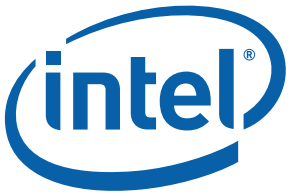


Intel® 5100 Memory Controller Hub Chipset

Datasheet

July 2009

Revision 005US

Order Number: 318378-005US

Contents

Figures

Tables

Revision History

Date	Revision	Description
July 2009	005	• Change Bit description in Section 3.8.8.33, "PEXCTRL3[7:2,0] - PCI Express* Control Register 3" • Changed DQS and DQS Lanes data in Table 67.
June 2009	004	• Removed description from bit 31:0 in Section 3.9.6.11, "REDMEMA[1:0]: Recoverable Memory Data Error Log Register A" on page 206.
July 2008	003	**Global Changes:** • Clarified that FSB{0/1}RS[2:0]# and FSB{0/1}RSP# signals are outputs only. **Register Changes:** • Section 3.9.1.4, "MCDEF3: MCDEF Register 3" corrected offset. • Section 3.9.6.12, "REDMEMB[1:0]: Recoverable Memory Data Error Log Register B" added DQS column to Table 67, "ECC Locator Mapping Information". • Section 3.12, "PCI Express* Per-Port Registers" removed all references to PCI Express* Write Combining capabilities. Also defeatured PCI Express* Interconnect-Built-In-Self-Test (IBIST) registers. **Functional Description:** • Table 117, "PCI Hot Plug* Signals on Virtual Pin Port" added.

Date	Revision	Description
January 2008	002	**Global Changes:** • Added remaining DMA Engine capabilities to document. • Removed A0 silicon descriptive text. • Corrected references to "Dual Channel mode" with proper reference of "two channels;" Dual Channel mode does not exist on the MCH. • Corrected spelling and grammar where appropriate. **Register Changes:** • Section 3.8.3.11, "EXSMRAMC - Expansion System Management RAM Control Register" corrected Function number typo from 2 to 0. • Section 3.8.3.7, "PAM6 - Programmable Attribute Map Register 6" corrected error in ESIENABLE6 address range. • Section 3.8.13.11, "EMASK_FSB[1:0]: FSB Error Mask Register" added note on parity. • Section 3.9.1.4, "MCDEF3: MCDEF Register 3" changed Register to reserved, moved Asynchronous Request for Self-refresh to White Paper under development. • Section 3.9.10.1, "DSRETC[1:0]: DRAM Self-Refresh Extended Timing and Control" updated introductory paragraph TRFC value for 32 GB mode of operation and corrected name of Register bits to better reflect their function. **Functional Description:** • Section 5.2.2, "DIMM Technology and Organization" updated DIMM Configuration descriptions and associated 32 GB mode diagram. • Section 5.2.3.1, "Permissible Configurations" added impact of mixing number of ranks between channels when added single ranks. • Section 5.2.4, "Memory RAS" corrected to indicate rank sparing not DIMM sparing. • Section 5.2.6.1, "Inbound ECC Code Layout for Memory Interface" updated title and description. • Section 5.2.8.2, "Open Loop Global Throttling" rewrote subsection. • Section 5.2.9, "Electrical Throttling" updated Table 91, "Electrical Throttle Window as Function of DIMM Technology". • Section 5.4, "XAPIC Interrupt Message Delivery" added nomenclature explanation and updated signal references in all subsections accordingly. • Section 5.8, "Software Guidance for MSI Handling" added from SU revision 001, Specification Clarification #1. • Section 5.10, "Interrupt Swizzling" added. • Section 5.13.9, "DMA Engine Implementation," Section 5.13.10, "DMA Engine Usage Model," Section 5.14, "Using DMA Engine Technology," Section 5.15, "Implementation Requirements," Section 5.16, "Programming Flow" and Section 5.17, "DMA Engine Driver" added for DMA Engine information. • Section 5.20.2, "SMBus and PCI Express* Interoperability Timeout Recommendation" added from SU revision 001, Specification Clarification #2. • Section 5.20.10, "Virtual Pin Ports" cleaned up text description. • Section 5.24, "Error List," Table 123, "Intel® 5100 Memory Controller Hub Chipset Error List" added DMA Errors to Error List table. **Others:** • Section 1.3.1, "BIOS SelfTest Utility" removed; SelfTest password is no longer required. • Section 1.3, "Intel® 5100 Memory Controller Hub Chipset Overview," added four rank per DIMM support. • Section 2.7, "Clocks, Reset and Miscellaneous" updated 48GB_Mode and ASYNCRFSH signal descriptions. • Section 6.1.2, "Power Characteristics," Section 125, "Operating Condition Power Supply Rails" updated note 8.
October 2007	001	Initial release

Revision Number Descriptions

Revision	Associated Life Cycle Milestone	Release Information
0.0	POP L3 Closure	Initial Documentation - Typically Internal Only
0.1–0.4	When Needed	Project Dependent - Typically Internal Only
0.5	Design Win Phase	First, Required Customer Release
0.6–0.7	When Needed	Project Dependent
0.7	Simulations Complete	Second, Recommended Customer Release
0.8–0.9	When Needed	Project Dependent
1.0	First Silicon Samples	Required Customer Release
1.1–1.4	When Needed	Project Dependent (Recommended)
1.5	Qualification Silicon Samples	Project Dependent
1.6–1.9	When Needed	Project Dependent
NDA - 2.0 Public - XXXXXX-001	First SKU Launch	Required Customer Release - Product Launch
2.1 and up	When Needed	Project Dependent

Note: Rows highlighted in gray are required revisions.

1.0 Introduction

The Intel® 5100 Memory Controller Hub Chipset (Intel® 5100 MCH Chipset, formerly code-named San Clemente or SC) is designed for systems based on the Dual-Core Intel® Xeon® processor 5100 series, Quad-Core Intel® Xeon® processor 5300 series, Dual-Core Intel® Xeon® processor 5200 series, Quad-Core Intel® Xeon® processor 5400 series, and Intel® Core™2 Duo Processor T9400 and supports FSB operation of 1066 MT/s and 1333 MT/s. The Intel® 5100 MCH Chipset-based platforms contain two main components: the Memory Controller Hub (MCH) for the host bridge and the I/O Controller Hub (ICH) for the I/O subsystem. The Intel® 5100 MCH Chipset-based platform uses the Intel® 82801IR I/O Controller Hub (ICH9R).

The Intel® 5100 MCH Chipset is implemented in a 0.13 µm silicon process, packaged in a 1432 pin FCBGA package with integrated heat spreader. The balls are on 1.092 mm (43 mil) centers. The overall package dimensions are 42.5 mm by 42.5 mm.

The Intel® 5100 MCH Chipset-based platform for Dual-Processor (DP) system designs supports a 771-land, FC-LGA4 (Flip Chip Land Grid Array 4) package for the Quad-Core and Dual-Core Intel® Xeon® Processor 5000 Sequence. This package uses the matching LGA771 socket. The surface mount, LGA771 socket supports Direct Socket Loading (DSL).

The Intel® 5100 MCH Chipset-based platform for Uni-Processor (UP) system designs supports 479-ball Micro-FCBGA (Flip Chip Ball Grid Array) and 478-pin Micro-FCPGA (Flip Chip Pin Grid Array) packages for the Intel® Core™2 Duo Processor T9400.

The Dual-Core Intel® Xeon® processor 5100 series and Quad-Core Intel® Xeon® processor 5300 series return a processor signature of 06Fxh, and the Dual-Core Intel® Xeon® processor 5200 series, Quad-Core Intel® Xeon® processor 5400 series and Intel® Core™2 Duo Processor T9400 return a processor signature of 1067xh, when the CPUID instruction is executed with EAX=1. The x represents the stepping number. Refer to the appropriate processor Specification Update for more information on processor signature.

Note: Unless otherwise specified, the term "processor" in this document refers to the Dual-Core Intel® Xeon® processor 5100 series, Quad-Core Intel® Xeon® processor 5300 series, Dual-Core Intel® Xeon® processor 5200 series, Quad-Core Intel® Xeon® processor 5400 series, and Intel® Core™2 Duo Processor T9400. Unless otherwise specified, the term "MCH" in this document refers to the Intel® 5100 MCH Chipset.

1.1 Terminology

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

Table 1. 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.
AGTL	Assisted Gunning Transceiver Logic
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.
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 (i.e., 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.
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® 5100 MCH Chipset that acts as a temporary repository for system data in flight between the various ports: FSBs, DIMMs, ESI, and PCI Express*.
Cfg, Config	Abbreviation for "Configuration".
Channel	In the MCH a DIMM DRAM channel is the set of signals that connects to one set of DIMMs. The MCH has up to two DRAM channels.
Character	The raw data byte in an encoded system (e.g., 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 an interface, or within a component. A Completion will consistently refer to a preceding request and may or may not include data and/or other information.
Core	The internal base logic in the MCH.
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 a deferred phase.

Table 1. Terminology (Sheet 2 of 6)

Terminology	Description
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
DID	Device Identification. Provides PCI device identification number.
DIMM	Dual-in-Line Memory Module. A packaging arrangement of memory devices on a socketable substrate.
DIMM Rank	That set of SDRAMs on one branch which provides the data packet
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 (i.e., the least significant two bits of the address are 00b).
ECC	Error Correcting Code
ESI	Enterprise South Bridge Interface. The interface connecting the MCH to the ICH9R.
Flit	The exchange of the unit of information at the link layer that synchronizes each data transfer, i.e., flow control with the use of credits. Flits are usually several phits long.
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.
Gb/s	Gigabits per second (10^9 bits per second).
GB/s	Gigabytes per second (10^9 bytes per second).
Half Duplex	A connection or channel that allows data or messages to be transmitted in either direction, but not simultaneously.
Hardwired	A parameter that has a fixed value.
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. (e.g., I/O read)
ICH9R	Ninth generation I/O controller hub with RAID, the Intel® 82801IR I/O Controller Hub. The I/O controller hub component that contains the legacy I/O functions. It communicates with the MCH over a proprietary interconnect called the ESI interface.
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."
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 I/O (SDRAM, SMBus).
Incoming	A transaction or data that enters the MCH.
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.

Table 1. Terminology (Sheet 3 of 6)

Terminology	Description
Intel® 5100 MCH Chipset	Intel® 5100 Memory Controller Hub Chipset (formerly code-named San Clemente)
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.
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)
MC	Memory Controller
MCH	The Memory Controller Hub component that contains the processor interface, DRAM controller, PCI Express* interface. It communicates with the I/O controller hub (ICH9R) 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. This terminology is utilized in this document.
MMCFG	Memory Mapped Configuration. A memory transaction that accesses configuration space.
MMIO	Memory Mapped I/O. Any memory access to PCI Express*.
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 MCH. 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.

Table 1. Terminology (Sheet 4 of 6)

Terminology	Description
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*, Rev. 2.3
Peer-to-peer	Transactions that occur between two devices below the PCI Express* or ESI ports.
Phit	The smallest physical unit of information at the physical layer, which is transferred across the width of one physical link in one cycle.
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.
POC	Power-On-Configuration settings determined at power-up by strapping pins. Additionally, the POC Register settings are driven on the address pins when RESET# is issued to the processor.
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 ICH9R 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.
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).

Table 1. Terminology (Sheet 5 of 6)

Terminology	Description
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 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**.
Rank	DDR2 DIMMs are divided into ranks, with each rank consists of a group of chips whose data lines equal 64 total bits wide.
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).
SC	Intel® 5100 Memory Controller Hub Chipset (formerly code-named San Clemente)
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. 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 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 SDRAM's via the SMBus protocol
Simplex	A connection or channel that allows data or messages to be transmitted in one direction only.
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.
SMBus	System Management Bus. Mastered by a system management controller to read and write configuration registers. Signaling and protocol are loosely based on the I^2C* Interface, 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® 5100 MCH Chipset 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_18	Stub Series Terminated Logic for 1.8 V (DDR2)
SSTL_2	Stub Series Terminated Logic for 2.6 V (DDR)
Sticky Bits	Register bits, whose value remains unchanged when system is RESET
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® 5100 MCH Chipset interface. The system bus in this document refers to operation at 266/533/1066, 333/667/1333 (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 and Materials

Intel® Electronic Design Kits (EDKs) provide online, real-time collateral updates. The following links take you to the EDK server and require you to log into Intel® Business Link (IBL).

- Quad-Core and Dual-Core Intel® Xeon® Processor 5000 Sequence with Intel® 5100 Memory Controller Hub Chipset for Communications, Embedded, and Storage Applications
- Intel® Core™2 Duo Processors T9400 and SL9400 and Intel® 5100 Memory Controller Hub Chipset for Communications and Embedded Applications

Table 2. Related Documents (Sheet 1 of 2)

Document	Document Number/Location
82801Ix I/O Controller Hub (ICH9) – External Design Specification (EDS)	Note 1
82801Ix I/O Controller Hub (ICH9) – Specification Update - NDA	Note 1
Debug Port Design Guide for Intel® 5000 Series Chipset Based Platforms (External Version)	Note 1
Dual-Core Intel® Xeon® Processor 5100 Series Datasheet	http://www.intel.com/ (313355)
Dual-Core Intel® Xeon® Processor 5100 Series Specification Update	http://www.intel.com/ (313356)
Dual-Core Intel® Xeon® Processor 5100 Series Thermal/Mechanical Design Guidelines	http://www.intel.com/ (313357)
Dual-Core Intel® Xeon® Processor 5200 Series Datasheet	http://www.intel.com/ (318590)
Dual-Core Intel® Xeon® Processor 5200 Series Specification Update	http://www.intel.com/ (318586)
Dual-Core Intel® Xeon® Processor 5200 Series Thermal/Mechanical Design Guidelines	http://www.intel.com/ (318675)
Dual-Core Intel® Xeon® Processor LV 5138 in Embedded Applications Thermal/Mechanical Design Guidelines	http://www.intel.com/ (315225)
Intel® 5000 Series Chipset MCH CRB Bridgeport/Bridgeport 2/ Hoodsport/Hoodsport 2 CPLD Code&Schematics (for PCI Hot Plug* reference material, Intel® 5000 Series MCH Chipset PDG password required)	Note 1
Intel® 5100 Memory Controller Hub Chipset (embedded) – External Design Specification (EDS) Addendum	Note 1
Intel® 5100 Memory Controller Hub Chipset B0 Stepping (embedded) – Electronic Pin List and Ballmap	Note 1
Intel® 5100 Memory Controller Hub Chipset for Communications, Embedded, and Storage Applications Thermal/Mechanical Design Guide	http://www.intel.com/ (318676)
Intel® 5100 Memory Controller Hub Chipset Material Declaration Data Sheets (MDDS) for product RoHS compliance information. Search on the web site with the Intel® 5100 MCH Chipset Product Code available from your local Intel sales representative.	http://intelmdds.pcnalert.com/MDDS/MDDSView.aspx
Intel® 5100 Memory Controller Hub Chipset Specification Update	http://www.intel.com/ (318385)
Intel® 64 and IA-32 Architectures Optimization Reference Manual	http://www.intel.com/ (248966)
Intel® 64 and IA-32 Architectures Software Developer's Manual Documentation Changes	http://www.intel.com/ (252046)
Intel® 64 and IA-32 Architectures Software Developer's Manual, Volumes 1–3	http://www.intel.com/ (253665, 253666, 253667, 253668, 253669)
Intel® Core™2 Duo Processor, Intel® Core™2 Solo Processor and Intel® Core™2 Extreme Processor on 45-nm Process Datasheet	http://www.intel.com/ (320120)

Notes:
1. Contact your Intel sales representative. Some documents may not be available at this time.

Table 2. Related Documents (Sheet 2 of 2)

Document	Document Number/Location
Intel® Core™2 Duo Processor, Intel® Core™2 Solo Processor and Intel® Core™2 Extreme Processor on 45-nm Process Specification Update	http://www.intel.com/ (320121)
Intel® Core™2 Duo Processors on 45-nm process for Embedded Applications Thermal Design Guide	http://www.intel.com/ (320028)
Intel® Core™2 Duo Processors T9400 and SL9400 and Intel® 5100 Memory Controller Hub Chipset for Communications and Embedded Applications – Platform Design Guide	Note 1
Intel® Core™2 Duo Processor T9400 and Intel® 5100 Memory Controller Hub Chipset – Platform Design Guide	Note 1
PCI Express* Base Specification, Rev. 1.0a	http://www.pcisig.com/specifications/pciexpress/
PCI Local Bus Specification, Rev. 2.3	http://www.pcisig.com/specifications/conventional/
Quad-Core and Dual-Core Intel® Xeon® Processor 5000 Sequence with Intel® 5100 Memory Controller Hub Chipset for Communications, Embedded, and Storage Applications – Platform Design Guide	Note 1
Quad-Core Intel® Xeon® Processor 5300 Series Datasheet	http://www.intel.com/ (315569)
Quad-Core Intel® Xeon® Processor 5300 Series Specification Update	http://www.intel.com/ (315338)
Quad-Core Intel® Xeon® Processor 5300 Series Thermal/Mechanical Design Guidelines	http://www.intel.com/ (315794)
Quad-Core Intel® Xeon® Processor 5400 Series Datasheet	http://www.intel.com/ (318589)
Quad-Core Intel® Xeon® Processor 5400 Series Specification Update	http://www.intel.com/ (318585)
Quad-Core Intel® Xeon® Processor 5400 Series Thermal/Mechanical Design Guidelines	http://www.intel.com/ (318611)
Quad-Core Intel® Xeon® Processor L5318 in Embedded Applications Thermal and Mechanical Design Guidelines	http://www.intel.com/ (318474)
RS - Intel® 5100 Memory Controller Hub Chipset BIOS Specification	Note 1
RS - Intel® Core™ Microarchitecture, Intel® Pentium® 4, and Intel® Xeon® Processor External HW Spec	Note 1
System Management Bus (SMBus) Specification, Version 2.0	http://www.smbus.org/specs/

Notes:
1. Contact your Intel sales representative. Some documents may not be available at this time.

1.3 Intel® 5100 Memory Controller Hub Chipset Overview

In a Intel® 5100 MCH Chipset-based platform, the MCH provides two FSB processor interfaces, two DDR2 memory channels, six x4 PCI Express* bus interfaces configurable to form x4, x8 or x16 ports, an Enterprise South Bridge Interface (ESI), and three SMBus interfaces for system management, PCI Hot Plug* control and DIMM Serial Presence Detect (SPD). The Intel® 5100 Memory Controller Hub Chipset device has DMA Engine (Crystal Beach [CB]) capabilities and supports Intel® I/O Acceleration Technology (Intel® I/OAT). Figure 1, "Intel® 5100 Memory Controller Hub Chipset-based System Block Diagram" shows a block diagram of a Intel® 5100 MCH Chipset-based platform.

This document contains product information and/or design guidelines specific to component manufacturing steppings. For more information on correlating product marking information to manufacturing stepping, please refer to the *Intel® 5100 Memory Controller Hub Chipset Specification Update*.

The Intel® 5100 MCH Chipset is designed for server, communications, storage, and embedded application systems based on the Dual-Core Intel® Xeon® processor 5100 series, Quad-Core Intel® Xeon® processor 5300 series, Dual-Core Intel® Xeon® processor 5200 series and Quad-Core Intel® Xeon® processor 5400 series and Intel® Core™2 Duo Processor T9400 for UP designs. The Intel® 5100 MCH Chipset supports two processors on dual independent point to point system buses operating at 266 MHz (1066 MT/s) or 333 MHz (1333 MT/s); the peak bandwidth of the two processor busses is respectively, 17 GB/s and 21 GB/s for outbound and 8 GB/s and 10 GB/s for inbound accesses. The MCH supports 36-bit addressability with a total of 32 GB or 48 GB depending upon mode.

The Quad-Core Intel® Xeon® processor 5300 series (65 nm process) and Quad-Core Intel® Xeon® processor 5400 series (45 nm process) have 2x4 MB and 2x6 MB shared L2 cache and a 266 MHz (1066 MT/s) system bus. The Dual-Core Intel® Xeon® processor 5100 series (65 nm process) and Dual-Core Intel® Xeon® processor 5200 series (45 nm process) have 4 MB and 6 MB shared L2 cache, respectively, and a 333 MHz (1333 MT/s) system bus. The Intel® Core™2 Duo Processor T9400 (45 nm process) has 6 MB shared L2 cache and a 266 MHz (1066 MT/s) system bus.

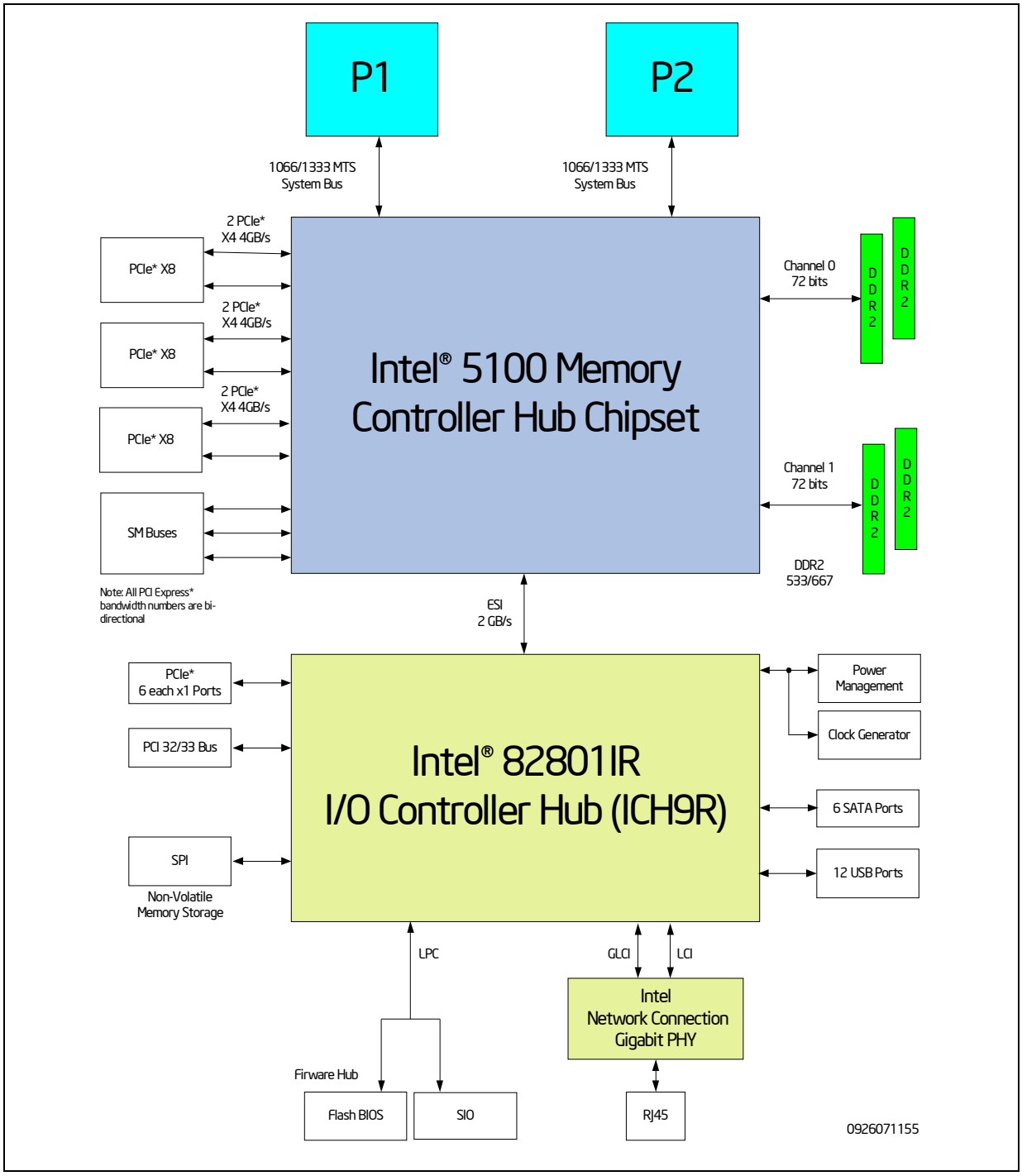
The MCH provides two channels of registered ECC memory DIMMs. In 32 GB Mode, each channel can support up to four ranks of DDR2 memory for a maximum physical memory configuration of 32 GB. The four ranks are recommended to be configured across two to three DIMMs. In 48 GB mode, each channel can support up to six ranks of DDR2 memory for a maximum physical memory configuration of 48 GB. The six ranks per channel are required to be configured across up to three DIMMs with up to four ranks per DIMM. For currently supported DIMM configurations, see the *Quad-Core and Dual-Core Intel® Xeon® Processor 5000 Sequence with Intel® 5100 Memory Controller Hub Chipset for Communications, Embedded, and Storage Applications – Platform Design Guide* or *Intel® Core™2 Duo Processors T9400 and SL9400 and Intel® 5100 Memory Controller Hub Chipset for Communications and Embedded Applications – Platform Design Guide*. The read bandwidth for each channel is 5.3 GB/s for DDR2 667 memory and 4.25 GB/s for DDR2 533. This provides a maximum bandwidth of 10.6 GB/s for two DDR2 channels.

The MCH offers six PCI Express* x4 ports compliant to *PCI Express* Base Specification*, Rev. 1.0a. Configured appropriately, the ports can be combined to form x4, x8 or x16 ports.

The ICH9R is an I/O controller hub supporting various I/O interfaces including six PCI Express* x1 ports, Serial ATA host controllers supporting up to six SATA ports at a speed of 3 Gb/s, twelve external USB 2.0 ports with port disable capability, a System Management Bus interface (SMBus), a Serial Peripheral Interface (SPI) and a Low Pincount bus (LPC) for BIOS and firmware storage. Additionally, the ICH9R contains a PCI Controller supporting up to four Bus Masters (*PCI Local Bus Specification*, Rev. 2.3 compliant), an integrated 10/100/1000 LAN controller and a dedicated ESI port for communication with the MCH. The ICH9R component provides the data buffering and arbitration required to ensure that system interfaces operate efficiently and provide the bandwidth necessary to enable the system to obtain peak performance.

The ICH9R component is ACPI compliant and 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 ICH9R also supports Alert Standard Format for remote management.

Figure 1. Intel® 5100 Memory Controller Hub Chipset-based System Block Diagram



2.0 Signal Description

This section provides a detailed description of MCH signals. The signals are arranged in functional groups according to their associated interface. Figure 2, "Intel® 5100 Memory Controller Hub Chipset Signal Diagram 32 GB Mode" and Figure 3, "Intel® 5100 Memory Controller Hub Chipset Signal Diagram 48 GB Mode" illustrate the signals in a block diagram format for 32 GB and 48 GB modes. 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, e.g., "{X/Y}", represent replications of major buses. Square-bracketed numerical indices, e.g., "[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, e.g., "{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 Tristate. This pin is driven to its inactive state prior to tristating.

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

AGTL+	Open Drain AGTL+ interface signal. The MCH integrates AGTL+ termination resistors, and supports VTT from 1.05 V to 1.2 V.
LVTTL	Low Voltage TTL 3.3 V compatible signals

SSTL_2 Stub Series Terminated Logic 2.6 V compatible 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 3. 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{P/N}	Expands to: RRP, RRN. This denotes inverted electrical duplicates.
RR# or RR[2:0]#	Denotes an active low signal or bus.

Table 4, "Buffer Signal Types" lists the reference terminology used for signal types.

Table 4. 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

Table 5. Processor Front Side Bus 0 Signals (Sheet 1 of 5)

<table>
<tr><th>Signal Name</th><th>Type</th><th>Description</th></tr>
<tr><td>FSB0A[35:3]#</td><td>I/O</td><td>Processor 0 Address Bus:
FSB0A[35:3]# define a 2^{36}-byte physical memory address space. In sub-phase 1 of the address phase, these signals transmit the address of a transaction. In sub-phase 2, these signals transmit transaction type information. FSB0A[35:3]# are protected by parity signals FSB0AP[1:0]#. FSB0A[35:3]# are source synchronous signals and are latched into the receiving buffers by FSB0ADSTB[1:0]#.
On the active-to-inactive transition of RESET#, the processors sample a subset of the FSB0A[35:3]# lands to determine their power-on configuration.
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 Secondary PCI initiators. FSB0A[35:3]# are transferred at 2x rate. Note that the address is inverted on the processor bus.</td></tr>
<tr><td>FSB0ADS#</td><td>I/O</td><td>Processor 0 Address Strobe:
FSB0ADS# is asserted to indicate the validity of the transaction address on FSB0A[35:3]#. All bus agents observe the FSB0ADS# activation to begin parity checking, protocol checking, address decode, internal snoop, or deferred reply ID match operations associated with the new transaction. The processor bus owner asserts FSB0ADS# to indicate the first of two cycles of a request phase.</td></tr>
<tr><td>FSB0ADSTB[1:0]#</td><td>I/O</td><td>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 on the strobes rising and falling edges.
<table>
<tr><th>Signals</th><th>Associated Strobes</th></tr>
<tr><td>FSB0REQ[4:0],
FSB0A[16:3]#</td><td>FSB0ADSTB[0]#</td></tr>
<tr><td>FSB0A[35:17]#</td><td>FSB0ADSTB[1]#</td></tr>
</table></td></tr>
<tr><td>FSB0AP[1:0]#</td><td>I/O</td><td>Processor 0 Address Parity:
FSB0AP[1:0]# provide parity protection on the address bus. FSB0AP[1:0]# are driven by the request initiator along with FSB0ADS#, FSB0A[35:3]#, and the transaction type on the FSB0REQ[4:0]# signals. A correct parity signal is high if an even number of covered signals are low and low if an odd number of covered signals are low. This allows parity to be high when all the covered signals are high. The following defines the coverage model of these signals.
<table>
<tr><th>Request Signals</th><th>Subphase 1</th><th>Subphase 2</th></tr>
<tr><td>FSB0A[35:24]#</td><td>FSB0AP[0]#</td><td>FSB0AP[1]#</td></tr>
<tr><td>FSB0A[23:3]#</td><td>FSB0AP[1]#</td><td>FSB0AP[0]#</td></tr>
<tr><td>FSB0REQ[4:0]#</td><td>FSB0AP[1]#</td><td>FSB0AP[0]#</td></tr>
</table></td></tr>
</table>

Table 5. Processor Front Side Bus 0 Signals (Sheet 2 of 5)

Signal Name	Type	Description
FSB0BINIT#	I/O	**Processor 0 Bus Initialization:** FSB0BINIT# may be observed and driven by all processor FSB agents. If the FSB0BINIT# driver is enabled during power on configuration, FSB0BINIT# is asserted to signal any bus condition that prevents reliable future operation. If FSB0BINIT# observation is enabled during power-on configuration and FSB0BINIT# is sampled asserted, symmetric agents reset their bus LOCK# activity and bus request arbitration state machines. The bus agents do not reset their I/O Queue (IOQ) and transaction tracking state machines upon observation of FSB0BINIT# assertion. Once the FSB0BINIT# assertion has been observed, the bus agents will re-arbitrate for the FSB and attempt completion of their bus queue and IOQ entries. If FSB0BINIT# observation is disabled during power-on configuration, a priority agent may handle an assertion of FSB0BINIT# as appropriate to the error handling architecture of the system.
FSB0BNR#	I/O	**Processor 0 Block Next Request:** FSB0BNR# is used to assert a bus stall by any bus agent who is unable to accept new bus transactions. During a bus stall, the current bus owner cannot issue any new transactions. Since multiple agents might need to request a bus stall at the same time, FSB0BNR# is a wired-OR signal which must connect the appropriate pins of all processor FSB agents. In order to avoid wired-OR glitches associated with simultaneous edge transitions driven by multiple drivers, FSB0BNR# is activated on specific clock edges and sampled on specific clock edges. FSB0BNR# 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	**Processor 0 Breakpoint Monitor/Debug Bus:** FSB0BPM[5:0]# are breakpoint and performance monitor signals. They are outputs from the processor which indicate the status of breakpoints and programmable counters used for monitoring processor performance. FSB0BPM[5:0]# should connect the appropriate pins of all FSB agents. FSB0BPM[4]# provides PRDY# (Probe Ready) functionality for the TAP port. PRDY# is a processor output used by debug tools to determine processor debug readiness. FSB0BPM[5]# provides PREQ# (Probe Request) functionality for the TAP port. PREQ# is used by debug tools to request debug operation of the processors. FSB0BPM[5:4]# must be bussed to all bus agents. Please refer to the appropriate platform design guidelines for more detailed information.
FSB0BPRI#	O	**Processor 0 Priority Agent Bus Request:** FSB0BPRI# is used to arbitrate for ownership of the processor FSB. It must connect the appropriate pins of all processor FSB agents. Observing FSB0BPRI# active (as asserted by the priority agent) causes all other agents to stop issuing new requests, unless such requests are part of an ongoing locked operation. The priority agent keeps FSB0BPRI# asserted until all of its requests are completed, then releases the bus by deasserting FSB0BPRI#. 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 FSB0BREQ[0]# signal low during RESET#. The signal is sampled by the processor on the active-to-inactive transition of FSB0RESET#.

Table 5. Processor Front Side Bus 0 Signals (Sheet 3 of 5)

<table>
<tr><th>Signal Name</th><th>Type</th><th>Description</th></tr>
<tr><td>FSB0D[63:0]#</td><td>I/O</td><td>Processor 0 Data Bus:
FSB0D[63:0]# are the data signals. These signals provide a 64-bit data path between the processor FSB agents. The data driver asserts FSB0DRDY# to indicate a valid data transfer.
FSB0D[63:0]# are quad-pumped signals, and will thus be driven four times in a common clock period. FSB0D[63:0]# are latched off the falling edge of both FSB0DSTBP[3:0]# and FSB0DSTBN[3:0]#. Each group of 16 data signals correspond to one data strobe pair of FSB0DSTBP[3:0]# and FSB0DSTBN[3:0]#. The below table shows the grouping of data signals to strobes and FSB0DBI[3:0]#.
Furthermore, the FSB0DBI[3:0]# signals determine the polarity of the data signals. Each group of 16 data signals corresponds to one of the FSB0DBI[3:0]# signals. When a specific FSB0DBI[3:0]# signal is active, the corresponding data group is inverted and therefore sampled active high.
<table>
<tr><th>Data Group</th><th>Data Strobe</th><th>Bus Inversion Signal</th></tr>
<tr><td>FSB0D[15:0]#</td><td>FSB0DSTB{P/N}[0]#</td><td>FSB0DBI[0]#</td></tr>
<tr><td>FSB0D[31:16]#</td><td>FSB0DSTB{P/N}[1]#</td><td>FSB0DBI[1]#</td></tr>
<tr><td>FSB0D[47:32]#</td><td>FSB0DSTB{P/N}[2]#</td><td>FSB0DBI[2]#</td></tr>
<tr><td>FSB0D[63:48]#</td><td>FSB0DSTB{P/N}[3]#</td><td>FSB0DBI[3]#</td></tr>
</table></td></tr>
<tr><td>FSB0DBI[3:0]#</td><td>I/O</td><td>Processor 0 Dynamic Data Bus Inversion:
FSB0DBI[3:0]# are source synchronous and indicate the polarity of the FSB0D[63:0]# signals. The FSB0DBI[3:0]# signals are activated when the data on the data bus is inverted. If more than half the data bits, within, within a 16-bit group, would have been asserted electronically low, the bus agent may invert the data bus signals for that particular sub-phase for that 16-bit group. The below table shows the signal relationships.
<table>
<tr><th>Bus Inversion Signal</th><th>Data Group</th></tr>
<tr><td>FSB0DBI[0]#</td><td>FSB0D[15:0]#</td></tr>
<tr><td>FSB0DBI[1]#</td><td>FSB0D[31:16]#</td></tr>
<tr><td>FSB0DBI[2]#</td><td>FSB0D[47:32]#</td></tr>
<tr><td>FSB0DBI[3]#</td><td>FSB0D[63:48]#</td></tr>
</table></td></tr>
<tr><td>FSB0DBSY#</td><td>I/O</td><td>Processor 0 Data Bus Busy:
FSB0DBSY# is asserted by the agent responsible for driving data on the processor FSB to indicate that the data bus is in use. The data bus is released after FSB0DBSY# is deasserted. This signal is used by the data bus owner to hold the data bus for transfers requiring more than one cycle.</td></tr>
<tr><td>FSB0DEFER#</td><td>O</td><td>Processor 0 Data Bus Defer:
FSB0DEFER# is asserted by an agent to indicate that a transaction cannot be guaranteed in-order completion. Defer indicates that the MCH will terminate the transaction currently being snooped with either a deferred response or with a retry response.</td></tr>
<tr><td>FSB0DP[3:0]#</td><td>I/O</td><td>Processor 0 Data Bus Parity:
FSB0DP[3:0]# provide parity protection for the FSB0D[63:0]# signals. They are driven by the agent responsible for driving the FSB0D[63:0]# signals.</td></tr>
<tr><td>FSB0DRDY#</td><td>I/O</td><td>Processor 0 Data Ready:
FSB0DRDY# is asserted by the data driver on each data transfer, indicating valid data on the data bus. FSB0DRDY# may be deasserted to insert idle clocks. FSB0DRDY# is asserted for each cycle that data is transferred.</td></tr>
</table>

Table 5. Processor Front Side Bus 0 Signals (Sheet 4 of 5)

<table>
<tr><th>Signal Name</th><th>Type</th><th>Description</th></tr>
<tr><td>FSB0DSTBP[3:0]#
FSB0DSTBN[3:0]#</td><td>I/O</td><td>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.
<table><tr><th>Data Group</th><th>Data Strobe</th><th>Bus Inversion Signal</th></tr><tr><td>FSB0D[15:0]#</td><td>FSB0DSTB{P/N}[0]#</td><td>FSB0DBI[0]#</td></tr><tr><td>FSB0D[31:16]#</td><td>FSB0DSTB{P/N}[1]#</td><td>FSB0DBI[1]#</td></tr><tr><td>FSB0D[47:32]#</td><td>FSB0DSTB{P/N}[2]#</td><td>FSB0DBI[2]#</td></tr><tr><td>FSB0D[63:48]#</td><td>FSB0DSTB{P/N}[3]#</td><td>FSB0DBI[3]#</td></tr></table></td></tr>
<tr><td>FSB0HIT#</td><td>I/O</td><td>Processor 0 Cache Hit:
FSB0HIT# along with FSB0HITM# convey transaction snoop operation results. Any FSB agent may assert both FSB0HIT# and FSB0HITM# together to indicate that it requires a snoop stall to extend the snoop window. The stall an be continued by reasserting FSB0HIT# and FSB0HITM# together. The FSB0HIT# signal indicates that a caching agent holds an unmodified version of the requested line.</td></tr>
<tr><td>FSB0HITM#</td><td>I/O</td><td>Processor 0 Cache Hit Modified:
FSB0HITM# along with FSB0HIT# convey transaction snoop operation results. Any FSB agent may assert both FSB0HITM# and FSB0HIT# together to indicate that it requires a snoop stall to extend the snoop window. The stall an be continued by reasserting FSB0HITM# and FSB0HIT# together. The FSB0HITM# signal indicates that a caching agent holds a modified version of the requested line.</td></tr>
<tr><td>FSB0LOCK#</td><td>I</td><td>Processor 0 Lock:
FSB0LOCK# indicates to the system that a transaction must occur atomically. For a locked sequence of transactions, FSB0LOCK# is asserted from the beginning of the first transaction to the end of the last transaction.
When the priority agent asserts FSB0BPRI# to arbitrate for ownership of the processor FSB, it will wait until it observes FSB0LOCK# deasserted. This enables symmetric agents to retain ownership of the processor FSB throughout the bus locked operation and ensure the atomicity of lock.</td></tr>
<tr><td>FSB0MCERR#</td><td>I/O</td><td>Processor 0 Machine Check Error:
FSB0MCERR# is asserted to indicate an unrecoverable error without a bus protocol violation. It may be driven by all processor FSB agents.
FSB0MCERR# assertion conditions are configurable at a system level. For more details regarding machine check architecture, refer to the Intel® 64 and IA-32 Architectures Software Developer's Manual, Volume 3: System Programming Guide.</td></tr>
<tr><td>FSB0REQ[4:0]#</td><td>I/O</td><td>Processor 0 Bus Request Command:
FSB0REQ[4:0]# define the attributes of the request. FSB0REQ[4:0]# are transferred at 2x rate and are source synchronous to FSB0ADSTB[1:0]#. 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. Refer to the FSB0AP[1:0]# signal description for details on parity checking of these signals.</td></tr>
<tr><td>FSB0RESET#</td><td>O</td><td>Processor 0 Reset:
FSB0RESET# is an output from the MCH. The MCH asserts FSB0RESET# while RESETI# (PLTRST# from ICH9R) is asserted and for approximately 1 ms after RESETI# is deasserted. The FSB0RESET# allows the processors to begin execution in a known state and invalidates their internal caches without writing back any of their contents.</td></tr>
</table>

Table 5. Processor Front Side Bus 0 Signals (Sheet 5 of 5)

Signal Name	Type	Description
FSB0RS[2:0]#	O	**Processor 0 Response Status Signals:** FSB0RS[2:0]# (Response Status) are driven by the response agent (the agent responsible for completion of the current transaction). FSB0RS[2:0]# 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:** FSB0RSP# provides parity protection for the FSB0RS[2:0] signals. Driven during assertion of FSB0RS[2:0]# as required for the agent responsible for completion of the current transaction. A correct parity signal is high if an even number of covered signals are low and low if an odd number of covered signals are low. When FSB0RS[2:0]# = 000 or idle is indicated, FSB0RSP# is high.
FSB0TRDY#	O	**Processor Bus 0 Target Ready:** FSB0TRDY# (Target Ready) is asserted by the target to indicate that it is ready to receive a write or implicit writeback data transfer, the target of the processor transaction is able to enter the data transfer phase.
FSB0VREF	Analog	**Processor 0 Voltage Reference:** FSB0VREF is Processor 0 voltage reference. Refer to the *Quad-Core and Dual-Core Intel® Xeon® Processor 5000 Sequence with Intel® 5100 Memory Controller Hub Chipset for Communications, Embedded, and Storage Applications – Platform Design Guide* or *Intel® Core™2 Duo Processors T9400 and SL9400 and Intel® 5100 Memory Controller Hub Chipset for Communications and Embedded Applications – Platform Design Guide* for the voltage value.

2.1.2 Processor Front Side Bus 1

Table 6. Processor Front Side Bus 1 Signals (Sheet 1 of 4)

<table>
<tr><th>Signal Name</th><th>Type</th><th>Description</th></tr>
<tr><td>FSB1A[35:3]#</td><td>I/O</td><td>Processor 1 Address Bus:
FSB1A[35:3]# define a 2^{36}-byte physical memory address space. In sub-phase 1 of the address phase, these signals transmit the address of a transaction. In sub-phase 2, these signals transmit transaction type information. FSB1A[35:3]# are protected by parity signals FSB1AP[1:0]#. FSB1A[35:3]# are source synchronous signals and are latched into the receiving buffers by FSB1ADSTB[1:0]#.
On the active-to-inactive transition of RESET#, the processors sample a subset of the FSB1A[35:3]# lands to determine their power-on configuration.
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 Secondary PCI initiators. FSB1A[35:3]# are transferred at 2x rate. Note that the address is inverted on the processor bus.</td></tr>
<tr><td>FSB1ADS#</td><td>I/O</td><td>Processor 1 Address Strobe:
FSB1ADS# is asserted to indicate the validity of the transaction address on FSB1A[35:3]#. All bus agents observe the FSB1ADS# activation to begin parity checking, protocol checking, address decode, internal snoop, or deferred reply ID match operations associated with the new transaction. The processor bus owner asserts FSB1ADS# to indicate the first of two cycles of a request phase.</td></tr>
<tr><td>FSB1ADSTB[1:0]#</td><td>I/O</td><td>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 on the strobes rising and falling edges.
<table>
<tr><th>Signals</th><th>Associated Strobes</th></tr>
<tr><td>FSB1REQ[4:0],
FSB1A[16:3]#</td><td>FSB1ADSTB[0]#</td></tr>
<tr><td>FSB1A[35:17]#</td><td>FSB1ADSTB[1]#</td></tr>
</table></td></tr>
<tr><td>FSB1AP[1:0]#</td><td>I/O</td><td>Processor 1 Address Parity:
FSB1AP[1:0]# provide parity protection on the address bus. FSB1AP[1:0]# are driven by the request initiator along with FSB1ADS#, FSB1A[35:3]#, and the transaction type on the FSB1REQ[4:0]# signals. A correct parity signal is high if an even number of covered signals are low and low if an odd number of covered signals are low. This allows parity to be high when all the covered signals are high. The following defines the coverage model of these signals.
<table>
<tr><th>Request Signals</th><th>Subphase 1</th><th>Subphase 2</th></tr>
<tr><td>FSB1A[35:24]#</td><td>FSB1AP[0]#</td><td>FSB1AP[1]#</td></tr>
<tr><td>FSB1A[23:3]#</td><td>FSB1AP[1]#</td><td>FSB1AP[0]#</td></tr>
<tr><td>FSB1REQ[4:0]#</td><td>FSB1AP[1]#</td><td>FSB1AP[0]#</td></tr>
</table></td></tr>
<tr><td>FSB1BINIT#</td><td>I/O</td><td>Processor 1 Bus Initialization:
FSB1BINIT# may be observed and driven by all processor FSB agents. If the FSB1BINIT# driver is enabled during power on configuration, FSB1BINIT# is asserted to signal any bus condition that prevents reliable future operation.
If FSB1BINIT# observation is enabled during power-on configuration and FSB1BINIT# is sampled asserted, symmetric agents reset their bus LOCK# activity and bus request arbitration state machines. The bus agents do not reset their I/O Queue (IOQ) and transaction tracking state machines upon observation of FSB1BINIT# assertion. Once the FSB1BINIT# assertion has been observed, the bus agents will re-arbitrate for the FSB and attempt completion of their bus queue and IOQ entries.
If FSB1BINIT# observation is disabled during power-on configuration, a priority agent may handle an assertion of FSB1BINIT# as appropriate to the error handling architecture of the system.</td></tr>
</table>

Table 6. Processor Front Side Bus 1 Signals (Sheet 2 of 4)

Signal Name	Type	Description
FSB1BNR#	I/O	**Processor 1 Block Next Request:** FSB1BNR# is used to assert a bus stall by any bus agent who is unable to accept new bus transactions. During a bus stall, the current bus owner cannot issue any new transactions. Since multiple agents might need to request a bus stall at the same time, FSB1BNR# is a wired-OR signal which must connect the appropriate pins of all processor FSB agents. In order to avoid wired-OR glitches associated with simultaneous edge transitions driven by multiple drivers, FSB1BNR# is activated on specific clock edges and sampled on specific clock edges. FSB1BNR# 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	**Processor 1 Breakpoint Monitor/Debug Bus:** FSB1BPM[5:0]# are breakpoint and performance monitor signals. They are outputs from the processor which indicate the status of breakpoints and programmable counters used for monitoring processor performance. FSB1BPM[5:0]# should connect the appropriate pins of all FSB agents. FSB1BPM[4]# provides PRDY# (Probe Ready) functionality for the TAP port. PRDY# is a processor output used by debug tools to determine processor debug readiness. FSB1BPM[5]# provides PREQ# (Probe Request) functionality for the TAP port. PREQ# is used by debug tools to request debug operation of the processors. FSB1BPM[5:4]# must be bussed to all bus agents. Please refer to the appropriate platform design guidelines for more detailed information.
FSB1BPRI#	O	**Processor 1 Priority Agent Bus Request:** FSB1BPRI# is used to arbitrate for ownership of the processor FSB. It must connect the appropriate pins of all processor FSB agents. Observing FSB1BPRI# active (as asserted by the priority agent) causes all other agents to stop issuing new requests, unless such requests are part of an ongoing locked operation. The priority agent keeps FSB1BPRI# asserted until all of its requests are completed, then releases the bus by deasserting FSB1BPRI#. 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 FSB1BREQ0# signal low during RESET#. The signal is sampled by the processor on the active-to-inactive transition of FSB1RESET#.
FSB1D[63:0]#	I/O	**Processor 1 Data Bus:** FSB1D[63:0]# are the data signals. These signals provide a 64-bit data path between the processor FSB agents. The data driver asserts FSB1DRDY# to indicate a valid data transfer. FSB1D[63:0]# are quad-pumped signals, and will thus be driven four times in a common clock period. FSB1D[63:0]# are latched off the falling edge of both FSB1DSTBP[3:0]# and FSB1DSTBN[3:0]#. Each group of 16 data signals correspond to one data strobe pair of FSB1DSTBP[3:0]# and FSB1DSTBN[3:0]#. The below table shows the grouping of data signals to strobes and FSB1DBI[3:0]#. Furthermore, the FSB1DBI[3:0]# signals determine the polarity of the data signals. Each group of 16 data signals corresponds to one of the FSB1DBI[3:0]# signals. When a specific FSB1DBI[3:0]# signal is active, the corresponding data group is inverted and therefore sampled active high.

Data Group	Data Strobe	Bus Inversion Signal
FSB1D[15:0]#	FSB1DSTB{P/N}[0]#	FSB1DBI[0]#
FSB1D[31:16]#	FSB1DSTB{P/N}[1]#	FSB1DBI[1]#
FSB1D[47:32]#	FSB1DSTB{P/N}[2]#	FSB1DBI[2]#
FSB1D[63:48]#	FSB1DSTB{P/N}[3]#	FSB1DBI[3]#

Table 6. Processor Front Side Bus 1 Signals (Sheet 3 of 4)

<table>
<tr><th>Signal Name</th><th>Type</th><th>Description</th></tr>
<tr><td>FSB1DBI[3:0]#</td><td>I/O</td><td>Processor 1 Dynamic Data Bus Inversion:
FSB1DBI[3:0]# are source synchronous and indicate the polarity of the FSB1D[63:0]# signals. The FSB1DBI[3:0]# signals are activated when the data on the data bus is inverted. If more than half the data bits, within, within a 16-bit group, would have been asserted electronically low, the bus agent may invert the data bus signals for that particular sub-phase for that 16-bit group. The below table shows the signal relationships.
<table>
<tr><th>Bus Inversion Signal</th><th>Data Group</th></tr>
<tr><td>FSB1DBI[0]#</td><td>FSB1D[15:0]#</td></tr>
<tr><td>FSB1DBI[1]#</td><td>FSB1D[31:16]#</td></tr>
<tr><td>FSB1DBI[2]#</td><td>FSB1D[47:32]#</td></tr>
<tr><td>FSB1DBI[3]#</td><td>FSB1D[63:48]#</td></tr>
</table></td></tr>
<tr><td>FSB1DBSY#</td><td>I/O</td><td>Processor 1 Data Bus Busy:
FSB1DBSY# is asserted by the agent responsible for driving data on the processor FSB to indicate that the data bus is in use. The data bus is released after FSB1DBSY# is deasserted. This signal is used by the data bus owner to hold the data bus for transfers requiring more than one cycle.</td></tr>
<tr><td>FSB1DEFER#</td><td>O</td><td>Processor 1 Data Bus Defer:
FSB1DEFER# is asserted by an agent to indicate that a transaction cannot be guaranteed in-order completion. Defer indicates that the MCH will terminate the transaction currently being snooped with either a deferred response or with a retry response.</td></tr>
<tr><td>FSB1DP[3:0]#</td><td>I/O</td><td>Processor 1 Data Bus Parity:
FSB1DP[3:0]# provide parity protection for the FSB1D[63:0]# signals. They are driven by the agent responsible for driving the FSB1D[63:0]# signals.</td></tr>
<tr><td>FSB1DRDY#</td><td>I/O</td><td>Processor 1 Data Ready:
FSB1DRDY# is asserted by the data driver on each data transfer, indicating valid data on the data bus. FSB1DRDY# may be deasserted to insert idle clocks. FSB1DRDY# is asserted for each cycle that data is transferred.</td></tr>
<tr><td>FSB1DSTBP[3:0]#
FSB1DSTBN[3:0]#</td><td>I/O</td><td>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.
<table>
<tr><th>Data Group</th><th>Data Strobe</th><th>Bus Inversion Signal</th></tr>
<tr><td>FSB1D[15:0]#</td><td>FSB1DSTB{P/N}[0]#</td><td>FSB1DBI[0]#</td></tr>
<tr><td>FSB1D[31:16]#</td><td>FSB1DSTB{P/N}[1]#</td><td>FSB1DBI[1]#</td></tr>
<tr><td>FSB1D[47:32]#</td><td>FSB1DSTB{P/N}[2]#</td><td>FSB1DBI[2]#</td></tr>
<tr><td>FSB1D[63:48]#</td><td>FSB1DSTB{P/N}[3]#</td><td>FSB1DBI[3]#</td></tr>
</table></td></tr>
<tr><td>FSB1HIT#</td><td>I/O</td><td>Processor 1 Cache Hit:
FSB1HIT# along with FSB1HITM# convey transaction snoop operation results. Any FSB agent may assert both FSB1HIT# and FSB1HITM# together to indicate that it requires a snoop stall to extend the snoop window. The stall an be continued by reasserting FSB1HIT# and FSB1HITM# together. The FSB1HIT# signal indicates that a caching agent holds an unmodified version of the requested line.</td></tr>
<tr><td>FSB1HITM#</td><td>I/O</td><td>Processor 1 Cache Hit Modified:
FSB1HITM# along with FSB1HIT# convey transaction snoop operation results. Any FSB agent may assert both FSB1HITM# and FSB1HIT# together to indicate that it requires a snoop stall to extend the snoop window. The stall an be continued by reasserting FSB1HITM# and FSB1HIT# together. The FSB1HITM# signal indicates that a caching agent holds a modified version of the requested line.</td></tr>
</table>

Table 6. Processor Front Side Bus 1 Signals (Sheet 4 of 4)

Signal Name	Type	Description
FSB1LOCK#	I	**Processor 1 Lock:** FSB1LOCK# indicates to the system that a transaction must occur atomically. For a locked sequence of transactions, FSB1LOCK# is asserted from the beginning of the first transaction to the end of the last transaction. When the priority agent asserts FSB1BPRI# to arbitrate for ownership of the processor FSB, it will wait until it observes FSB1LOCK# 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:** FSB1MCERR# is asserted to indicate an unrecoverable error without a bus protocol violation. It may be driven by all processor FSB agents. FSB1MCERR# assertion conditions are configurable at a system level. For more details regarding machine check architecture, refer to the *Intel® 64 and IA-32 Architectures Software Developer's Manual, Volume 3: System Programming Guide*.
FSB1REQ[4:0]#	I/O	**Processor 1 Bus Request Command:** FSB1REQ[4:0]# define the attributes of the request. FSB1REQ[4:0]# are transferred at 2x rate and are source synchronous to FSB1ADSTB[1:0]#. 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. Refer to the FSB1AP[1:0]# signal description for details on parity checking of these signals.
FSB1RESET#	O	**Processor 1 Reset:** FSB1RESET# is an output from the MCH. The MCH asserts FSB1RESET# while RESETI# (PLTRST# from ICH9R) is asserted and for approximately 1 ms after RESETI# is deasserted. The FSB1RESET# allows the processors to begin execution in a known state and invalidates their internal caches without writing back any of their contents.
FSB1RS[2:0]#	O	**Processor 1 Response Status Signals:** FSB1RS[2:0]# (Response Status) are driven by the response agent (the agent responsible for completion of the current transaction). FSB1RS[2:0]# 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
FSB1RSP#	O	**Processor 1 Response Status Parity:** FSB1RSP# provides parity protection for the FSB1RS[2:0] signals. Driven during assertion of FSB1RS[2:0]# as required for the agent responsible for completion of the current transaction. A correct parity signal is high if an even number of covered signals are low and low if an odd number of covered signals are low. When FSB1RS[2:0]# = 000 or idle is indicated, FSB1RSP# is high.
FSB1TRDY#	O	**Processor Bus 1 Target Ready:** FSB1TRDY# (Target Ready) is asserted by the target to indicate that it is ready to receive a write or implicit writeback data transfer, the target of the processor transaction is able to enter the data transfer phase.
FSB1VREF	Analog	**Processor 1 Voltage Reference:** FSB1VREF is Processor 1 voltage reference. Refer to the *Quad-Core and Dual-Core Intel® Xeon® Processor 5000 Sequence with Intel® 5100 Memory Controller Hub Chipset for Communications, Embedded, and Storage Applications – Platform Design Guide* or *Intel® Core™2 Duo Processors T9400 and SL9400 and Intel® 5100 Memory Controller Hub Chipset for Communications and Embedded Applications – Platform Design Guide* for the voltage value.

2.2 DDR2 Memory Channels

2.2.1 DDR2 Channel 0

DDR2 Channel 0 contains the following signals.

Table 7. DDR2 Channel 0 Signals (Sheet 1 of 3)

Name	Type	Description
CH0_A[15] /CH0_ODT[4]	O	**Memory Channel 0 Address bus (Bit 15)/On Die Termination (Bit 4):** When 48GB_Mode is strapped High, the signal functions as CH0_ODT[4]. This is to enable on die termination signal to the fifth rank on channel 0. When 48GB_Mode is strapped Low, the signal functions as CH0_A[15] which provides multiplexed row and column address to SDRAM as the sixteenth Address bit on channel 0.
CH0_A[14:0]	O	**Memory Channel 0 Address bus (Address Bits 14:0):** Provides multiplexed row and column address to SDRAM.
CH0_BA[2:0]	O	**Memory Channel 0 Bank Address:** Selects the bank within a rank, up to eight each.
CH0_CAS#	O	**Memory Channel 0 Column Address Strobe:** Used in write and pre-charge operations of DRAM. Specifies the SDRAM command in combination with CH0_CS#, CH0_RAS# and CH0_WE#.
CH0_CB[7:0]	I/O	**Memory Channel 0 Correction Bits:** Eight bits used for ECC calculations across channel 0.
CH0_CKE[3] /CH0_ODT[5]	O	**Memory Channel 0 Clock Enable (Bit 3)/On Die Termination (Bit 5):** When 48GB_Mode is strapped High, the signal functions as CH0_ODT[5]. This is to enable on die termination signal to the sixth rank on channel 0. In 48GB_Mode, there is only one Clock Enable required for each of the three DIMM slots required for channel 0. When 48GB_Mode is strapped Low, the signal functions as CH0_CKE[3], a command register enable per rank of a possible four ranks per channel, where CH0_CKE[0] is for the first rank and CH0_CKE[3] is for the fourth rank on channel 0.
CH0_CKE[2:0]	O	**Memory Channel 0 Clock Enable (Bits 2:0) :** When 48GB_Mode is strapped High, signals function as command register enables per DIMM for each of the required 3 DIMM slots on channel 0, where CH0_CKE[0] is for the first DIMM slot and CH0_CKE[2] is for the third DIMM slot. The first DIMM slot is furthest from the MCH. When 48GB_Mode is strapped Low, signals function as command register enable per rank of a possible four ranks per channel, where CH0_CKE[0] is for the first rank and CH0_CKE[3] is for the fourth rank. The first rank is in the first DIMM slot which is the furthest DIMM slot from the MCH. Refer to the *Quad-Core and Dual-Core Intel® Xeon® Processor 5000 Sequence with Intel® 5100 Memory Controller Hub Chipset for Communications, Embedded, and Storage Applications – Platform Design Guide* or *Intel® Core™2 Duo Processors T9400 and SL9400 and Intel® 5100 Memory Controller Hub Chipset for Communications and Embedded Applications – Platform Design Guide* for more details.
CH0_DCLKN[3] /CH0_CS[5]#	O	**Memory Channel 0 DDR Clock Negative (Clock 3)/Chip Select (Bit 5):** When 48GB_Mode is strapped High, the signal functions as CH0_CS[5]#. This is to enable chip select signal to the sixth rank on channel 0. When 48GB_Mode is strapped Low, the signal functions as CH0_DCLKN[3], the negative polarity of fourth DRAM Clock on channel 0 (Registered DIMMs, no unbuffered).
CH0_DCLKN[2:0]	O	**Memory Channel 0 DDR Clock Negative (Clocks 2:0):** Negative polarity of DRAM Clock (Registered DIMMs, no unbuffered). The first three clocks.

