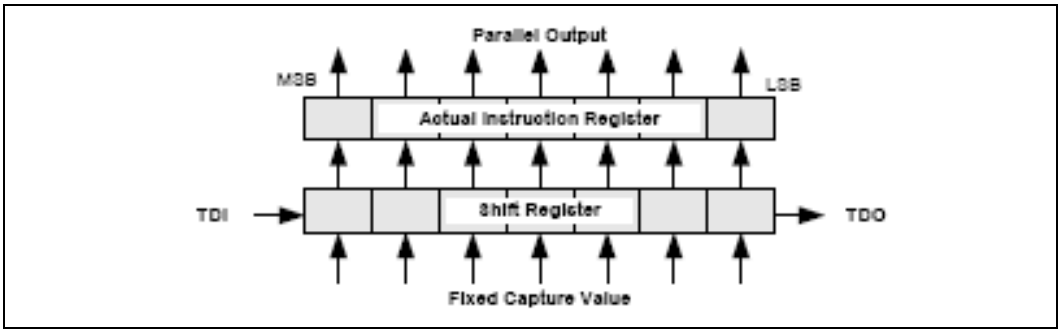


Figure 6-4 shows the operation of the instruction register during the Capture-IR, Shift-IR and Update-IR states. Shaded areas indicate the bits that are updated. In Capture-IR, the shift register portion of the instruction register is loaded in parallel with the fixed value “0000001”. In Shift-IR, the shift register portion of the instruction register forms a serial data path between TDI and TDO. In Update-IR, the shift register contents are latched in parallel into the actual instruction register. Note that the only time the outputs of the actual instruction register change is during Update-IR. Therefore, a new instruction shifted into the TAP does not take effect until the Update-IR state is visited.

Figure 6-4. TAP Instruction Register Operation

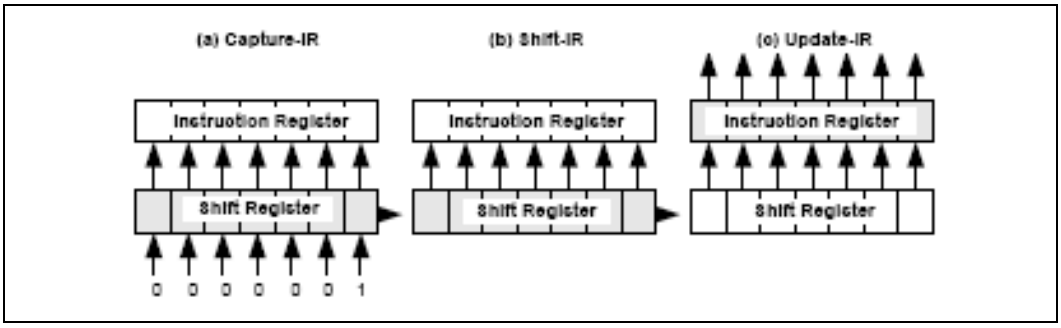
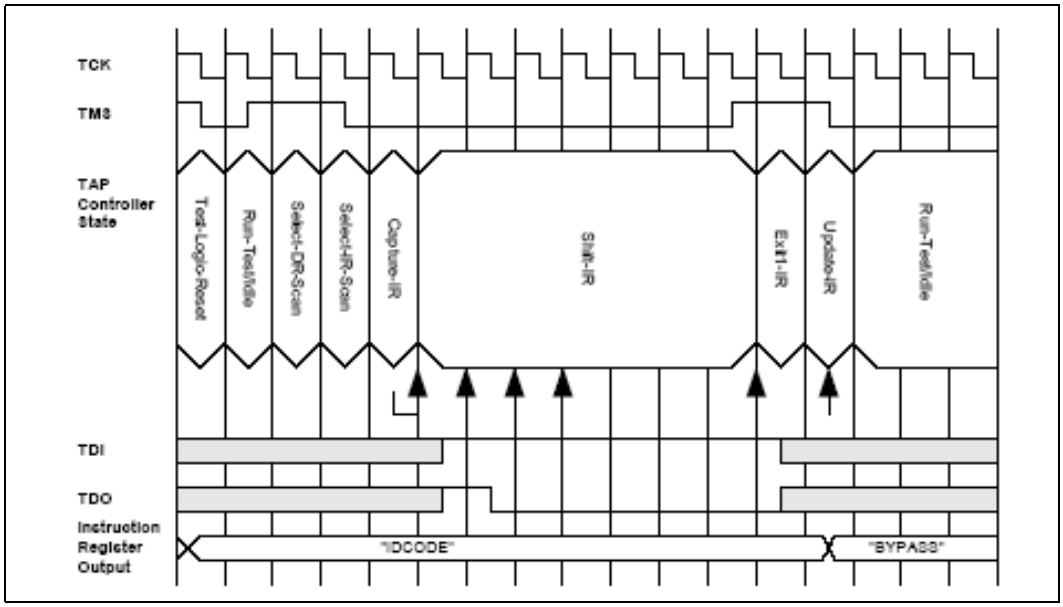


Figure 6-5 illustrates the timing when loading the BYPASS instruction (opcode 1111111b) into the TAP instruction register. Vertical arrows on the figure show the specific clock edges on which the Capture-IR, Shift-IR and Update-IR actions actually take place. Capture-IR (which preloads the instruction register with 0000001b) and Shift-IR operate on rising edges of TCK, and Update- IR (which updates the actual instruction register) takes place on the falling edge of TCK.

Figure 6-5. TAP Instruction Register Access



6.1.7 Accessing the Data Registers

The test data registers in the Intel 5000X Chipset components are architected in the same way as the instruction register, with components (that is, either the “capture” or “update” functionality) removed from the basic structure as needed. Data registers are accessed just as the instruction register is, only using the “select-DR-scan” branch of the TAP finite state machine in Table 6-2. A specific data register is selected for access by each TAP instruction. Note that the only controller states in which data register contents actually change are Capture-DR, Shift-DR, Update-DR and Run-Test/ Idle. For each of the TAP instructions described below, therefore, it is noted what operation (if any) occurs in the selected data register in each of these four states.

6.1.8 Public TAP Instructions

Table 6-3 contains descriptions of the encoding and operation of the public TAP instructions. There are four 1149.1-defined instructions implemented in the Intel 5000X Chipset devices. These instructions select from among three different TAP data registers – the boundary scan, device ID, and bypass registers. The public instructions can be executed with only the standard connection of the JTAG port pins. This means the only clock required will be TCK. Full details of the operation of these instructions can be found in the 1149.1 standard. The opcodes are 1149.1-compliant, and are consistent with the Intel-standard encodings. A brief description of each instruction follows. For more thorough descriptions refer to the IEEE 1149.1 specification.

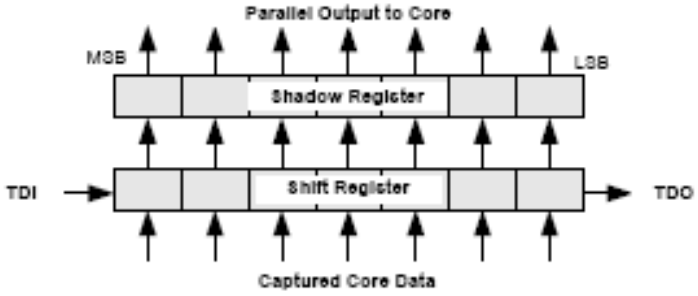
Table 6-3. Public TAP Instructions

Instruction	Encoding	Data Register Selected	Description
BYPASS	11111111	Boundary Scan	The BYPASS command selects the Bypass register, a single bit register connected between the TDI and TDO pins. This allows more rapid movement of test data to and from other components in the system.
EXTEST	00000000	Boundary Scan	The EXTEST instruction allows circuitry or wiring external to the devices to be tested. Boundary Scan register cells at outputs are used to apply stimulus, while Boundary Scan register cells at inputs are used to capture data.
SAMPLE/ PRELOAD	00000001	Boundary Scan	The SAMPLE/PRELOAD instruction is used to allow scanning of the Boundary Scan register without causing interference to the normal operation of the device. Two functions can be performed by use of the SAMPLE/PRELOAD instruction: 1. SAMPLE allows a snapshot of the data flowing into and out of the device to be taken without affecting the normal operation of the device. 2. PRELOAD allows an initial pattern to be placed into the Boundary Scan register cells. This allows initial known data to be present prior to the selection of another Boundary Scan test operation.
IDCODE	0000010	IDCODE	The IDCODE instruction is forced into the parallel output latches of the instruction register during the Test-Logic-Tap state. This allows the Device Identification register to be selected by manipulation of the broadcast TMS and TCK signals for testing purposes, as well as by a conventional instruction register scan operation.
CLAMP	0000100	Bypass	This allows static “guarding” values to be set into components that are not specifically being tested while maintaining the Bypass register as the serial path through the device.
HIGHZ	0001000	Bypass	The HIGHZ instruction is used to force all outputs of the device (except TDO) into a high impedance state. This instruction shall select the Bypass register to be connected between TDI and TDO in the Shift-DR controller state.

6.1.9 Public Data Instructions

This section describes the data registers that are accessed by the public and private instructions. Data shifts into all chains through the MSB of the data register as shown in Figure 6-6 which is the same as the instruction register.

Figure 6-6. TAP Data Register



6.1.10 Public Data Register Control

Table 6-4 define the actions that occur in the selected data register in controller states that can alter data register contents. If a TAP state does not affect the selected data register, then the corresponding table entry will be blank. Not all data registers have a parallel output latch. All data registers have a parallel input latch. Several table entries are still under investigation.

Table 6-4. Actions of Public TAP Instructions During Various TAP States

Instruction	Capture-DR	Shift-DR	Update-DR
Bypass	Reset Bypass Register	Shift Bypass register	
HighZ	Reset Bypass Register	Shift Bypass register	
IDcode	Load device ID into register	Shift ID register	
Extest	Load input pin values into Boundary Scan shift register	Shift Boundary Scan shift register	Load Boundary Scan shift register into Boundary Scan register; drive pins accordingly
Sample/Preload	Load pin values into Boundary Scan shift register	Shift Boundary Scan shift register	Load Boundary Scan shift register into Boundary Scan register

6.1.11 Bypass Register

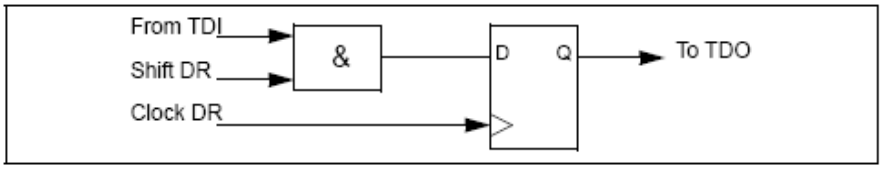
This register provides the minimal length path between TDI and TDO. It is loaded with a logical 0 during the Capture-DR state. The Bypass Register is a single bit register and is used to provide a minimum-length serial path through the device. This allows more rapid movement of test data to and from other components in the system. When in Bypass Mode, the operation of the test logic shall have no effect on the operation of the devices normal logic. Refer to Figure 6-7 for an implementation example.

6.1.11.1 Bypass Register Definition

JTAG encode: 1111111

Bit	Attr	Default	Description
1	RW	0	Bypass: a one bit register used to bypass the chip for board testing.

Figure 6-7. Bypass Register Implementation



6.1.12 Device ID Register

This register contains the device identification code in the format shown in Table 6-5. Three fields are predefined as the version number (stepping number), the manufacturer's identification code, and a logical 1 field. The component identification field is sub-divided into 3 fields. The Product Segment field identifies if the component is intended for CPU, laptop, desktop, server, and so forth. Product Type further defines

the component within a segment by stating it to be a CPU, memory, chipset, and so forth. The last field is a sequential component number assignment. This value will be maintained as sequential as possible depending on when each component's request was satisfied in the corporate database.

Table 6-5. Intel 5000X Chipset Device ID Codes

Device	Version	Component Identification Fields			Manufacturing ID	"1"	Entire Code (hex)
		Product Segment	Product Type	Component Number			
	4	6	5	5	11	1	32
Intel 5000X Chipset MCH – A0	0000	000100	01000	01000	00000001001	1	0x0118013

6.1.12.1 Device ID Register

JTAG encode: 0000010			
Bit	**Attr**	**Default**	**Description**
31:28	R	0000	Version: This number changes for each stepping including metal "dash" steppings. The most significant 2 bits are the stepping number: 00 A-step; 01 B-step, 10 C-step, and 11 D-step. The least significant 2 bits is the revision within a stepping.
27:22	R	000100	Product Segment: Number assigned that determines the market segment into which this component belongs. Since this format is new, the value for chipset is shown with others as an example. R&D:000 000 CPU:100 000 Desktop:010 000 Laptop:001 000 Server:000 100 and so forth.
21:17	R	01000	Product Type: Number assigned to further define the component within the market segment. Since this format is new, the value for chipset is shown with others as an example. Test:00 000 CPU:10 010 Memory:00 100 Modem:00 101 Chipset:01 000 and so forth.
16:12	R	Listed in next column	Component Number: Sequential listing based on request to database. Intel 5000X Chipset MCH:01000b
11:1	R	00000001001	Manufacturing ID: This number is assigned to Intel.
0	R	1	'1'

6.1.13 Boundary Scan Register

The following requirements apply to those interfaces that continue to support boundary scan (bscan) or the miscellaneous I/O signals.

- Each signal or clock pin (with the exception of the TAP specific pins TCK, TDI, TDO, TMS, and TRST#) will have an associated Boundary-Scan Register Cell. Differential Driver or Receiver Pin Pairs that cannot be used independently shall be considered

a single pin (that is, one Boundary-Scan Register Cell after the differential receiver).

- Internal Signals which control the direction of I/O pins shall also have associated Boundary- Scan Register Cells.
- Each Output pin (with the exception of TDO) shall be able to be driven to a tristate condition for HIGHZ test.

6.2 Extended Debug Port (XDP)

The Extended Debug Port is covered in the XDP Design Guide.

6.3 Intel Interconnect BIST Support

Intel Interconnect BIST (Intel IBIST) is an advanced method of detecting and diagnosing interconnect circuit faults. Intel IBIST covers both DC and AC faults, and operates at the full bandwidth of the interface being tested. As bus speeds increase beyond 500 MHz additional testability features such as Boundary Scan (IEEE 1149.1) (a virtual bus interconnect test methodology) will become useless due to frequency and timing issues. Further, the board/system fault spectrum associated with high-speed system buses has expanded beyond simple opens/shorts due to limited tolerance to transmission-line loss, impedance discontinuities, return path discontinuities, ISI, crosstalk, power supply collapse, nonlinear driver effects, non optimum VOH, VOL levels, non-ideal termination, and non-centered VREF.

Intel IBIST is an on die technology that tests the IO buffer register, driver, package interconnect, circuit board interconnect, receiver package interconnect, receiver and receiver register. Intel IBIST uses a built in pattern generator that operates off of internal or programmed patterns. The on die implementation of Intel IBIST is customized to the interface being tested. Tailoring Intel IBIST to the interface being tested allows optimization of Intel IBIST to test the fault spectrum of the interface. This customization includes the programming of built-in bit pattern test vectors for the interface under test.

The on-die IBIST architecture is designed for use in system level bus testing to assist platform electrical verification and platform HVM. The goals of the IBIST technology are:

- Increase platform level test coverage on Intel IBIST enabled high performance buses,
- Reduce platform test time
- Facilitate platform level test access and availability
- Enable advancements in platform management.

Intel IBIST has significant time to market and reliability benefits from a validation perspective, as it provides:

- Reduces test pattern development time
- Reduces complex system configuration time, by enabling concurrent traffic without the bus protocol and driver limitations
- Reduces debug time, more deterministic way to isolate faults

The Intel IBIST architecture is designed to detect complex high frequency faults, it also provides simple opens/shorts detection.

The Intel IBIST methodology is a comprehensive bus level testing solution, which covers:

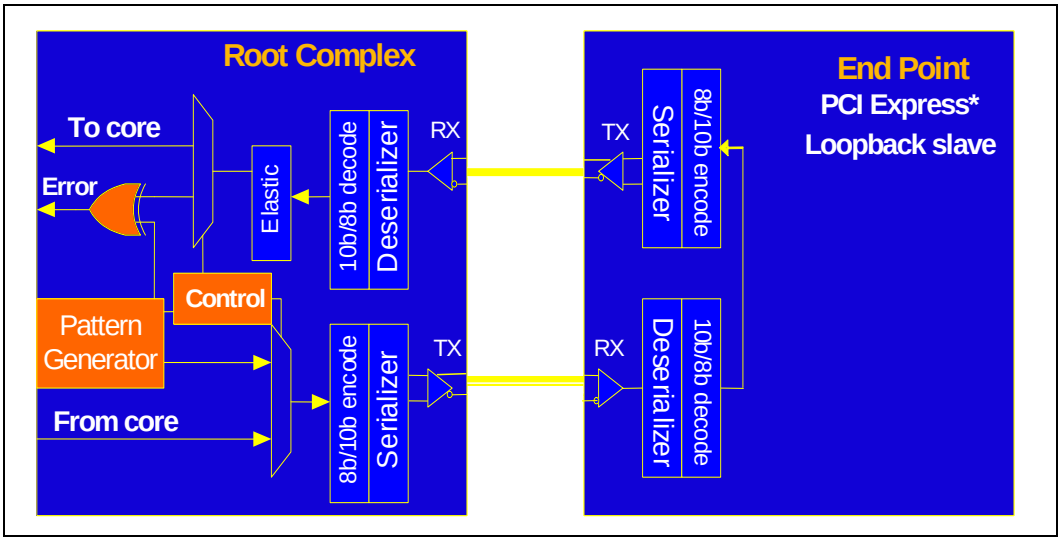
- Component-to-component
- Component-to-adapter
- Examination of entire high speed interconnect bus topologies (from I/O drivers to PCB traces characteristics to receivers and power delivery sub-system)

The Intel 5000X Chipset MCH supports Intel IBIST on all PCI Express and Fully Buffered DIMM interfaces. Intel 5000X Chipset MCH IBIST is controlled by Intel IBIST registers discussed in sections Section 3.9.26 and Section 3.11.13.

6.3.1 PCI Express Intel IBIST

The Intel IBIST architecture utilizes the same component interconnect and timing paths as normal operation, output and input latches are inside the Intel IBIST loop. Precise control of patterns is now possible due to the fact that Intel IBIST operation is independent of system bus protocol. The Intel IBIST architecture is located in the physical layer of the PCI Express Architecture. The Link and Transaction layers are isolated from the physical layer during Intel IBIST testing.

Figure 6-1. PCI Express Intel IBIST Architecture

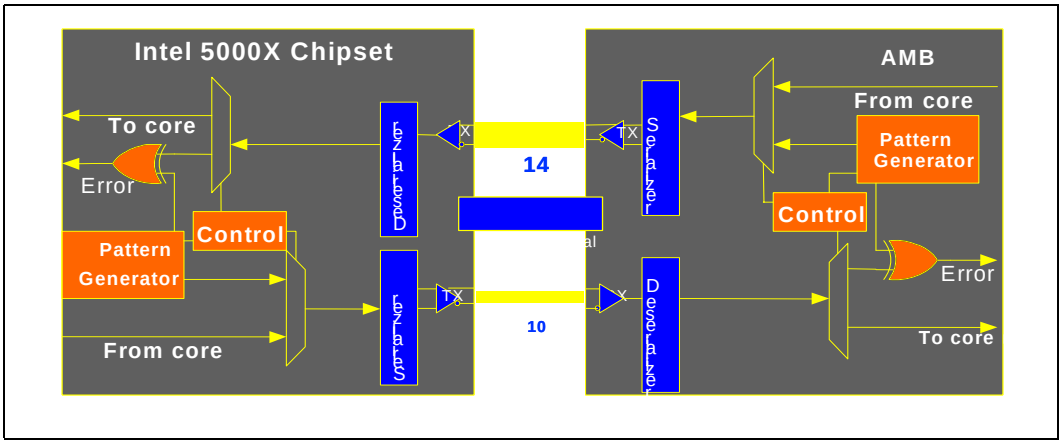


6.3.2 FB-DIMM Intel IBIST

The FB-DIMM lane is a non-symmetrical link, Intel 5000X Chipset MCH is a master in the southbound link and AMB is the master on the northbound link. The FB-DIMM uses Intel IBIST to verify the integrity of the link between the host controller and the AMB. The host controller includes one Intel IBIST engine (generator and checker), and the AMB includes two Intel IBIST engines, one for northbound and one for southbound links. FB-DIMM Intel IBIST supports both loopback and master to master topology.

- Intel 5000X Chipset MCH sends test pattern through the southbound FB-DIMM link, the AMB receives the test pattern and compares it against the expected pattern.
- AMB sends test pattern through the northbound FB-DIMM link, the Intel 5000X Chipset MCH receives the test pattern and compares it against the expected pattern.

Figure 6-2. FB-DIMM Intel IBIST Architecture



Please refer to the *Intel IBIST Functional Specification* for detail information on Intel IBIST.

§

7 Electrical Characteristics

This chapter provides the absolute maximum ratings and DC Characteristics for the Intel 5000X Chipset MCH.

7.1 Absolute Maximum Ratings

Table 7-1 lists the maximum environmental stress ratings for the Intel 5000X Chipset MCH. Functional operation at or exceeding the absolute maximum and minimum ratings is neither implied nor guaranteed. Functional operating parameters are listed in the AC tables.

Warning: Stressing the device beyond the "Absolute Maximum Ratings" may cause permanent damage. These are stress ratings only. Operating beyond the "operating conditions" is not recommended and extended exposure beyond "operating conditions" may affect reliability.

Table 7-1. Absolute Maximum Ratings

Symbol	Parameter	Min	Max	Unit
$T_{storage}$	Storage Temperature	-40.0	85.0	°C
Vcc	MCH Supply Voltage with respect to Vss	-0.50	1.85	V
V_{TT}	FSB Termination Supply Voltage input with respect to Vss	-0.30	1.85	V

7.1.1 Thermal Characteristics

For information on thermal characteristics, consult the *Intel® 5000P/5000V/5000X Chipset Memory Controller Hub (MCH) Thermal/Mechanical Design Guidelines.*

7.1.2 Power Characteristics

Table 7-2. Operating Condition Power Supply Rails

Symbol	Parameter	AC Min	DC Min	Nom	DC Max	AC Max	Unit	Notes
V_{TT}	Host AGTL+ Termination Voltage	1.140	1.164	1.20	1.236	1.260	V	1,2
I_{TT}	Host AGTL+ Termination Current			2.6	4.8		A	
VCC	1.5 V MCH Supply Voltage	1.425	1.455	1.53	1.575	1.605	V	1, 2
ICC	1.5 V MCH Supply Current			17.6	22.0		A	3
ICC	1.5 V MCH Supply Current			19.3	24.1		A	4
ICC	1.5 V MCH Supply Current			15.3	19.1		A	5
AMB VCC	1.5 V FBD Supply Voltage	1.425	1.455	1.5	1.575	1.605	V	6
AMB/DRAM	1.8 V DDR2/AMB VDD	1.701	1.755	1.8	1.845	1.899	V	6
DRAM	0.9 V DDR2 VDD/2	0.8505	0.8775				V	6
Other	3.3 V Supply Voltage	3.1185	3.2175	3.3	3,3825	3,4815	V	6

Notes:
1. Under no circumstances may the supply voltage go past the AC min/max window. The supply voltage may go outside the DC min/max window for transient events,
2. The supply voltage must stay within the DC min/max window in a static system (no active switching). The DC window only assumes voltage regulator ripple and motherboard induced noise.
3. Intel 5000P Chipset MCH with 4 active FB-DIMM channels. Total Core + I/O current drawn off the 1.5 V rail.
4. Intel 5000X Chipset MCH with 4 active FB-DIMM channels. Total Core + I/O current drawn off the 1.5 V rail.
5. Intel 5000V Chipset MCH with 2 active FB-DIMM channels. Total Core + I/O current drawn off the 1.5 V rail.
6. Tolerances are specified at the AMB DRAM package(s) and FB-DIMM voltage regulator designs should be scaled accordingly to support these tolerances for the 1.5 V and 1.8 V rails, DC min/max per the JEDEC specification. AC tolerances from VR loop BW ~30 kHz to 1MHz.

Table 7-3. Analog and Bandgap Voltage and Current Specifications

Symbol	Parameter	Min	Nom	Max	Unit	Notes
VCCA	Analog MCH Supply Voltage	1.4055	1.5	1.545	V	1
ICCA	Analog MCH Supply Current			28.9	mA	
FSBVCCA	Analog PLL Voltage	1.4055	1.5	1.545	V	1
FSBICCA	Analog PLL Current			28.9	mA	
FBDVCCA	Analog FBD Voltage	1.4055	1.5	1.545		
FBDICCA	Analog FBD Current			52	mA	
PEVCCA	Analog PCI Express Voltage	1.4055	1.5	1.545	V	1
PEICCA	Analog PCI Express Current			48	mA	1
PEVCCBG	Analog PCI Express Bandgap Voltage	2.425	2.5	2.575	V	1
PEICCBG	Analog PCI Express Bandgap Current			600	µA	1

1. The analog voltage is intended to be a filtered copy of its associated supply voltage. Refer to the *Dual-Core Intel® Xeon® Processor-Based Servers Platform Design Guide (PDG)* for the recommended implementation and frequency response requirements of each filter.

7.2 DC Characteristics

This section documents the DC characteristics of the MCH. The specifications are split into five sections:

- Clocks
- FSB Interface
- FB-DIMM (Fully Buffered DIMM) Memory Interface[1]
- PCI Express/ ESI Interface
- Miscellaneous Interface
  - — SMBus Interface
  - — JTAG Interface

1. Refer to the *Intel® 6400/6402 Advanced Memory Buffer External Design Specification (EDS) Addendum* for additional details on the FB-DIMM interface.