Table 7. DDR2 Channel 0 Signals (Sheet 2 of 3)

Name	Type	Description
CH0_DCLKP[3] /CH0_CS[4]#	O	**Memory Channel 0 DDR Clock Negative (Clock 3)/Chip Select (bit 4):** When 48GB_Mode is strapped High, the signal functions as CH0_CS[4]#. This is to enable chip select signal to the fifth rank on channel 0. When 48GB_Mode is strapped Low, the signal functions as CH0_DCLKP[3], the positive polarity of fourth DRAM Clock on channel 0 (Registered DIMMs, no unbuffered).
CH0_DCLKP[2:0]	O	**Memory Channel 0 DDR Clock Positive (Clocks 2:0):** Positive polarity of DRAM Clock (Registered DIMMs, no unbuffered). The first three clocks.
CH0_CRES1 CH0_CRES2	I	**Memory Channel 0 DDR2 Resistive Compensation I/Os:** The DDR circuits generate the logic reference used by inbound receivers by using CH0_CRES1 and CH0_CRES2. The CH0_CRES2 connects to the same power supply that is used by the DRAM drivers while CH0_CRES1 is connected to the board ground. By using an internal divider network, various required reference points can be generated.
CH0_CRESRET	I	**Memory Channel 0 Compensation Reference Return VSS:** Internal ground reference for the impedance and slew rate reference resistors.
CH0_CS[5:4]#	O	**Memory Channel 0 Chip Select:** When 48GB_Mode is strapped High, these signals in addition to CH0_CS[3:0]# specify the SDRAM command in combination with CH0_CAS#, CH0_RAS# and CH0_WE#. CH0_CS[5:0]# select one of six possible ranks, where CH0_CS[0]# selects the first rank and CH0_CS[5]# selects the sixth rank. When 48GB_Mode is strapped Low, these signals function as CH0_DCLKN[3] and CH0_DCLKP[15], respectively.
CH0_CS[3:0]#	O	**Memory Channel 0 Chip Select:** When 48GB_Mode is strapped High, these signals in addition to CH0_CS[5:4]# specify the SDRAM command in combination with CH0_CAS#, CH0_RAS# and CH0_WE#. CH0_CS[5:0]# select one of six possible ranks, where CH0_CS[0]# selects the first rank and CH0_CS[5]# selects the sixth rank. When 48GB_Mode is strapped Low, the signals specify the SDRAM command in combination with CH0_CAS#, CH0_RAS# and CH0_WE#. Selects one of four possible ranks, where CH0_CS[0]# selects the first rank and CH0_CS[3]# selects the fourth rank.
CH0_DQ[63:0]	I/O	**Memory Channel 0 Data Bus:** 64-bit data bus
CH0_DQSN[17:0]	I/O	**Memory Channel 0 Data Strobe Negative:** Negative polarity of Strove. Strobe for correction bits, CH0_CB[7:0], and data bus, CH0_DQ[63:0]. Each nibble of the 64-bit data bus and 8-bit check bit bus are associated with a strobe signal for a total of 18 strobe signals.
CH0_DQSP[17:0]	I/O	**Memory Channel 0 Data Strobe Positive:** Positive polarity of Strove. Strobe for correction bits, CH0_CB[7:0], and data bus, CH0_DQ[63:0]. Each nibble of the 64-bit data bus and 8-bit check bit bus are associated with a strobe signal for a total of 18 strobe signals.
CH0_DRVCRES	I	**Memory Channel 0 Driver Impedance Compensation Resistor:**
CH0_ODT[5:4]	O	**Memory Channel 0 On Die Termination (Bits 5:4):** When 48GB_Mode is strapped High, these signals in addition to CH0_ODT[3:0] operate as the dynamic on die termination enables for each of up to six ranks on channel 0, where CH0_ODT[0] is for the first rank and CH0_ODT[5] is for the sixth rank. When 48GB_Mode is strapped Low, these signals function as CH0_CKE[3] and CH0_A[15], respectively.
CH0_ODT[3:0]	O	**Memory Channel 0 On Die Termination (Bits 3:0):** When 48GB_Mode is strapped High, these signals in addition to CH0_ODT[5:4] operate as the dynamic on die termination enables for each of up to six ranks on channel 0, where CH0_ODT[0] is for the first rank and CH0_ODT[5] is for the sixth rank. When 48GB_Mode is strapped Low, these signals function as CH0_CKE[3] and CH0_A[15], respectively.

Table 7. DDR2 Channel 0 Signals (Sheet 3 of 3)

Name	Type	Description
CH0_RAS#	O	**Memory Channel 0 Row Address Strobe:** Used in write and pre-charge operations of DRAM. Specifies the SDRAM command in combination with CH0_CS#, CH0_CAS# and CH0_WE#.
CH0_SLEWCRES	I	**Memory Channel 0 Slew rate/DDR2_VTT Sense pin:** Connects to an external reference resistor to generate an internal bias which controls the slew rate of the drivers.
CH0_WE#	O	**Memory Channel 0 Write enable:** Used in write and pre-charge operations of DRAM. Specifies the SDRAM command in combination with CH0_CS#, CH0_CAS# and CH0_RAS#.

2.2.2 DDR2 Channel 1

DDR2 Channel 1 contains the following signals.

Table 8. DDR2 Channel 1 Signals (Sheet 1 of 3)

Name	Type	Description
CH1_A[15] /CH1_ODT[4]	O	**Memory Channel 1 Address bus (Bit 15)/On Die Termination (Bit 4):** When 48GB_Mode is strapped High, the signal functions as CH1_ODT[4]. This is to enable on die termination signal to the fifth rank on channel 1. When 48GB_Mode is strapped Low, the signal functions as CH1_A[15] which provides multiplexed row and column address to SDRAM as the sixteenth Address bit on channel 1.
CH1_A[14:0]	O	**Memory Channel 1 Address bus (Address Bit 14:0):** Provides multiplexed row and column address to SDRAM.
CH1_BA[2:0]	O	**Memory Channel 1 Bank Address:** Selects the bank within a rank, up to eight each.
CH1_CAS#	O	**Memory Channel 1 Column Address Strobe:** Used in write and pre-charge operations of DRAM. Specifies the SDRAM command in combination with CH1_CS#, CH1_RAS# and CH1_WE#.
CH1_CB[7:0]	I/O	**Memory Channel 1 Correction Bits:** Eight bits used for ECC calculations across channel 1.
CH1_CKE[3] /CH1_ODT[5]	O	**Memory Channel 1 Clock Enable (Bit 3)/On Die Termination (Bit 5):** When 48GB_Mode is strapped High, the signal functions as CH1_ODT[5]. This is to enable on die termination signal to the sixth rank on channel 1. In 48GB_Mode, there is only one Clock Enable required for each of the three DIMM slots required for channel 1. When 48GB_Mode is strapped Low, the signal functions as CH1_CKE[3], a command register enable per rank of a possible four ranks per channel, where CH1_CKE[0] is for the first rank and CH1_CKE[3] is for the fourth rank on channel 1.
CH1_CKE[2:0]	O	**Memory Channel 1 Clock Enable (Bits 2:0):** When 48GB_Mode is strapped High, signals function as command register enables per DIMM for each of the required 3 DIMM slots on channel 1, where CH1_CKE[0] is for the first DIMM slot and CH1_CKE[2] is for the third DIMM slot. The first DIMM slot is furthest from the MCH. When 48GB_Mode is strapped Low, signals function as command register enable per rank of a possible four ranks per channel, where CH1_CKE[0] is for the first rank and CH1_CKE[3] is for the fourth rank. The first rank is in the first DIMM slot which is the furthest DIMM slot from the MCH. Refer to the *Quad-Core and Dual-Core Intel® Xeon® Processor 5000 Sequence with Intel® 5100 Memory Controller Hub Chipset for Communications, Embedded, and Storage Applications – Platform Design Guide* or *Intel® Core™2 Duo Processors T9400 and SL9400 and Intel® 5100 Memory Controller Hub Chipset for Communications and Embedded Applications – Platform Design Guide* for more details.

Table 8. DDR2 Channel 1 Signals (Sheet 2 of 3)

Name	Type	Description
CH1_DCLKN[3] /CH1_CS[5]#	O	**Memory Channel 1 DDR Clock Negative (Clock 3)/Chip Select (Bit 5):** When 48GB_Mode is strapped High, the signal functions as CH1_CS[5]#. This is to enable chip select signal to the sixth rank on channel 1. When 48GB_Mode is strapped Low, the signal functions as CH1_DCLKN[3], the negative polarity of fourth DRAM Clock on channel 1 (Registered DIMMs, no unbuffered).
CH1_DCLKN[2:0]	O	**Memory Channel 1 DDR Clock Negative (Clocks 2:0):** Negative polarity of DRAM Clock (Registered DIMMs, no unbuffered). The first three clocks.
CH1_DCLKP[3] /CH1_CS[4]#	O	**Memory Channel 1 DDR Clock Negative (Clock 3)/Chip Select (Bit 4):** When 48GB_Mode is strapped High, the signal functions as CH1_CS[4]#. This is to enable chip select signal to the fifth rank on channel 1. When 48GB_Mode is strapped Low, the signal functions as CH1_DCLKP[3], the positive polarity of fourth DRAM Clock on channel 1 (Registered DIMMs, no unbuffered).
CH1_DCLKP[2:0]	O	**Memory Channel 1 DDR Clock Positive (Clocks 2:0):** Positive polarity of DRAM Clock (Registered DIMMs, no unbuffered). The first three clocks.
CH1_CRES1 CH1_CRES2	I	**Memory Channel 1 DDR2 resistive compensation I/Os:** The DDR circuits generate the logic reference used by inbound receivers by using CH1_CRES1 and CH1_CRES2. The CH1_CRES2 connects to the same power supply that is used by the DRAM drivers while CH1_CRES1 is connected to the board ground. By using an internal divider network, various required reference points can be generated.
CH1_CRESRET	I	**Memory Channel 1 Compensation Reference Return VSS:** Internal ground reference for the impedance and slew rate reference resistors.
CH1_CS[5:4]#	O	**Memory Channel 1 Chip Select:** When 48GB_Mode is strapped High, these signals in addition to CH1_CS[3:0]# specify the SDRAM command in combination with CH1_CAS#, CH1_RAS# and CH1_WE#. CH1_CS[5:0]# select one of six possible ranks, where CH1_CS[0]# selects the first rank and CH1_CS[5]# selects the sixth rank. When 48GB_Mode is strapped Low, these signals function as CH1_DCLKN[3] and CH1_DCLKP[15], respectively.
CH1_CS[3:0]#	O	**Memory Channel 1 Chip Select:** When 48GB_Mode is strapped High, these signals in addition to CH1_CS[5:4]# specify the SDRAM command in combination with CH1_CAS#, CH1_RAS# and CH1_WE#. CH1_CS[5:0]# select one of six possible ranks, where CH1_CS[0]# selects the first rank and CH1_CS[5]# selects the sixth rank. When 48GB_Mode is strapped Low, the signals specify the SDRAM command in combination with CH1_CAS#, CH1_RAS# and CH1_WE#. Selects one of four possible ranks, where CH1_CS[0]# selects the first rank and CH1_CS[3]# selects the fourth rank.
CH1_DQ[63:0]	I/O	**Memory Channel 1 Data Bus:** 64-bit data bus
CH1_DQSN[17:0]	I/O	**Memory Channel 1 Data Strobe Negative:** Negative polarity of Strove. Strobe for correction bits, CH1_CB[7:0], and data bus, CH1_DQ[63:0]. Each nibble of the 64-bit data bus and 8-bit check bit bus are associated with a strobe signal for a total of 18 strobe signals.
CH1_DQSP[17:0]	I/O	**Memory Channel 1 Data Strobe Positive:** Positive polarity of Strove. Strobe for correction bits, CH1_CB[7:0], and data bus, CH1_DQ[63:0]. Each nibble of the 64-bit data bus and 8-bit check bit bus are associated with a strobe signal for a total of 18 strobe signals.
CH1_DRVCRES	I	**Memory Channel 1 Driver Impedance Compensation Resistor:**

Table 8. DDR2 Channel 1 Signals (Sheet 3 of 3)

Name	Type	Description
CH1_ODT[5:4]	O	**Memory Channel 1 On Die Termination (Bits 5:4):** When 48GB_Mode is strapped High, these signals in addition to CH1_ODT[3:0] operate as the dynamic on die termination enables for each of up to six ranks on channel 1, where CH1_ODT[0] is for the first rank and CH1_ODT[5] is for the sixth rank. When 48GB_Mode is strapped Low, these signals function as CH1_CKE[3] and CH1_A[15], respectively.
CH1_ODT[3:0]	O	**Memory Channel 1 On Die Termination (Bits 3:0):** When 48GB_Mode is strapped High, these signals in addition to CH1_ODT[5:4] operate as the dynamic on die termination enables for each of up to six ranks on channel 1, where CH1_ODT[0] is for the first rank and CH1_ODT[5] is for the sixth rank. When 48GB_Mode is strapped Low, these signals function as CH1_CKE[3] and CH1_A[15], respectively.
CH1_RAS#	O	**Memory Channel 1 Row Address Strobe:** Used in write and pre-charge operations of DRAM. Specifies the SDRAM command in combination with CH1_CS#, CH1_CAS# and CH1_WE#.
CH1_SLEWCRES	I	**Memory Channel 1 Slew rate/DDR2_VTT Sense pin:** Connects to an external reference resistor to generate an internal bias which controls the slew rate of the drivers.
CH1_WE#	O	**Memory Channel 1 Write enable:** Used in write and pre-charge operations of DRAM. Specifies the SDRAM command in combination with CH1_CS#, CH1_CAS# and CH1_RAS#.

2.3 PCI Express* Signal List

2.3.1 PCI Express* Common Signals

Table 9. 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:**
PERCOMPO	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® 5100 MCH Chipset and the ICH9R.

Table 10. PCI Express* Port 0, Enterprise South Bridge Interface (ESI) Signals

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 can be combined with PCI Express* port 3 to form a single PCI Express* x8 port. The combined x8 PCI Express* port is controlled by the configuration registers of the lowest x4 port number, the other port registers are inactive.

Table 11. PCI Express* Port 2 Signals

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 a x4 port. PCI Express* port 3 is combinable with PCI Express* port 2 to form a single PCI Express* x8 port. The combined x8 PCI Express* port is controlled by the configuration registers of the lowest x4 port number, the other port registers are inactive.

Table 12. PCI Express* Port 3 Signals

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* Port 4

PCI Express* port 4 is a x4 port. PCI Express* port 4 is combinable with PCI Express* port 5 to form a single PCI Express* x8 port or ports 5, 6, and 7 to form a single x16 port. The combined x8 or x16 PCI Express* port is controlled by the configuration registers of the lowest x4 port number, the other port registers are inactive.

Table 13. PCI Express* Port 4 Signals

Signal Name	Type	Description
PE4RP[3:0]	I	**PCI Express* Port 4 Positive Phase Inbound:** (Receive) Signals
PE4RN[3:0]	I	**PCI Express* Port 4, Negative Phase Inbound:** (Receive) Signals
PE4TP[3:0]	O	**PCI Express* Port 4, Positive Phase Outbound:** (Transmit) Signals
PE4TN[3:0]	O	**PCI Express* Port 4, Negative Phase Outbound:** (Transmit) Signal:

2.3.6 PCI Express* Port 5

PCI Express* port 5 is a x4 port. PCI Express* port 5 is combinable with PCI Express* port 4 to form a single PCI Express* x8 port or ports 4, 6 and 7 to form a single x16 port. The combined x8 or x16 PCI Express* port is controlled by the configuration registers of the lowest x4 port number, the other port registers are inactive.

Table 14. PCI Express* Port 5 Signals

Signal Name	Type	Description
PE5RP[3:0]	I	**PCI Express* Port 5 Positive Phase Inbound:** (Receive) Signals
PE5RN[3:0]	I	**PCI Express* Port 5 Negative Phase Inbound:** (Receive) Signals
PE5TP[3:0]	O	**PCI Express* Port 5 Positive Phase Outbound:** (Transmit) Signals
PE5TN[3:0]	O	**PCI Express* Port 5 Negative Phase Outbound:** (Transmit) Signals

2.3.7 PCI Express* Port 6

PCI Express* port 6 is a x4 port. PCI Express* port 6 is combinable with PCI Express* port 7 to form a single PCI Express* x8 port or ports 4, 5 and 7 to form a single x16 port. The combined x8 or x16 PCI Express* port is controlled by the configuration registers of the lowest x4 port number, the other port registers are inactive.

Table 15. PCI Express* Port 6 Signals (Sheet 1 of 2)

Signal Name	Type	Description
PE6RP[3:0]	I	**PCI Express* Port 6 Positive Phase Inbound:** (Receive) Signals

Table 15. PCI Express* Port 6 Signals (Sheet 2 of 2)

Signal Name	Type	Description
PE6RN[3:0]	I	**PCI Express* Port 6 Negative Phase Inbound:** (Receive) Signals
PE6TP[3:0]	O	**PCI Express* Port 6 Positive Phase Outbound:** (Transmit) Signals
PE6TN[3:0]	O	**PCI Express* Port 6 Negative Phase Outbound:** (Transmit) Signals

2.3.8 PCI Express* Port 7

PCI Express* port 7 is a x4 port. PCI Express* port 7 is combinable with PCI Express* port 6 to form a single PCI Express* x8 port or ports 4, 5 and 6 to form a single x16 port. The combined x8 or x16 PCI Express* port is controlled by the configuration registers of the lowest x4 port number, the other port registers are inactive.

Table 16. PCI Express* Port 7 Signals

Signal Name	Type	Description
PE7RP[3:0]	I	**PCI Express* Port 7 Positive Phase Inbound:** (Receive) Signals
PE7RN[3:0]	I	**PCI Express* Port 7 Negative Phase Inbound:** (Receive) Signals
PE7TP[3:0]	O	**PCI Express* Port 7 Positive Phase Outbound:** (Transmit) Signals
PE7TN[3:0]	O	**PCI Express* Port 7 Negative Phase Outbound:** (Transmit) Signals

2.3.9 PCI Express* Graphics Port

In the Intel® 5100 MCH Chipset PCI Express* ports 4, 5, 6 and 7 are combined to form a single high performance x16 graphics port. The combined x16 PCI Express* port is controlled by the configuration registers of the lowest x4 port number, port 4 in this case, the other port registers, ports 5, 6 and 7 are inactive.

Table 17. PCI Express* Graphics Port Signals (Sheet 1 of 2)

Signal Name	Type	Description
PE4RP[3:0]	I	PCI Express* Graphics Port First x4, Positive Phase Inbound
PE4RN[3:0]	I	PCI Express* Graphics Port First x4, Negative Phase Inbound
PE4TP[3:0]	O	PCI Express* Graphics Port First x4, Positive Phase Outbound
PE4TN[3:0]	O	PCI Express* Graphics Port First x4, Negative Phase Outbound
PE5RP[3:0]	I	PCI Express* Graphics Port Second x4, Positive Phase Inbound
PE5RN[3:0]	I	PCI Express* Graphics Port Second x4, Negative Phase Inbound
PE5TP[3:0]	O	PCI Express* Graphics Port Second x4, Positive Phase Outbound
PE5TN[3:0]	O	PCI Express* Graphics Port Second x4, Negative Phase Outbound
PE6RP[3:0]	I	PCI Express* Graphics Port Third x4, Positive Phase Inbound
PE6RN[3:0]	I	PCI Express* Graphics Port Third x4, Negative Phase Inbound
PE6TP[3:0]	O	PCI Express* Graphics Port Third x4, Positive Phase Outbound
PE6TN[3:0]	O	PCI Express* Graphics Port Third x4, Negative Phase Outbound

Table 17. PCI Express* Graphics Port Signals (Sheet 2 of 2)

Signal Name	Type	Description
PE7RP[3:0]	I	PCI Express* Graphics Port Fourth x4, Positive Phase Inbound
PE7RN[3:0]	I	PCI Express* Graphics Port Fourth x4, Negative Phase Inbound
PE7TP[3:0]	O	PCI Express* Graphics Port Fourth x4, Positive Phase Outbound
PE7TN[3:0]	O	PCI Express* Graphics Port Fourth x4, Negative Phase Outbound

2.4 SMBus Interfaces

There are three SMBus interfaces supporting three basic functions. These functions are:

- System management
- PCI Hot Plug*
- DDR2 DIMM serial presence detect

Table 18. SMBus Interfaces Signals

Signal Name	Type	Description
CFGSMBCLK	I/O	**Slave SMBus Clock:** SMBus Clock for the slave CFGSMBus interface.
CFGSMBDATA	I/O	**Slave SMBus Data:** SMBus Address/Data for the slave CFGSMBus interface.
GPIOSMBCLK	I/O	**PCI SMBus Clock:** PCI Hot Plug* Master VPI, SMBus Clock. The master GPIOSMBus interface only supports PCI Hot Plug* capabilities; it does not support general purpose I/O functions.
GPIOSMBDATA	I/O	**PCI SMBus Data:** PCI Hot Plug* Master VPI, SMBus Address/Data. The master GPIOSMBus interface only supports PCI Hot Plug* capabilities; it does not support general purpose I/O functions.
SPD0SMBCLK	I/O	**DDR2 DIMM Channel 0/1 SMBus Clock:** DDR2 DIMM Memory Serial Presence Detect 0, SMBus Clock for the master SPD0SMBus interface. The SPD0SMBus interface only supports DIMM configuration/status capabilities; it is not designed for support of general purpose I/O functions.
SPD0SMBDATA	I/O	**DDR2 DIMM Channel 0/1 SMBus Data:** DDR2 DIMM Memory Serial Presence Detect 0, SMBus Address/Data

Notes:

1. These signals are Open Drain (OD) and require pull-ups, see *Quad-Core and Dual-Core Intel® Xeon® Processor 5000 Sequence with Intel® 5100 Memory Controller Hub Chipset for Communications, Embedded, and Storage Applications – Platform Design Guide* or *Intel® Core™2 Duo Processors T9400 and SL9400 and Intel® 5100 Memory Controller Hub Chipset for Communications and Embedded Applications – Platform Design Guide* for pull-up requirements.

2.5 Extended Debug Port (XDP) Signal List

Table 19. Extended Debug Port (XDP) Signals

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

Table 20. JTAG Bus Signals

Signal Name	Type	Description
TCK	I	**Test Clock:** TCK provides the JTAG clock input for the MCH Test Bus (also known as the Test Access Port).
TDI	I	**Test Data In:** TDI transfers serial test data into the MCH. TDI provides the serial input needed for JTAG specification support.
TDO	O	**Test Data Out:** TDO transfers serial test data out of the MCH. TDO provides the serial output needed for JTAG specification support.
TMS	I	**Test Mode Select:** TMS is a JTAG specification support signal used by debug tools. See the *Debug Port Design Guide for Intel® 5000 Series Chipset Based Platforms (External Version)* for further information.
TRST#	I	**Test Reset:** TRST# resets the Test Access Port (TAP) logic. TRST# must be driven low during power on Reset. Asynchronous reset of the JTAG interface.

2.7 Clocks, Reset and Miscellaneous

Table 21. Clocks, Reset and Miscellaneous Signals (Sheet 1 of 2)

Signal Name	Type	Description
48GB_Mode	I	**48 GB DDR2 Memory Mode Selection:** This pin is a strap pin to control whether 48 GB maximum (48 GB mode) or 32 GB maximum (32 GB mode) memory support is enabled. By strapping this pin High, 6 ranks are supported per DDR channel, enabling 48 GB memory support. By strapping this pin Low, four ranks are supported per DDR channel, enabling 32 GB memory support. This mode provides backward compatibility for boards designed for A0 versions of silicon limited to 32 GB of memory. The strapping of the 48GB_Mode pin affects the function of the following signals CH0_A[15], CH0_CKE[3], CH0_DCLKN[3], CH0_DCLKP[3], CH1_A[15], CH1_CKE[3], CH1_DCLKN[3], and CH1_DCLKP[3]. See the *Quad-Core and Dual-Core Intel® Xeon® Processor 5000 Sequence with Intel® 5100 Memory Controller Hub Chipset for Communications, Embedded, and Storage Applications – Platform Design Guide* or *Intel® Core™2 Duo Processors T9400 and SL9400 and Intel® 5100 Memory Controller Hub Chipset for Communications and Embedded Applications – Platform Design Guide* for connection restrictions. ***Note:*** The 48 GB mode is the recommended mode of operation for current designs.
ASYNCRFSH	I	**Asynchronous Request for Self-refresh:** Allows an asynchronous request to force DIMMs into Self-Refresh.
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.
ERR[2:0]#	O	**Error Output:** Error output signal recommended configuration: ERR[0] = Correctable and recoverable error from the memory subsystem ERR[1] = Uncorrectable error from the Intel® 5100 MCH Chipset ERR[2] = Fatal error from the Intel® 5100 MCH Chipset Note: The error assignments to the ERR[2:0] pins are configurable with the use of the , "PEX_ERR_DOCMD[7:2,0] - PCI Express* Error Do Command Register". These signals are Open Drain (OD) outputs; see the *Quad-Core and Dual-Core Intel® Xeon® Processor 5000 Sequence with Intel® 5100 Memory Controller Hub Chipset for Communications, Embedded, and Storage Applications – Platform Design Guide* or *Intel® Core™2 Duo Processors T9400 and SL9400 and Intel® 5100 Memory Controller Hub Chipset for Communications and Embedded Applications – Platform Design Guide* for pull-up requirements.
FSBCRES	Analog	**Processor Bus Compensation:**
FSBODTCRES	Analog	**Processor Bus Compensation:**
FSBSLWCRES	Analog	**Processor Bus Slew Rate Compensation:**
FSBSLWCTRL	Power/ Other	**Processor Bus Slew Rate Control:**

Table 21. Clocks, Reset and Miscellaneous Signals (Sheet 2 of 2)

Signal Name	Type	Description
PWRGOOD	I	**Power Good:** PWRGOOD is an input. The MCH requires this signal to be a clean indication that all clocks and power supplies are stable and within their specifications. "Clean" implies that the signal will remain low (capable of sinking leakage current), without glitches, from the time that the power supplies are turned on until they come within specification. The signal must then transition monotonically to a high state. Section 5.19, "System Reset"describes the relationship of PWRGOOD to the RESET# signal. PWRGOOD can be driven inactive at any time, but clocks and power must again be stable before a subsequent rising edge of PWRGOOD. The PWRGOOD signal must be supplied to the MCH; it is used to protect internal circuits against voltage sequencing issues. It should be driven high throughout boundary scan operation.
PSEL[2:0]	I	**Processor Speed Select:** The Processor BCLK[1:0] frequency select signals PSEL[2:0] are used to select the processor input clock frequency. The required frequency is determined by the processors, chipset, and clock synthesizer. All FSB agents must operate at the same frequency. The below table indicates the currently supported MCH FSB speeds. **PSEL[2:0] / FSB Speed (MHz)** 000 — 267 100 — 333 For more information about these signals, including termination recommendations, refer to the appropriate platform design guideline.
RESETI#	I	**MCH Reset:** This is the hard reset
RSVD	No Connect	**Reserved Pin:** These pins are required to be No Connects, not connected to any signal or supply reference/voltage. The MCh may not behave as designed, if these pins are connect to a signal or voltage level.
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

2.8 Power and Ground Signals

Table 22. Power and Ground Signals (Sheet 1 of 2)

Signal Name	Description
COREVCCA	**Core VCC (1.5 V):** Analog Voltage for the PLL
COREVSSA	**Core VSS (0 V):** Analog Voltage for PLL
FSBVCCA	**FSB VCC (1.5 V):** Analog Voltage for the PLL
V3REF	**SMBus VCC (3.3 V):** Common Voltage for SMBus buses and other Open Drain 3.3 V referenced signals. V3REF supplies the following signal groups: ERR[2:0]#, SMBus, RESETI#, PWRGOOD.

Table 22. Power and Ground Signals (Sheet 2 of 2)

Signal Name	Description
VCC	**VCC Supply (1.5 V):** This is the core voltage.
VSS	**Ground Return (0 V):** Common return for power supplies
VTT	**VTT Supply:** FSB I/O supply. Supports multiple FSB supply references: 1.05 V, 1.1 V or 1.2 V.
VCCDDR	**VCC for System Memory (1.8 V):** The DDR2 memory interface power.
VCCPE	**VCC for PCI Express* ports (1.5 V):** The supply voltage for the PCI Express* ports.

Figure 2. Intel® 5100 Memory Controller Hub Chipset Signal Diagram 32 GB Mode

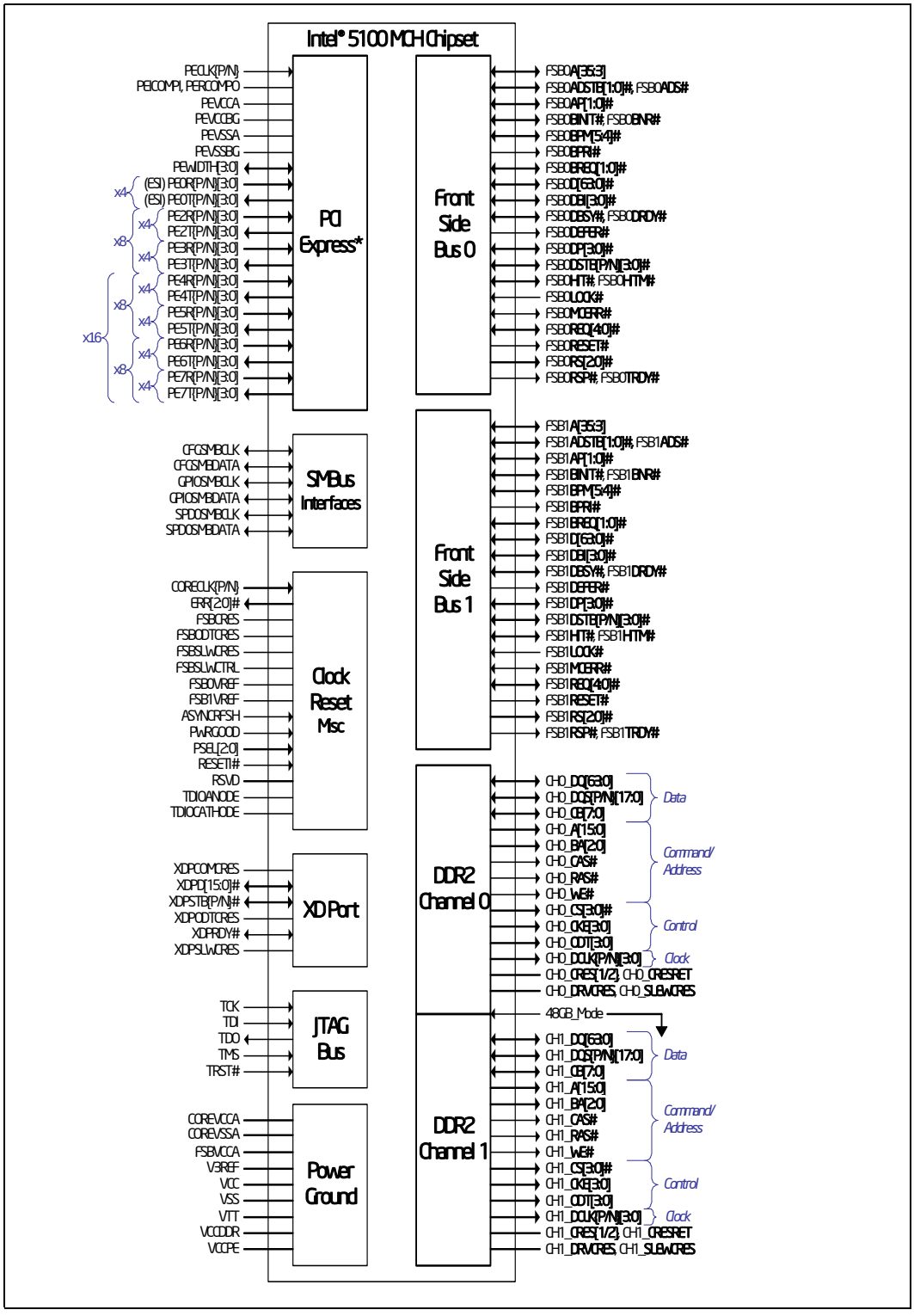
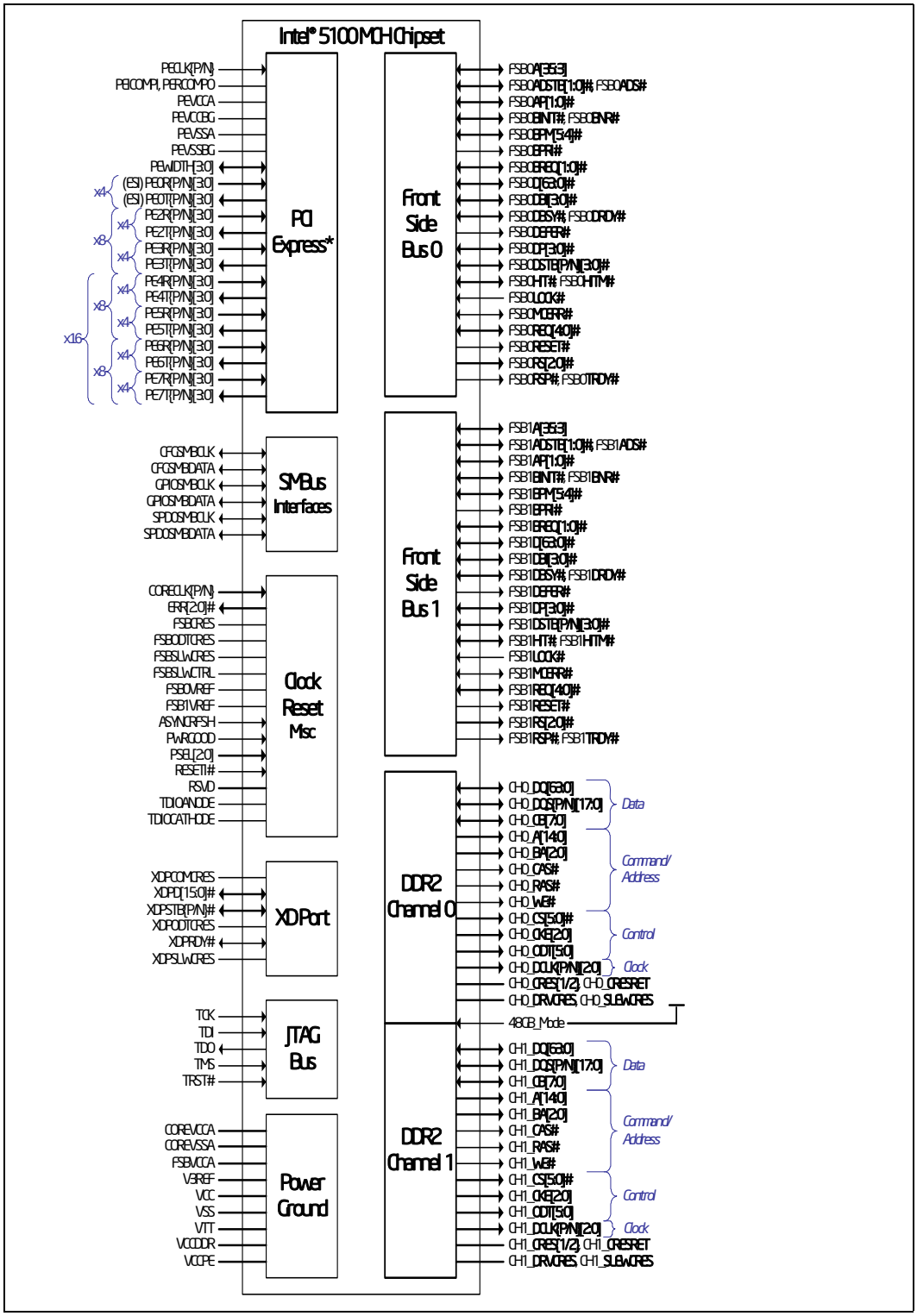


Figure 3. Intel® 5100 Memory Controller Hub Chipset Signal Diagram 48 GB Mode

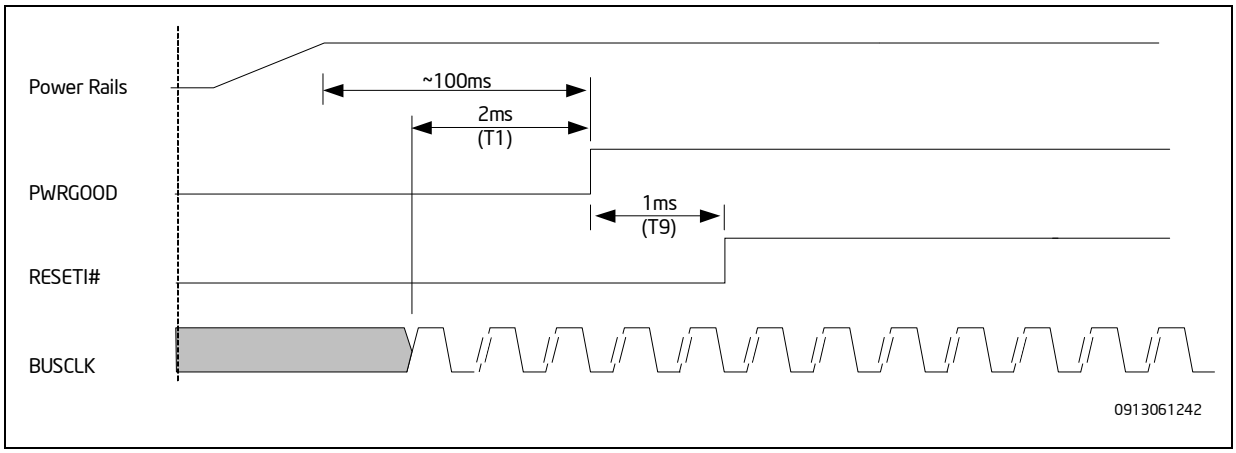


2.9 Intel® 5100 Memory Controller Hub Chipset Sequencing Requirements

Power Plane and Sequencing Requirements:

- Clock Valid Timing:
- BUSCLK must be valid at least 2 ms prior to rising edge of PWRGOOD. See Section 5.22.1, "Reference Clocks" for definition of BUSCLK.

Figure 4. Intel® 5100 Memory Controller Hub Chipset Clock and Reset Requirements



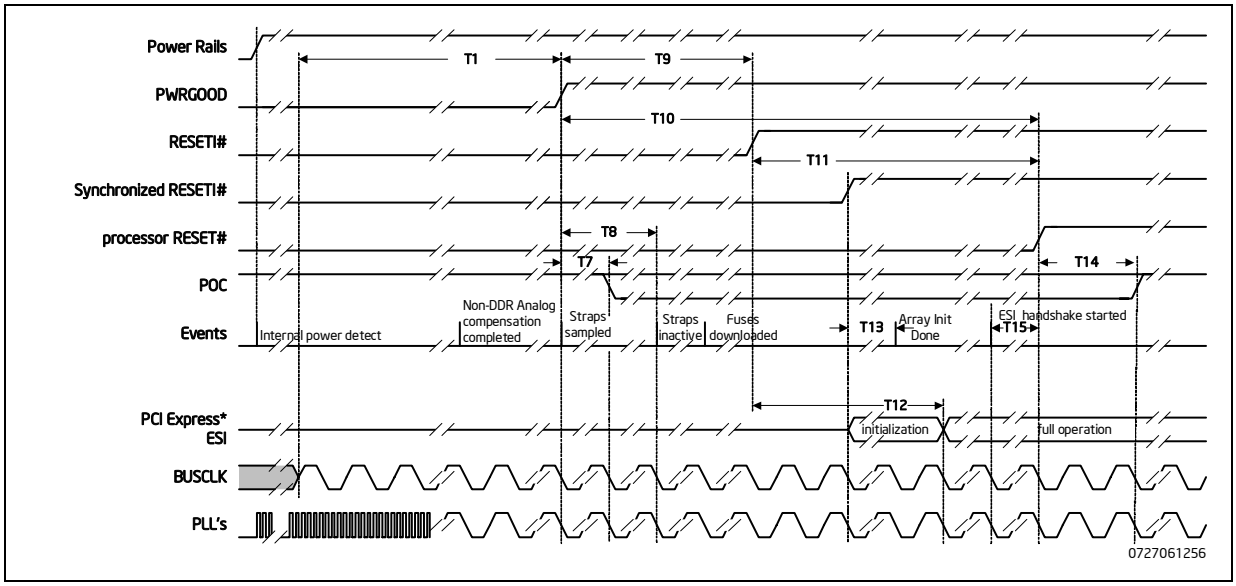
2.10 Reset Requirements

2.10.1 Timing Diagrams

2.10.1.1 Power-Up

The power-up sequence is illustrated in Figure 5, "Power-Up."

Figure 5. Power-Up

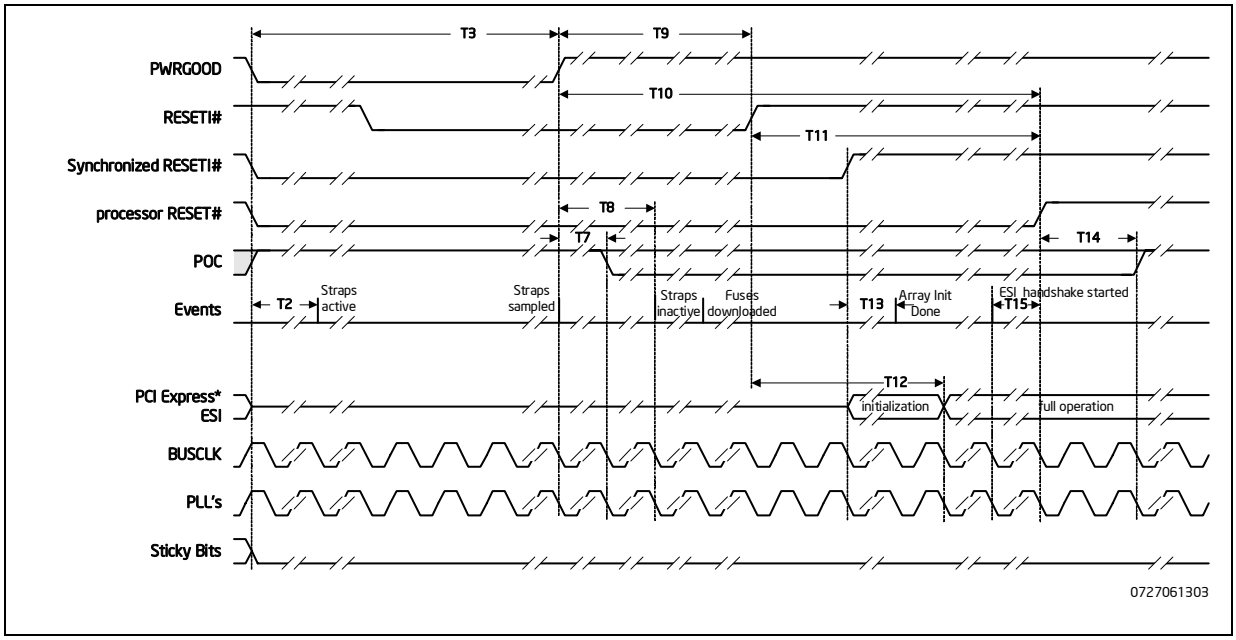


Note: Synchronized RESETI# is the RESETI# signal synchronized with the necessary internal clock domain, the PLLs are the internal PLLs locking to the BUSCLK signal and POC is Power-On Configuration, see Section 1.1, "Terminology".

2.10.1.2 Power Good

The PWRGOOD reset sequence is illustrated in Figure 6, "PWRGOOD."

Figure 6. PWRGOOD

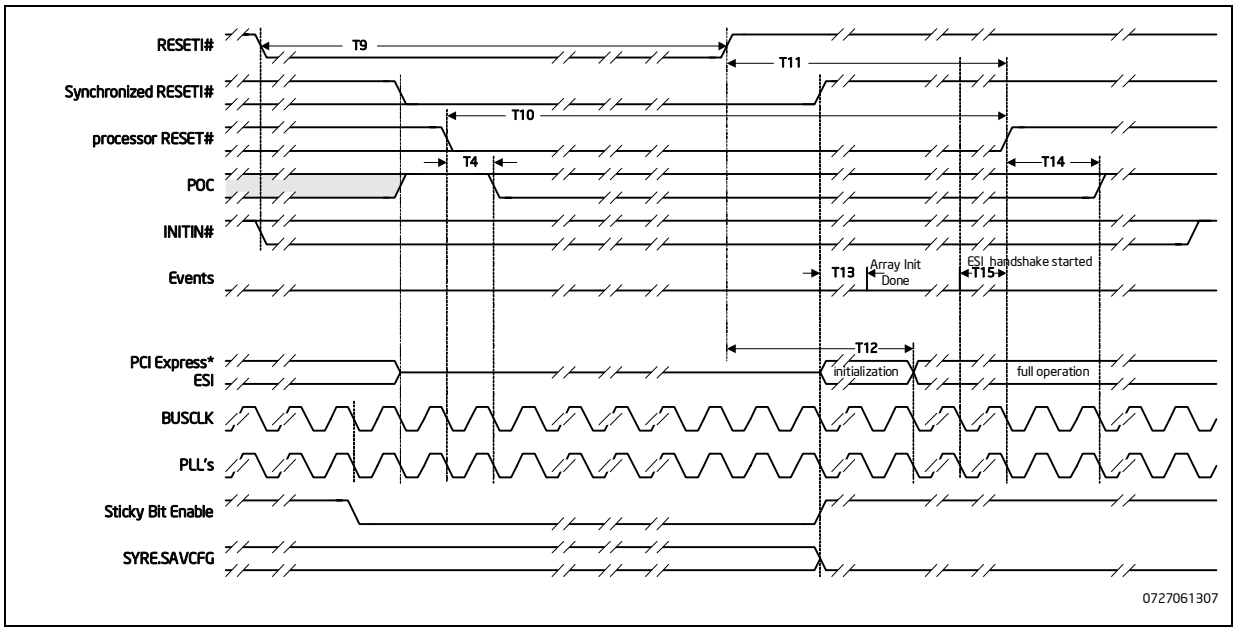


Note: Synchronized RESETI# is the RESETI# signal synchronized with the necessary internal clock domain, the PLLs are the internal PLLs locking to the BUSCLK signal and POC is Power-On Configuration, see Section 1.1, "Terminology".

2.10.1.3 Hard Reset

The Hard Reset sequence is illustrated in Figure 7, "Hard Reset."

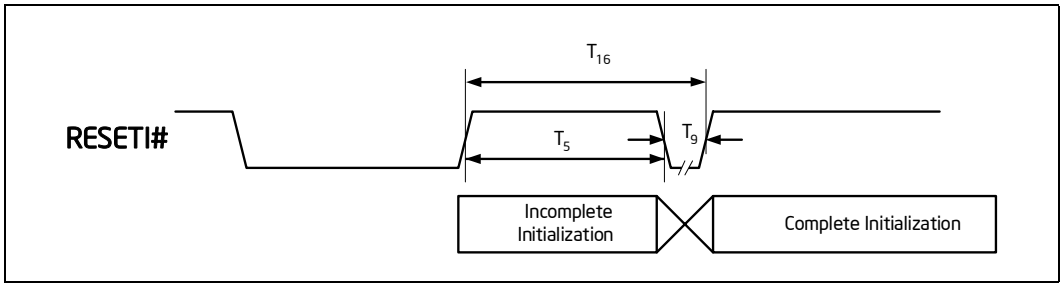
Figure 7. Hard Reset



2.10.1.4 RESETI# Retriggering Limitations

Figure 8, "RESETI# Retriggering Limitations" shows the timing for a RESETI# retrigger.

Figure 8. RESETI# Retriggering Limitations



2.10.2 Reset Timing Requirements

Table 23, "Power Up and Hard Reset Timings" specifies the timings drawn in Figure 5, "Power-Up," Figure 6, "PWRGOOD," Figure 7, "Hard Reset," and Figure 8, "RESETI# Retriggering Limitations." Nominal clock frequencies are described. Specifications still hold for derated clock frequencies.

Table 23. Power Up and Hard Reset Timings (Sheet 1 of 2)

Timing	Description	Min	Max	Comments
T1	Power supplies and system clocks stable to PWRGOOD signal assertion	2 ms		PLL specification. See Figure 4 and Figure 5.
T2	PWRGOOD deassertion to straps active		40 ns	See Figure 6.
T3	PWRGOOD deassertion	80 ns		Minimum PWRGOOD deassertion time while power and platform clocks are stable. See Figure 6.
T4	Power-On-Configuration (POC) after processor RESET# assertion delay	one BUSCLK		See Figure 7.
T5	Platform reset deassertion to platform reset assertion	50 BUSCLK's		Minimum re-trigger time on RESETI# deassertion. See Figure 8.
T7	PWRGOOD assertion to POC active	two BUSCLK's		POC turn-on delay after strap disable. See Figure 5 and Figure 6.
T8	PWRGOOD assertion to straps inactive	12 ns	18 ns	Strap Hold Time. See Figure 5 and Figure 6.
T9	RESETI# signal assertion during PWRGOOD/ PWROK signal assertion	1 ms		This delay can be provided by the ICH or by system logic. See Figure 4, Figure 5, Figure 6 and Figure 7. PWROK is an ICH9R input.
T10	RESET# assertion during processor PWRGOOD assertion	1 ms	10 ms	Processor EMTS. See Figure 5, Figure 6 and Figure 7.
T11	RESETI# signal deassertion to processor RESET# signal deassertion	480 µs[1]		***Note:*** This is a special Dual-Core Intel® Xeon® processor 5100 series requirement to have a longer POC assertion setup time on the FSB. See Figure 5, Figure 6 and Figure 7.
T12	RESETI# signal deassertion to completion of PCI Express* initialization sequence		1,250,000 PECLKs	PCI Express* clock is 100 MHz. See Figure 5, Figure 6 and Figure 7.
T13	Internal Memory Array Initialization duration		200 cycles	CORECLK cycles. See Figure 5, Figure 6 and Figure 7.
T14	POC hold time after RESET# deassertion	two BUSCLK's	19 BUSCLKs	Processor EMTS specification. See Figure 5, Figure 6 and Figure 7.

Table 23. Power Up and Hard Reset Timings (Sheet 2 of 2)

Timing	Description	Min	Max	Comments
T15	Initiation of ESI reset sequence to processor RESET# signal deassertion		10,000 PECLKs + T17	ICH9R specification. See Figure 5, Figure 6 and Figure 7.
T16	RESETI# re-trigger delay	T5 + T9		See Figure 8.
T17	CPU_RESET_DONE capture timer	2,000 BUSCLK's		No Figure reference.

1. In the Intel® 5100 MCH Chipset, the T11 duration is implemented through a counter with max value of 162,000 core clocks. For 333 MHz, this gives a period of 486 µs for the POC setup time while @266 MHz, the period is 607.5 µs.

Table 24, "Critical Intel® 5100 Memory Controller Hub Chipset Initialization Timings" summarizes the Intel® 5100 MCH Chipset Initialization timings.

Table 24. Critical Intel® 5100 Memory Controller Hub Chipset Initialization Timings

Sequence	Started by	Maximum Length	Covered by Timing parameter
Intel® 5100 MCH Chipset Core, FSB	Stable power and master clock	666,667 333 MHz cycles	T1
Intel® 5100 MCH Chipset PCI Express* PLL lock	Stable power and master clock	200,000 100 MHz cycles	
Array initialization	Synchronized RESETI# Deassertion	200 cycles	T13
Fuse download	PWRGOOD Assertion	333,333 333 MHz cycles	T9

2.10.3 Miscellaneous Requirements and Limitations

- Power rails and stable BUSCLK, CH{0/1}_DCLK{P/N}, and PECLK master clocks remain within specifications through all but power-up reset.
- Frequencies (for example, 266/333 MHz) described in this section are nominal. The Intel® 5100 MCH Chipset reset sequences must work for the frequency of operation range specified in Section 5.22, "Clocking."
- Hard Reset can be initiated by code running on a processor, SMBus, or PCI agents.
- Hard Reset is not guaranteed to correct all illegal configurations or malfunctions. Software can configure sticky bits in the Intel® 5100 MCH Chipset to disable interfaces that will not be accessible after Hard Reset. Signaling errors or protocol violations prior to reset (from processor bus or PCI Express*) may hang interfaces that are not cleared by Hard Reset. A PWRGOOD or power-on reset is required to clear these conditions, if present.
- 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 default values of the Power-On-Configuration (POC) 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® 5100 MCH Chipset cannot meet the SPD data $t_{SU, STO}$ timing specification. Since the Intel® 5100 MCH Chipset 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 µs (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® 5100 Memory Controller Hub Chipset Customer Reference Platform (CRP) Reset Topology

Typical implementation and routing of PWRGOOD, system Reset and interrupt connections for a Intel® 5100 MCH Chipset-based platform are described in the *Quad-Core and Dual-Core Intel® Xeon® Processor 5000 Sequence with Intel® 5100 Memory Controller Hub Chipset for Communications, Embedded, and Storage Applications – Platform Design Guide* and *Intel® Core™2 Duo Processors T9400 and SL9400 and Intel® 5100 Memory Controller Hub Chipset for Communications and Embedded Applications – Platform Design Guide*.

2.12 Signals Used as Straps

2.12.1 Functional Straps

The PEWIDTH signals are used to determine the widths of the six PCI Express* ports and 48 GB mode signals determine which DDR2 memory mode to enable.

Table 25. Signals Used as Straps

Signal Name	Type	Description
48GB_Mode	Power/ Other	**48 GB DDR2 Memory Mode Selection:** The strapping of this pin determines the maximum memory supported, 32 GB or 48 GB. See above DDR2 signal descriptions for more information.
PEWIDTH[3:0]	Power/ Other	**PCI Express* Port Width Strapping Pins**: For strapping options refer to Table 95, "PCI Express* Link Width Strapping Options for Port CPCI Configuration in Intel® 5100 Memory Controller Hub Chipset" on page 312

3.0 Register Description

The Intel® 5100 MCH Chipset contains 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 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 (Device 0 controls PCI bus 0, i.e., DRAM configuration, other chipset operating parameters, and optional features).

The MCH supports PCI configuration space accesses using the mechanism denoted as Configuration Mechanism 1 as defined in the *PCI Local Bus Specification*, Rev. 2.3. All the registers are organized by bus, device, function, etc., as defined in the *PCI Express* Base Specification*, Rev. 1.0a. The MCH supports registers in PCI Express* extended space. All registers in the Intel® 5100 MCH 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 the SMBus. The memory mapped access mechanism is further broken down into different ranges. The internal registers of this chipset 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 (i.e., lower addresses contain the least significant parts of the field).

In addition, the MCH can forward accesses to all PCI/PCI Express* configuration registers from the ICH or other PCI Express* bridges through the same mechanisms.

3.1 Register Terminology

Registers and register bits are assigned one or more of the attributes described in Table 26. These attributes define the behavior of the bit(s) that are contained within a register. Sticky bits retain their states across hard resets. All other bits are set to default values by a hard reset.

Table 26. Register Attributes Summary Table

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.

Table 26. Register Attributes Summary Table

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 the bit and prevent it from being 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 modified. The *PCI Local Bus Specification*, Rev. 2.3 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, an MCH and a south bridge device 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® 5100 MCH Chipset-based platform the configuration structure is significantly different. The MCH and the south bridge device, the ICH9R in this implementation, 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 ICH9R appear to be on PCI bus 0. The system's primary PCI

expansion bus is physically attached to the ICH9R, 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.

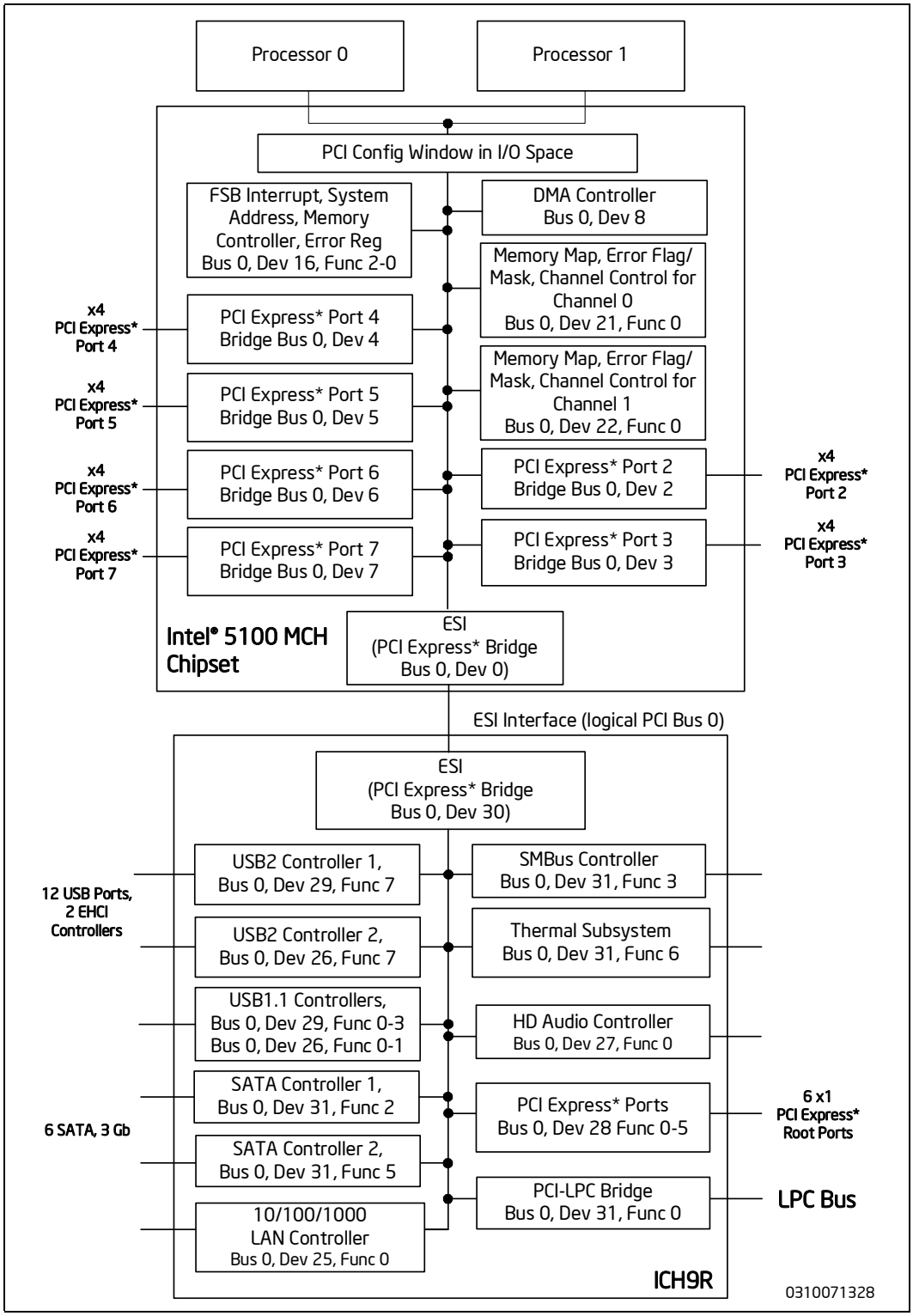
The MCH contains 12 PCI devices within a single physical component. The configuration registers for these devices are mapped as devices residing on PCI bus 0.

- **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. PCI Express* port 0 resides at DID of 65C0h.
- **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 65E2h.
- **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 65E3h.
- **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 and 5 are combined into a single x8 port, Device 4, Function 0 contains the configuration registers and Device 5, Function 0 (port 5) configuration registers are inactive. 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 65E4h.
- **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 and 5 are combined into a single x8 port Device 4, Function 0 contains the configuration registers, and these configuration registers are inactive. 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 65E5h.
- **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 6 and 7 are combined into a single x8 port Device 6, Function 0 contains the configuration registers, and Device 7, Function 0 (port 7) configuration registers are inactive. 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 65E6h.
- **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 6 and 7 are combined into a single x8 port Device 6, Function 0 contains the configuration registers, and these configuration registers are inactive. 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 65E7h.
- **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 65FFh.

- **Device 16**: Device 16, Function 0 is routed to the Front Side Bus (FSB) Controller, Interrupt and System Address registers. Function 1 is routed to the Front Side Bus Address Mapping, Memory Control, and Error registers. Function 2 is routed to FSB Error Registers. These devices reside at DID 65F0h.
- **Device 17**: Reserved. These devices reside at DID 65F1h.
- **Device 19:** Device 19, Function 0 is routed to Miscellaneous registers. These devices reside at DID 65F3h.
- **Device 21:** Device 21, Function 0, Channel 0 Memory Map, Error Flag/Mask, and Channel Control registers. These devices reside at DID 65F5h.
- **Device 22:** Device 22, Function 0, Channel 1 Memory Map, Error Flag/Mask, and Channel 1 Control registers. These devices reside at DID 65F6h.

Figure 9. Conceptual Intel® 5100 Memory Controller Hub Chipset PCI Configuration Diagram



3.3 Routing Configuration Accesses

The Intel® 5100 MCH Chipset supports both PCI Type 0 and Type 1 configuration access mechanisms as defined in the *PCI Local Bus Specification*, Rev. 2.3. *PCI Local Bus Specification*, Rev. 2.3 defines hierarchical PCI busses. Type 0 configuration accesses are used for registers located within a PCI device that resides on the local PCI bus, i.e., the PCI bus on which the transaction is initiated. 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. If 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 to (behind) the 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 1s.

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 Local Bus Specification*, Rev. 2.3 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 Local Bus Specification*, Rev. 2.3 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.

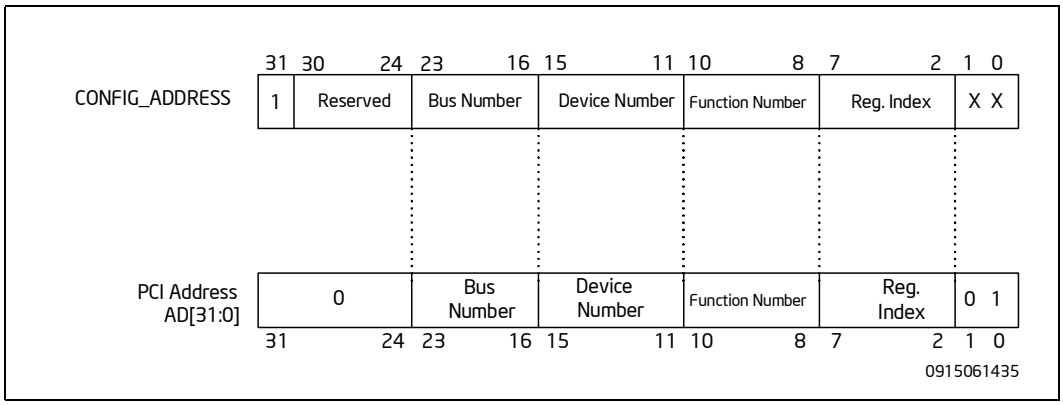
The ESI bridge entity within the MCH is hardwired as Device 0 on PCI Bus 0. The ESI bridge passes PCI south bridge configuration requests to the south bridge. The ICH9R is the south bridge device for the Intel® 5100 MCH Chipset platform.

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 10, "Type 1 Configuration Address to PCI Address Mapping." This configuration cycle will be sent over the ESI to ICH9R.

If the cycle is forwarded to the ICH9R via ESI, the ICH9R 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 ICH9R's PCI Express* ports, or a downstream PCI bus.

Figure 10. Type 1 Configuration Address to PCI Address Mapping



3.4 Device Mapping

Each component in a Intel® 5100 MCH Chipset-based 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 within a Intel® 5100 MCH Chipset must support Type 0 configuration accesses. All MCH registers in the Intel® 5100 MCH Chipset appear on Bus #0.

All Intel® 5100 MCH Chipset 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. Table 27, "Configuration Address Bit Mapping" 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 27. Configuration Address Bit Mapping (Sheet 1 of 2)

	Source/ Destination	Bus	Device	Function	DW Offset		Byte in DW	Type
					[11:8]	[5:0]		
PCI Express* Config Txns (including ESI)	Both	Bus[7:0]	Dev[4:0]	Func[2:0]	Extended Reg[3:0]	Reg[5:0]	First 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 DDR2 registers and will be master aborted. Peer-to-peer accesses targeting valid MMIO space will be forwarded through appropriate decoding.							

Table 27. Configuration Address Bit Mapping (Sheet 2 of 2)

	Source/ Destination	Bus	Device	Function	DW Offset		Byte in DW	Type
					[11:8]	[5:0]		
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[7:0]	Dev[4:0]	Func[2:0]	Not present	Reg[5:0]	Not present	N/A
CFC on FSB	Source	CFGADR Register—see row above					BE[7:4]	N/A
SMBus Config Access	Source	Bus[7:0]	Dev[4:0]	Func[2:0]	Reg[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 28, "Functions Specially Handled by Intel® 5100 Memory Controller Hub Chipset."	Func[2:0]	DW Offset[9:6]	DW Offset[5:0]	Byte[3:0]	N/A

3.4.1 Special Device and Function Routing

All devices in the Intel® 5100 MCH Chipset reside on Bus 0. Table 28, "Functions Specially Handled by Intel® 5100 Memory Controller Hub Chipset" describes the devices and functions that the MCH implements or routes specially.

Table 28. Functions Specially Handled by Intel® 5100 Memory Controller Hub Chipset (Sheet 1 of 2)

Component	Register Group	DID	Device	Function	Comment
MCH	ESI Port (Port 0)	65C0h	0	0	Depending on what is connected to these ports, some may not be accessible.
MCH	PCI Express* Port 2	65E2h	2	0	
MCH	PCI Express* Port 3	65E3h	3	0	
MCH	PCI Express* Port 4	65E4h	4	0	
MCH	PCI Express* Port 5	65E5h	5	0	
MCH	PCI Express* Port 6	65E6h	6	0	
MCH	PCI Express* Port 7	65E7h	7	0	
MCH	DMA Engine	65FFh	8	0	Device mapping for DMA Engine
MCH	DMA Engine MMIO Space	N/A	8	1	
MCH	Miscellaneous Registers.	65F3h	19	0	Debug in higher address offsets.
MCH	Memory Map, Error Flag/Mask, RAS, Channel 0 Control	65F5h	21	0	Debug in higher address offsets.
MCH	Memory Map, Error Flag/Mask, RAS, Channel 1 Control	65F6h	22	0	Debug in higher address offsets.
MCH	Processor Bus, Boot, Interrupt, System Address	65F0h	16	0	

Table 28. Functions Specially Handled by Intel® 5100 Memory Controller Hub Chipset (Sheet 2 of 2)

Component	Register Group	DID	Device	Function	Comment
MCH	Address Mapping, Memory Control, Error Logs	65F0h	16	1	
MCH	FSB Error Registers	65F0h	16	2	
MCH	PCI Express* Port 2-3	65F7h	2	0	x8 mode. Only port 2 is active
MCH	PCI Express* Port 4-5	65F8h	4	0	x8 mode. Only port 4 is active
MCH	PCI Express* Port 6-7	65F9h	6	0	x8 mode. Only port 6 is active
MCH	PCI Express* Port 4-7	65FAh	4	0	x16 mode. Only port 4 is active

To comply with the *PCI Local Bus Specification*, Rev. 2.3, accesses to non-existent functions, registers, and bits will be master aborted. This behavior is defined in Table 29, "Access to "Non-Existent" Register Bits."

Table 29. Access to "Non-Existent" Register Bits

Access to	Writes	Reads
Devices listed in Table 28, "Functions Specially Handled by Intel® 5100 Memory Controller Hub Chipset," but to functions not listed	Have no effect	MCH returns all ones
Devices listed in Table 28, "Functions Specially Handled by Intel® 5100 Memory Controller Hub Chipset," 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 the Intel® 5100 MCH Chipset state. The first address is the DWORD location (CF8h) that references a read/write register 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. Therefore, 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 27, "Configuration Address Bit Mapping."

Table 30. 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	0h	Reserved.
23:16	RW	0h	**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	0h	**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 31. 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 Intel® 5100 Memory Controller Hub Chipset 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.

Table 32, "Mapping for Fixed Memory Mapped Registers" defines the memory address of the registers in this region.

Table 32. Mapping for Fixed Memory Mapped Registers (Sheet 1 of 2)

Register	Memory Address
BOFL0	FE60_C000
BOFL1	FE60_C400
BOFL2	FE60_C800
BOFL3	FE60_CC00
SPAD0	FE60_D000
SPAD1	FE60_D400

Table 32. Mapping for Fixed Memory Mapped Registers (Sheet 2 of 2)

Register	Memory Address
SPAD2	FE60_D800
SPAD3	FE60_DC00
SPADS0	FE60_E000
SPADS1	FE60_E400
SPADS2	FE60_E800
SPADS3	FE60_EC00
HECBASE	FE61_6400

3.7 Detailed Configuration Space Maps

Table 33. Device 0, Function 0: PCI Express* PCI Space

DID			VID	00h	PEXSLOTCAP			80h
PEXSTS			PEXCMD	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
		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 34. 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
SCSPCAPID	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
	170h		1F0h
	174h		1F4h
	178h		1F8h
	17Ch		1FCh

Table 35. Device 2-3, Function 0: PCI Express* PCI Space

DID			VID	00h	PEXSLOTCAP			80h
PEXSTS			PEXCMD	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
				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 36. 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
SCSPCAPID	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
	170h		1F0h
	174h		1F4h
	178h		1F8h
	17Ch		1FCh

Table 37. Device 4, Function 0: PCI Express* PCI Space

DID			VID	00h	PEXSLOTCAP			80h
PEXSTS			PEXCMD	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
				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 38. 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
SCSPCAPID	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
	170h		1F0h
	174h		1F4h
	178h		1F8h
	17Ch		1FCh

Table 39. Device 5-7, Function 0: PCI Express* PCI Space

DID			VID	00h	PEXSLOTCAP			80h
PEXSTS			PEXCMD	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
				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 40. 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
SCSPCAPID	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
	170h		1F0h
	174h		1F4h
	178h		1F8h
	17Ch		1FCh

Table 41. 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
				48h	BOFL2	C8h
				4Ch	BOFL3	CCh
				50h	SPAD0	D0h
				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
FSBS0				74h		F4h
FSBC1				78h		F8h
FSBS1				7Ch		FCh

Table 42. Device 16, Function 0: Processor Bus 0 Error Registers

	100h	NERR_NF_FSB	NERR_FAT_FSB	FERR_NF_FSB	FERR_FAT_FSB	180h
	104h	NRECFSB				184h
	108h	RECFSB				188h
	10Ch	NRECADDRL				18Ch
	110h	EMASK_FSB			NRECADDRH	190h
	114h	ERR1_FSB		ERR0_FSB		194h
	118h	MCERR_FSB		ERR2_FSB		198h
	11Ch					19Ch
	120h					1A0h
	124h					1A4h
	128h					1A8h
	12Ch					1ACh
	130h					1B0h
	134h					1B4h
	138h					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 43. Device 16, Function 0: Processor Bus 1 Error Registers

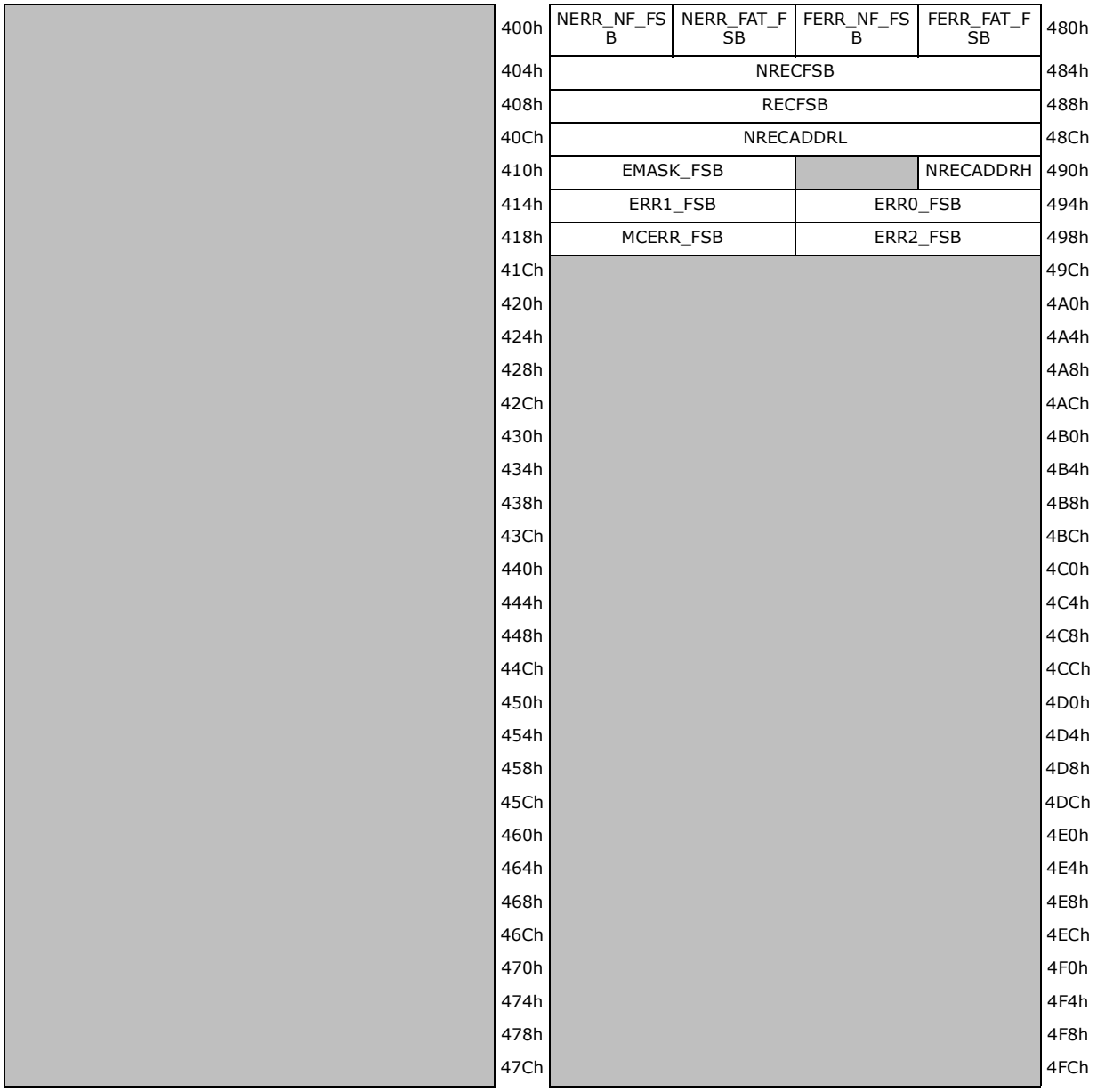


Table 44. Device 16, Function 1: Memory Branch Map, Control, Errors

DID	VID	00h		MIR0	80h
		04h		MIR1	84h
CCR	RID	08h		Reserved	88h
HDR		0Ch		AMIR0	8Ch
		10h		AMIR1	90h
		14h		Reserved	94h
		18h			98h
		1Ch			9Ch
		20h	FERR_NF_MEM		A0h
		24h	NERR_NF_MEM		A4h
		28h	EMASK_MEM		A8h
SID	SVID	2Ch	ERR0_MEM		ACh
		30h	ERR1_MEM		B0h
		34h	ERR2_MEM		B4h
		38h	MCERR_MEM		B8h
		3Ch			BCh
MC		40h			C0h
MS		44h			C4h
	SPDDATA	48h			C8h
SPDCMD		4Ch			CCh
ERRPER		50h			D0h
DDRFRQ		54h			D4h
MCA		58h			D8h
		5Ch			DCh
	GBLACT	60h			E0h
THRTHIGH	THRTLOW	64h			E4h
THRTSTS1	THRTSTS0	68h			E8h
	TOLM	6Ch			ECh
		70h			F0h
		74h			F4h
		78h			F8h
		7Ch	MCDEF3		FCh

Table 45. Device 16, Function 1: Memory Gearing Registers

	100h		180h
	104h		184h
	108h		188h
	10Ch		18Ch
	110h		190h
	114h		194h
	118h		198h
	11Ch		19Ch
	120h		1A0h
	124h		1A4h
	128h		1A8h
	12Ch		1ACh
	130h		1B0h
	134h		1B4h
	138h		1B8h
	13Ch		1BCh
	140h		1C0h
	144h		1C4h
	148h		1C8h
	14Ch		1CCh
	150h		1D0h
	154h		1D4h
	158h		1D8h
	15Ch		1DCh
MEMTOHOSTGRCFG0	160h		1E0h
MEMTOHOSTGRCFG1	164h		1E4h
MEMNDGRCFG0	168h		1E8h
MEMNDGRCFG1	16Ch		1ECh
HOSTTOMEMGRCFG0	170h		1F0h
HOSTTOMEMGRCFG1	174h		1F4h
	178h		1F8h
	17Ch		1FCh

Table 46. Device 16, Function 1: Memory DFx Registers

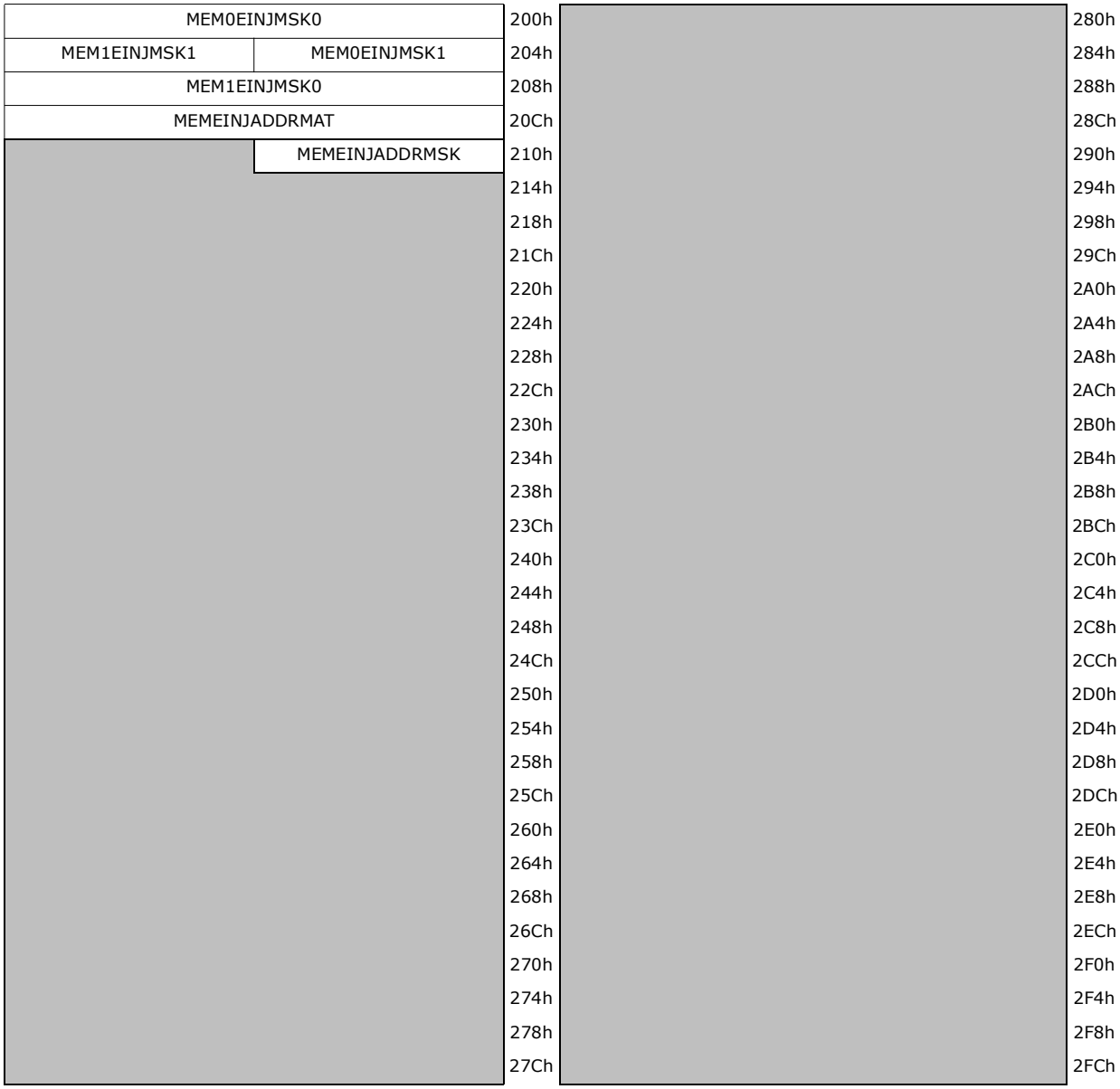


Table 47. Device 16, Function 2: RAS

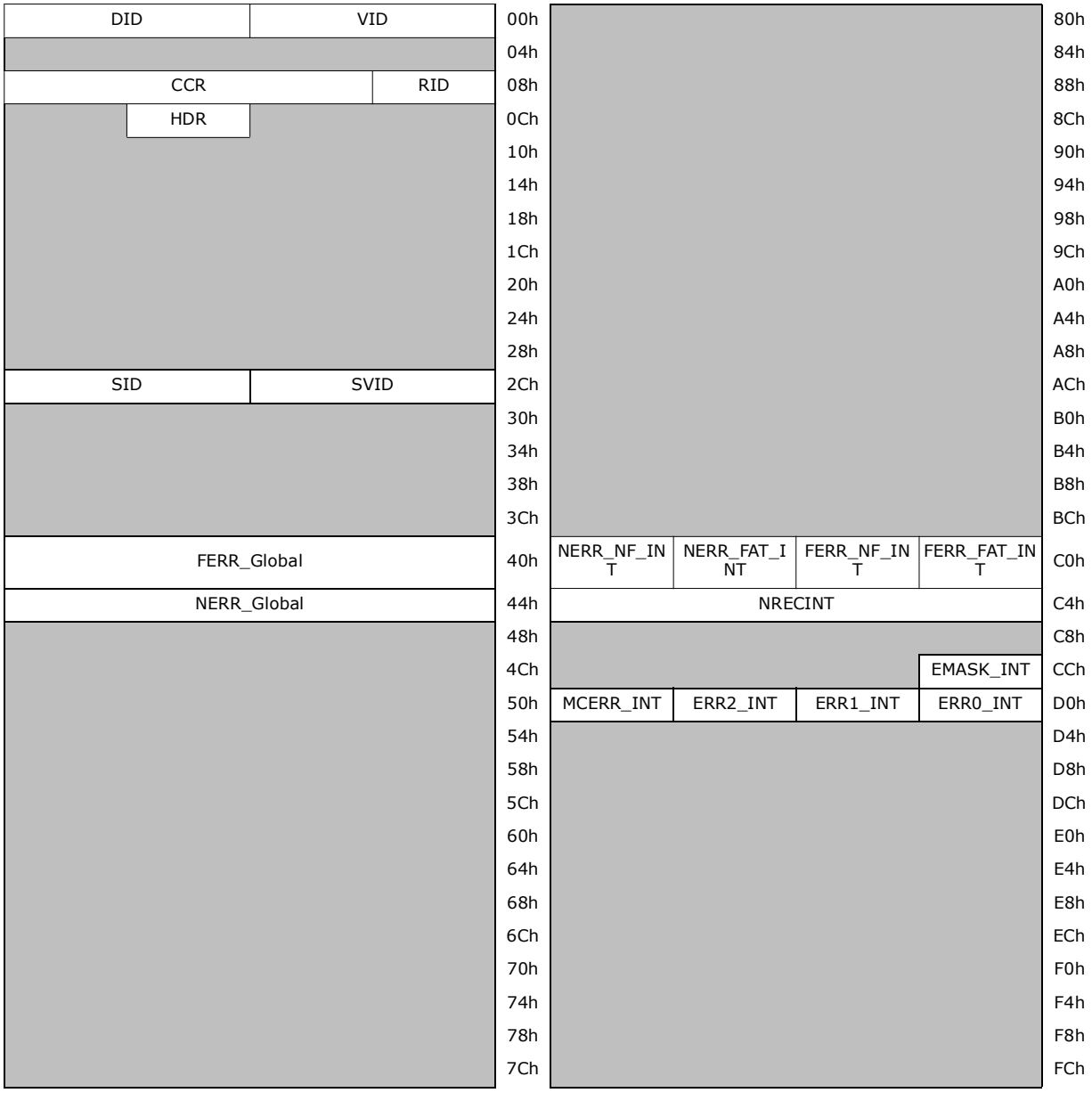


Table 48. Device 21, 22, Function 0: DIMM Map, Control, RAS

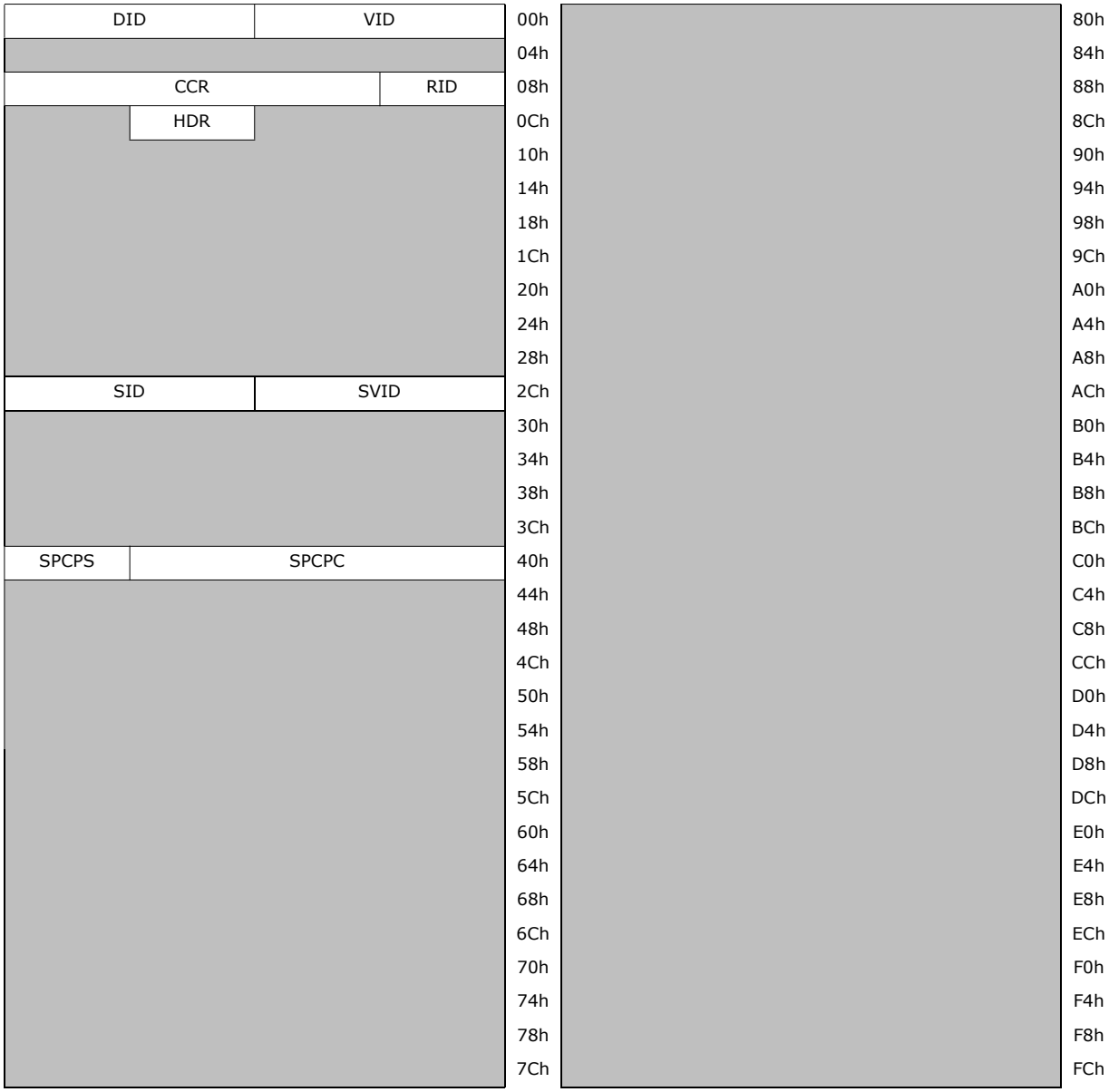


Table 49. Device 21, 22, Function 0: Memory Map, Control, Errors

		100h	180h
	CERRCNT_EXT	104h	184h
		108h	188h
		10Ch	18Ch
		110h	190h
		114h	194h
		118h	198h
		11Ch	19Ch
		120h	1A0h
		124h	1A4h
		128h	1A8h
		12Ch	1ACh
	MTR4	130h	1B0h
		134h	1B4h
	RANKTHRESHOLD4	138h	1B8h
		13Ch	1BCh
		140h	1C0h
DSRETC		144h	1C4h
		148h	1C8h
		14Ch	1CCh
		150h	1D0h
MTR0		154h	1D4h
MTR2		158h	1D8h
		15Ch	1DCh
		160h	1E0h
		164h	1E4h
		168h	1E8h
		16Ch	1ECh
RANKTHRESHOLD0		170h	1F0h
RANKTHRESHOLD2		174h	1F4h
		178h	1F8h
		17Ch	1FCh

CERRCNT
VALIDLOG
NRECMEMA
NRECMEMB
REDMEMA
REDMEMB
RECMEMA
RECMEMB
MTR5
RANKTHRESHOLD5
DRPADCTL
DRTA
DRTB
MTR1
MTR3
DMIR0
DMIR1
DMIR2
DMIR3
DMIR4
RANKTHRESHOLD1
RANKTHRESHOLD3
BADRAM
BADCNT

3.8 Register Definitions

Warning: Address locations that are not listed are considered reserved register locations. Reads to reserved registers may return non-zero values. Writes to reserved locations may cause system failure.

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, 2-7, 8
Function: 0
Offset: 00h
Device: 16
Function: 0, 1, 2
Offset: 00h
Device: 21, 22
Function: 0
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 within the MCH. Writes to this register have no effect. See Table 28, "Functions Specially Handled by Intel® 5100 Memory Controller Hub Chipset" for the DID of each MCH function.

Device: 0, 2-7, 8
Function: 0
Offset: 02h
Device: 16
Function: 0, 1, 2
Offset: 02h
Device: 21, 22
Function: 0
Offset: 02h

Bit	Attr	Default	Description
15:0	RWO	See Table 28, "Functions Specially Handled by Intel® 5100 Memory Controller Hub Chipset."	**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 chipsets for every stepping. There is 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® 5100 MCH Chipset 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® 5100 MCH Chipset'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® 5100 MCH Chipset RID registers before setting the SRID/CRID register select flip flop.

The RID values for all devices and functions in the Intel® 5100 MCH Chipset 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® 5100 MCH Chipset 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, 2-7, 8
Function: 0
Offset: 08h
Device: 16
Function: 0, 1, 2
Offset: 08h
Device: 21, 22
Function: 0
Offset: 08h

Bit	Attr	Default	Description
7:4	RO	0h	**Major Revision** Steppings which require all masks to be regenerated[1]. 1000: A stepping for the Intel® 5100 MCH Chipset 1001: B stepping for the Intel® 5100 MCH Chipset as an example 1010: C stepping for the Intel® 5100 MCH Chipset as an example Others: *Reserved* This field is set appropriately by the hardware based on the current stepping and may differ from this default value.
3:0	RO	0h	**Minor Revision** Incremented for each stepping which does not modify all masks. Reset for each major revision[1]. 0h: M0 stepping 1h: M1 stepping 2h: M2 stepping Others: *Reserved* ***Note:*** The Metal steppings indicated are a subset of the Major revision. For example, an A stepping with M0 as minor revision typically means A0. The field will be set appropriately

1. 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 11, "Intel® 5100 Memory Controller Hub Chipset Implementation of SRID and CRID Registers."

3.8.1.3.1 Stepping Revision ID (SRID)

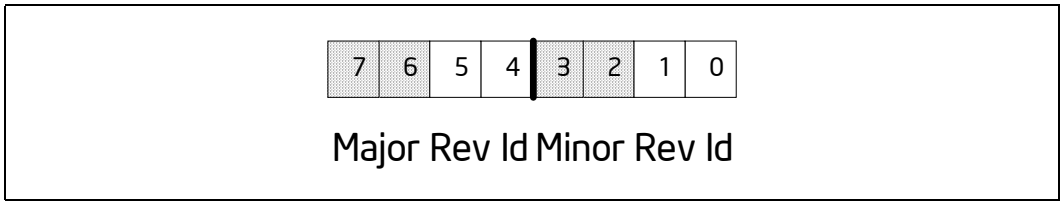
The SRID is a 4-bit hardwired value assigned by Intel, based on product's stepping. The SRID is not a directly addressable PCI register. The SRID value is reflected through the RID register when appropriately addressed. The 4 bits of the SRID are reflected as the two least significant bits of the major and minor revision field respectively. See Figure 11, "Intel® 5100 Memory Controller Hub Chipset Implementation of SRID and CRID Registers."

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 11, "Intel® 5100 Memory Controller Hub Chipset Implementation of SRID and CRID Registers."

Figure 11. Intel® 5100 Memory Controller Hub Chipset 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[1]: 0, 2-7
Function: 0
Offset: 09h
Device: 16
Function: 0, 1, 2
Offset: 09h
Device: 21, 22
Function: 0
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 04h, indicating "PCI-to-PCI Bridge" For all other Devices: 0, 16, 19, 21, 22 default is 00h, indicating "Host Bridge". See footnote 1, for DMA Engine device CCR.

Device[1]:	**0, 2-7**
Function:	**0**
Offset:	**09h**
Device:	**16**
Function:	**0, 1, 2**
Offset:	**09h**
Device:	**21, 22**
Function:	**0**
Offset:	**09h**

Bit	Attr	Default	Description
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.

1. The DMA Engine CCR for device 8 is defined separately in Section 3.11.3, "CCR: Class Code Register."

3.8.1.5 HDR - Header Type Register

This register identifies the header layout of the configuration space.

Device:	**0, 2-7, 8**
Function:	**0**
Offset:	**0Eh**
Device:	**16**
Function:	**0, 1, 2**
Offset:	**0Eh**
Device:	**21, 22**
Function:	**0**
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, 16, 19, 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**
Function:	**0**
Offset:	**2Ch**
Device:	**16**
Function:	**0, 1, 2**
Offset:	**2Ch**
Device:	**21, 22**
Function:	**0**
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.

3.8.2 SID - Subsystem Identity

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

Device:	**0, 8**
Function:	**0**
Offset:	**2Eh**
Device:	**16**
Function:	**0, 1, 2**
Offset:	**2Eh**
Device:	**21, 22**
Function:	**0**
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.

3.8.3 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 LOCK MEMCONFIG command, see D_LCK bit Section 3.8.3.8, "SMRAMC - System Management RAM Control Register." Routing to particular ports of a given interface type are defined by the following registers.

Table 50. Address Mapping Registers

Interface Type	Address Routing Registers
Memory	MIR, AMIR, PAM, SMRAM, EXSMRC, EXSMRAMC, TOLM, EXSMRTOP
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, PEXCMD (devices 2-7)
ESI	Subtractive decode[1] (device 0)

1. Any request not falling in the above ranges will be subtractively decoded and sent to ICH9R via the ESI

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

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

3.8.3.1 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 (ICH9R) 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 (ICH9R) 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

3.8.3.2 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

Device: 16 Function: 0 Offset: 5Ah			
Bit	**Attr**	**Default**	**Description**
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

3.8.3.3 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 FFFFh.

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

3.8.3.4 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

3.8.3.5 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

3.8.3.6 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

3.8.3.7 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: 0E C000h - 0E FFFFh Attribute Register** This field controls the steering of read and write cycles that address the BIOS area from 0E C000h - 0E FFFFh. 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: 0E 8000h - 0E BFFFh Attribute Register** This field controls the steering of read and write cycles that address the BIOS area from 0E 8000h - 0E BFFFh. 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

3.8.3.8 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_SMRAM_EN, ESMMTOP, TSEG_SZ and TSEG_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.

3.8.3.9 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 (i.e., 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.

Device: 16 Function: 0 Offset: 62h			
Bit	**Attr**	**Default**	**Description**
5:4	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 Appendix , "Extended SMRAM Space (TSEG)" for more details. Once D_LCK is set, this bit becomes read only.
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.

3.8.3.10 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
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 4 GB (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, "Extended SMRAM Space (TSEG)" as follows: ESMMTOP-TSEG_SZ <= Address < ESMMTOP TSEG_SZ can be 512 kB, 1 MB, 2 MB, or 4 MB, 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.

3.8.3.11 EXSMRAMC - Expansion System Management RAM Control Register

Device: 16 Function: 0 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, "Inbound Transactions" 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, etc., are included with the PCI Express* registers described in this chapter.

3.8.3.12 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.
11:0	RV	0h	Reserved

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	0h	*Reserved*

Device: 16 Function: 0 Offset: 6Eh			
Bit	**Attr**	**Default**	**Description**
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:DDR 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.
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 Section 2.10, "Reset Requirements." 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. This bit is self clearing.
13	RWST	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	Reserved
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 DDR channels into self-refresh mode in response to assertion 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. This bit is self clearing.
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 *RS - Intel® Core™ Microarchitecture, Intel® Pentium® 4, and Intel® Xeon® Processor External HW Spec*
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*

Device: 16 Function: 0 Offset: 42h			
Bit	**Attr**	**Default**	**Description**
11:0	RWST	7FFh	**DCRT: ESI CPU Reset Done Ack Determinism Timer** This field provides the determinism timer threshold for the Intel® 5100 MCH Chipset 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 ESI 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 ESI 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® 5100 MCH Chipset. 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
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 mapped 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.
2:0	RV	0	*Reserved*

3.8.6.2 FSBC1: Processor Bus Controller

This register allows control of the FSB1 Processor Bus.

Device: 16 Function: 0 Offset: 78h			
Bit	**Attr**	**Default**	**Description**
31	RV	0	*Reserved*
30	RW	0	**DisFSB1: Disable FSB1** 0: Do not disable the FSB1 (default) 1: Disables the FSB1. Disables all logic and Vterm circuits associated with FSB1 and must be used in conjunction with PROCENABLE.FSBEN.
29:0	RV	0FFA800Fh	*Reserved*

3.8.6.3 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 the Intel® 5100 MCH Chipset 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 the Intel® 5100 MCH Chipset has seen Ab[30] asserted, indicating there is more than 1 core in a processor.

3.8.6.4 XTPR[15:0] - External Task Priority Register

These registers control redirectable interrupt priority for xAPIC agents connected to the MCH. Up to eight agents on each bus are supported. These agents may be two dual core processor sockets each with four threads (two threads per core). The contents of

these registers are modified by the xTPR_Update transaction on the processor bus. Index into the XTPR registers is defined by Table 51. Index 1 is set if Ab[30] or Ab[22] is set. This corresponds to the most significant logical thread ID or the most significant Bus Agent ID, whichever is present. Section 5.4.3, "Interrupt Redirection" describes more details on this register.

Table 51. 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. Whenever this register is updated, the "CLUSTER" bit in the register is also updated.

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]). ***Note:*** 0: flat 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 ICH9R. 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 (e.g., 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, PCI Hot Plug* etc.

Table 52. Accessibility of the Intel® 5100 Memory Controller Hub Chipset PCI Express* Device

PCI Express* Port	Device	x8	Registers may be accessed if:
7	7	possible combination	Port 7 is connected to a x4 device
6	6		Port 6 is connected to a x4 or x8 device
5	5	possible combination	Port 5 is connected to a 4x device
4	4		Port 4 is connected to a x4, x8 or x16 device
3	3	possible combination	Port 3 is connected to a 4x device
2	2		Port 2 is connected to a x4 or x8 device
0	0	ESI - Not combinable	Port0 is connected to a x4 ICH9R port through ESI and cannot be combined with any other port

Figure 12, "PCI Express* Configuration Space" 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® 5100 MCH Chipset, the supported capabilities are:
  - Message Signaled Interrupts
  - PCI 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 Signaling 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 PCI Hot Plug* Controller and Power Management capabilities.

Figure 12. PCI Express* Configuration Space

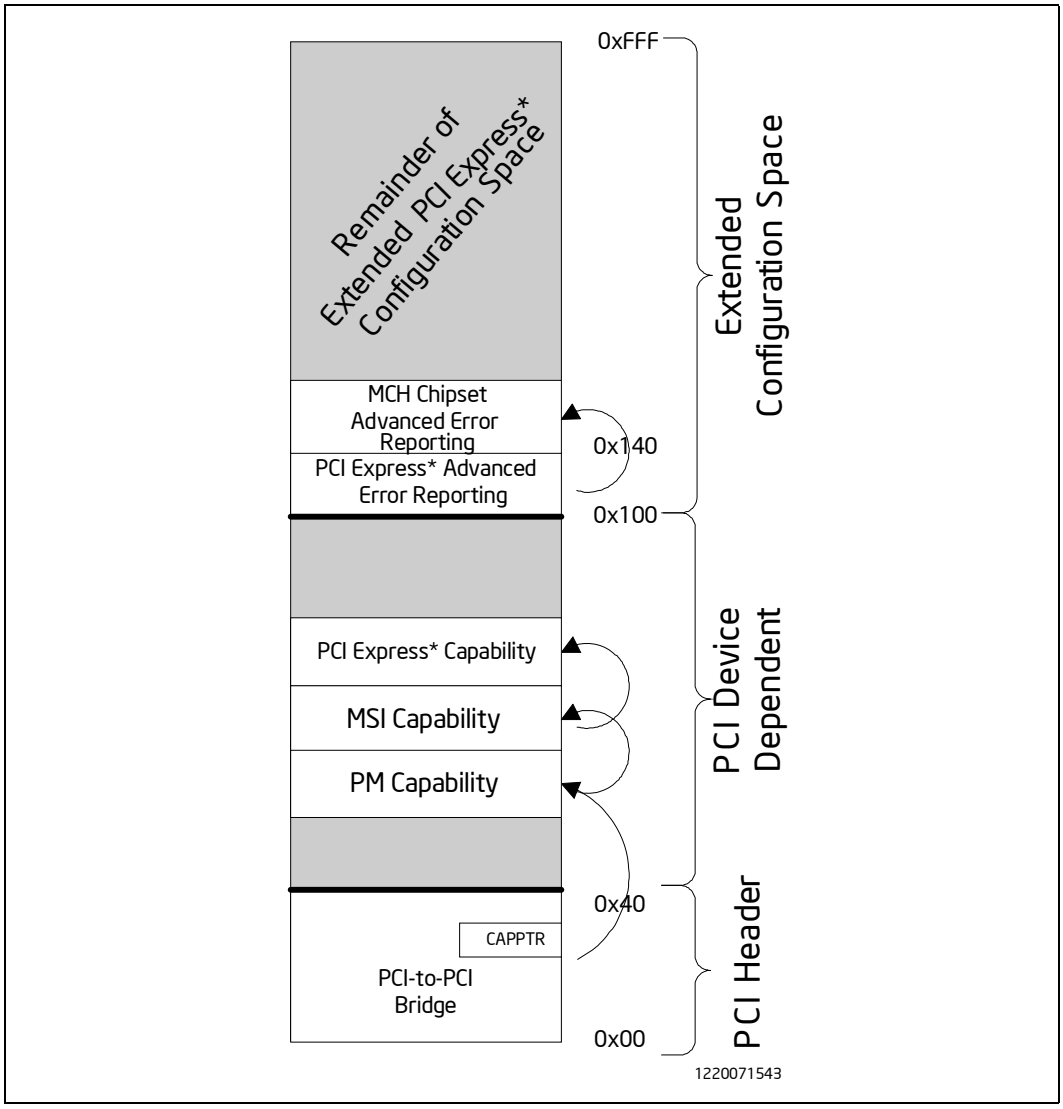


Figure 12, "PCI Express* Configuration Space" shows the configuration register offset addresses for each of the PCI Express* ports as defined in the *PCI Express* Base Specification*, Rev. 1.0a. It is also compatible with the standard PCI 2.3 capability structure and comprises of a linked list where 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 100h 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 PEXCMD[7:2,0]- Command Register

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

Device: 7-2, 0
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 to the ESI port to external interrupt controller where an interrupt can be generated to the CPU. This bit does not affect the ability of the Intel® 5100 MCH Chipset to route interrupt messages received at the PCI Express* port. However, this bit controls the generation of legacy interrupts to the ESI port for PCI Express* errors detected internally in this port (e.g., Malformed TLP, CRC error, completion timeout etc.) or when receiving root port error messages or interrupts due to HP/PM events generated in legacy mode within the Intel® 5100 MCH Chipset. Refer to the INTP register in Section 3.8.8.27, "INTP[7:2,0] - Interrupt Pin Register" for interrupt routing to ESI port. 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® 5100 MCH Chipset is enabled to send fatal/non-fatal errors. 0: The Intel® 5100 MCH Chipset is disabled from generating fatal/non-fatal errors. The errors are also enabled by the PEXDEVCTRL register in Section 3.8.11.4, "PEXDEVCTRL[7:2,0] - PCI Express* Device Control Register." In addition, for Type 1 configuration space header devices, e.g., Virtual PCI-to-PCI bridge), this bit, when set, enables transmission of ERR_NONFATAL and ERR_FATAL error messages[1] forwarded from the PCI Express* interface. This bit does not affect the transmission of forwarded ERR_COR messages. Refer to the Intel® 5100 MCH Chipset 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: 7-2, 0 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* 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 PCI Hot Plug*, device configuration. When the CPURESET# signal is asserted during a power good or hard reset and after the ESI completes its training, the LPC device in the ICH9R (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® 5100 MCH Chipset's PCI Express* port using the above rules since BME is reset. However, in general, it is illegal for an I/O device to issue inbound requests until the CPURESET# has been deasserted to prevent any possible malfunction in the Intel® 5100 MCH Chipset 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 the ICH9R. For port 0, this bit is hardwired to 0 since the ESI is not a PCI-to-PCI 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 the ICH9R. For port 0, this bit is hardwired to 0 since the ESI is not a PCI-to-PCI bridge.

1. 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, "BCTRL[7:2] - Bridge Control Register."

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

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

Device: 7-2, 0
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® 5100 MCH Chipset.
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, e.g., 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)
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.

Device: 7-2, 0 Function: 0 Offset: 06h			
Bit	**Attr**	**Default**	**Description**
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: 7-2, 0 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: 7-2, 0 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® 5100 MCH Chipset.

Device: 7-2, 0 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® 5100 MCH Chipset 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 primary side (MCH) of the PCI Express* port.

Device: 2-3, 4-7
Function: 0
Offset: 18h

Bit	Attr	Default	Description
7:0	RO	00h	**PBUSNUM: 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-to-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-to-PCI bridge. This number is programmed by the PCI configuration software to allow mapping of configuration cycles to devices connected to PCI Express*.

Device: 2-3, 4-7
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-to-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: 2-3, 4-7
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: 2-3, 4-7
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, "IOLIM[7:2] - I/O Limit Register") 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, "Special I/O Addresses" and Section 4.5.2, "Outbound I/O Access."

Device: 2-3, 4-7
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: 2-3, 4-7
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.

Device: 2-3, 4-7 Function: 0 Offset: 1Dh			
Bit	**Attr**	**Default**	**Description**
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 (i.e., PCI Express* side) of the "virtual" PCI-to-PCI bridge embedded within MCH.

Device: 2-3, 4-7 Function: 0 Offset: 1Eh			
Bit	**Attr**	**Default**	**Description**
15	RWC	0	**SDPE: Detected Parity Error** This bit is set by the Intel® 5100 MCH Chipset 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**). This corresponds to IO4 as defined in Table 123, "Intel® 5100 Memory Controller Hub Chipset Error List."
14	RWC	0	**SRSE: Received System Error** This bit is set by the MCH when it receives a ERR_FATAL or ERR_NONFATAL message. See Section 3.8.8.28, "BCTRL[7:2] - Bridge Control Register." (Note that BCTRL.BCSERRE is not a gating item for the recording of this error on the secondary side.)
13	RWC	0	**SRMAS: 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.
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 Section 3.8.8.28, "BCTRL[7:2] - Bridge Control Register" 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 53, "Intel® 5100 Memory Controller Hub Chipset PEXSTS and SECSTS Master/Data Parity Error RAS Handling" for details on the data parity error handling matrix in the Intel® 5100 MCH Chipset.

Device: 2-3, 4-7 Function: 0 Offset: 1Eh			
Bit	**Attr**	**Default**	**Description**
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 53. Intel® 5100 Memory Controller Hub Chipset PEXSTS and SECSTS Master/Data Parity Error RAS Handling

Register Name	OB Post	OB Compl	IN Post	IB Compl
PEXSTS[15].DPE[1]	yes	yes	no	no
PEXSTS[8].MDPERR	no	yes	no	no
SECSTS[15].SDPE	no	no	yes	yes
SECSTS[8].SMDPERR	no	no	no	yes

1. In general, the DPE field is the superset of the MDPERR from a virtual PCI-to-PCI bridge perspective but there may be cases where a PEXSTS[8].MDPERR may not be logged in the PEXSTS[15].DPE field in the Intel® 5100 MCH Chipset 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, "Main Memory Region," Section 4.4.2, "Address Disposition for Processor," and Section 4.4.3, "Inbound Transactions" for further details on address mapping.

Device: 2-3, 4-7 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.

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, i.e., 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: 2-3, 4-7
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: 2-3, 4-7 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 – 64-bit 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 1h, 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: 2-3, 4-7 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: 2-3, 4-7
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: 2-3, 4-7
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: 7-2, 0
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-to-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 read only register and is provided for backwards compatibility.

Device: 7-2, 0
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: 7-2, 0
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. 01h: Generate INTA 02h: Generate INTB 03h: Generate INTC 04h: 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 PEXCMD register that are specific to PCI-to-PCI bridges. The BCTRL provides additional control for the secondary interface (i.e., PCI Express*) as well as some bits that affect the overall behavior of the "virtual" PCI-to-PCI bridge embedded within the MCH, e.g., VGA compatible address range mapping.

Device: 7-2
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.

Device: 7-2 Function: 0 Offset: 3Eh			
Bit	**Attr**	**Default**	**Description**
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® 5100 MCH Chipset 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 2 ms 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 timeout has expired, the Intel® 5100 MCH Chipset 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.
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.

Device: 7-2 Function: 0 Offset: 3Eh			
Bit	**Attr**	**Default**	**Description**
2	RW	0	**ISAEN: ISA Enable** Modifies the response by the Intel® 5100 MCH Chipset 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® 5100 MCH Chipset will not forward to PCI Express* any I/O transactions addressing the last 768 bytes in each 1 kB block even if the addresses are within the range defined by the IOBASE and IOLIM registers. See Section 4.5.2, "Outbound I/O Access." 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® 5100 MCH Chipset will use the PEWIDTH[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® 5100 MCH Chipset.

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® 5100 MCH Chipset 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*

Device: 0 Function: 0 Offset: 40h			
Bit	**Attr**	**Default**	**Description**
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, i.e., link strap or soft initialization of link widths. 0: IOU clusters trained the links using the PEWIDTH[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 54, "GIO Port Mode Selection."
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 54, "GIO Port Mode Selection."
0	RWST	0	**LWOEN: Link Width over-ride Enable** 0: Disables software from setting the PCI Express* link width through this register and the Link width is controlled by the external pins PEWIDTH[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 (PEWIDTH[3:0] and use the described table for configuring the link width. The values will take effect after a hard reset.

Table 54. GIO Port Mode Selection (Sheet 1 of 2)

GIO Port (IOU0)					GIO Port (IOU1)				
GPMNXT0[2:0] (IOU0)	**Port0 (ESI)**	**Port1 (RSVD)**	**Port2**	**Port3**	**GPMNXT1[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

Table 54. GIO Port Mode Selection (Sheet 2 of 2)

GIO Port (IOU0)					GIO Port (IOU1)				
GPMNXT0[2:0] (IOU0)	Port0 (ESI)	Port1 (RSVD)	Port2	Port3	GPMNXT1[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® 5100 MCH Chipset. 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® 5100 MCH 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.
			0: Disable DMA Engine configuration accesses to Device 8, Fn 0 and Fn 1. The Intel® 5100 MCH Chipset will master abort requests to the DMA Engine configuration/memory mapped space.

3.8.8.31 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*

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

Bit	Attr	Default	Description
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 the Intel® 5100 MCH Chipset 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 the Intel® 5100 MCH Chipset 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® 5100 MCH Chipset. 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, the PEXCTRL.TIMEOUT_ENABLE (bit 22) has to be set. ***Note:*** In the MCH, the IOU will log a completion timeout 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 (i.e., either timeout is disabled). However, it does not affect the functionality and the config transaction will be outstanding indefinitely till a 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® 5100 MCH Chipset 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:*** In the MCH, the IOU will log a completion timeout error (IO6) for any outstanding non-configuration transaction that crosses the counter limit even if this register field is cleared (i.e., timeout is disabled). However, it does not affect the functionality and the non config-transaction will be outstanding indefinitely till a 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: • 4 DW 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 256 bytes return and **COALESCE_EN** = 1. 0: Up to 128 bytes return if COALESCE_EN = 1 ***Note:*** It is recommended that this field should not be set to 1 (256 bytes completion combining).
11	RW	0	**COALESCE_FORCE**: Force coalescing of accesses. When 1, forces the Intel® 5100 MCH Chipset 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 recommended that COALESCE_FORCE should not be set to '1.'
10	RW	0	**COALESCE_EN**: Read completion coalescing enable When 1, enables read return of >64 bytes. 1: Returns of >64 bytes enabled. (See **Max_rdcmp_lmt_EN above)**. 0: Returns are 64 bytes 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 bytes completion combining
9:2	RV	21h	*Reserved*
9	RW	0	**PMEGPEEN**: PME GPE Enable 1: Enables Assert_PMEGPE (Deassert_PMEGPE) messages to be sent over the ESI 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: PCI Hot Plug* GPE Enable** 1: Enables Assert_HPGPE (Deassert_HPGPE) messages to be sent from the root complex for PCI Hot Plug* events. 0: Disables Assert_HPGPE (Deassert_HPGPE) messages for PCI 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] = I/O Port defines the 8-bit I/O port that is used for routing power, attention, PCI Hot Plug*, presence, MRL and other events defined in Section 3.8.11.10, "PEXSLOTCTRL[7:2, 0] - PCI Express* Slot Control Register."
2	RW	1	**DIS_VPP: Disable VPP** The Intel® 5100 MCH Chipset 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® 5100 MCH Chipset will use this bit to decide whether end of interrupts (EOI) need to be sent to an APIC controller/bridge (e.g., 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:*** The Intel® 5100 MCH Chipset will block the EOIs from being sent to unconnected ports (hanging) or to slave ports (secondary). In general, EOI should be disabled for active ports that have a non-I/O APIC controller attached to them for performance considerations.
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® 5100 MCH Chipset 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 0600h. 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 ESI 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

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

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

Device: 7-2, 0 Function: 0 Offset: 4Ch			
Bit	**Attr**	**Default**	**Description**
7:1	RV	00h	*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.

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

Device: 7-2, 0 Function: 0 Offset: 4Dh			
Bit	**Attr**	**Default**	**Description**
7:5	RV	0	*Reserved*
4	if (port0) RV else {RWO} endif	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.
3:1	RV	0h	*Reserved* (See RV definitions Section 3.1, "Register Terminology".)
0	RW	0	**MSIRASERREN: MSI RAS Error Enable** 1: Enables MSI messages to be sent to the root complex for RAS error events on PCI Express ports. 0: Disables sending of MSI messages for RAS error events on PCI Express ports to the root complex. Note that for MSI RAS Error messages to be sent, both MSIRASERREN and MSICTRL[7:2] MSIEN bits defined in Section 3.8.10.3 have to be set.

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

3.8.8.34 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, PCI 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

Bit	Attr	Default	Description
31:18	RW	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 the Intel® 5100 MCH Chipset core running at 333 MHz, for SMBUS protocols, the maximum value recommended for this field is 744h (or 1860 decimal) to achieve a 20 ms timeout threshold (i.e., 20 ms =~ 10,752x1860) such that it provides headroom to the chipset for the global SMBUS timeout of 25 ms. ***Note:*** Example: With 744h as default and 333 MHz core clock, 1. Max timeout value: If bits 31:28 were set to 744h (1860d), the timeout delay is calculated as follows: 1860x7 (for the rollovers) x512 (lower 9 bits)x3.0 ns (for 333 MHz) = 1860x107542=19.998 ms=~20 ms 2. Min. timeout value: If bits 31:28 were set to 744h (1860d), the delay calculation would be like this: 1860x6 (too close to the limit, so missed full count for one rollover) x512 (lower 9 bits) x3.0 ns (for 333 MHz)= 17.141 ms=~17 ms
17:2	RV	1385h	*Reserved*
1	RWST	0	**PME_TURN_OFF: Send PME Turn Off Message** When set, the Intel® 5100 MCH Chipset will issue a PME Turn Off Message to all enabled PCI Express* ports excluding the ESI port. The Intel® 5100 MCH Chipset will clear this bit once the Message is sent. ***Note:*** In the Intel® 5100 MCH Chipset implementation, an end device that is D3 PM state and the Link being in L2 will not respond to any transaction to the device until it is woken up by the WAKE# signal in the platform. Under these conditions, if software sets the PME_Turn_OFF (bit 1) of this register, the Intel® 5100 MCH Chipset will not send the message until the Link is brought back into L0, i.e., PME_TURN_OFF bit will remain set until the message is dispatched. Furthermore, a surprise link Down error is logged.† Expected Usage: Software should not set this bit if the link is already in L2 prior.
0	RWC	0	**PME_TO_ACK: Received PME Timeout Acknowledge Message** The Intel® 5100 MCH Chipset 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 ESI will not generate a PME_TO_Ack. However, if a PME_TO_Ack is received at the Intel® 5100 MCH Chipset ESI port, it will be Master Aborted.

3.8.8.35 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, etc.). 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. For more information see Section 5.10, "Interrupt Swizzling."

3.8.9 PCI Express* Power Management Capability Structure

The Intel® 5100 MCH Chipset 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: 7-2, 0 Function: 0 Offset: 50h			
Bit	**Attr**	**Default**	**Description**
31:27	RO	11001	**PMES: PME Support** Identifies power states in the Intel® 5100 MCH Chipset which can send an "Assert_PMEGPE/Deassert_PMEGPE" message. Bits 31, 30 and 27 must be set to '1' for PCI-to-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® 5100 MCH Chipset)** XX1XXb - Assert_PMEGPE/Deassert_PMEGPE can be sent from D2 **(Not supported by Intel® 5100 MCH Chipset)** X1XXXb - Assert_PMEGPE/Deassert_PMEGPE can be sent from D3 hot **(Supported by Intel® 5100 MCH Chipset)** 1XXXXb - Assert_PMEGPE/Deassert_PMEGPE can be sent from D3 cold **(Not supported by Intel® 5100 MCH Chipset)**
26	RO	0	**D2S: D2 Support** The Intel® 5100 MCH Chipset does not support power management state D2.
25	RO	0	**D1S: D1 Support** The Intel® 5100 MCH Chipset 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*, Rev. 1.0a.
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: 7-2, 0 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 Bus Power Management Interface Specification*, revision 1.1, for details. This is not implemented in the Power Management capability for the Intel® 5100 MCH Chipset 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*.

Device: 7-2, 0 Function: 0 Offset: 54h			
Bit	**Attr**	**Default**	**Description**
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® 5100 MCH Chipset 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 be 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 ICH9R through the "Assert(Deassert)_PMEGPE"message. This has no effect on the Intel® 5100 MCH Chipset 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 (e.g., 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: 7-2, 0 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: 7-2, 0
Function: 0
Offset: 59h

Bit	Attr	Default	Description
7:0	RO	6Ch	**NXTPTR: Next Pointer** This field is set to 6Ch 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: 7-2, 0
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 (e.g., 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.
3:1	RO	000	**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 1 message (encoding of 000).
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® 5100 MCH Chipset to use MSI messages to request context specific service for events such as PCI Hot Plug*, PM and RAS.

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: 7-2, 0
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.

Device: 7-2, 0 Function: 0 Offset: 5Ch			
Bit	**Attr**	**Default**	**Description**
11:4	RW	00h	**AEXDSTID: Address Extended Destination ID** This field is not used by IA-32 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: 7-2, 0 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	000	**DM: Delivery Mode** 000: Fixed 001: Lowest Priority 010: SMI/HMI 011: *Reserved* 100: NMI 101: INIT 110: *Reserved* 111: ExtINT

Device: 7-2, 0 Function: 0 Offset: 60h			
Bit	**Attr**	**Default**	**Description**
7:0	RW	0h	**IV: Interrupt Vector** The interrupt vector (LSB) will be modified by the Intel® 5100 MCH Chipset to provide context sensitive interrupt information for different events that require attention from the processor, e.g., PCI Hot Plug*, Power Management and RAS error events. Depending on the number of Messages enabled by the processor in Section 3.8.10.3, "MSICTRL[7:2,0] - Message Control Register," Table 55, "IV Handling and Processing by Intel® 5100 Memory Controller Hub Chipset" illustrates the breakdown.

Table 55. IV Handling and Processing by Intel® 5100 Memory Controller Hub Chipset

Number of Messages Enabled by Software (MSICTRL.MMEN)	Events	IV[7:0]
1	All	xxxxxxxx[1]

1. 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 Table 55, "IV Handling and Processing by Intel® 5100 Memory Controller Hub Chipset" 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: 7-2, 0 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: 7-2, 0 Function: 0 Offset: 6Eh			
Bit	**Attr**	**Default**	**Description**
15:14	RV	0h	*Reserved.*

Device: 7-2, 0 Function: 0 Offset: 6Eh			
Bit	**Attr**	**Default**	**Description**
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 ESI port, this value should always be 0b since it is not PCI Hot Plug*-able and it is required for boot. The internal hardware allows the remaining ports to be slotted/PCI Hot Plug*-able, where 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 *PCI Express* Base Specification*, Rev. 1.0a since the PCI Express* port is a "root port" in the Intel® 5100 MCH Chipset.
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: 7-2, 0 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 the Intel® 5100 MCH Chipset 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 the Intel® 5100 MCH Chipset 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 the Intel® 5100 MCH Chipset. 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 the Intel® 5100 MCH Chipset. 1: *Reserved*

Device: 7-2, 0 Function: 0 Offset: 70h			
Bit	**Attr**	**Default**	**Description**
12	RO	0	**ABPD: Attention Button Present** This bit when set indicates that an Attention Button is implemented. 0: ABPD is disabled in the Intel® 5100 MCH Chipset. 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® 5100 MCH Chipset does not support Endpoint L1 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 the Intel® 5100 MCH Chipset does not support L0s implementation and for backup, this field is set to the maximum value.
5	RO	0	**ETFS: Extended Tag Field Supported** This field indicates the maximum supported size of the Tag field. 0: In the Intel® 5100 MCH Chipset, 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: 256 bytes max payload size Others - *Reserved* Note that the Intel® 5100 MCH Chipset only supports up to a maximum of 256 bytes payload (e.g., 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: 7-2, 0
Function: 0
Offset: 74h

Bit	Attr	Default	Description
15	RV	0h	*Reserved.*
14:12	RW	000	**MRRS: Max_Read_Request_Size** This field sets maximum Read Request size generated by the Intel® 5100 MCH Chipset. The PCI Express* port must not generate read requests with size exceeding the set value. 000: 128 bytes max read request size 001: 256 bytes max read request size 010: 512 bytes max read request size 011: 1024 bytes max read request size 100: 2048 bytes max read request size 101: 4096 bytes max read request size 110: Reserved 111: Reserved The MCH will not generate read requests larger than 64 bytes in general on the outbound side due to the internal microarchitecture (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® 5100 MCH Chipset never sets or modifies the "No snoop bit" in the received TLP even if ENNOSNP is enabled. For outbound traffic, the Intel® 5100 MCH Chipset does not need to snoop.
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, "PMCSR[7:2,0] - Power Management Control and Status Register."
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. The Intel® 5100 MCH Chipset does not implement this bit (Root complex) and is hardwired to 0
8	RO	0	**ETFEN: Extended Tag Field Enable** This bit enables the PCI Express* port to use an 8-bit Tag field as a requester. The Intel® 5100 MCH Chipset does not use this field (Root complex) and is hardwired to 0.

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

Bit	Attr	Default	Description
7:5	RW	000	**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® 5100 MCH Chipset 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 bytes max payload size 001: 256 bytes max payload size 010: 512 bytes max payload size 011: 1024 bytes max payload size 100: 2048 bytes max payload size 101: 4096 bytes max payload size others: Reserved ***Note:*** The MCH supports max payload sizes only up to 256 bytes. If Software programs a value that exceeds 256 bytes 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 #12 of the *Intel® 5100 Memory Controller Hub Chipset Specification Update*, order number 318385, read completion coalescing cannot be used if MPS=256 bytes is set by software. Read completion combining up to 128 bytes would work only if the MPS is set by software. Read completion combining up to 128 bytes would work only if the MPS is set to 128 bytes. See PEXCTRL.COALESCE_EN field.
4	RO	0	**ENRORD: Enable Relaxed Ordering** The Intel® 5100 MCH Chipset 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, "PEXRTCTRL[7:2,0] - PCI Express* Root Control Register."
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
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 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: 7-2, 0
Function: 0
Offset: 76h

Bit	Attr	Default	Description
15:6	RV	000h	*Reserved.*
5	RO	0	**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. 0: No Unsupported Request detected at the port 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
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: 7-2, 0
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	6h	*Reserved*
17:15	RO	111	**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 64 µs The Intel® 5100 MCH Chipset does not support L1 acceptable latency and is set to the maximum value for safety
14:12	RO	111	**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 the Intel® 5100 MCH Chipset 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® 5100 MCH Chipset does not initiate L0s active state Power Management but it does permit a downstream device from placing the link in L0s

Device: 7-2, 0 Function: 0 Offset: 78h			
Bit	**Attr**	**Default**	**Description**
9:4	RO	if (port 0,3,5,7) {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 56, "Maximum Link Width Default Value for Different PCI Express* Ports."
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 56. 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 56, "Maximum Link Width Default Value for Different PCI Express* Ports" 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: 7-2, 0 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 (e.g., logic analyzers) monitoring the Link time to achieve bit and Symbol lock before the Link enters the L0 or Recovery states and resumes communication.