7.2.1 Clock DC Characteristics

Table 7-4. Clock DC Characteristics

Symbol	Signal Group	Parameter	Min	Nom	Max	Unit	Notes
333 MHz FSB Clock (CORECLKN / CORECLKP)							
V_{IL}	(h)	Input Low Voltage	-0.150	0	0.150	V	1
V_{IH}	(h)	Input High Voltage	0.660	0.700	0.850	V	
$V_{CROSS(abs)}$	(h)	Absolute Crossing Point	0.250		0.550	V	2, 7
$V_{CROSS(rel)}$	(h)	Relative Crossing Point	0.250 + 0.5 x (V_{Havg} – 0.700)		0.550 - 0.5 x (0.700 - V_{Havg})	V	7, 8
ΔV_{CROSS}	(h)	Range of Crossing Points			0.140	V	
V_{OS}	(h)	Overshoot			V_{IH} + 0.300	V	3
V_{US}	(h)	Undershoot	-0.300			V	4
V_{RBM}	(h)	Ringback Margin	0.200			V	5
V_{TR}	(h)	Threshold Region	V_{CROSS} – 0.100		V_{CROSS} + 0.100	V	6
266 MHz FSB Clock (CORECLKN / CORECLKP)							
V_{IL}	(h)	Input Low Voltage	-0.150	0	0.150	V	1
V_{IH}	(h)	Input High Voltage	0.660	0.700	0.850	V	
$V_{CROSS(abs)}$	(h)	Absolute Crossing Point	0.250		0.550	V	2, 7
$V_{CROSS(rel)}$	(h)	Relative Crossing Point	0.250 + 0.5 x (V_{Havg} – 0.700)		0.550 - 0.5 x (0.700 - V_{Havg})	V	7, 8
ΔV_{CROSS}	(h)	Range of Crossing Points			0.140	V	
V_{OS}	(h)	Overshoot			V_{IH} + 0.300	V	3
V_{US}	(h)	Undershoot	-0.300			V	4
V_{RBM}	(h)	Ringback Margin	0.200			V	5
V_{TR}	(h)	Threshold Region	V_{CROSS} – 0.100		V_{CROSS} + 0.100	V	6
100 MHz PCI Express Clock (PECLKN / PECLKP)							
V_{IL}	(q)	Input Low Voltage	-0.150	0		V	
V_{IH}	(q)	Input High Voltage	0.660	0.700	0.850	V	
$V_{CROSS(abs)}$	(q)	Absolute Crossing Point	0.250		0.550	V	2, 7
$V_{CROSS(rel)}$	(q)	Relative Crossing Point	0.250 + 0.5 x (V_{Havg} – 0.700)		0.550 + 0.5 x (V_{Havg} – 0.700)	V	7, 8
ΔV_{CROSS}	(q)	Range of Crossing Points			0.140	V	1, 2
V_{OS}	(q)	Overshoot			V_{IH} + 0.300	V	3
V_{US}	(q)	Undershoot	-0.300			V	4
V_{RBM}	(q)	Ringback Margin	0.200			V	5
V_{TR}	(q)	Threshold Region	V_{CROSS} – 0.100		V_{CROSS} + 0.100	V	6
133/167 MHz FB-DIMM Clock (FBDxxCLKN/ FBDxxCLKP)							
V_{IL}	(k)	Input Low Voltage	-0.150	0		V	
V_{IH}	(k)	Input High Voltage	0.660	0.700	0.850	V	
$V_{CROSS(abs)}$	(k)	Absolute Crossing Point	0.250		0.550	V	2, 7
$V_{CROSS(rel)}$	(k)	Relative Crossing Point	0.250 + 0.5 x (V_{Havg} – 0.700)		0.550 + 0.5 x (V_{Havg} – 0.700)	V	7, 8

Notes:
1. Refer to chapter 5 of *Intel® 5000P Chipset/Intel® 5000V Chipset and Intel® 5000X Chipset External Design Specification (EDS) Addendum*.
2. Crossing voltage is defined as the instantaneous voltage when the rising edge of CORECLKP is equal to the falling edge of CORECLKN.
3. Overshoot is defined as the absolute value of the maximum voltage.
4. Undershoot is defined as the absolute value of the minimum voltage.
5. Ringback Margin is defined as the absolute voltage difference between the maximum Rising Edge Ringback and the maximum Falling Edge Ringback. Both maximum Rising and Falling Ringbacks should not cross the threshold region.
6. Threshold Region is defined as a region centered around the crossing point voltage in which the differential receiver switches. It includes input threshold hysteresis.
7. The crossing point must meet the absolute and relative crossing point specifications simultaneously.
8. VHavg (the average of V_{IH}) can be measured directly using “Vtop” on Agilent* scopes and “High” on Tektronix scopes.

7.2.2 FSB Interface DC Characteristics

Table 7-5. FSB Interface DC Characteristics

Symbol	Signal Group	Parameter	Min	Nom	Max	Unit	Notes
V_{IL}	• (a) (b)	Host AGTL+ Input Low Voltage	0		GTLREF – (0.1 × V_{TT})	V	1, 2
V_{IH}	• (a) (b)	Host AGTL+ Input High Voltage	GTLREF + (0.1 × V_{TT})		V_{TT}	V	1, 3
V_{OL}	• (a) (c)	Host AGTL+ Output Low Voltage			0.4	V	
V_{OH}	• (a) (c)	Host AGTL+ Output High Voltage	0.90 x V_{TT}		V_{TT}	V	4
I_{OL}	• (a) (c)	Host AGTL+ Output Low Current			V_{TT} / (0.50 x R_{tt_min} + R_{on_min})	mA	8
I_{LI}	• (a) (b)	Host AGTL+ Input Leakage Current	n/a		+/- 200	uA	5, 6
I_{LO}	• (a) (b)	Host AGTL+ Output Leakage Current	n/a		+/- 200	uA	5, 6
R_{on}	•	Buffer on Resistance	7		11	Ω	
GTLREF	• (e)	Host Bus Reference Voltage	(0.98 x 0.67) x V_{TT}	0.67 x V_{TT}	(1.02 x 0.67) x V_{TT}	V	1
R_{TT}	•	Host Termination Resistance Common Clock, Async on Stripline	45	50	55	Ω	7

Notes:
1. **GTLREF is equivalent to FSBxFSBVREF**. GTLREF is generated from V_{TT} on the baseboard by a voltage divider or 1% resistors.
2. V_{IL} is defined as the voltage range at a receiving agent that will be interpreted as an electrical low value.
3. V_{IH} is defined as the voltage range at a receiving agent that will be interpreted as an electrical high value.
4. V_{IH} and V_{OH} may experience excursions above V_{CC}. However, input signal drivers must comply with the signal quality specifications chapter in the document.
5. Leakage to VSS with land held at V_{TT}.
6. Leakage to V_{TT} with land held at 300 mV.
7. Use 50 ohm ±15% for all Microstrip.
8. $I_{OL}$is defined as current when Output Low. The formula computes the total current drawn by the driver from VR (Voltage Regulator). Half of the total current goes through RTT on the chipset, and another half goes through the RTT on the CPU (the End-Bus-Agency).

7.2.3 FB-DIMM DC Characteristics

Table 7-6. FB-DIMM Transmitter (Tx) Output DC Characteristics

Symbol	Signal Group	Parameter	Min	Nom	Max	Unit	Notes
V_{TX-CM_S}	(i) (j)	DC Common Mode Output Voltage for Small Voltage Swing	135		280	mV	1
V_{TX-CM_L}	(i) (j)	DC Common Mode Output Voltage for Large Voltage Swing			375	mV	1
V_{TX-SE}	(i) (j)	Single-ended Voltage	0		700	mV	
RLTX-Diff	(i) (j)	Differential Return Loss	-10			dB	
RLTX-CM	(i) (j)	Common Mode Return Loss	-6			dB	
$Z_{TX-MATCH-DC}$	(i) (j)	D+/D- TX Impedance Difference			4%	Ω	2
$Z_{TX-COM-ESI-IMP-DC}$	(i) (j)	D+/D- TX Common Mode High Impedance State	5		20	kΩ	

Notes:
1. Defined as: $V_{TX-CM} = DC_{(avg)}$ of $| V_{TX-D+} + V_{TX-D-} | / 2$
2. TX DC impedance matching between D+ and D- on a given lane.

Table 7-7. FB-DIMM Receiver (Rx) Output DC Characteristics

Symbol	Signal Group	Parameter	Min	Nom	Max	Unit	Notes
V_{RX-CM}	(i) (j)	DC Common Mode Input Voltage	190		400	mV	1
$Z_{RX-MATCH-DC}$	(i) (j)	D+/D- RX Impedance Difference			4%	Ω	2
RLRX-Diff	(i) (j)	Differential Return Loss	-10			dB	
RLRX-CM	(i) (j)	Common Mode Return Loss	-6			dB	

Notes:
1. DC (avg) of $| V_{RX-D+} + V_{RX-D-} | / 2$.
2. RX DC impedance matching between D+ and D- on a given lane.

7.2.4 PCI Express/ ESI Interface DC Characteristics

Table 7-8. PCI Express/ ESI Differential Transmitter (Tx) Output DC Characteristics

Symbol	Signal Group	Parameter	Min	Nom	Max	Unit	Notes
VTX-DIF-DC	(o) (p)	Differential Peak to Peak Output Voltage	0.8		1.2	V	2
VTX-CM-DC-ACTIVE-IDLE-DELTA	(o) (p)	Absolute Delta of DC Common Mode Voltage During L0 and Electrical Idle	0		100	mV	2
VTX-CM-DC-LINE-DELTA	(o) (p)	Absolute Delta of DC Common Mode Voltage between D+ and D-	0		25	mV	2
VTX-IDLE-DIFFp	(o) (p)	Electrical Idle Differential Peak Output Voltage			20	mV	2
VTX-RCV-DETECT	(o) (p)	The amount of voltage change allowed during Receiver Detection			600	mV	
VTX-DC-CM	(o) (p)	The TX DC Common Mode Voltage	0		3.6	V	2
ITX-SHORT	(o) (p)	The Short Circuit Current Limit			90	mA	
ZTX-DIFF-DC	(o) (p)	DC Differential TX Impedance	80	100	120	Ω	
ZTX-DC	(o) (p)	Transmitter DC Impedance	40			Ω	

Notes:
1. No test load is necessarily associated with this value.
2. Specified at the measurement point into a timing and voltage compliance test load and measured over any 250 consecutive TX UIs.

Table 7-9. PCI Express/ ESI Differential Receiver (Rx) Input DC Characteristics

Symbol	Signal Group	Parameter	Min	Nom	Max	Unit	Notes
ZRX-DIFF-DC	(o) (p)	DC Differential Input Impedance	80	100	120	Ω	5
ZRX-DC	(o) (p)	DC Input Impedance	40	50	60	Ω	2, 3
ZRX-High-Imp-DC	(o) (p)	Power Down DC Input Common Mode Impedance	200k			Ω	6
VRX-IDLE-DET-DIFFp	(o) (p)	Electrical Idle Detect Threshold	65		175	mV	

Notes:
1. No test load is necessarily associated with this value.
2. Specified at the measurement point and measured over any 250 consecutive UIs. If the clock to the RX and TX are not derived from the same reference clock, the TX UI recovered from 3500 consecutive UI must be used as a reference for the eye diagram.
3. A TRX-EYE=0.40UI provides for a total sum of 0.60 UI deterministic and random jitter budget for the Transmitter and interconnect collected any 250 consecutive UIs. The TRX-EYE-MEDIAN-to-MAX-JITTER specification ensures a jitter distribution in which the median and the maximum deviation from the median is less than half of the total.6 UI jitter budget collected over any 250 consecutive TX UIs. It should be noted that the median is not the same as the mean. The jitter median describes the point in time where the number of jitter points on either side is approximately equal as opposed to the averaged time value. If the clocks to the RX and TX are not derived from the same reference clock, the TX UI recovered from 3500 consecutive UI must be used as the reference for the eye diagram.
4. The Receiver input impedance shall result in a differential return loss greater than or equal to 15 dB with the D+ line biased to 300 mV and the D- line biased to -300 mV and a common mode return loss greater than or equal to 6 dB (no bias required) over a frequency range of 50 MHz to 1.25 GHz. This input impedance requirement applies to all valid input levels. The reference impedance for return loss measurements for is 50 ohms to ground for both the D+ and D- line (that is, as measured by a Vector Network Analyzer with 50 ohm probes). Note: that the series capacitors CTX is optional for the return loss measurement.

5. Impedance during all LTSSM states. When transitioning from a Fundamental Reset to Detect (the initial state of the LTSSM) there is a 5 ms transition time before Receiver termination values must be met on all un-configured Lanes of a Port.
6. The RX DC Common Mode Impedance that exists when no power is present or Fundamental Reset is asserted. This helps ensure that the Receiver Detect circuit will not falsely assume a Receiver is powered on when it is not. This term must be measured at 300 mV above the RX ground.

7.2.5 Miscellaneous DC Characteristics

Table 7-10. SMBus DC Characteristics

Symbol	Signal Group	Parameter	Min	Nom	Max	Unit	Notes
V_{IH}	(w)	Input High Voltage	2.1			V	
V_{IL}	(w)	Input Low Voltage			0.8	V	
V_{OL}	(w)	Output Low Voltage			0.4	V	1
I_{OL}	(w)	Output Low Current			4	mA	
I_{Leak}	(w)	Leakage Current			10	µA	
C_{Pad}	(w)	Pad Capacitance			10	pF	

Notes:
1. At Vol max, Iol = max.

Table 7-11. JTAG DC Characteristics

Symbol	Signal Group	Parameter	Min	Nom	Max	Unit	Notes
V_{IH}	(y)	Input High Voltage	0.9			V	
V_{IL}	(y)	Input Low Voltage			0.5	V	
V_{OL}	(z)	Output Low Voltage			0.4	V	
I_{Leak}	(y) (z)	Leakage Current			2.9	µA	

Table 7-12. 1.5 V CMOS DC Characteristics

Symbol	Signal Group	Parameter	Min	Nom	Max	Unit	Notes
V_{IH}	(d) (cc)	Input High Voltage	1.0		1.6	V	
V_{IL}	(d) (cc)	Input Low Voltage	-0.2		0.5	V	
V_{OH}	(cc)	Output High Voltage	1.1			V	
V_{OL}	(cc)	Output Low Voltage			0.4	V	
I_{Leak}	(cc)	Leakage Current			70	µA	
V_{ABS}	(d) (cc)	Input Damage Thresholds	-0.2		1.6	V	

Table 7-13. 3.3 V CMOS DC Characteristics (Sheet 1 of 2)

Symbol	Signal Group	Parameter	Min	Nom	Max	Unit	Notes
V_{IH}	(dd)	Input High Voltage	2.1			V	
V_{IL}	(dd)	Input Low Voltage			0.8	V	
V_{OH}	(ee)	Output High Voltage				V	

Table 7-13. 3.3 V CMOS DC Characteristics (Sheet 2 of 2)

Symbol	Signal Group	Parameter	Min	Nom	Max	Unit	Notes
V_{OL}	(ee)	Output Low Voltage			0.4	V	
I_{Leak}	(ee)	Leakage Current			10	µA	
V_{ABS}	(dd)	Input Damage Thresholds	-0.3		3.5	V	

§

8 Ballout and Package Information

8.1 Intel 5000X Chipset MCH Ballout

The following section presents preliminary ballout information for the Intel 5000X Chipset MCH. This ballout is subject to change and is to be used for informational purposes only.

Figure 8-1. Intel 5000X Chipset Quadrant Map

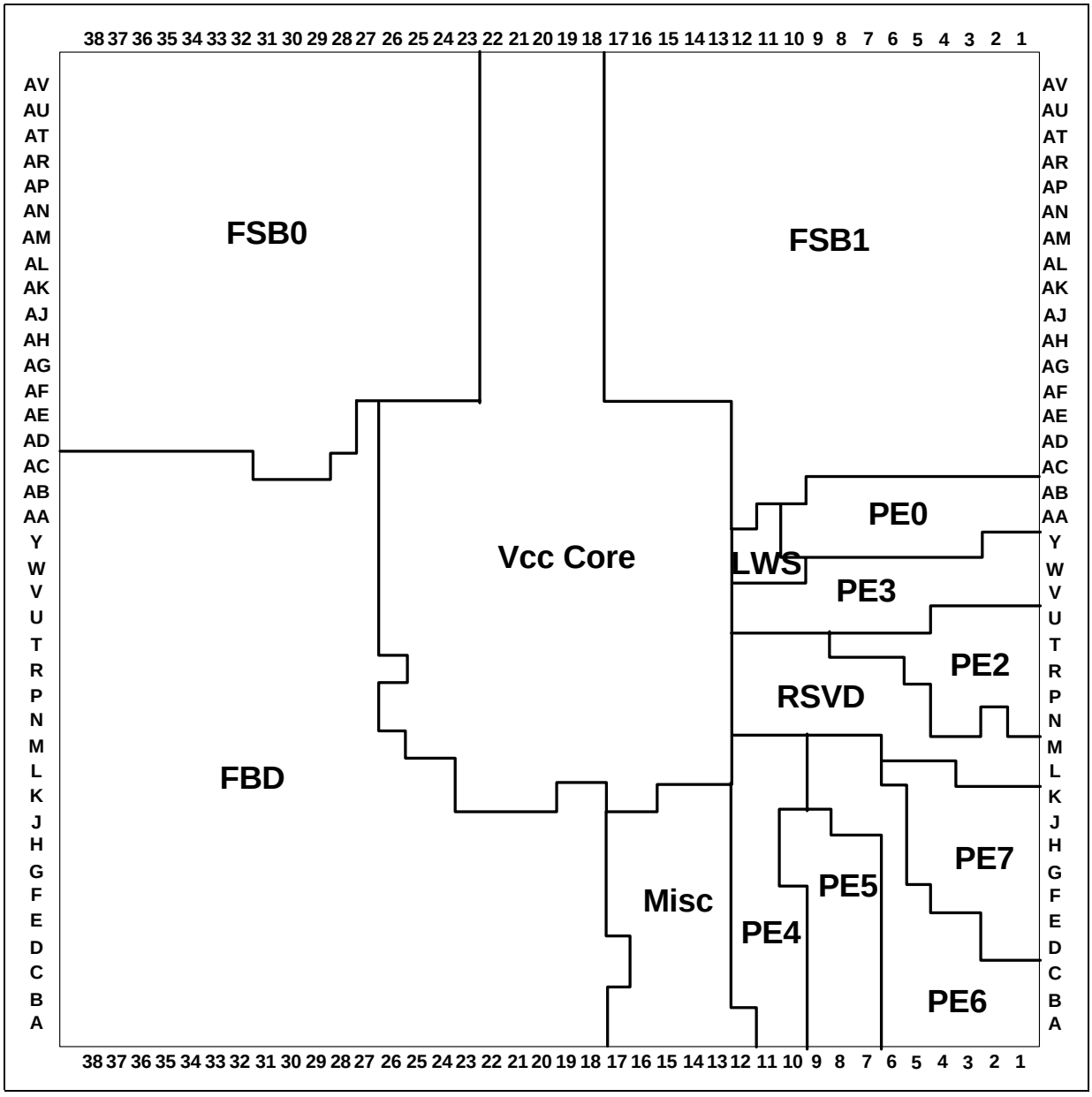


Figure 8-2. Intel 5000X Chipset MCH Ballout Left Side (Top View)

	38	37	36	35	34	33	32	31	30	29	28	27	26	
AV			VSS	VSS	FSB0DEFER#	FSB0HITM#	VSS	FSB0RS[0]#	FSB0BNR#	VSS	FSB0A[17]#	RSVD	VSS	AV
AU		VSS	FSB0D[21]#	FSBSLWCRES	FSB0BPRI#	VSS	FSB0HIT#	FSB0RS[2]#	VSS	FSB0ADS#	FSB0A[21]#	VSS	FSB0A[25]#	AU
AT	VSS	FSB0D[22]#	FSB0D[19]#	FSBCRES	VSS	FSB0DP[1]#	FSB0TRDY#	VSS	FSB0LOCK#	FSB0DRDY#	VSS	FSB0A[23]#	FSB0A[24]#	AT
AR	FSB0DSTBP[1]#	FSB0DSTBN[1]#	VSS	VSS	FSBODTCRES	RSVD	VSS	FSB0DP[0]#	FSB0DBSY#	VSS	FSB0BPM[4]#	FSB0A[19]#	VSS	AR
AP	FSB0D[27]#	FSB0DBI[1]#	FSB0D[23]#	FSB0D[18]#	FSB0D[20]#	VSS	RSVD	FSB0DP[2]#	VSS	FSB0BPM[5]#	RSVD	VSS	FSB0A[7]#	AP
AN	VSS	VSS	FSB0D[26]#	FSB0D[24]#	VSS	FSB0D[16]#	FSB0D[5]#	VSS	FSB0RESET#	FSB0DP[3]#	VSS	FSB0RSP#	FSB0A[6]#	AN
AM	FSB0D[28]#	FSB0D[31]#	VSS	VSS	FSB0D[29]#	FSB0D[12]#	VSS	RSVD	FSB0VREF	VSS	FSB0BREQ[0]#	FSB0VREF	VSS	AM
AL	FSB0D[50]#	FSB0D[52]#	FSB0D[51]#	FSB0D[25]#	FSB0D[17]#	VSS	FSB0DBI[0]#	FSB0D[7]#	VSS	FSB0D[1]#	FSB0RS[1]#	VSS	FSB0REQ[0]#	AL
AK	VSS	VSS	FSB0D[53]#	FSB0D[30]#	VSS	FSB0DSTBP[0]#	FSB0DSTBN[0]#	VSS	FSB0D[6]#	FSB0D[3]#	VSS	FSB0BINIT#	FSB0AP[1]#	AK
AJ	FSB0D[56]#	FSB0D[55]#	VSS	VSS	FSB0D[49]#	FSB0D[15]#	VSS	FSB0D[8]#	FSB0D[10]#	VSS	FSB0D[0]#	FSB0MCERR#	VSS	AJ
AH	FSB0D[57]#	FSB0DBI[3]#	FSB0D[61]#	FSB0DSTBP[3]#	FSB0DSTBN[3]#	VSS	FSB0D[9]#	FSB0D[11]#	VSS	FSB0D[4]#	FSB0D[2]#	VSS	FSB0AP[0]#	AH
AG	VSS	VSS	FSB0D[60]#	FSB0D[54]#	VSS	FSB0D[32]#	FSB0D[36]#	VSS	FSB0D[33]#	FSB0D[13]#	VSS	FSB0D[14]#	FSB0BREQ[1]#	AG
AF	FSB0D[59]#	FSB0D[48]#	VSS	VSS	FSB0VREF	FSB0D[37]#	VSS	FSB0D[35]#	FSB0DBI[2]#	VSS	FSB0D[45]#	VSS	VSS	AF
AE	FSB0D[58]#	FSB0D[63]#	FSB0D[62]#	VCCSF	FSB0D[34]#	VSS	FSB0D[39]#	FSB0D[40]#	VSS	FSB0D[43]#	FSB0D[47]#	VSS	VTT	AE
AD	VSS	VSS	VSS	FSB0D[38]#	VSS	FSB0DSTBP[2]#	FSB0DSTBN[2]#	VSS	FSB0D[41]#	FSB0D[46]#	VSS	VSS	VCCSF	AD
AC	VSS	FBD0SBON[6]	TESTHI	VSS	FBD0SBOP[8]	FBD0SBON[8]	VSS	FSB0D[44]#	FSB0D[42]#	VSS	VSS	VSS	VCCSF	AC
AB	FBD0SBON[5]	FBD0SBOP[6]	VSS	FBD0SBOP[7]	FBD0SBON[7]	VSS	FBD0NBIN[2]	FBD0NBIP[2]	VSS	VSS	VSS	VSS	VCCFBD	AB
AA	FBD0SBOP[5]	VSS	FBD0SBOP[9]	FBD0SBON[9]	VSS	FBD0SBON[2]	FBD0SBOP[2]	VSS	VSS	VSS	VSS	VCCFBD	VCCFBD	AA
Y	VSS	FBD0SBON[4]	FBD0SBOP[4]	VSS	FBD0SBOP[3]	FBD0SBON[3]	VSS	FBD0NBIN[1]	FBD0NBIP[1]	VSS	FBD0NBIN[0]	FBD0NBIP[0]	VCCFBD	Y
W	FBD0SBON[1]	VSS	VSS	FBD0NBIN[9]	FBD0NBIP[9]	VSS	FBD0SBOP[0]	FBD0SBON[0]	VSS	FBD0NBIN[5]	FBD0NBIP[5]	VSS	VCCFBD	W
V	FBD0SBOP[1]	VSS	FBD0NBIN[10]	FBD0NBIP[10]	VSS	FBD0NBIN[7]	FBD0NBIP[7]	VSS	FBD0NBIN[13]	FBD0NBIP[13]	VSS	FBD0NBIP[3]	VCCFBD	V
U	VSS	FBD0NBIN[11]	FBD0NBIP[11]	VSS	FBD0NBIN[8]	FBD0NBIP[8]	VSS	FBD0NBIN[12]	FBD0NBIP[12]	VSS	FBD0NBIP[4]	FBD0NBIN[3]	VCCFBD	U
T	FBD01CLKN	RSVD	VSS	FBD01VCCA	FBD01VSSA	VSS	FBD0NBIN[6]	FBD0NBIP[6]	VSS	VSS	FBD0NBIN[4]	VCCFBD	VCCFBD	T
R	FBD01CLKP	VSS	FBD1SBOP[6]	FBD1SBON[6]	VSS	FBD1SBOP[8]	FBD1SBON[8]	VSS	VSS	VSS	VSS	VSS	VSS	R
P	VSS	FBD1SBOP[5]	FBD1SBON[5]	VSS	FBD1SBOP[7]	FBD1SBON[7]	VSS	VSS	VSS	VSS	FBD1NBIN[0]	FBD1NBIP[0]	VCCFBD	P
N	FBD1SBOP[9]	FBD1SBON[9]	VSS	FBD1SBON[4]	FBD1SBOP[4]	VSS	FBD1NBIN[12]	VSS	VSS	FBD1NBIN[3]	FBD1NBIP[3]	VSS	VCCFBD	N
M	VSS	VSS	FBD1SBON[3]	FBD1SBOP[3]	VSS	FBD1NBIN[6]	FBD1NBIP[12]	VSS	FBD1NBIN[4]	FBD1NBIP[4]	VSS	FBD1NBIN[1]	FBD1NBIP[1]	M
L	FBD1SBON[2]	FBD1SBON[1]	FBD1SBOP[1]	VSS	FBD1NBIN[7]	FBD1NBIP[6]	VSS	FBD1NBIN[5]	FBD1NBIP[5]	VSS	FBD1NBIN[2]	FBD1NBIP[2]	VSS	L
K	FBD1SBOP[2]	VSS	VSS	FBD1NBIN[8]	FBD1NBIP[7]	VSS	FBD1NBIN[13]	FBD1NBIP[13]	VSS	VSS	VSS	VSS	VSS	K
J	VSS	FBD1SBON[0]	FBD1SBOP[0]	FBD1NBIP[8]	VSS	FBD2SBON[8]	FBD2SBOP[8]	VSS	VSS	VSS	VSS	VSS	VSS	J
H	FBD1NBIN[11]	FBD1NBIN[10]	FBD1NBIP[10]	VSS	FBD2SBON[7]	FBD2SBOP[7]	VSS	VSS	VSS	VSS	VSS	VSS	VSS	H
G	FBD1NBIP[11]	VSS	VSS	FBD2SBON[6]	FBD2SBOP[6]	VSS	VSS	VSS	VSS	FBD2SBON[0]	FBD2SBOP[0]	VSS	FBD3SBOP[1]	G
F	VSS	FBD1NBIN[9]	FBD1NBIP[9]	FBDICOMPBIAS	VSS	FBD2SBON[4]	FBD2SBOP[4]	VSS	FBD2SBON[1]	FBD2SBOP[1]	VSS	FBD23VSSA	VSS	F
E	FBD2NBIN[11]	FBDBGBIASEXT	FBDRESIN	VSS	FBD2SBON[9]	FBD2SBOP[9]	VSS	FBD2SBON[2]	FBD2SBOP[2]	VSS	FBD23CLKN	FBD23VCCA	VSS	E
D	FBD2NBIP[11]	FBD2NBIN[10]	VSS	FBD2SBON[5]	FBD2SBOP[5]	VSS	FBD2SBON[3]	FBD2SBOP[3]	VSS	RSVD	FBD23CLKP	VSS	FBD3NBIN[11]	D
C	VSS	FBD2NBIP[10]	FBD2NBIP[9]	VSS	FBD2NBIP[7]	VSS	VSS	FBD2NBIP[13]	VSS	VSS	FBD2NBIP[3]	VSS	VSS	C
B		VSS	FBD2NBIN[9]	FBD2NBIP[8]	FBD2NBIN[7]	FBD2NBIP[6]	FBD2NBIP[12]	FBD2NBIN[13]	FBD2NBIP[5]	FBD2NBIP[4]	FBD2NBIN[3]	FBD2NBIP[2]	FBD2NBIP[1]	B
A			VSS	FBD2NBIN[8]	VSS	FBD2NBIN[6]	FBD2NBIN[12]	VSS	FBD2NBIN[5]	FBD2NBIN[4]	VSS	FBD2NBIN[2]	FBD2NBIN[1]	A
	38	37	36	35	34	33	32	31	30	29	28	27	26	