Device: 7-2, 0 Function: 0 Offset: 7Ch			
Bit	**Attr**	**Default**	**Description**
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® 5100 MCH Chipset supports only 64-byte 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® 5100 MCH Chipset.

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 etc.

Device: 7-2, 0 Function: 0 Offset: 7Eh			
Bit	**Attr**	**Default**	**Description**
15:13	RV	0h	*Reserved*

Device: 7-2, 0 Function: 0 Offset: 7Eh			
Bit	**Attr**	**Default**	**Description**
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® 5100 MCH Chipset initializes this bit to '1' because the expected state of the platform is to have one clock source shared between the Intel® 5100 MCH Chipset 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, "PEXLNKCTRL[7:2,0] - PCI Express* Link Control Register" 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[1]** 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® 5100 MCH Chipset. Refer to Table 57, "Negotiated Link Width For Different PCI Express* Ports After Training" 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*

1. The NLNKWD field is set to a default value corresponding to x4 internally within the Intel® 5100 MCH Chipset. 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 57. 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[1]
4	x16	010000[2]

1. Ports 3, 5, and 7 report 000000 as appropriate.
2. 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.

Note: The ESI port, device 0, does not support PCI Hot Plug* capabilities.

Device: 7-2, 0
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 the Intel® 5100 MCH Chipset.
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
6:0	RV	0h	*Reserved*
6	RWO	0h	**HPC: PCI Hot Plug* Capable** This field defines PCI Hot Plug* support capabilities for the PCI Express* port 0: indicates that this slot is not capable of supporting PCI Hot Plug* operations. 1: indicates that this slot is capable of supporting PCI Hot Plug* operations
5	RO	0h	**HPS: PCI Hot Plug* Surprise** This field indicates that a device in this slot may be removed from the system without prior notification. 0: Indicates that PCI Hot Plug* surprise is not supported 1: Indicates that PCI Hot Plug* surprise is supported
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

Device: 7-2, 0 Function: 0 Offset: 80h			
Bit	**Attr**	**Default**	**Description**
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
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 PCI Hot Plug* and Power Management. Software issues a command to a PCI 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 (PEXSLOTSTS.CMDCMP) for the command completed event. The PCI Hot Plug* capabilities of this register are not supported for the Intel® 5100 MCH Chipset; default values to be maintained. 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.

Note: The ESI port, device 0, does not support PCI Hot Plug* capabilities.

Device: 0, 2-3, 4-7 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

Device: 0, 2-3, 4-7
Function: 0
Offset: 84h

Bit	Attr	Default	Description
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® 5100 MCH Chipset 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® 5100 MCH Chipset sends appropriate POWER_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® 5100 MCH Chipset, in addition. For PCI Express* modules with advanced form factor that incorporate LEDs and on-board decoding logic, the PCI Express* messages are interpreted directly (No virtual pins).
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® 5100 MCH Chipset drives 1.5 Hz square wave) 11: Off Default is set to 11b (OFF) When this field is written, the Intel® 5100 MCH Chipset 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® 5100 MCH Chipset, in addition. For PCI Express* modules with advanced form factor that incorporate LEDs and on-board decoding logic, the PCI Express* messages are interpreted directly (No virtual pins).
5	RW	0h	**HPINTEN: PCI Hot Plug* Interrupt Enable** This field enables the generation of PCI Hot Plug* interrupts and events in the PCI Express* port. 0: disables PCI Hot Plug* events and interrupts 1: enables PCI Hot Plug* events and interrupts
4	RW	0h	**CCIEN: Command Completed Interrupt Enable** Always write to '0', the Intel® 5100 MCH Chipset does not support PCI Hot Plug* capability. This field enables the generation of PCI Hot Plug* interrupts when a command is completed by the PCI Hot Plug* controller connected to the PCI Express* port 0: Disables PCI Hot Plug* interrupts on a command completion by a PCI Hot Plug* Controller 1: Enables PCI Hot Plug* interrupts on a command completion by a PCI Hot Plug* Controller
3	RW	0h	**PRSINTEN: Presence Detect Changed Enable** Always write to '0', the Intel® 5100 MCH Chipset does not support PCI Hot Plug* capability. This bit enables the generation of PCI Hot Plug* interrupts or wake messages via a presence detect changed event. 0: Disables generation of PCI Hot Plug* interrupts or wake messages when a presence detect changed event happens. 1: Enables generation of PCI Hot Plug* interrupts or wake messages when a presence detect changed event happens.

Device: 0, 2-3, 4-7 Function: 0 Offset: 84h			
Bit	**Attr**	**Default**	**Description**
2	RW	0h	**MRLINTEN: MRL Sensor Changed Enable** Always write to '0', the Intel® 5100 MCH Chipset does not support PCI Hot Plug* capability. This bit enables the generation of PCI Hot Plug* interrupts or wake messages via a MRL Sensor changed event. 0: Disables generation of PCI Hot Plug* interrupts or wake messages when an MRL Sensor changed event happens. 1: Enables generation of PCI Hot Plug* interrupts or wake messages when an MRL Sensor changed event happens.
1	RW	0h	**PWRINTEN:** Power Fault Detected Enable Always write to '0', the Intel® 5100 MCH Chipset does not support PCI Hot Plug* capability. This bit enables the generation of PCI Hot Plug* interrupts or wake messages via a power fault event. 0: Disables generation of PCI Hot Plug* interrupts or wake messages when a power fault event happens. 1: Enables generation of PCI Hot Plug* interrupts or wake messages when a power fault event happens.
0	RW	0h	**ATNINTEN: Attention Button Pressed Enable** Always write to '0', the Intel® 5100 MCH Chipset does not support PCI Hot Plug* capability. This bit enables the generation of PCI Hot Plug* interrupts or wake messages via an attention button pressed event. 0: Disables generation of PCI Hot Plug* interrupts or wake messages when the attention button is pressed. 1: Enables generation of PCI 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 PCI Hot Plug* and Power Management.

Device: 0, 2-3, 4-7 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	**CMDCMP: Command Completed** This bit is set by the Intel® 5100 MCH Chipset when the PCI 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.

Device: 0, 2-3, 4-7 **Function:** 0 **Offset:** 86h			
Bit	**Attr**	**Default**	**Description**
3	RWC	0h	**PRSINT: Presence Detect Changed** This bit is set by the Intel® 5100 MCH Chipset 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® 5100 MCH Chipset 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® 5100 MCH Chipset 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® 5100 MCH Chipset when the attention button is pressed. It is subsequently cleared by software after the field has been read and processed.

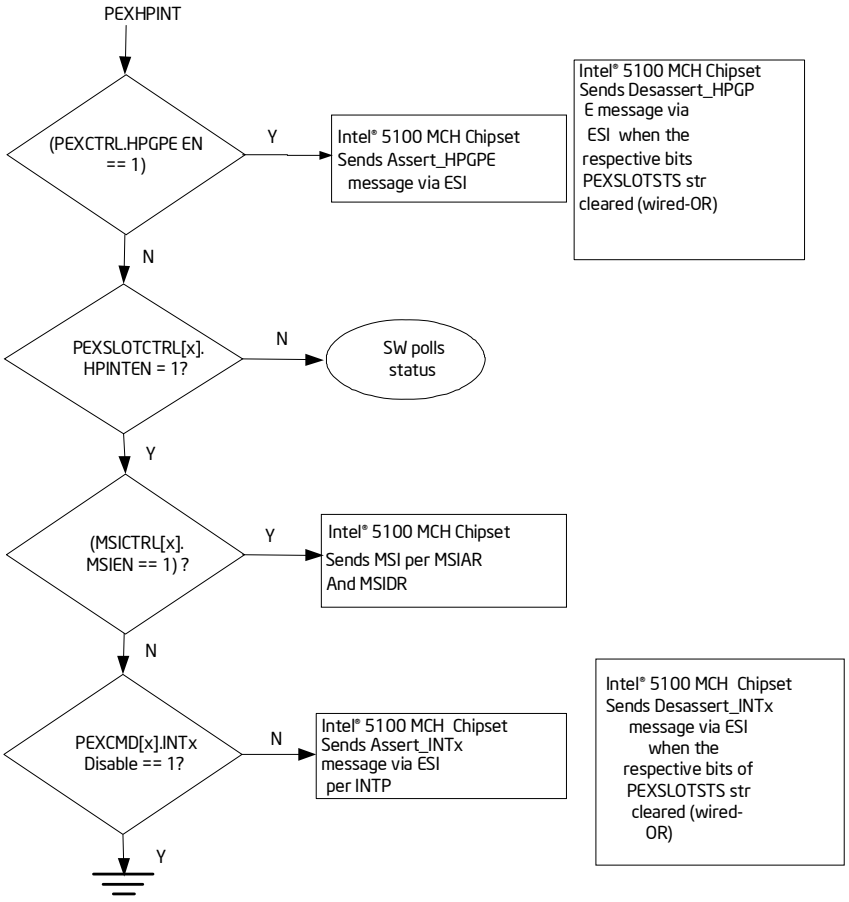
Note that the Assert_INTx/Assert_HPGPE message is sent to ESI port when any of the events defined in bits[4:0] (CMDCOMP, PRSINT, MRLSC, PWRINT, ABP) of the PEXSLOTSTS register are set provided the corresponding events in bits [4:0] of the Section 3.8.11.10 and HPINTEN are enabled. Software writes to 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 (e.g., ABP, PRSINT etc.) will cause an MSI message to be sent to the FSB for each occurrence, i.e., 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 13.

Note: The ESI port, device 0, does not support PCI Hot Plug* capabilities.

Figure 13. PCI 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: 7-2, 0
Function: 0
Offset: 88h

Bit	Attr	Default	Description
15:4	RV	0h	*Reserved.*

Device: 7-2, 0 Function: 0 Offset: 88h			
Bit	**Attr**	**Default**	**Description**
3	RW	0	**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, "PEXRTSTS[7:2,0] - PCI Express* Root Status Register," is set when this bit is set from a cleared state. 0: Disables interrupt generation for PME messages.
2	RW	0	**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	0	**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	0	**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: 7-2, 0 Function: 0 Offset: 8Ch			
Bit	**Attr**	**Default**	**Description**
31:18	RV	0h	*Reserved.*
17	RO	0	**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® 5100 MCH Chipset 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.

Device: 7-2, 0 Function: 0 Offset: 8Ch			
Bit	**Attr**	**Default**	**Description**
16	RWC	0	**PMESTATUS: PME Status**[1] 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.

1. 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 ICH9R 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, "MSICTRL[7:2,0] - Message Control Register."

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	0	**DL23R: Over-ride L23 Ready - Recommend setting this bit to 1.** 0: Wait for PME_Enter_L23 on all PCI Express* ports 1: Do not wait for PME_Enter_L23 on all PCI Express* ports
13:12	RV	0	*Reserved*
11	RWC	0	**PTE: PME_TO_Ack Time Expired** 0: Default mode, where the SC hardware broadcasts PME_turn_off message to all enabled PCI Express* ports. 1: Signal that time expiration has occurred when the PTOV field described below crosses the threshold in the Intel® 5100 MCH Chipset.
10:9	RW	0h	**PTOV: PME_TO_Ack Timeout Value** 00: 1 ms (default) 01: 10 ms 10: 50 ms 11: Reserved This register field provides the timer limit for the Intel® 5100 MCH Chipset to keep track of the elapsed time from sending "PME_Turn_off" to receiving a "PME_TO_Ack".
8:4	RV	10h	*Reserved*

Device: 0 Function: 0 Offset: D4h			
Bit	**Attr**	**Default**	**Description**
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 ICH9R 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 ICH9R. 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 ESI. 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: 7-2, 0 Function: 0 Offset: 100h			
Bit	**Attr**	**Default**	**Description**
31:20	RO	000h	**NCAPOFF: Next Capability Offset** This field points to the next Capability in extended configuration space.
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, "PEX_FAT_FERR[7:2,0] - PCI Express* First Fatal Error Register."

Device: 2-3, 4-7 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**

Device: 2-3, 4-7 Function: 0 Offset: 104h			
Bit	**Attr**	**Default**	**Description**
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 Timeout 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*, Rev. 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, "PEX_FAT_FERR[7:2,0] - PCI Express* First Fatal Error Register."

Device: 0 Function: 0 Offset: 104h			
Bit	**Attr**	**Default**	**Description**
31:22	RV	0h	*Reserved*
21	RWCST	0	**IO18Err: ESI Reset timeout**
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 Timeout 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**

Device: 0 Function: 0 Offset: 104h			
Bit	**Attr**	**Default**	**Description**
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*, Rev. 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: 7-2 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 Timeout 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**
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*, Rev. 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 timeout**

Device: 0 Function: 0 Offset: 108h			
Bit	**Attr**	**Default**	**Description**
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 Timeout 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**
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*, Rev. 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 timeout**
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 Timeout Severity**
13	RWST	1	**IO5Severity: Flow Control Protocol Error Severity**
12	RWST	0	**IO4Severity: Poisoned TLP Severity**
11:6	RV	0h	*Reserved*

Device: 0 Function: 0 Offset: 10Ch			
Bit	**Attr**	**Default**	**Description**
5	RWST	1	**IO19Severity: Surprise Link-down Severity**
4	RWST	1	**IO0Severity: Data Link Protocol Error Severity** (See Figure 3-17 in *PCI Express* Base Specification*, Rev. 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*, Rev. 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: 7-2 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 Timeout Severity**
13	RWST	1	**IO5Severity: Flow Control Protocol Error Severity**
12	RWST	0	**IO4Severity: Poisoned TLP Severity**
11:6	RV	0h	*Reserved*
5	RWST	1	**IO19Severity: Surprise Link-down Severity**
4	RWST	1	**IO0Severity: Data Link Protocol Error Severity** (See Figure 3-17 in *PCI Express* Base Specification*, Rev. 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*, Rev. 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, "CORERRMSK[7:2,0] - Correctable Error Mask"). These registers are discussed starting from Section 3.8.12.25, "PEX_NF_COR_FERR[7:2,0] - PCI Express* First Non-Fatal or Correctable Error Register."

Device: 7-2, 0
Function: 0
Offset: 110h

Bit	Attr	Default	Description
31:13	RV	0h	*Reserved*
12	RWCST	0	**IO16Err: Replay Timer Timeout 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 signaled. They are still logged in the CORERRSTS register.

Device: 7-2, 0
Function: 0
Offset: 114h

Bit	Attr	Default	Description
31:13	RV	0h	*Reserved*
12	RWST	0	**IO16Msk: Replay Timer Timeout Mask**
11:9	RV	0h	*Reserved*
8	RWST	0	**IO15Msk: Replay_Num Rollover Mask**
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: 7-2, 0 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** The Intel® 5100 MCH Chipset does not support ECRC.
6	RO	0	**ECRCGENEN: ECRC Generation Enable** The Intel® 5100 MCH Chipset does not generate ECRC.
5	RO	0	**ECRCGENCAP: ECRC Generation Capable** The Intel® 5100 MCH Chipset 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: 7-2, 0 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: 7-2, 0 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: 7-2, 0 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: 7-2, 0
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: 7-2, 0
Function: 0
Offset: 12Ch

Bit	Attr	Default	Description
31:3	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, "PEXRTCTRL[7:2,0] - PCI Express* Root Control Register" and the Root Error Command register defined in Section 3.8.12.15, "RPERRCMD[7:2,0] - Root Port Error Command."

Device: 7-2, 0
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.

Device: 7-2, 0 Function: 0 Offset: 130h			
Bit	**Attr**	**Default**	**Description**
26:7	RV	0h	Reserved
6	RWCST	0	**FAT_ERR_Rcvd:** Fatal Error Messages Received Set when one or more Fatal Uncorrectable error Messages[1] 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, i.e., log from the second 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, i.e., log the first error message
1	RWCST	0	**MULT_ERR_COR_RCVD:** Multiple Correctable Error Received Set when a correctable error message is received and ERR_CORR_RCVD is already set, i.e., log from the second 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, i.e., log the first error message

1. 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, "RPERRSTS[7:2,0] - Root Error Status Register." This register is updated regardless of the settings of the Root Control register defined in Section 3.8.11.12, "PEXRTCTRL[7:2,0] - PCI Express* Root Control Register" and the Root Error Command register defined in.Section 3.8.12.15, "RPERRCMD[7:2,0] - Root Port Error Command."

Device: 7-2, 0 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, i.e., 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, i.e., log ID of the first correctable error.

3.8.12.18 SCSPCAPID[7:2,0] - Intel® 5100 Memory Controller Hub Chipset-specific Capability ID

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

Device: 7-2, 0
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: 7-2, 0
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.
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	**IO18ESIRstDetMsk:** ESI Reset timeout
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 Timeout 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: 2-3, 4-7
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 Timeout Status
13	RW	0	**IO5DetMsk:** Flow Control Protocol Error Status
12	RW	0	**IO4DetMsk:** Poisoned TLP Status

Device: 2-3, 4-7 Function: 0 Offset: 148h			
Bit	**Attr**	**Default**	**Description**
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: 7-2, 0 Function: 0 Offset: 14Ch			
Bit	**Attr**	**Default**	**Description**
31:13	RV	0h	Reserved
12	RW	0	**IO16DetMsk:** Replay Timer Timeout 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: 7-2, 0 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, "UNCERRSEV[7:2] - Uncorrectable Error Severity."

Device: 7-2, 0
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 timeout
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
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, "UNCERRSEV[7:2] - Uncorrectable Error Severity."

Device: 7-2, 0
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)

Device: 7-2, 0 Function: 0 Offset: 158h			
Bit	**Attr**	**Default**	**Description**
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)
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: 7-2, 0 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 timeout
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

Device: 7-2, 0 Function: 0 Offset: 15Ch			
Bit	**Attr**	**Default**	**Description**
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: 7-2, 0 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)

Device: 7-2, 0 Function: 0 Offset: 160h			
Bit	**Attr**	**Default**	**Description**
0	RWCST	0	**Next_NFAT_COR_Err_IO0:** PEX - Data Link Layer Protocol Error (uncorrectable)

3.8.12.28 PEX_UNIT_FERR[7:2] - 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, e.g., VPP error due to a malfunctioning port on the SMBus that did not receive acknowledge due to a PCI 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: 2-3, 4-7 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, e.g., VPP error due to a malfunctioning port on the SMBus that did not receive acknowledge due to a PCI 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: 2-3, 4-7 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.

3.8.13 Error Registers

This section describes the registers that record the first and next errors, logging, detection masks, signaling 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® 5100 MCH Chipset 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:** FSB0 Fatal Error
27:24	RV	0h	Reserved
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® 5100 MCH Chipset Non-Fatal Error
14	RWCST	0	**Global_FERR_14:** DMA Engine Device Non Fatal Error
13	RWCST	0	**Global_FERR_13:** FSB1 Non-Fatal Error
12	RWCST	0	**Global_FERR_12:** FSB0 Non-Fatal Error
11:10	RV	0h	Reserved
9	RWCST	0	**Global_FERR_09:** DDR Channel 1 Non-Fatal Error
8	RWCST	0	**Global_FERR_08:** DDR Channel 0 Non-Fatal Error
7	RWCST	0	**Global_FERR_07:** PCI Express* Device 7 Non-Fatal Error
6	RWCST	0	**Global_FERR_06:** PCI Express* Device 6 Non-Fatal Error

Device: 16 Function: 2 Offset: 40h			
Bit	**Attr**	**Default**	**Description**
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:** FSB0 Fatal Error
27:24	RV	0h	Reserved
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

Device: 16 Function: 2 Offset: 44h			
Bit	**Attr**	**Default**	**Description**
15	RWCST	0	**Global_NERR_15:** Internal Intel® 5100 MCH Chipset 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:** FSB0 Non-Fatal Error
11:9	RV	0h	Reserved
8	RWCST	0	**Global_NERR_08:** DDR Channel 0 or 1 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	0h	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 first 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 first fatal error.

Device: 16
Function: 0
Offset: 483h, 183h

Bit	Attr	Default	Description
7:3	RV	0h	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
4:2	RV	0h	Reserved
1	RWST	1	**F2Msk:** Unsupported Processor Bus Transaction
0	RWST	1	**F1Msk:** Request/Address Parity Error

Note: For systems without parity, the BIOS will have to write to this register to mask the 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
4:2	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
4:2	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
4:2	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

Device: 16 Function: 0 Offset: 49Ah, 19Ah			
Bit	**Attr**	**Default**	**Description**
5	RWST	1	**F6Msk:** Data Parity Error
4:2	RV	0h	Reserved
1	RWST	1	**F2Msk:** Unsupported Processor Bus Transaction
0	RWST	1	**F1Msk:** Request/Address Parity Error

3.8.13.16 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**
7:3	RV	0h	*Reserved*
7:4	RV	0h	*Reserved*
3	RWCST	0	**B4Err:** Virtual Pin Port Error (VPP_PEX)
2	RWCST	0	**B3Err:** Coherency Violation Error for WEWB
1	RV	0	*Reserved*
0	RWCST	0	**B1Err:** DM Parity Error (Data Manager)

3.8.13.17 FERR_NF_INT - Internal First Non-Fatal Error Register

Device: 16 Function: 2 Offset: C1h			
Bit	**Attr**	**Default**	**Description**
7:1	RV	0h	*Reserved*
0	RWCST	0	**B5Err:** Single Address Map Error (COH)

3.8.13.18 NERR_FAT_INT - Internal Next Fatal Error Register

Device: 16 Function: 2 Offset: C2h			
Bit	**Attr**	**Default**	**Description**
7:3	RV	0h	*Reserved*
7:4	RV	0h	*Reserved*
3	RWCST	0	**B4Err:** Virtual Pin Port Error (VPP_PEX)
2	RWCST	0	**B3Err:** Coherency Violation Error (COH) for EWB
1	RV	0	*Reserved*

Device: 16 Function: 2 Offset: C2h			
Bit	**Attr**	**Default**	**Description**
0	RWCST	0	**B1Err:** DM Parity Error (DM)

3.8.13.19 NERR_NF_INT - Internal Next Non-Fatal Error Register

Device: 16 Function: 2 Offset: C3h			
Bit	**Attr**	**Default**	**Description**
7:1	RV	0h	Reserved
0	RWCST	0	**B5Err:** Address Map Error (COH)

3.8.13.20 NRECINT - Non Recoverable Internal Intel® 5100 Memory Controller Hub Chipset 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 3.8.13.16, "FERR_FAT_INT - Internal First Fatal Error Register." For debugging VPP errors in this register, e.g., 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, "PEX_UNIT_FERR[7:2] - PCI Express* First Unit Error Register" to determine the failing port.

Device: 16 Function: 2 Offset: C4h			
Bit	**Attr**	**Default**	**Description**
31:21	RV	0h	Reserved
20:13	ROST	0h	**DM** (Data Manager) **entry**
12:11	RV	00	Reserved
10:8	ROST	000	**Internal Block that detected the Failure** 000: Default no errors detected 001: VPP_PEX_PORT2-3 010: VPP_PEX_PORT4-7 011: VPP_MEM 100: COH 101: DM Others: *Reserved*
7	RV	0	Reserved
6:0	ROST	0h	**COH Entry of Failed Location**

3.8.13.21 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**
7:5	RV	7h	Reserved
4	RWST	1	**B5Msk:** Address Map Error
3	RV	1	*Reserved*
3	RWST	1	**B4Msk:** Virtual Pin Port Error
2	RWST	1	**B3Msk:** Coherency Violation Error for EWB
1	RV	1	*Reserved*
0	RWST	1	**B1Msk:** DM Parity Error

3.8.13.22 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**
7:5	RV	7h	Reserved
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	RV	1	*Reserved*
0	RWST	1	**B1Err2Msk:** DM Parity Error

3.8.13.23 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**
7:5	RV	7h	Reserved
4	RWST	1	**B5Err1Msk:** Address Map Error
3	RWST	1	**B4Err1Msk:** SMBus Virtual Pin Error
2	RWST	1	**B3Err1Msk:** Coherency Violation Error
1	RV	1	*Reserved*
0	RWST	1	**B1Err1Msk:** DM Parity Error

3.8.13.24 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
7:5	RV	7h	Reserved
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	RV	1	*Reserved*
0	RWST	1	**B1Err0Msk:** DM Parity Error

3.8.13.25 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 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
7:5	RV	7h	*Reserved*
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	RV	1	*Reserved*
0	RWST	1	**B1McErrMsk:** DM Parity Error

3.9 Memory Control Registers

3.9.1 General Registers

3.9.1.1 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*

Device: 16
Function: 1
Offset: 40h

Bit	Attr	Default	Description
30	RW	0	**RETRY: Retry Enable:** Retry uncorrectable ECC error (more than single bit error) `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	0h	*Reserved*
21	RW	0	**INITDONE: Initialization Complete.** This scratch bit communicates software state from the Intel® 5100 MCH Chipset to BIOS. BIOS sets this bit to 1 after initialization of the DRAM memory array is complete. This bit has no effect on the Intel® 5100 MCH Chipset operation.
20	RV	0	*Reserved*
19:18	RW	00	**PHT: Page Hit Threshold** Determines max allowable page hits before a page will be precharged. A page hit is a consecutive CAS access the same page (which has been left open). If there is no available page hit, then autoprecharge will occur. 00: 0 (no hit allowed, all accesses are with autoprecharge) 01: 1 10: 3 11: 7 (Up to 7 hits allowed => max 8 accesses to a page before it is closed) Note: This field alters the address interleaving scheme depending on whether it is 00 or not. Modifying this field will therefore invalidate the DRAM contents.
17:14	RV	0h	*Reserved*
13	RW	0	**FSMEN1: Channel 1 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 :** a) memory reads b) memory writes c) refreshes Not preserved by SAVCFG
12	RW	0	**FSMEN0: Channel 0 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 :** a) memory reads b) memory writes c) refreshes Not preserved by SAVCFG
11:9	RV	0h	*Reserved*
8	RW	0	**SCRBALGO: Enhanced Mode Algorithm for x8 uncorrectable error detection** 0: Normal mode where BADRAM is not used to flag bad devices. 1: Enhanced mode where BADRAM is used to flag bad devices.

Device: 16 Function: 1 Offset: 40h			
Bit	**Attr**	**Default**	**Description**
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.
5	RW	0	**ERRDETEN: Error Detection Enable** '1' = Northbound ECC checking enabled. '0' = Northbound ECC checking disabled Error logging and data poisoning are disabled when Northbound 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® 5100 MCH Chipset will stop the scrub cycle immediately.
3:0	RV	0h	*Reserved*

3.9.1.2 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	2h	**TO: Starvation Timeout** This timeout function is to ensure that any read request not issued within the specified time will be given the highest priority. A value of zero represents eight cycles. Each increment adds eight cycles. Maximum is 128 cycles.
27:0	RV	DB21915h	*Reserved*

3.9.1.3 MS: Memory Status Register

Miscellaneous status not reflected in other registers.

Device: 16 Function: 1 Offset: 44h			
Bit	**Attr**	**Default**	**Description**
31:16	RV	0000h	*Reserved*

Device: 16 Function: 1 Offset: 44h			
Bit	**Attr**	**Default**	**Description**
8	RO	0	**48GB_Mode: 48 GB mode operation status** 0: 32 GB Mode is enabled utilizing up to four ranks/channel 1: 48 GB Mode is enabled utilizing up to six ranks/channel B0: The value of the bit provides the strapped state of the 48GB_Mode input. The 48GB_Mode input is strapped High for 48 GB support over six ranks/channel. The 48GB_Mode input is strapped Low for 32 GB support over four ranks/channel. The BIOS can read this bit to determine the mode of operation of the Intel® 5100 Memory Controller Hub Chipset.
7:0	RV	00h	*Reserved*

3.9.1.4 MCDEF3: MCDEF Register 3

Device: 16 Function: 1 Offset: FCh			
Bit	**Attr**	**Default**	**Description**
31:0	RV	0h	*Reserved*

3.9.1.5 ERRPER: Error Period Prescaler

ERRPER is part of the RAS system for initiating sparing operations. Sparing will be kicked off when a rank is assumed to be bad (“failed”) because a preprogrammed number of correctable errors have been detected on that rank.

SER analysis determines a known expected correctable error rate and this is used to “leak” expected errors from the count of actual correctable errors that have been detected. This prevents a rank from being declared bad because of the natural expected SER rate. The ERRPER register is a prescaler for the SER rate counters.

The Error Period counter increments the rank counters (one such counter per rank) when it reaches the ERRPER value. The error period counter increments every 16 cycles. It is cleared on reset and wraps when it reaches this value. Table 58 shows the timing characteristics of this register.

Non-zero CERRCNT (detected correctable errors) counts are decremented when rank counters reach the programmed RANKTHRESHOLD values which should match the expected SER rate. for the rank size

Table 58. Timing Characteristics of ERRPER

Core Frequency	Per Increment	Maximum Period (increment period x ((2^32)-1))
333 MHz	48 ns	206.2 s
266 MHz	60 ns	257.7 s

Device: 16 Function: 1 Offset: 50h			
Bit	**Attr**	**Default**	**Description**
31:0	RW	0h	**THRESH: Global ERRPER increment threshold** A value of 0 prevents incrementing the error period counter, and therefore the rank counter, and thus prevents decrementing of CERRCNT. When the value in this register is reached, the rank counters are incremented by 1. For example a value of 1G (369AC9FFh) will mean the rank counters increment one every 16 X 10^9 cycles, where the 16 comes from the initial divide by 16.

3.9.1.6 MTR[1:0][5:0] - Memory Technology Registers

These registers define the organization of the DIMM's. There is one MTR for each rank. The parameters for these devices can be obtained by serial presence detect.

MTR[n][5:0] defines ranks [5:0] on channel[n].

This register must not be modified while servicing memory requests.

The following three settings are mutually exclusive:

- 65,536 rows (NUMROW = "11")
- 4,096 columns (NUMCOL = "10")

Device: 22, 21 Function: 0 Offset: 1B2h, 1B0h, 15Ah, 158h, 156h, 154h			
Bit	**Attr**	**Default**	**Description**
15:11	RV	00h	Reserved
10	RW	0	**PRESENT: Ranks are present** This bit is set if ranks are present and their technologies are compatible.
9	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 DRTA.TFAW configuration field.
8	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)
7	RV	0	*Reserved*
6	RW	0	**NUMBANK: Technology - Number of Banks** Defines the number of (real, not shadow) banks on these DIMMs '0' = four-banked '1' = eight-banked
5:4	RV	00	*Reserved*
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" = 65,536, 16 rows

Device: 22, 21 Function: 0 Offset: 1B2h, 1B0h, 15Ah, 158h, 156h, 154h			
Bit	**Attr**	**Default**	**Description**
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 (Not Supported) "11" = Reserved

3.9.1.7 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: 22, 21 Function: 0 Offset: 16Ch, 168h, 164h, 160h, 15Ch			
Bit	**Attr**	**Default**	**Description**
31:27	RV	000h	Reserved
26:16	RW	00h	**LIMIT** This field defines the highest address in the range. Memory requests participate in this DMIR range if LIMIT[i] > M[37:28] >= LIMIT[i-1]. For i = 0, LIMIT[i-1]=0.
15	RV	0	Reserved
14:12	RW	000	**RANK3** Defines which rank participates in WAY3. Only bits [1:0] are used.
11	RV	0	Reserved
10:8	RW	000	**RANK2** Defines which rank participates in WAY2. Only bits [1:0] are used.
7	RV	0	Reserved
6:4	RW	000	**RANK1** Defines which rank participates in WAY1. Only bits [1:0] are used.
3	RV	0h	Reserved
2:0	RW	000	**RANK0** Defines which rank participates in WAY0. Only bits [1:0] are used.

3.9.2 Memory Throttling Control Registers

The Intel® 5100 MCH Chipset employs activation based throttling where the number of activates to a given rank are monitored and possibly limited as required. There are two levels of throttling, low (normal operation) and high (activations throttling triggered), and the levels of activity permitted at both these levels are selected by the BIOS.

Note that throttling decreases performance and increases memory access latency in heavy traffic. Extremely low values (below 10h) will lead to dramatically increased latency for reads which could result in requestor timeouts, etc.

Note that throttling must always be disabled for RCVEN calibration.

3.9.2.1 GBLACT: Global Activation Throttle Register

This register contains the limit for Global Activation throttle control.

Device: 16 **Function:** 1 **Offset:** 60h			
Bit	**Attr**	**Default**	**Description**
7:0	RW	0h	**GBLACTLM: Global Activation Throttle Limit** This field controls the activation of Global throttling based on the number of activations sampled per rank on each channel. If the number of activations in the global throttling window exceeds the number indicated by the GBLACTLM filed in this register, then global throttling is started by setting the THRTSTS[1:0].GBLTHRT bit for the respective channel and the Global activation throttling logic to use the THRTHIGH register for throttling. The granularity of this field is 65536 or 16 activations. Refer to Table 59, "Global Activation Throttling as Function of Global Activation Throttling Limit (GBLACTM) Register Fields." If Software sets this value greater than 168, the chipset will cap the GBLACTLM field to 168.

Table 59. Global Activation Throttling as Function of Global Activation Throttling Limit (GBLACTM) Register Fields

GBLACT.GBLACTM Range (0...168)	Number of Activations
0	No Throttling (unlimited activations)
1	65536
2	131072
16	1048576
32	2097152
64	4194304
96	6291456
100	6553600
128	8388608
150	9830400
168	11010048 (100% BW)

3.9.2.2 THRTSTS[1:0]: Throttling Status Register

Records the global activation throttle status and internal thermal throttling value for channels 0/1

Device: 16 Function: 1 Offset: 6Ah, 68h			
Bit	**Attr**	**Default**	**Description**
15:9	RV	0h	*Reserved*
8	RO	0h	**GBLTHRT: Global Activation Throttle**[1] This field is set by the Intel® 5100 MCH Chipset to indicate the start of the Global Activation throttling based on the number of activations sampled in the global window. If the number of activations in the global window (16384x1344 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 windows in a row have gone by without any rank 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 throttling algorithm. 0: No throttling (unlimited activation) 1: 4 activations per rank per activation window. 2: 8 activations per rank per activation window 168: 672 activations per rank per activation window This field will be set by the Intel® 5100 MCH Chipset and the value of this field will vary between THRTLOW and THRTHIGH registers based on the throttling.

1. The Intel® 5100 MCH Chipset will use an internal signal called GBLTHRT* from its combinatorial cluster for controlling open loop throttling.

3.9.2.3 THRTHIGH: Thermal Throttle High Register

Device: 16 Function: 1 Offset: 65h			
Bit	**Attr**	**Default**	**Description**
7:0	RW	0h	**THRTHIGHLM: Thermal Throttle High Limit** A throttling level that is applied when the THRTSTS.GBLTHRT* bit is set by the Global Throttling Window logic. The maximum value this field can be initialized by software is 168 (decimal). This corresponds to 672 activations per throttling window and gives 100% BW. The granularity of this field is 4 activations. 0: No throttling (unlimited activation) 1: 4 activations per rank per activation window 2: 8 activations per rank per activation window 168: 672 activations per rank per activation window If Software sets this value greater than 168, the chipset will cap the THRTHIGHLM field to 168. This field should be less than or equal to the THRTLOW.THRTLOWLM.

3.9.2.4 THRTLOW: Thermal Throttle 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 THRTSTS.GBLTHRT* bit is not set by the Global Throttling Window logic. ***Note:*** The GBLTHRT* is an internal signal from the Intel® 5100 MCH Chipset open loop combinatorial cluster before it is latched in the THRTSTS.GLTHRT register. This will prevent any stale/delayed information from being used for the open loop throttling logic. The maximum value this field can be initialized by software is 168 (decimal). This corresponds to 672 activations per throttling window and gives 100% BW. The granularity of this field is 4 activations. 0: No throttling (unlimited activation) 1: 4 activations per rank per activation window 2: 8 activations per rank per activation window 168: 672 activations per rank per activation window If Software sets this value greater than 168, the chipset will cap the THRTLOWLM field to 168.

The two thermal throttle registers set the limit for activations to a given rank during a thermal throttle activation window (1344 cycles). Should the limit ("HIGH" or "LOW" as selected by GBLTHRT) be exceeded then further activations will be prevented for the duration of the window.

3.9.3 Memory Gearing Registers

3.9.3.1 DDRFRQ: DDR Frequency Ratio

This register specifies the CORE:DDR frequency ratio. The "other" (Section 3.9.3.2, "MEMTOHOSTGRCFG0: MEM to Host Gear Ratio Configuration 0" through Section 3.9.3.7, "HOSTTOMEMGRCFG1: Host to MEM Gear Ratio Configuration 1") gearing configuration registers must be set prior to changing the "DDRFRQ" field of this register. This register must be written once after a hard reset.

Device: 16 Function: 1 Offset: 56h			
Bit	**Attr**	**Default**	**Description**
7:6	RV	0	*Reserved*
5:4	ROST	01	**CURDDRFRQ: CORE:DDR Frequency Ratio (current ratio)** '00' = **1:1**. BUSCLK=266 MHz, DDR=533 MHz. '01' = **1:1**. BUSCLK=333 MHz, DDR=667 MHz. '10' = **4:5**. BUSCLK=266 MHz, DDR=667 MHz. '11' = **5:4**. BUSCLK=333 MHz, DDR=533 MHz.
3	RWST	1	**COREFREQ: CORE Frequency** This frequency ratio tells the SPD master which divider ratio to employ in order to operate at 100 kHz. '1' = Core is operating at 333 MHz. SPD Divider ratio = 3,334 '0' = Core is operating at 266 MHz. SPD Divider ratio = 2,667 BIOS programming is optional. If BIOS doesn't program this bit, then if the core is operating at 266 MHz, the SPD link will operate at 80 kHz.

Device: 16 Function: 1 Offset: 56h			
Bit	**Attr**	**Default**	**Description**
2:0	RWST	001	**DDRFRQ: CORE:DDR Frequency Ratio (write new ratio to these bits)** '000' = **1:1**. BUSCLK=266 MHz, DDR=533 MHz. '001' = **1:1**. BUSCLK=333 MHz, DDR=667 MHz. '010' = **4:5**. BUSCLK=266 MHz, DDR=667 MHz. '011' = **5:4**. BUSCLK=333 MHz, DDR=533 MHz. '1xx' = Reserved Writing this field will set the relationship between the CORE- and MEM-domain.

3.9.3.2 MEMTOHOSTGRCFG0: MEM to Host Gear Ratio Configuration 0

This register consists of 8 nibbles of mux select data for the proper selection of gearing behavior in the MC. This is the first of two registers to control the behavior for the DRAM to host (Northbound) data flow.

Device: 16 Function: 1 Offset: 160h			
Bit	**Attr**	**Default**	**Description**
31:0	RWST	11111111h	MEMHSTGRMUX: MEM to Host Clock Gearing mux selector. Eight nibbles of mux select for memory/DDR2 to MC/core geared clock boundary crossing phase enables. Refer to Table 60, "MEM to Host Gear Ratio Mux for MEMTOHOSTGRCFG0" for the programming details.

Table 60. MEM to Host Gear Ratio Mux for MEMTOHOSTGRCFG0

FSB: Memory Frequency	Gear Ratio	Option	Value
333:333 267:267	1:1	only	11111111h
333:267	5:4	conservative	00010320h
267:333	4:5	conservative	00005430h

3.9.3.3 MEMTOHOSTGRCFG1: MEM to Host Gear Ratio Configuration 1

This register consists of 8 nibbles of mux select data for the proper selection of gearing behavior in the MC for the 1:1 and 4:5 modes. This is the second register for MEM to Host gearing control.

Device: 16 Function: 1 Offset: 164h			
Bit	**Attr**	**Default**	**Description**
31:0	RWST	00000000h	**MEMHSTGRMUX: MEM to Host Clock Gearing mux selector.** Eight nibbles of mux select for memory/DDR2 to MC/core geared clock boundary crossing phase enables. Refer to Table 61, "MEM to Host Gear Ratio Mux for MEMTOHOSTGRCFG1" for the programming details.

Table 61. MEM to Host Gear Ratio Mux for MEMTOHOSTGRCFG1

FSB: Memory Frequency	Gear Ratio	Option	Value
333:333 267:267	1:1	only	00000000h
333:267	5:4	conservative	00040000h
267:333	4:5	conservative	00001020h

3.9.3.4 MEMNDGRCFG0: MEM Next Data Gear Ratio Configuration 0

This register consists of 8 nibbles of mux select data for the proper selection of gearing behavior in the MC. This is the first of two registers to control the behavior for the DRAM to host (Northbound) data flow for the "data_next" signaling.

Device: 16
Function: 1
Offset: 168h

Bit	Attr	Default	Description
31:0	RWST	11111111h	**MEMNDGRMUX: MEM to Host Clock Gearing mux selector.** Eight nibbles of mux select for memory/DDR2 to MC/core geared clock boundary crossing phase enables. Refer to Table 62, "MEM to Host Gear Ratio Mux for MEMNDGRCFG0" for the programming details.

Table 62. MEM to Host Gear Ratio Mux for MEMNDGRCFG0

FSB: Memory Frequency	Gear Ratio	Option	Value
333:333 267:267	1:1	only	11111111h
333:267	5:4	conservative	00003054h
267:333	4:5	conservative	00000325h

3.9.3.5 MEMNDGRCFG1: MEM Next Data Gear Ratio Configuration 1

This register consists of 8 nibbles of mux select data for the proper selection of gearing behavior in the MC for the 1:1 and 4:5 modes. This is the second register for MEM to Host gearing control for the "data_next" signaling.

Device: 16
Function: 1
Offset: 16Ch

Bit	Attr	Default	Description
31:0	RWST	00000000h	**MEMNDGRMUX: MEM to Host Clock Gearing mux selector.** Eight nibbles of mux select for memory/DDR2 to MC/core geared clock boundary crossing phase enables. Refer to Table 63, "MEM to Host Gear Ratio Mux for MEMNDGRCFG1" for the programming details.

Table 63. MEM to Host Gear Ratio Mux for MEMNDGRCFG1

FSB: Memory Frequency	Gear Ratio	Option	Value
333:333 267:267	1:1	only	00000000h
333:267	5:4	conservative	00002000h
267:333	4:5	conservative	00000406h

3.9.3.6 HOSTTOMEMGRCFG0: Host to MEM Gear Ratio Configuration 0

This register consists of 8 nibbles of mux select data for the proper selection of gearing behavior on the Host to MC path (south bound).

Device: 16
Function: 1
Offset: 170h

Bit	Attr	Default	Description
31:0	RWST	11111111h	**HSTMEMGRMUX: Host to MEM Clock Gearing mux selector.** Eight nibbles of mux select for FSB/core to memory/DDR2 geared clock boundary crossing phase enables. Refer to Table 64, "Host to MEM Gear Ratio Mux Select" for the programming details.

Table 64. Host to MEM Gear Ratio Mux Select

FSB: Memory Frequency	Gear Ratio	Option	Value
333:333 267:267	1:1	only	11111111h
333:267	5:4	conservative	00005430h
267:333	4:5	conservative	00010320h

3.9.3.7 HOSTTOMEMGRCFG1: Host to MEM Gear Ratio Configuration 1

This register consists of 8 nibbles of mux select data for the proper selection of gearing behavior on the Host to MC path (south bound).

Device: 16
Function: 1
Offset: 174h

Bit	Attr	Default	Description
31:0	RWST	00000000h	**HSTMEMGRMUX: Host to MEM Clock Gearing mux selector.** Eight nibbles of mux select for FSB/core to memory/DDR2 geared clock boundary crossing phase enables. Refer to Table 65, "Host to MEM Gear Ratio Mux Select" for the programming details.

Table 65. Host to MEM Gear Ratio Mux Select

FSB: Memory Frequency	Gear Ratio	Option	Value
333:333 267:267	1:1	only	00000000h
333:267	5:4	conservative	00001020h
267:333	4:5	conservative	00040000h

3.9.4 DRAM Timing Registers

The DRTA and DRTB registers defines timing parameters that work with the DDR2 DRAMs in the appropriate channel. The parameters for these devices can be obtained by serial presence detect. This register must be set to provide timings that satisfy the specifications of all SDRAM's detected. For example, if SDRAM's present have different TRCs, the maximum should be used to program this register. Consult the appropriate JEDEC DDR2 SDRAM specifications for the technology of the devices in use.

Devices are usually presented as 4-4-4, 5-5-5, etc. These numbers indicate T_{CAC}, T_{RCD} and T_{RP} with a fourth optional parameter, T_{RAS} (which has both maximum and minimum values):

T_{CAC}/T_{CL}: Column access time (minimum time between CAS strobe and data access)

T_{RCD}: RAS to CAS delay (minimum delay between RAS and CAS strobes).

T_{RP}: Precharge time (minimum time required to precharge a row)

T_{RAS}: Minimum/Maximum time a row can be active before it receives a precharge command

In the calculations in the text below, "T" is used to denote a time in ns while "n" refers to the number of clock cycles. So, for example:

- T_{FAW} refers to 37.5 ns
- N_{FAW} refers to 10 cycles (@ 266 MHz)

The actual register filed names are always T_{FAW} (though delays are programmed in clock cycles). BL, the burst length, is always set to eight (four clock cycles) for Intel® 5100 MCH Chipset.

Most of the bit fields represent timing rules that define a minimum separation of events that will be enforced by the device. Transactions that would break one of the timing rules (if issued to the DIMMs) are deemed to be "in conflict" and will not be issued.

3.9.4.1 DRTA[1:0]: DRAM Timing Register A

Device: 22, 21
Function: 0
Offset: 14Ch

Bit	Attr	Default	Description
31:27	RW	0Ah	**T_{FAW}: Electrical Throttling Window** This parameter is the smallest window over which four activations can be issued to a given rank: no more than four activations can be issued within any given (sliding) T_{FAW} window. This prevents DRAM power-supply droop violations. This parameter is defined in mclk cycles and is set to greater than or equal to the largest T_{FAW} of any DIMM on the memory subsystem. Parameter depends on DRAM page size, programmed as a number of clock cycles: Default value is Ah for 10x3.75 (266 MHz) cycles.
26:21	RW	13h	**T_{WRC}: 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 in mclk cycles and is set to greater than or equal to the largest T_{WRC} of any DIMM on the memory subsystem. This parameter is defined as follows: $(N_{CL} - 1) + BL/2 + (N_{RCD} + N_{WR} + N_{RP})$ (rounded up to the nearest integer), where N_{RCD} is the DDR RAS-to-CAS delay (maximum of 5), N_{CL} is the CAS-to-first-read-data latency, BL is the burst length, N_{WR} is the write recovery time and N_{RP} is the precharge time. Default is 4-1+8/2+4+4+4 = 19 cycles for 4-4-4 devices.
20:15	RW	10h	**T_{RC}: 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 mclk cycles and is set to greater than or equal to the largest T_{RC} of any DIMM on the memory subsystem. Default is 60 ns (16 cycles) for 533 MHz devices.
14:7	RW	22h	**T_{RFC}: 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 mclk cycles and is set to greater than or equal to the largest T_{RFC} of any DIMM on the memory subsystem. Default is 127.5 ns (34 cycles) for 1 Gb 533 MHz DRAM devices.
6:2	RW	0Ch	**T_{PHA}: Page hit read to activate command delay (same bank)** This parameter is required because of page hit mode. The normal rule used to separate activations to the same bank is T_{RC} which ensures a minimum separation between the activations. However, if one or more page hits occurs on the first activate, the normal rule will not hold off the next activate for long enough. So this rule imposes a minimum separation between a page hit read and the next activate to the same bank. The critical case is a single page hit (T_{RC} protects the no hit case). T_{RAS}(minimum) limits the earliest time the precharge can occur after the initial activate. We subtract 4 because the conflict will be applied to the page hit "activate" which is always 4 cycles later that the initial activate. The equation to use to calculate this value is then the maximum of $N_{PHA} = N_{RAS}min - 4 + N_{RP}$ $N_{PHA} = N_{RCD} + 2 + N_{RTP} + N_{RP}$ Where N_{RCD}, $N_{RAS}min$, N_{RP}, and N_{RTP} are JEDEC supplied parameters (in cycles). Default value is 12 cycles for 533 MHz 4-4-4 devices.
1:0	RW	00	**T_{REF}: Refresh command to Refresh command delay** This parameter specifies the average delay between two refresh commands to the same rank over a period of nine refresh intervals (nine T_{REF}s). This parameter ensures that a sufficient number of refreshes per time interval are issued to each rank. This parameter is defined in multiples of 3.9 µs. This parameter is set to less than or equal to the smallest T_{REF} of any DIMM on the memory subsystem. A value of zero disables refresh and clears the refresh counter. The default is 0 to prevent refreshes during the init sequence. Software should program a non-zero value after the init sequence has completed or on S3 exit. Programming a value of 3 is not supported

3.9.4.2 DRTB[1:0]: DDR Timing Register B

This register defines timing parameters that work with all DDR ports in the appropriate channel. This register must be set to provide timings that satisfy the specifications of all detected DDR ports. For example, if DDR ports have different T_{R2W}s, the max should be used to program this register.

Device: 22, 21
Function: 0
Offset: 150h

Bit	Attr	Default	Description
31:14	RV	04h	*Reserved*
13:11	RW	5h	**FAR: The logical far rank number**
10:8	RW	0h	**NEAR: The logical far rank number**
7:4	RV	0010	*Reserved*
3:2	RW	01	**T_{AL}: Additive Latency** Measured in mclk cycles. Our RAS->CAS spacing will be 3 cycles for all DRAMs.
1:0	RW	01	**T_{CL}: CAS Latency** This parameter depends on the type of memory installed - measured in mclk cycles: 00: 3 cycles 01: 4 cycles 10: 5 cycles 11: 6 cycles

3.9.4.3 DRPADCTL[1:0]: DRAM Pads Control Register

Device: 22, 21
Function: 0
Offset: 148h

Bit	Attr	Default	Description
31:24	RW	00h	**SLEWOVERRIDE: Slew Rate Override**
23	RW	0	**RANK45DIS**: Allows Chip Select / CKE / ODT disables for ranks 4&5. Used in 6 rank mode only - has no effect in 4 rank mode.
22:20	RW	0h	**CLOCKCNTL: Clock Control** Coarse delay of CMD/ADD output clock master DLL
19:14	RW	11h	**LEGOVERRIDE: Leg Over Ride** Drive override for all DDR drivers
13:11	RV	0h	*Reserved*
10	RW	1	**DQSx4MODE: DQS x4 Mode** Enables DQSx4 Mode for x8 DIMMs
9	RW	0	**CMDDIS: Command Disable** Disables CMD (RAS/CAS/WE) and ADD signals (save power if channel is not in use.
8	RW	0	**ADHIDIS: Address High Disable** Disables AD[15:14] and BA[2] pins ('1' disables). For smaller 4 bank devices.
7:4	RW	0h	**CKDIS: Clock disable** Clock disable for DIMMs 3-0 ('1' disables). For 2 DIMM solution, to save power, this value should be written as Ch.
3:0	RW	0h	**RANKDIS:** Chip Select/CKE/ODT disables for ranks 3-0 ('1' disables)

3.9.5 Memory Map Registers

3.9.5.1 TOLM - Top Of Low Memory

This register defines the low MMIO gap below 4 GB. See Section 3.9.5.2, "MIR[1:0]: Memory Interleave Range."

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. It does not denote the actual low MMIO gap but the upperbound of the system low memory. Addresses **equal to** or greater than the TOLM, and less than 4 GB, are decoded as low MMIO, MMCFG (if map within this range by HECBASE), chipset, interrupt/SMM and firmware as described in Section 4.0, "System Address Map." 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. The smallest gap between TOLM and 4 GB (for low MMIO, MMCFG, chipset, interrupt/SMM and firmware) is 256 MB because the largest value of TOLM is 0Fh. TOLM cannot be set higher than the total amount of physical memory. 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*

3.9.5.2 MIR[1:0]: Memory Interleave Range

These registers define each memory branch's interleave participation in processor-physical (A) space.

The MC uses all 39 physical address bits for DRAM memory addressing. However, when the next-most significant bit is set in the LIMIT field of the MIR[1:0] registers defined below, the rest of the bits in the LIMIT field are ignored, and the LIMIT field is interpreted as 512 GB.

The MIR addresses A[38:28] in multiples of 256 MB boundaries. The MMIO gap is defined as 4 GB - TOLM or mathematically (10H-TOLM.TOLM)x256 MB

Each MIR register defines a range. If the processor-physical address falls in the range defined by an MIR, the "way" fields in that MIR define channel participation in the interleave. The way-sensitive address bit is A[6]. For a MIR to be effective, WAY0 and WAY1 fields can not both be set to 0b. Matching addresses participate in the corresponding ways.

Compensation for MMIO gap size is performed by adjusting the limit of each range upward if it is above TOLM as shown in Table 66, "Interleaving of Address Is Governed by MIR[i] if."

MIR updates can only occur in the "Reset", "Ready", "Fault", and "Disabled" configuration register states.

Table 66. Interleaving of Address Is Governed by MIR[i] if

Limit with Respect to TOLM	Match MIR[i]
if MIR[i].LIMIT[11:0] <= TOLM	then MIR[i].LIMIT[11:0] > A[38:28] >= MIR[i-1][1].LIMIT[11:0]
if MIR[i].LIMIT[11:0] > TOLM > MIR[i-1].LIMIT[11:0]	then MIR[i].LIMIT[11:0] + (10H - TOLM) > A[38:28] >= MIR[i-1][1].LIMIT[11:0]
if MIR[i].LIMIT[11:0] > MIR[i-1].LIMIT[11:0] >= TOLM	then MIR[i].LIMIT[11:0] + (10H - TOLM) > A[38:28] >= MIR[i-1][1].LIMIT[11:0] + (10H - TOLM)

1. for MIR[0], MIR[i-1] is defined to be 0

Device: 16
Function: 1
Offset: 84h, 80h

Bit	Attr	Default	Description
15:4	RW	000h	**LIMIT:** This field defines the highest address in the range A[38:28] prior to modification by the TOLM register. ***Note:*** The maximum value is 400h (512 GB), and the minimum value is 1h (256 MB). The most-significant bit of this field is ignored.
3:2	RV	00	*Reserved*
1	RW	0	**WAY1:** Channel 1 participates in this MIR range if : this bit is set AND (the way-sensitive address bit is 1b OR **WAY0** of this MIR is 0b).
0	RW	0	**WAY0:** Channel 0 participates in this MIR range if : this bit is set AND (the way-sensitive address bit is 0b OR **WAY1** of this MIR is 0b).

3.9.5.3 AMIR[1:0]: 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: 90h, 8Ch

Bit	Attr	Default	Description
15:0	RW	0000h	**ADJLIMIT: Adjusted MIR Limit**

3.9.6 Memory Error Registers

Note: Since errors recorded in the NERR regs are not signaled through the ERR[2:0]# or MCERR# pins, FERR masking strategy should consider loss of external signaling on subsequent error (NERR).

ECC errors are defined on a 32 byte ECC word. Since there are two words in each cache line, an error can occur in both halves of the cache line. This can result in both FERR and NERR registers being populated. If an address bus error occurs, this will normally corrupt ECC in both halves of the cache line.

Normally, if a M1Err is indicated another uncorrectable error bit is set to indicate what type of transaction (scrub, sparing) generated the error.

3.9.6.1 FERR_NF_MEM: MC First Non Fatal Errors

The first non-fatal error for a channel is flagged in these registers. More than one flag can be set if different errors occur in the same cycle. For prioritization within the same error number, channel 0 has higher priority than channel 1. For the Chan_Indx, channel 0 errors will have higher priority, for example the channel 1 index will read as 0 if errors are logged simultaneously on both channels.

Device: 16
Function: 1
Offset: A0h

Bit	Attr	Default	Description
31:29	RV	0h	*Reserved*
28	RWCST	0	**Chan_Indx: Logs channel in which the highest-order error occurred** Valid only when one of the lower bits is non-zero.
27:22	RV	00h	*Reserved*
21	RWCST	0	**M21Err: Spare Copy Completed**
20	RWCST	0	**M20Err: Spare Copy Initiated**
19	RV	0	*Reserved*
18	RWCST	0	**M18Err: SPD protocol Error**
17	RV	0	*Reserved*
16	RWCST	0	**M16Err: Correctable Patrol Data ECC**
15	RWCST	0	**M15Err: Correctable Spare-Copy Data ECC**
14	RWCST	0	**M14Err: Correctable Demand Data ECC**
13	RV	0	*Reserved*
12	RWCST	0	**M12Err: Non-Aliased Uncorrectable Patrol Data ECC**
11	RWCST	0	**M11Err: Non-Aliased Uncorrectable Spare-Copy Data ECC**
10	RWCST	0	**M10Err: Non-Aliased Uncorrectable Demand Data ECC**
9:7	RV	0h	*Reserved*
6	RWCST	0	**M6Err: Aliased Uncorrectable Patrol Data ECC**
5	RWCST	0	**M5Err: Aliased Uncorrectable Spare-Copy Data ECC**
4	RWCST	0	**M4Err: Aliased Uncorrectable Demand Data ECC**
3:2	RV	0h	*Reserved*
1	RWCST	0	**M1Err: Uncorrectable Data ECC on Replay**
0	RV	0	*Reserved*

3.9.6.2 NERR_NF_MEM: MC Next Non-Fatal Errors

If an error is already flagged in FERR_NF_MEM, subsequent and lower-priority non-fatal errors are logged in NERR_NF_MEM.

Device: 16
Function: 1
Offset: A4h

Bit	Attr	Default	Description
31:22	RV	000h	*Reserved*
21	RWCST	0	**M21Err: Spare Copy Completed**

Device: 16 Function: 1 Offset: A4h			
Bit	**Attr**	**Default**	**Description**
20	RWCST	0	**M20Err: Spare Copy Initiated**
19	RV	0	*Reserved*
18	RWCST	0	**M18Err: SPD protocol Error**
17	RV	0	*Reserved*
16	RWCST	0	**M16Err: Correctable Patrol Data ECC**
15	RWCST	0	**M15Err: Correctable Spare-Copy Data ECC**
14	RWCST	0	**M14Err: Correctable Demand Data ECC**
13	RV	0	*Reserved*
12	RWCST	0	**M12Err: Non-Aliased Uncorrectable Patrol Data ECC**
11	RWCST	0	**M11Err: Non-Aliased Uncorrectable Spare-Copy Data ECC**
10	RWCST	0	**M10Err: Non-Aliased Uncorrectable Demand Data ECC**
9:7	RV	0h	*Reserved*
6	RWCST	0	**M6Err: Aliased Uncorrectable Patrol Data ECC**
5	RWCST	0	**M5Err: Aliased Uncorrectable Spare-Copy Data ECC**
4	RWCST	0	**M4Err: Aliased Uncorrectable Demand Data ECC**
3:2	RV	0h	*Reserved*
1	RWCST	0	**M1Err: Uncorrectable Data ECC on Replay**
0	RV	0	*Reserved*

3.9.6.3 EMASK_MEM: MC Error Mask Register

A '0' in any field enables logging of that error.

Device: 16 Function: 1 Offset: A8h			
Bit	**Attr**	**Default**	**Description**
31:22	RV	000h	*Reserved*
21	RWST	1	**M21Err: Spare Copy Completed**
20	RWST	1	**M20Err: Spare Copy Initiated**
19	RV	0	*Reserved*
18	RWST	1	**M18Err: SPD protocol Error**
17	RV	0	*Reserved*
16	RWST	1	**M16Err: Correctable Patrol Data ECC**
15	RWST	1	**M15Err: Correctable Spare-Copy Data ECC**
14	RWST	1	**M14Err: Correctable Demand Data ECC**
13	RV	0	*Reserved*
12	RWST	1	**M12Err: Non-Aliased Uncorrectable Patrol Data ECC**
11	RWST	1	**M11Err: Non-Aliased Uncorrectable Spare-Copy Data ECC**
10	RWST	1	**M10Err: Non-Aliased Uncorrectable Demand Data ECC**

Device: 16 Function: 1 Offset: A8h			
Bit	**Attr**	**Default**	**Description**
9:7	RV	0h	*Reserved*
6	RWST	1	**M6Err: Aliased Uncorrectable Patrol Data ECC**
5	RWST	1	**M5Err: Aliased Uncorrectable Spare-Copy Data ECC**
4	RWST	1	**M4Err: Aliased Uncorrectable Demand Data ECC**
3:2	RV	0h	*Reserved*
1	RWST	1	**M1Err: Uncorrectable Data ECC on Replay**
0	RV	0	*Reserved*

3.9.6.4 ERR0_MEM: MC Error 0 Mask Register

A '0' in any field enables logging of that error. This register enables the signaling of Err[0] when an error flag is set. The field definition for this register is identical to that of EMASK_MEM.

Device: 16 Function: 1 Offset: ACh			
Bit	**Attr**	**Default**	**Description**
31:22	RV	000h	*Reserved*
21	RW	1	**M21Err: Spare Copy Completed**
20	RW	1	**M20Err: Spare Copy Initiated**
19	RV	0	*Reserved*
18	RW	1	**M18Err: SPD protocol Error**
17	RV	0	*Reserved*
16	RW	1	**M16Err: Correctable Patrol Data ECC**
15	RW	1	**M15Err: Correctable Spare-Copy Data ECC**
14	RW	1	**M14Err: Correctable Demand Data ECC**
13	RV	0	*Reserved*
12	RW	1	**M12Err: Non-Aliased Uncorrectable Patrol Data ECC**
11	RW	1	**M11Err: Non-Aliased Uncorrectable Spare-Copy Data ECC**
10	RW	1	**M10Err: Non-Aliased Uncorrectable Demand Data ECC**
9:7	RV	0h	*Reserved*
6	RW	1	**M6Err: Aliased Uncorrectable Patrol Data ECC**
5	RW	1	**M5Err: Aliased Uncorrectable Spare-Copy Data ECC**
4	RW	1	**M4Err: Aliased Uncorrectable Demand Data ECC**
3:2	RV	0h	*Reserved*
1	RW	1	**M1Err: Uncorrectable Data ECC on Replay**
0	RV	0	*Reserved*

3.9.6.5 ERR1_MEM: MC Error 1 Mask Register

A '0' in any field enables logging of that error. This register enables the signaling of Err[1] when an error flag is set. The field definition for this register is identical to that of EMASK_MEM.

Device: 16
Function: 1
Offset: B0h

Bit	Attr	Default	Description
31:22	RV	000h	*Reserved*
21	RW	1	**M21Err: Spare Copy Completed**
20	RW	1	**M20Err: Spare Copy Initiated**
19	RV	0	*Reserved*
18	RW	1	**M18Err: SPD protocol Error**
17	RV	0	*Reserved*
16	RW	1	**M16Err: Correctable Patrol Data ECC**
15	RW	1	**M15Err: Correctable Spare-Copy Data ECC**
14	RW	1	**M14Err: Correctable Demand Data ECC**
13	RV	0	*Reserved*
12	RW	1	**M12Err: Non-Aliased Uncorrectable Patrol Data ECC**
11	RW	1	**M11Err: Non-Aliased Uncorrectable Spare-Copy Data ECC**
10	RW	1	**M10Err: Non-Aliased Uncorrectable Demand Data ECC**
9:7	RV	0h	*Reserved*
6	RW	1	**M6Err: Aliased Uncorrectable Patrol Data ECC**
5	RW	1	**M5Err: Aliased Uncorrectable Spare-Copy Data ECC**
4	RW	1	**M4Err: Aliased Uncorrectable Demand Data ECC**
3:2	RV	0h	*Reserved*
1	RW	1	**M1Err: Uncorrectable Data ECC on Replay**
0	RV	0	*Reserved*

3.9.6.6 ERR2_MEM: MEM 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. The field definition for this register is identical to that of EMASK_MEM.

Device: 16
Function: 1
Offset: B4h

Bit	Attr	Default	Description
31:22	RV	000h	*Reserved*
21	RW	1	**M21Err: Spare Copy Completed**
20	RW	1	**M20Err: Spare Copy Initiated**
19	RV	0	*Reserved*
18	RW	1	**M18Err: SPD protocol Error**
17	RV	0	*Reserved*

Device: 16 Function: 1 Offset: B4h			
Bit	**Attr**	**Default**	**Description**
16	RW	1	**M16Err: Correctable Patrol Data ECC**
15	RW	1	**M15Err: Correctable Spare-Copy Data ECC**
14	RW	1	**M14Err: Correctable Demand Data ECC**
13	RV	0	*Reserved*
12	RW	1	**M12Err: Non-Aliased Uncorrectable Patrol Data ECC**
11	RW	1	**M11Err: Non-Aliased Uncorrectable Spare-Copy Data ECC**
10	RW	1	**M10Err: Non-Aliased Uncorrectable Demand Data ECC**
9:7	RV	0h	*Reserved*
6	RW	1	**M6Err: Aliased Uncorrectable Patrol Data ECC**
5	RW	1	**M5Err: Aliased Uncorrectable Spare-Copy Data ECC**
4	RW	1	**M4Err: Aliased Uncorrectable Demand Data ECC**
3:2	RV	0h	*Reserved*
1	RW	1	**M1Err: Uncorrectable Data ECC on Replay**
0	RV	0	*Reserved*

3.9.6.7 MCERR_MEM: MEM MCERR Mask Register

A '0' in any field enables that error. This register enables the signaling of MCERR when an error flag is set. The field definition for this register is identical to that of EMASK_MEM.

Device: 16 Function: 1 Offset: B8h			
Bit	**Attr**	**Default**	**Description**
31:22	RV	000h	*Reserved*
21	RW	1	**M21Err: Spare Copy Completed**
20	RW	1	**M20Err: Spare Copy Initiated**
19	RV	0	*Reserved*
18	RW	1	**M18Err: SPD protocol Error**
17	RV	0	*Reserved*
16	RW	1	**M16Err: Correctable Patrol Data ECC**
15	RW	1	**M15Err: Correctable Spare-Copy Data ECC**
14	RW	1	**M14Err: Correctable Demand Data ECC**
13	RV	0	*Reserved*
12	RW	1	**M12Err: Non-Aliased Uncorrectable Patrol Data ECC**
11	RW	1	**M11Err: Non-Aliased Uncorrectable Spare-Copy Data ECC**
10	RW	1	**M10Err: Non-Aliased Uncorrectable Demand Data ECC**
9:7	RV	0h	*Reserved*
6	RW	1	**M6Err: Aliased Uncorrectable Patrol Data ECC**
5	RW	1	**M5Err: Aliased Uncorrectable Spare-Copy Data ECC**

Device: 16 Function: 1 Offset: B8h			
Bit	**Attr**	**Default**	**Description**
4	RW	1	**M4Err: Aliased Uncorrectable Demand Data ECC**
3:2	RV	0h	*Reserved*
1	RW	1	**M1Err: Uncorrectable Data ECC on Replay**
0	RV	0	*Reserved*

3.9.6.8 VALIDLOG[1:0]: Valid Log Markers

Each of the six Error Logs maintain their individual channel indices here. These indices complement the indices captured in FERR_NF_MEM and NERR_NF_MEM.

Device: 22, 21 Function: 0 Offset: 18Ch			
Bit	**Attr**	**Default**	**Description**
31:3	RV	0	*Reserved*
2	RWCST	0	**REDMEMVALID: REDMEM log is valid** '0' = Invalid '1' = Valid
1	RWCST	0	**RECMEMVALID: RECMEM log is valid** '0' = Invalid '1' = Valid This bit isn't set until after the replay if the initial request produced an ECC-uncorrectable error.
0	RWCST	0	**NRECMEMVALID: NRECMEM log is valid** '0' = Invalid '1' = Valid

3.9.6.9 NRECMEMA[1:0]: Non-Recoverable Memory Error Log Register A

This register latches information on the first fatal or non-recoverable uncorrectable memory error.

Device: 22, 21 Function: 0 Offset: 190h			
Bit	**Attr**	**Default**	**Description**
31:20	RV	000h	*Reserved*
19:15	ROST	00h	**MERR:** Identifies error that triggered the NRECMEM log.
14:12	ROST	0h	**BANK: Bank of the failed request**
11	RV	0	*Reserved*
10:8	ROST	000	**RANK: Rank of the failed request**
7:0	ROST	00h	**DM_BUF_ID: DM Buffer ID of the failed request**

3.9.6.10 NRECMEMB[1:0]: Non-Recoverable Memory Error Log Register B

This register latches information on the first detected fatal or non-recoverable uncorrectable memory error.

Device: 22, 21 Function: 0 Offset: 194h			
Bit	**Attr**	**Default**	**Description**
31	ROST	0	**RDWR (should always be 0)** '0' = Read '1' = Write
30:29	RV	00	*Reserved*
28:16	ROST	000h	**CAS: CAS address of the failed request** The CAS address will map from 12:0 while bit 10 (autoprecharge) is hardwired to 0.
15:0	ROST	0h	**RAS: RAS address of the failed request**

3.9.6.11 REDMEMA[1:0]: Recoverable Memory Data Error Log Register A

This register latches information on the first detected correctable ECC error.

Device: 22, 21 Function: 0 Offset: 198h			
Bit	**Attr**	**Default**	**Description**
31:0	ROST	0h	**SYNDROME**:

3.9.6.12 REDMEMB[1:0]: Recoverable Memory Data Error Log Register B

This register latches information on the first detected ECC error. Note that the ECC_locator field in this register is valid only when the corresponding FERR_NF_MEM register bits are set.

Device: 22, 21 Function: 0 Offset: 19Ch			
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 67, "ECC Locator Mapping Information" and Section 5.2.6.1, "Inbound ECC Code Layout for Memory Interface"

Table 67. ECC Locator Mapping Information (Sheet 1 of 2)

Symbols	DQS Lane	DQ Lane	Locator Bit
DS[1:0]	DQS0	DQ[3:0]	0
DS[3:2]	DQS1	DQ[11:8]	1
DS[5:4]	DQS2	DQ[19:16]	2
DS[7:6]	DQS3	DQ[27:24]	3
DS[9:8]	DQS4	DQ[35:32]	4
DS[11:10]	DQS5	DQ[43:40]	5
DS[13:12]	DQS6	DQ[51:48]	6
DS[15:14]	DQS7	DQ[59:56]	7

Table 67. ECC Locator Mapping Information (Sheet 2 of 2)

Symbols	DQS Lane	DQ Lane	Locator Bit
CS[1:0]	DQS8	DQ[67:64]	8
DS[17:16]	DQS9	DQ[7:4]	9
DS[19:18]	DQS10	DQ[15:12]	10
DS[21:20]	DQS11	DQ[23:20]	11
DS[23:22]	DQS12	DQ[31:28]	12
DS[25:24]	DQS13	DQ[39:36]	13
DS[27:26]	DQS14	DQ[47:44]	14
DS[29:28]	DQS15	DQ[55:52]	15
DS[31:30]	DQS16	DQ[63:60]	16
CS[3:2]	DQS17	DQ[71:68]	17

3.9.6.13 RECMEMA[1:0]: Recoverable Memory Error Log Register A

This register latches information on the first detected non-fatal memory error. An uncorrectable ECC error is non-fatal.

Device: 22, 21
Function: 0
Offset: 1A0h

Bit	Attr	Default	Description
31:20	RV	0	*Reserved*
19:15	ROST	0h	**MERR:** identifies error that triggered the RECMEM and REDMEM logs
14:12	ROST	0h	**BANK: Bank of the failed request**
11	RV	0	*Reserved*
10:8	ROST	000	**RANK: Rank of the failed request**
7:0	ROST	00h	**DM_BUF_ID: DM Buffer ID of the failed request**

3.9.6.14 RECMEMB[1:0]: Recoverable Memory Error Log Register B

This register latches information on the first detected non-fatal memory error. An uncorrectable ECC error is non-fatal

Device: 22, 21
Function: 0
Offset: 1A4h

Bit	Attr	Default	Description
31	ROST	0	**RDWR:** should always be 0 '0' = Read '1' = Write
30:29	RV	00	*Reserved*
28:16	ROST	000h	**CAS: CAS address of the failed request** The CAS address will map from 12:0 while bit 10 (autoprecharge) is hardwired to 0.
15:0	ROST	0h	**RAS: RAS address of the failed request**

3.9.7 Sparing Registers

There are one set of registers for the single memory channel. The registers appear in function 0 of different devices as shown in Table 28, "Functions Specially Handled by Intel® 5100 Memory Controller Hub Chipset."

3.9.7.1 SPCPC[1:0]: Spare Copy Control

These controls set up sparing for each channel. Channel zero (device 21) takes precedence over channel one (device 22): if both spare-control-enabled channels' spare error thresholds trigger in the same cycle, sparing will only commence on channel zero. Sparing will not commence on a competing channel until it's in-progress competitor's spare control enable is cleared and it's CERRCNT criteria is still met.

Device: 22, 21
Function: 0
Offset: 40h

Bit	Attr	Default	Description
23:19	RV	0h	*Reserved*
18:16	RW	00	**FORCERANK:** "From" Rank for Forced Spare Copy
15:8	RW	0h	**SETH: Spare Error Threshold** A spare fail-over operation will commence when the SPAREN bit is set and a CERRCNT.RANK[i] count for one and only one rank hits this threshold. The SETH field of the SPCPC registers must be programmed during initial BIOS configuration. If the SETH field of the SPCPC registers are attempted to be re-programmed after initial BIOS configuration with a value smaller than the current value of CERRCNT.RANK[i], sparing will not occur.
7	RV	0h	*Reserved*
6	RW	0	**FORCE: Initiate Spare Copy** '0'~>'1' transition while SPCPS.DSCIP = 0 initiates spare copy.
5	RW	0	**SPAREN: Spare Control Enable** '1' enables sparing, '0' disables sparing. The SPRANK field defines other characteristics of the sparing operation. 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.
4:3	RV	0h	*Reserved*
2:0	RWL	00	**SPRANK: Spare Rank** Target of the spare copy operation. This rank should not initially appear in a DMIR.RANK field. After the spare copy, the device 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.

3.9.7.2 SPCPS[1:0]: Spare Copy Status

Device: 22, 21
Function: 0
Offset: 43h

Bit	Attr	Default	Description
7	RO	0	**LBTHR: Leaky Bucket Threshold Reached** '0' = Leaky-bucket threshold not reached '1' = Leaky-bucket count matches SPCPC.SETH. Generates error M20. Cleared by reducing the offending count(s) in the CERRCNT registers.

Device: 22, 21 Function: 0 Offset: 43h			
Bit	**Attr**	**Default**	**Description**
6	RO	0	**DSCIP: DIMM-Rank Sparing Copy In Progress** '0' = DIMM-rank sparing copy not in progress. '1' = DIMM-rank sparing copy in progress. Set when SPCPC.SPAREN is set, and only one rank in CERRCNT is at threshold. This bit remains set until SFO is set. This bit is cleared when SFO is set. Error M20 is set when this bit transitions from '0' to '1'.
5	RO	0	**SFO: Spare Fail-Over** '0' = Spare has not been substituted for failing rank. '1' = Spare has been substituted for failing rank. Generates error M21. Cleared when SPCPC.SPAREN is cleared.
4:3	RV	0h	*Reserved*
2:0	RO	000	**FR: Failed Rank** Rank that was spared. Updated with the CERRCNT rank that has reached threshold when DSCIP is set. Read value only valid when DSCIP is set.

3.9.8 Memory RAS Registers

There are two sets of the following registers, one set for each Memory branch. They each appear in function 0 of different devices as shown in Table 28, "Functions Specially Handled by Intel® 5100 Memory Controller Hub Chipset."

3.9.8.1 RANKTHRESHOLD[1:0][5:0]: RANK Count Threshold

RANKTHRESHOLD[1:0][5:0] defines the rank threshold.

There are 12 time counters, one per each rank. Each counter is cleared on reset or when it reaches its RANKTHRESHOLD value. The output of the Error Period register is what increments the counters. Non-zero CERRCNT counts are decremented when their respective count register reaches the RANKTHRESHOLD values. Table 68, "Timing Characteristics of RANKTHRESHOLD" indicates the timing characteristics of this register based on the range of possible ERRPER values (0 to FFFFFFFFh).

Table 68. Timing Characteristics of RANKTHRESHOLD

Core Frequency	Per Increment	Maximum Period (increment period x ((2^16)-1))
333 MHz	48 ns - 206.2 s	3.15 µs - 156.4 days
266 MHz	60 ns - 257.7 s	3.93 µs - 195.5 s

Device: 22, 21 Function: 0 Offset: 1BAh, 1B8h, 176h, 174h, 172h, 170h			
Bit	**Attr**	**Default**	**Description**
15:0	RW	0h	**THRESH: CERRCNT decrement threshold.** When the counter reaches the value in this register, the leaky bucket counters are decremented by 1. A value of 0 prevents incrementing counter and thus decrementing CERRCNT.

The table below lists some recommended settings for the RANKTHRESHOLD registers. This assumes a soft error rate (EER) of 1 error per week per GB of installed memory. The drip rate is divided by two for conservatism (this means a rank getting over half the EER will have a CERRCNT that increments). The assumed value of ERRPER is the 369AC9FFh to give a scaling of 10^9 (converts ns to seconds).

Table 69. RANKTHRESHOLD Recommended Settings

Rank Size	EER[1]	EEP[2]	Drip Rate (2xEEP)	ERRPER Setting	RANKTHRESHOLD Setting based on 2xEEP	
					266 MHz	333 MHz
8 GB	8	21	42	10^9	9D8h	C4Eh
4 GB	4	42	84	10^9	13B0h	189Ch
2 GB	2	84	168	10^9	2760h	3138h
1 GB	1	168	336	10^9	4EC0h	6270h
512 MB	0.5	336	672	10^9	9D80h	C4E0h
256 MB	0.25	672	1344	10^9	FFFFh[3]	FFFFh[3]

1: EER - Expected Error Rate: Expected Number of Soft Errors / Week
2: EEP - Expected Error Period (= 1/EER): Number of hours between errors
3: Counter saturated - use max value (or higher prescaler value).

3.9.8.2 CERRCNT[1:0]: Correctable Error Count

These registers implement the "leaky-bucket" counters for correctable errors for each rank. Each field "limits" at a value of "255" ("1111 1111"). Non-zero counts are decremented when the RANKTHRESHOLD 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 1111" clears and thaws the count. Changing SPCPC.SETH will thaw the count, but will not clear it.

Note: This register counts the number of correctable errors based on code word granularity. Since each 32 bytes of a cache line plus 4 bytes of ECC constitute a code word, the correctable errors are incremented if the ECC check indicates an correctable error. Thus, there can be up to 2 correctable errors per cache line if both portions were deemed correctable.

Note: Aliased uncorrectable errors are not counted as correctable errors.

This register "works" whether or not sparing is enabled.

Device: 22, 21
Function: 0
Offset: 180h

Bit	Attr	Default	Description
31:24	RWCST	0h	**RANK3: Error Count for Rank 3**
23:16	RWCST	0h	**RANK2: Error Count for Rank 2**
15:8	RWCST	0h	**RANK1: Error Count for Rank 1**
7:0	RWCST	0h	**RANK0: Error Count for Rank 0**

3.9.8.3 CERRCNT_EXT[1:0]: Correctable Error Count

Extension of CERRCNT register for ranks 4 and 5.

Device: 22, 21 Function: 0 Offset: 184h			
Bit	**Attr**	**Default**	**Description**
15:8	RWCST	0h	**RANK5: Error Count for Rank 5**
7:0	RWCST	0h	**RANK4: Error Count for Rank 4**

3.9.8.4 BADRAM[1:0]: Bad DRAM Marker

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 RAM's are presumed "good". Only ranks containing x4 DRAM are "marked".

Device: 22, 21 Function: 0 Offset: 178h			
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**

3.9.8.5 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: 22, 21 Function: 0 Offset: 17Ch			
Bit	**Attr**	**Default**	**Description**
31:24	RV	0h	*Reserved*
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**

3.9.9 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. Address matching error injection can be programmed to occur on demand writes only. It will not occur for writes that occur as a result of scrubbing/sparing.

3.9.9.1 MEM[1:0]EINJMSK0: Memory Error Injection Mask0 Register

This register contains the error injection mask register to determine which bits get corrupted for error detection testing.

Device: 16
Function: 1
Offset: 208h, 200h

Bit	Attr	Default	Description
31	RWST	0	**ADDRMATCHEN: Address Match Enable - determines if address match is enabled for error injection:** 0: Address match disabled (default) 1: Address match enabled
30	RWST	0	**RESPDONEDIS: Response Function Done Disable Signal** This bit will disable the done signal returned from the response logic. 0: Use the "Done" return signal to remove the assertion 1: Disable the "Done" signal from alerting the response function
29:28	RWST	01	**HLINESEL: Half cache line selection** 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 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	RWST	0	**EINJEN: Error injection enable** 0: Disable error injection 1: Enable error injection
26	RWST	0	**EINJFUNCTSEL: Error Injection Function Select** 0: Select DINJ0 response function. 1: Select DINJ1 response function.
25:10	RWST	0h	**XORMSK: XOR mask bit for first device pointer** [25:18]: XOR mask for transfer 1 (lower half cache line) or 3 (upper half cache line). [17:10]: XOR mask for transfer 0 (lower half cache line) or 2 (upper half cache line).
9:5	RWST	0h	**SEC2RAM: Second device pointer location [17:0]** There are 18 - x8 device locations across both DIMM channels [1:0] that the XOR mask can be applied.
4:0	RWST	0h	**FIR2RAM: First device pointer location [17:0]** There are 18 - x8 device locations across both DIMM channels [1:0] that the XOR mask can be applied.

3.9.9.2 MEM[1:0]EINJMSK1: 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: 206h, 204h			
Bit	**Attr**	**Default**	**Description**
15:0	RWST	0h	**XORMSK: XOR mask bit for second device location** [15:8]: XOR mask for transfer 1 (lower half cache line) or 3 (upper half cache line). [7:0]: XOR mask for transfer 0 (lower half cache line) or 2 (upper half cache line).

3.9.9.3 MEMEINJADDRMAT: Error Injection Address Match Register

Contains the address match for error injection. The address mask register determines which of the lower bits [11:6] are ignored for determining a match. Enabled on channel basis by bit 31 of MEM[1:0]EINJMSK0

Device: 16 Function: 1 Offset: 20Ch			
Bit	**Attr**	**Default**	**Description**
31:0	RWST	0000h	SYSADDRMAT: System Address[35:6] match

3.9.9.4 MEMEINJADDRMSK: Error Injection Address Mask Register

Contains the address match for error injection. The address mask register determines which of the lower bits [11:6] are ignored for determining a match. Enabled on channel basis by bit 31 of MEM[1:0]EINJMSK0

Device: 16 Function: 1 Offset: 210h			
Bit	**Attr**	**Default**	**Description**
15:6	RV	000h	*Reserved*
5:0	RWST	00h	SYSADDRMSK: System Address[11:6] Mask Address lines are ignored for match if the corresponding mask bit is set. By default, no masking, an exact match is required.

3.9.10 Memory Interface Control

3.9.10.1 DSRETC[1:0]: DRAM Self-Refresh Extended Timing and Control

This register sets the timing of operations to different ranks while the auto-refresh FSM controls the DRAM command bus. This allows power intensive commands to be staggered. Both fields should be set so as not to violate T_{RFC}. In both 4 rank and 6 rank modes, the min value allowed for DRARTIM is T_{RFC} (because of CKE sharing, self-refreshes can start on any rank and have to be spaced from the last autorefresh) while the min value allowed for DSRENT is $2xT_{RFC}$ / N, because of CKE sharing.

Device: 22, 21 Function: 0 Offset: 144h			
Bit	**Attr**	**Default**	**Description**
15:8	RWST	14h	**DRARTIM: auto-refresh timing- stagger of commands between ranks**

Device: 22, 21 Function: 0 Offset: 144h			
Bit	**Attr**	**Default**	**Description**
7:0	RWST	14h	**DRSRENT: self-refresh entry timing - stagger of commands between ranks**

3.9.11 Serial Presence Detect Registers

These registers appear in function 0 of different devices as shown in Table 28, "Functions Specially Handled by Intel® 5100 Memory Controller Hub Chipset."

3.9.11.1 SPDDATA - 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: 16 Function: 1 Offset: 48h			
Bit	**Attr**	**Default**	**Description**
15	RO	0	**RDO: Read Data Valid.** This bit is set by the MCH when the Data field of this register receives read data from the SPD EEPROM after completion of an SPDR command. It is cleared by the MCH when a subsequent SPDR command is issued.
14	RO	0	**WOD: Write Operation Done.** This bit is set by the MCH when a SPDW command has been completed on the SPD bus. It is cleared by the MCH when a subsequent SPDW command is issued.
13	RO	0	**SBE: SPD Bus Error.** This bit is set by the MCH if it initiates an SPD bus transaction that does not complete successfully. It is cleared by the MCH when an SPDR or SPDW command is issued.
12	RO	0	**BUSY: Busy state.** This bit is set by the MCH while an SPD command is executing.
11:8	RV	0h	*Reserved*
7:0	RO	00h	**DATA: Data.** Holds data read from SPDR commands.

3.9.11.2 SPDCMD: Serial Presence Detect Command Register

A write to this register initiates a DIMM EEPROM access through the SPD bus.

Device: 16 Function: 1 Offset: 4Ch			
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.

Device: 16
Function: 1
Offset: 4Ch

Bit	Attr	Default	Description
27	RWST	1	**CKOVRD: Clock Over-ride.** '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 "budge" 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.
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 70. Device 8, Function 0, DMA Engine Configuration Map

DID	VID	00h	FERR_CHANCMD	FERR_CHANERR	80h
PEXSTS	PEXCMD	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_CHANCMP		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
PEXDEVSTS	PEXDEVCTRL	74h			F4h
		78h			F8h
CB_ERR_DOCMD		7Ch			FCh

The VID and DID of the DMA Engine are defined in Section 3.8.1.1, "VID - Vendor Identification Register" and Section 3.8.1.2, "DID - Device Identification Register." The RID is defined in Section 3.8.1.3, "RID - Revision Identification Register" while the HDR register appears in Section 3.8.1.5, "HDR - Header Type Register." The SVID and SID can be found in Section 3.8.1.6, "SVID - Subsystem Vendor Identification Register."

3.11 CB_BAR MMIO Registers

These MMIO registers are placed in the configuration ring for CPU MMIO accesses but the 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 71. Device 8, Function 1, DMA Engine DMABAR MMIO Registers (General, DMA Channel 0) Mapped through Configuration

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	CHDSCSTATE0	CHANSRCLENREMPF0	CHANSRCADDRPFH0	E4h
				68h	CHANDSTADDRPFL0			E8h
				6Ch	CHANSTATE0	CHANDSTLENREMPF0	CHANDSTADDRPFH0	ECh
				70h	CHANSRCADDRFL0			F0h
				74h	FETCHSTATE0	CHANSRCLENREMF0	CHANSRCADDRFH0	F4h
				78h	CHANDSTADDRFL0			F8h
				7Ch	ERRNOTIFYSTATE0	CHANDSTLENREMF0	CHANDSTADDRFH0	FCh

Table 72. Device 8, Function 1: DMABAR MMIO Channel 2 and 3 Registers

DMA_COMP1		CHANCTRL1	100h	DMA_COMP2		CHANCTRL2	180h
CHANSTS1			104h 108h	CHANSTS2			184h 188h
CHAINADDR1			10Ch 110h	CHAINADDR2			18Ch 190h
		CHANCMD1	114h			CHANCMD2	194h
CHANCMP1			118h 11Ch	CHANCMP2			198h 19Ch
CDAR1			120h 124h	CDAR2			1A0h 1A4h
CHANERR1			128h	CHANERR2			1A8h
CHANERRMSK1			12Ch	CHANERRMSK2			1ACh
			130h 134h 138h 13Ch				1B0h 1B4h 1B8h 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
CHDSCSTATE 1	CHANSRCLENREMPF1	CHANSRCAD DRPFH1	164h	CHDSCSTATE 2	CHANSRCLENREMPF2	CHANSRCAD DRPFH2	1E4h
CHANDSTADDRPFL1			168h	CHANDSTADDRPFL2			1E8h
CHANSTATE1	CHANDSTLENREMPF1	CHANDSTAD DRPFH1	16Ch	CHANSTATE2	CHANDSTLENREMPF2	CHANDSTAD DRPFH2	1ECh
CHANSRCADDRFL1			170h	CHANSRCADDRFL2			1F0h
FETCHSTATE 1	CHANSRCLENREMF1	CHANSRCAD DRFH1	174h	FETCHSTATE 2	CHANSRCLENREMF2	CHANSRCAD DRFH2	1F4h
CHANDSTADDRFL1			178h	CHANDSTADDRFL2			1F8h
ERRNOTIFYS TATE1	CHANDSTLENREMF1	CHANDSTAD DRFH1	17Ch	ERRNOTIFYS TATE2	CHANDSTLENREMF2	CHANDSTAD DRFH2	1FCh

Table 73. 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 74. 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
			308h				388h
BRIDGE_ID2			30Ch	BRIDGE_ID3			38Ch
			310h				390h
			314h				394h
	PORTPRI2	STRMMAP_OFFSET2	318h		PORTPRI3	STRMMAP_OFFSET3	398h
		STRM_COMP2	31Ch			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
			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

3.11.1 PEXCMD: PCI Command Register

Device: 8
Function: 0
Offset: 04h

Bit	Attr	Default	Description
15:11	RV	0h	*Reserved*

Device: 8
Function: 0
Offset: 04h

Bit	Attr	Default	Description
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 cache line 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® 5100 MCH Chipset 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*, Rev. 2.3. 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 PEXSTS: 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. Refer to Section 5.24, "Error List".
14	RWC	0	**SSE:** Signaled 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: Signaled 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, "PEXCMD: PCI Command Register" is cleared, this bit is never set. Refer to Section 5.24, "Error List".
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:** INTx 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*, Rev. 2.3. This register maintains compatibility with PCI configuration space. Since this register is part of the standard PCI header, there is a PEXSTS 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[1]. 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.

1. 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 the Intel® 5100 MCH Chipset.
39:10	RW	003F9C00h	**CB_BASE_WIN:** DMABase Window This marks the 1 kB 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® 5100 MCH Chipset 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® 5100 MCH 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 INTL: Interrupt Line Register

The Interrupt Line register is used to communicate interrupt line routing information between initialization code and the device driver. The Intel® 5100 MCH Chipset does not have a dedicated interrupt line and is not used.

Device: 8 Function: 0 Offset: 3Ch			
Bit	**Attr**	**Default**	**Description**
7:0	RWO	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 the Intel® 5100 MCH Chipset since the PCI Express* port does not have interrupt lines.

3.11.7 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

3.11.8 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.*
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® 5100 MCH Chipset 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, i.e., 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)[1] 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 0's 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.

1. In the degraded mode, when the total prefetches are set to 20 (i.e., 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.

3.11.9 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.

3.11.9.1 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-to-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 the Intel® 5100 MCH Chipset) XX1XXb - PMEOUT can be asserted from D2 (Not supported by the Intel® 5100 MCH Chipset) X1XXXb - PMEOUT can be asserted from D3 hot 1XXXXb - PMEOUT can be asserted from D3 cold
26	RO	0	**D2S:** D2 Support The Intel® 5100 MCH Chipset does not support power management state D2.
25	RO	0	**D1S:** D1 Support The Intel® 5100 MCH Chipset 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* Base Specification*, Rev. 1.0a.
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.

3.11.9.2 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 Bus Power Management Interface Specification*, revision 1.1, for details. This is not implemented in the e Power Management capability for the Intel® 5100 MCH Chipset 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® 5100 MCH Chipset 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 ICH9R through the "Assert(Deassert)_PMEGPE"message. This has no effect on the Intel® 5100 MCH Chipset 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

3.11.10 MSICAPID - Message Signaled 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

3.11.11 MSINXPTR - Message Signaled Interrupt Next Pointer Register

Device: 8
Function: 0
Offset: 59h

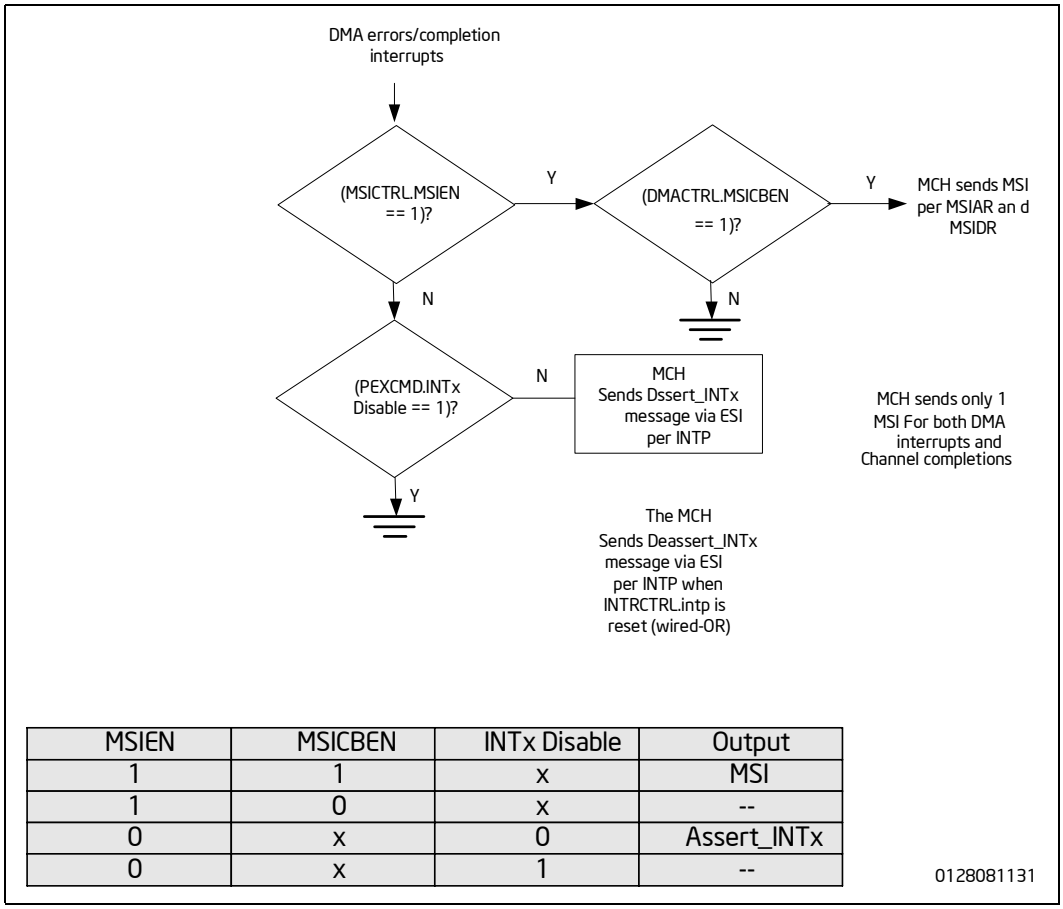
Bit	Attr	Default	Description
7:0	RO	6Ch	**NXTPTR: MSI Next Pointer**: The DMA Engine device is implemented as a PCI Express* device and this points to the PCI Express* capability structure.

3.11.12 MSICTRL - Message Signaled 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 Intel® 5100 MCH Chipset 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 3.11.14, "MSIDR: Message Signaled Interrupt Data Register" below for discussion on how the interrupts are handled.
3:1	RO	0h	**MMCAP:** Multiple Message Capable The Intel® 5100 MCH Chipset 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 14, "Intel® 5100 Memory Controller Hub Chipset DMA Error/Channel Completion Interrupt Handling Flow."

Figure 14. Intel® 5100 Memory Controller Hub 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	--

3.11.13 MSIAR: Message Signaled 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 IA-32 processor.
3	RW	0	**ARDHINT:** Address Redirection Hint 0: directed 1: redirectable

Device: 8 Function: 0 Offset: 5Ch			
Bit	**Attr**	**Default**	**Description**
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

3.11.14 MSIDR: Message Signaled 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	000	**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 as programmed by BIOS/Software will be used by the Intel® 5100 MCH Chipset to provide context sensitive interrupt information for different events such as DMA Errors, DMA completions that require attention from the processor. See Table 75, "IV Vector Table for DMA Errors and Interrupts" for IV handling for DMA.

Table 75. 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[1]

1. 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

3.11.15 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.

3.11.16 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.

3.11.17 PEXCAP - PCI Express* Capabilities Register

Device: 8
Function: 0
Offset: 6Eh

Bit	Attr	Default	Description
15:14	RV	0h	*Reserved*
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	1001	**DPT:** Device/Port Type: DMA Engine device represents a PCI Express* Endpoint.
3:0	RO	0001	**VERS:** Capability Version: DMA Engine supports *PCI Express* Base Specification*, Rev. 1.0a.

3.11.18 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

Device: 8 Function: 0 Offset: 70h			
Bit	**Attr**	**Default**	**Description**
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 bytes max payload size others- *Reserved*

3.11.19 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

Device: 8 Function: 0 Offset: 74h			
Bit	**Attr**	**Default**	**Description**
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 bytes max payload size 001: 256 bytes max payload size 010: 512 bytes max payload size 011: 1024 bytes max payload size 100: 2048 bytes max payload size 101: 4096 bytes max payload size ***Note:*** This field has no impact internally to the Intel® 5100 MCH Chipset 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 Section 3.8.11.4, "PEXDEVCTRL[7:2,0] - PCI Express* Device Control Register."
4	RO	0	**ENRORD: Enable Relaxed Ordering** No relaxed ordering is supported by the Intel® 5100 MCH Chipset. 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® 5100 MCH Chipset DMA Engine device as it does not report any non-fatal errors.
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® 5100 MCH Chipset DMA Engine device as it does not report any correctable errors.

3.11.20 PEXDEVSTS - PCI Express* Device Status Register

Device: 8 Function: 0 Offset: 76h			
Bit	**Attr**	**Default**	**Description**
15:6	RV	0h	*Reserved*

Device: 8 Function: 0 Offset: 76h			
Bit	**Attr**	**Default**	**Description**
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 that the Intel® 5100 MCH Chipset 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 the Intel® 5100 MCH 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 3.11.19, "PEXDEVCTRL - Device Control Register.") 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 3.11.19, "PEXDEVCTRL - Device Control Register.") 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 3.11.19, "PEXDEVCTRL - Device Control Register.") 1: correctable errors detected 0: No correctable errors detected

3.11.21 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.

3.11.21.1 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.

3.11.21.2 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 3.11.21.4.

Device: 8
Function: 0
Offset: 80h

Bit	Attr	Default	Description
15	RO	0	**FERR_Unaffiliated_Error_DMA15: FERR_Unaffiliated_Error_DMA15** The Intel® 5100 MCH Chipset does not detect unaffiliated errors
14	RO	0	**FERR_Soft_Error_DMA14**: **FERR_Soft_Error_DMA14** The Intel® 5100 MCH Chipset 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[1] or has not been configured. This address will be checked and set by the DMA Engine during execution, i.e., 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 3.11.22.2
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 (e.g., 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. (e.g., starvation). When this bit has been set, the address of the failed descriptor is in the Channel Status register.
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 (e.g., more than one command bit set).

Device: 8 Function: 0 Offset: 80h			
Bit	**Attr**	**Default**	**Description**
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, i.e., 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, e.g., 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 (e.g., > 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.

1. An illegal address is one which is flagged by the Intel® 5100 MCH Chipset system address decoder as invalid (e.g., 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.

3.11.21.3 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
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 (i.e., more than 1 bit is set in this register)

3.11.21.4 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)

3.11.21.5 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.

3.11.21.6 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.

3.11.21.7 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.

3.11.21.8 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.

3.11.21.9 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.

3.11.21.10 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.

3.11.21.11 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 3.11.23.8).

Device: 8
Function: 0
Offset: BCh

Bit	Attr	Default	Description
15	RO	0	**NERR_Unaffiliated_Error_DMA15** The Intel® 5100 MCH Chipset does not detect unaffiliated errors.
14	RO	0	**NERR_Soft_Error_DMA14** The Intel® 5100 MCH Chipset 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, i.e., 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 3.11.22.2).
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.
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 (e.g., 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. (e.g., 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 (e.g., 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, e.g., 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.

Device: 8 Function: 0 Offset: BCh			
Bit	**Attr**	**Default**	**Description**
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 (e.g., > 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.

3.11.22 DMA Registers

Table 76 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 128 bytes 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 (i.e., 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 76. 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

3.11.22.1 CHANCNT - Channel Count

The Channel Count register specifies the number of channels that are implemented.

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® 5100 MCH Chipset supports four DMA Channels. ***Note:*** This field will be set to "00000" by the chipset when the DMA Engine is disabled. Refer to Section 3.11.8 for details.

3.11.22.2 XFERCAP - Transfer Capacity

The Transfer Capacity specifies the minimum of the maximum DMA transfer size supported on all channels.

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® 5100 MCH Chipset as a power of 2. 01100: The value of 12 indicates 4 kB maximum transfer in Intel® 5100 MCH Chipset and is the default value Others: *Reserved*

3.11.22.3 INTRCTRL - Interrupt Control

The Interrupt Control register provides for control of DMA interrupts.

Offset: 03h			
Bit	**Attr**	**Default**	**Description**
7:3	RV	0h	*Reserved*
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 (i.e., does not send MSI nor drive the legacy interrupt signal).

3.11.22.4 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 3.11.23.2.

Offset: 04h			
Bit	**Attr**	**Default**	**Description**
31:4	RV	0h	*Reserved*

Offset: 04h			
Bit	**Attr**	**Default**	**Description**
3:0	RC[1]	0h	**ChanAttn:** Channel Attention Each bit specifies the interrupt status of each DMA channel. Bit 0 represents channel 0, bit 1 represents channel 1, etc. 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.

1. Reading this register clears all of the bits.

3.11.22.5 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: 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

3.11.22.6 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[1] 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: 0Ah			
Bit	**Attr**	**Default**	**Description**
15:2	RO	00C0h	**Fst_PPR_OFF:** Points to the first set of Per-port registers. 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*

1. 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 223.

3.11.22.7 INTRDELAY - Interrupt Delay Register

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	RW	0	**Interrupt Delay Time** Specifies the number of µs that the chipset delays generation of an interrupt (legacy or MSI) from the time that interrupts are enabled (i.e., Master Interrupt Enable bit in the INTRCTRL bit is set). The MCH does not support interrupt delay timer and hence hardwired to 0.

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 (i.e., 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 (i.e., 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 µs 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.

3.11.22.8 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: 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® 5100 MCH Chipset address decoding logic.
0	RO	0	**Deg_Mode:** Degraded Mode The Intel® 5100 MCH Chipset does not support degraded mode for DMA Engine operation and is hardwired to 0.

3.11.22.9 CHAN_SYSERR_MSK[3:0]: Channel System Error Mask Register

The Channel System Error Mask Register[1] provides selective control to mask errors that may cause SERR, SCI/NMI through BIOS control via ERR[2:0]# pins in the Intel® 5100 MCH Chipset. 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[1] to the system and can set them accordingly. The default value is to disable the system error signaling.

Offset: 4Ch, 48h, 44h, 40h			
Bit	**Attr**	**Default**	**Description**
31:16	RV	0h	*Reserved*
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

3.11.23 DMA Channel Specific Registers

As described in Table 71, 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 128 bytes for each channel. They are located in Bus 0, Device 1, Function 1 at the offsets shown in Section 3.11.

3.11.23.1 DMA_COMP[3:0]: DMA Compatibility Register

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.

1. 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.
1. An example would be an illegal write address to reserved space (firmware) that causes the system to crash (blue-screen in windows OS)

3.11.23.2 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: 200h, 180h, 100h, 80h			
Bit	**Attr**	**Default**	**Description**
15:9	RV	0h	Reserved
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® 5100 MCH Chipset 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 PEXCMD 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.

3.11.23.3 CHANSTS[3:0]: Channel Status Register

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 (64 bytes 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.

Offset: 204h, 184h, 104h, 84h			
Bit	**Attr**	**Default**	**Description**
5	RV	0	Reserved
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® 5100 MCH Chipset 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® 5100 MCH Chipset 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 3.11.23.5 000 - 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[1], 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 (e.g., 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.

3.11.23.4 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: 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.

1. 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.

3.11.23.5 CHANCMD[3:0] - DMA Channel Command Register

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 3.11.23.7) 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[1] 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[2] it. (See Section 3.11.23.8. 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.

1. 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.
2. The erroneous CHANCMD register is first cleared by Software and then the CHANERR.chan_cmd error register field is reset by Software.

3.11.23.6 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, i.e., 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: 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[1].
2:0	RV	0h	Reserved

1. This prevents straddling across cache lines since the contents of the CHANSTS registers are 8-bytes wide.

3.11.23.7 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 3.11.23.3

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 (i.e., the descriptor that is currently being processed by the DMA channel).

3.11.23.8 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® 5100 MCH Chipset 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: 228h, 1A8h, 128h, A8h			
Bit	**Attr**	**Default**	**Description**
31:16	RV	0h	*Reserved*
15	RO	0	**Unaffil_err: Unaffiliated Error** The Intel® 5100 MCH Chipset does not detect unaffiliated Errors.
14	RO	0	**Soft_err: Soft Error** The Intel® 5100 MCH Chipset does not implement soft error detection.

Offset: 228h, 1A8h, 128h, A8h			
Bit	**Attr**	**Default**	**Description**
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 3.11.8, Section 3.11.1 and Section 3.11.12. MSICTRL.MSIEN = 1, DMACTRL.MSICBEN = 0, PEXCMD.INTx Disable = 1 MSICTRL.MSIEN = 0, DMACTRL.MSICBEN = 1, PEXCMD.INTx Disable = 1 MSICTRL.MSIEN = 0, DMACTRL.MSICBEN = 0, PEXCMD.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® 5100 MCH Chipset 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 (e.g., 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: 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, e.g., 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 (e.g., >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.

3.11.23.9 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_CHANERR 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: 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
1	RWST	0	**DMA_xfrer_daddr_err_Msk:** DMA Transfer Destination Address Error Mask

Offset: 22Ch, 1ACh, 12Ch, ACh			
Bit	**Attr**	**Default**	**Description**
0	RWST	0	**DMA_trans_saddr_err_Msk:** DMA Transfer Source Address Error Mask

3.12 PCI Express* Per-Port Registers

Table 74 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 the PCI Express* proprietary enhancements. 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 (e.g., 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 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 (PCI Hot Plug*).

3.12.0.1 NXTPPRSET2 - Next Per Port Register Set

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.

3.12.0.2 NXTPPRSET3 - Next Per Port Register Set

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*

3.12.0.3 PPRSETLEN[3:2] - Per-Port Register Set Length Register

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).

3.12.0.4 STRMPRI[3:2]: Stream Priority Register

Offset:	383h, 303h		
Bit	**Attr**	**Default**	**Description**
7:4	RO	0000	**Highest_pri_supported: Highest Priority Supported** The Intel® 5100 MCH Chipset 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® 5100 MCH Chipset will treat the stream priority for the exceeding streams to default to that of priority 0.

3.12.0.5 REQID[3:2] - Requestor ID Register

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.

3.12.0.6 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: 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® 5100 MCH Chipset supports by indicating the highest StreamID value for which the device can specify a priority. The Intel® 5100 MCH Chipset supports only 1 stream and the highest stream ID is 0.

3.12.0.7 STRMIDFMT[3:2] - Stream ID Format Register

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® 5100 MCH Chipset 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.

3.12.0.8 BRIDGE_ID2 - Bridge ID Register

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 Stream Priority capability at a time (see In-Use bit).

3.12.0.9 BRIDGE_ID3 - Bridge ID Register

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 CB 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 CB 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 CB 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 Stream Priority capability at a time.

3.12.0.10 STRMMAP_OFFSET2: Stream Priority Mapping Offset Register

Offset: 318h			
Bit	**Attr**	**Default**	**Description**
15:2	RO	0h	**Strmid_offset: Offset of Stream ID Map** The Intel® 5100 MCH Chipset 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.

3.12.0.11 STRMMAP_OFFSET3: Stream Priority Mapping Offset Register

Offset: 398h			
Bit	**Attr**	**Default**	**Description**
15:2	RO	0h	**Strmid_offset: Offset of Stream ID Map** The Intel® 5100 MCH Chipset 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.

3.12.0.12 PORTPRI2: Port Priority Register

Offset: 31Ah			
Bit	**Attr**	**Default**	**Description**
7:4	RV	0	*Reserved*
3:0	RO	0h	**Port_pri: Port Priority** The Intel® 5100 MCH Chipset does not implement port priority for port 2 and this register will consistently return zero.

3.12.0.13 PORTPRI3: Port Priority Register

Offset: 39Ah			
Bit	**Attr**	**Default**	**Description**
7:4	RV	0	*Reserved*
3:0	RO	0h	**Port_pri: Port Priority** The Intel® 5100 MCH Chipset does not implement port priority for port 3 and this register will consistently return zero.

3.12.0.14 STRM_COMP[3:2] - Stream Priority Compatibility Register

Offset: 39Ch, 31Ch			
Bit	**Attr**	**Default**	**Description**
15:1	RV	0h	*Reserved*
0	RO	0	**v1_comp:** v1 Compatibility The Intel® 5100 MCH Chipset 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.

3.12.0.15 BR_MEM_BASE[3:2] - Bridge Memory Base Register

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 port 2). 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 Stream Priority capability at a time (see In-Use bit).

3.12.0.16 BR_MEM_LIMIT[3:2] - Bridge Memory Limit Register

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 port 2). Refer to Section 3.8.8.17

3.12.0.17 BR_PMEM_BASE[3:2] - Bridge Prefetchable Memory Base Register

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 port 2). 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 Stream Priority capability at a time (see In-Use bit).

3.12.0.18 BR_PMEM_LIMIT[3:2] - Bridge Prefetchable Memory Limit Register

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 port 2). 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.

3.12.0.19 BR_PBASE_UPPER32_P[3:2] - Bridge Prefetchable Base Upper 32 Register

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 port 2). 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.

3.12.0.20 BR_PLIMIT_UPPER32_P[3:2] - Bridge Prefetchable Limit Upper 32 Register

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 3.8.8.21If 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.

4.0 System Address Map

The Intel® 5100 MCH Chipset supports 36 bits of memory address space. The processors designed for the Intel® 5100 MCH Chipset, support only 36 bits of memory addressing and 16 bits of addressable I/O space.

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 (e.g., disable, read/write, write only, or read only). Attribute programming is described in Section 3.0, "Register Description". The Intel® 5100 MCH 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 Section 3.0, "Register Description".

This section 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® 5100 MCH Chipset memory map includes a number of programmable ranges. All of these ranges must be unique and non-overlapping. 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.

4.1 System Memory Address Ranges

The Intel® 5100 MCH Chipset supports 36 bits (up to 48 GB) of physical memory.

Other address spaces supported by the Intel® 5100 MCH Chipset are:

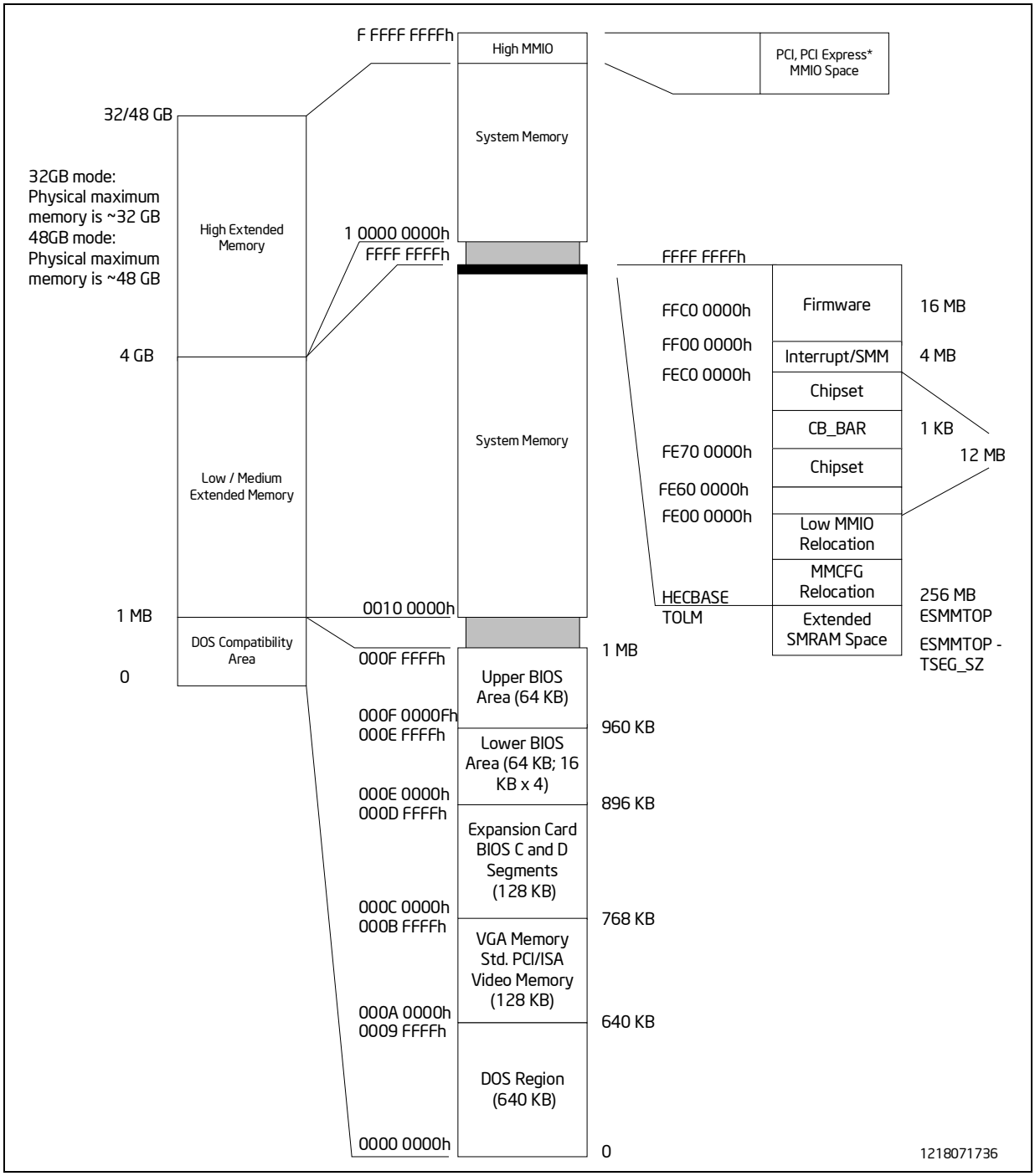
- 36-bit local address supported over the DDR2 channels for physical memory space.
- 32 and 64-bit address bit formats supported for PCI Express* interfaces.

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.

4.1.1 32/64-bit Addressing

For inbound and outbound writes and reads, the Intel® 5100 MCH Chipset supports 64-bit address format. If an outbound transaction's address is a 32-bit address, the Intel® 5100 MCH Chipset 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® 5100 MCH Chipset 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. The Intel® 5100 MCH Chipset implements 40-bit addressing on some internal interfaces. If configuration registers for these interfaces expose extra address bits, the BIOS should initialize the most-significant bits to zero because UP/DP processors only support 36-bit addressing.

Figure 15. 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 77. 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 SC.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 SC.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® 5100 MCH Chipset is set and BCTRL[x].VGA16bdecode is set to zero, then ISAEN bits of all peer PCI Express* ports with valid I/O range (PEXCMD.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 4 kB I/O addresses (PEXCMD[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 Display Adapter (MDA) Range (B 0000h–B 7FFFh)

The Intel® 5100 MCH 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® 5100 MCH Chipset, these regions are used to provide address space to PCI devices requiring memory space below 1 MB. The range is divided into eight sub-ranges. These ranges are defined by SC.PAM registers. There is a PAM register for each sub-range that defines the routing of reads and writes.

Table 78. 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 (ICH9R). 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® 5100 MCH 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 SC.PAM registers. This area can be mapped either the ESI port (ICH9R) 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 (ICH9R). 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® 5100 MCH 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 SC.PAM registers. The power-on default is set to read/write disabled with transactions forwarded to the ESI port (ICH9R). 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® 5100 MCH 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 the Intel® 5100 MCH 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 FEC8 FFFFh and FEE0 0000h to FEEF FFFFh
- High BIOS or Firmware area is the last 16 MB before the 4 GB boundary

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 Low/Medium Memory (TOLM) is limited to 4 GB of 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, "Main Memory Region."

4.3.2 15 MB - 16 MB Window (ISA Hole)

The Intel® 5100 MCH 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® 5100 MCH 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. The Intel® 5100 MCH 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 within the VGA address space, A 0000h - B FFFFh. This region is controlled by the Intel® 5100 MCH Chipset registers SC.EXSMRC.TSEG_SZ and SC.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.

The Intel® 5100 MCH 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® 5100 MCH Chipset 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® 5100 MCH Chipset. 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® 5100 MCH Chipset and the south bridge, such as an ICH9R.

The location of this MMCFG window is defined by the SC.HECBASE register. The HECBASE register could also be accessed through a fixed location. The default value of SC.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 real-address mode in that there are no privileges or address mapping. The Intel® 5100 MCH Chipset allows the relocation of HECBASE above 4 GB. However, SMM code cannot access extended configuration space if HECBASE is relocated above 4 GB. This is the SMM limitation. Page Size Extension (PSE) is supported in SMM, but Page Address Extension (PAE) support in SMM is not supported in P6 family processors. Refer to the *Intel® 64 and IA-32 Architectures Software Developer's Manual, Volume 3*, Section 24.1.

For more information on the memory mapped configuration mechanism described here, please see Section 4.6, "Configuration Space".

4.3.5 Low Memory Mapped I/O (MMIO)

This is the first of two Intel® 5100 MCH 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. Table 79, "Low Memory Mapped I/O" shows the registers used to define the MMIO ranges for each PCI Express*/ESI device. These registers are compatible with *PCI Express* Base Specification*, Rev. 1.0a and the *PCI-to-PCI Bridge Architecture*

Specification, revision 1.1. 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 79. Low Memory Mapped I/O

I/O Port	MCH Base[1]	MCH Limit[1]
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 SC.PMLU and SC.PMBU are 0's. Otherwise, the prefetchable memory space will be located in high MMIO space.
2. MCH does not need base/limit for ICH9R because subtractive decoding will send the accesses to the ICH9R. This is OK for software also, since the ICH9R is considered part of the same bus as the MCH.

The Intel® 5100 MCH Chipset 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 SC.PMLU and SC.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.

FE60 0000h - FE6F FFFFh: This range is used for fixed memory mapped Intel® 5100 MCH 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:

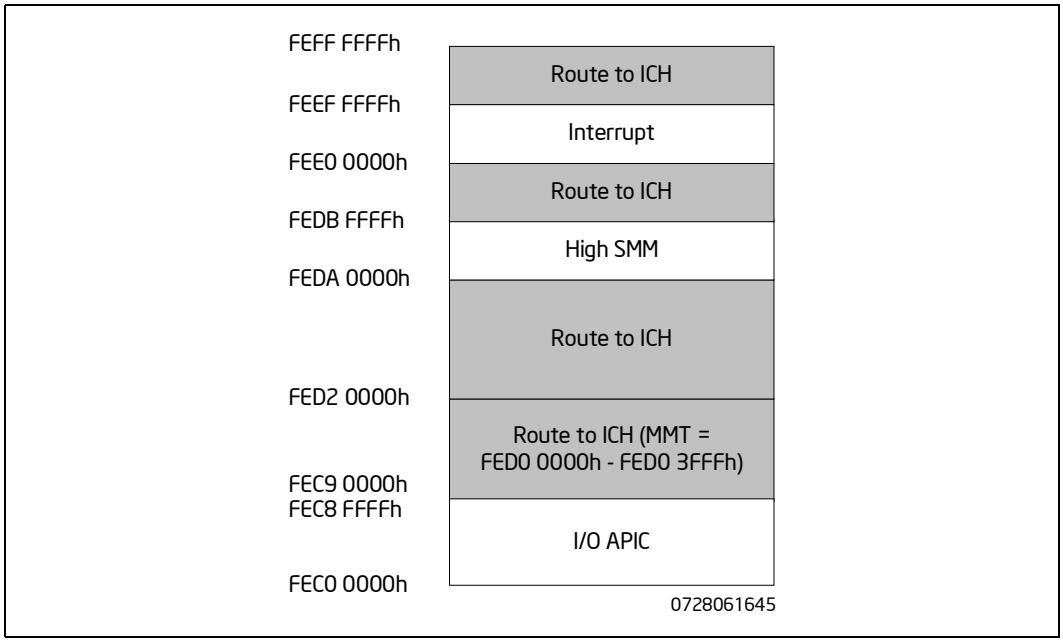
- Four Scratch Pad Registers
- Four Sticky Scratch Pad Registers
- Four Boot flag registers
- HECBASE register for MMCFG

These registers are described in Section 3.0, "Register Description". The Intel® 5100 MCH Chipset 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 16. 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 ICH9R 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 FFFFh is used to communicate with the IOAPIC controllers in the Intel® 6700PXH 64-bit PCI Hub or ICH9R 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 80. 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 (ICH9R for master abort)	FEC0 6000h to FEC8 FFFFh

For PCI 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® 5100 MCH Chipset 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® 5100 MCH Chipset allocates 16 MB of firmware space from FF00 0000h to FFFF FFFFh. Requests in this range are directed to the Compatibility Bus. The ICH9R 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 SC.MIR.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, "Main Memory Region"

4.3.8.2 High MMIO

The high memory mapped I/O region is located above the top of memory as defined by SC.MIR.LIMIT. These SC.PMBU and SC.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.LIMIT and it falls within the SC.PMBU+PMBASE and SC.PMLU+PMLIMIT range, it should be routed to the appropriate PCI Express* port. For accesses above MIR.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® 5100 MCH Chipset uses MIR.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® 5100 MCH Chipset 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 SC.MIR.LIMIT registers will decode an access into a specific interleaving range. Within the interleaving range, the SC.MIR.LIMIT register indicates which DDR2 memory channel the address is associated with.

4.4 Memory Address Disposition

The following section presents a summary of address dispositions for the Intel® 5100 MCH Chipset.

4.4.1 Registers Used for Address Routing

Table 81, "Intel® 5100 Memory Controller Hub Chipset Memory Mapping Registers" is a summary of the registers used to control memory address disposition. These registers are described in detail in Section 3.0, "Register Description".

Table 81. Intel® 5100 Memory Controller Hub Chipset Memory Mapping Registers

Name	Function
MIR[1:0]	Memory Interleaving Registers (DDR2 channel interleaving)
AMIR[1:0]	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.
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[1]
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 4 GB, 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 4 GB, this register will be set to all 0's.
PEXCMD (dev 2-7)	MSE (Memory Space Enable) bit enables the memory and prefetchable ranges.

1. 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® 5100 MCH Chipset. Table 82, "Address Disposition for Processor" defines the disposition of outbound requests entering the Intel® 5100 MCH Chipset on the processor bus. Table 86, "Address Disposition for Inbound Transactions" defines the disposition of inbound requests entering the Intel® 5100 MCH Chipset 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 ICH9R.