Figure 8-3. Intel 5000X Chipset MCH Ballout Center (Top View)

	25	24	23	22	21	20	19	18	17	16	15	14	
AV	FSB0A[28]#	FSB0A[31]#	VSS	FSB0A[35]#	VTT	VTT	VTT	VTT	VSS	VSS	VSS	VSS	AV
AU	FSB0A[22]#	VSS	FSB0A[29]#	FSB0A[34]#	VTT	VTT	VTT	VTT	FSBVCCA	COREVSSA	VSS	VSS	AU
AT	VSS	FSB0A[27]#	FSB0A[30]#	VSS	VTT	VTT	VTT	VTT	COREVCCA	VSS	VSS	FSB1VREF	AT
AR	FSB0A[26]#	FSB0A[20]#	VSS	FSB0A[33]#	VTT	VTT	VTT	VTT	VSS	RSVD	RSVD	VSS	AR
AP	FSB0A[18]#	VSS	FSB0ADSTB[1]#	FSB0A[32]#	VTT	VTT	VTT	VTT	CORECLKN	FSB1D[55]#	VSS	FSB1D[47]#	AP
AN	VSS	FSB0A[4]#	RSVD	VSS	VTT	VTT	VTT	VTT	CORECLKP	VSS	FSB1D[53]#	FSB1D[50]#	AN
AM	FSB0A[5]#	RSVD	VSS	FSB0A[9]#	VTT	VTT	VTT	VTT	VSS	FSB1D[49]#	FSB1D[51]#	VSS	AM
AL	FSB0A[3]#	VSS	FSB0ADSTB[0]#	FSB0A[12]#	VTT	VTT	VTT	VTT	VCC	FSB1D[56]#	VSS	FSB1D[52]#	AL
AK	VSS	FSB0REQ[2]#	RSVD	VSS	VTT	VTT	VTT	VTT	RSVD	VSS	FSB1DSTBP[3]#	FSB1DSTBN[3]#	AK
AJ	FSB0REQ[4]#	FSB0REQ[3]#	VSS	FSB0A[11]#	VTT	VTT	VTT	VTT	VSS	FSB1D[61]#	FSB1D[60]#	VSS	AJ
AH	FSB0REQ[1]#	VSS	FSB0A[8]#	FSB0A[13]#	VTT	VTT	VTT	VTT	VSS	FSB1D[54]#	VSS	FSB1D[48]#	AH
AG	VSS	FSB0A[10]#	FSB0A[15]#	VSS	VTT	VTT	VTT	VTT	VSS	VSS	FSB1D[59]#	FSB1D[62]#	AG
AF	FSB0A[14]#	VSS	VSS	FSB0A[16]#	VTT	VTT	VTT	VTT	VSS	FSB1D[63]#	FSB1D[58]#	VSS	AF
AE	VTT	VTT	VTT	VTT	VTT	VTT	VTT	VTT	VTT	VTT	VTT	VTT	AE
AD	VTT	VTT	VTT	VTT	VTT	VTT	VTT	VTT	VTT	VTT	VTT	VTT	AD
AC	VCCSF	VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCC	VTT	AC
AB	VCCFBD	VCC	VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCCSF	AB
AA	VCCFBD	VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCC	VSS	AA
Y	VCCFBD	VCC	VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCCPE	Y
W	VCCFBD	VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCC	VCCPE	W
V	VCCFBD	VCC	VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCCPE	V
U	VCCFBD	VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCC	VCCPE	U
T	VCCFBD	VCC	VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCCPE	T
R	VCCFBD	VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCC	VCCPE	R
P	VCCFBD	VCC	VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCCPE	P
N	VCCFBD	VCCFBD	VCCFBD	VCCFBD	VCCFBD	VCCFBD	VCC	VSS	VCC	VSS	VCCPE	VCCPE	N
M	VCCFBD	VCCFBD	VCCFBD	VCCFBD	VCCFBD	VCCFBD	VSS	VCC	VCC	VCC	VCCPE	VCCPE	M
L	VSSSEN	VCCSEN	VCCFBD	VCCFBD	VCCFBD	VCCFBD	VCC	VCC	VCC	VCC	VCCPE	VCCPE	L
K	VSS	VSS	VCCFBD	VCCFBD	VCCFBD	VSS	FBD3SBOP[8]	FBD3SBON[8]	VSS	VSS	SPD1SMBDATA	CFGSMBCLK	K
J	VSS	FBD3SBOP[3]	FBD3SBON[3]	VSS	FBD3SBOP[5]	FBD3SBON[5]	VSS	FBD3SBON[7]	VSS	SPD1SMBCLK	XDPSLWCRES	CFGSMBDATA	J
H	FBD3SBOP[2]	FBD3SBON[2]	VCCFBD	FBD3SBOP[9]	FBD3SBON[9]	VCCFBD	FBD3SBON[6]	FBD3SBOP[7]	PWRGOOD	SPD3SMBDATA	SPD3SMBCLK	VSS	H
G	FBD3SBON[1]	VSS	FBD3SBOP[4]	FBD3SBON[4]	VSS	FBD3NBIP[0]	FBD3SBOP[6]	VSS	RESET#	TESTHI_V3REF	VSS	XDPODTCRES	G
F	VCCFBD	FBD3NBIN[9]	FBD3NBIP[9]	VSS	VSS	FBD3NBIN[0]	VSS	FBD3NBIP[1]	TESTHI_V3REF	VSS	SPD2SMBCLK	XDPCOMCRES	F
E	FBD3NBIN[10]	FBD3NBIP[10]	VCCFBD	FBD3SBON[0]	VSS	VCCFBD	FBD3NBIN[5]	FBD3NBIN[1]	VSS	XDPD[15]#	SPD2SMBDATA	XDPD[5]#	E
D	FBD3NBIP[11]	VSS	FBD3NBIP[7]	FBD3SBOP[0]	VSS	FBD3NBIP[13]	FBD3NBIP[5]	VSS	FBD3NBIN[2]	XDPD[14]#	XDPD[8]#	VSS	D
C	FBD2NBIP[0]	VSS	FBD3NBIN[7]	VSS	FBD3NBIP[12]	FBD3NBIN[13]	VSS	FBD3NBIN[3]	FBD3NBIP[2]	XDPD[12]#	VSS	XDPDSTBP#	C
B	FBD2NBIN[0]	FBD3NBIN[8]	VSS	FBD3NBIP[6]	FBD3NBIN[12]	VSS	FBD3NBIN[4]	FBD3NBIP[3]	VSS	VSS	XDPD[9]#	XDPD[7]#	B
A	VSS	FBD3NBIP[8]	VSS	FBD3NBIN[6]	VSS	VCCFBD	FBD3NBIP[4]	VSS	XDPRDY#	XDPD[11]#	XDPD[13]#	XDPD[10]#	A
	25	24	23	22	21	20	19	18	17	16	15	14	

Figure 8-4. Intel 5000X Chipset MCH Ballout Right Side (Top View)

	13	12	11	10	9	8	7	6	5	4	3	2	1	
AV	FSBSLWCTRL	FSB1D[40]#	VSS	FSB1D[39]#	FSB1D[37]#	VSS	FSB1D[32]#	FSB1D[9]#	VSS	FSB1D[10]#	VSS			AV
AU	VSS	VSS	FSB1DB[2]#	FSB1D[38]#	VSS	FSB1D[33]#	FSB1D[13]#	VSS	FSB1D[15]#	FSB1DSTBP[0]#	FSB1DSTBN[0]#	VSS		AU
AT	VSS	FSB1D[42]#	FSB1D[41]#	VSS	FSB1D[34]#	FSB1D[36]#	VSS	FSB1D[11]#	FSB1D[8]#	VSS	FSB1D[7]#	FSB1D[6]#	VSS	AT
AR	FSB1D[45]#	FSB1D[46]#	VSS	FSB1DSTBP[2]#	FSB1D[35]#	VSS	FSB1D[14]#	FSB1D[12]#	VSS	FSB1D[5]#	FSB1D[4]#	VSS	FSB1D[3]#	AR
AP	FSB1D[43]#	VSS	FSB1D[44]#	FSB1DSTBN[2]#	VSS	FSB1D[22]#	FSB1DBI[0]#	VSS	FSB1D[1]#	FSB1D[2]#	VSS	FSB1ADS#	FSB1D[0]#	AP
AN	VSS	FSB1D[25]#	FSB1D[27]#	VSS	FSB1D[24]#	FSB1D[23]#	VSS	FSB1D[18]#	FSB1VREF	VSS	FSB1BPM[5]#	FSB1BPM[4]#	VSS	AN
AM	FSB1D[28]#	FSB1D[26]#	VSS	FSB1DSTBP[1]#	FSB1D[21]#	VSS	RSVD	FSB1D[20]#	VSS	FSB1DBSY#	FSB1DRDY#	VSS	FSB1BREQ[1]#	AM
AL	FSB1D[30]#	VSS	FSB1D[29]#	FSB1DSTBN[1]#	VSS	FSB1D[17]#	FSB1D[16]#	VSS	FSB1RS[2]#	FSB1LOCK#	VSS	FSB1BREQ[0]#	FSB1RS[1]#	AL
AK	VSS	FSB1D[31]#	FSB1DB[1]#	VSS	FSB1D[19]#	FSB1HIT#	VSS	FSB1TRDY#	FSB1RS[0]#	VSS	FSB1BNR#	FSB1RSP#	VSS	AK
AJ	FSB1D[57]#	FSB1DP[1]#	VSS	FSB1BPRI#	FSB1DEFER#	VSS	FSB1HITM#	FSB1REQ[2]#	VSS	FSB1BINIT#	FSB1A[26]#	VSS	RSVD	AJ
AH	FSB1DBI[3]#	VSS	FSB1MCERR#	VCCSF	VSS	FSB1A[6]#	RSVD	VSS	FSB1A[19]#	FSB1VREF	VSS	FSB1A[24]#	FSB1A[25]#	AH
AG	VSS	FSB1AP[0]#	FSB1DP[0]#	FSB1AP[1]#	FSB1REQ[0]#	FSB1A[7]#	VSS	RSVD	FSB1A[18]#	VSS	FSB1ADSTB[1]#	RSVD	VSS	AG
AF	FSB1DP[3]#	FSB1DP[2]#	VSS	VSS	FSB1REQ[3]#	VSS	FSB1A[8]#	FSB1A[11]#	VSS	FSB1A[17]#	FSB1A[28]#	VSS	FSB1A[27]#	AF
AE	VTT	VSS	FSB1RESET#	FSB1REQ[4]#	VSS	RSVD	FSB1A[9]#	VSS	FSB1A[21]#	FSB1A[30]#	VSS	FSB1A[31]#	FSB1A[32]#	AE
AD	VTT	FSB1A[3]#	FSB1REQ[1]#	VSS	FSB1A[4]#	FSB1A[12]#	VSS	FSB1A[20]#	FSB1A[23]#	VSS	FSB1A[29]#	FSB1A[33]#	VSS	AD
AC	VTT	FSB1A[5]#	VSS	FSB1ADSTB[0]#	FSB1A[13]#	VSS	FSB1A[15]#	FSB1A[22]#	VSS	FSB1A[34]#	FSB1A[35]#	VSS	PSEL[0]	AC
AB	VCCSF	VSS	FSB1A[10]#	FSB1A[14]#	VSS	PE0RP[2]	PE0RN[2]	VSS	PE0TN[2]	PE0TP[2]	VSS	PSEL[1]	PSEL[2]	AB
AA	VCCSF	FSB1A[16]#	PEWIDTH[0]	VSS	PE0TN[3]	PE0TP[3]	VSS	PE0RN[3]	PE0RP[3]	VSS	PE0TP[1]	PE0TN[1]	VSS	AA
Y	VCCPE	PEWIDTH[1]	VSS	PE0RP[0]	PE0RN[0]	VSS	PE0TP[0]	PE0TN[0]	VSS	PE0RP[1]	PE0RN[1]	VSS	RSVD	Y
W	VCCPE	VCCPE	PEWIDTH[2]	PEWIDTH[3]	VSS	PE3RN[2]	PE3RP[2]	VCCPE	PE3TP[1]	PE3TN[1]	VSS	PE3RN[0]	RSVD	W
V	VCCPE	VSS	VSS	VSS	PE3TN[3]	PE3TP[3]	VSS	PE3RN[1]	PE3RP[1]	VSS	PE3TN[0]	PE3RP[0]	VSS	V
U	VCCPE	VSS	VSS	PE3RN[3]	PE3RP[3]	VCCPE	PE3TN[2]	PE3TP[2]	VSS	PE2TP[1]	PE3TP[0]	VCCPE	PE2RN[3]	U
T	VCCPE	VSS	VSS	VSS	VSS	PE2TP[0]	PE2TN[0]	VSS	PE2RP[0]	PE2TN[1]	VSS	PE2TN[3]	PE2RP[3]	T
R	VCCPE	PEICOMPI	PEVCCBG	VCCPE	RSVD	RSVD	VSS	RSVD	PE2RN[0]	VCCPE	PE2RN[2]	PE2TP[3]	VSS	R
P	VCCPE	PERCOMPO	VSS	RSVD	RSVD	VSS	RSVD	RSVD	VSS	PE2RN[1]	PE2RP[2]	VSS	PE2TN[2]	P
N	VCCPE	VCCPE	PEVSSBG	RSVD	VSS	RSVD	RSVD	VCCPE	RSVD	PE2RP[1]	VSS	RSVD	PE2TP[2]	N
M	VCCPE	RSVD	RSV1	VSS	PE6TP[0]	PE6TN[0]	VSS	RSVD	RSVD	VSS	RSVD	RSVD	VSS	M
L	VCCPE	GPIOSMBDATA	VSS	PE4RN[3]	VSSQUIET	VCCPE	PE7TN[2]	RSVD	VSS	PE7RN[3]	RSVD	VCCPE	PEVSSA	L
K	GPIOSMBCLK	VSS	PE4TN[3]	PE4RP[3]	VSS	PE6RN[3]	PE7TP[2]	VSS	PE7TN[3]	PE7RP[3]	VSS	PECLKN	PEVCCA	K
J	VSS	PE4TN[0]	PE4TP[3]	VCCPE	PE5TP[0]	PE6RP[3]	VSS	PE6TN[3]	PE7TP[3]	VSS	RSVD	PECLKP	VSS	J
H	SPD0SMBCLK	PE4TP[0]	VSS	PE5RP[0]	PE5TN[0]	VSS	PE5RN[3]	PE6TP[3]	VSS	PE7RN[2]	PE7RP[2]	VSS	RSVD	H
G	SPD0SMBDATA	VCCPE	PE4RP[1]	PE5RN[0]	VSS	PE5RN[2]	PE5RP[3]	VSS	PE7TP[0]	PE7TN[0]	VSS	PE7TP[1]	PE7TN[1]	G
F	V3REF	PE4RP[0]	PE4RN[1]	VSS	PE5TP[1]	PE5RP[2]	VSS	PE6RP[2]	PE6RN[2]	VSS	PE7RP[0]	PE7RN[0]	VSS	F
E	VSS	PE4RN[0]	VSS	PE4RN[2]	PE5TN[1]	VSS	PE5TN[3]	ERR[0]#	VSS	PE6RP[1]	PE6RN[1]	VSS	PE7RN[1]	E
D	XDPD[3]#	VSS	PE4TN[2]	PE4RP[2]	VSS	PE5TP[2]	PE5TP[3]	VSS	PE6TP[1]	PE6TN[1]	VSS	ERR[2]#	PE7RP[1]	D
C	XDPDSTBN#	PE4TP[1]	PE4TP[2]	VSS	PE5RP[1]	PE5TN[2]	VSS	PE6RP[0]	PE6RN[0]	VSS	PE6TP[2]	PE6TN[2]	VSS	C
B	XDPD[6]#	PE4TN[1]	VSS	XDPD[1]#	PE5RN[1]	VSS	TDI	TDO	VSS	TDIOCATHODE	VSS	VSS		B
A	VSS	XDPD[4]#	XDPD[0]#	XDPD[2]#	VSS	TRST#	TMS	TCK	ERR[1]#	TDIOANODE	VSS			A
	13	12	11	10	9	8	7	6	5	4	3	2	1	

Table 8-1. Intel 5000X Chipset MCH Signals (by Ball Number) (Sheet 1 of 19)

Ball No.	Signal Name	Buffer Type	Direction
A3	VSS	Power/Other	
A4	TDIOANODE	Analog	
A5	ERR[1]#	CMOS	O
A6	TCK	JTAG	I
A7	TMS	JTAG	I
A8	TRST#	JTAG	I
A9	VSS	Power/Other	
A10	XDPD[2]#	XDP	I/O
A11	XDPD[0]#	XDP	I/O
A12	XDPD[4]#	XDP	I/O
A13	VSS	Power/Other	
A14	XDPD[10]#	XDP	I/O
A15	XDPD[13]#	XDP	I/O
A16	XDPD[11]#	XDP	I/O
A17	XDPRDY#	XDP	I/O
A18	VSS	Power/Other	
A19	FBD3NBIP[4]	FBD	I
A20	VCCFBD	Power/Other	
A21	VSS	Power/Other	
A22	FBD3NBIN[6]	FBD	I
A23	VSS	Power/Other	
A24	FBD3NBIP[8]	FBD	I
A25	VSS	Power/Other	
A26	FBD2NBIN[1]	FBD	I
A27	FBD2NBIN[2]	FBD	I
A28	VSS	Power/Other	
A29	FBD2NBIN[4]	FBD	I
A30	FBD2NBIN[5]	FBD	I
A31	VSS	Power/Other	
A32	FBD2NBIN[12]	FBD	I
A33	FBD2NBIN[6]	FBD	I
A34	VSS	Power/Other	
A35	FBD2NBIN[8]	FBD	I
A36	VSS	Power/Other	
B2	VSS	Power/Other	
B3	VSS	Power/Other	
B4	TDIOCATHODE	Analog	
B5	VSS	Power/Other	
C7	VSS	Power/Other	

Ball No.	Signal Name	Buffer Type	Direction
B6	TDO	JTAG	Ouput
B7	TDI	JTAG	I
B8	VSS	Power/Other	
B9	PE5RN[1]	PEX	I
B10	XDPD[1]#	XDP	I/O
B11	VSS	Power/Other	
B12	PE4TN[1]	PEX	O
B13	XDPD[6]#	XDP	I/O
B14	XDPD[7]#	XDP	I/O
B15	XDPD[9]#	XDP	I/O
B16	VSS	Power/Other	
B17	VSS	Power/Other	
B18	FBD3NBIP[3]	FBD	I
B19	FBD3NBIN[4]	FBD	I
B20	VSS	Power/Other	
B21	FBD3NBIN[12]	FBD	I
B22	FBD3NBIP[6]	FBD	I
B23	VSS	Power/Other	
B24	FBD3NBIN[8]	FBD	I
B25	FBD2NBIN[0]	FBD	I
B26	FBD2NBIP[1]	FBD	I
B27	FBD2NBIP[2]	FBD	I
B28	FBD2NBIN[3]	FBD	I
B29	FBD2NBIP[4]	FBD	I
B30	FBD2NBIP[5]	FBD	I
B31	FBD2NBIN[13]	FBD	I
B32	FBD2NBIP[12]	FBD	I
B33	FBD2NBIP[6]	FBD	I
B34	FBD2NBIN[7]	FBD	I
B35	FBD2NBIP[8]	FBD	I
B36	FBD2NBIN[9]	FBD	I
B37	VSS	Power/Other	
C1	VSS	Power/Other	
C2	PE6TN[2]	PEX	O
C3	PE6TP[2]	PEX	O
C4	VSS	Power/Other	
C5	PE6RN[0]	PEX	I
C6	PE6RP[0]	PEX	I
D7	PE5TP[3]	PEX	O

Table 8-1. Intel 5000X Chipset MCH Signals (by Ball Number) (Sheet 2 of 19)

Ball No.	Signal Name	Buffer Type	Direction	Ball No.	Signal Name	Buffer Type	Direction
C8	PE5TN[2]	PEX	O	D8	PE5TP[2]	PEX	O
C9	PE5RP[1]	PEX	I	D9	VSS	Power/Other	
C10	VSS	Power/Other		D10	PE4RP[2]	PEX	I
C11	PE4TP[2]	PEX	O	D11	PE4TN[2]	PEX	O
C12	PE4TP[1]	PEX	O	D12	VSS	Power/Other	
C13	XDPDSTBN#	XDP	I/O	D13	XDPD[3]#	XDP	I/O
C14	XDPDSTBP#	XDP	I/O	D14	VSS	Power/Other	
C15	VSS	Power/Other		D15	XDPD[8]#	XDP	I/O
C16	XDPD[12]#	XDP	I/O	D16	XDPD[14]#	XDP	I/O
C17	FBD3NBIP[2]	FBD	I	D17	FBD3NBIN[2]	FBD	I
C18	FBD3NBIN[3]	FBD	I	D18	VSS	Power/Other	
C19	VSS	Power/Other		D19	FBD3NBIP[5]	FBD	I
C20	FBD3NBIN[13]	FBD	I	D20	FBD3NBIP[13]	FBD	I
C21	FBD3NBIP[12]	FBD	I	D21	VSS	Power/Other	
C22	VSS	Power/Other		D22	FBD3SBOP[0]	FBD	O
C23	FBD3NBIN[7]	FBD	I	D23	FBD3NBIP[7]	FBD	I
C24	VSS	Power/Other		D24	VSS	Power/Other	
C25	FBD2NBIP[0]	FBD	I	D25	FBD3NBIP[11]	FBD	I
C26	VSS	Power/Other		D26	FBD3NBIN[11]	FBD	I
C27	VSS	Power/Other		D27	VSS	Power/Other	
C28	FBD2NBIP[3]	FBD	I	D28	FBD23CLKP	Analog	
C29	VSS	Power/Other		D29	RSVD	No Connect	
C30	VSS	Power/Other		D30	VSS	Power/Other	
C31	FBD2NBIP[13]	FBD	I	D31	FBD2SBOP[3]	FBD	O
C32	VSS	Power/Other		D32	FBD2SBON[3]	FBD	O
C33	VSS	Power/Other		D33	VSS	Power/Other	
C34	FBD2NBIP[7]	FBD	I	D34	FBD2SBOP[5]	FBD	O
C35	VSS	Power/Other		D35	FBD2SBON[5]	FBD	O
C36	FBD2NBIP[9]	FBD	I	D36	VSS	Power/Other	
C37	FBD2NBIP[10]	FBD	I	D37	FBD2NBIN[10]	FBD	I
C38	VSS	Power/Other		D38	FBD2NBIP[11]	FBD	I
D1	PE7RP[1]	PEX	I	E1	PE7RN[1]	PEX	I
D2	ERR[2]#	CMOS	O	E2	VSS	Power/Other	
D3	VSS	Power/Other		E3	PE6RN[1]	PEX	I
D4	PE6TN[1]	PEX	O	E4	PE6RP[1]	PEX	I
D5	PE6TP[1]	PEX	O	E5	VSS	Power/Other	
D6	VSS	Power/Other		E6	ERR[0]#	CMOS	O
E7	PE5TN[3]	PEX	O	F7	VSS	Power/Other	
E8	VSS	Power/Other		F8	PE5RP[2]	PEX	I

Table 8-1. Intel 5000X Chipset MCH Signals (by Ball Number) (Sheet 3 of 19)

Ball No.	Signal Name	Buffer Type	Direction
E9	PE5TN[1]	PEX	O
E10	PE4RN[2]	PEX	I
E11	VSS	Power/Other	
E12	PE4RN[0]	PEX	I
E13	VSS	Power/Other	
E14	XDPD[5]#	XDP	I/O
E15	SPD2SMBDATA	SMB	I/O
E16	XDPD[15]#	XDP	I/O
E17	VSS	Power/Other	
E18	FBD3NBIN[1]	FBD	I
E19	FBD3NBIN[5]	FBD	I
E20	VCCFBD	Power/Other	
E21	VSS	Power/Other	
E22	FBD3SBON[0]	FBD	O
E23	VCCFBD	Power/Other	
E24	FBD3NBIP[10]	FBD	I
E25	FBD3NBIN[10]	FBD	I
E26	VSS	Power/Other	
E27	FBD23VCCA	Power/Other	
E28	FBD23CLKN	Analog	I
E29	VSS	Power/Other	
E30	FBD2SBOP[2]	FBD	O
E31	FBD2SBON[2]	FBD	O
E32	VSS	Power/Other	
E33	FBD2SBOP[9]	FBD	O
E34	FBD2SBON[9]	FBD	O
E35	VSS	Power/Other	
E36	FBDRESIN	Analog	
E37	FBDBGBIASEXT	Analog	
E38	FBD2NBIN[11]	FBD	I
F1	VSS	Power/Other	
F2	PE7RN[0]	PEX	I
F3	PE7RP[0]	PEX	I
F4	VSS	Power/Other	
F5	PE6RN[2]	PEX	I
F6	PE6RP[2]	PEX	I
G7	PE5RP[3]	PEX	I
G8	PE5RN[2]	PEX	I
G9	VSS	Power/Other	

Ball No.	Signal Name	Buffer Type	Direction
F9	PE5TP[1]	PEX	O
F10	VSS	Power/Other	
F11	PE4RN[1]	PEX	I
F12	PE4RP[0]	PEX	I
F13	V3REF	Analog	
F14	XDPCOMCRES	Analog	
F15	SPD2SMBCLK	SMB	I/O
F16	VSS	Power/Other	
F17	TESTHI_V3REF	Power/Other	
F18	FBD3NBIP[1]	FBD	I
F19	VSS	Power/Other	
F20	FBD3NBIN[0]	FBD	I
F21	VSS	Power/Other	
F22	VSS	Power/Other	
F23	FBD3NBIP[9]	FBD	I
F24	FBD3NBIN[9]	FBD	I
F25	VCCFBD	Power/Other	
F26	VSS	Power/Other	
F27	FBD23VSSA	Power/Other	
F28	VSS	Power/Other	
F29	FBD2SBOP[1]	FBD	O
F30	FBD2SBON[1]	FBD	O
F31	VSS	Power/Other	
F32	FBD2SBOP[4]	FBD	O
F33	FBD2SBON[4]	FBD	O
F34	VSS	Power/Other	
F35	FBDICOMPBIAS	Analog	
F36	FBD1NBIP[9]	FBD	I
F37	FBD1NBIN[9]	FBD	I
F38	VSS	Power/Other	
G1	PE7TN[1]	PEX	O
G2	PE7TP[1]	PEX	O
G3	VSS	Power/Other	
G4	PE7TN[0]	PEX	O
G5	PE7TP[0]	PEX	O
G6	VSS	Power/Other	
H7	PE5RN[3]	PEX	I
H8	VSS	Power/Other	
H9	PE5TN[0]	PEX	O

Table 8-1. Intel 5000X Chipset MCH Signals (by Ball Number) (Sheet 4 of 19)

Ball No.	Signal Name	Buffer Type	Direction	Ball No.	Signal Name	Buffer Type	Direction
G10	PE5RN[0]	PEX	I	H10	PE5RP[0]	PEX	I
G11	PE4RP[1]	PEX	I	H11	VSS	Power/Other	
G12	VCCPE	Power/Other		H12	PE4TP[0]	PEX	O
G13	SPD0SMBDATA	SMB	I/O	H13	SPD0SMBCLK	SMB	I/O
G14	XDPODTCRES	Analog		H14	VSS	Power/Other	
G15	VSS	Power/Other		H15	SPD3SMBCLK	SMB	I/O
G16	TESTHI_V3REF	Power/Other		H16	SPD3SMBDATA	SMB	I/O
G17	RESETI#	CMOS	I	H17	PWRGOOD	CMOS	I
G18	VSS	Power/Other		H18	FBD3SBOP[7]	FBD	O
G19	FBD3SBOP[6]	FBD	O	H19	FBD3SBON[6]	FBD	O
G20	FBD3NBIP[0]	FBD	I	H20	VCCFBD	Power/Other	
G21	VSS	Power/Other		H21	FBD3SBON[9]	FBD	O
G22	FBD3SBON[4]	FBD	O	H22	FBD3SBOP[9]	FBD	O
G23	FBD3SBOP[4]	FBD	O	H23	VCCFBD	Power/Other	
G24	VSS	Power/Other		H24	FBD3SBON[2]	FBD	O
G25	FBD3SBON[1]	FBD	O	H25	FBD3SBOP[2]	FBD	O
G26	FBD3SBOP[1]	FBD	O	H26	VSS	Power/Other	
G27	VSS	Power/Other		H27	VSS	Power/Other	
G28	FBD2SBOP[0]	FBD	O	H28	VSS	Power/Other	
G29	FBD2SBON[0]	FBD	O	H29	VSS	Power/Other	
G30	VSS	Power/Other		H30	VSS	Power/Other	
G31	VSS	Power/Other		H31	VSS	Power/Other	
G32	VSS	Power/Other		H32	VSS	Power/Other	
G33	VSS	Power/Other		H33	FBD2SBOP[7]	FBD	O
G34	FBD2SBOP[6]	FBD	O	H34	FBD2SBON[7]	FBD	O
G35	FBD2SBON[6]	FBD	O	H35	VSS	Power/Other	
G36	VSS	Power/Other		H36	FBD1NBIP[10]	FBD	I
G37	VSS	Power/Other		H37	FBD1NBIN[10]	FBD	I
G38	FBD1NBIP[11]	FBD	I	H38	FBD1NBIN[11]	FBD	I
H1	RSVD	No Connect		J1	VSS	Power/Other	
H2	VSS	Power/Other		J2	PECLKP	Analog	
H3	PE7RP[2]	PEX	I	J3	RSVD	No Connect	
H4	PE7RN[2]	PEX	I	J4	VSS	Power/Other	
H5	VSS	Power/Other		J5	PE7TP[3]	PEX	O
H6	PE6TP[3]	PEX	O	J6	PE6TN[3]	PEX	O
J7	VSS	Power/Other		K7	PE7TP[2]	PEX	O
J8	PE6RP[3]	PEX	I	K8	PE6RN[3]	PEX	I
J9	PE5TP[0]	PEX	O	K9	VSS	Power/Other	
J10	VCCPE	Power/Other		K10	PE4RP[3]	PEX	I

Table 8-1. Intel 5000X Chipset MCH Signals (by Ball Number) (Sheet 5 of 19)

Ball No.	Signal Name	Buffer Type	Direction	Ball No.	Signal Name	Buffer Type	Direction
J11	PE4TP[3]	PEX	O	K11	PE4TN[3]	PEX	O
J12	PE4TN[0]	PEX	O	K12	VSS	Power/Other	
J13	VSS	Power/Other		K13	GPIOSMBCLK	SMB	I/O
J14	CFGSMBDATA	SMB	I/O	K14	CFGSMBCLK	SMB	I/O
J15	XDPSLWCRES	Analog		K15	SPD1SMBDATA	SMB	I/O
J16	SPD1SMBCLK	SMB	I/O	K16	VSS	Power/Other	
J17	VSS	Power/Other		K17	VSS	Power/Other	
J18	FBD3SBON[7]	FBD	O	K18	FBD3SBON[8]	FBD	O
J19	VSS	Power/Other		K19	FBD3SBOP[8]	FBD	O
J20	FBD3SBON[5]	FBD	O	K20	VSS	Power/Other	
J21	FBD3SBOP[5]	FBD	O	K21	VCCFBD	Power/Other	
J22	VSS	Power/Other		K22	VCCFBD	Power/Other	
J23	FBD3SBON[3]	FBD	O	K23	VCCFBD	Power/Other	
J24	FBD3SBOP[3]	FBD	O	K24	VSS	Power/Other	
J25	VSS	Power/Other		K25	VSS	Power/Other	
J26	VSS	Power/Other		K26	VSS	Power/Other	
J27	VSS	Power/Other		K27	VSS	Power/Other	
J28	VSS	Power/Other		K28	VSS	Power/Other	
J29	VSS	Power/Other		K29	VSS	Power/Other	
J30	VSS	Power/Other		K30	VSS	Power/Other	
J31	VSS	Power/Other		K31	FBD1NBIP[13]	FBD	I
J32	FBD2SBOP[8]	FBD	O	K32	FBD1NBIN[13]	FBD	I
J33	FBD2SBON[8]	FBD	O	K33	VSS	Power/Other	
J34	VSS	Power/Other		K34	FBD1NBIP[7]	FBD	I
J35	FBD1NBIP[8]	FBD	I	K35	FBD1NBIN[8]	FBD	I
J36	FBD1SBOP[0]	FBD	O	K36	VSS	Power/Other	
J37	FBD1SBON[0]	FBD	O	K37	VSS	Power/Other	
J38	VSS	Power/Other		K38	FBD1SBOP[2]	FBD	O
K1	PEVCCA	Analog		L1	PEVSSA	Analog	
K2	PECLKN	Analog		L2	VCCPE	Power/Other	
K3	VSS	Power/Other		L3	RSVD	No Connect	
K4	PE7RP[3]	PEX	I	L4	PE7RN[3]	PEX	I
K5	PE7TN[3]	PEX	O	L5	VSS	Power/Other	
K6	VSS	Power/Other		L6	RSVD	No Connect	
L7	PE7TN[2]	PEX	O	M7	VSS	Power/Other	
L8	VCCPE	Power/Other		M8	PE6TN[0]	PEX	O
L9	VSSQUIET	Analog		M9	PE6TP[0]	PEX	O
L10	PE4RN[3]	PEX	I	M10	VSS	Power/Other	
L11	VSS	Power/Other		M11	RSV1	No Connect	

Table 8-1. Intel 5000X Chipset MCH Signals (by Ball Number) (Sheet 6 of 19)

Ball No.	Signal Name	Buffer Type	Direction	Ball No.	Signal Name	Buffer Type	Direction
L12	GPIOSMBDATA	SMB	I/O	M12	RSVD	No Connect	
L13	VCCPE	Power/Other		M13	VCCPE	Power/Other	
L14	VCCPE	Power/Other		M14	VCCPE	Power/Other	
L15	VCCPE	Power/Other		M15	VCCPE	Power/Other	
L16	VCC	Power/Other		M16	VCC	Power/Other	
L17	VCC	Power/Other		M17	VCC	Power/Other	
L18	VCC	Power/Other		M18	VCC	Power/Other	
L19	VCC	Power/Other		M19	VSS	Power/Other	
L20	VCCFBD	Power/Other		M20	VCCFBD	Power/Other	
L21	VCCFBD	Power/Other		M21	VCCFBD	Power/Other	
L22	VCCFBD	Power/Other		M22	VCCFBD	Power/Other	
L23	VCCFBD	Power/Other		M23	VCCFBD	Power/Other	
L24	VCCSEN	Temp		M24	VCCFBD	Power/Other	
L25	VSSSEN	Temp		M25	VCCFBD	Power/Other	
L26	VSS	Power/Other		M26	FBD1NBIP[1]	FBD	I
L27	FBD1NBIP[2]	FBD	I	M27	FBD1NBIN[1]	FBD	I
L28	FBD1NBIN[2]	FBD	I	M28	VSS	Power/Other	
L29	VSS	Power/Other		M29	FBD1NBIP[4]	FBD	I
L30	FBD1NBIP[5]	FBD	I	M30	FBD1NBIN[4]	FBD	I
L31	FBD1NBIN[5]	FBD	I	M31	VSS	Power/Other	
L32	VSS	Power/Other		M32	FBD1NBIP[12]	FBD	I
L33	FBD1NBIP[6]	FBD	I	M33	FBD1NBIN[6]	FBD	I
L34	FBD1NBIN[7]	FBD	I	M34	VSS	Power/Other	
L35	VSS	Power/Other		M35	FBD1SBOP[3]	FBD	O
L36	FBD1SBOP[1]	FBD	O	M36	FBD1SBON[3]	FBD	O
L37	FBD1SBON[1]	FBD	O	M37	VSS	Power/Other	
L38	FBD1SBON[2]	FBD	O	M38	VSS	Power/Other	
M1	VSS	Power/Other		N1	PE2TP[2]	PEX	O
M2	RSVD	No Connect		N2	RSVD	No Connect	
M3	RSVD	No Connect		N3	VSS	Power/Other	
M4	VSS	Power/Other		N4	PE2RP[1]	PEX	I
M5	RSVD	No Connect		N5	RSVD	No Connect	
M6	RSVD	No Connect		N6	VCCPE	Power/Other	
N7	RSVD	No Connect		P7	RSVD	No Connect	
N8	RSVD	No Connect		P8	VSS	Power/Other	
N9	VSS	Power/Other		P9	RSVD	No Connect	
N10	RSVD	No Connect		P10	RSVD	No Connect	
N11	PEVSSBG	Analog		P11	VSS	Power/Other	
N12	VCCPE	Power/Other		P12	PERCOMPO	Analog	

Table 8-1. Intel 5000X Chipset MCH Signals (by Ball Number) (Sheet 7 of 19)

Ball No.	Signal Name	Buffer Type	Direction	Ball No.	Signal Name	Buffer Type	Direction
N13	VCCPE	Power/Other		P13	VCCPE	Power/Other	
N14	VCCPE	Power/Other		P14	VCCPE	Power/Other	
N15	VCCPE	Power/Other		P15	VSS	Power/Other	
N16	VSS	Power/Other		P16	VCC	Power/Other	
N17	VCC	Power/Other		P17	VSS	Power/Other	
N18	VSS	Power/Other		P18	VCC	Power/Other	
N19	VCC	Power/Other		P19	VSS	Power/Other	
N20	VCCFBD	Power/Other		P20	VCC	Power/Other	
N21	VCCFBD	Power/Other		P21	VSS	Power/Other	
N22	VCCFBD	Power/Other		P22	VCC	Power/Other	
N23	VCCFBD	Power/Other		P23	VSS	Power/Other	
N24	VCCFBD	Power/Other		P24	VCC	Power/Other	
N25	VCCFBD	Power/Other		P25	VCCFBD	Power/Other	
N26	VCCFBD	Power/Other		P26	VCCFBD	Power/Other	
N27	VSS	Power/Other		P27	FBD1NBIP[0]	FBD	I
N28	FBD1NBIP[3]	FBD	I	P28	FBD1NBIN[0]	FBD	I
N29	FBD1NBIN[3]	FBD	I	P29	VSS	Power/Other	
N30	VSS	Power/Other		P30	VSS	Power/Other	
N31	VSS	Power/Other		P31	VSS	Power/Other	
N32	FBD1NBIN[12]	FBD	I	P32	VSS	Power/Other	
N33	VSS	Power/Other		P33	FBD1SBON[7]	FBD	O
N34	FBD1SBOP[4]	FBD	O	P34	FBD1SBOP[7]	FBD	O
N35	FBD1SBON[4]	FBD	O	P35	VSS	Power/Other	
N36	VSS	Power/Other		P36	FBD1SBON[5]	FBD	O
N37	FBD1SBON[9]	FBD	O	P37	FBD1SBOP[5]	FBD	O
N38	FBD1SBOP[9]	FBD	O	P38	VSS	Power/Other	
P1	PE2TN[2]	PEX	O	R1	VSS	Power/Other	
P2	VSS	Power/Other		R2	PE2TP[3]	PEX	O
P3	PE2RP[2]	PEX	I	R3	PE2RN[2]	PEX	I
P4	PE2RN[1]	PEX	I	R4	VCCPE	Power/Other	
P5	VSS	Power/Other		R5	PE2RN[0]	PEX	I
P6	RSVD	No Connect		R6	RSVD	No Connect	
R7	VSS	Power/Other		T7	PE2TN[0]	PEX	O
R8	RSVD	No Connect		T8	PE2TP[0]	PEX	O
R9	RSVD	No Connect		T9	VSS	Power/Other	
R10	VCCPE	Power/Other		T10	VSS	Power/Other	
R11	PEVCCBG	Analog		T11	VSS	Power/Other	
R12	PEICOMPI	Analog		T12	VSS	Power/Other	
R13	VCCPE	Power/Other		T13	VCCPE	Power/Other	

Table 8-1. Intel 5000X Chipset MCH Signals (by Ball Number) (Sheet 8 of 19)

Ball No.	Signal Name	Buffer Type	Direction	Ball No.	Signal Name	Buffer Type	Direction
R14	VCCPE	Power/Other		T14	VCCPE	Power/Other	
R15	VCC	Power/Other		T15	VSS	Power/Other	
R16	VSS	Power/Other		T16	VCC	Power/Other	
R17	VCC	Power/Other		T17	VSS	Power/Other	
R18	VSS	Power/Other		T18	VCC	Power/Other	
R19	VCC	Power/Other		T19	VSS	Power/Other	
R20	VSS	Power/Other		T20	VCC	Power/Other	
R21	VCC	Power/Other		T21	VSS	Power/Other	
R22	VSS	Power/Other		T22	VCC	Power/Other	
R23	VCC	Power/Other		T23	VSS	Power/Other	
R24	VSS	Power/Other		T24	VCC	Power/Other	
R25	VCCFBD	Power/Other		T25	VCCFBD	Power/Other	
R26	VSS	Power/Other		T26	VCCFBD	Power/Other	
R27	VSS	Power/Other		T27	VCCFBD	FBD	I
R28	VSS	Power/Other		T28	FBD0NBIN[4]	FBD	I
R29	VSS	Power/Other		T29	VSS	Power/Other	
R30	VSS	Power/Other		T30	VSS	Power/Other	
R31	VSS	Power/Other		T31	FBD0NBIP[6]	FBD	I
R32	FBD1SBON[8]	FBD	O	T32	FBD0NBIN[6]	FBD	I
R33	FBD1SBOP[8]	FBD	O	T33	VSS	Power/Other	
R34	VSS	Power/Other		T34	FBD01VSSA	Power/Other	
R35	FBD1SBON[6]	FBD	O	T35	FBD01VCCA	Power/Other	
R36	FBD1SBOP[6]	FBD	O	T36	VSS	Power/Other	
R37	VSS	Power/Other		T37	RSVD	No Connect	
R38	FBD01CLKP	Analog	I	T38	FBD01CLKN	Analog	I
T1	PE2RP[3]	PEX	I	U1	PE2RN[3]	PEX	I
T2	PE2TN[3]	PEX	O	U2	VCCPE	Power/Other	
T3	VSS	Power/Other		U3	PE3TP[0]	PEX	O
T4	PE2TN[1]	PEX	O	U4	PE2TP[1]	PEX	O
T5	PE2RP[0]	PEX	I	U5	VSS	Power/Other	
T6	VSS	Power/Other		U6	PE3TP[2]	PEX	O
U7	PE3TN[2]	PEX	O	V7	VSS	Power/Other	
U8	VCCPE	Power/Other		V8	PE3TP[3]	PEX	O
U9	PE3RP[3]	PEX	I	V9	PE3TN[3]	PEX	O
U10	PE3RN[3]	PEX	I	V10	VSS	Power/Other	
U11	VSS	Power/Other		V11	VSS	Power/Other	
U12	VSS	Power/Other		V12	VSS	Power/Other	
U13	VCCPE	Power/Other		V13	VCCPE	Power/Other	
U14	VCCPE	Power/Other		V14	VCCPE	Power/Other	

Table 8-1. Intel 5000X Chipset MCH Signals (by Ball Number) (Sheet 9 of 19)

Ball No.	Signal Name	Buffer Type	Direction
U15	VCC	Power/Other	
U16	VSS	Power/Other	
U17	VCC	Power/Other	
U18	VSS	Power/Other	
U19	VCC	Power/Other	
U20	VSS	Power/Other	
U21	VCC	Power/Other	
U22	VSS	Power/Other	
U23	VCC	Power/Other	
U24	VSS	Power/Other	
U25	VCCFBD	Power/Other	
U26	VCCFBD	Power/Other	
U27	FBD0NBIN[3]	FBD	I
U28	FBD0NBIP[4]	FBD	I
U29	VSS	Power/Other	
U30	FBD0NBIP[12]	FBD	I
U31	FBD0NBIN[12]	FBD	I
U32	VSS	Power/Other	
U33	FBD0NBIP[8]	FBD	I
U34	FBD0NBIN[8]	FBD	I
U35	VSS	Power/Other	
U36	FBD0NBIP[11]	FBD	I
U37	FBD0NBIN[11]	FBD	I
U38	VSS	Power/Other	
V1	VSS	Power/Other	
V2	PE3RP[0]	PEX	I
V3	PE3TN[0]	PEX	O
V4	VSS	Power/Other	
V5	PE3RP[1]	PEX	I
V6	PE3RN[1]	PEX	I
W7	PE3RP[2]	PEX	I
W8	PE3RN[2]	PEX	I
W9	VSS	Power/Other	
W10	PEWIDTH[3]	Power/Other	I
W11	PEWIDTH[2]	Power/Other	I
W12	VCCPE	Power/Other	
W13	VCCPE	Power/Other	
W14	VCCPE	Power/Other	
W15	VCC	Power/Other	

Ball No.	Signal Name	Buffer Type	Direction
V15	VSS	Power/Other	
V16	VCC	Power/Other	
V17	VSS	Power/Other	
V18	VCC	Power/Other	
V19	VSS	Power/Other	
V20	VCC	Power/Other	
V21	VSS	Power/Other	
V22	VCC	Power/Other	
V23	VSS	Power/Other	
V24	VCC	Power/Other	
V25	VCCFBD	Power/Other	
V26	VCCFBD	Power/Other	
V27	FBD0NBIP[3]	FBD	I
V28	VSS	Power/Other	
V29	FBD0NBIP[13]	FBD	I
V30	FBD0NBIN[13]	FBD	I
V31	VSS	Power/Other	
V32	FBD0NBIP[7]	FBD	I
V33	FBD0NBIN[7]	FBD	I
V34	VSS	Power/Other	
V35	FBD0NBIP[10]	FBD	I
V36	FBD0NBIN[10]	FBD	I
V37	VSS	Power/Other	
V38	FBD0SBOP[1]	FBD	O
W1	RSVD	No Connect	
W2	PE3RN[0]	PEX	I
W3	VSS	Power/Other	
W4	PE3TN[1]	PEX	O
W5	PE3TP[1]	PEX	O
W6	VCCPE	Power/Other	
Y7	PE0TP[0]	PEX	O
Y8	VSS	Power/Other	
Y9	PE0RN[0]	PEX	I
Y10	PE0RP[0]	PEX	I
Y11	VSS	Power/Other	
Y12	PEWIDTH[1]	Power/Other	I
Y13	VCCPE	Power/Other	
Y14	VCCPE	Power/Other	
Y15	VSS	Power/Other	

Table 8-1. Intel 5000X Chipset MCH Signals (by Ball Number) (Sheet 10 of 19)

Ball No.	Signal Name	Buffer Type	Direction	Ball No.	Signal Name	Buffer Type	Direction
W16	VSS	Power/Other		Y16	VCC	Power/Other	
W17	VCC	Power/Other		Y17	VSS	Power/Other	
W18	VSS	Power/Other		Y18	VCC	Power/Other	
W19	VCC	Power/Other		Y19	VSS	Power/Other	
W20	VSS	Power/Other		Y20	VCC	Power/Other	
W21	VCC	Power/Other		Y21	VSS	Power/Other	
W22	VSS	Power/Other		Y22	VCC	Power/Other	
W23	VCC	Power/Other		Y23	VSS	Power/Other	
W24	VSS	Power/Other		Y24	VCC	Power/Other	
W25	VCCFBD	Power/Other		Y25	VCCFBD	Power/Other	
W26	VCCFBD	Power/Other		Y26	VCCFBD	Power/Other	
W27	VSS	Power/Other		Y27	FBD0NBIP[0]	FBD	I
W28	FBD0NBIP[5]	FBD	I	Y28	FBD0NBIN[0]	FBD	I
W29	FBD0NBIN[5]	FBD	I	Y29	VSS	Power/Other	
W30	VSS	Power/Other		Y30	FBD0NBIP[1]	FBD	I
W31	FBD0SBON[0]	FBD	O	Y31	FBD0NBIN[1]	FBD	I
W32	FBD0SBOP[0]	FBD	O	Y32	VSS	Power/Other	
W33	VSS	Power/Other		Y33	FBD0SBON[3]	FBD	O
W34	FBD0NBIP[9]	FBD	I	Y34	FBD0SBOP[3]	FBD	O
W35	FBD0NBIN[9]	FBD	I	Y35	VSS	Power/Other	
W36	VSS	Power/Other		Y36	FBD0SBOP[4]	FBD	O
W37	VSS	Power/Other		Y37	FBD0SBON[4]	FBD	O
W38	FBD0SBON[1]	FBD	O	Y38	VSS	Power/Other	
Y1	RSVD	No Connect		AA1	VSS	Power/Other	
Y2	VSS	Power/Other		AA2	PE0TN[1]	PEX	O
Y3	PE0RN[1]	PEX	I	AA3	PE0TP[1]	PEX	O
Y4	PE0RP[1]	PEX	I	AA4	VSS	Power/Other	
Y5	VSS	Power/Other		AA5	PE0RP[3]	PEX	I
Y6	PE0TN[0]	PEX	O	AA6	PE0RN[3]	PEX	I
AA7	VSS	Power/Other		AB7	PE0RN[2]	PEX	I
AA8	PE0TP[3]	PEX	O	AB8	PE0RP[2]	PEX	I
AA9	PE0TN[3]	PEX	O	AB9	VSS	Power/Other	
AA10	VSS	Power/Other		AB10	FSB1A[14]#	Source Sync	I/O
AA11	PEWIDTH[0]	Power/Other	I	AB11	FSB1A[10]#	Source Sync	I/O
AA12	FSB1A[16]#	Source Sync	I/O	AB12	VSS	Power/Other	
AA13	VCC	Power/Other		AB13	VCC	Power/Other	
AA14	VSS	Power/Other		AB14	VCC	Power/Other	
AA15	VCC	Power/Other		AB15	VSS	Power/Other	
AA16	VSS	Power/Other		AB16	VCC	Power/Other	

Table 8-1. Intel 5000X Chipset MCH Signals (by Ball Number) (Sheet 11 of 19)

Ball No.	Signal Name	Buffer Type	Direction
AA17	VCC	Power/Other	
AA18	VSS	Power/Other	
AA19	VCC	Power/Other	
AA20	VSS	Power/Other	
AA21	VCC	Power/Other	
AA22	VSS	Power/Other	
AA23	VCC	Power/Other	
AA24	VSS	Power/Other	
AA25	VCCFBD	Power/Other	
AA26	VCCFBD	Power/Other	
AA27	VCCFBD	Power/Other	
AA28	VSS	Power/Other	
AA29	VSS	Power/Other	
AA30	VSS	Power/Other	
AA31	VSS	Power/Other	
AA32	FBD0SBOP[2]	FBD	O
AA33	FBD0SBON[2]	FBD	O
AA34	VSS	Power/Other	
AA35	FBD0SBON[9]	FBD	O
AA36	FBD0SBOP[9]	FBD	O
AA37	VSS	Power/Other	
AA38	FBD0SBOP[5]	FBD	O
AB1	PSEL[2]	CMOS	I
AB2	PSEL[1]	CMOS	I
AB3	VSS	Power/Other	
AB4	PE0TP[2]	PEX	O
AB5	PE0TN[2]	PEX	O
AB6	VSS	Power/Other	
AC7	FSB1A[15]#	Source Sync	I/O
AC8	VSS	Power/Other	
AC9	FSB1A[13]#	Source Sync	I/O
AC10	FSB1ADSTB[0]#	Source Sync	I/O
AC11	VSS	Power/Other	
AC12	FSB1A[5]#	Source Sync	I/O
AC13	VTT	Power/Other	
AC14	VTT	Power/Other	
AC15	VCC	Power/Other	
AC16	VSS	Power/Other	
AC17	VCC	Power/Other	

Ball No.	Signal Name	Buffer Type	Direction
AB17	VSS	Power/Other	
AB18	VCC	Power/Other	
AB19	VSS	Power/Other	
AB20	VCC	Power/Other	
AB21	VSS	Power/Other	
AB22	VCC	Power/Other	
AB23	VSS	Power/Other	
AB24	VCC	Power/Other	
AB25	VCCFBD	Power/Other	
AB26	VCCFBD	Power/Other	
AB27	VSS	Power/Other	
AB28	VSS	Power/Other	
AB29	VSS	Power/Other	
AB30	VSS	Power/Other	
AB31	FBD0NBIP[2]	FBD	I
AB32	FBD0NBIN[2]	FBD	I
AB33	VSS	Power/Other	
AB34	FBD0SBON[7]	FBD	O
AB35	FBD0SBOP[7]	FBD	O
AB36	VSS	Power/Other	
AB37	FBD0SBOP[6]	FBD	O
AB38	FBD0SBON[5]	FBD	O
AC1	PSEL[0]	CMOS	I
AC2	VSS	Power/Other	
AC3	FSB1A[35]#	Source Sync	I/O
AC4	FSB1A[34]#	Source Sync	I/O
AC5	VSS	Power/Other	
AC6	FSB1A[22]#	Source Sync	I/O
AD7	VSS	Power/Other	
AD8	FSB1A[12]#	Source Sync	I/O
AD9	FSB1A[4]#	Source Sync	I/O
AD10	VSS	Power/Other	
AD11	FSB1REQ[1]#	Source Sync	I/O
AD12	FSB1A[3]#	Source Sync	I/O
AD13	VTT	Power/Other	
AD14	VTT	Power/Other	
AD15	VTT	Power/Other	
AD16	VTT	Power/Other	
AD17	VTT	Power/Other	

Table 8-1. Intel 5000X Chipset MCH Signals (by Ball Number) (Sheet 12 of 19)