Table 82. Address Disposition for Processor (Sheet 1 of 2)

Address Range	Conditions	Intel® 5100 MCH Chipset Behavior
DOS	0 to 09FFFFh	Coherent Request to Main Memory. Route to main memory according to SC.MIR registers. Apply Coherence Protocol.
SMM/VGA	0A0000h to 0BFFFFh	See Table 84, "SMM Memory Region Access Control from Processor" and Table 85, "Decoding Processor Requests to SMM and VGA Spaces."
C and D BIOS segments	0C0000h to 0DFFFFh and PAM=11	Non-coherent request to main memory. Route to appropriate DDR2 DIMM device according to SC.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	
E and F BIOS segments	0E0000h to 0FFFFFh and PAM=11	Non-coherent request to main memory. Route to appropriate DDR2 DIMM device according to SC.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 SC.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 84, "SMM Memory Region Access Control from Processor" and Table 85, "Decoding Processor Requests to SMM and VGA Spaces."
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+256 MB	Convert to a configuration access and route according to the Configuration Access Disposition.
Intel® 5100 MCH Chipset-specific	FE00_0000h to FEBF_FFFFh AND valid Intel® 5100 MCH Chipset memory mapped register address	Issue configuration access to memory mapped register inside the Intel® 5100 MCH Chipset or to the DIMM based on the context.
	FE00_0000h to FEBF_FFFFh AND (NOT a valid Intel® 5100 MCH Chipset 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 80, "I/O APIC Address Mapping."
ICH9R/ICH9R timers	FEC9_0000h to FED1_FFFF	Issue request to ESI.

Table 82. Address Disposition for Processor (Sheet 2 of 2)

Address Range	Conditions	Intel® 5100 MCH Chipset Behavior
High SMM	FEDA_0000h to FEDB_FFFF	See Table 84, "SMM Memory Region Access Control from Processor" and Table 85, "Decoding Processor Requests to SMM and VGA Spaces."
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[x].LIMIT (depending on the physical memory external)	Coherent request to main memory. Route to main memory according to SC.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. There will be a subtractive agent within ICH, where it will attempt to decode the address. For undecoded address, transactions will be aborted. Non-posted transactions will be acknowledged with unsupported request (UR), and posted request will be dropped.

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 the Intel® 5100 MCH Chipset will set SC.EXSMRAMC.E_SMERR bit. Table 83, "Enabled SMM Ranges" defines when a SMM range is enabled. All the enable bits: G_SMRAME, H_SMRAM_EN, and TSEG_EN are located in the SC.EXSMRC register.

Table 83. 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 84, "SMM Memory Region Access Control from Processor" defines the access control of SMM memory regions from processors.

Table 84. SMM Memory Region Access Control from Processor

G_SMRAME	D_LCK	D_CLS	D_OPEN	SMMEM#	Code Access to SMM Memory[1]	Data Access to SMM Memory[2]
0[3]	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[4]	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)

1. BRLC
2. Data access transaction other than BRLC
3. For access to TSEG region (address range between ESMMTOP - TSEG_SZ and ESMMTOP), the Intel® 5100 MCH Chipset will route to identical system memory by definition (as TSEG is not enabled).
4. It is a programming error if D_CLS and D_OPEN are both set to 1, the Intel® 5100 MCH Chipset's behavior is undefined. The Intel® 5100 MCH Chipset could master abort SMM access.

The Intel® 5100 MCH Chipset 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 SC.SMRAMC, SC.EXSMRC, and SC.BCTRL registers. Table 85, "Decoding Processor Requests to SMM and VGA Spaces" defines the Intel® 5100 MCH Chipset'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 the Intel® 5100 MCH Chipset behavior if it falls in an enabled SMM space. Note that the D_CLS only applies to the legacy (A_0000-B_FFFFh) 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 86, "Address Disposition for Inbound Transactions."

Table 85. Decoding Processor Requests to SMM and VGA Spaces (Sheet 1 of 2)

SMM region	Transaction Address Range	SMM Memory Address Range	SMM Access Control[1]	G_SMRAME	H_SMRAME	T_EN	EWB/IWB	Routing
Legacy VGA/SMM [2]	A_0000h to B_FFFFh	A_0000h to B_FFFFh	x	0	x	x	x	to the VGA-enabled port (in BCTRL); otherwise, ESI[3]
			yes	1	**1**	x	x	
			no	1	x	x	x	
			yes	1	0	x	x	to SMM memory

Table 85. Decoding Processor Requests to SMM and VGA Spaces (Sheet 2 of 2)

SMM region	Transaction Address Range	SMM Memory Address Range	SMM Access Control[1]	G_SMRAME	H_SMRAME	T_EN	EWB/IWB	Routing
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[4]
			no	1	1	x	1	
			no	1	1	x	0	block access: master abort set EXSMRAMC.E_SMERR

1. SMM memory access control, see Table 84, "SMM Memory Region Access Control from Processor."
2. Software must not cache this region.
3. One and only one BCTRL can set the VGAEN; otherwise, send to ESI.
4. 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® 5100 MCH Chipset ESI may master abort a transaction without forwarding it back to the ESI.

Table 86. Address Disposition for Inbound Transactions (Sheet 1 of 2)

Address Range	Conditions	Intel® 5100 MCH Chipset Behavior
DOS	0 to 09FFFFh	Coherent Request to Main Memory. Route to main memory according to SC.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[1]
C, D, E, and F BIOS segments	0C0000h to 0FFFFFh and PAM=11[2]	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 DDR2 DIMM according to SC.MIR registers.

Table 86. Address Disposition for Inbound Transactions (Sheet 2 of 2)

Address Range	Conditions	Intel® 5100 MCH Chipset Behavior
Low/Medium Memory	10_0000 <= Addr < ESMMTOP - TSEG_SZ	Coherent Request to Main Memory. Route to main memory according to SC.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+256 MB	Inbound MMCFG access is not allowed and will be aborted.
Intel® 5100 MCH Chipset-specific	FE00_0000h to FEBF_FFFFh and valid Intel® 5100 MCH Chipset memory mapped register address	Inbound MMCFG access is not allowed and will be aborted.
	FE00_0000h to FEBF_FFFFh AND NOT a valid Intel® 5100 MCH Chipset 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 80, "I/O APIC Address Mapping"
ICH9R/ICH9R 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 Section 5.3, "Interrupts" 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
High Memory	1_0000_0000 to MIR[x].LIMIT (depending on the physical memory external)	Coherent Request to Main Memory. Route to main memory according to SC.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. There will be a subtractive agent within ICH, where it will attempt to decode the address. For undecoded address, transactions will be aborted. Non-posted transactions will be acknowledged with unsupported request (UR), and posted request will be dropped.

1. One and only one BCTRL can set the VGAEN; otherwise, send to ESI for master abort.
2. Other combinations of PAM's are not allowed if inbound accesses to this region can occur. 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, "Special I/O Addresses", 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® 5100 MCH Chipset:

- 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.

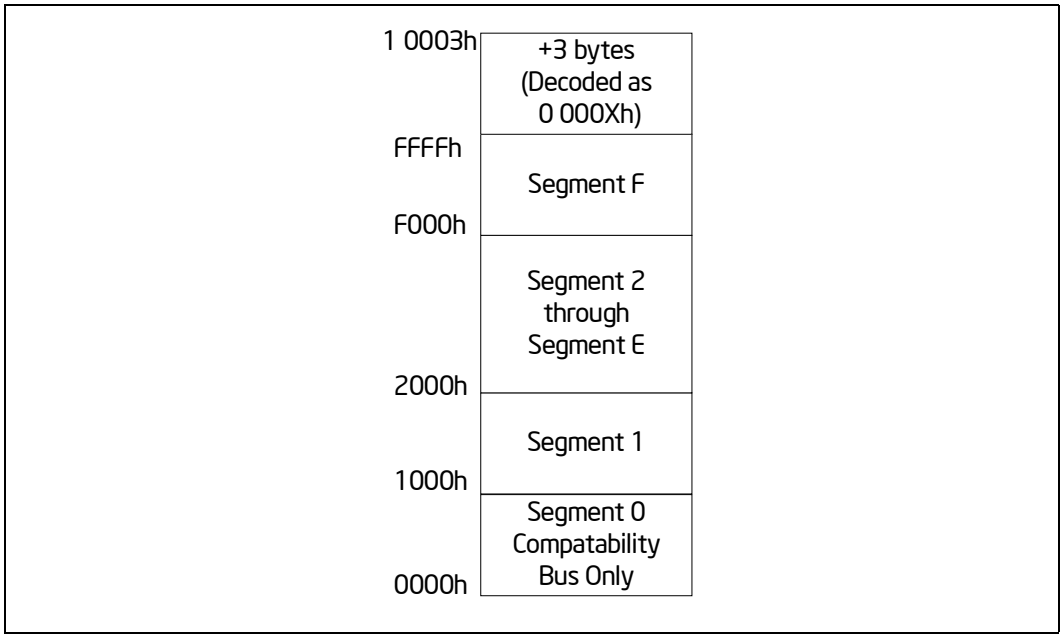
Historically, the 64 K I/O space actually was 64 K+3 bytes. For the extra 3 bytes, A#[16] is asserted. The Intel® 5100 MCH Chipset decodes only A#[15:3] when the request encoding indicates an I/O cycle. Therefore accesses with A#[16] asserted are decoded as if they were accesses to address 0 and are forwarded to the Compatibility Bus.

At power-on, all I/O accesses are mapped to the Compatibility Bus.

4.5.2 Outbound I/O Access

The Intel® 5100 MCH 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 one 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 17. System I/O Address Space



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 accesses to the MCH (i.e., CF8/CFC) are not supported and will receive a Master Abort response.

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*, Rev. 2.3.

5.0 Functional Description

This section describes each of the Intel® 5100 MCH Chipset 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 Intel® 5100 MCH Chipset provides the dual independent processor bus supporting the Dual-Core Intel® Xeon® processor 5100 series, Quad-Core Intel® Xeon® processor 5300 series, Dual-Core Intel® Xeon® processor 5200 series, and Quad-Core Intel® Xeon® processor 5400 series. This Intel® 5100 MCH Chipset-based platform supports a 771-land, FC-LGA4 (Flip Chip Land Grid Array 4) package for the processors.

The Intel® 5100 MCH Chipset also provides a single Front Side Bus (FSB) supporting the Intel® Core™2 Duo Processor T9400 as part of Uni-Processor (UP) system designs. This Intel® 5100 MCH Chipset-based platform supports 479-ball Micro-FCBGA (Flip Chip Ball Grid Array) and 478-pin Micro-FCPGA (Flip Chip Pin Grid Array) packages for the processors.

The Intel® 5100 MCH Chipset supports 1066/1333 MT/s FSB which is a quad-pumped bus running off a 266/333 MHz system clock. Each of the two processor FSB DIBs support peak address generation rates of 266/333 Million Addresses/second. Each FSB data bus is a quad pumped 64-bit interface which allows peak bandwidths of 8.5/10.7 GB/s (1066/1333 MT/s). The MCH supports 36-bit host addressing. The MCH supports up to 32 GB or 48 GB depending upon the mode of the processor memory address space. Host-initiated I/O cycles are decoded to PCI Express*, ESI interface or MCH configuration space. Host-initiated memory cycles are decoded to PCI Express*, ESI or system memory.

5.1.1 FSB Overview

The Intel® 5100 MCH Chipset 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 may complete deferrable transactions with either defer-replies or in-order responses. 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 address strobe (ADS) being sourced every other clock. The address bus generates a maximum bandwidth of 266/333 Million Addresses/second (MA/s).

The FSB data bus is quad pumped (4x) and supports peak bandwidths of 8.5/10.7 GB/s (1066/1333 MT/s). Parity protection is applied to the data bus. This yields a combined bandwidth of 17/21 GB/s for both FSBs.

Interrupts are also delivered via the FSB.

5.1.2 FSB Dynamic Bus Inversion

The Intel® 5100 MCH Chipset 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 87. 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 Intel® 5100 MCH Chipset drives data, each 16-bit segment is analyzed. If more than eight 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 Intel® 5100 MCH Chipset 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® 5100 MCH Chipset, FSB interrupts may originate from the processor on the system bus, or from a downstream device on the Enterprise South Bridge Interface (ESI). In the later case, the MCH drives the Interrupt Message Transaction onto the system bus.

In the Intel® 5100 MCH Chipset the ICH9R contains IOxAPICs, and its interrupts are generated as upstream ESI memory writes. Furthermore, *PCI Local Bus Specification*, Rev. 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 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. The write nature of the message "pushes" all applicable pre-interrupt traffic through to the Intel® 5100 MCH Chipset core, and the Intel® 5100 MCH Chipset core architecture guarantees that the subsequent APIC message cannot pass any posted data already within the Intel® 5100 MCH Chipset.

5.2 System Memory Controller

The Intel® 5100 MCH Chipset provides two DDR2 memory channels. Up to three DIMMs can be connected to each DDR2 channel (up to six DIMMs for the entire array) supporting up to 16 GB per channel for a total of 32 GB of memory space for 32GB_Mode or supporting up to 24 GB per channel for a total of 48 GB of memory space for 48GB_Mode.

The key features of the DDR memory interface are summarized in the following list.

- Two independent DDR2 Channels
- Supports DDR2-533 and DDR2-667 technology
- Minimum configuration 256 MB/single channel; 512 MB for two channels
- Maximum capacity 24 GB for single channel, 48 GB for two channels
- Supports 256 Mb to 2 Gb devices
- Supports x4 and x8 devices (RDQS mode for x8 devices)
- All writes are full cache line, data mask not supported
- Supports only Registered ECC DIMMs
- Supports up to six ranks/channel
- Intelligent Page-Hit policy
- Supports DIMM Self-refresh mode for low power (S3) mode
- Supports ECC and SDDC (x4 only)
- Address glitch detection built into ECC Algorithm
- Rank sparing capability on each channel
- Patrol and background scrubbing

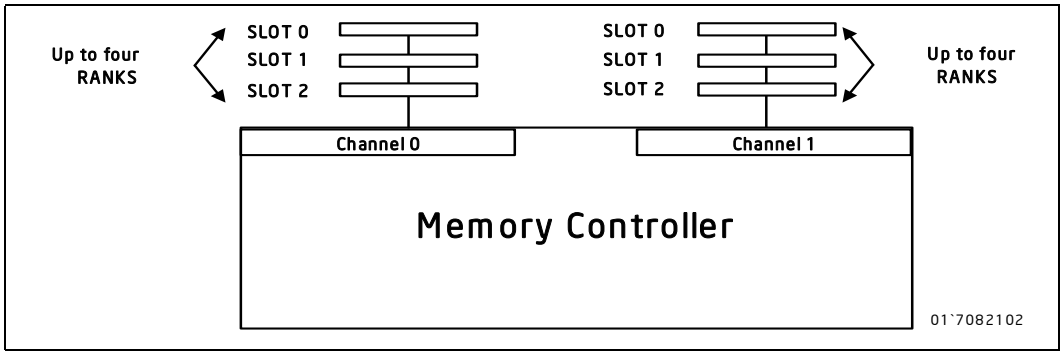
5.2.1 Memory Size

The minimum configuration is a single channel populated with 256 MB: one channel containing one DIMM with 256 MB of DDR2 devices (nine 32Mx8 devices). The MCH can support up to 48 GB of physical memory using the largest system memory configuration supported by 2 Gb DRAM devices. The maximum configuration allows 36-bit address space with 48 GB of accessible memory. Both channels support up to six memory ranks for a total of 48 GB installed system memory.

5.2.2 DIMM Technology and Organization

The Intel® 5100 MCH Chipset in standard 32 GB operation, supports up to four ranks per channel. The Intel® 5100 MCH Chipset's DIMM organization is shown in Figure 18, "Representative Memory System 32 GB Mode." Signal integrity simulations for the DIMM configurations are highly recommended for proper operation of the interface. The Intel® 5100 MCH Chipset, in 48 GB Mode, supports up to six ranks per channel, up to three single rank, up to three double (dual) rank, or up to one quad rank DDR2 Registered DIMM(s). For currently supported DIMM configurations, see the *Quad-Core and Dual-Core Intel® Xeon® Processor 5000 Sequence with Intel® 5100 Memory Controller Hub Chipset for Communications, Embedded, and Storage Applications – Platform Design Guide* or *Intel® Core™2 Duo Processors T9400 and SL9400 and Intel® 5100 Memory Controller Hub Chipset for Communications and Embedded Applications – Platform Design Guide*.

Figure 18. Representative Memory System 32 GB Mode



5.2.3 DIMM Configuration Rules

The Intel® 5100 MCH Chipset in standard 32 GB operation, supports a maximum of four ranks/channel for eight ranks overall. The Intel® 5100 MCH Chipset in 48 GB Mode, supports a maximum of six ranks/channel for twelve ranks overall. The permissible configurations for the ranks/channel are described in the below section.

5.2.3.1 Permissible Configurations

- All DIMMs must consists of x4 or x8 DDR2 DRAMs.
- The memory upgrade granularity is single DIMM. However, it must be noted that consecutive memory accesses typically alternate between channels to allow pre-charging for the next access; installing different numbers of ranks or amounts of memory between channels can force consecutive memory accesses to one channel effecting memory access times.
- Up to eight different types of DIMMs can be installed. The same type of DIMM does not need to be installed in each rank. A rank on one DIMM can be different than a rank on another DIMM.
- Electrical considerations may restrict DIMM placement
- Defective ranks in any position may be logically disabled by removing them from the DMIR configuration registers.
- Four-banked or eight-banked technology is supported
- Both channels will be operated at the same speed, 533 MHz or 667 MHz, so the slowest speed DIMM installed across both channels will be the operating speed for both channels. However, DIMMs with different timing parameters on different ranks can be installed together on a channel, but only one set of aggressively sufficient timings will be applied to all DIMMs on that channel. As a consequence, within a channel, faster DIMMs will be operated at timings (CAS/AL) supported by the slowest DIMM.

5.2.3.2 Memory Technology

Table 88. SDRAM Signal Allocations for Different Technologies

Technology	Organization	SDRAM Row bits	SDRAM Column lines	SDRAM Bank lines
256 Mb	4x8Mx8	RA12-RA0	CA9-CA0	B1-B0
	4x16Mx4		CA11, CA9-CA0	
512 Mb	4x16Mx8	RA13-RA0	CA9-CA0	
	4x32Mx4		CA11, CA9-CA0	
1 Gb	8x16Mx8		CA9-CA0	B2-B0
	8x32Mx4		CA11, CA9-CA0	
2 Gb	8x32Mx8	RA14-RA0	CA9-CA0	
	8x64Mx4		CA11, CA9-CA0	

5.2.4 Memory RAS

The Intel® 5100 MCH Chipset has many memory RAS features. In conjunction with Rank Sparing, Patrol and Demand Scrubbing, ECC and SDDC, a memory location can be poisoned. Memory mirroring is not a supported feature on the Intel® 5100 MCH Chipset.

5.2.4.1 Memory Sparing

At configuration time, a DIMM rank is set aside to replace a defective DIMM rank. Each channel can allow a single rank to be designated as spare by enabling rank sparing, setting SPCPC.SPAREN and designating the spare rank with the SPCPC.SPRANK bits. The spare rank must not be allocated in the DMIR registers.

When the correctable error rate for a failing DIMM rank reaches a pre-determined threshold (CERRCNT, SPCPS.LBTHR), an interrupt is issued to initiate a spare copy which copies the contents of the failed rank to the spare. While the copy engine is automatically reading locations from the failing DIMM rank and writing them to the spare (see Section 3.9.7.1, "SPCPC[1:0]: Spare Copy Control" and Section 3.9.7.2, "SPCPS[1:0]: Spare Copy Status"), 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. During the spare copy operation, the Memory Controller (MC) allocates approximately 20% of bandwidth to operations to read from the failed rank and write to the spare. At the completion of the copy, the failing DIMM rank is disabled and the spare 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.LIMITs are not updated.

Spare copy operation initiation and completion generate error messages (M20/M21) which can be used by software to note the failed rank etc. Patrol scrubbing is suspended during a spare copy operation.

Sparing occurs under the control of the CERRCNT register which maintains a running count of correctable errors for each rank. This counter uses a leaky bucket algorithm (SPCPS.LBTHR) to compensate for the anticipated level of correctable errors. Should the counter (for any one rank) reach the threshold value specified in SPCPC.SETH the spare copy operation is initiated for the failed rank. Only one rank can be spared so further sparing is disabled at that point.

The spare rank may be equal or larger than the failed rank, but not smaller. Using a smaller capacity rank to spare for a larger capacity rank is illegal and will result in undetermined behavior.

Sparing can be forced to activate by setting SPCPC.FORCE and designating the failed rank in SPCPC.FORCERANK.

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.

5.2.4.2 Data Poisoning in Memory

Data Poisoning in memory is defined as all zeroes in the code word (32B) except for the least significant bytes being FF00FFh. The Intel® 5100 MCH Chipset poisons a memory location based on the events described in Table 89, "Memory Poisoning Table."

Table 89. Memory Poisoning Table

Event	Correctable Error	UnCorrectable Error
Normal Memory Read	Correct Data to be given register The Intel® 5100 MCH Chipset logs M14 error. (Correctable demand data ECC Error) Correct Data to be written back to memory	Detects an Uncorrectable and logs a M10 error (Non-aliased uncorrectable 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 M16 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.
Rank Spare Copy	Correct Data to be written back to memory and log M15 error. (Correctable spare copy data ECC error)	If error persistent 1. Log and Signal M11 Error (Non Aliased uncorrectable spare copy data ECC error). 2. Poison Location in Spare rank

5.2.4.3 Patrol Scrubbing

To enable this function, the Memory Controller (MC), MC.SCRBEN configuration bit must be set. The scrub unit starts at DIMM Rank 0/Address 0 upon reset. Every 16k 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 etc) do not prolong the issuing of these transactions to DDR. Using this method, roughly 64 GB of memory behind the Intel® 5100 MCH Chipset can be completely scrubbed every day (estimate). Error logs include RAS/CAS/BANK/RANK.

Normally, one channel is scrubbed in its entirety before proceeding to the other channel. In the instance of a fail-down to non-redundant operation that off-lines the channel that was being scrubbed, the scrub pointer merely migrates to the other channel without being cleared. In this unique instance, the scrub cycles for that channel are incomplete.

5.2.4.4 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.

5.2.4.5 Normal Correction

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 completed.

5.2.4.6 Enhanced Correction

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 value at zero, saturation at the value of the MC.BADRAMTH configuration register field, increment on correctable errors, 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 it marks 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.

5.2.4.7 Single Device Data Correction (SDDC) Support

The Intel® 5100 MCH Chipset employs a single device data correction (SDDC) algorithm for the memory subsystem that will recover from a x4 component failure. The algorithm does not recover from a x8 component failure. The chip disable is a 32-byte two-phase code. 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 component failures.
- 100% Detection of all double x4 component failures.
  - — double bit errors
  - — double wire faults
  - — single wire fault in addition to single bit error
  - — single x4 device error in addition to single bit error
  - — single x4 device error in addition to single wire fault
  - — double x4 device error

Refer to Section 5.2.6.1, "Inbound ECC Code Layout for Memory Interface" for details.

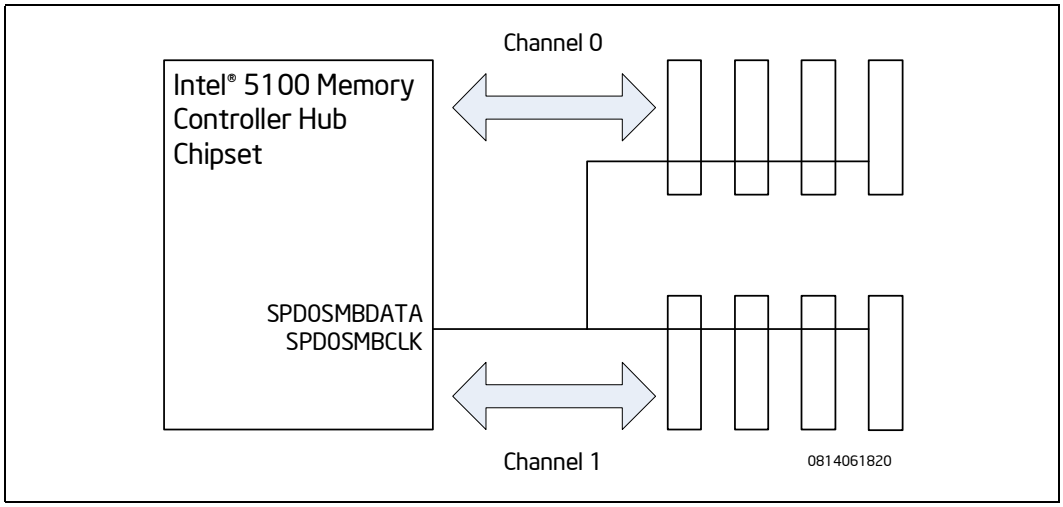
5.2.5 DIMM Memory Configuration Mechanism

Before any cycles to the memory interface can be supported, the Intel® 5100 MCH Chipset 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 Serial Presence Detect (SMBus) interface on the MCH. The SMBus interface is a two-wire bus used to extract the DRAM type and size information from the Serial Presence Detect port on the DIMMs.

The DIMMs contain a 6-pin Serial Presence Detect interface, which includes SCL (serial clock), SDA (serial data), and SA[2:0] (serial address). Devices on the SMBus bus have a 7-bit address. For the DIMMs, the upper four bits are fixed at 1010. The lower three bits are strapped via the SA[2:0] pins. SCL and SDA are connected to the respective SPD0SMBDATA, SPD0SMBCLK pins on the MCH, see Figure 19, "Connection of DIMM Serial I/O Signals."

The Intel® 5100 MCH Chipset integrates a 100 kHz SPD controller to access the DIMM SPD EEPROM's. There is one SPD port. SPD0SMBDATA, and SPD0SMBCLK are defined for channel 0 and channel 1. There can be a maximum of eight SPD EEPROM's associated with an SPD bus. Therefore, the SPD interface is wired as indicated in Figure 19, "Connection of DIMM Serial I/O Signals."

Figure 19. Connection of DIMM Serial I/O Signals



Board layout must map chip selects to SPD Slave Addresses as shown in Table 90, "SPD Addressing." The slave address is written to the **SPDCMD** configuration register (see Section 3.9.11.2, "SPDCMD: Serial Presence Detect Command Register"). For details on the use of SA[2:0] in generating an SPD bus address related to DIMM EPROM access see Section 5.20.6, "DDR2 DIMM SPD0 SMBus Interface".

Table 90. SPD Addressing

Channel	SLOT	Slave Address SA[2:0]	Channel	SLOT	Slave Address SA[2:0]
0	0	0	1	0	4
	1	1		1	5
	2	2		2	6
	3	3		3	7

5.2.6 DRAM ECC Code

The Intel® 5100 MCH Chipset supports the DRAM device failure correction code (SDDC aka SECC). As applied by the Intel® 5100 MCH Chipset, this code has the following properties:

- Correction of any x4 device failure (e.g. stuck at 0)
- 100% detection of x4 single device failure in addition to single bit soft error in another device.
- Detection of all two wire faults on the DIMMs. This includes any pair of single bit errors.

See also Section 5.2.4.7, "Single Device Data Correction (SDDC) Support".

The rank address is also encoded into the ECC algorithm. This allows a degree of protection against address line glitches, but not consistent stuck-at type address line faults.

5.2.6.1 Inbound ECC Code Layout for Memory Interface

When the Intel® 5100 MCH Chipset transfers data to and from the MCH DDR2 memory, an ECC encoding scheme is used to minimize data corruption. The ECC coding scheme uses code words based upon the unit of access, a cache line. Each 64-byte cache line plus eight bytes of parity form two ECC code words. A cache line on the Intel® 5100 MCH Chipset requires a burst of eight DDR2 data bus transfers (BL=8). Each half burst or 4x nine bytes comprises a code word. Each code word can then be considered a half cache line. In a burst of eight, the first four encoded values (72 bits each) form the first code word, and the second four encoded values form the second code word.

The ECC code layout is systematic, in other words, the data is separated from the check-bits rather than all being encoded together. It consists of 32 8-bit data symbols and four 8-bit check-bit symbols (72 nibbles, 288 bits). The symbols are further broken down into nibbles.

The ECC code corrects any two adjacent 8-bit symbols in error. The code layout ensures that the four consecutive nibbles transferred, during a burst of four DDR2 data bus transfers, form 16 contiguous bits or two adjacent symbols in a single code word. The symbols are arranged so that the data from every x4 DRAM is mapped to two adjacent symbols of a single code word, so the failure of any single x4 DRAM device can be corrected. If the device corresponding to nibble 0 fails, nibbles 0–3 (16 bits, two symbols) in the first code word, and nibbles 72–75 in the second code word, will be affected and are correctable by the code.

The ECC code is transferred across the DDR2 channel, or equivalently, the DDR2 data bus in symbols. The symbols are mapped to DRAM bits in the DIMM for cache line transfers. The Check Bytes signals as defined by the Intel® 5100 MCH Chipset are not limited to ECC data bytes in this implementation; all of the Check Byte signals and Data

signals are equally utilized for data and the ECC code layout. The same mapping of symbols to data and code bits applies to Northbound and Southbound data for each of the two independent DDR2 memory channels of the MCH.

The Intel® 5100 MCH Chipset supports cache line or 64-byte aligned accesses as well as 32-byte aligned cache line accesses. In 32-byte aligned accesses, data is returned critical word first; the MCH returns the addressed cache line but with the two halves reversed in time order so that the "addressed" half cache line arrives first – the addressed half cache line is referred to as the "critical" word or half of the cache line.

5.2.7 DDR2 Protocol

5.2.7.1 Posted CAS

Posted CAS timing is used. RAS to CAS delay will be 3 cycles in all cases.

5.2.7.2 Refresh

Regardless of the number of DIMMs installed, each rank will get a minimum of one refresh every eight periods defined by the DRTA.TREF configuration register field. The refreshes cycle through all eight DIMM ranks.

5.2.7.3 Access Size

All memory accesses are 64 bytes in size, issued with a burst length of 8.

5.2.7.4 Transfer Mode

Each DIMM is programmed to use a burst-length of 64 bytes (eight transfers) across the channel. The Mode Register of each DIMM must be programmed for a burst length of eight, and interleave mode.

5.2.7.5 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 (i.e., 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.

5.2.8 Memory Thermal Management

The Intel® 5100 MCH Chipset implements an adaptive throttling methodology along with electrical throttling to limit the number of memory requests to the DIMMs. The methodology is comprised of the following:

1. Activation throttling: Consists of open loop throttling of activations on the DDR2 interface. Open Loop Global Activation Control limits requests when the number of activations crosses an event threshold in a large time window.
2. Electrical throttling is used to prevent silent data corruption by limiting the number of activations per rank within a short sliding window period.

5.2.8.1 Closed Loop Thermal Activate Throttle Control

Since there is no means to obtain the DIMM temperatures, Closed Loop Thermal Throttling is not supported.

5.2.8.2 Open Loop Global Throttling

In the open loop global throttling window scheme, the number of activations per rank is counted for a large time period called the "Global Throttling window". The Global throttling window is chosen as an integral multiple of 1344 MCH core clocks (the thermal throttling window or throttling activation window). The internal core clocks are used for calculating the windows and not the DDR clocks. Under normal operating conditions, the Global Throttling Window is 0.65625×2^{25} clocks in duration which translates to 16384x1344 clocks, approximately 66.06 ms for DDR2-667 and 82.58 ms for DDR2-533.

During the Global throttling window, the number of activations is counted for each rank (24-bit counters are required). Normally the throttling logic throttles at a "low" level using the THRTLOW register. However, if the number exceeds the number indicated by the GBLACT.GBLACTLM register defined in Section 3.9.2, "Memory Throttling Control Registers", then the THRTSTS[1:0].GBLTHRT bit is set for the respective channel at the end of the current Global Throttling window, causing the activation throttling logic to throttle at a "high" level using the THRTHIGH register. The THRTSTS[1:0].GBLTHRT will remain active until 16 (or two) global throttling windows in a row have passed without any rank exceeding the GBLACTLM value; whereupon, the Memory Controller indicates the end of the high throttling period by clearing the THRTSTS[1:0].GBLTHRT register field and returning to the use of the THRTLOW register for activation throttling.

If part way through the count of 16 (or two) 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 two) global throttle windows throttling at the THRTHIGH level.

If the number of activations to a rank exceeds the limit specified by the register within a given throttling activation window, then further requests are blocked and CKE will be driven low to the affected rank for the remainder of the throttling activation window.

The global throttling window prevents shorts peaks in bandwidth from causing activation throttling when there has not been sufficient DRAM activity over a long period of time to warrant throttling at the high level.

5.2.8.3 Global Activation Throttling Software Usage

In practice, the throttle settings for THRTHIGH are likely to be set by BIOS such that the memory controller throttle logic will actually prevent the GBLACT limit from being exceeded. The THRTLOW is often used as a Global Throttle Window, where the GBLACT.GBLACTLM is exceeded, causing the MC to use a larger throttling period THRTHIGH for 16 (or two) global windows. During each of those global windows, GBLACT limit is not exceeded, because the throttling will prevent it from being exceeded. After 16 (or two) global throttling windows, it switches back to THRTLOW, and on the next global window GBLACT is again exceeded, causing another 16 (or two) windows. Hence, a cumulative pattern of 16, 1, 16, 1 (or 2, 1, 2, 1) global throttling windows occur preventing excessive heat dissipation in the DIMMs by prolonging the throttle period.

Note: It should be mentioned that the throttling control policies implemented on the Intel® 5100 MCH Chipset use 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 THRTSTS.THRMTHRT register bit value when dealing with different technologies and speeds.

5.2.8.4 Dynamic Update of Thermal Throttling Registers

In general, the Intel® 5100 MCH Chipset registers should not be 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® 5100 MCH Chipset thermal throttling registers where it is desirable for BIOS or special OEM software in BMC to exercise dynamic control on throttling for open 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 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.

5.2.8.5 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/software to take preemptive action in adjusting the throttling, for example 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® 5100 MCH Chipset is updated with the new throttle values. These requirements are captured below.

5.2.8.6 Dynamic Change Operation for Open Loop Thermal Throttling (OLTT)

The Intel® 5100 MCH Chipset memory throttle control register affected by OLTT include THRTHIGH (T2), THRTLOW (T1) AND GBLACT.

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 activations exceed the GBLACT.GBLACTLM in a global throttling window, OLTT is entered and GBLTHRT is set by the Intel® 5100 MCH Chipset for 16 consecutive global throttling windows (irrespective of the new parameters) as described in Section 5.2.8.2, “Open Loop Global Throttling”. Note that OLTT is NOT history-based algorithm. Hence if software assigns new values to THRTLOW or THRHIGH values at some point in time, the Memory Controller (MC) cluster will update the registers and use the new values for limiting the activations immediately via THRMTHRT register for 16 consecutive global throttling windows.

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.

5.2.8.7 Disabling Open Loop Throttling

The following registers in the Intel® 5100 MCH Chipset can be initialized to disable throttling if software desires to turn off throttling.

- THRTHIGH.THRTHIGHLM = 0
- THRTLOW.THRTLOWLM = 0
- GBLACT.BLACTLM = 0

Above changes force Open loop throttling off.

5.2.9 Electrical Throttling

Electrical throttling is a mechanism that limits the number of activations (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 3.9.1.6, "MTR[1:0][5:0] - Memory Technology Registers". These bits occur on a per rank basis per channel as to whether electrical throttling should be used.

The per rank electrical throttling limit is four activations per 37.5 ns window (JEDEC consensus) and is summarized in Table 91, "Electrical Throttle Window as Function of DIMM Technology" for various DIMM technologies. For DDR2, the number of activations per T_{RC} (RAS cycle time) is around $4xT_{RC}/T_{FAW}=4x60/37.5=6.4$.

Table 91. Electrical Throttle Window as Function of DIMM Technology

DIMM Modes	Intel® 5100 MCH Chipset Core: DIMM clock Ratio	Electrical Throttle Window[1] (memory clocks per rank)
DDR2-533	All	10
DDR2-667	All	13
Conservative (safe mode)	All	20

1. Maximum four activations per rank is allowed within the window.

The MTR.ETHROTTLE registers bit enables/disables electrical throttling. The DRTA.TFAW configuration register field limits the number of activations per sliding electrical throttle window. A shift register whose length (max 20) is determined by TFAW is used to record activations to each rank. New activations shift in a '1' while old activations are "dropped" at the end of the TFAW window. If the limit is reached, then further activations 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 DDR2-667, then it can allow four activations within the last 13 clocks, the remaining seven bits are masked (forced to 0) so they do not prevent activations.

5.2.10 Normal Self-refresh Entry

The Intel® 5100 MCH Chipset puts the DIMMs into self-refresh for S3 power mode under control of the power management system (in response to a BIOS request). The sequence of operations involved in the operation is as follows.

1. MCH Power Management asserts Queue Flush - at this point, the system is committed to a power cycle (assertion of power good reset)
2. Memory Controller (MC) responds by blocking all new requests
3. MC starts to flush posted write Queue
4. Both read and write Queues are drained

5. Memory Controller (MC) asserts Queues Empty when all Queues have been drained
6. MCH Power Management requests Self-refresh
7. MC initiates self-refresh command sequence on all ranks
8. MC issues Self-refresh Ack when all ranks are in self-refresh
9. Power can be removed from the MC core logic.

This sequence cannot be broken in any way - once the Queue Flush is asserted, a hard reset will be required before resuming normal operation.

5.3 Interrupts

The Intel® 5100 MCH 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 Signaled Interrupts (MSI). In this document, I/O(x)APIC is an interrupt controller that is found in the ICH9R component of the chipset.

The legacy 8259 functionality is embedded in the ICH9R component. The Intel® 5100 MCH 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.

5.4 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® 5100 MCH Chipset-based 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 in each of the two request phases of the FSB transaction, request phase A (REQa) and request phase B (REQb). The values of the FSBxREQ[4:0]# signals are FSBxREQa[4:0]=01001b and FSBxREQb[4:0]=11100b for the respective request phases. For more information, see FSB0A[35:3]#/FSB1A[35:3]# and FSB0REQ[4:0]#/FSB1REQ[4:0]# signal descriptions in Section 2.1, "Processor Front Side Bus Signals."

The naming convention throughout this section indicates 'a' for request phase A, 'b' for request phase B and x indicates the FSB bus for which the interrupt is present. As an example, FSBxAb[5]# refers to the FSB0/FSB1 (see Table 93) address pin 5, FSBxA[5]#, during the second request phase of the FSB transaction protocol, request phase B.

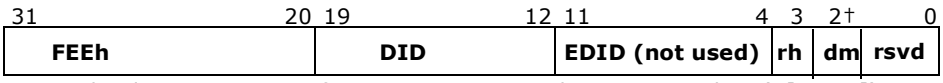
The XAPIC architecture provides for lowest priority delivery, through interrupt redirection by the chipset. If the redirectable hint bit is set in the XAPIC message, the chipset may redirect the interrupt to another processor. Note that redirection of interrupts can be to any processor on either Processor 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 5.4.3, "Interrupt Redirection".

5.4.1 XAPIC Interrupt Message Format

Interrupt messages have an address of 000_FEEz_zzzYh. The 16-bit "zzzz" field (destination field) determines the target to which the interrupt is being sent. The Y field is mapped to A[3] (redirectable interrupt) and A[2] (destination mode). Figure 20, "XAPIC Address Encoding" shows the address definition in IA-32 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 20, "XAPIC Address Encoding" applies to both the PCI Express* bus and processor bus. Note that the current assumption is that no conclusions can be made 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® 5100 MCH Chipset will ensure that each interrupt is seen on both FSBs.

Figure 20. XAPIC Address Encoding



DID: 8-bit destination ID. Software may assign each ID to any value. (A[19:12])
EDID: Not used, is a reserved field in the Processor EHS.
rh: redirection hint bit, 0=directed, 1=redirectable (A[3])
dm: destination mode, 0=physical, 1=logical (A[2])
† PCI/PCI Express* transaction encoding. Copied to FSBxAb[5]# 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 92. 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

5.4.2 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 FSBxAb[5]# on the processor bus.

5.4.2.1 Physical Destination Mode (XAPIC)

In physical mode, the APIC ID is eight bits, supporting up to 255 agents. Each processor has a Local APIC ID Register where the lower five bits are initialized by hardware (Cluster ID=ID[4:3], Bus Agent ID=ID[2:1], thread ID=ID[0]). The upper three bits default to 0's at system reset. These values can be modified by software. The Cluster ID is set by address bits FSBxA[12:11]# during reset. By default, the Intel® 5100 MCH Chipset will drive FSBxA[12:11]# to '00 for FSB0, and '01 for FSB1. The value driven on bit FSBxA[12]# during reset can be modified through the Section 3.8.5.3, "POC - Power-On Configuration Register" on the Intel® 5100 MCH Chipset.

The Intel® 5100 MCH Chipset will not rely on the cluster ID or any other fields in the APIC ID to route interrupts. The Intel® 5100 MCH 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 eight 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 5.4.3.2, "Redirection Algorithm".

5.4.2.2 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® 5100 MCH Chipset 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® 5100 MCH 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 the Intel® 5100 MCH 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.

5.4.2.3 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 ICH9R. Table 93, "Intel® 5100 Memory Controller Hub Chipset XAPIC Interrupt Message Routing and Delivery" shows the routing rules used for routing XAPIC messages in a Intel® 5100 MCH Chipset. Table 93, "Intel® 5100 Memory Controller Hub Chipset XAPIC Interrupt Message Routing and Delivery" is valid for both broadcast and non-broadcast interrupts.

Table 93. Intel® 5100 Memory Controller Hub 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 Section 5.4.3) is performed by the Intel® 5100 MCH Chipset and is delivered to both FSBs.

5.4.3 Interrupt Redirection

The XAPIC architecture provides for lowest priority delivery through interrupt redirection by the Intel® 5100 MCH 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.

5.4.3.1 XTPR Registers

To accomplish redirection, the Intel® 5100 MCH Chipset 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® 5100 MCH Chipset that destination field of the message is flat or cluster (note that the XAPIC message indicates whether the destination mode is physical or logical). This bit is also updated by an XTPR_Update transaction (FSBxAa[3]#) and is also programmable through configuration space. The default logical mode at reset is flat.

More details on the Intel® 5100 MCH Chipset XTPR registers are described in the XTPR register definition in Section 3.8.4, "Interrupt Redirection Registers"

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.

5.4.3.2 Redirection Algorithm

Redirection is performed if an interrupt redirection hint bit (A[3]) is set. See Figure 20 and Section 5.4 for naming conventions. Below is the algorithm used in determining to which processor the interrupt will be redirected.

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 (A[2] for I/O, FSBxAb[5]# for IPIs) and XTPR[0].CLUSTER 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 that 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. CLUSTER: If Destination Mode = 1 (A[2] for I/O, FSBxAb[5]# for IPI's) and SC.XTPR[0].CLUSTER is enabled (1) in the XTPR, then this is Clustered-Logical Destination Mode. (Otherwise, proceed to the next step).
   To select the arbitration pool:
   If (A[19:16] (DID[7:4]) == XTPR[n].LOGID[7:4])
   AND ((A[15:12] (DID[3:0]) AND XTPR[n].LOGID[3:0]) >0h)

AND XTPR[n].TPREN =1)
then XTPR[n] is included in the arbitration pool.

4. PHYSICAL: If Destination Mode = 0 (A[2] for I/O, FSBxAb[5]# for IPIs), then this is Physical Destination Mode. All enabled XTPRs are included in the arbitration pool.
5. If there are no XTPRs in the arbitration pool, then forward to FSB with A[3]=0, but otherwise "without modification". Otherwise, continue to the next step.
6. XTPRs in the pool are categorized into four priority buckets depending on the priority. The priority bucket levels are defined by register bits BUCKET[0:2] in the REDIRCTL register. The BUCKET fields are used to set priority group membership.
   If (0 <= XTPR.PRIORITY < BUCKET0) then priority bucket = 0
   If (BUCKET0 <= XTPR.PRIORITY < BUCKET1) then priority bucket = 1
   If (BUCKET1 <= XTPR.PRIORITY < BUCKET2) then priority bucket = 2
   If (BUCKET2 <= XTPR.PRIORITY < 16) then priority bucket = 3
7. All XTPRs in the arbitration pool are compared. The XTPR register with the lowest priority bucket value (0=lowest, 3=highest) is the "winner".
8. If more than one XTPR register in the arbitration pool has the same lowest priority bucket value, LRU arbitration logic will pick an XTPR that was not recently picked.
9. The "winning" XTPR register provides the values to be substituted in the FSBxAa[19:12]# field of the FSB Interrupt Message Transaction driven by the Intel® 5100 MCH Chipset. FSBxA[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 (A[3]).

5.4.3.3 XTPR Update

The Intel® 5100 MCH Chipset decodes FSB XTPR_Update transactions for interrupt redirection based on lowest priority bucket. Due to specific implementation and timing issues in the Intel® 5100 MCH Chipset, when an XTPR_update transaction and a pending interrupt (e.g., inbound MSI from PCI Express* port) happens concurrently, the interrupt may be delivered to the prior processor thread (i.e., the CPU before XTPR_update took effect instead of the newly redirected CPU). This is not perceived to be a functional problem since the Intel® 5100 MCH 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.

5.4.4 End Of Interrupt (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® 5100 MCH Chipset will broadcast the EOI transaction to all I/O(x)APIC's. The SCPEXCTRL.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 5.5.2, "Hardware IRQ IOxAPIC Interrupts".

Note: The Intel® 5100 MCH Chipset will translate the EOI on the FSB into an EOI TLP message type on the PCI Express*/ESI ports.

5.5 I/O Interrupts

For I/O interrupts from the ICH9R 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 ICH9R) 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 5.4.1, "XAPIC Interrupt Message Format". Therefore, to all components other than the Intel® 6700PXH 64-bit PCI Hub, ICH9R, and the processor, an interrupt is an inbound write following the format mentioned in Section 5.4.1, "XAPIC Interrupt Message Format". The Intel® 5100 MCH 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_0000h to FECF_FFFFh range is used to communicate with the IOAPIC controllers in the Intel® 6700PXH 64-bit PCI Hub or ICH9R.

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.

5.5.1 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*, Rev. 2.3 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.

5.5.2 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 ICH9R. 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® 5100 MCH 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® 5100 MCH Chipset supports the INTA (interrupt acknowledge) special bus cycle for legacy 8259 support. These are routed to the compatibility ICH9R in the system. The INTA will return data that provides the interrupt vector.

5.5.3 Message Signaled Interrupts

A second mechanism for devices to send interrupts is to issue the Message Signaled Interrupt (MSI) introduced in the *PCI Local Bus Specification*, Rev. 2.3. 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® 5100 MCH Chipset over the PCI Express* ports.

MSI is expected to be supported by the operating systems when the Intel® 5100 MCH Chipset is available. A Intel® 5100 MCH Chipset 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 Section 5.5.4, "Non-MSI Interrupts - "Fake MSI"."

5.5.4 Non-MSI Interrupts - "Fake MSI"

For interrupts coming through the Intel® 6700PXH 64-bit PCI Hub and ICH9R 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 ICH9R since it is always present in a system. Although the Intel® 5100 MCH 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 PCI Express* endpoint and the I/OxAPIC.

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® 5100 MCH Chipset and is redirected down the appropriate PCI Express* port following the Intel® 5100 MCH 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® 5100 MCH Chipset core, and the Intel® 5100 MCH Chipset core architecture guarantees that the subsequent APIC message cannot pass any posted data already within the Intel® 5100 MCH Chipset.

5.6 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.
- IA-32 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® 5100 MCH Chipset completes the interrupt normally and forwards the interrupt to the other bus.
- For redirectable XAPIC interrupts, the Intel® 5100 MCH Chipset will generate an interrupt message to both processor buses the Intel® 5100 MCH 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 = FFh), the Intel® 5100 MCH Chipset will forward the broadcast interrupt to the other processor bus.
- Interrupts are not deferred.

5.6.1 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.

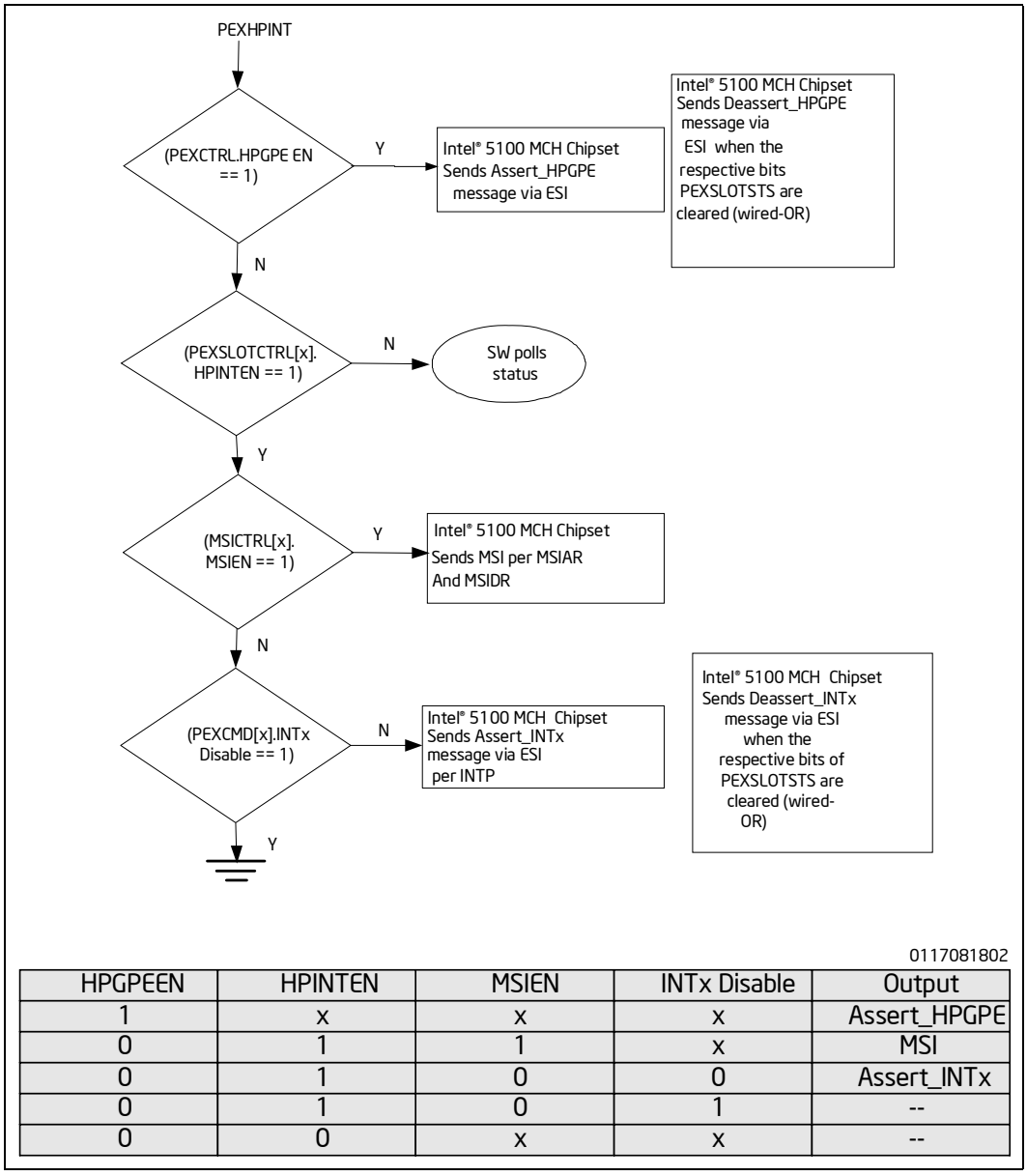
There are no ordering requirements between IPIs. There are no ordering requirements between IPIs and subsequent request. The IPIs are claimed on the FSB (front side bus) and are not deferred. Therefore, software must not rely on the ordered delivery between the IPI and subsequent transactions. If ordering 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.

5.7 Chipset Generated Interrupts

The Intel® 5100 MCH Chipset 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 following is a list of interrupts that can be generated.

1. Chipset error - The Intel® 5100 MCH Chipset asserts appropriate ERR pin, depending on severity. This can be routed by the system to generate an interrupt at an interrupt controller (the Intel® 5100 MCH Chipset pins ERR[2:0], MCERR). The ERR[0] pin denotes a correctable and recoverable error. The ERR[1] pin denotes an uncorrectable error from the Intel® 5100 MCH Chipset. The ERR[2] pin denotes a fatal error output from the Intel® 5100 MCH Chipset.
2. PCI Express* error - The Intel® 5100 MCH Chipset asserts appropriate ERR pin, depending on severity. This can be routed by the system to generate an interrupt.
   a. The Intel® 5100 MCH Chipset can receive error indications from the PCI Express* ports. These are in the form of inbound ERR_COR/UNC/FATAL messages. The Intel® 5100 MCH Chipset will assert the appropriate ERR signal just like any internal Intel® 5100 MCH Chipset error.
3. PCI Hot Plug* - The Intel® 5100 MCH Chipset sends Assert_HPGPE (Deassert_HPGPE) or generates an MSI or a legacy interrupt on behalf of a PCI Hot Plug* event.
   a. The Intel® 5100 MCH Chipset generated PCI Hot Plug* event such as PresDet change, Attn button, MRL sensor changed, power fault, etc. Each of these events have a corresponding bit in the PCI 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 21 for the PCI Hot Plug* interrupt flow priority.
   b. PCI Hot Plug* event from downstream. This could be either an MSI or a GPE message.
   — MSI: Handled like a normal MSI interrupt (see Section 5.5.3)
   — GPE message: Upon receipt of a Assert_GPE message from PCI Express*, the Intel® 5100 MCH Chipset will send Assert_GPE signal to the ESI port. To generate an SCI (ACPI), this signal will be routed to the ICH9R appropriate GPIO pin to match the GPE0_EN register settings. When the PCI Hot Plug* event has been serviced, the Intel® 5100 MCH Chipset will receive a Deassert_GPE message. At this point the Intel® 5100 MCH Chipset can send 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 the Intel® 5100 MCH Chipset 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® 5100 MCH Chipset 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 ICH9R. No action is required from the Intel® 5100 MCH Chipset.

Figure 21. PCI 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	--

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 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, the Intel® 5100 MCH Chipset 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 ICH9R 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® 5100 MCH Chipset can send Deassert_PMEGPE to ESI port.

Table 94, "Chipset Generated Interrupts" summarizes the different types of chipset generated interrupts that were discussed. Although the interrupt and software mechanism is flexible and can be changed depending on how the system is hooked up, for reference, Table 94, "Chipset Generated Interrupts" also describes what software mechanism is expected to be used.

Table 94. Chipset Generated Interrupts

Source	Signaling Mechanism	Intel® 5100 MCH Chipset Signal Method	Expected Software Mechanism
Chipset Error	Intel® 5100 MCH Chipset registers	ERR[2:0], MCERR, ICH9R Reset	Any
PCI Express* Error	PCI Express* ERR_COR/UNC/ FATAL message	ERR[2:0], MCERR, ICH9R Reset	Any
PCI Hot Plug* (PresDet chg, Attn button, etc.)	Intel® 5100 MCH Chipset registers For card-these registers are set via the VPP/SM bus interface. For module- these registers are set by inband PCI Hot Plug* messages.	MSI or Assert_INTx, Deassert_INTx, or Assert_HPGPE, Deassert_HPGPE	SCI->ACPI or MSI
PCI Hot Plug* from downstream device	MSI	MSI interrupt (processor bus)	MSI
PCI Hot Plug* from downstream device (non-native, Intel part)	PCI Express* Assert/Deassert GPE	Assert_GPE, Deassert_GPE to ESI	SCI->ACPI
PCI Hot Plug* from downstream device (non-native, non-Intel part)	Sideband signals directly to ICH9R	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

5.7.1 Intel® 5100 Memory Controller Hub Chipset Generation of MSIs

The Intel® 5100 MCH Chipset generates MSIs on behalf of PCI Hot Plug* events if SC.MSICTRL.MSIEN is set. Refer to Figure 21. The Intel® 5100 MCH Chipset will interpret PCI Hot Plug* events and generate an MSI interrupt based on SC.MSIAR and SC.MSIDR registers. When the Intel® 5100 MCH Chipset detects any PCI Hot Plug* event, it will generate an interrupt transaction to both processor buses. The address will be the value in SC.MSIAR. The data value will be the value in MSIDR.

Internal to the Intel® 5100 MCH Chipset, 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.

5.7.1.1 MSI Ordering in Intel® 5100 Memory Controller Hub Chipset

Ordering issues on internally generated MSIs could manifest in the Intel® 5100 MCH Chipset if software/device drivers rely on certain usage models, e.g. interrupt rebalancing, PCI 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, i.e., 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 the Intel® 5100 MCH Chipset PCI Express* (internal) configuration spaces will not generally guarantee ordering of internal MSIs from a root port/DMA Engine device as required. This is because the Intel® 5100 MCH Chipset uses a configuration ring methodology which houses the registers for the various PCI Express* ports, Memory Controller (MC), DMA Engine, Dfx etc. 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® 5100 MCH Chipset will implement a "pending MSI signal" that is broadcast from the MSI/PCI Hot Plug* blocks to the Coherency Engine (CE) 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 are 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® 5100 MCH Chipset 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® 5100 MCH Chipset. 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, i.e., 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.

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.

5.8 Software Guidance for MSI Handling

There are two conditions under which the MCH expects software to handle Message Signaled Interrupts (MSI) appropriately. The first is if one or more interrupt status bits are set to '1' and a new bit gets set to '1.' The MCH will send an MSI for the new bit. This may cause extraneous Interrupt Service Routine (ISR) calls. The second condition occurs when one or more interrupt status bits are set to '1' and software clears some (but not all) bits. The MCH will not send an MSI for the remaining uncleared bits. This may cause lost interrupts.

ISRs should record all events in the status registers that they process and clear all events detected. There should be no lingering status upon ISR exit. It is the software's responsibility to handle MSIs.

If an ISR does not follow the requirement to read the Interrupt Control register (INTRCTRL), and the ISR is called twice back-to-back such that it reads Attention Status register (ATTNSTATUS) both times, a second MSI could overwrite the first one and hang the system.

5.9 Legacy/8259 Interrupts

8259 interrupt controller is supported in Intel® 5100 MCH Chipsets. 8259 interrupt request is delivered using the interrupt group sideband signals LINT[1:0] (aka., 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® 5100 MCH Chipset forwards the interrupt acknowledge to the ICH9R where the active 8259 controller resides.

The Intel® 5100 MCH 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® 5100 MCH Chipset which will forward these interrupts to the ICH9R.

The Intel® 5100 MCH 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 ICH9R. 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 ICH9R (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® 5100 MCH Chipset PCI Express* port will only send a maximum of four 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® 5100 MCH Chipset core components do not interact with the LINT[1:0] and SMI signals. They are present on the ICH9R and the processor. The Intel® 5100 MCH Chipset interrupt signals described in Section 5.7, "Chipset Generated Interrupts" can be routed to the ICH9R to generate an SMI interrupt. Similarly SCI interrupts can be generated by routing the Intel® 5100 MCH Chipset interrupt signals to the appropriate ICH9R pin.

5.10 Interrupt Swizzling

The Intel® 5100 MCH Chipset has interrupt swizzling logic to rebalance and distribute inbound PCI Express* interrupts for performance and load balancing considerations. Register INTxSWZCTRL[7:2,0] described in Section 3.8.8.35, "INTXSWZCTRL[7:2,0]: PCI Express* Interrupt Swizzle Control Register" provides software/BIOS the ability to swizzle the PCI Express* interrupt (INTx) from each PCI Express* port and remap them to a different interrupt pin.

INTA (as depicted in Figure 23, "No Interrupt Swizzle") is usually overloaded since it is reserved per the PCI/PCI Express* specification for PCI/PCI Express* devices (if the device uses interrupts), so it is of particular concern. Figure 22, "Interrupt Swizzle" depicts interrupt swizzling where INTx is distributed. Figure 22 and Figure 23 depict examples of interrupt swizzling and are *only* used for illustration purposes, as system board interrupt routing is platform-specific.

For optimal system performance, it is recommended that the system BIOS utilizes the INTXSWZCTRL[7:2,0] register to achieve a more balanced PCI Express* interrupt distribution. Note that the PCI interrupt routing table in the system BIOS needs to be modified to match the particular swizzling scheme required for the specific system design.