Ball No.	Signal Name	Buffer Type	Direction	Ball No.	Signal Name	Buffer Type	Direction
AC18	VSS	Power/Other		AD18	VTT	Power/Other	
AC19	VCC	Power/Other		AD19	VTT	Power/Other	
AC20	VSS	Power/Other		AD20	VTT	Power/Other	
AC21	VCC	Power/Other		AD21	VTT	Power/Other	
AC22	VSS	Power/Other		AD22	VTT	Power/Other	
AC23	VCC	Power/Other		AD23	VTT	Power/Other	
AC24	VSS	Power/Other		AD24	VTT	Power/Other	
AC25	VCC	Power/Other		AD25	VTT	Power/Other	
AC26	VCC	Power/Other		AD26	VCC	Power/Other	
AC27	VSS	Power/Other		AD27	VSS	Power/Other	
AC28	VSS	Power/Other		AD28	VSS	Power/Other	
AC29	VSS	Power/Other		AD29	FSB0D[46]#	Source Sync	I/O
AC30	FSB0D[42]#	Source Sync	I/O	AD30	FSB0D[41]#	Source Sync	I/O
AC31	FSB0D[44]#	Source Sync	I/O	AD31	VSS	Power/Other	
AC32	VSS	Power/Other		AD32	FSB0DSTBN[2]#	Source Sync	I/O
AC33	FBD0SBON[8]	FBD	O	AD33	FSB0DSTBP[2]#	Source Sync	I/O
AC34	FBD0SBOP[8]	FBD	O	AD34	VSS	Power/Other	
AC35	VSS	Power/Other		AD35	FSB0D[38]#	Source Sync	I/O
AC36	TESTHI	No Connect		AD36	VSS	Power/Other	
AC37	FBD0SBON[6]	FBD	O	AD37	VSS	Power/Other	
AC38	VSS	Power/Other		AD38	VSS	Power/Other	
AD1	VSS	Power/Other		AE1	FSB1A[32]#	Source Sync	I/O
AD2	FSB1A[33]#	Source Sync	I/O	AE2	FSB1A[31]#	Source Sync	I/O
AD3	FSB1A[29]#	Source Sync	I/O	AE3	VSS	Power/Other	
AD4	VSS	Power/Other		AE4	FSB1A[30]#	Source Sync	I/O
AD5	FSB1A[23]#	Source Sync	I/O	AE5	FSB1A[21]#	Source Sync	I/O
AD6	FSB1A[20]#	Source Sync	I/O	AE6	VSS	Power/Other	
AE7	FSB1A[9]#	Source Sync	I/O	AF7	FSB1A[8]#	Source Sync	I/O
AE8	RSVD	No Connect		AF8	VSS	Power/Other	
AE9	VSS	Power/Other		AF9	FSB1REQ[3]#	Source Sync	I/O
AE10	FSB1REQ[4]#	Source Sync	I/O	AF10	VSS	Power/Other	
AE11	FSB1RESET#	Common Clk	I	AF11	VSS	Power/Other	
AE12	VSS	Power/Other		AF12	FSB1DP[2]#	Common Clk	I/O
AE13	VTT	Power/Other		AF13	FSB1DP[3]#	Common Clk	I/O
AE14	VTT	Power/Other		AF14	VSS	Power/Other	
AE15	VTT	Power/Other		AF15	FSB1D[58]#	Source Sync	I/O
AE16	VTT	Power/Other		AF16	FSB1D[63]#	Source Sync	I/O
AE17	VTT	Power/Other		AF17	VSS	Power/Other	
AE18	VTT	Power/Other		AF18	VTT	Power/Other	

Table 8-1. Intel 5000X Chipset MCH Signals (by Ball Number) (Sheet 13 of 19)

Ball No.	Signal Name	Buffer Type	Direction	Ball No.	Signal Name	Buffer Type	Direction
AE19	VTT	Power/Other		AF19	VTT	Power/Other	
AE20	VTT	Power/Other		AF20	VTT	Power/Other	
AE21	VTT	Power/Other		AF21	VTT	Power/Other	
AE22	VTT	Power/Other		AF22	FSB0A[16]#	Source Sync	I/O
AE23	VTT	Power/Other		AF23	VSS	Power/Other	
AE24	VTT	Power/Other		AF24	VSS	Power/Other	
AE25	VTT	Power/Other		AF25	FSB0A[14]#	Source Sync	I/O
AE26	VTT	Power/Other		AF26	VSS	Power/Other	
AE27	VSS	Power/Other		AF27	VSS	Power/Other	
AE28	FSB0D[47]#	Source Sync	I/O	AF28	FSB0D[45]#	Source Sync	I/O
AE29	FSB0D[43]#	Source Sync	I/O	AF29	VSS	Power/Other	
AE30	VSS	Power/Other		AF30	FSB0DBI[2]#	Source Sync	I/O
AE31	FSB0D[40]#	Source Sync	I/O	AF31	FSB0D[35]#	Source Sync	I/O
AE32	FSB0D[39]#	Source Sync	I/O	AF32	VSS	Power/Other	
AE33	VSS	Power/Other		AF33	FSB0D[37]#	Source Sync	I/O
AE34	FSB0D[34]#	Source Sync	I/O	AF34	FSB0VREF	Power/Other	
AE35	VCC	Power/Other		AF35	VSS	Power/Other	
AE36	FSB0D[62]#	Source Sync	I/O	AF36	VSS	Power/Other	
AE37	FSB0D[63]#	Source Sync	I/O	AF37	FSB0D[48]#	Source Sync	I/O
AE38	FSB0D[58]#	Source Sync	I/O	AF38	FSB0D[59]#	Source Sync	I/O
AF1	FSB1A[27]#	Source Sync	I/O	AG1	VSS	Power/Other	
AF2	VSS	Power/Other		AG2	RSVD	No Connect	
AF3	FSB1A[28]#	Source Sync	I/O	AG3	FSB1ADSTB[1]#	Source Sync	I/O
AF4	FSB1A[17]#	Source Sync	I/O	AG4	VSS	Power/Other	
AF5	VSS	Power/Other		AG5	FSB1A[18]#	Source Sync	I/O
AF6	FSB1A[11]#	Source Sync	I/O	AG6	RSVD	No Connect	
AG7	VSS	Power/Other		AH7	RSVD	No Connect	
AG8	FSB1A[7]#	Source Sync	I/O	AH8	FSB1A[6]#	Source Sync	I/O
AG9	FSB1REQ[0]#	Source Sync	I/O	AH9	VSS	Power/Other	
AG10	FSB1AP[1]#	Common Clk	I/O	AH10	VCC	Power/Other	
AG11	FSB1DP[0]#	Common Clk	I/O	AH11	FSB1MCERR#	Common Clk	I/O
AG12	FSB1AP[0]#	Common Clk	I/O	AH12	VSS	Power/Other	
AG13	VSS	Power/Other		AH13	FSB1DBI[3]#	Source Sync	I/O
AG14	FSB1D[62]#	Source Sync	I/O	AH14	FSB1D[48]#	Source Sync	I/O
AG15	FSB1D[59]#	Source Sync	I/O	AH15	VSS	Power/Other	
AG16	VSS	Power/Other		AH16	FSB1D[54]#	Source Sync	I/O
AG17	VSS	Power/Other		AH17	VSS	No Connect	
AG18	VTT	Power/Other		AH18	VTT	Power/Other	
AG19	VTT	Power/Other		AH19	VTT	Power/Other	

Table 8-1. Intel 5000X Chipset MCH Signals (by Ball Number) (Sheet 14 of 19)

Ball No.	Signal Name	Buffer Type	Direction
AG20	VTT	Power/Other	
AG21	VTT	Power/Other	
AG22	VSS	Power/Other	
AG23	FSB0A[15]#	Source Sync	I/O
AG24	FSB0A[10]#	Source Sync	I/O
AG25	VSS	Power/Other	
AG26	FSB0BREQ[1]#	Common Clk	I/O
AG27	FSB0D[14]#	Source Sync	I/O
AG28	VSS	Power/Other	
AG29	FSB0D[13]#	Source Sync	I/O
AG30	FSB0D[33]#	Source Sync	I/O
AG31	VSS	Power/Other	
AG32	FSB0D[36]#	Source Sync	I/O
AG33	FSB0D[32]#	Source Sync	I/O
AG34	VSS	Power/Other	
AG35	FSB0D[54]#	Source Sync	I/O
AG36	FSB0D[60]#	Source Sync	I/O
AG37	VSS	Power/Other	
AG38	VSS	Power/Other	
AH1	FSB1A[25]#	Source Sync	I/O
AH2	FSB1A[24]#	Source Sync	I/O
AH3	VSS	Power/Other	
AH4	FSB1VREF	Power/Other	
AH5	FSB1A[19]#	Source Sync	I/O
AH6	VSS	Power/Other	
AJ7	FSB1HITM#	Common Clk	I/O
AJ8	VSS	Power/Other	
AJ9	FSB1DEFER#	Common Clk	O
AJ10	FSB1BPRI#	Common Clk	O
AJ11	VSS	Power/Other	
AJ12	FSB1DP[1]#	Common Clk	I/O
AJ13	FSB1D[57]#	Source Sync	I/O
AJ14	VSS	Power/Other	
AJ15	FSB1D[60]#	Source Sync	I/O
AJ16	FSB1D[61]#	Source Sync	I/O
AJ17	VSS	Power/Other	
AJ18	VTT	Power/Other	
AJ19	VTT	Power/Other	
AJ20	VTT	Power/Other	

Ball No.	Signal Name	Buffer Type	Direction
AH20	VTT	Power/Other	
AH21	VTT	Power/Other	
AH22	FSB0A[13]#	Source Sync	I/O
AH23	FSB0A[8]#	Source Sync	I/O
AH24	VSS	Power/Other	
AH25	FSB0REQ[1]#	Source Sync	I/O
AH26	FSB0AP[0]#	Common Clk	I/O
AH27	VSS	Power/Other	
AH28	FSB0D[2]#	Source Sync	I/O
AH29	FSB0D[4]#	Source Sync	I/O
AH30	VSS	Power/Other	
AH31	FSB0D[11]#	Source Sync	I/O
AH32	FSB0D[9]#	Source Sync	I/O
AH33	VSS	Power/Other	
AH34	FSB0DSTBN[3]#	Source Sync	I/O
AH35	FSB0DSTBP[3]#	Source Sync	I/O
AH36	FSB0D[61]#	Source Sync	I/O
AH37	FSB0DBI[3]#	Source Sync	I/O
AH38	FSB0D[57]#	Source Sync	I/O
AJ1	RSVD	No Connect	
AJ2	VSS	Power/Other	
AJ3	FSB1A[26]#	Source Sync	I/O
AJ4	FSB1BINIT#	Common Clk	I/O
AJ5	VSS	Power/Other	
AJ6	FSB1REQ[2]#	Source Sync	I/O
AK7	VSS	Power/Other	
AK8	FSB1HIT#	Common Clk	I/O
AK9	FSB1D[19]#	Source Sync	I/O
AK10	VSS	Power/Other	
AK11	FSB1DBI[1]#	Source Sync	I/O
AK12	FSB1D[31]#	Source Sync	I/O
AK13	VSS	Power/Other	
AK14	FSB1DSTBN[3]#	Source Sync	I/O
AK15	FSB1DSTBP[3]#	Source Sync	I/O
AK16	VSS	Power/Other	
AK17	RSVD	No Connect	
AK18	VTT	Power/Other	
AK19	VTT	Power/Other	
AK20	VTT	Power/Other	

Table 8-1. Intel 5000X Chipset MCH Signals (by Ball Number) (Sheet 15 of 19)

Ball No.	Signal Name	Buffer Type	Direction	Ball No.	Signal Name	Buffer Type	Direction
AJ21	VTT	Power/Other		AK21	VTT	Power/Other	
AJ22	FSB0A[11]#	Source Sync	I/O	AK22	VSS	Power/Other	
AJ23	VSS	Power/Other		AK23	RSVD	No Connect	
AJ24	FSB0REQ[3]#	Source Sync	I/O	AK24	FSB0REQ[2]#	Source Sync	I/O
AJ25	FSB0REQ[4]#	Source Sync	I/O	AK25	VSS	Power/Other	
AJ26	VSS	Power/Other		AK26	FSB0AP[1]#	Source Sync	I/O
AJ27	FSB0MCERR#	Common Clk	I/O	AK27	FSB0BINIT#	Common Clk	I/O
AJ28	FSB0D[0]#	Source Sync	I/O	AK28	VSS	Power/Other	
AJ29	VSS	Power/Other		AK29	FSB0D[3]#	Source Sync	I/O
AJ30	FSB0D[10]#	Source Sync	I/O	AK30	FSB0D[6]#	Source Sync	I/O
AJ31	FSB0D[8]#	Source Sync	I/O	AK31	VSS	Power/Other	
AJ32	VSS	Power/Other		AK32	FSB0DSTBN[0]#	Source Sync	I/O
AJ33	FSB0D[15]#	Source Sync	I/O	AK33	FSB0DSTBP[0]#	Source Sync	I/O
AJ34	FSB0D[49]#	Source Sync	I/O	AK34	VSS	Power/Other	
AJ35	VSS	Power/Other		AK35	FSB0D[30]#	Source Sync	I/O
AJ36	VSS	Power/Other		AK36	FSB0D[53]#	Source Sync	I/O
AJ37	FSB0D[55]#	Source Sync	I/O	AK37	VSS	Power/Other	
AJ38	FSB0D[56]#	Source Sync	I/O	AK38	VSS	Power/Other	
AK1	VSS	Power/Other		AL1	FSB1RS[1]#	Common Clk	I
AK2	FSB1RSP#	Common Clk	I	AL2	FSB1BREQ[0]#	Common Clk	I/O
AK3	FSB1BNR#	Common Clk	I/O	AL3	VSS	Power/Other	
AK4	VSS	Power/Other		AL4	FSB1LOCK#	Common Clk	I/O
AK5	FSB1RS[0]#	Common Clk	I	AL5	FSB1RS[2]#	Common Clk	I
AK6	FSB1TRDY#	Common Clk	O	AL6	VSS	Power/Other	
AL7	FSB1D[16]#	Source Sync	I/O	AM7	RSVD	No Connect	
AL8	FSB1D[17]#	Source Sync	I/O	AM8	VSS	Power/Other	
AL9	VSS	Power/Other		AM9	FSB1D[21]#	Source Sync	I/O
AL10	FSB1DSTBN[1]#	Source Sync	I/O	AM10	FSB1DSTBP[1]#	Source Sync	I/O
AL11	FSB1D[29]#	Source Sync	I/O	AM11	VSS	Power/Other	
AL12	VSS	Power/Other		AM12	FSB1D[26]#	Source Sync	I/O
AL13	FSB1D[30]#	Source Sync	I/O	AM13	FSB1D[28]#	Source Sync	I/O
AL14	FSB1D[52]#	Source Sync	I/O	AM14	VSS	Power/Other	
AL15	VSS	Power/Other		AM15	FSB1D[51]#	Source Sync	I/O
AL16	FSB1D[56]#	Source Sync	I/O	AM16	FSB1D[49]#	Source Sync	I/O
AL17	VCC	Power/Other		AM17	VSS	Power/Other	
AL18	VTT	Power/Other		AM18	VTT	Power/Other	
AL19	VTT	Power/Other		AM19	VTT	Power/Other	
AL20	VTT	Power/Other		AM20	VTT	Power/Other	
AL21	VTT	Power/Other		AM21	VTT	Power/Other	

Table 8-1. Intel 5000X Chipset MCH Signals (by Ball Number) (Sheet 16 of 19)

Ball No.	Signal Name	Buffer Type	Direction	Ball No.	Signal Name	Buffer Type	Direction
AL22	FSB0A[12]#	Source Sync	I/O	AM22	FSB0A[9]#	Source Sync	I/O
AL23	FSB0ADSTB[0]#	Source Sync	I/O	AM23	VSS	Power/Other	
AL24	VSS	Power/Other		AM24	RSVD	No Connect	
AL25	FSB0A[3]#	Source Sync	I/O	AM25	FSB0A[5]#	Source Sync	I/O
AL26	FSB0REQ[0]#	Source Sync	I/O	AM26	VSS	Power/Other	
AL27	VSS	Power/Other		AM27	FSB0VREF	Power/Other	
AL28	FSB0RS[1]#	Common Clk	I	AM28	FSB0BREQ[0]#	Common Clk	I/O
AL29	FSB0D[1]#	Source Sync	I/O	AM29	VSS	Power/Other	
AL30	VSS	Power/Other		AM30	FSB0VREF	Power/Other	
AL31	FSB0D[7]#	Source Sync	I/O	AM31	RSVD	No Connect	
AL32	FSB0DBI[0]#	Source Sync	I/O	AM32	VSS	Power/Other	
AL33	VSS	Power/Other		AM33	FSB0D[12]#	Source Sync	I/O
AL34	FSB0D[17]#	Source Sync	I/O	AM34	FSB0D[29]#	Source Sync	I/O
AL35	FSB0D[25]#	Source Sync	I/O	AM35	VSS	Power/Other	
AL36	FSB0D[51]#	Source Sync	I/O	AM36	VSS	Power/Other	
AL37	FSB0D[52]#	Source Sync	I/O	AM37	FSB0D[31]#	Source Sync	I/O
AL38	FSB0D[50]#	Source Sync	I/O	AM38	FSB0D[28]#	Source Sync	I/O
AM1	FSB1BREQ[1]#	Common Clk	I/O	AN1	VSS	Power/Other	
AM2	VSS	Power/Other		AN2	FSB1BPM[4]#	Common Clk	I/O
AM3	FSB1DRDY#	Common Clk	I/O	AN3	FSB1BPM[5]#	Common Clk	I/O
AM4	FSB1DBSY#	Common Clk	I/O	AN4	VSS	Power/Other	
AM5	VSS	Power/Other		AN5	FSB1VREF	Power/Other	
AM6	FSB1D[20]#	Source Sync	I/O	AN6	FSB1D[18]#	Source Sync	I/O
AN7	VSS	Power/Other		AP7	FSB1DBI[0]#	Source Sync	I/O
AN8	FSB1D[23]#	Source Sync	I/O	AP8	FSB1D[22]#	Source Sync	I/O
AN9	FSB1D[24]#	Source Sync	I/O	AP9	VSS	Power/Other	
AN10	VSS	Power/Other		AP10	FSB1DSTBN[2]#	Source Sync	I/O
AN11	FSB1D[27]#	Source Sync	I/O	AP11	FSB1D[44]#	Source Sync	I/O
AN12	FSB1D[25]#	Source Sync	I/O	AP12	VSS	Power/Other	
AN13	VSS	Power/Other		AP13	FSB1D[43]#	Source Sync	I/O
AN14	FSB1D[50]#	Source Sync	I/O	AP14	FSB1D[47]#	Source Sync	I/O
AN15	FSB1D[53]#	Source Sync	I/O	AP15	VSS	Power/Other	
AN16	VSS	Power/Other		AP16	FSB1D[55]#	Source Sync	I/O
AN17	CORECLKP	Analog	I	AP17	CORECLKN	Analog	I
AN18	VTT	Power/Other		AP18	VTT	Power/Other	
AN19	VTT	Power/Other		AP19	VTT	Power/Other	
AN20	VTT	Power/Other		AP20	VTT	Power/Other	
AN21	VTT	Power/Other		AP21	VTT	Power/Other	
AN22	VSS	Power/Other		AP22	FSB0A[32]#	Source Sync	I/O

Table 8-1. Intel 5000X Chipset MCH Signals (by Ball Number) (Sheet 17 of 19)

Ball No.	Signal Name	Buffer Type	Direction
AN23	RSVD	No Connect	
AN24	FSB0A[4]#	Source Sync	I/O
AN25	VSS	Power/Other	
AN26	FSB0A[6]#	Source Sync	I/O
AN27	FSB0RSP#	Common Clk	I
AN28	VSS	Power/Other	
AN29	FSB0DP[3]#	Common Clk	I/O
AN30	FSB0RESET#	Common Clk	I
AN31	VSS	Power/Other	
AN32	FSB0D[5]#	Source Sync	I/O
AN33	FSB0D[16]#	Source Sync	I/O
AN34	VSS	Power/Other	
AN35	FSB0D[24]#	Source Sync	I/O
AN36	FSB0D[26]#	Source Sync	I/O
AN37	VSS	Power/Other	
AN38	VSS	Power/Other	
AP1	FSB1D[0]#	Source Sync	I/O
AP2	FSB1ADS#	Common Clk	I/O
AP3	VSS	Power/Other	
AP4	FSB1D[2]#	Source Sync	I/O
AP5	FSB1D[1]#	Source Sync	I/O
AP6	VSS	Power/Other	
AR7	FSB1D[14]#	Source Sync	I/O
AR8	VSS	Power/Other	
AR9	FSB1D[35]#	Source Sync	I/O
AR10	FSB1DSTBP[2]#	Source Sync	I/O
AR11	VSS	Power/Other	
AR12	FSB1D[46]#	Source Sync	I/O
AR13	FSB1D[45]#	Source Sync	I/O
AR14	VSS	Power/Other	
AR15	RSVD	No Connect	
AR16	RSVD	No Connect	
AR17	VSS	Power/Other	
AR18	VTT	Power/Other	
AR19	VTT	Power/Other	
AR20	VTT	Power/Other	
AR21	VTT	Power/Other	
AR22	FSB0A[33]#	Source Sync	I/O
AR23	VSS	Power/Other	
AP23	FSB0ADSTB[1]#	Source Sync	I/O
AP24	VSS	Power/Other	
AP25	FSB0A[18]#	Source Sync	I/O
AP26	FSB0A[7]#	Source Sync	I/O
AP27	VSS	Power/Other	
AP28	RSVD	No Connect	
AP29	FSB0BPM[5]#	Common Clk	I/O
AP30	VSS	Power/Other	
AP31	FSB0DP[2]#	Common Clk	I/O
AP32	RSVD	No Connect	
AP33	VSS	Power/Other	
AP34	FSB0D[20]#	Source Sync	I/O
AP35	FSB0D[18]#	Source Sync	I/O
AP36	FSB0D[23]#	Source Sync	I/O
AP37	FSB0DBI[1]#	Source Sync	I/O
AP38	FSB0D[27]#	Source Sync	I/O
AR1	FSB1D[3]#	Source Sync	I/O
AR2	VSS	Power/Other	
AR3	FSB1D[4]#	Source Sync	I/O
AR4	FSB1D[5]#	Source Sync	I/O
AR5	VSS	Power/Other	
AR6	FSB1D[12]#	Source Sync	I/O
AT7	VSS	Power/Other	
AT8	FSB1D[36]#	Source Sync	I/O
AT9	FSB1D[34]#	Source Sync	I/O
AT10	VSS	Power/Other	
AT11	FSB1D[41]#	Source Sync	I/O
AT12	FSB1D[42]#	Source Sync	I/O
AT13	VSS	Power/Other	
AT14	FSB1VREF	Power/Other	
AT15	VSS	Power/Other	
AT16	VSS	Power/Other	
AT17	COREVCCA	Power/Other	
AT18	VTT	Power/Other	
AT19	VTT	Power/Other	
AT20	VTT	Power/Other	
AT21	VTT	Power/Other	
AT22	VSS	Power/Other	
AT23	FSB0A[30]#	Source Sync	I/O

Table 8-1. Intel 5000X Chipset MCH Signals (by Ball Number) (Sheet 18 of 19)

Ball No.	Signal Name	Buffer Type	Direction
AR24	FSB0A[20]#	Source Sync	I/O
AR25	FSB0A[26]#	Source Sync	I/O
AR26	VSS	Power/Other	
AR27	FSB0A[19]#	Source Sync	I/O
AR28	FSB0BPM[4]#	Common Clk	I/O
AR29	VSS	Power/Other	
AR30	FSB0DBSY#	Common Clk	I/O
AR31	FSB0DP[0]#	Common Clk	I/O
AR32	VSS	Power/Other	
AR33	RSVD	No Connect	
AR34	FSBODTCRES	Analog	
AR35	VSS	Power/Other	
AR36	VSS	Power/Other	
AR37	FSB0DSTBN[1]#	Source Sync	I/O
AR38	FSB0DSTBP[1]#	Source Sync	I/O
AT1	VSS	Power/Other	
AT2	FSB1D[6]#	Source Sync	I/O
AT3	FSB1D[7]#	Source Sync	I/O
AT4	VSS	Power/Other	
AT5	FSB1D[8]#	Source Sync	I/O
AT6	FSB1D[11]#	Source Sync	I/O
AU8	FSB1D[33]#	Source Sync	I/O
AU9	VSS	Power/Other	
AU10	FSB1D[38]#	Source Sync	I/O
AU11	FSB1DBI[2]#	Source Sync	I/O
AU12	VSS	Power/Other	
AU13	VSS	Power/Other	
AU14	VSS	Power/Other	
AU15	VSS	Power/Other	
AU16	COREVSSA	Power/Other	
AU17	FSBVCCA	Power/Other	
AU18	VTT	Power/Other	
AU19	VTT	Power/Other	
AU20	VTT	Power/Other	
AU21	VTT	Power/Other	
AU22	FSB0A[34]#	Source Sync	I/O
AU23	FSB0A[29]#	Source Sync	I/O
AU24	VSS	Power/Other	
AU25	FSB0A[22]#	Source Sync	I/O

Ball No.	Signal Name	Buffer Type	Direction
AT24	FSB0A[27]#	Source Sync	I/O
AT25	VSS	Power/Other	
AT26	FSB0A[24]#	Source Sync	I/O
AT27	FSB0A[23]#	Source Sync	I/O
AT28	VSS	Power/Other	
AT29	FSB0DRDY#	Common Clk	I/O
AT30	FSB0LOCK#	Common Clk	I/O
AT31	VSS	Power/Other	
AT32	FSB0TRDY#	Common Clk	O
AT33	FSB0DP[1]#	Common Clk	I/O
AT34	VSS	Power/Other	
AT35	FSBCRES	Analog	
AT36	FSB0D[19]#	Source Sync	I/O
AT37	FSB0D[22]#	Source Sync	I/O
AT38	VSS	Power/Other	
AU2	VSS	Power/Other	
AU3	FSB1DSTBN[0]#	Source Sync	I/O
AU4	FSB1DSTBP[0]#	Source Sync	I/O
AU5	FSB1D[15]#	Source Sync	I/O
AU6	VSS	Power/Other	
AU7	FSB1D[13]#	Source Sync	I/O
AV11	VSS	Power/Other	
AV12	FSB1D[40]#	Source Sync	I/O
AV13	FSBSLWCTRL	Power/Other	
AV14	VSS	Power/Other	
AV15	VSS	Power/Other	
AV16	VSS	Power/Other	
AV17	VSS	Power/Other	
AV18	VTT	Power/Other	
AV19	VTT	Power/Other	
AV20	VTT	Power/Other	
AV21	VTT	Power/Other	
AV22	FSB0A[35]#	Source Sync	I/O
AV23	VSS	Power/Other	
AV24	FSB0A[31]#	Source Sync	I/O
AV25	FSB0A[28]#	Source Sync	I/O
AV26	VSS	Power/Other	
AV27	RSVD	No Connect	
AV28	FSB0A[17]#	Source Sync	I/O

Table 8-1. Intel 5000X Chipset MCH Signals (by Ball Number) (Sheet 19 of 19)

Ball No.	Signal Name	Buffer Type	Direction
AU26	FSB0A[25]#	Source Sync	I/O
AU27	VSS	Power/Other	
AU28	FSB0A[21]#	Source Sync	I/O
AU29	FSB0ADS#	Common Clk	I/O
AU30	VSS	Power/Other	
AU31	FSB0RS[2]#	Common Clk	I
AU32	FSB0HIT#	Common Clk	I/M105O
AU33	VSS	Power/Other	
AU34	FSB0BPRI#	Common Clk	O
AU35	FSBSLWCRES	Analog	
AU36	FSB0D[21]#	Source Sync	I/O
AU37	VSS	Power/Other	
AV3	VSS	Power/Other	
AV4	FSB1D[10]#	Source Sync	I/O
AV5	VSS	Power/Other	
AV6	FSB1D[9]#	Source Sync	I/O
AV7	FSB1D[32]#	Source Sync	I/O
AV8	VSS	Power/Other	
AV9	FSB1D[37]#	Source Sync	I/O
AV10	FSB1D[39]#	Source Sync	I/O
AV29	VSS	Power/Other	
AV30	FSB0BNR#	Common Clk	I/O
AV31	FSB0RS[0]#	Common Clk	I
AV32	VSS	Power/Other	
AV33	FSB0HITM#	Common Clk	I/O
AV34	FSB0DEFER#	Common Clk	O
AV35	VSS	Power/Other	
AV36	VSS	Power/Other	

Table 8-2. Intel 5000X Chipset MCH Signals (by Signal Name) (Sheet 1 of 19)

Ball No.	Signal Name	Buffer Type	Direction
K14	CFGSMBCLK	SMB	I/O
J14	CFGSMBDATA	SMB	I/O
AP17	CORECLKN	Analog	I
AN17	CORECLKP	Analog	I
AT17	COREVCCA	Power/Other	
AU16	COREVSSA	Power/Other	
E6	ERR[0]#	CMOS	O
A5	ERR[1]#	CMOS	O
D2	ERR[2]#	CMOS	O
T38	FBD01CLKN	Analog	I
R38	FBD01CLKP	Analog	I
T35	FBD01VCCA	Power/Other	
T34	FBD01VSSA	Power/Other	
Y28	FBD0NBIN[0]	FBD	I
Y31	FBD0NBIN[1]	FBD	I
V36	FBD0NBIN[10]	FBD	I
U37	FBD0NBIN[11]	FBD	I
U31	FBD0NBIN[12]	FBD	I
V30	FBD0NBIN[13]	FBD	I
AB32	FBD0NBIN[2]	FBD	I
U27	FBD0NBIN[3]	FBD	I
T28	FBD0NBIN[4]	FBD	I
W29	FBD0NBIN[5]	FBD	I
T32	FBD0NBIN[6]	FBD	I
V33	FBD0NBIN[7]	FBD	I
U34	FBD0NBIN[8]	FBD	I
W35	FBD0NBIN[9]	FBD	I
Y27	FBD0NBIP[0]	FBD	I
Y30	FBD0NBIP[1]	FBD	I
V35	FBD0NBIP[10]	FBD	I
U36	FBD0NBIP[11]	FBD	I
U30	FBD0NBIP[12]	FBD	I
V29	FBD0NBIP[13]	FBD	I
AB31	FBD0NBIP[2]	FBD	I
V27	FBD0NBIP[3]	FBD	I
U28	FBD0NBIP[4]	FBD	I
W28	FBD0NBIP[5]	FBD	I
T31	FBD0NBIP[6]	FBD	I
M26	FBD1NBIP[1]	FBD	I
V32	FBD0NBIP[7]	FBD	I
U33	FBD0NBIP[8]	FBD	I
W34	FBD0NBIP[9]	FBD	I
W31	FBD0SBON[0]	FBD	O
W38	FBD0SBON[1]	FBD	O
AA33	FBD0SBON[2]	FBD	O
Y33	FBD0SBON[3]	FBD	O
Y37	FBD0SBON[4]	FBD	O
AB38	FBD0SBON[5]	FBD	O
AC37	FBD0SBON[6]	FBD	O
AB34	FBD0SBON[7]	FBD	O
AC33	FBD0SBON[8]	FBD	O
AA35	FBD0SBON[9]	FBD	O
W32	FBD0SBOP[0]	FBD	O
V38	FBD0SBOP[1]	FBD	O
AA32	FBD0SBOP[2]	FBD	O
Y34	FBD0SBOP[3]	FBD	O
Y36	FBD0SBOP[4]	FBD	O
AA38	FBD0SBOP[5]	FBD	O
AB37	FBD0SBOP[6]	FBD	O
AB35	FBD0SBOP[7]	FBD	O
AC34	FBD0SBOP[8]	FBD	O
AA36	FBD0SBOP[9]	FBD	O
P28	FBD1NBIN[0]	FBD	I
M27	FBD1NBIN[1]	FBD	I
H37	FBD1NBIN[10]	FBD	I
H38	FBD1NBIN[11]	FBD	I
N32	FBD1NBIN[12]	FBD	I
K32	FBD1NBIN[13]	FBD	I
L28	FBD1NBIN[2]	FBD	I
N29	FBD1NBIN[3]	FBD	I
M30	FBD1NBIN[4]	FBD	I
L31	FBD1NBIN[5]	FBD	I
M33	FBD1NBIN[6]	FBD	I
L34	FBD1NBIN[7]	FBD	I
K35	FBD1NBIN[8]	FBD	I
F37	FBD1NBIN[9]	FBD	I
P27	FBD1NBIP[0]	FBD	I
A26	FBD2NBIN[1]	FBD	I

Table 8-2. Intel 5000X Chipset MCH Signals (by Signal Name) (Sheet 2 of 19)

Ball No.	Signal Name	Buffer Type	Direction
H36	FBD1NBIP[10]	FBD	I
G38	FBD1NBIP[11]	FBD	I
M32	FBD1NBIP[12]	FBD	I
K31	FBD1NBIP[13]	FBD	I
L27	FBD1NBIP[2]	FBD	I
N28	FBD1NBIP[3]	FBD	I
M29	FBD1NBIP[4]	FBD	I
L30	FBD1NBIP[5]	FBD	I
L33	FBD1NBIP[6]	FBD	I
K34	FBD1NBIP[7]	FBD	I
J35	FBD1NBIP[8]	FBD	I
F36	FBD1NBIP[9]	FBD	I
J37	FBD1SBON[0]	FBD	O
L37	FBD1SBON[1]	FBD	O
L38	FBD1SBON[2]	FBD	O
M36	FBD1SBON[3]	FBD	O
N35	FBD1SBON[4]	FBD	O
P36	FBD1SBON[5]	FBD	O
R35	FBD1SBON[6]	FBD	O
P33	FBD1SBON[7]	FBD	O
R32	FBD1SBON[8]	FBD	O
N37	FBD1SBON[9]	FBD	O
J36	FBD1SBOP[0]	FBD	O
L36	FBD1SBOP[1]	FBD	O
K38	FBD1SBOP[2]	FBD	O
M35	FBD1SBOP[3]	FBD	O
N34	FBD1SBOP[4]	FBD	O
P37	FBD1SBOP[5]	FBD	O
R36	FBD1SBOP[6]	FBD	O
P34	FBD1SBOP[7]	FBD	O
R33	FBD1SBOP[8]	FBD	O
N38	FBD1SBOP[9]	FBD	O
E28	FBD23CLKN	Analog	I
D28	FBD23CLKP	Analog	
E27	FBD23VCCA	Power/Other	
F27	FBD23VSSA	Power/Other	
B25	FBD2NBIN[0]	FBD	I
F29	FBD2SBOP[1]	FBD	O
E30	FBD2SBOP[2]	FBD	O
D37	FBD2NBIN[10]	FBD	I
E38	FBD2NBIN[11]	FBD	I
A32	FBD2NBIN[12]	FBD	I
B31	FBD2NBIN[13]	FBD	I
A27	FBD2NBIN[2]	FBD	I
B28	FBD2NBIN[3]	FBD	I
A29	FBD2NBIN[4]	FBD	I
A30	FBD2NBIN[5]	FBD	I
A33	FBD2NBIN[6]	FBD	I
B34	FBD2NBIN[7]	FBD	I
A35	FBD2NBIN[8]	FBD	I
B36	FBD2NBIN[9]	FBD	I
C25	FBD2NBIP[0]	FBD	I
B26	FBD2NBIP[1]	FBD	I
C37	FBD2NBIP[10]	FBD	I
D38	FBD2NBIP[11]	FBD	I
B32	FBD2NBIP[12]	FBD	I
C31	FBD2NBIP[13]	FBD	I
B27	FBD2NBIP[2]	FBD	I
C28	FBD2NBIP[3]	FBD	I
B29	FBD2NBIP[4]	FBD	I
B30	FBD2NBIP[5]	FBD	I
B33	FBD2NBIP[6]	FBD	I
C34	FBD2NBIP[7]	FBD	I
B35	FBD2NBIP[8]	FBD	I
C36	FBD2NBIP[9]	FBD	I
G29	FBD2SBON[0]	FBD	O
F30	FBD2SBON[1]	FBD	O
E31	FBD2SBON[2]	FBD	O
D32	FBD2SBON[3]	FBD	O
F33	FBD2SBON[4]	FBD	O
D35	FBD2SBON[5]	FBD	O
G35	FBD2SBON[6]	FBD	O
H34	FBD2SBON[7]	FBD	O
J33	FBD2SBON[8]	FBD	O
E34	FBD2SBON[9]	FBD	O
G28	FBD2SBOP[0]	FBD	O
G25	FBD3SBON[1]	FBD	O
H24	FBD3SBON[2]	FBD	O

Table 8-2. Intel 5000X Chipset MCH Signals (by Signal Name) (Sheet 3 of 19)

Ball No.	Signal Name	Buffer Type	Direction
D31	FBD2SBOP[3]	FBD	O
F32	FBD2SBOP[4]	FBD	O
D34	FBD2SBOP[5]	FBD	O
G34	FBD2SBOP[6]	FBD	O
H33	FBD2SBOP[7]	FBD	O
J32	FBD2SBOP[8]	FBD	O
E33	FBD2SBOP[9]	FBD	O
F20	FBD3NBIN[0]	FBD	I
E18	FBD3NBIN[1]	FBD	I
E25	FBD3NBIN[10]	FBD	I
D26	FBD3NBIN[11]	FBD	I
B21	FBD3NBIN[12]	FBD	I
C20	FBD3NBIN[13]	FBD	I
D17	FBD3NBIN[2]	FBD	I
C18	FBD3NBIN[3]	FBD	I
B19	FBD3NBIN[4]	FBD	I
E19	FBD3NBIN[5]	FBD	I
A22	FBD3NBIN[6]	FBD	I
C23	FBD3NBIN[7]	FBD	I
B24	FBD3NBIN[8]	FBD	I
F24	FBD3NBIN[9]	FBD	I
G20	FBD3NBIP[0]	FBD	I
F18	FBD3NBIP[1]	FBD	I
E24	FBD3NBIP[10]	FBD	I
D25	FBD3NBIP[11]	FBD	I
C21	FBD3NBIP[12]	FBD	I
D20	FBD3NBIP[13]	FBD	I
C17	FBD3NBIP[2]	FBD	I
B18	FBD3NBIP[3]	FBD	I
A19	FBD3NBIP[4]	FBD	I
D19	FBD3NBIP[5]	FBD	I
B22	FBD3NBIP[6]	FBD	I
D23	FBD3NBIP[7]	FBD	I
A24	FBD3NBIP[8]	FBD	I
F23	FBD3NBIP[9]	FBD	I
E22	FBD3SBON[0]	FBD	O
AR25	FSB0A[26]#	Source Sync	I/O
AT24	FSB0A[27]#	Source Sync	I/O
AV25	FSB0A[28]#	Source Sync	I/O

Ball No.	Signal Name	Buffer Type	Direction
J23	FBD3SBON[3]	FBD	O
G22	FBD3SBON[4]	FBD	O
J20	FBD3SBON[5]	FBD	O
H19	FBD3SBON[6]	FBD	O
J18	FBD3SBON[7]	FBD	O
K18	FBD3SBON[8]	FBD	O
H21	FBD3SBON[9]	FBD	O
D22	FBD3SBOP[0]	FBD	O
G26	FBD3SBOP[1]	FBD	O
H25	FBD3SBOP[2]	FBD	O
J24	FBD3SBOP[3]	FBD	O
G23	FBD3SBOP[4]	FBD	O
J21	FBD3SBOP[5]	FBD	O
G19	FBD3SBOP[6]	FBD	O
H18	FBD3SBOP[7]	FBD	O
K19	FBD3SBOP[8]	FBD	O
H22	FBD3SBOP[9]	FBD	O
E37	FBDBGBIASEXT	Analog	
F35	FBDICOMPBIAS	Analog	
E36	FBDRESIN	Analog	
AG24	FSB0A[10]#	Source Sync	I/O
AJ22	FSB0A[11]#	Source Sync	I/O
AL22	FSB0A[12]#	Source Sync	I/O
AH22	FSB0A[13]#	Source Sync	I/O
AF25	FSB0A[14]#	Source Sync	I/O
AG23	FSB0A[15]#	Source Sync	I/O
AF22	FSB0A[16]#	Source Sync	I/O
AV28	FSB0A[17]#	Source Sync	I/O
AP25	FSB0A[18]#	Source Sync	I/O
AR27	FSB0A[19]#	Source Sync	I/O
AR24	FSB0A[20]#	Source Sync	I/O
AU28	FSB0A[21]#	Source Sync	I/O
AU25	FSB0A[22]#	Source Sync	I/O
AT27	FSB0A[23]#	Source Sync	I/O
AT26	FSB0A[24]#	Source Sync	I/O
AU26	FSB0A[25]#	Source Sync	I/O
AL34	FSB0D[17]#	Source Sync	I/O
AP35	FSB0D[18]#	Source Sync	I/O
AT36	FSB0D[19]#	Source Sync	I/O

Table 8-2. Intel 5000X Chipset MCH Signals (by Signal Name) (Sheet 4 of 19)

Ball No.	Signal Name	Buffer Type	Direction
AU23	FSB0A[29]#	Source Sync	I/O
AL25	FSB0A[3]#	Source Sync	I/O
AT23	FSB0A[30]#	Source Sync	I/O
AV24	FSB0A[31]#	Source Sync	I/O
AP22	FSB0A[32]#	Source Sync	I/O
AR22	FSB0A[33]#	Source Sync	I/O
AU22	FSB0A[34]#	Source Sync	I/O
AV22	FSB0A[35]#	Source Sync	I/O
AN24	FSB0A[4]#	Source Sync	I/O
AM25	FSB0A[5]#	Source Sync	I/O
AN26	FSB0A[6]#	Source Sync	I/O
AP26	FSB0A[7]#	Source Sync	I/O
AH23	FSB0A[8]#	Source Sync	I/O
AM22	FSB0A[9]#	Source Sync	I/O
AU29	FSB0ADS#	Common Clk	I/O
AL23	FSB0ADSTB[0]#	Source Sync	I/O
AP23	FSB0ADSTB[1]#	Source Sync	I/O
AH26	FSB0AP[0]#	Common Clk	I/O
AK26	FSB0AP[1]#	Source Sync	I/O
AK27	FSB0BINIT#	Common Clk	I/O
AV30	FSB0BNR#	Common Clk	I/O
AR28	FSB0BPM[4]#	Common Clk	I/O
AP29	FSB0BPM[5]#	Common Clk	I/O
AU34	FSB0BPRI#	Common Clk	O
AM28	FSB0BREQ[0]#	Common Clk	I/O
AG26	FSB0BREQ[1]#	Common Clk	I/O
AJ28	FSB0D[0]#	Source Sync	I/O
AL29	FSB0D[1]#	Source Sync	I/O
AJ30	FSB0D[10]#	Source Sync	I/O
AH31	FSB0D[11]#	Source Sync	I/O
AM33	FSB0D[12]#	Source Sync	I/O
AG29	FSB0D[13]#	Source Sync	I/O
AG27	FSB0D[14]#	Source Sync	I/O
AJ33	FSB0D[15]#	Source Sync	I/O
AN33	FSB0D[16]#	Source Sync	I/O
AL36	FSB0D[51]#	Source Sync	I/O
AL37	FSB0D[52]#	Source Sync	I/O
AK36	FSB0D[53]#	Source Sync	I/O
AG35	FSB0D[54]#	Source Sync	I/O
AH28	FSB0D[2]#	Source Sync	I/O
AP34	FSB0D[20]#	Source Sync	I/O
AU36	FSB0D[21]#	Source Sync	I/O
AT37	FSB0D[22]#	Source Sync	I/O
AP36	FSB0D[23]#	Source Sync	I/O
AN35	FSB0D[24]#	Source Sync	I/O
AL35	FSB0D[25]#	Source Sync	I/O
AN36	FSB0D[26]#	Source Sync	I/O
AP38	FSB0D[27]#	Source Sync	I/O
AM38	FSB0D[28]#	Source Sync	I/O
AM34	FSB0D[29]#	Source Sync	I/O
AK29	FSB0D[3]#	Source Sync	I/O
AK35	FSB0D[30]#	Source Sync	I/O
AM37	FSB0D[31]#	Source Sync	I/O
AG33	FSB0D[32]#	Source Sync	I/O
AG30	FSB0D[33]#	Source Sync	I/O
AE34	FSB0D[34]#	Source Sync	I/O
AF31	FSB0D[35]#	Source Sync	I/O
AG32	FSB0D[36]#	Source Sync	I/O
AF33	FSB0D[37]#	Source Sync	I/O
AD35	FSB0D[38]#	Source Sync	I/O
AE32	FSB0D[39]#	Source Sync	I/O
AH29	FSB0D[4]#	Source Sync	I/O
AE31	FSB0D[40]#	Source Sync	I/O
AD30	FSB0D[41]#	Source Sync	I/O
AC30	FSB0D[42]#	Source Sync	I/O
AE29	FSB0D[43]#	Source Sync	I/O
AC31	FSB0D[44]#	Source Sync	I/O
AF28	FSB0D[45]#	Source Sync	I/O
AD29	FSB0D[46]#	Source Sync	I/O
AE28	FSB0D[47]#	Source Sync	I/O
AF37	FSB0D[48]#	Source Sync	I/O
AJ34	FSB0D[49]#	Source Sync	I/O
AN32	FSB0D[5]#	Source Sync	I/O
AL38	FSB0D[50]#	Source Sync	I/O
AT30	FSB0LOCK#	Common Clk	I/O
AJ27	FSB0MCERR#	Common Clk	I/O
AL26	FSB0REQ[0]#	Source Sync	I/O
AH25	FSB0REQ[1]#	Source Sync	I/O

Table 8-2. Intel 5000X Chipset MCH Signals (by Signal Name) (Sheet 5 of 19)

Ball No.	Signal Name	Buffer Type	Direction
AJ37	FSB0D[55]#	Source Sync	I/O
AJ38	FSB0D[56]#	Source Sync	I/O
AH38	FSB0D[57]#	Source Sync	I/O
AE38	FSB0D[58]#	Source Sync	I/O
AF38	FSB0D[59]#	Source Sync	I/O
AK30	FSB0D[6]#	Source Sync	I/O
AG36	FSB0D[60]#	Source Sync	I/O
AH36	FSB0D[61]#	Source Sync	I/O
AE36	FSB0D[62]#	Source Sync	I/O
AE37	FSB0D[63]#	Source Sync	I/O
AL31	FSB0D[7]#	Source Sync	I/O
AJ31	FSB0D[8]#	Source Sync	I/O
AH32	FSB0D[9]#	Source Sync	I/O
AL32	FSB0DBI[0]#	Source Sync	I/O
AP37	FSB0DBI[1]#	Source Sync	I/O
AF30	FSB0DBI[2]#	Source Sync	I/O
AH37	FSB0DBI[3]#	Source Sync	I/O
AR30	FSB0DBSY#	Common Clk	I/O
AV34	FSB0DEFER#	Common Clk	O
AR31	FSB0DP[0]#	Common Clk	I/O
AT33	FSB0DP[1]#	Common Clk	I/O
AP31	FSB0DP[2]#	Common Clk	I/O
AN29	FSB0DP[3]#	Common Clk	I/O
AT29	FSB0DRDY#	Common Clk	I/O
AK32	FSB0DSTBN[0]#	Source Sync	I/O
AR37	FSB0DSTBN[1]#	Source Sync	I/O
AD32	FSB0DSTBN[2]#	Source Sync	I/O
AH34	FSB0DSTBN[3]#	Source Sync	I/O
AK33	FSB0DSTBP[0]#	Source Sync	I/O
AR38	FSB0DSTBP[1]#	Source Sync	I/O
AD33	FSB0DSTBP[2]#	Source Sync	I/O
AH35	FSB0DSTBP[3]#	Source Sync	I/O
AU32	FSB0HIT#	Common Clk	I/M105O
AV33	FSB0HITM#	Common Clk	I/O
AE2	FSB1A[31]#	Source Sync	I/O
AE1	FSB1A[32]#	Source Sync	I/O
AD2	FSB1A[33]#	Source Sync	I/O
AC4	FSB1A[34]#	Source Sync	I/O
AC3	FSB1A[35]#	Source Sync	I/O

Ball No.	Signal Name	Buffer Type	Direction
AK24	FSB0REQ[2]#	Source Sync	I/O
AJ24	FSB0REQ[3]#	Source Sync	I/O
AJ25	FSB0REQ[4]#	Source Sync	I/O
AN30	FSB0RESET#	Common Clk	I
AV31	FSB0RS[0]#	Common Clk	I
AL28	FSB0RS[1]#	Common Clk	I
AU31	FSB0RS[2]#	Common Clk	I
AN27	FSB0RSP#	Common Clk	I
AT32	FSB0TRDY#	Common Clk	O
AF34	FSB0VREF	Power/Other	
AM27	FSB0VREF	Power/Other	
AM30	FSB0VREF	Power/Other	
AB11	FSB1A[10]#	Source Sync	I/O
AF6	FSB1A[11]#	Source Sync	I/O
AD8	FSB1A[12]#	Source Sync	I/O
AC9	FSB1A[13]#	Source Sync	I/O
AB10	FSB1A[14]#	Source Sync	I/O
AC7	FSB1A[15]#	Source Sync	I/O
AA12	FSB1A[16]#	Source Sync	I/O
AF4	FSB1A[17]#	Source Sync	I/O
AG5	FSB1A[18]#	Source Sync	I/O
AH5	FSB1A[19]#	Source Sync	I/O
AD6	FSB1A[20]#	Source Sync	I/O
AE5	FSB1A[21]#	Source Sync	I/O
AC6	FSB1A[22]#	Source Sync	I/O
AD5	FSB1A[23]#	Source Sync	I/O
AH2	FSB1A[24]#	Source Sync	I/O
AH1	FSB1A[25]#	Source Sync	I/O
AJ3	FSB1A[26]#	Source Sync	I/O
AF1	FSB1A[27]#	Source Sync	I/O
AF3	FSB1A[28]#	Source Sync	I/O
AD3	FSB1A[29]#	Source Sync	I/O
AD12	FSB1A[3]#	Source Sync	I/O
AE4	FSB1A[30]#	Source Sync	I/O
AP8	FSB1D[22]#	Source Sync	I/O
AN8	FSB1D[23]#	Source Sync	I/O
AN9	FSB1D[24]#	Source Sync	I/O
AN12	FSB1D[25]#	Source Sync	I/O
AM12	FSB1D[26]#	Source Sync	I/O

Table 8-2. Intel 5000X Chipset MCH Signals (by Signal Name) (Sheet 6 of 19)

Ball No.	Signal Name	Buffer Type	Direction
AD9	FSB1A[4]#	Source Sync	I/O
AC12	FSB1A[5]#	Source Sync	I/O
AH8	FSB1A[6]#	Source Sync	I/O
AG8	FSB1A[7]#	Source Sync	I/O
AF7	FSB1A[8]#	Source Sync	I/O
AE7	FSB1A[9]#	Source Sync	I/O
AP2	FSB1ADS#	Common Clk	I/O
AC10	FSB1ADSTB[0]#	Source Sync	I/O
AG3	FSB1ADSTB[1]#	Source Sync	I/O
AG12	FSB1AP[0]#	Common Clk	I/O
AG10	FSB1AP[1]#	Common Clk	I/O
AJ4	FSB1BINIT#	Common Clk	I/O
AK3	FSB1BNR#	Common Clk	I/O
AN2	FSB1BPM[4]#	Common Clk	I/O
AN3	FSB1BPM[5]#	Common Clk	I/O
AJ10	FSB1BPRI#	Common Clk	O
AL2	FSB1BREQ[0]#	Common Clk	I/O
AM1	FSB1BREQ[1]#	Common Clk	I/O
AP1	FSB1D[0]#	Source Sync	I/O
AP5	FSB1D[1]#	Source Sync	I/O
AV4	FSB1D[10]#	Source Sync	I/O
AT6	FSB1D[11]#	Source Sync	I/O
AR6	FSB1D[12]#	Source Sync	I/O
AU7	FSB1D[13]#	Source Sync	I/O
AR7	FSB1D[14]#	Source Sync	I/O
AU5	FSB1D[15]#	Source Sync	I/O
AL7	FSB1D[16]#	Source Sync	I/O
AL8	FSB1D[17]#	Source Sync	I/O
AN6	FSB1D[18]#	Source Sync	I/O
AK9	FSB1D[19]#	Source Sync	I/O
AP4	FSB1D[2]#	Source Sync	I/O
AM6	FSB1D[20]#	Source Sync	I/O
AM9	FSB1D[21]#	Source Sync	I/O
AJ13	FSB1D[57]#	Source Sync	I/O
AF15	FSB1D[58]#	Source Sync	I/O
AG15	FSB1D[59]#	Source Sync	I/O
AT2	FSB1D[6]#	Source Sync	I/O
AJ15	FSB1D[60]#	Source Sync	I/O
AJ16	FSB1D[61]#	Source Sync	I/O

Ball No.	Signal Name	Buffer Type	Direction
AN11	FSB1D[27]#	Source Sync	I/O
AM13	FSB1D[28]#	Source Sync	I/O
AL11	FSB1D[29]#	Source Sync	I/O
AR1	FSB1D[3]#	Source Sync	I/O
AL13	FSB1D[30]#	Source Sync	I/O
AK12	FSB1D[31]#	Source Sync	I/O
AV7	FSB1D[32]#	Source Sync	I/O
AU8	FSB1D[33]#	Source Sync	I/O
AT9	FSB1D[34]#	Source Sync	I/O
AR9	FSB1D[35]#	Source Sync	I/O
AT8	FSB1D[36]#	Source Sync	I/O
AV9	FSB1D[37]#	Source Sync	I/O
AU10	FSB1D[38]#	Source Sync	I/O
AV10	FSB1D[39]#	Source Sync	I/O
AR3	FSB1D[4]#	Source Sync	I/O
AV12	FSB1D[40]#	Source Sync	I/O
AT11	FSB1D[41]#	Source Sync	I/O
AT12	FSB1D[42]#	Source Sync	I/O
AP13	FSB1D[43]#	Source Sync	I/O
AP11	FSB1D[44]#	Source Sync	I/O
AR13	FSB1D[45]#	Source Sync	I/O
AR12	FSB1D[46]#	Source Sync	I/O
AP14	FSB1D[47]#	Source Sync	I/O
AH14	FSB1D[48]#	Source Sync	I/O
AM16	FSB1D[49]#	Source Sync	I/O
AR4	FSB1D[5]#	Source Sync	I/O
AN14	FSB1D[50]#	Source Sync	I/O
AM15	FSB1D[51]#	Source Sync	I/O
AL14	FSB1D[52]#	Source Sync	I/O
AN15	FSB1D[53]#	Source Sync	I/O
AH16	FSB1D[54]#	Source Sync	I/O
AP16	FSB1D[55]#	Source Sync	I/O
AL16	FSB1D[56]#	Source Sync	I/O
AE10	FSB1REQ[4]#	Source Sync	I/O
AE11	FSB1RESET#	Common Clk	I
AK5	FSB1RS[0]#	Common Clk	I
AL1	FSB1RS[1]#	Common Clk	I
AL5	FSB1RS[2]#	Common Clk	I
AK2	FSB1RSP#	Common Clk	I