Figure 22. Interrupt Swizzle

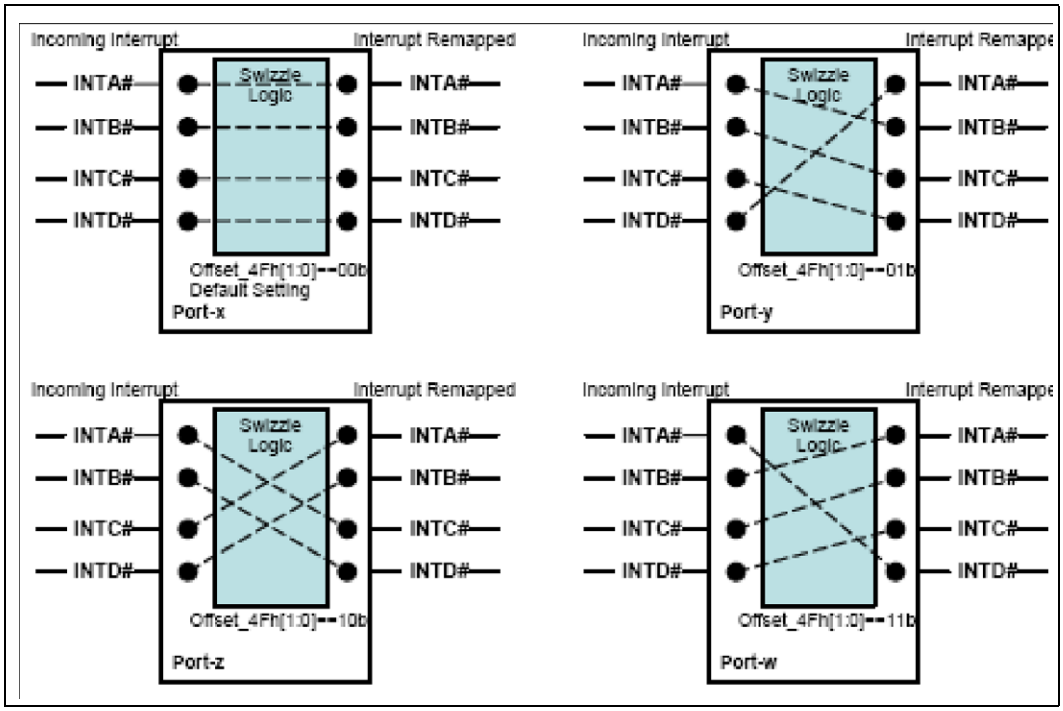
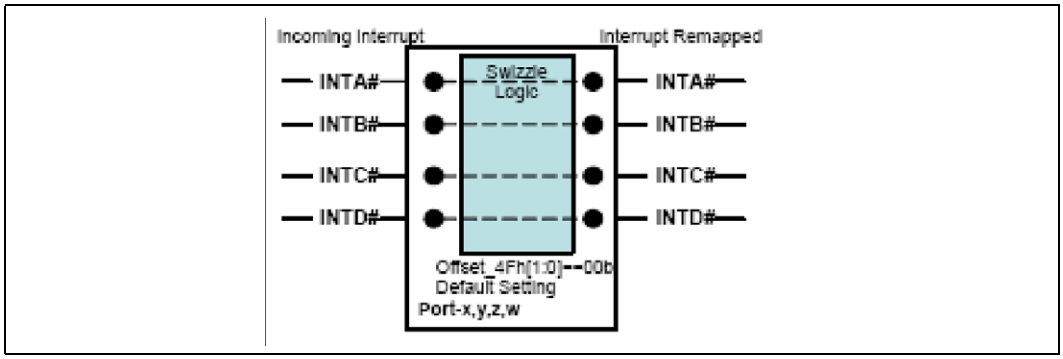


Figure 23. No Interrupt Swizzle



For additional information, please refer to document number 314337, *Interrupt Swizzling Solution for Intel® 5000 Chipset Series-based Platforms – Application Note* available on http://www.intel.com/.

5.11 Interrupt Error Handling

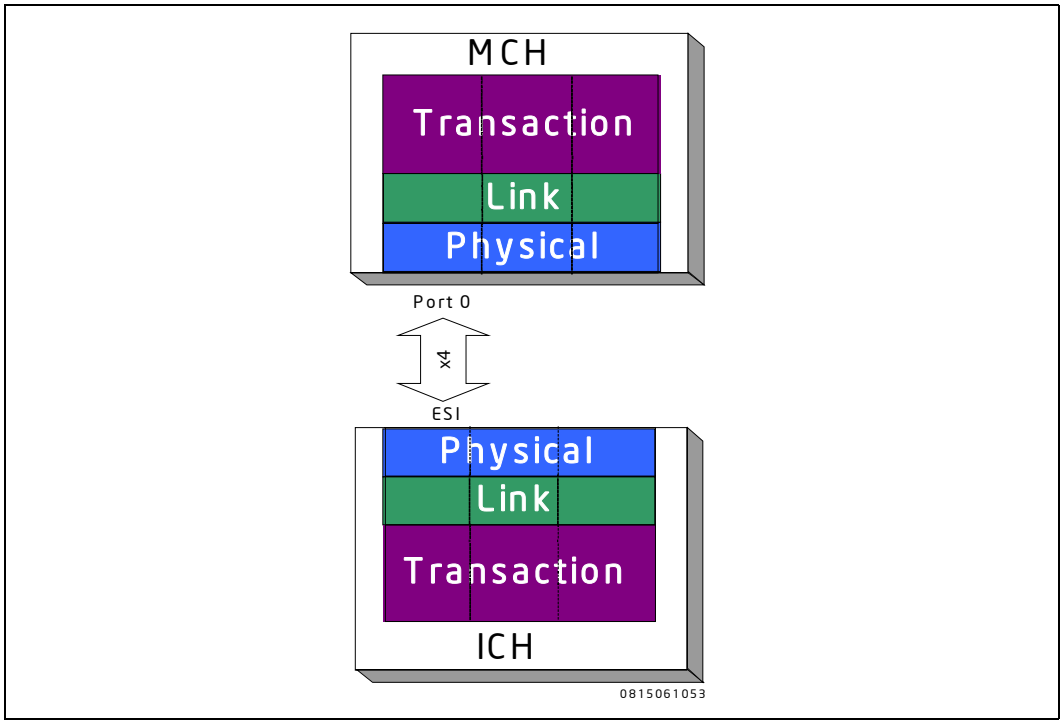
Software must configure the system so that each interrupt has a valid recipient. In the event that an interrupt doesn't have a valid recipient, since the Intel® 5100 MCH Chipset will not necessarily know that the interrupt is targeted for a non-existing processor, will deliver the interrupt to the processor buses following the interrupt routing rules described in this section. 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 Section 5.2.4, "Memory RAS".

5.12 Enterprise South Bridge Interface (ESI)

The Enterprise South Bridge Interface (ESI) in the Intel® 5100 MCH Chipset is the chip-to-chip connection to the ICH9R see Figure 24, "Intel® 5100 Memory Controller Hub Chipset to ICH9R Enterprise South Bridge Interface." The ESI is an extension of the standard *PCI Express* Base Specification*, Rev. 1.0a 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 transfer capabilities. Base functionality is completely transparent permitting current and legacy software to operate normally. For the purposes of this document, the ICH9R will be used as a reference point for the ESI discussion in the Intel® 5100 MCH Chipset.

Figure 24. Intel® 5100 Memory Controller Hub Chipset to ICH9R Enterprise South Bridge Interface



When operating with only the ESI port, the available bi-directional bandwidth to the south bridge is 2 GB/s (1 GB/s each direction).

5.12.1 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 ICH9R 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 (e.g., 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 the Intel® 5100 MCH Chipset.

5.12.2 Power Management Support

The ICH9R 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 ICH9R 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 ICH9R. 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.

5.12.2.1 Rst_Warn and Rst_Warn_Ack

The Rst_Warn message is generated by the ICH9R 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 tristated or toggled, if operations resume following this reset without power cycling. To protect against this, the ICH9R will send a reset warning to the MCH. The Intel® 5100 MCH Chipset is NOT required to quiesce the DRAM's prior to reset. The Intel® 5100 MCH Chipset does not put the DIMMs in self-refresh when PLTRST# is asserted.

The MCH completes the handshake by generating the Rst_Warn_Ack message to the ICH9R at the earliest.

5.12.2.2 STPCLK Propagation

The ICH9R has a sideband signal called STPCLK. This signal is used to place IA-32 CPUs into a low power mode. Traditionally, this signal has been routed directly from the I/O controller hub to the CPUs.

5.12.3 Special Interrupt Support

The ICH9R 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.

5.12.4 Inbound Interrupts

To the MCH, interrupts from the ICH9R are simply inbound non-coherent write commands routed to the processor buses. The MCH does not support the serial APIC bus.

5.12.5 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, etc.).

When the MCH accepts these PCI Express* interrupt messages, it aggregates and passes the corresponding "Assert_INTx" messages to the ICH9R'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.

5.12.6 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 SC.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.

5.12.7 Error Handling

Table 123, "Intel® 5100 Memory Controller Hub Chipset Error List" describes the errors detected on ESI through the standard PCI Express* and Advanced error reporting mechanism.

5.12.7.1 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.

5.12.7.2 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.

5.13 PCI Express* Ports

The Intel® 5100 MCH Chipset contains two classes of PCI Express* derived ports. These are:

- Enterprise South Bridge Interface (ESI), Port 0
- DMA Engine/General purpose ports, Port 2, Port 3, Port 4, Port 5, Port 6, and Port 7

Note: There is no PCI Express* port designated as Port 1.

The ESI port is the primary interface to the ICH9R. The Intel® 5100 MCH Chipset supports six general purpose x4 PCI Express* ports. These ports are combinable to form three high performance x8 ports. The MCH also supports the combination of Port 4 through Port 7 to form a high performance x16 graphics PCI Express* port.

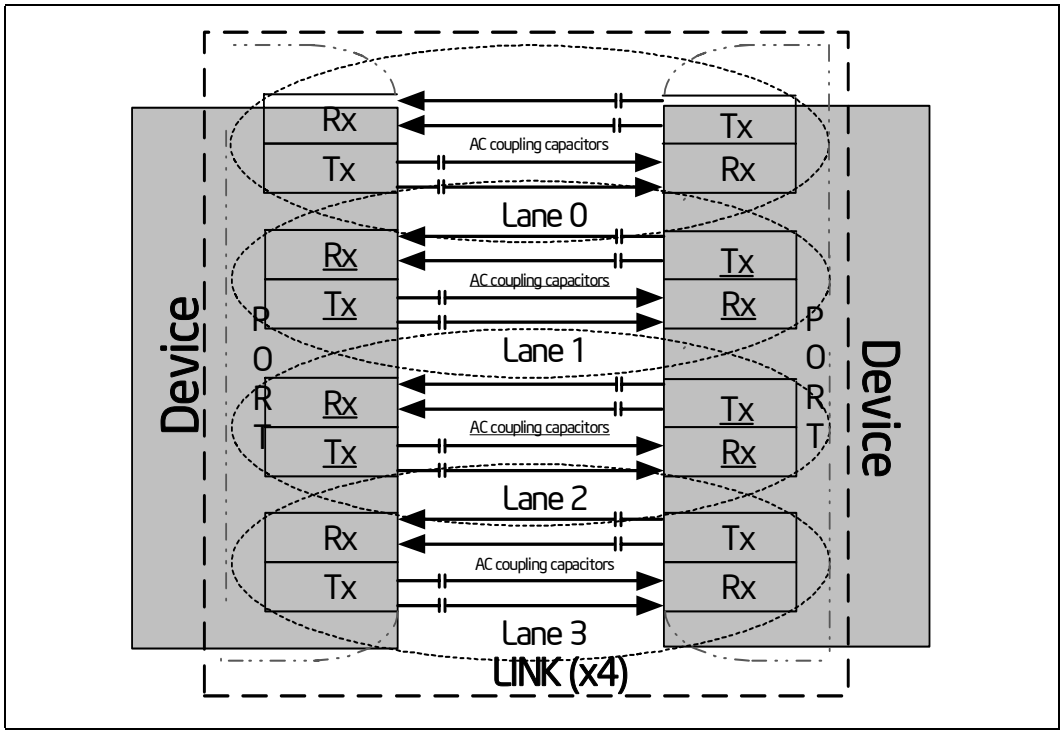
The following sections describe the characteristics of each of these port classes in detail.

5.13.1 Intel® 5100 Memory Controller Hub Chipset PCI Express* Port Overview

The Intel® 5100 MCH Chipset 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* Base 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 Figure 25, "x4 PCI Express* Bit Lane."

Figure 25. x4 PCI Express* Bit Lane



The raw bit-rate per PCI Express* bit lane is 2.5 Gb/s. This results in a real bandwidth per bit lane pair of 250 MB/s given the 8b/10b 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® 5100 MCH Chipset PCI Express* port are organized as four bi-directional bit lanes, and are referred to as a x4 port.

5.13.2 PCI Express* General Purpose Ports

PCI Express* Port 2 through Port 7 are configurable for general purpose I/O applications and interfaces. The following port pairs can be combined to provide x8 ports, Ports 2 and 3, Ports 4 and 5, Ports 6 and 7. When combining ports the controlling ports registers default to the lower port numbers address space. Thus when ports 4 and 5 are combined, the control registers are associated with port 4. These ports are depicted in Figure 26, "Intel® 5100 Memory Controller Hub Chipset PCI Express* General Purpose Ports."

The MCH can also combine these four general purpose x4 ports into a single optimized x16 high performance graphics interface. This interface is depicted in Figure 27, "PCI Express* High Performance x16 Port."

Figure 26. Intel® 5100 Memory Controller Hub Chipset PCI Express* General Purpose Ports

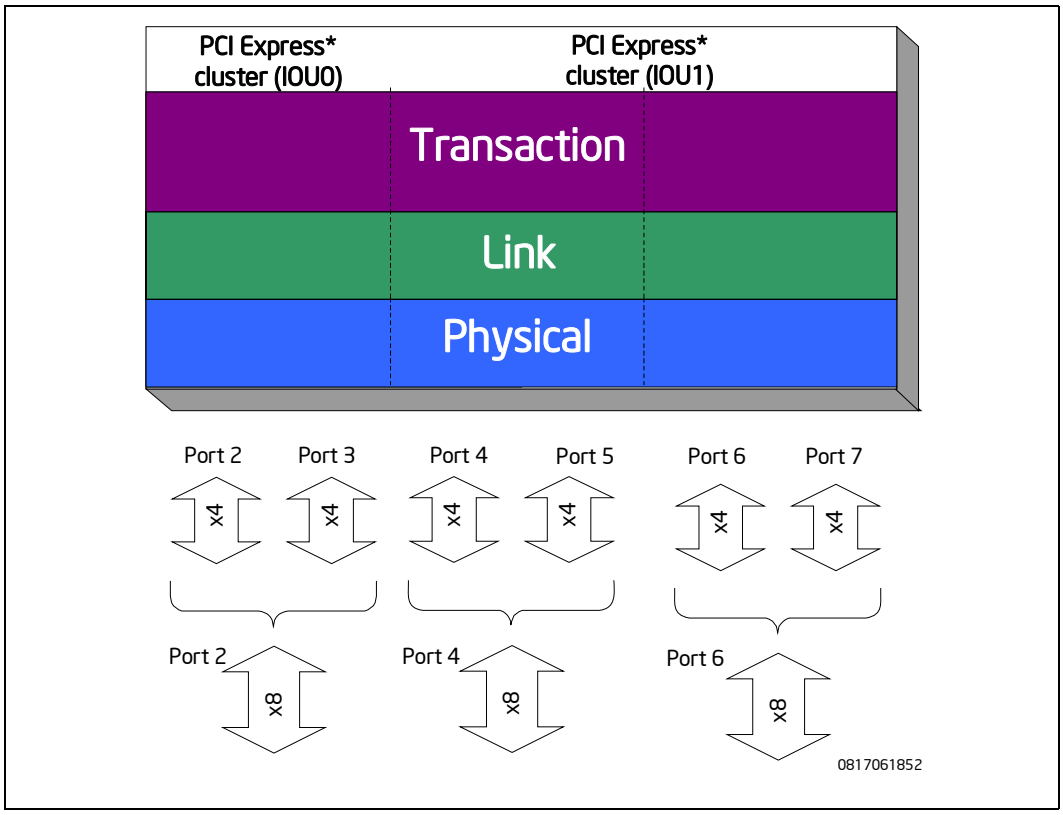
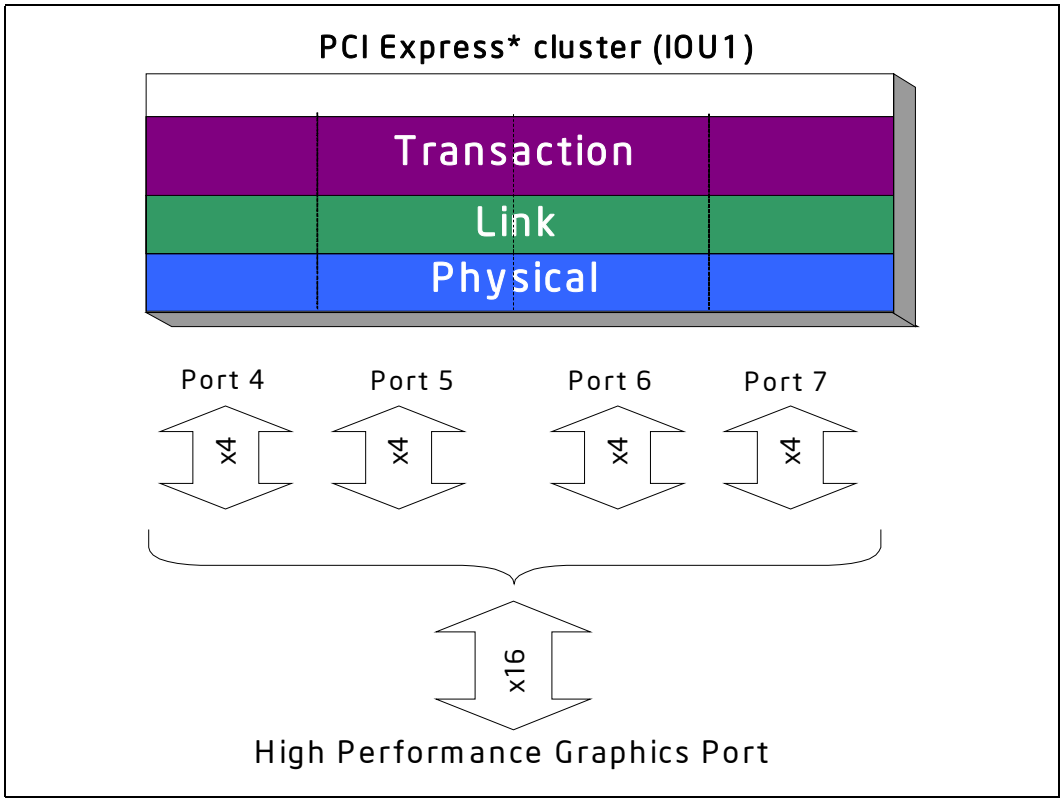


Figure 27. PCI Express* High Performance x16 Port



5.13.3 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 95, "PCI Express* Link Width Strapping Options for Port CPCI Configuration in Intel® 5100 Memory Controller Hub Chipset" illustrates the PEWIDTH strapping options for various link widths in the PCI Express* ports in the MCH.

Table 95. PCI Express* Link Width Strapping Options for Port CPCI Configuration in Intel® 5100 Memory Controller Hub Chipset

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
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.[1]					

Note:
1. Link negotiation configuration is not recommended due to an erratum; see Erratum 1, PCI Express* auto link negotiation occasionally fails to correctly detect link width in *Intel® 5100 Memory Controller Hub Chipset Specification Update*.

Note: The *PCI Express* Base Specification*, Rev. 1.0a requires that a port be capable of negotiating and operating at the native width and 1x. The Intel® 5100 MCH Chipset will support the following link widths for its PCI Express* ports x16, x8, x4 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 and finally a x1 link. Note that the x8 and x4 link widths will only use the LSB positions from lane 0 while a x1 can be connected to any of the four positions (lane 0, lane 1, lane 2, lane 3) providing a higher tolerance to single point lane failures. When settling on a narrower width and the straps are used to over-ride 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.

5.13.4 PCI Express* Port Support Summary

Table 96, "Options and Limitations" describes the options and limitations supported by the MCH PCI Express* ports.

Table 96. 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 ICH9R. (Total: 6+1=7 ports)
Max payload	256 bytes
PCI Hot Plug*	The MCH does support PCI Hot Plug*.
Virtual Channels	The MCH only supports VC0

Table 96. Options and Limitations (Sheet 2 of 2)

Parameter	Support
Traffic Streams	The MCH does not support two streams of two priority levels (High/low) on the PCI Express* ports 2 and 3. All of the ports support only one stream and one priority level (including the ESI) on VC0.
Isochrony	MCH does not support isochrony
ECRC	The MCH does not support ECRC
Ordering	The MCH only supports strict PCI ordering
No Snoop	The 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® 5100 MCH Chipset requirements for an integrated DP chassis and which do not require cable or repeater support. Only an internal trace connector latency of 20 in. of FR4 is assumed.
Poisoning	MCH will poison data that it cannot correct

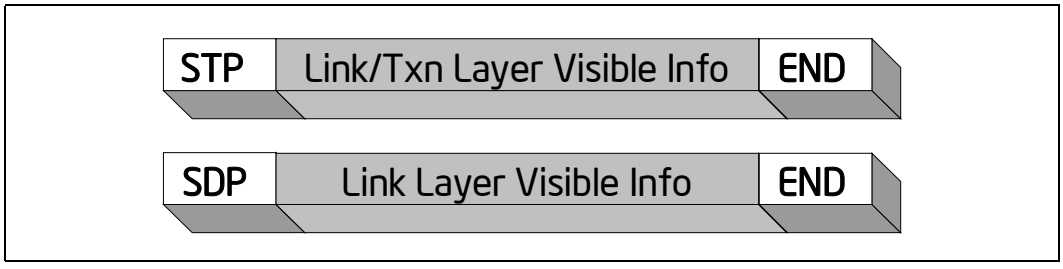
5.13.5 PCI Express* Port Physical Layer Characteristics

The PCI Express* physical layer implements high-speed differential serial signaling using the following techniques:

- Differential signaling (1.6 V 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 28, "PCI Express* Packet Visibility By Physical Layer" 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 28. PCI Express* Packet Visibility By Physical Layer



5.13.5.1 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.

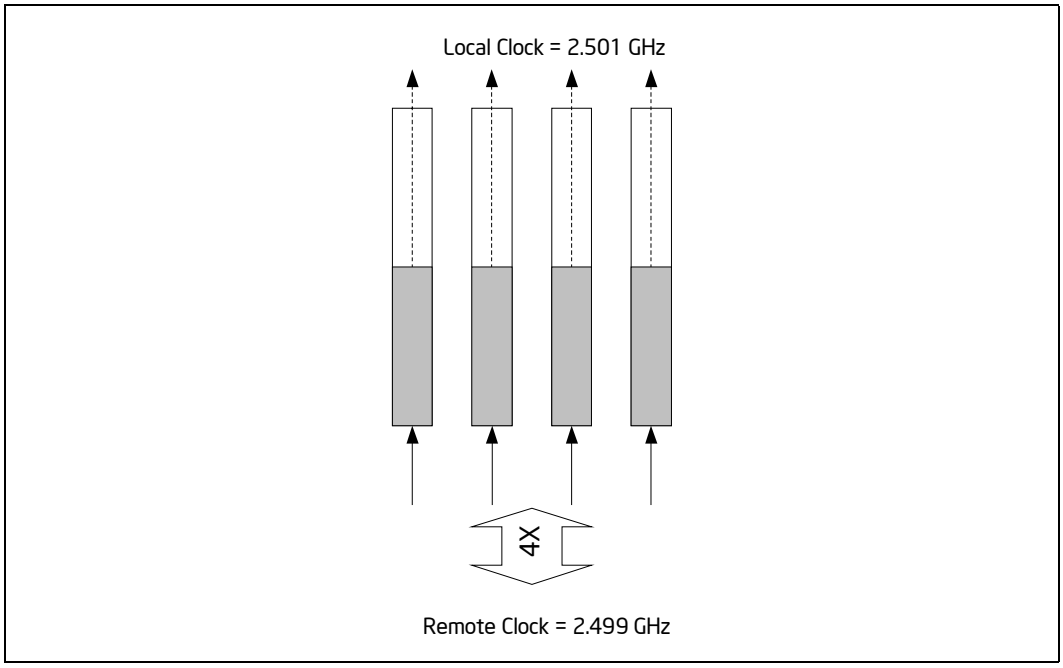
5.13.5.2 8b/10b Encoder/Decoder and Framing

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.

5.13.5.3 Elastic Buffers

Every PCI Express* port implements an independent elastic buffer for each PCI Express* lane. The elastic buffers are required since the Intel® 5100 MCH Chipset 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 29. PCI Express* Elastic Buffer (x4 Example)

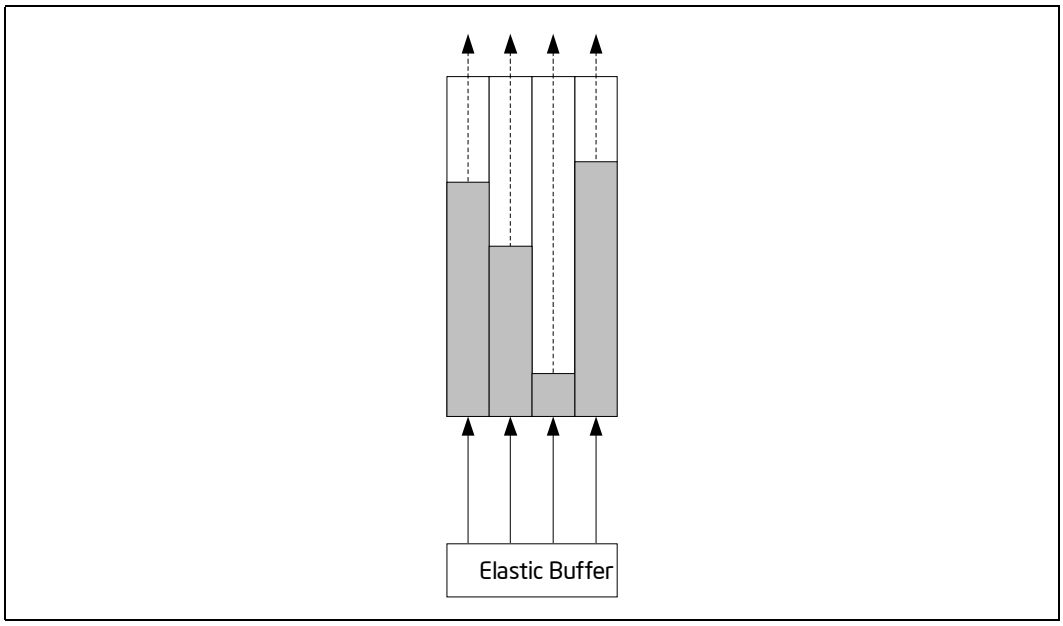


The elastic buffer is eight symbols deep. This accounts for three clocks of synchronization delay, the longest possible TLP allowed by the Intel® 5100 MCH Chipset (256 bytes), a 600 ppm difference between transmitter and receiver clocks, and worst case skip ordered sequence interval of 1538, framing overheads, and a few symbols of margin.

5.13.5.4 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 30. 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 20 ns of skew between lanes.

5.13.5.5 Lane Width Connections

The PCI Express* ports between the MCH and the PCI Express* connector can be connected straight through or reversed. The degradation lanes for specific port widths are specified in this section as well, i.e., when it is not possible to train to the highest configured lane width, the link reduces the lane width for either straight through or reversed connections. The specific port widths supported are the lane connections as depicted in Table 97, "Intel® 5100 Memory Controller Hub Chipset Lane Reversal Matrix". As an example of a straight through connection and the associated degradation, when Port 4 through Port 7 are configured as a x16 port, lane 0 through lane 15 of the MCH can be connected to lane 0 through lane 15 of the PCI Express* connector or PCI Express* link partner. If the connection does not support the x16 link, a x8 link may be established using lane 0 through lane 7 of the MCH and lane 0 through lane 7 of the link partner. If the connection does not support the x8 link, a x4 link may

be established using lane 0 through lane 3 of the MCH and lane 0 through lane 3 of the link partner. If the connection does not support the x4 link, a x1 link may be established using lane 0 from the MCH and lane 0 from the link partner.

As an example of lane reversal, when Port 4 through Port 7 are configured as a x16 port, lane 0 through lane 15 of the MCH can be connected to lane 15 through lane 0 of the PCI Express* connector or PCI Express* link partner. If the connection does not support the x16 link, a x8 link may be established using lane 0 through lane 7 of the MCH and lane 15 through lane 8 of the link partner. If the connection does not support the x8 link, a x4 link may be established using lane 0 through lane 3 of the MCH and lane 15 through lane 12 of the link partner. If the connection does not support the x4 link, a x1 link may be established using lane 0 from the MCH and lane 15 from the link partner.

The key to remembering which port combinations are allowed to be combined for any particular lane width, the control port is always required to be one of the ports used in order to negotiate the link state, the lanes must consist of consecutive x1, x4 or x8 ports in the allowed combinations. In the case of the above x16 lanes, port 4 is the control port. Any lesser lane width x8, x4 or x1 must utilize port 4 to establish a valid link.

Table 97. Intel® 5100 Memory Controller Hub Chipset Lane Reversal Matrix

PEWIDTH[3:0]	Port 2	Port 3	Port 4	Port 5	Port 6	Port 7
0000	x4 (0..3), 0, (3..0), 3	x4 (0..3), 0, (3..0), 3	x4 (0..3), 0, (3..0), 3	x4 (0..3), 0, (3..0), 3	x4 (0..3), 0, (3..0), 3	x4 (0..3), 0, (3..0), 3
0001	x4 (0..3), 0, (3..0), 3	x4 (0..3), 0, (3..0), 3	x4 (0..3), 0, (3..0), 3	x4 (0..3), 0, (3..0), 3	x8 (0..7, 0..3, 0, 7..0, 7:4, 7)	
0010	x4 (0..3), 0, (3..0), 3	x4 (0..3), 0, (3..0), 3	x8 (0..7, 0..3, 0, 7..0, 7:4, 7)		x4 (0..3), 0, (3..0), 3	x4 (0..3), 0, (3..0), 3
0011	x4 (0..3), 0, (3..0), 3	x4 (0..3), 0, (3..0), 3	x8 (0..7, 0..3, 0, 7..0, 7:4, 7)		x8 (0..7, 0..3, 0, 7..0, 7:4, 7)	
0100	x4 (0..3), 0, (3..0), 3	x4 (0..3), 0, (3..0), 3	x16 (0..15, 0..7, 0..3, 0, 15..0, 15..8, 15:12, 15)			
others						
1000	x8 (0..7, 0..3, 0, 7..0, 7:4, 7)		x4 (0..3), 0, (3..0), 3	x4 (0..3), 0, (3..0), 3	x4 (0..3), 0, (3..0), 3	x4 (0..3), 0, (3..0), 3
1001	x8 (0..7, 0..3, 0, 7..0, 7:4, 7)		x4 (0..3), 0, (3..0), 3	x4 (0..3), 0, (3..0), 3	x8 (0..7, 0..3, 0, 7..0, 7:4, 7)	
1010	x8 (0..7, 0..3, 0, 7..0, 7:4, 7)		x8 (0..7, 0..3, 0, 7..0, 7:4, 7)		x4 (0..3), 0, (3..0), 3	x4 (0..3), 0, (3..0), 3
1011	x8 (0..7, 0..3, 0, 7..0, 7:4, 7)		x8 (0..7, 0..3, 0, 7..0, 7:4, 7)		x8 (0..7, 0..3, 0, 7..0, 7:4, 7)	
1100	x8 (0..7, 0..3, 0, 7..0, 7:4, 7)		x16 (0..15, 0..7, 0..3, 0, 15..0, 15..8, 15:12, 15)			

Notes:
1. Port 0 (ESI) lane reversal is not supported, i.e., only 1 mode (normal x4).
2. The x2 lane reversal is not supported in any configuration.

5.13.5.6 Polarity Inversion

The *PCI Express* Base Specification*, Rev. 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® 5100 MCH Chipset supports polarity inversion.

5.13.6 Link Layer

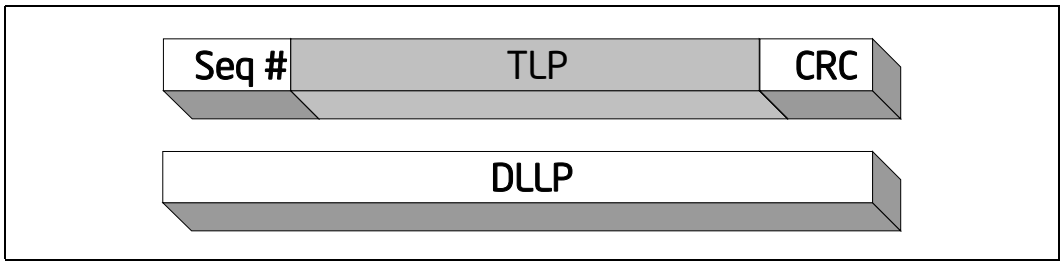
The Data Link Layer of the PCI Express* protocol is primarily responsible for data integrity. This is accomplished with the following elements:

- Sequence number assignment for each packet
- ACK/NAK protocol to ensure successful transmission of every packet
- CRC protection of packets
- Timeout mechanism to detect "lost" packets
- Credit exchange

Figure 31, "PCI Express* Packet Visibility By Link Layer" 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 31. PCI Express* Packet Visibility By Link Layer



5.13.6.1 Data Link Layer Packets (DLLP)

Refer to *PCI Express* Base Specification*, Rev. 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 16-bit 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.

5.13.6.2 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.

5.13.6.3 Link Level Retry

The *PCI Express* Base Specification*, Rev. 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 32-bit CRC (using the polynomial defined in the *PCI Express* Base Specification*, Rev. 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.

5.13.6.4 ACK Timeout

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 timeout 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*, Rev. 1.0a for details on this mechanism for the discussion on Retry Management and the recommended timer values.

5.13.7 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® 5100 MCH Chipset 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 98, "PCI Express* Credit Mapping for Inbound Transactions." The *PCI Express* Base Specification*, Rev. 1.0a defines which TLPs are covered by each flow control type.

Table 98. PCI Express* Credit Mapping for Inbound Transactions

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)
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.[1]	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]

1. 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. In other words, guarantee by construction

Table 99. 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 98, "PCI Express* Credit Mapping for Inbound Transactions" and Table 99, "PCI Express* Credit Mapping for Outbound Transactions." 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*, Rev. 1.0a.

5.13.7.1 Credit Update Mechanism, Flow Control Protocol (FCP)

After reset, credit information is initialized with the values indicated in Table 98, "PCI Express* Credit Mapping for Inbound Transactions" by following the flow control initialization protocol defined in the *PCI Express* Base Specification*, Rev. 1.0a. Since the MCH supports only VC0, only this channel is initialized.

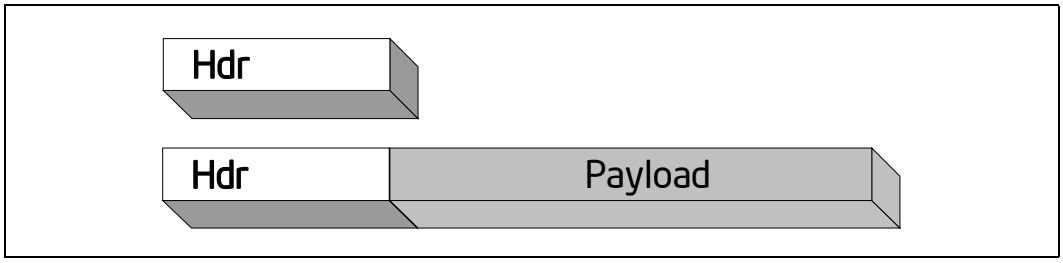
5.13.8 Transaction Layer

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 32, "PCI Express* Packet Visibility By Transaction Layer" illustrates the scope of the transaction layer on a PCI Express* packet. Some transaction layer packets have only a header (e.g., read request). Some transaction layer packets have a header followed by data (e.g., write requests and read completions).

Figure 32. PCI Express* Packet Visibility By Transaction Layer



5.13.9 DMA Engine Implementation

The MCH will support the DMA Engine technology on all PCI Express* Ports for I/O 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

5.13.10 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 Hardware Model is obsolete and no longer supported. Section 5.14, "Using DMA Engine Technology" though Section 5.17, "DMA Engine Driver" describe how the software 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, i.e., 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.

5.14 Using DMA Engine Technology

This section explains the mechanism by which software can request the use of and configure DMA Engine technology capabilities.

5.14.1 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 (e.g., 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 (e.g., DMA channels and Stream Priority) for exclusive use.
3. Release DMA Engine resources (e.g., DMA channels 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.

5.14.2 Basic Approaches

5.14.2.1 Software Model - Assistance from OS-level Software

The DMA Engine device is an I/O device (i.e., 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 (e.g., DMA channels). Optionally, the DMA Engine driver could provide an abstracted programming interface to use all DMA Engine capabilities (e.g., 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 (e.g., 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 (e.g., 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.

5.14.3 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.

5.15 Implementation Requirements

This section specifies dependencies on the various programming components for various tasks.

5.15.1 Software Model Dependencies

Table 100 lists dependencies for the Software Model.

Table 100. Software Model Dependencies (Sheet 1 of 3)

Task	Realm	Requirements
System Initialization	Chipset Hardware	• Initialize DMA Engine registers with specified default values
	BIOS	• Mark DMA Engine as an Integrated Device (PCI base class 8h, subclass/interface 80h) 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 Software Driver	• none (not loaded yet)
	I/O Device Hardware	• Does nothing - waits for I/O Software Driver to configure and enable the I/O device.
	I/O Device Software Driver	• none (not loaded yet)
	Other Restrictions, Issues, Notes	***Note:*** DMA Engine services exposed as a Root Complex Integrated Device.

Table 100. Software Model Dependencies (Sheet 2 of 3)

Task	Realm	Requirements
OS Initialization	Chipset Hardware	• none
	BIOS	• none
	DMA Engine Device Software 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 Hardware	• Does nothing - waits for I/O Software Driver to configure and enable the I/O device.
	I/O Device Software 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 Hardware	• N/A - the chipset still supports CB_Query[1] message and inbound MMIO reads & writes, but they are not used in the software model.
	BIOS	• none
	DMA Engine Device Software Driver	• Exposes DMA Engine via DMA Engine Driver Software 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 Hardware	• Waits for its software driver to program the I/O device and command it to use DMA Engine resources.
	I/O Device Software Driver	• Searches & detects DMA Engine Driver's Software I/F • Registers with DMA Engine driver's software I/F • Query for DMA Engine services via DMA Engine driver's software I/F • Passes Client's Requestor ID and CFG Information via DMA Engine driver's software 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 software 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 Hardware	• None
	BIOS	• None
	DMA Engine Device Software Driver	• Programs DMA Engine registers for Client Device reserving DMA Engine resource
	I/O Device Hardware	• Waits for its software driver to command it to use DMA Engine resources
	I/O Device Software Driver	• Via DMA Engine software driver, client reserves and configures specific DMA Engine resources
	Other Restrictions, Issues, Notes	• Restriction: Each DMA Engine resource can have exactly one owner, and sharing is not permitted unless completely managed by the DMA Engine driver. ***Note:*** After each Load, Start, or Reset - DMA Engine resources must be reserved before use.

Table 100. Software Model Dependencies (Sheet 3 of 3)

Task	Realm	Requirements
De-allocate DMA Engine resources	Chipset Hardware	• None
	BIOS	• None
	DMA Engine Device Software Driver	• Manages client request to free DMA Engine resources and programs DMA Engine device registers to de-allocate resources
	I/O Device Hardware	• None
	I/O Device Software Driver	• Client, via DMA Engine software 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 Hardware	• Support inbound MMIO reads and writes.
	BIOS	• None
	DMA Engine Device Software Driver	• Manages DMA Engine errors etc. • Signals Client of DMA Engine errors
	I/O Device Hardware	• None
	I/O Device Software Driver	• Makes necessary adjustments for any DMA Engine errors
	Other Restrictions, Issues, Notes	• None
Managing OS driven plug and play of client hardware device	Chipset Hardware	
	BIOS	
	DMA Engine Device Software Driver	• Manages client request to free DMA Engine resources and programs DMA Engine device registers to free indicated resources
	I/O Device Hardware	
	I/O Device Software Driver	• Client must release DMA Engine resources during PnP events such as Stop, Surprise removal, etc.
	Other Restrictions, Issues, Notes	***Note:*** Whether a STOP is treated the same as an UNLOAD (i.e., free all resources on stop and reclaim them on start) is up to the Client Hardware/Software 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 Hardware and manage system sleep state transitions	Chipset Hardware	
	BIOS	
	DMA Engine Device Software Driver	• Must be able to take device power state transition commands from OS and notify client software to de-allocate DMA Engine resources
	I/O Device Hardware	• Must be able to drain DMA Engine traffic & release DMA Engine resources when software issues the command to move to D3 state • Must indicate to software that it is now OK to report transition to D3 state
	I/O Device Software Driver	• Must be able to take device power state transition commands from OS and request Client Hardware to move to new (e.g., D3) state • Must be able to drain DMA Engine traffic in hardware if any & release DMA Engine resources when OS software issues command to move to D3 state • Must indicate to OS software 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 Software Drivers will be notified to De-allocate DMA Engine resources

1. 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.

5.16 Programming Flow

This section describes the steps for a client to use DMA Engine facilities.

Note: Locking of CB MMIO space is undesirable, unpredictable, and might result in deadlock. Thus, an attempt to generate a locked request to CB MMIO location is invalid.

5.16.1 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.

5.16.2 Using DMA

In the following discussion, the term Device means I/O device and its I/O device driver.

The DMA Engine driver reads Section 3.11.22.1, "CHANCNT - Channel Count" 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 Section 3.11.22.3, "INTRCTRL - Interrupt Control" register. The I/O driver registers its interrupt handler with the DMA Engine device (see Section 5.16.3, "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 Section 3.11.23.6, "CHANCMP[3:0]: Channel Completion Address Register" to specify where the DMA channel writes the Completion Status.

The DMA channel is now operational.

Client reads the Section 3.11.22.2, "XFERCAP - Transfer Capacity" register to learn the maximum transfer capacity for the DMA channel.

To start DMA operation, the client builds DMA descriptors in main memory, writes Section 3.11.23.4, "CHAINADDR[3:0] - Descriptor Chain Address Register" to indicate the first descriptor in the chain, and then initiates the DMA transfer in Section 3.11.23.5, "CHANCMD[3:0] - DMA Channel Command Register"

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.

5.16.3 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.

5.16.3.1 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 multiprocessor 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 101 illustrates the ISR's reaction to reading INTRCTRL.

Table 101. 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)

5.16.3.2 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 (i.e., 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.

5.16.3.3 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 5.16.3, "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.

5.17 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 Section 3.11.22.3, "INTRCTRL - Interrupt Control" which points to the first set of per port registers. These registers form a linked list with the Section 3.12.0.1, "NXTPPRSET2 - Next Per Port Register Set" pointing to the next register set and Section 3.12.0.2, "NXTPPRSET3 - Next Per Port Register Set" pointing to the next register set, the last register set, which has a value of 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 5.16.3.1, "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 Section 3.12.0.15, "BR_MEM_BASE[3:2] - Bridge Memory Base Register" on page 256, Section 3.12.0.16, "BR_MEM_LIMIT[3:2] - Bridge Memory Limit Register" on page 256, Section 3.12.0.17, "BR_PMEM_BASE[3:2] - Bridge Prefetchable Memory Base Register" on page 256, 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).

5.17.1 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® 5100 MCH Chipset 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 requests from the ESI port and Ports 2/3 and classify 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 DMA Engine with the Coherency Engine (CE) to share the allocated 8.53/10.6 GB/s bandwidth in each direction.

5.17.1.1 Level 3 - IOU0, IOU1, DMA Arbitration

The Level 3 arbitration shares available bandwidth in the CE by a programmable, weighted, round robin scheme that scans for requests from the two IOUs (IOU0 and IOU1) and the DMA Engine optimizing bandwidth and avoiding starvation. The CE 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.

5.17.2 Supported PCI Express* Transactions

Table 102, "Incoming PCI Express* Requests" lists all the transactions supported by the Intel® 5100 MCH Chipset which are expected to be received from the PCI Express* interface. Similarly, Table 104, "Outgoing PCI Express* Requests" lists all the transactions to be expected by an attached PCI Express* component. Refer to the *PCI Express* Base Specification*, Rev. 1.0a for the specific protocol requirements of this interface.

Table 102. Incoming PCI Express* Requests

PCI Express* Transaction	Address Space or Message	Intel® 5100 Memory Controller Hub Chipset 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® 5100 MCH Chipset 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.
	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 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 103. Incoming PCI Express* Completions

PCI Express* Transaction	Address Space or Message	Intel® 5100 Memory Controller Hub Chipset Response
Completions for Outbound Writes	I/O or Configuration[1]	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.

1. Outbound Memory writes are posted and have no completions

Table 104. 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® 5100 MCH Chipset 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® 5100 MCH Chipset in response to the DMA Engine query and contains pertinent information on DMA Engine version, BAR offset etc.

Table 105. 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.

5.17.2.1 Unsupported Messages

If the Intel® 5100 MCH Chipset decodes any vendor message (which is not defined in Table 102, "Incoming PCI Express* Requests"), the MCH will take the following actions as specified in the Vendor defined message section of the *PCI Express* Base Specification*, Rev. 1.0a.

- Vendor Type 0 - Unsupported Request
- Vendor Type 1 - Drop request.

5.17.2.2 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)

5.17.3 Transaction Descriptor

The *PCI Express* Base Specification*, Rev. 1.0a defines a field in the header called the Transaction Descriptor. This descriptor comprises three sub-fields:

- Transaction ID
- Attributes
- Virtual Channel ID

5.17.3.1 Transaction ID

The Transaction ID uniquely identifies every transaction in the system. The Transaction ID comprises four sub-fields described in Table 106.

Table 106. 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.		

5.17.3.2 Attributes

PCI Express* supports two attribute hints described in Table 107.

Table 107. 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.	

5.17.3.3 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.

5.17.4 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.

5.17.4.1 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 106. 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 5.17.5.

5.17.4.1.1 Inbound Memory Reads

Read Completion Policy

For inbound read requests, the MCH is allowed to split completions along a Read Completion Boundary (RCB) of 64 bytes, 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 256 bytes, 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.

5.17.4.2 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 (e.g., 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 (e.g., 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.

5.17.4.3 Zero-Length Reads/Writes

The *PCI Express* Base Specification*, Rev. 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*, Rev. 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.

5.17.4.4 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 continuous inbound traffic is not stalled waiting for the above write sequence even under heavily loaded 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.

5.17.4.5 PHOLD Support

The MCH supports the PHOLD protocol. This protocol is used for legacy ISA devices which did not allow the possibility for being both a master and a slave device simultaneously. Example devices that use the PHOLD protocol are legacy floppy drives and parallel port.

5.17.4.6 Interrupt Handling

A PCI Express* device represents interrupts with either MSI or inbound interrupt messages (Assert_INTx/Deassert_INTx).

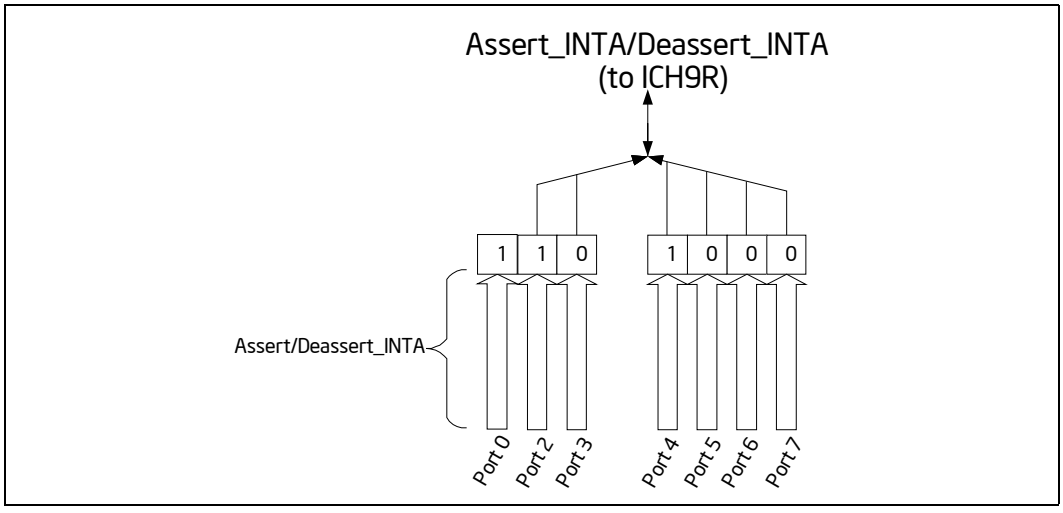
Each PCI Express* port of the MCH is responsible for tracking assert/deassert messages for each of the four interrupts (INTA, INTB, INTC, INTD) and representing them with four output virtual (Assert_INTA, Assert_INTB, Assert_INTC, and Assert_INTD) messages to the ICH9R.

Figure 33, "Legacy Interrupt Routing (INTA Example)" illustrates how the PCI Express* interrupts are routed to the ICH9R. 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 33. Legacy Interrupt Routing (INTA Example)



5.17.4.7 Error Messages

PCI Express* reports many error conditions through explicit error messages: ERR_COR, ERR_UNC, ERR_FATAL.

5.17.4.8 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 ICH9R component issues these messages as a response to its integrated PCI Hot Plug* controllers.

5.17.4.9 Outbound Transactions

The MCH will generate the outbound transactions listed in Table 104. 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 5.17.5.2, "Outbound Transaction Ordering Rules".

5.17.4.10 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.

5.17.4.10.1 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.

5.17.4.10.2 Outbound Posted Transactions

Once a posted request (memory mapped I/O write) is issued from the PCI Express* 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.

5.17.4.11 Outbound Vendor-Specific Messages

The MCH supports only vendor-specific EOI messages outbound.

5.17.4.11.1 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 (ICH9R).

5.17.4.12 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*, Rev. 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® 5100 MCH Chipset 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.

5.17.4.13 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.

5.17.4.14 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.

5.17.5 Ordering Rules

This section describes the MCH ordering rules for transactions progressing through the PCI Express* unit.

5.17.5.1 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.

5.17.5.1.1 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.

RULE 2: Inbound posted write requests must be allowed to progress past stalled inbound non-posted requests.

RULE 3: 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*, Rev. 2.3 for details on the Producer - Consumer ordering model.

RULE 4: Outbound non-posted read or write completions must push ahead *all* prior inbound posted write transactions from that PCI Express* port.

RULE 5: To optimize performance, Inbound, coherent, posted writes will issue ownership requests (RFO) without waiting for prior ownership requests to complete.

RULE 6: Inbound messages follow the same ordering rules as inbound posted writes.

Inbound messages are listed in Table 102, "Incoming PCI Express* Requests". 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.

5.17.5.2 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*, Rev. 1.0a.

5.17.5.2.1 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.

RULE 2: Outbound posted write requests must be allowed to progress past stalled outbound non-posted requests.

RULE 3: 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*, Rev. 2.3 for details on the Producer - Consumer ordering model.

RULE 4: Posted outbound messages must follow the same ordering rules as outbound posted writes.

RULE 5: If a non-posted inbound request requires multiple sub-completions, then those sub-completions must be delivered in linearly increasing address order.

5.17.5.3 MCH Ordering Implementation

Table 108 summarizes the rules enforced on transactions from a given PCI Express* port by the MCH.

Table 108. 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

5.17.5.3.1 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 (e.g., 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 5.17.4.13, "Peer-to-peer Transactions" and Table 5.17.4.14, "Peer-to-peer Configuration Cycles".

5.17.5.4 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.

5.17.5.4.1 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.

5.17.6 Prefetching Policies

The MCH does not perform any speculative prefetching for PCI Express* interface component reads. The PCI Express* component south of the Intel® 5100 MCH Chipset 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.

5.17.7 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, 0600h) 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.

5.17.8 No Isochronous Support

The Intel® 5100 MCH Chipset does not support isochrony. Only the default virtual channel (channel 0) is supported on the PCI Express* interfaces.

5.17.9 PCI Hot Plug*

Native PCI 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. PCI Hot Plug* is not supported on the Intel® 5100 MCH Chipset.

Each Intel® 5100 MCH Chipset PCI Express* port supports the optional PCI Hot Plug* capability described in *PCI Express* Base Specification*, Rev. 1.0a. The PCI Hot Plug* model implies a PCI Hot Plug* controller per port which is identified to software as a capability of the peer-to-peer bridge configuration space.

PCI Hot Plug* support requires that the MCH supports a set of PCI Hot Plug* messages to manage the states between the PCI 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 PCI 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 PCI Hot Plug* messages.

5.18 Power Management

The Intel® 5100 MCH Chipset power management support includes:

- ACPI supported
- System States: S0, S1 (desktop), S3, S4, S5, C0, C1, C2 (desktop)

5.18.1 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 ICH9R).
  - G2/S5: Soft off. Requires total system reboot.
  - G3: Mechanical Off. All power lost (except real time clock).

5.19 System Reset

The Intel® 5100 MCH Chipset is a integral part of the I/O subsystem tree, however, the ICH9R is responsible for general propagation of system reset throughout the platform through the PLTRST# signal. The ICH9R facilitates any specialized synchronization of reset mechanisms required by the various system components.

5.19.1 Intel® 5100 Memory Controller Hub Chipset Reset Types

The Intel® 5100 MCH Chipset differentiates among five types of reset as defined in Table 109, "Intel® 5100 Memory Controller Hub Chipset Reset Classes."

Table 109. Intel® 5100 Memory Controller Hub Chipset Reset Classes

Type	Mechanism	Effect/Description
Power-Good	PWRGOOD 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). Tristates all MCH outputs, or drives them to "safe" levels.
Hard	**RESETI**# 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. Tristates all MCH outputs, or drives them to "safe" levels.
Processor-only	Configuration Write	Propagated to all processors via the FSB{0/1}RESET# pins on the FSB. The MCH does not undergo an internal reset.
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.

5.19.1.1 Power-Good Mechanism

The initial boot of an Intel® 5100 MCH Chipset is facilitated by the Power-Good (PwrGd) 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 or system PwrGd signal a fixed interval (> 2 ms) 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® 5100 MCH Chipset and the ICH9R receive the system PwrGd signal via dedicated pins, PWRGOOD and PWROK respectively, as asynchronous inputs, 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 PWRGOOD is deasserted. The ICH9R is expected to assert its PLTRST# output and maintain its assertion for 1 ms after power is good, PWROK is asserted. The PLTRST# output from ICH9R is expected to drive the RESETI# input pin on the Intel® 5100 MCH Chipset, which will in turn hold the processor complex in reset via assertion of the FSB{0/1}RESET# 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 power-on reset has deasserted.

A system 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 tristate their outputs or to drive them to "safe" levels during a power-on reset.

The only system information that will remain after a system PwrGd reset is either contained in battery-backed or non-volatile storage.

5.19.1.2 Hard Reset Mechanism

Once the Intel® 5100 MCH Chipset 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 ICH9R component via the PLTRST# output pin, which is commonly connected directly to the Intel® 5100 MCH Chipset RESETI# input pin. The ICH9R may be caused to assert PLTRST# via both software and hardware mechanisms. The Intel® 5100 MCH Chipset will recognize a hard reset any time RESETI# is asserted while PWRGOOD remains asserted.

The Intel® 5100 MCH Chipset will propagate a hard reset to the FSB and to all subordinate PCI Express* subsystems. The FSB components are reset via the FSB{0/1}RESET# 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® 5100 MCH Chipset-based platform in general will require a full configuration and initialization sequence to be brought back on-line.

5.19.1.3 Processor-Only Reset Mechanism

For power management and other reasons, the Intel® 5100 MCH Chipset 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.

5.19.1.4 Targeted Reset Mechanism

The targeted reset is provided for PCI 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* Base Specification*, Rev. 1.0a for a description of the PCI 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.

5.19.1.5 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.

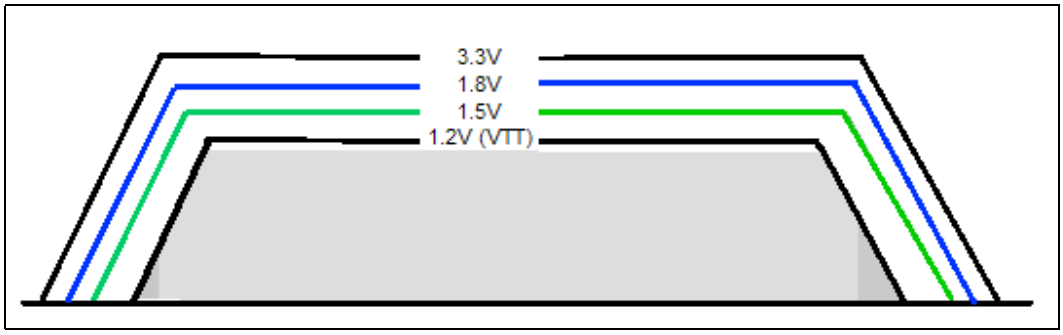
When BINIT# is asserted on the FSB, all bus agents (CPUs and MCH) are required to reset their internal FSB arbiters and all FSB tracking state machines and logic to their default states. This will effectively "un-hang" the bus to provide a path into chipset configuration space. Note that the MCH device 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.

5.19.2 Intel® 5100 Memory Controller Hub Chipset Power Sequencing

General power sequencing requirements for the Intel® 5100 MCH Chipset are simple. In general higher voltages must come up before lower voltages. Figure 34, "Intel® 5100 Memory Controller Hub Chipset Power Sequencing" depicts the sequencing of the four main voltages powering the Intel® 5100 MCH Chipset.

Figure 34. Intel® 5100 Memory Controller Hub Chipset Power Sequencing



Note: Power-up -> 3.3 V must ramp ahead and stay above 1.8 V, which must ramp ahead and stay above 1.5 V which must ramp and stay above 1.2 V (VTT). 3.3 V must always be at least 0.7 V greater than 1.5 V. Duration of the power ramp of each individual supply must be between 0.1 ms and 100 ms. The power down sequence is not as critical. When transiting to power down states, 1.8 V may stay up as required while the other supplies decay to 0 V as needed. See the *Quad-Core and Dual-Core Intel® Xeon® Processor 5000 Sequence with Intel® 5100 Memory Controller Hub Chipset for Communications, Embedded, and Storage Applications – Platform Design Guide* or *Intel® Core™2 Duo Processors T9400 and SL9400 and Intel® 5100 Memory Controller Hub Chipset for Communications and Embedded Applications – Platform Design Guide* for the most current information.

5.19.3 Reset Sequencing

Figure 35, "Power-On Reset Sequence" 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 35, "Power-On Reset Sequence," 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 ICH9R. 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 ICH9R to coordinate across the deasserting edge of the FSB{0/1}RESET# output from the MCH.

Table 110, "Reset Sequences and Durations" 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 35. Power-On Reset Sequence

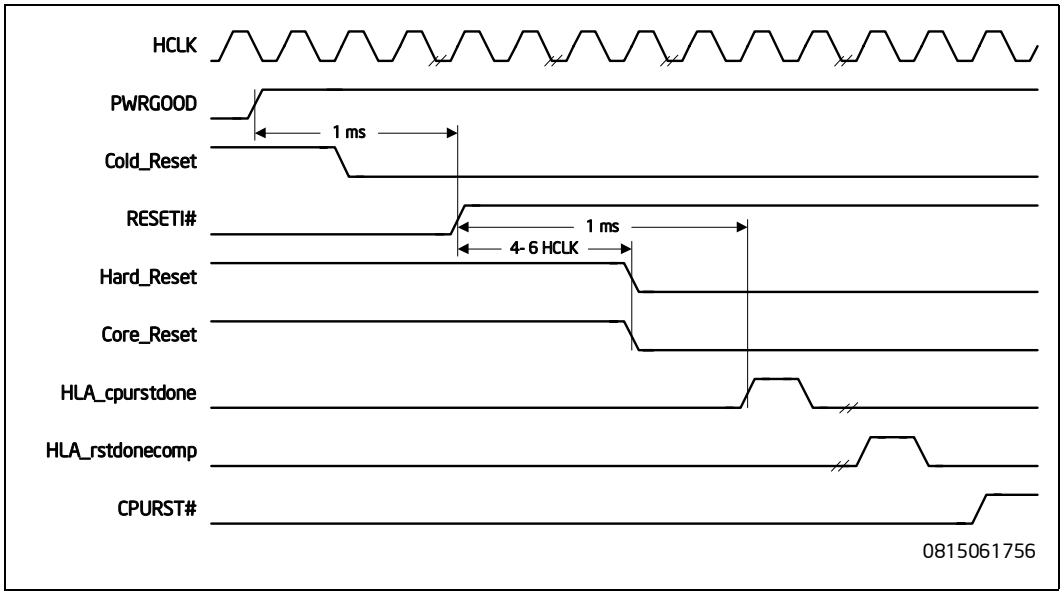


Table 110. Reset Sequences and Durations (Sheet 1 of 2)

From	To	Duration	Source	Comment
Power on	System PwrGd	>2 mS	Platform	Control logic on the platform must ensure that there are at least 2 mS of stable power before System PwrGd (MCH PWRGOOD) is asserted.
PWRGOOD	RESETI# deassertion	1 mS	ICH9R	ICH9R enforces delay between detecting PWROK asserted and releasing PLTRST# (note that ICH9R PLTRST# is directly connected to MCH RESETI#).

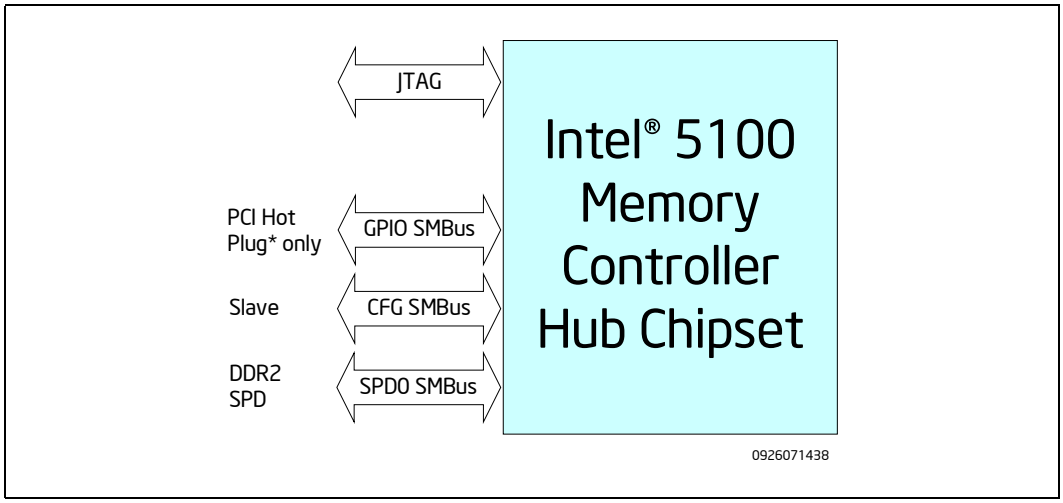
Table 110. Reset Sequences and Durations (Sheet 2 of 2)

From	To	Duration	Source	Comment
RESETI# deassertion	Hard/Core deassertion	4-6 HCLK	MCH	MCH waits for a common rising edge on all internal clocks, then releases core reset(s).
RESETI# deassertion	FSB{0/1} RESET# deassertion	1 mS	MCH	MCH enforces delay between RESETI# and FSB{0/1}RESET# deassertion. ESI handshake is incremental to the timer.