Table 8-2. Intel 5000X Chipset MCH Signals (by Signal Name) (Sheet 7 of 19)

Ball No.	Signal Name	Buffer Type	Direction
AG14	FSB1D[62]#	Source Sync	I/O
AF16	FSB1D[63]#	Source Sync	I/O
AT3	FSB1D[7]#	Source Sync	I/O
AT5	FSB1D[8]#	Source Sync	I/O
AV6	FSB1D[9]#	Source Sync	I/O
AP7	FSB1DBI[0]#	Source Sync	I/O
AK11	FSB1DBI[1]#	Source Sync	I/O
AU11	FSB1DBI[2]#	Source Sync	I/O
AH13	FSB1DBI[3]#	Source Sync	I/O
AM4	FSB1DBSY#	Common Clk	I/O
AJ9	FSB1DEFER#	Common Clk	O
AG11	FSB1DP[0]#	Common Clk	I/O
AJ12	FSB1DP[1]#	Common Clk	I/O
AF12	FSB1DP[2]#	Common Clk	I/O
AF13	FSB1DP[3]#	Common Clk	I/O
AM3	FSB1DRDY#	Common Clk	I/O
AU3	FSB1DSTBN[0]#	Source Sync	I/O
AL10	FSB1DSTBN[1]#	Source Sync	I/O
AP10	FSB1DSTBN[2]#	Source Sync	I/O
AK14	FSB1DSTBN[3]#	Source Sync	I/O
AU4	FSB1DSTBP[0]#	Source Sync	I/O
AM10	FSB1DSTBP[1]#	Source Sync	I/O
AR10	FSB1DSTBP[2]#	Source Sync	I/O
AK15	FSB1DSTBP[3]#	Source Sync	I/O
AK8	FSB1HIT#	Common Clk	I/O
AJ7	FSB1HITM#	Common Clk	I/O
AL4	FSB1LOCK#	Common Clk	I/O
AH11	FSB1MCERR#	Common Clk	I/O
AG9	FSB1REQ[0]#	Source Sync	I/O
AD11	FSB1REQ[1]#	Source Sync	I/O
AJ6	FSB1REQ[2]#	Source Sync	I/O
AF9	FSB1REQ[3]#	Source Sync	I/O
N4	PE2RP[1]	PEX	I
P3	PE2RP[2]	PEX	I
T1	PE2RP[3]	PEX	I
T7	PE2TN[0]	PEX	O
T4	PE2TN[1]	PEX	O
P1	PE2TN[2]	PEX	O
T2	PE2TN[3]	PEX	O

Ball No.	Signal Name	Buffer Type	Direction
AK6	FSB1TRDY#	Common Clk	O
AH4	FSB1VREF	Power/Other	
AN5	FSB1VREF	Power/Other	
AT14	FSB1VREF	Power/Other	
AT35	FSBCRES	Analog	
AR34	FSBODTCRES	Analog	
AU35	FSBSLWCRES	Analog	
AV13	FSBSLWCTRL	Power/Other	
AU17	FSBVCCA	Power/Other	
K13	GPIOSMBCLK	SMB	I/O
L12	GPIOSMBDATA	SMB	I/O
Y9	PE0RN[0]	PEX	I
Y3	PE0RN[1]	PEX	I
AB7	PE0RN[2]	PEX	I
AA6	PE0RN[3]	PEX	I
Y10	PE0RP[0]	PEX	I
Y4	PE0RP[1]	PEX	I
AB8	PE0RP[2]	PEX	I
AA5	PE0RP[3]	PEX	I
Y6	PE0TN[0]	PEX	O
AA2	PE0TN[1]	PEX	O
AB5	PE0TN[2]	PEX	O
AA9	PE0TN[3]	PEX	O
Y7	PE0TP[0]	PEX	O
AA3	PE0TP[1]	PEX	O
AB4	PE0TP[2]	PEX	O
AA8	PE0TP[3]	PEX	O
R5	PE2RN[0]	PEX	I
P4	PE2RN[1]	PEX	I
R3	PE2RN[2]	PEX	I
U1	PE2RN[3]	PEX	I
T5	PE2RP[0]	PEX	I
K11	PE4TN[3]	PEX	O
H12	PE4TP[0]	PEX	O
C12	PE4TP[1]	PEX	O
C11	PE4TP[2]	PEX	O
J11	PE4TP[3]	PEX	O
G10	PE5RN[0]	PEX	I
B9	PE5RN[1]	PEX	I

Table 8-2. Intel 5000X Chipset MCH Signals (by Signal Name) (Sheet 8 of 19)

Ball No.	Signal Name	Buffer Type	Direction
T8	PE2TP[0]	PEX	O
U4	PE2TP[1]	PEX	O
N1	PE2TP[2]	PEX	O
R2	PE2TP[3]	PEX	O
W2	PE3RN[0]	PEX	I
V6	PE3RN[1]	PEX	I
W8	PE3RN[2]	PEX	I
U10	PE3RN[3]	PEX	I
V2	PE3RP[0]	PEX	I
V5	PE3RP[1]	PEX	I
W7	PE3RP[2]	PEX	I
U9	PE3RP[3]	PEX	I
V3	PE3TN[0]	PEX	O
W4	PE3TN[1]	PEX	O
U7	PE3TN[2]	PEX	O
V9	PE3TN[3]	PEX	O
U3	PE3TP[0]	PEX	O
W5	PE3TP[1]	PEX	O
U6	PE3TP[2]	PEX	O
V8	PE3TP[3]	PEX	O
E12	PE4RN[0]	PEX	I
F11	PE4RN[1]	PEX	I
E10	PE4RN[2]	PEX	I
L10	PE4RN[3]	PEX	I
F12	PE4RP[0]	PEX	I
G11	PE4RP[1]	PEX	I
D10	PE4RP[2]	PEX	I
K10	PE4RP[3]	PEX	I
J12	PE4TN[0]	PEX	O
B12	PE4TN[1]	PEX	O
D11	PE4TN[2]	PEX	O
E1	PE7RN[1]	PEX	I
H4	PE7RN[2]	PEX	I
L4	PE7RN[3]	PEX	I
F3	PE7RP[0]	PEX	I
D1	PE7RP[1]	PEX	I
H3	PE7RP[2]	PEX	I
K4	PE7RP[3]	PEX	I
G4	PE7TN[0]	PEX	O

Ball No.	Signal Name	Buffer Type	Direction
G8	PE5RN[2]	PEX	I
H7	PE5RN[3]	PEX	I
H10	PE5RP[0]	PEX	I
C9	PE5RP[1]	PEX	I
F8	PE5RP[2]	PEX	I
G7	PE5RP[3]	PEX	I
H9	PE5TN[0]	PEX	O
E9	PE5TN[1]	PEX	O
C8	PE5TN[2]	PEX	O
E7	PE5TN[3]	PEX	O
J9	PE5TP[0]	PEX	O
F9	PE5TP[1]	PEX	O
D8	PE5TP[2]	PEX	O
D7	PE5TP[3]	PEX	O
C5	PE6RN[0]	PEX	I
E3	PE6RN[1]	PEX	I
F5	PE6RN[2]	PEX	I
K8	PE6RN[3]	PEX	I
C6	PE6RP[0]	PEX	I
E4	PE6RP[1]	PEX	I
F6	PE6RP[2]	PEX	I
J8	PE6RP[3]	PEX	I
M8	PE6TN[0]	PEX	O
D4	PE6TN[1]	PEX	O
C2	PE6TN[2]	PEX	O
J6	PE6TN[3]	PEX	O
M9	PE6TP[0]	PEX	O
D5	PE6TP[1]	PEX	O
C3	PE6TP[2]	PEX	O
H6	PE6TP[3]	PEX	O
F2	PE7RN[0]	PEX	I
M2	RSVD	No Connect	
M3	RSVD	No Connect	
M5	RSVD	No Connect	
M6	RSVD	No Connect	
M12	RSVD	No Connect	
N2	RSVD	No Connect	
N5	RSVD	No Connect	
N7	RSVD	No Connect	

Table 8-2. Intel 5000X Chipset MCH Signals (by Signal Name) (Sheet 9 of 19)

Ball No.	Signal Name	Buffer Type	Direction
G1	PE7TN[1]	PEX	O
L7	PE7TN[2]	PEX	O
K5	PE7TN[3]	PEX	O
G5	PE7TP[0]	PEX	O
G2	PE7TP[1]	PEX	O
K7	PE7TP[2]	PEX	O
J5	PE7TP[3]	PEX	O
K2	PECLKN	Analog	
J2	PECLKP	Analog	
R12	PEICOMPI	Analog	
P12	PERCOMPO	Analog	
K1	PEVCCA	Analog	
R11	PEVCCBG	Analog	
L1	PEVSSA	Analog	
N11	PEVSSBG	Analog	
AA11	PEWIDTH[0]	Power/Other	I
Y12	PEWIDTH[1]	Power/Other	I
W11	PEWIDTH[2]	Power/Other	I
W10	PEWIDTH[3]	Power/Other	I
AC1	PSEL[0]	CMOS	I
AB2	PSEL[1]	CMOS	I
AB1	PSEL[2]	CMOS	I
H17	PWRGOOD	CMOS	I
G17	RESETI#	CMOS	I
M11	RSV1	No Connect	
D29	RSVD	No Connect	
H1	RSVD	No Connect	
J3	RSVD	No Connect	
L3	RSVD	No Connect	
L6	RSVD	No Connect	
G13	SPD0SMBDATA	SMB	I/O
J16	SPD1SMBCLK	SMB	I/O
K15	SPD1SMBDATA	SMB	I/O
F15	SPD2SMBCLK	SMB	I/O
E15	SPD2SMBDATA	SMB	I/O
H15	SPD3SMBCLK	SMB	I/O
H16	SPD3SMBDATA	SMB	I/O
A6	TCK	JTAG	I
B7	TDI	JTAG	I

Ball No.	Signal Name	Buffer Type	Direction
N8	RSVD	No Connect	
N10	RSVD	No Connect	
P6	RSVD	No Connect	
P7	RSVD	No Connect	
P9	RSVD	No Connect	
P10	RSVD	No Connect	
R6	RSVD	No Connect	
R8	RSVD	No Connect	
R9	RSVD	No Connect	
T37	RSVD	No Connect	
W1	RSVD	No Connect	
Y1	RSVD	No Connect	
AE8	RSVD	No Connect	
AG2	RSVD	No Connect	
AG6	RSVD	No Connect	
AH7	RSVD	No Connect	
AJ1	RSVD	No Connect	
AK17	RSVD	No Connect	
AK23	RSVD	No Connect	
AM7	RSVD	No Connect	
AM24	RSVD	No Connect	
AM31	RSVD	No Connect	
AN23	RSVD	No Connect	
AP28	RSVD	No Connect	
AP32	RSVD	No Connect	
AR15	RSVD	No Connect	
AR16	RSVD	No Connect	
AR33	RSVD	No Connect	
AV27	RSVD	No Connect	
H13	SPD0SMBCLK	SMB	I/O
T18	VCC	Power/Other	
T20	VCC	Power/Other	
T22	VCC	Power/Other	
T24	VCC	Power/Other	
U15	VCC	Power/Other	
U17	VCC	Power/Other	
U19	VCC	Power/Other	
U21	VCC	Power/Other	
U23	VCC	Power/Other	

Table 8-2. Intel 5000X Chipset MCH Signals (by Signal Name) (Sheet 10 of 19)

Ball No.	Signal Name	Buffer Type	Direction
A4	TDIOANODE	Analog	
B4	TDIOCATHODE	Analog	
B6	TDO	JTAG	Ouput
AC36	TESTHI	No Connect	
F17	TESTHI_V3REF	Power/Other	
G16	TESTHI_V3REF	Power/Other	
A7	TMS	JTAG	I
A8	TRST#	JTAG	I
F13	V3REF	Analog	
L16	VCC	Power/Other	
L17	VCC	Power/Other	
L18	VCC	Power/Other	
L19	VCC	Power/Other	
M16	VCC	Power/Other	
M17	VCC	Power/Other	
M18	VCC	Power/Other	
N17	VCC	Power/Other	
N19	VCC	Power/Other	
P16	VCC	Power/Other	
P18	VCC	Power/Other	
P20	VCC	Power/Other	
P22	VCC	Power/Other	
P24	VCC	Power/Other	
R15	VCC	Power/Other	
R17	VCC	Power/Other	
R19	VCC	Power/Other	
R21	VCC	Power/Other	
R23	VCC	Power/Other	
T16	VCC	Power/Other	
AC17	VCC	Power/Other	
AC19	VCC	Power/Other	
AC21	VCC	Power/Other	
AC23	VCC	Power/Other	
AC25	VCC	Power/Other	
AC26	VCC	Power/Other	
AD26	VCC	Power/Other	
AE35	VCC	Power/Other	
AH10	VCC	Power/Other	
AL17	VCC	Power/Other	
V16	VCC	Power/Other	
V18	VCC	Power/Other	
V20	VCC	Power/Other	
V22	VCC	Power/Other	
V24	VCC	Power/Other	
W15	VCC	Power/Other	
W17	VCC	Power/Other	
W19	VCC	Power/Other	
W21	VCC	Power/Other	
W23	VCC	Power/Other	
Y16	VCC	Power/Other	
Y18	VCC	Power/Other	
Y20	VCC	Power/Other	
Y22	VCC	Power/Other	
Y24	VCC	Power/Other	
AA13	VCC	Power/Other	
AA15	VCC	Power/Other	
AA17	VCC	Power/Other	
AA19	VCC	Power/Other	
AA21	VCC	Power/Other	
AA23	VCC	Power/Other	
AB13	VCC	Power/Other	
AB14	VCC	Power/Other	
AB16	VCC	Power/Other	
AB18	VCC	Power/Other	
AB20	VCC	Power/Other	
AB22	VCC	Power/Other	
AB24	VCC	Power/Other	
AC15	VCC	Power/Other	
R25	VCCFBD	Power/Other	
T25	VCCFBD	Power/Other	
T26	VCCFBD	Power/Other	
T27	VCCFBD	FBD	I
U25	VCCFBD	Power/Other	
U26	VCCFBD	Power/Other	
V25	VCCFBD	Power/Other	
V26	VCCFBD	Power/Other	
W25	VCCFBD	Power/Other	
W26	VCCFBD	Power/Other	

Table 8-2. Intel 5000X Chipset MCH Signals (by Signal Name) (Sheet 11 of 19)

Ball No.	Signal Name	Buffer Type	Direction
A20	VCCFBD	Power/Other	
E20	VCCFBD	Power/Other	
E23	VCCFBD	Power/Other	
F25	VCCFBD	Power/Other	
H20	VCCFBD	Power/Other	
H23	VCCFBD	Power/Other	
K21	VCCFBD	Power/Other	
K22	VCCFBD	Power/Other	
K23	VCCFBD	Power/Other	
L20	VCCFBD	Power/Other	
L21	VCCFBD	Power/Other	
L22	VCCFBD	Power/Other	
L23	VCCFBD	Power/Other	
M20	VCCFBD	Power/Other	
M21	VCCFBD	Power/Other	
M22	VCCFBD	Power/Other	
M23	VCCFBD	Power/Other	
M24	VCCFBD	Power/Other	
M25	VCCFBD	Power/Other	
N20	VCCFBD	Power/Other	
N21	VCCFBD	Power/Other	
N22	VCCFBD	Power/Other	
N23	VCCFBD	Power/Other	
N24	VCCFBD	Power/Other	
N25	VCCFBD	Power/Other	
N26	VCCFBD	Power/Other	
P25	VCCFBD	Power/Other	
P26	VCCFBD	Power/Other	
T13	VCCPE	Power/Other	
T14	VCCPE	Power/Other	
U2	VCCPE	Power/Other	
U8	VCCPE	Power/Other	
U13	VCCPE	Power/Other	
U14	VCCPE	Power/Other	
V13	VCCPE	Power/Other	
V14	VCCPE	Power/Other	
W6	VCCPE	Power/Other	
W12	VCCPE	Power/Other	
W13	VCCPE	Power/Other	
Y25	VCCFBD	Power/Other	
Y26	VCCFBD	Power/Other	
AA25	VCCFBD	Power/Other	
AA26	VCCFBD	Power/Other	
AA27	VCCFBD	Power/Other	
AB25	VCCFBD	Power/Other	
AB26	VCCFBD	Power/Other	
G12	VCCPE	Power/Other	
J10	VCCPE	Power/Other	
L2	VCCPE	Power/Other	
L8	VCCPE	Power/Other	
L13	VCCPE	Power/Other	
L14	VCCPE	Power/Other	
L15	VCCPE	Power/Other	
M13	VCCPE	Power/Other	
M14	VCCPE	Power/Other	
M15	VCCPE	Power/Other	
N6	VCCPE	Power/Other	
N12	VCCPE	Power/Other	
N13	VCCPE	Power/Other	
N14	VCCPE	Power/Other	
N15	VCCPE	Power/Other	
P13	VCCPE	Power/Other	
P14	VCCPE	Power/Other	
R4	VCCPE	Power/Other	
R10	VCCPE	Power/Other	
R13	VCCPE	Power/Other	
R14	VCCPE	Power/Other	
C7	VSS	Power/Other	
C10	VSS	Power/Other	
C15	VSS	Power/Other	
C19	VSS	Power/Other	
C22	VSS	Power/Other	
C24	VSS	Power/Other	
C26	VSS	Power/Other	
C27	VSS	Power/Other	
C29	VSS	Power/Other	
C30	VSS	Power/Other	
C32	VSS	Power/Other	

Table 8-2. Intel 5000X Chipset MCH Signals (by Signal Name) (Sheet 12 of 19)

Ball No.	Signal Name	Buffer Type	Direction
W14	VCCPE	Power/Other	
Y13	VCCPE	Power/Other	
Y14	VCCPE	Power/Other	
L24	VCCSEN	Temp	
A3	VSS	Power/Other	
A9	VSS	Power/Other	
A13	VSS	Power/Other	
A18	VSS	Power/Other	
A21	VSS	Power/Other	
A23	VSS	Power/Other	
A25	VSS	Power/Other	
A28	VSS	Power/Other	
A31	VSS	Power/Other	
A34	VSS	Power/Other	
A36	VSS	Power/Other	
B2	VSS	Power/Other	
B3	VSS	Power/Other	
B5	VSS	Power/Other	
B8	VSS	Power/Other	
B11	VSS	Power/Other	
B16	VSS	Power/Other	
B17	VSS	Power/Other	
B20	VSS	Power/Other	
B23	VSS	Power/Other	
B37	VSS	Power/Other	
C1	VSS	Power/Other	
C4	VSS	Power/Other	
F4	VSS	Power/Other	
F7	VSS	Power/Other	
F10	VSS	Power/Other	
F16	VSS	Power/Other	
F19	VSS	Power/Other	
F21	VSS	Power/Other	
F22	VSS	Power/Other	
F26	VSS	Power/Other	
F28	VSS	Power/Other	
F31	VSS	Power/Other	
F34	VSS	Power/Other	
F38	VSS	Power/Other	

Ball No.	Signal Name	Buffer Type	Direction
C33	VSS	Power/Other	
C35	VSS	Power/Other	
C38	VSS	Power/Other	
D3	VSS	Power/Other	
D6	VSS	Power/Other	
D9	VSS	Power/Other	
D12	VSS	Power/Other	
D14	VSS	Power/Other	
D18	VSS	Power/Other	
D21	VSS	Power/Other	
D24	VSS	Power/Other	
D27	VSS	Power/Other	
D30	VSS	Power/Other	
D33	VSS	Power/Other	
D36	VSS	Power/Other	
E2	VSS	Power/Other	
E5	VSS	Power/Other	
E8	VSS	Power/Other	
E11	VSS	Power/Other	
E13	VSS	Power/Other	
E17	VSS	Power/Other	
E21	VSS	Power/Other	
E26	VSS	Power/Other	
E29	VSS	Power/Other	
E32	VSS	Power/Other	
E35	VSS	Power/Other	
F1	VSS	Power/Other	
H35	VSS	Power/Other	
J1	VSS	Power/Other	
J4	VSS	Power/Other	
J7	VSS	Power/Other	
J13	VSS	Power/Other	
J17	VSS	Power/Other	
J19	VSS	Power/Other	
J22	VSS	Power/Other	
J25	VSS	Power/Other	
J26	VSS	Power/Other	
J27	VSS	Power/Other	
J28	VSS	Power/Other	

Table 8-2. Intel 5000X Chipset MCH Signals (by Signal Name) (Sheet 13 of 19)

Ball No.	Signal Name	Buffer Type	Direction
G3	VSS	Power/Other	
G6	VSS	Power/Other	
G9	VSS	Power/Other	
G15	VSS	Power/Other	
G18	VSS	Power/Other	
G21	VSS	Power/Other	
G24	VSS	Power/Other	
G27	VSS	Power/Other	
G30	VSS	Power/Other	
G31	VSS	Power/Other	
G32	VSS	Power/Other	
G33	VSS	Power/Other	
G36	VSS	Power/Other	
G37	VSS	Power/Other	
H2	VSS	Power/Other	
H5	VSS	Power/Other	
H8	VSS	Power/Other	
H11	VSS	Power/Other	
H14	VSS	Power/Other	
H26	VSS	Power/Other	
H27	VSS	Power/Other	
H28	VSS	Power/Other	
H29	VSS	Power/Other	
H30	VSS	Power/Other	
H31	VSS	Power/Other	
H32	VSS	Power/Other	
L32	VSS	Power/Other	
L35	VSS	Power/Other	
M1	VSS	Power/Other	
M4	VSS	Power/Other	
M7	VSS	Power/Other	
M10	VSS	Power/Other	
M19	VSS	Power/Other	
M28	VSS	Power/Other	
M31	VSS	Power/Other	
M34	VSS	Power/Other	
M37	VSS	Power/Other	
M38	VSS	Power/Other	
N3	VSS	Power/Other	
J29	VSS	Power/Other	
J30	VSS	Power/Other	
J31	VSS	Power/Other	
J34	VSS	Power/Other	
J38	VSS	Power/Other	
K3	VSS	Power/Other	
K6	VSS	Power/Other	
K9	VSS	Power/Other	
K12	VSS	Power/Other	
K16	VSS	Power/Other	
K17	VSS	Power/Other	
K20	VSS	Power/Other	
K24	VSS	Power/Other	
K25	VSS	Power/Other	
K26	VSS	Power/Other	
K27	VSS	Power/Other	
K28	VSS	Power/Other	
K29	VSS	Power/Other	
K30	VSS	Power/Other	
K33	VSS	Power/Other	
K36	VSS	Power/Other	
K37	VSS	Power/Other	
L5	VSS	Power/Other	
L11	VSS	Power/Other	
L26	VSS	Power/Other	
L29	VSS	Power/Other	
R16	VSS	Power/Other	
R18	VSS	Power/Other	
R20	VSS	Power/Other	
R22	VSS	Power/Other	
R24	VSS	Power/Other	
R26	VSS	Power/Other	
R27	VSS	Power/Other	
R28	VSS	Power/Other	
R29	VSS	Power/Other	
R30	VSS	Power/Other	
R31	VSS	Power/Other	
R34	VSS	Power/Other	
R37	VSS	Power/Other	

Table 8-2. Intel 5000X Chipset MCH Signals (by Signal Name) (Sheet 14 of 19)

Ball No.	Signal Name	Buffer Type	Direction	Ball No.	Signal Name	Buffer Type	Direction
N9	VSS	Power/Other		T3	VSS	Power/Other	
N16	VSS	Power/Other		T6	VSS	Power/Other	
N18	VSS	Power/Other		T9	VSS	Power/Other	
N27	VSS	Power/Other		T10	VSS	Power/Other	
N30	VSS	Power/Other		T11	VSS	Power/Other	
N31	VSS	Power/Other		T12	VSS	Power/Other	
N33	VSS	Power/Other		T15	VSS	Power/Other	
N36	VSS	Power/Other		T17	VSS	Power/Other	
P2	VSS	Power/Other		T19	VSS	Power/Other	
P5	VSS	Power/Other		T21	VSS	Power/Other	
P8	VSS	Power/Other		T23	VSS	Power/Other	
P11	VSS	Power/Other		T29	VSS	Power/Other	
P15	VSS	Power/Other		T30	VSS	Power/Other	
P17	VSS	Power/Other		T33	VSS	Power/Other	
P19	VSS	Power/Other		T36	VSS	Power/Other	
P21	VSS	Power/Other		U5	VSS	Power/Other	
P23	VSS	Power/Other		U11	VSS	Power/Other	
P29	VSS	Power/Other		U12	VSS	Power/Other	
P30	VSS	Power/Other		U16	VSS	Power/Other	
P31	VSS	Power/Other		U18	VSS	Power/Other	
P32	VSS	Power/Other		U20	VSS	Power/Other	
P35	VSS	Power/Other		U22	VSS	Power/Other	
P38	VSS	Power/Other		U24	VSS	Power/Other	
R1	VSS	Power/Other		U29	VSS	Power/Other	
R7	VSS	Power/Other		U32	VSS	Power/Other	
U35	VSS	Power/Other		Y29	VSS	Power/Other	
U38	VSS	Power/Other		Y32	VSS	Power/Other	
V1	VSS	Power/Other		Y35	VSS	Power/Other	
V4	VSS	Power/Other		Y38	VSS	Power/Other	
V7	VSS	Power/Other		AA1	VSS	Power/Other	
V10	VSS	Power/Other		AA4	VSS	Power/Other	
V11	VSS	Power/Other		AA7	VSS	Power/Other	
V12	VSS	Power/Other		AA10	VSS	Power/Other	
V15	VSS	Power/Other		AA14	VSS	Power/Other	
V17	VSS	Power/Other		AA16	VSS	Power/Other	
V19	VSS	Power/Other		AA18	VSS	Power/Other	
V21	VSS	Power/Other		AA20	VSS	Power/Other	
V23	VSS	Power/Other		AA22	VSS	Power/Other	
V28	VSS	Power/Other		AA24	VSS	Power/Other	

Table 8-2. Intel 5000X Chipset MCH Signals (by Signal Name) (Sheet 15 of 19)

Ball No.	Signal Name	Buffer Type	Direction	Ball No.	Signal Name	Buffer Type	Direction
V31	VSS	Power/Other		AA28	VSS	Power/Other	
V34	VSS	Power/Other		AA29	VSS	Power/Other	
V37	VSS	Power/Other		AA30	VSS	Power/Other	
W3	VSS	Power/Other		AA31	VSS	Power/Other	
W9	VSS	Power/Other		AA34	VSS	Power/Other	
W16	VSS	Power/Other		AA37	VSS	Power/Other	
W18	VSS	Power/Other		AB3	VSS	Power/Other	
W20	VSS	Power/Other		AB6	VSS	Power/Other	
W22	VSS	Power/Other		AB9	VSS	Power/Other	
W24	VSS	Power/Other		AB12	VSS	Power/Other	
W27	VSS	Power/Other		AB15	VSS	Power/Other	
W30	VSS	Power/Other		AB17	VSS	Power/Other	
W33	VSS	Power/Other		AB19	VSS	Power/Other	
W36	VSS	Power/Other		AB21	VSS	Power/Other	
W37	VSS	Power/Other		AB23	VSS	Power/Other	
Y2	VSS	Power/Other		AB27	VSS	Power/Other	
Y5	VSS	Power/Other		AB28	VSS	Power/Other	
Y8	VSS	Power/Other		AB29	VSS	Power/Other	
Y11	VSS	Power/Other		AB30	VSS	Power/Other	
Y15	VSS	Power/Other		AB33	VSS	Power/Other	
Y17	VSS	Power/Other		AB36	VSS	Power/Other	
Y19	VSS	Power/Other		AC2	VSS	Power/Other	
Y21	VSS	Power/Other		AC5	VSS	Power/Other	
Y23	VSS	Power/Other		AC8	VSS	Power/Other	
AC11	VSS	Power/Other		AF24	VSS	Power/Other	
AC16	VSS	Power/Other		AF26	VSS	Power/Other	
AC18	VSS	Power/Other		AF27	VSS	Power/Other	
AC20	VSS	Power/Other		AF29	VSS	Power/Other	
AC22	VSS	Power/Other		AF32	VSS	Power/Other	
AC24	VSS	Power/Other		AF35	VSS	Power/Other	
AC27	VSS	Power/Other		AF36	VSS	Power/Other	
AC28	VSS	Power/Other		AG1	VSS	Power/Other	
AC29	VSS	Power/Other		AG4	VSS	Power/Other	
AC32	VSS	Power/Other		AG7	VSS	Power/Other	
AC35	VSS	Power/Other		AG13	VSS	Power/Other	
AC38	VSS	Power/Other		AG16	VSS	Power/Other	
AD1	VSS	Power/Other		AG17	VSS	Power/Other	
AD4	VSS	Power/Other		AG22	VSS	Power/Other	
AD7	VSS	Power/Other		AG25	VSS	Power/Other	

Table 8-2. Intel 5000X Chipset MCH Signals (by Signal Name) (Sheet 16 of 19)

Ball No.	Signal Name	Buffer Type	Direction
AD10	VSS	Power/Other	
AD27	VSS	Power/Other	
AD28	VSS	Power/Other	
AD31	VSS	Power/Other	
AD34	VSS	Power/Other	
AD36	VSS	Power/Other	
AD37	VSS	Power/Other	
AD38	VSS	Power/Other	
AE3	VSS	Power/Other	
AE6	VSS	Power/Other	
AE9	VSS	Power/Other	
AE12	VSS	Power/Other	
AE27	VSS	Power/Other	
AE30	VSS	Power/Other	
AE33	VSS	Power/Other	
AF2	VSS	Power/Other	
AF5	VSS	Power/Other	
AF8	VSS	Power/Other	
AF10	VSS	Power/Other	
AF11	VSS	Power/Other	
AF14	VSS	Power/Other	
AF17	VSS	Power/Other	
AF23	VSS	Power/Other	
AJ29	VSS	Power/Other	
AJ32	VSS	Power/Other	
AJ35	VSS	Power/Other	
AJ36	VSS	Power/Other	
AK1	VSS	Power/Other	
AK4	VSS	Power/Other	
AK7	VSS	Power/Other	
AK10	VSS	Power/Other	
AK13	VSS	Power/Other	
AK16	VSS	Power/Other	
AK22	VSS	Power/Other	
AK25	VSS	Power/Other	
AK28	VSS	Power/Other	
AK31	VSS	Power/Other	
AK34	VSS	Power/Other	
AK37	VSS	Power/Other	
AG28	VSS	Power/Other	
AG31	VSS	Power/Other	
AG34	VSS	Power/Other	
AG37	VSS	Power/Other	
AG38	VSS	Power/Other	
AH3	VSS	Power/Other	
AH6	VSS	Power/Other	
AH9	VSS	Power/Other	
AH12	VSS	Power/Other	
AH15	VSS	Power/Other	
AH17	VSS	No Connect	
AH24	VSS	Power/Other	
AH27	VSS	Power/Other	
AH30	VSS	Power/Other	
AH33	VSS	Power/Other	
AJ2	VSS	Power/Other	
AJ5	VSS	Power/Other	
AJ8	VSS	Power/Other	
AJ11	VSS	Power/Other	
AJ14	VSS	Power/Other	
AJ17	VSS	Power/Other	
AJ23	VSS	Power/Other	
AJ26	VSS	Power/Other	
AN1	VSS	Power/Other	
AN4	VSS	Power/Other	
AN7	VSS	Power/Other	
AN10	VSS	Power/Other	
AN13	VSS	Power/Other	
AN16	VSS	Power/Other	
AN22	VSS	Power/Other	
AN25	VSS	Power/Other	
AN28	VSS	Power/Other	
AN31	VSS	Power/Other	
AN34	VSS	Power/Other	
AN37	VSS	Power/Other	
AN38	VSS	Power/Other	
AP3	VSS	Power/Other	
AP6	VSS	Power/Other	
AP9	VSS	Power/Other	

Table 8-2. Intel 5000X Chipset MCH Signals (by Signal Name) (Sheet 17 of 19)

Ball No.	Signal Name	Buffer Type	Direction	Ball No.	Signal Name	Buffer Type	Direction
AK38	VSS	Power/Other		AP12	VSS	Power/Other	
AL3	VSS	Power/Other		AP15	VSS	Power/Other	
AL6	VSS	Power/Other		AP24	VSS	Power/Other	
AL9	VSS	Power/Other		AP27	VSS	Power/Other	
AL12	VSS	Power/Other		AP30	VSS	Power/Other	
AL15	VSS	Power/Other		AP33	VSS	Power/Other	
AL24	VSS	Power/Other		AR2	VSS	Power/Other	
AL27	VSS	Power/Other		AR5	VSS	Power/Other	
AL30	VSS	Power/Other		AR8	VSS	Power/Other	
AL33	VSS	Power/Other		AR11	VSS	Power/Other	
AM2	VSS	Power/Other		AR14	VSS	Power/Other	
AM5	VSS	Power/Other		AR17	VSS	Power/Other	
AM8	VSS	Power/Other		AR23	VSS	Power/Other	
AM11	VSS	Power/Other		AR26	VSS	Power/Other	
AM14	VSS	Power/Other		AR29	VSS	Power/Other	
AM17	VSS	Power/Other		AR32	VSS	Power/Other	
AM23	VSS	Power/Other		AR35	VSS	Power/Other	
AM26	VSS	Power/Other		AR36	VSS	Power/Other	
AM29	VSS	Power/Other		AT1	VSS	Power/Other	
AM32	VSS	Power/Other		AT4	VSS	Power/Other	
AM35	VSS	Power/Other		AT7	VSS	Power/Other	
AM36	VSS	Power/Other		AT10	VSS	Power/Other	
AT13	VSS	Power/Other		AC14	VTT	Power/Other	
AT15	VSS	Power/Other		AD13	VTT	Power/Other	
AT16	VSS	Power/Other		AD14	VTT	Power/Other	
AT22	VSS	Power/Other		AD15	VTT	Power/Other	
AT25	VSS	Power/Other		AD16	VTT	Power/Other	
AT28	VSS	Power/Other		AD17	VTT	Power/Other	
AT31	VSS	Power/Other		AD18	VTT	Power/Other	
AT34	VSS	Power/Other		AD19	VTT	Power/Other	
AT38	VSS	Power/Other		AD20	VTT	Power/Other	
AU2	VSS	Power/Other		AD21	VTT	Power/Other	
AU6	VSS	Power/Other		AD22	VTT	Power/Other	
AU9	VSS	Power/Other		AD23	VTT	Power/Other	
AU12	VSS	Power/Other		AD24	VTT	Power/Other	
AU13	VSS	Power/Other		AD25	VTT	Power/Other	
AU14	VSS	Power/Other		AE13	VTT	Power/Other	
AU15	VSS	Power/Other		AE14	VTT	Power/Other	
AU24	VSS	Power/Other		AE15	VTT	Power/Other	

Table 8-2. Intel 5000X Chipset MCH Signals (by Signal Name) (Sheet 18 of 19)

Ball No.	Signal Name	Buffer Type	Direction
AU27	VSS	Power/Other	
AU30	VSS	Power/Other	
AU33	VSS	Power/Other	
AU37	VSS	Power/Other	
AV3	VSS	Power/Other	
AV5	VSS	Power/Other	
AV8	VSS	Power/Other	
AV11	VSS	Power/Other	
AV14	VSS	Power/Other	
AV15	VSS	Power/Other	
AV16	VSS	Power/Other	
AV17	VSS	Power/Other	
AV23	VSS	Power/Other	
AV26	VSS	Power/Other	
AV29	VSS	Power/Other	
AV32	VSS	Power/Other	
AV35	VSS	Power/Other	
AV36	VSS	Power/Other	
L9	VSSQUIET	Analog	
L25	VSSSEN	Temp	
AC13	VTT	Power/Other	
AH20	VTT	Power/Other	
AH21	VTT	Power/Other	
AJ18	VTT	Power/Other	
AJ19	VTT	Power/Other	
AJ20	VTT	Power/Other	
AJ21	VTT	Power/Other	
AK18	VTT	Power/Other	
AK19	VTT	Power/Other	
AK20	VTT	Power/Other	
AK21	VTT	Power/Other	
AL18	VTT	Power/Other	
AL19	VTT	Power/Other	
AL20	VTT	Power/Other	
AL21	VTT	Power/Other	
AM18	VTT	Power/Other	
AM19	VTT	Power/Other	
AM20	VTT	Power/Other	
AM21	VTT	Power/Other	
AE16	VTT	Power/Other	
AE17	VTT	Power/Other	
AE18	VTT	Power/Other	
AE19	VTT	Power/Other	
AE20	VTT	Power/Other	
AE21	VTT	Power/Other	
AE22	VTT	Power/Other	
AE23	VTT	Power/Other	
AE24	VTT	Power/Other	
AE25	VTT	Power/Other	
AE26	VTT	Power/Other	
AF18	VTT	Power/Other	
AF19	VTT	Power/Other	
AF20	VTT	Power/Other	
AF21	VTT	Power/Other	
AG18	VTT	Power/Other	
AG19	VTT	Power/Other	
AG20	VTT	Power/Other	
AG21	VTT	Power/Other	
AH18	VTT	Power/Other	
AH19	VTT	Power/Other	
AV18	VTT	Power/Other	
AV19	VTT	Power/Other	
AV20	VTT	Power/Other	
AV21	VTT	Power/Other	
F14	XDPCOMCRES	Analog	
A11	XDPD[0]#	XDP	I/O
B10	XDPD[1]#	XDP	I/O
A14	XDPD[10]#	XDP	I/O
A16	XDPD[11]#	XDP	I/O
C16	XDPD[12]#	XDP	I/O
A15	XDPD[13]#	XDP	I/O
D16	XDPD[14]#	XDP	I/O
E16	XDPD[15]#	XDP	I/O
A10	XDPD[2]#	XDP	I/O
D13	XDPD[3]#	XDP	I/O
A12	XDPD[4]#	XDP	I/O
E14	XDPD[5]#	XDP	I/O
B13	XDPD[6]#	XDP	I/O

Table 8-2. Intel 5000X Chipset MCH Signals (by Signal Name) (Sheet 19 of 19)

Ball No.	Signal Name	Buffer Type	Direction
AN18	VTT	Power/Other	
AN19	VTT	Power/Other	
AN20	VTT	Power/Other	
AN21	VTT	Power/Other	
AP18	VTT	Power/Other	
AP19	VTT	Power/Other	
AP20	VTT	Power/Other	
AP21	VTT	Power/Other	
AR18	VTT	Power/Other	
AR19	VTT	Power/Other	
AR20	VTT	Power/Other	
AR21	VTT	Power/Other	
AT18	VTT	Power/Other	
AT19	VTT	Power/Other	
AT20	VTT	Power/Other	
AT21	VTT	Power/Other	
AU18	VTT	Power/Other	
AU19	VTT	Power/Other	
AU20	VTT	Power/Other	
AU21	VTT	Power/Other	

Ball No.	Signal Name	Buffer Type	Direction
B14	XDPD[7]#	XDP	I/O
D15	XDPD[8]#	XDP	I/O
B15	XDPD[9]#	XDP	I/O
C13	XDPDSTBN#	XDP	I/O
C14	XDPDSTBP#	XDP	I/O
G14	XDPODTCRES	Analog	
A17	XDPRDY#	XDP	I/O
J15	XDPSLWCRES	Analog	

8.2 Package Information

Figure 8-5. Bottom View

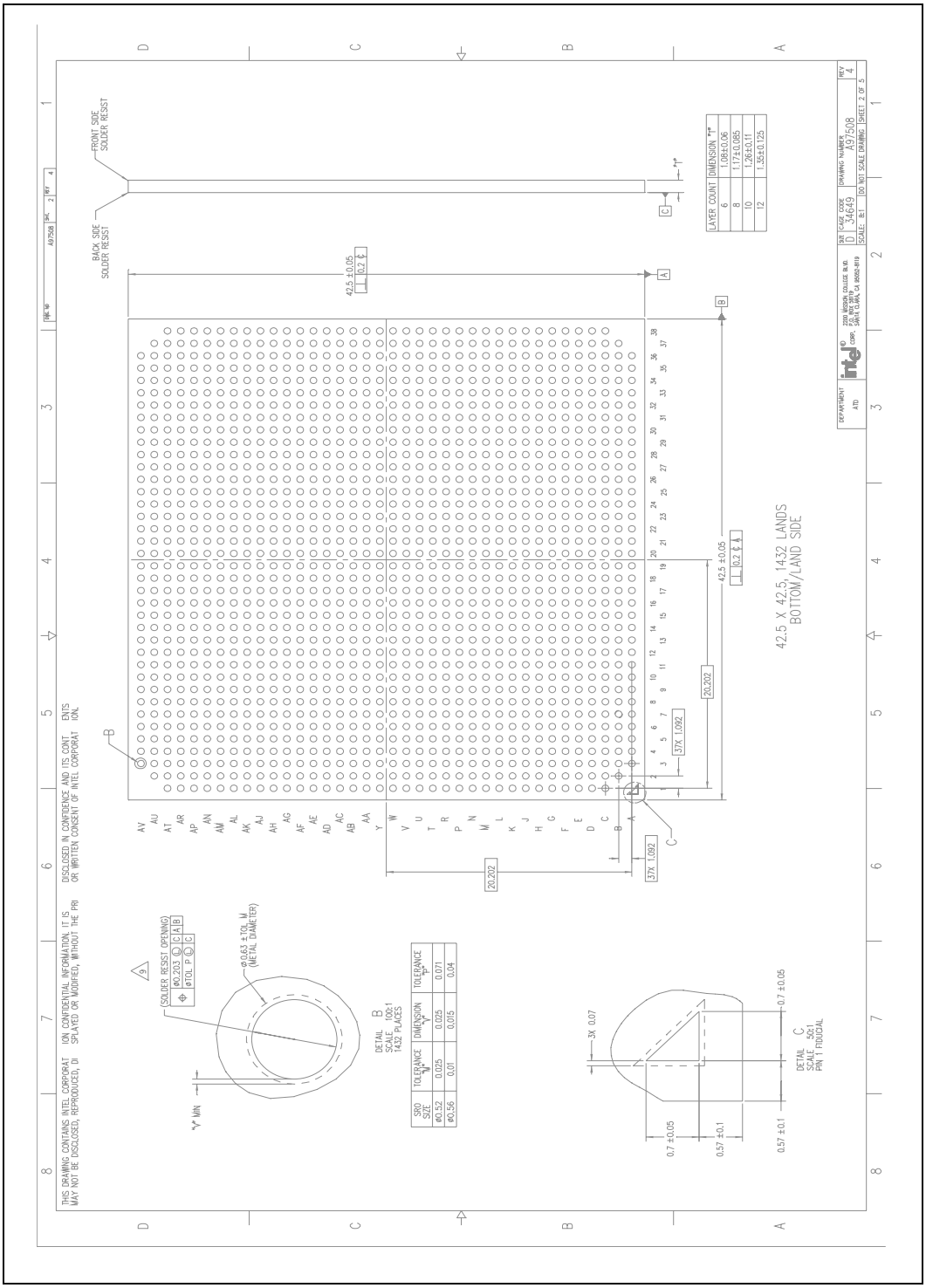


Figure 8-6. Top View

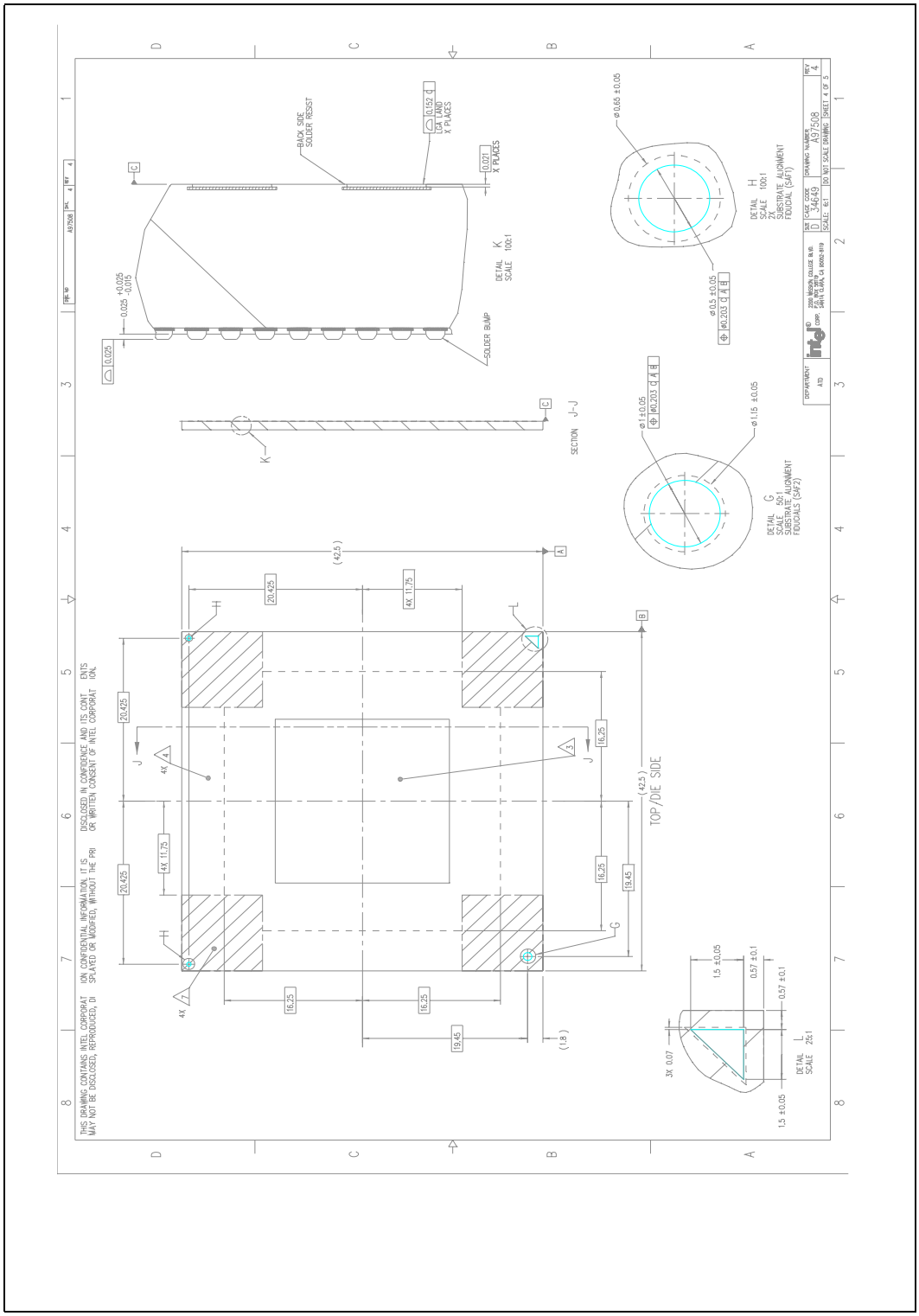


Figure 8-7. Package Stackup

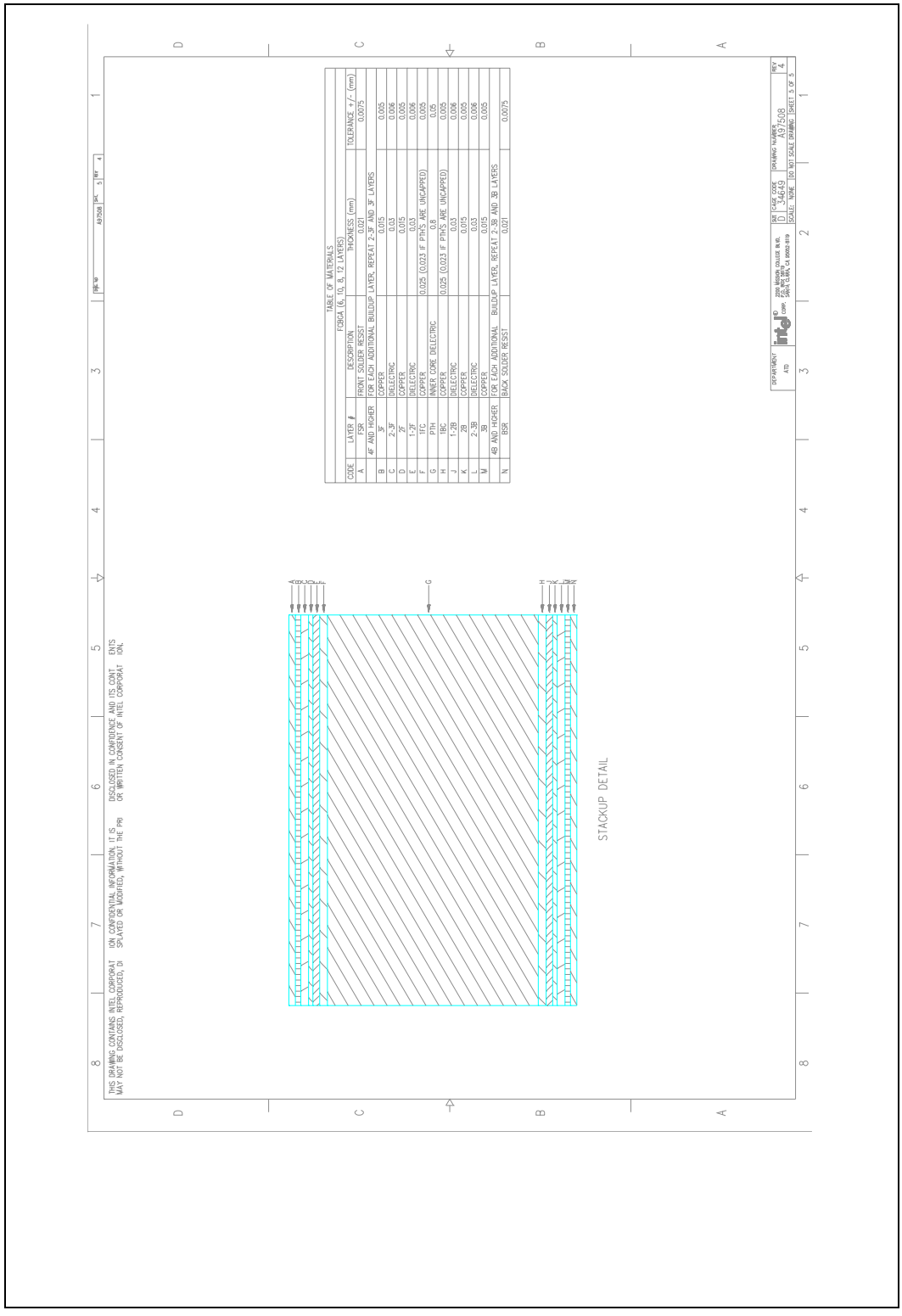
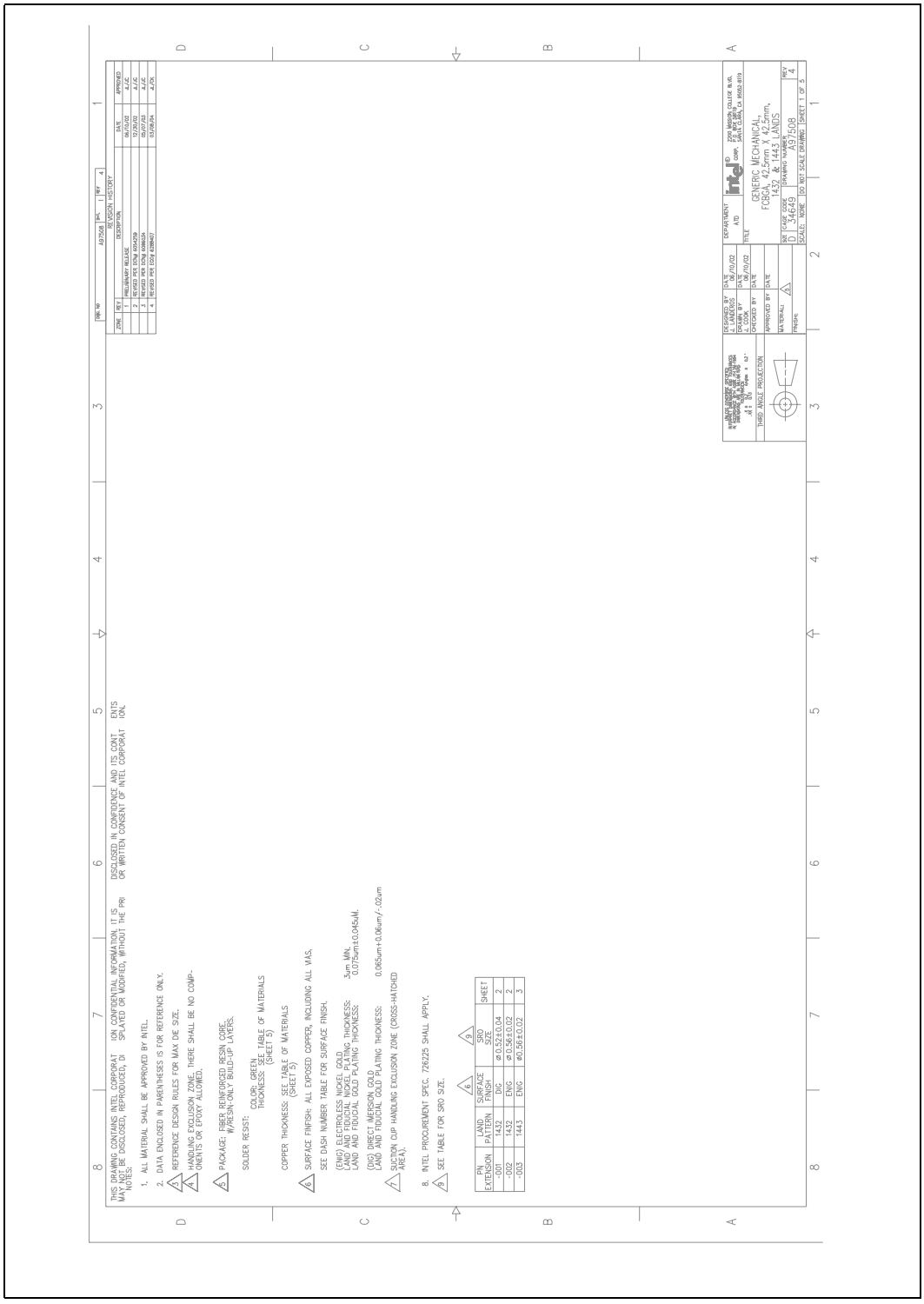


Figure 8-8. Notes



§