5.20 SMBus Interfaces Description

The Intel® 5100 MCH Chipset provides three fully functional *System Management Bus (SMBus) Specification*, Version 2.0 compliant target interfaces. These interfaces are used to support platform level operations such as Serial Presence Detect for DDR2 DIMMs, PCI Hot Plug*, and configuration of platform devices. Each of these interfaces have dedicated uses as shown in Figure 36, "Intel® 5100 Memory Controller Hub Chipset SMBus Interfaces," and as described in Section 2.4, "SMBus Interfaces."

Figure 36. Intel® 5100 Memory Controller Hub Chipset SMBus Interfaces



SMBus SPD0 is dedicated to memory serial presence detect and DDR2 DIMM configuration. SMBus SPD0 is dedicated to channel 0 and channel 1 DDR2 DIMMs. The SMBus GPIO is dedicated to PCI Hot Plug* support and is not currently a supported feature.

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.

5.20.1 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*, Version 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 kB. 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, see the below SMBus descriptions. 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.

5.20.2 SMBus and PCI Express* Interoperability Timeout Recommendation

The *System Management Bus (SMBus) Specification*, Version 2.0, Section 3.1.1, SMBus Common AC Specifications, states the timeout value to be 25 ms and the *PCI Express* Base Specification*, Rev. 1.0a, Section 2.8, Completion Timeout Mechanism, specifies PCI Express* global timeout of 50 ms. Due to Erratum 22, SMBus TLOW:SEXT specification may be exceeded on SMBus 0 when the north bridge is clocked with a 266 MHz BUSCLK, once a timeout occurs, SMBus hangs until a hard reset. Any SMBus transaction through the Intel® 5100 MCH Chipset targeting any device outside of the Intel® 5100 MCH Chipset taking longer than 25 ms to complete will cause the SMBus to hang. Please refer to *Intel® 5100 Memory Controller Hub Chipset Specification Update* for details on Erratum 22.

Intel recommends that the PCI Express* Global Control Register (PEXGCTRL) Completion Timeout be programmed to less than the SMBus timeout. Currently, Intel sets the timeout to approximately 20 ms (Completion timeout register value = 744h) in the BIOS. For a more detailed description of the PEXGCTRL register, please refer to Section 3.8.8.34, "PEXGCTRL - PCI Express* Global Control Register." For an Intel® 5100 MCH Chipset-based system to encounter this error (SMBus hang), two conditions must be met. One, the Intel® 5100 MCH Chipset-based system would need to have a PEXGCTRL Completion Timeout value greater than 25 ms. Two, an SMBus master on the Intel® 5100 MCH Chipset-based system would have to read data from a PCI or PCI Express* device through the MCH; then, if this operation takes more than 25 ms, the SMBus will hang. It is important to note that an SMBus master reading from the Intel® 5100 MCH Chipset (Bus 0) will not encounter the SMBus hang. Systems that do not access any SMBus devices outside of the Intel® 5100 MCH Chipset through the Intel® 5100 MCH Chipset can safely set the timeout value to 50 ms to conform to the *PCI Express* Base Specification*, Rev. 1.0a. There is no planned fix for this.

For further clarification, here is an example scenario of a case in which a change to the recommended timeout is required. In this scenario, a video card is unable to respond within the recommended timeout. Furthermore, the default Microsoft Windows* driver maps the video frame buffer in MMIO space as Uncacheable Speculative Write Combining (USWC) memory. The CPU may speculatively execute a load (read) to the frame buffer region. The video controller may be unable to respond to the speculative read and could issue a retry. The system will continue to reissue the load to the video controller which may respond back with a retry. After 20 ms, the Intel® 5100 MCH Chipset timer expires and generates an NMI followed by a Microsoft Windows* blue screen. To work around this problem, set the PEXGCTRL Completion Timeout to a maximum value of 3FFFh. The PEXGCTRL timeout should be set to '744h' for any SMBus transactions through the Intel® 5100 MCH Chipset targeting a PCI Express* device outside of the Intel® 5100 MCH Chipset. Once the SMBus transactions are completed, the PEXGCTRL timeout should be set back to the maximum value of 3FFFh. Ensure that the system is silent on the PCI Express* side before changing the timer value (3FFFh to 744h and vice versa).

5.20.3 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 111. 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 111, "SMBus Transaction Field Summary" 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.

5.20.3.1 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.

Note: Packet Error Code (PEC) is not a supported feature as indicated in bit [4] below.

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

5.20.3.2 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.

5.20.3.3 Address Byte 3 Field

This field should be programmed with the bus number of the desired configuration register in the lower five 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; 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.	

5.20.3.4 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.	

5.20.3.5 Address Byte 1 Field

This field indicates the upper address bits for the 4 kB 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 4 kB region of register offset.

5.20.3.6 Address Byte 0 Field

This field indicates the lower eight address bits for the register with the 4 kB region, regardless whether it is a configuration or memory-map type of access.

Position	Description
7:0	Register Offset.

5.20.3.7 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.

5.20.3.8 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 Timeout. 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.

5.20.3.9 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.

5.20.4 SMBus Transaction Pictographs

The Intel® 5100 MCH Chipset 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 37. DWORD Configuration Read Protocol (SMBus Block Write/Block Read, PEC Disabled)

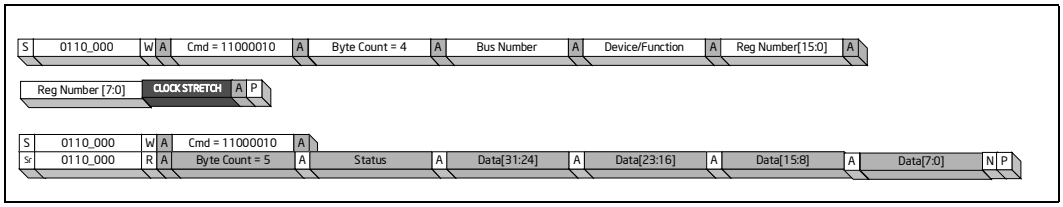


Figure 38. DWORD Configuration Write Protocol (SMBus Block Write, PEC Disabled)

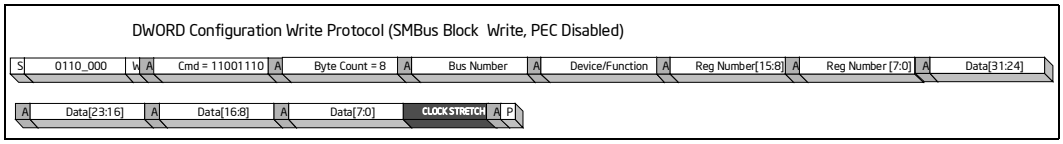


Figure 39. DWORD Memory Read Protocol (SMBus Block Write/Block Read, PEC Disabled)

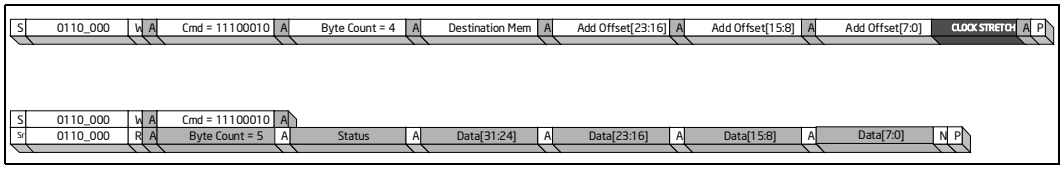


Figure 40. DWORD Memory Write Protocol

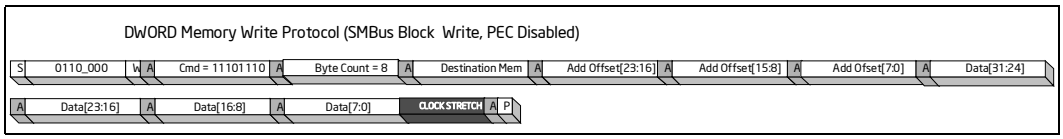


Figure 41. DWORD Configuration Read Protocol (SMBus Word Write/Word Read, PEC Disabled)

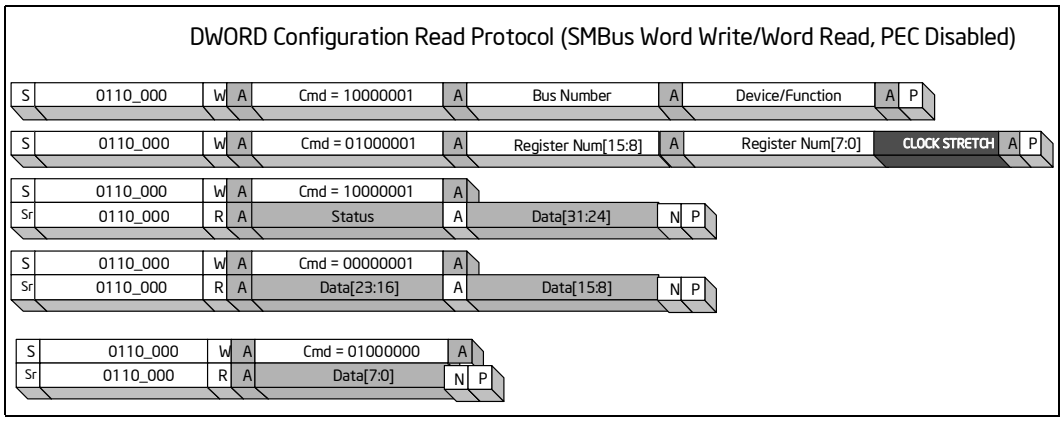


Figure 42. DWORD Configuration Write Protocol (SMBus Word Write, PEC Disabled)

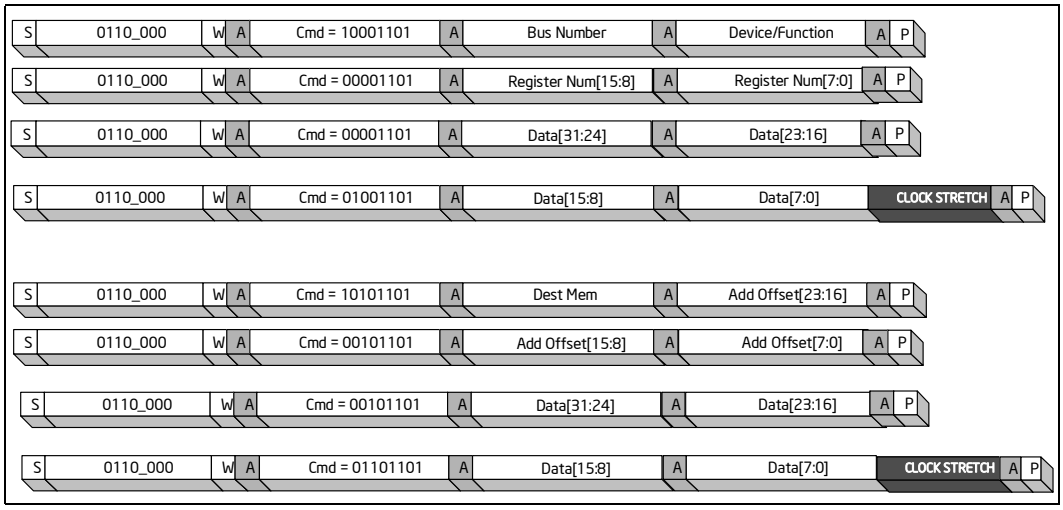


Figure 43. DWORD Memory Read Protocol (SMBus Word Write/Word Read, PEC Disabled)

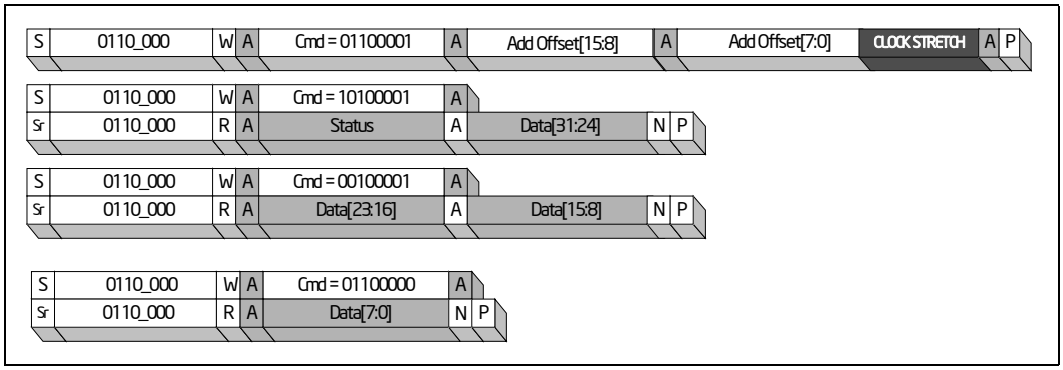
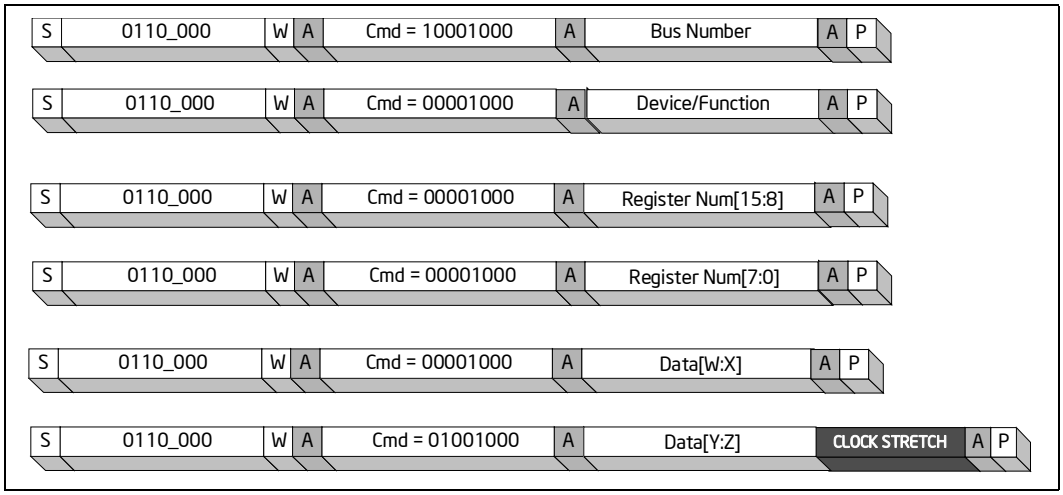


Figure 44. WORD Configuration Write Protocol (SMBus Byte Write, PEC Disabled)



5.20.5 Slave SMBus, SMBus 0

System Management software in a Intel® 5100 MCH Chipset can initiate system management accesses to the configuration registers via the Slave SMBus, SMBus 0.

The mechanism for the Server Management (SM) software to access configuration registers is through a *System Management Bus (SMBus) Specification*, Version 2.0 compliant slave port. Some Intel® 5100 MCH Chipset components contain this slave port and allow accesses to their configuration registers. The product specific details are compatible with the ICH9R SMBus configuration access mechanism. Most of the Intel® 5100 MCH Chipset 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® 5100 MCH Chipset component's internal configuration registers. Refer to the *System Management Bus (SMBus) Specification*, Version 2.0 for the bus protocol, timings, and waveforms.

Each component on the Intel® 5100 MCH Chipset must have a unique address. Intel® 5100 MCH Chipset component addresses are defined in Table 112, "SMBus Address for Intel® 5100 Memory Controller Hub Chipset-based Platform."

Table 112. SMBus Address for Intel® 5100 Memory Controller Hub Chipset-based Platform

Component	SMBus Address (7:1)
Intel® 5100 MCH Chipset	1100_000

5.20.5.1 Supported SMBus Commands

Intel® 5100 MCH Chipset components *System Management Bus (SMBus) Specification*, Version 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, etc. 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 performs 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 113, "SMBus Command Encoding" below.

Table 113. 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 *PEC_en* 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.

Note: Packet Error Code (PEC) is not a supported feature; see Section 5.20.3.1, "Command Field."

5.20.5.2 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 timeouts. The status field encoding is defined in Table 114, "Status Field Encoding for SMBus Reads."

Table 114. Status Field Encoding for SMBus Reads

Bit	Description
7	Internal Timeout. This bit is set if an SMBus request is not completed
6	Reserved
5	Internal Master Abort
4	Internal Target Abort
3:1	Reserved
0	Successful

Examples of configuration reads are illustrated in Figure 45, "SMBus Configuration Read (Block Write/Block Read, PEC Enabled)" through Figure 47, "SMBus Configuration Read (Write Bytes/Read Bytes, PEC Enabled)." 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 *System Management Bus (SMBus) Specification*, Version 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 "Register".

Figure 45. SMBus Configuration Read (Block Write/Block Read, PEC Enabled)

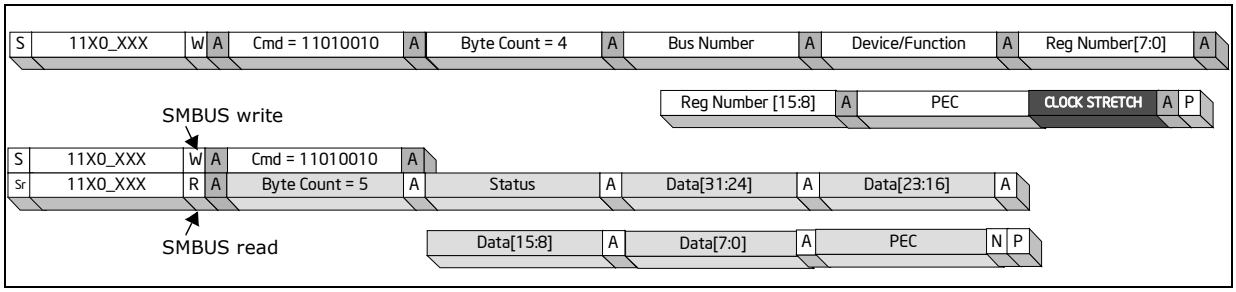


Figure 46, "SMBus Configuration Read (Word Writes/Word Reads, PEC Enabled)" is an example using word reads. The final data is a byte read.

Figure 46. SMBus Configuration Read (Word Writes/Word Reads, PEC Enabled)

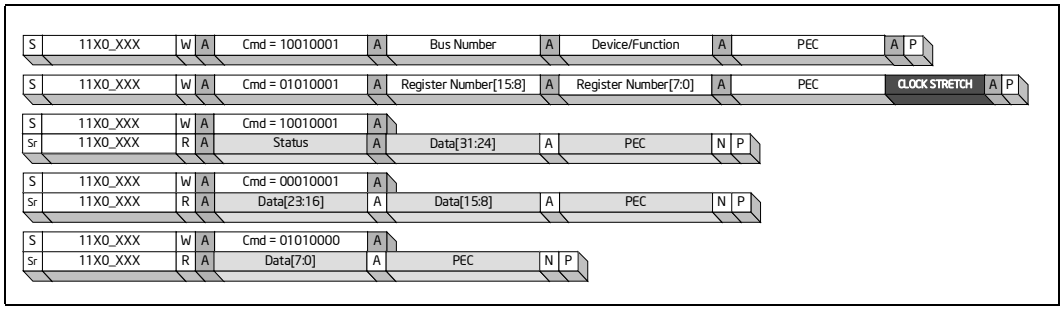
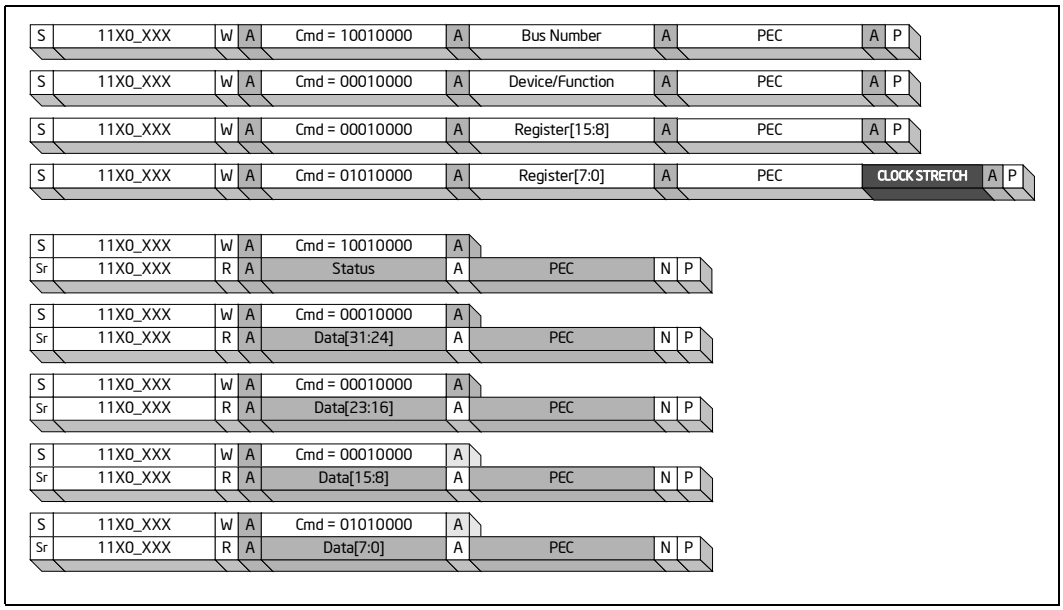


Figure 47, "SMBus Configuration Read (Write Bytes/Read Bytes, PEC Enabled)" uses byte reads.

Figure 47. SMBus Configuration Read (Write Bytes/Read Bytes, PEC Enabled)



5.20.5.3 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 in Figure 48, "SMBus Configuration Write (Block Write, PEC Enabled)" through Figure 50, "SMBus Configuration Write (Write Bytes, PEC Enabled)." For the definition of the diagram conventions below, refer to the *System Management Bus (SMBus) Specification*, Version 2.0. For diagram compactness, "Register Number" is also sometimes referred to as "Reg Number" or "Register".

Figure 48. SMBus Configuration Write (Block Write, PEC Enabled)

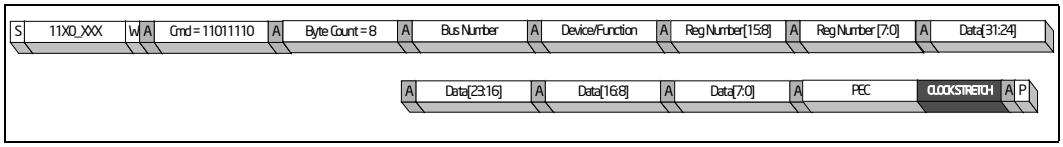


Figure 49. SMBus Configuration Write (Word Writes, PEC Enabled)

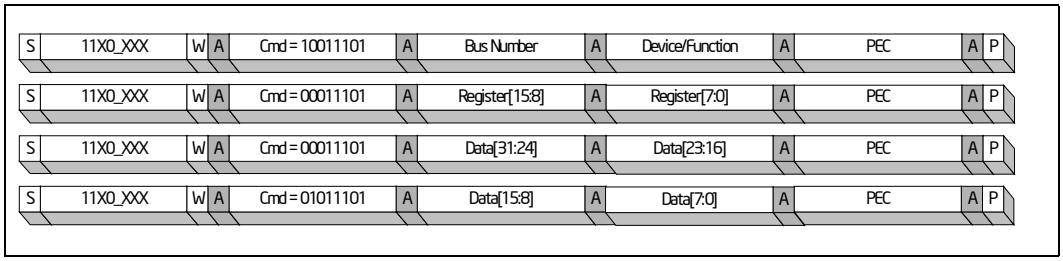
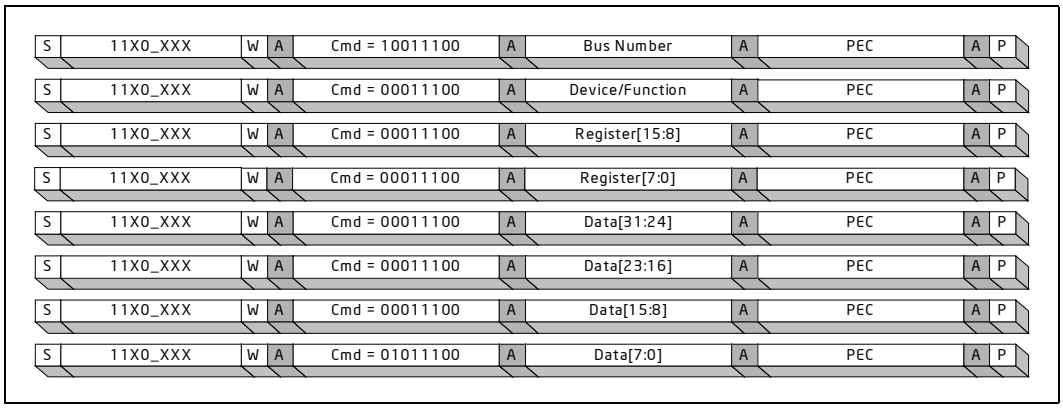


Figure 50. SMBus Configuration Write (Write Bytes, PEC Enabled)



5.20.5.4 SMBus Error Handling

The SMBus slave interface handles two types of errors: Internal and PEC. For example, internal errors can occur when the Intel® 5100 MCH 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.

5.20.5.5 SMBus Interface Reset

- The slave interface state machine can be reset by the master in two ways:
- The master holds SCL low for 25 ms cumulative. Cumulative in this case means that all the "low time" for SCL is counted between the Start and Stop bit. If this totals 25 ms before reaching the Stop bit, the interface is reset.
- The master holds SCL continuously high for 50 ms.

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.

5.20.6 DDR2 DIMM SPD0 SMBus Interface

The MCH integrates a 100 kHz SPD controller to access the DIMM configuration information. SMBus There can be a maximum of eight SPD DIMMs associated with each SPD bus. The DIMM SPD interface is wired as depicted in Figure 19, "Connection of DIMM Serial I/O Signals."

Board layout must map chip selects to SPD Slave Addresses as shown in Table 90, "SPD Addressing." The slave address is written to the SPDCMD configuration register.

5.20.6.1 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.

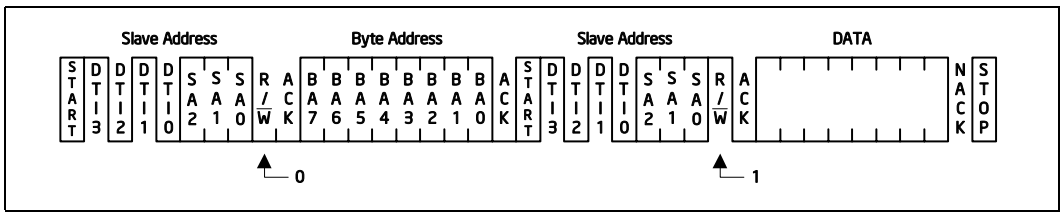
Valid SPD data is stored in the DATA field of the SPD configuration register upon successful completion of the SPDR command (indicated by 1 in the RDO field). Data to be written by an SPDW command is placed in the DATA field of the SPDCMD configuration register.

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.

5.20.6.2 Request Packet for SPD Random Read

Upon receiving the SPDR command, the MCH generates the Random Read Register command sequence as shown in Figure 51, "Random Byte Read Timing." 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.

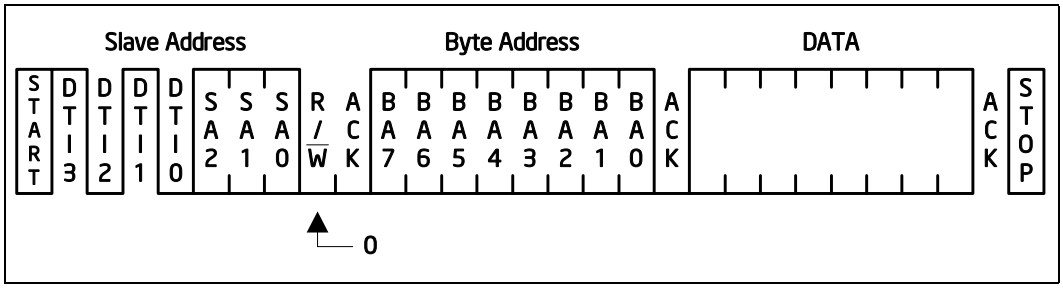
Figure 51. Random Byte Read Timing



5.20.6.3 Request Packet for SPD Byte Write

Upon receiving the SPDW command, the MCH generates the Byte Write Register command sequence as shown in Figure 52, "Byte Write Register Timing." The MCH indicates that the SIO command has completed by setting the WOD bit of the SPD configuration register to 1.

Figure 52. Byte Write Register Timing



5.20.6.4 SPD Protocols

The Intel® 5100 MCH Chipset supports the SPD protocols shown in Table 115, "Intel® 5100 Memory Controller Hub Chipset Supported SPD Protocols."

Table 115. Intel® 5100 Memory Controller Hub Chipset Supported SPD Protocols

Intel® 5100 MCH Chipset Supported SPD Protocols
Random Byte Read
Byte Write

5.20.6.5 SPD Bus Timeout

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 timeout counter within the MCH begins counting after the last bit of data is transferred to the DIMM, while the MCH waits for a response.

5.20.7 PCI Hot Plug* Support, GPIOSMBus

GPIOSMBus is the PCI Hot Plug* port. GPIOSMBus is a PCI Hot Plug* Virtual Pin Port (VPP) that operates using the SM Bus Masters protocol as defined in *System Management Bus (SMBus) Specification*, Version 2.0.

GPIOSMBus supplies support for PCI Hot Plug* devices. Support for PCI Express* is an option described in *PCI Express* Base Specification*, Rev. 1.0a. The PCI Hot Plug* model implies a PCI Hot Plug* controller per port which is identified to software as a capability of the PCI-to-PCI Bridge configuration space.

PCI Hot Plug* support requires that the Intel® 5100 MCH Chipset supports a set of PCI Hot Plug* messages to manage the states between the PCI 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 PCI Hot Plug* messages are required only if the LED indicators reside on the actual card and are accessed through the endpoint device. The Intel® 5100 MCH Chipset supports all of the PCI Hot Plug* messages so that the platform is not constrained to any particular form factor.

A standard PCI Hot Plug* usage model is beneficial to customers who buy systems with PCI 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 Standard PCI Hot Plug* Controller (SHPC), it is necessary to make some assumptions about the interface between a human operator and a PCI 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.

5.20.7.1 PCI 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. PCI 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.

5.20.7.2 Attention Button

An Attention Button is a momentary-contact push-button, located adjacent to each PCI 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.

5.20.8 PCI Hot Plug* Controller

PCI Hot Plug* requires that the Intel® 5100 MCH Chipset implement a PCI Hot Plug* controller for every PCI Hot Plug*-able interface. The PCI 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*, Rev. 1.0a.

5.20.9 PCI Hot Plug* Usage Model

Not all concepts from the PCI Hot Plug* definition apply directly to PCI Express* interfaces. The *PCI Express* Base Specification*, Rev. 1.0a 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 PCI Hot Plug* slot. PCI Express* is a point-to-point interface, making PCI 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 PCI Hot Plug* command and status codes are misleading when applied to PCI Express*.

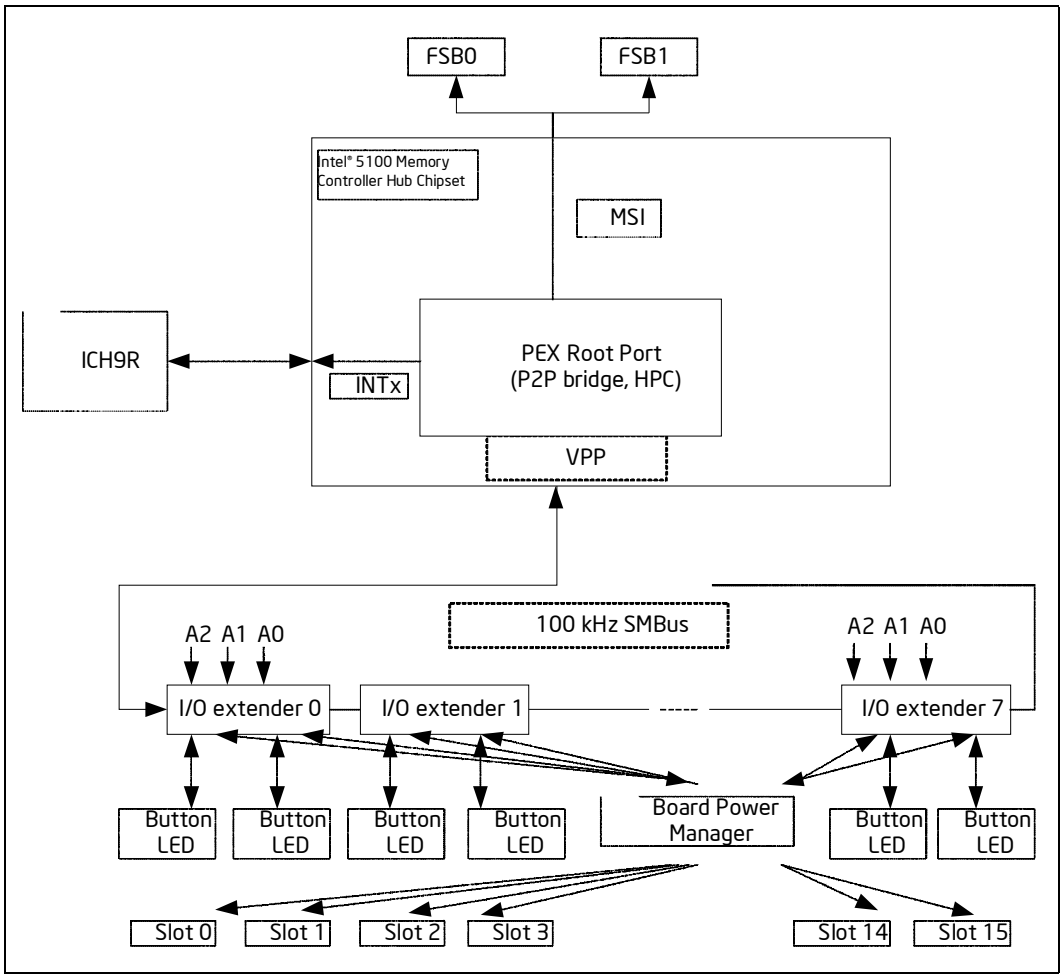
The compatible set of PCI Hot Plug* registers may be accessed via memory-mapped transactions, or via the Intel® 5100 MCH Chipset configuration mechanism. For specific information on the PCI Hot Plug* register set, refer to the Section 3.7, "Detailed Configuration Space Maps."

5.20.10 Virtual Pin Ports

Shown in the Figure 53 is a high level block diagram of virtual pin ports and theoretical maximum number of PCI Express* card slots that could be supported for PCI Hot Plug* operations. In this VPP usage model, 16 slots (max) are shown in Figure 53. However, the Intel® 5100 MCH Chipset only supports six PCI Express* ports and, therefore, can only support up to six PCI Express* slots[1] for the I/O PCI Hot Plug* operations. Accordingly, the Intel® 5100 MCH Chipset PCI Express* virtual pin port will only process six PCI Hot Plug* slots.

Note: Port 0, the ESI slot, is not PCI Hot Plug*-able.

Figure 53. PCI Hot Plug*/VPP Block Diagram



1. This does not include the ESI (port 0) which is not PCI Hot Plug*-able.

The Intel® 5100 MCH Chipset masters a 100 kHz PCI Hot Plug* SMBus interface through 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[1] I/O Extender. The Intel® 5100 MCH Chipset only supports SMBus devices with registers mapped as per Table 116. 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® 5100 MCH Chipset has a crossbar which associates each PCI Hot Plug* Unit 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 Hot Plug* Unit slots. The VPP register holds the SMBus address and port number of the I/O Port associated with the PCI Hot Plug* Unit. A[2:0] pins on each I/O Extender (i.e., Phillips* PCA9555 or compatible components) connected to the Intel® 5100 MCH Chipset must strapped uniquely. Table 117 defines how the eight PCI Hot Plug* signals are mapped to pins on the VPP.

Table 116. I/O Port Registers in I/O Extender Supported by Intel® 5100 Memory Controller Hub Chipset

Register	Name	Intel® 5100 MCH Chipset 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® 5100 MCH Chipset
5	Polarity Inversion Port 1	
6	Configuration Port 0	Direction set as per Table 117
7	Configuration Port 1	

5.20.10.1 Operation

When the Intel® 5100 MCH Chipset comes out of reset, the I/O ports are inactive. After a reset, the Intel® 5100 MCH Chipset is not aware of how many I/O Ports are connected to it, what their addresses are, nor what PCI Express* ports are PCI Hot Plug*-able. The Intel® 5100 MCH Chipset does not master any commands on the SMBus until a PCI Hot Plug* Capable bit is set.

For a PCI Express* slot, an additional DIS_VPP bit is used to differentiate card or module PCI Hot Plug* support, DIS_VPP bit needs to be set to 0 to enable PCI Hot Plug* support for PCI Express* card slot.

When BIOS sets a PCI Hot Plug* Capable bit (PEXSLOTCAP.HPC and PEXCTRL.DIS_VPP for PCI Express*), the Intel® 5100 MCH Chipset initializes the associated VPP with Direction and Voltage Logic Level configuration as per Table 117. VPP registers for PCI Express* which do not have the PCI 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 PCI 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 117.

When the Intel® 5100 MCH Chipset 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 PCI Hot Plug* capability bit is set. To minimize the completion time of this sequence and minimize complexity, both ports are always read or written.

1. The Intel® 5100 MCH Chipset VPP supports Phillips* PCA9555 or compatible I/O Extender only.

For the maximum number of six I/O Ports, and assuming no clock stretching, this sequence can take up to 51 ms. If new PCI Hot Plug* capability bits are not being set, this is the maximum timing uncertainty in sampling or driving these signals.

Table 117 describes the PCI Hot Plug* Signals used for PCI Hot Plug*.

Table 117. PCI Hot Plug* Signals on 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	High_true	MRL	MRL is open	MRL is closed
7	Input	Low_true	GPI#	Power good on Slot	No Power good on Slot

The Intel® 5100 MCH Chipset will send Assert_INTx/Deassert_INTx or Assert_HPGPE/Deassert_HPGPE messages to the ESI port as virtual pin messages to enable the ICH9R take the appropriate action for handling the PCI Hot Plug* (legacy/ACPI interrupt mode) in non-MSI mode.

5.21 Trusted Platform Module (TPM)

The Intel® 5100 MCH Chipset supports TPM v1.2 locality 0 access from CPU on the system. TPM locality 0 access is mapped to FED4_0xxxh MMIO address range. With TPM, the Intel® 5100 MCH Chipset allows security to exist within the platform. The TPM is accessed through address space FED4_0xxxh. The Intel® 5100 MCH Chipset will only open FED4_0xxxh - FED4_0FFFh to processors in the system. It is available at all times and is capable of being accessed by all processors. The Intel® 5100 MCH Chipset will convert these CPU accesses into corresponding Read/Write messages on the ESI port and send it to Intel® 82801IR I/O Controller Hub.

Only the Memory Read is non-posted and requires a completion. However, Memory Writes are posted. The completion for the Memory Read will be the same as for normal read completion cycles. If a zero length read/write is targeted to FED4_0xxxh space, the Intel® 5100 MCH Chipset has the ability to either complete the request internally or send the request south through the ESI port.

The Intel® 5100 MCH Chipset will decode TPM locality 1-4 addresses on the FSB and they will be routed to the ESI port as standard Memory transactions. The same rule also applies to Peer-to-peer accesses for locality 1-4 accesses.

Locked accesses to FED4_0xxxh range are not supported and will be master aborted by the Intel® 5100 MCH Chipset. Since the Intel® 5100 MCH Chipset does not have an internal master abort mechanism, it will be passed as MMIO transactions to ESI. All peer-to-peer requests that attempt to access the TPM space will be allowed as normal memory transactions on the ESI port (i.e., it will not be converted to Read/Write). Refer to Table 118. Such a peer-to-peer transaction will be master aborted by the ICH9R port to the originating port in the Intel® 5100 MCH Chipset.

Table 118. Decode Table in Intel® 5100 Memory Controller Hub Chipset for TPM Locality

Address Range	Transactions on FSB	Transactions on ESI
FED4_0000 - FED4_0FFF (Locality 0)	Memory Rd/Write	Read/Write (Memory Rd/Write for Peer-to-peer)
FED4_1000 - FED4_1FFF (Locality 1)	Memory Rd/Write	Memory Rd/Write
FED4_2000 - FED4_2FFF (Locality 2)	Memory Rd/Write	Memory Rd/Write
FED4_3000 - FED4_3FFF (Locality 3)	Memory Rd/Write	Memory Rd/Write
FED4_4000 - FED4_4FFF (Locality 4)	Memory Rd/Write	Memory Rd/Write

Note: The Intel® 5100 MCH Chipset will forward zero length MMIO read/write requests that originate from the FSB targeting the FED4_0xxxh range (ring 0) access to an appropriate Read/Write memory cycle of zero length on ESI. It is the responsibility of the Intel® 82801IR I/O Controller Hub to convert these transactions appropriately or internally complete the transaction back to the Intel® 5100 MCH Chipset. In addition, the Intel® 5100 MCH Chipset as a component does not impose any transaction length or address alignment restrictions for these MMIO reads/writes when they are mapped to transactions on ESI. They are passed through as they appear on FSB but as transactions on the ESI. The Intel® 5100 MCH Chipset supports attribute aliasing (WC, UC) for these CPU accesses but it is the responsibility of the software to employ the correct attributes as the case may be and should be cognizant of the inherent ordering/ serialization issues in the CPU. These requests are deferred on the FSB by the Intel® 5100 MCH Chipset, and it is entered into the ordering domain only after it is placed in IOU for dispatch to the ESI port. Also note that the ESI port in the Intel® 5100 MCH Chipset does not support Chipset Write coalescing (CSWC) (that is, does not combine smaller packets to form a larger TLP).

It is recommended that software use the "UC" attribute for the FED4_0xxxh range.

5.22 Clocking

The following sections describe the Intel® 5100 MCH Chipset clocks.

5.22.1 Reference Clocks

The BUSCLK, and CORECLK (herein referred to "in aggregate" as "BUSCLK") reference clocks, operating at 266/333 MHz, are supplied to the Intel® 5100 MCH Chipset. These are the processor bus, core, and 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 CH(0/1)_DCLK reference clocks, (herein referred to as DDRCLK) operating at half the DDR2 frequency (operating at the SDRAM command-clock frequency) are supplied by the Intel® 5100 MCH Chipset to the DIMMs.

The PECLK reference clock, operating at 100 MHz, is supplied to the Intel® 5100 MCH Chipset. 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® 5100 MCH Chipset core treats this frequency domain asynchronously.

The BUSCLK 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 119. Intel® 5100 Memory Controller Hub Chipset Frequencies for Processors and Core

Core	Domain	Frequency	Reference Clock
266 MHz	BUSCLK	266 MHz	
	FSB 1x		
	FSB 2x	533 MHz	
	FSB 4x	1,067 MHz	
333 MHz	BUSCLK	333 MHz	
	FSB 1x		
	FSB 2x	667 MHz	
	FSB 4x	1333 MHz	

Table 120. Intel® 5100 Memory Controller Hub Chipset Frequencies for Memory

DDR	Domain	Frequency	Reference Clock
533 MHz	DDR UI	3.2 GHz	DDRCLK
	DDR packet	266 MHz	
	DDR CLK	133 MHz	
667 MHz	DDR UI	4.0 GHz	
	DDR packet	333 MHz	
	DDR CLK	167 MHz	

Table 121. Intel® 5100 Memory Controller Hub Chipset Frequencies for PCI Express*

Domain	Frequency	Reference Clock
PCI Express* phit	2.5 GHz	PECLK
PCI Express* flit	250 MHz	PECLK

Notes:
1. See Section 1.1, "Terminology" for Phit and flit definitions.

5.22.2 JTAG

TCK is asynchronous to core clock. For private TAP register accesses, one TCK cycle is a minimum of ten core cycles. The TCK high time is a minimum of five core cycles in duration. The TCK low time is a minimum of five 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® 5100 MCH 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 100 kHz Virtual Pin Interface (VPI) clock (one of the SMBCLKs) is derived from the core clock.

5.22.5 Clock Pins

Table 122. Clock Pins

Pin Name	Pin Description
CORECLKP	Processor bus clock
CORECLKN	Processor bus clock (Complement)
PECLKP	PCI Express* clock
PECLKN	PCI Express* clock (Complement)
CH{0/1}_DCLKP	DDR clocks
CH{0/1}_DCLKN	DDR clocks (Complement)
PSEL[2:0]	BUSCLK:CORECLK Bus Ratio Selector
VCCDDR	Analog power supply for DDR PLLs
FSBVCCA	Analog power supply for processor bus PLL
PEVCCA	Analog power supply for PCI Express* PLLs
PEVSSA	Analog ground for PCI Express* PLLs
COREVCCA	Analog power supply for Core PLL
COREVSSA	Analog ground for Core PLL
TCK	TAP clock
GPIOSMBCLK	GPIO (Virtual Pin Port) clock
CFGSMBCLK	SMBus clock
SPD0SMBCLK	SMBus clock
XDPSTBP#	Debug bus data strobe
XDPSTBN#	Debug bus data strobe (Complement)
FSB{0/1}STBP[3:0]#	Processor bus data strobes
FSB{0/1}STBN[3:0]#	Processor bus data strobes (Complements)
FSB{0/1}ADSTB[1:0]#	Processor bus address strobes

5.22.6 High Frequency Clocking Support

5.22.6.1 Spread Spectrum Support

Intel® 5100 MCH Chipset 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, i.e., the modulation profile. The Intel® 5100 MCH Chipset 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® 5100 MCH Chipset cannot be stopped.

5.22.6.3 Jitter

The DDR UI clocks are produced by PLLs that multiply the DDRCLK frequency by 24. 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 1 ms 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® 5100 MCH Chipset 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® 5100 MCH Chipset. 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 266 MHz mode, three-quarters of the core frequency in 333 MHz mode.

However, the phase between the frequency-locked domains is not controlled. This scheme results in the possibility of a metastability resonance where, e.g., 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 Thermal Diode

The MCH incorporates an on-die diode that may be used to monitor the die temperature (junction temperature). A thermal sensor located on the motherboard or a stand-alone measurement kit may monitor the die temperature of the MCH for thermal management or characterization using the TDIOANODE and TDIOCATHODE signals.

5.24 Error List

This section provides a summary of errors detected by the Intel® 5100 MCH Chipset. Table 123, "Intel® 5100 Memory Controller Hub Chipset Error List," errors are listed by the unit/interfaces. Some units/interfaces may provide additional error logging registers.

Table 123, "Intel® 5100 Memory Controller Hub Chipset Error List" provides the list of detected errors of the MCH.

Table 123. Intel® 5100 Memory Controller Hub Chipset Error List (Sheet 1 of 7)

ERR # in MCH	Error Name	Definition	Error Type	Log Register	Cause/Actions
FSB Errors					
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, NRECADDRH, NRECADDRL 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
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. Data parity is signaled for each FSB clock based on the contents of the 4 data phases. 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.	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.	Do not propagate to other FSB bus, reset arb. unit, and programatically reset platform

Notes:
1. IO3 error logging in Intel® 5100 MCH Chipset has been defeatured due to *PCI Express* Base Specification*, Rev. 1.0a ECN (Dec. 2004). However, PEXLNKSTS.TERR provides training indication.
2. Aliased uncorrectable errors are uncorrectable errors that masquerades as correctable errors to the Memory Controller.

Table 123. Intel® 5100 Memory Controller Hub Chipset Error List (Sheet 2 of 7)

ERR # in MCH	Error Name	Definition	Error Type	Log Register	Cause/Actions
F9	FSB protocol Error	MCH detected FSB protocol error, for example, HITM on BIL and HITM on EWB and HITM on BLW (lock write).	Fatal	FERR_FAT_FSB/ NERR_FAT_FSB. NRECFSB, NRECADDRH, NRECADDRL 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. (including alignment error, i.e., not on 64B boundary)	Fatal		Halt DMA Engine
DMA3	Descriptor Error	The DMA channel sets this bit indicating that the current transfer has encountered an error (not otherwise covered under other DMA error bits) when executing a DMA descriptor.	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 CHAINADDR 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 (e.g., 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

Notes:
1. IO3 error logging in Intel® 5100 MCH Chipset has been defeatured due to *PCI Express* Base Specification*, Rev. 1.0a ECN (Dec. 2004). However, PEXLNKSTS.TERR provides training indication.
2. Aliased uncorrectable errors are uncorrectable errors that masquerades as correctable errors to the Memory Controller.

Table 123. Intel® 5100 Memory Controller Hub Chipset Error List (Sheet 3 of 7)

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 (e.g., 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 (e.g., 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

Notes:
1. IO3 error logging in Intel® 5100 MCH Chipset has been defeatured due to *PCI Express* Base Specification*, Rev. 1.0a ECN (Dec. 2004). However, PEXLNKSTS.TERR provides training indication.
2. Aliased uncorrectable errors are uncorrectable errors that masquerades as correctable errors to the Memory Controller.

Table 123. Intel® 5100 Memory Controller Hub Chipset Error List (Sheet 4 of 7)

ERR # in MCH	Error Name	Definition	Error Type	Log Register	Cause/Actions
PCI Express* Errors					
IO0	PCI Express* - Data Link Layer Protocol Error	MCH detects a DL layer protocol error from the DLLP.	Default =Fatal (Check UNCERR SEV)	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 CORERRSTS for their respective Error Types. Log PEXDEVSTS for IO12 and other I/O errors based on UNCERRSEV.	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 UNCERR SEV)		Log header of packet. The header log may be not valid for all cases of unsupported request. Check corresponding bit in UNCERRSEV register for severity level (Fatal or Non Fatal)
IO3[1]	PCI Express* - Training Error	PCI Express* link initially trained successfully, but failed re-training.	Default =Fatal (Check UNCERR SEV)		Log header of packets with errors Check corresponding bit in UNCERRSEV register for severity level (Fatal or Non Fatal) Removed by PCI-SIG Errata to *PCI Express* Base Specification*, Rev. 1.0a. No longer supported.
IO4	PCI Express* - Poisoned TLP	Received a poisoned transaction layer packet from the south bridge.	Default =UnCorr (Check UNCERR SEV)		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 UNCERR SEV)		Log header of packets with errors Check corresponding bit in UNCERRSEV register for severity level (Fatal or Non Fatal)

Notes:
1. IO3 error logging in Intel® 5100 MCH Chipset has been defeatured due to *PCI Express* Base Specification*, Rev. 1.0a ECN (Dec. 2004). However, PEXLNKSTS.TERR provides training indication.
2. Aliased uncorrectable errors are uncorrectable errors that masquerades as correctable errors to the Memory Controller.

Table 123. Intel® 5100 Memory Controller Hub Chipset Error List (Sheet 5 of 7)

ERR # in MCH	Error Name	Definition	Error Type	Log Register	Cause/Actions
IO6	PCI Express* - Completion Timeout	Pending transaction did not complete within the time limit.	Default =UnCorr (Check UNCERR SEV)	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 CORERRSTS for their respective Error Types. Log PEXDEVSTS for IO12 and other I/O errors based on UNCERRSEV	Check corresponding bit in UNCERRSEV register for severity level (Fatal or Non Fatal)
IO7	PCI Express* - Completer Abort	Received return CA status for unknown error on the component. This is equivalent to a target abort on PCI.	Default =UnCorr (Check UNCERR SEV)		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 UNCERR SEV)		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 UNCERR SEV)		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 UNCERR SEV)		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 CORERRSTS for their respective Error Types.
IO12	PCI Express* - Receiver Error	Log header of packets with errors	Corr		Log CORERRSTS for their respective Error Types.
IO13	PCI Express* - Bad TLP Error	Received bad CRC or a bad sequence number in a transport layer packet.	Corr		Log CORERRSTS for their respective Error Types.
IO14	PCI Express* - BAD DLLP	Received bad CRC in a data link layer packet.	Corr		Log CORERRSTS for their respective Error Types.
IO15	PCI Express* - Replay_Num Rollover	Replay maximum count for the Retry Buffer has been exceeded.	Corr		Log CORERRSTS for their respective Error Types.
IO16	PCI Express* - Replay Timer Timeout	Replay timer timed out waiting for an Ack or Nak DLLP.	Corr		Log CORERRSTS for their respective Error Types.
IO17	PCI Express* - Received Correctable Error Message	MCH received a correctable error message from the south bridge.	Corr		Log CORERRSTS for their respective Error Types.

Notes:

1. IO3 error logging in Intel® 5100 MCH Chipset has been defeatured due to *PCI Express* Base Specification*, Rev. 1.0a ECN (Dec. 2004). However, PEXLNKSTS.TERR provides training indication.
2. Aliased uncorrectable errors are uncorrectable errors that masquerades as correctable errors to the Memory Controller.

Table 123. Intel® 5100 Memory Controller Hub Chipset Error List (Sheet 6 of 7)

ERR # in MCH	Error Name	Definition	Error Type	Log Register	Cause/Actions
IO18	ESI reset timeout	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.
IO19	Surprise Link Down	IOU LTSSM detected a link down condition (surprise) during normal operation	Fatal	Log PEX_FAT_FERR/NERR or PEX_NF_COR_FERR/ NERR based on their respective Error types and Severity (UNCERRSEV)	Link went down suddenly and status bits are set for software to take any action.
MCH Internal Errors					
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.
B3	MCH - Coherency Violation EWB Error	MCH detected a cache coherency protocol error for EWB.	Fatal	FERR_FAT_INT/ NERR_FAT_INT and NRECINT	Log CE entry on FERR This applies to SF enable mode and is not applicable for the Intel® 5100 MCH Chipset.
B4	Virtual Pin Interface Error	MCH detected an error on the virtual pin interface	Fatal	FERR_FAT_INT/ NERR_FAT_INT and NRECINT	Log VPP error in the respective PEX and DDR Channel cluster debug registers
B5	MCH - Address Map Error	MCH detected address mapping error due to software programming error. The errors are described in system address map section	UnCorr	FERR_NF_INT/ NERR_NF_INT and NRECINT	MCH might malfunction.
B8	MCH Coherency Violation BIL Error	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_NF_INT/ NERR_NF_INT and NRECINT	Log CE entry on FERR. This applies to SF enable mode and is not applicable for the Intel® 5100 MCH Chipset.
Memory Subsystem					
M1	Uncorrectable Data ECC Error on DDR Replay	The MCH detected an uncorrectable data ECC error during replay of the head of the DDR replay queue	Uncorr	FERR_NF_MEM/ NERR_NF_MEM and VALIDLOG[1:0], NRECMEMA and NRECMEMB	See Figure 54, "DDR Error Recovery Scheme" on page 376.
M4	Aliased Uncorrectable[2] Demand Data ECC Error	The MCH determined that a normally "correctable" error could be an aliased (x4 only) full device failure plus an additional single bit error.	Rec	FERR_NF_MEM/ NERR_NF_MEM and VALIDLOG[1:0], RECMEMA and RECMEMB	See Figure 54, "DDR Error Recovery Scheme" on page 376.

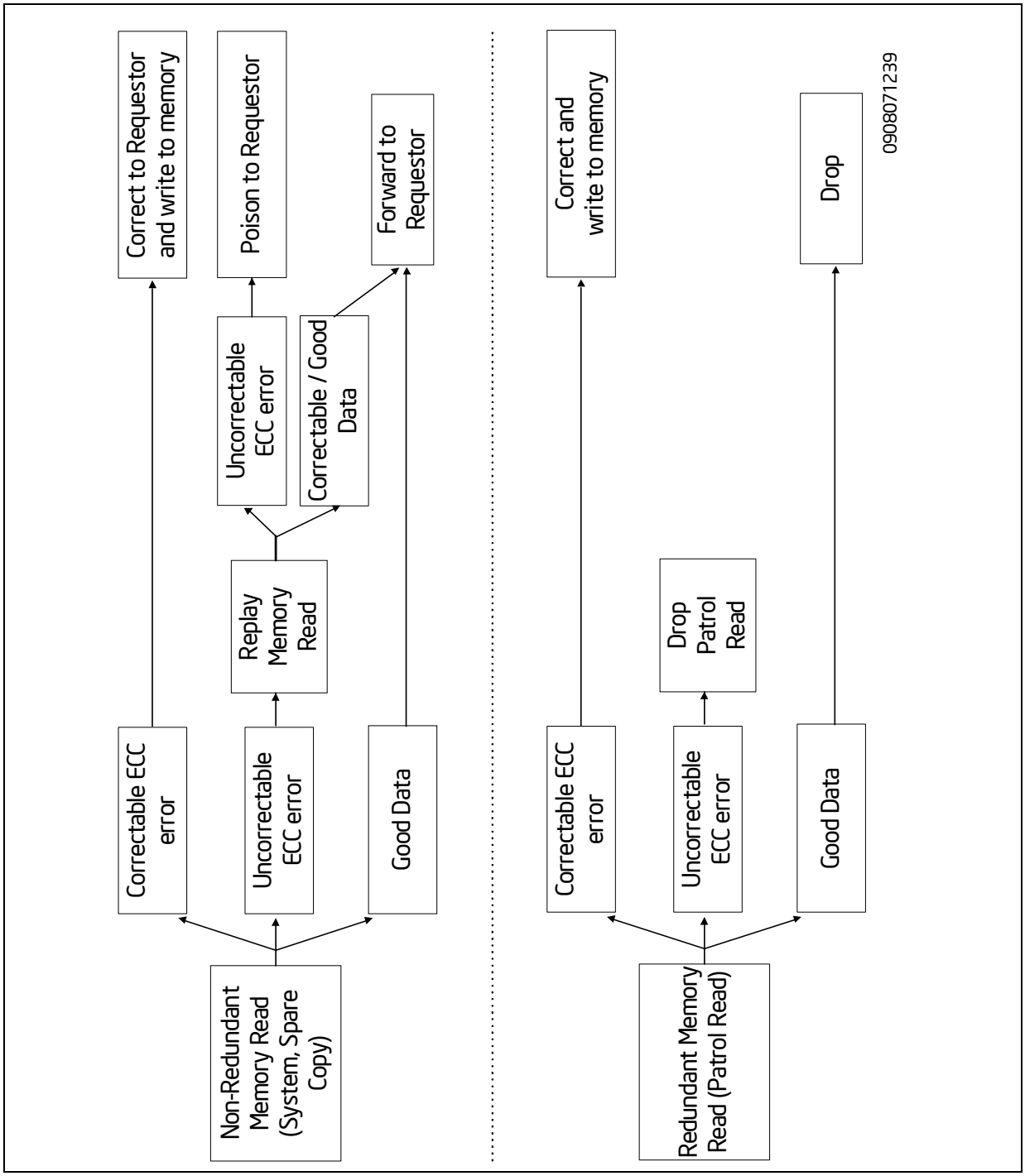
Notes:
1. IO3 error logging in Intel® 5100 MCH Chipset has been defeatured due to *PCI Express* Base Specification*, Rev. 1.0a ECN (Dec. 2004). However, PEXLNKSTS.TERR provides training indication.
2. Aliased uncorrectable errors are uncorrectable errors that masquerades as correctable errors to the Memory Controller.

Table 123. Intel® 5100 Memory Controller Hub Chipset Error List (Sheet 7 of 7)

ERR # in MCH	Error Name	Definition	Error Type	Log Register	Cause/Actions
M5	Aliased Uncorrectable Spare-Copy Data ECC Error	During a Sparing copy read from the failing rank the MCH determined that a normally "correctable" error could be an aliased (x4 only) full device failure plus an additional single bit error.	Rec	FERR_NF_MEM/ NERR_NF_MEM and VALIDLOG[1:0], RECMEMA and RECMEMB	See Figure 54, "DDR Error Recovery Scheme" on page 376.
M6	Aliased Uncorrectable Patrol Data ECC Error	During a Patrol Scrub, the MCH determined that a normally "correctable" error could be an aliased (x4 only) full device failure plus an additional single bit error.	Rec	FERR_NF_MEM/ NERR_NF_MEM and VALIDLOG[1:0], RECMEMA and RECMEMB	See Figure 54, "DDR Error Recovery Scheme" on page 376.
M10	Non-Aliased Uncorrectable Demand Data ECC Error	The MCH detected uncorrectable data on a demand read.	Rec	FERR_NF_MEM/ NERR_NF_MEM and VALIDLOG[1:0], RECMEMA and RECMEMB	See Figure 54, "DDR Error Recovery Scheme" on page 376.
M11	Non-Aliased Uncorrectable Spare-Copy Data ECC Error	The MCH detected uncorrectable data rank during a spare copy read.	Rec	FERR_NF_MEM/ NERR_NF_MEM and VALIDLOG[1:0], RECMEMA and RECMEMB	See Figure 54, "DDR Error Recovery Scheme" on page 376.
M12	Non-Aliased Uncorrectable Patrol Data ECC Error	During a patrol scrub, the MCH detected uncorrectable data.	Rec	FERR_NF_MEM/ NERR_NF_MEM and VALIDLOG[1:0], RECMEMA and RECMEMB	See Figure 54, "DDR Error Recovery Scheme" on page 376.
M14	Correctable Demand Data ECC Error.	The MCH detected correctable data.	Corr	FERR_NF_MEM/ NERR_NF_MEM and VALIDLOG[1:0], REDMEMA and REDMEMB	See Figure 54, "DDR Error Recovery Scheme" on page 376.
M15	Correctable Spare-Copy Data ECC Error	The MCH detected correctable data from the failing rank during a spare copy.	Corr	FERR_NF_MEM/ NERR_NF_MEM and VALIDLOG[1:0], REDMEMA and REDMEMB	See Figure 54, "DDR Error Recovery Scheme" on page 376.
M16	Correctable Patrol Data ECC Error	During a patrol scrub, the MCH detected correctable data.	Corr	FERR_NF_MEM/ NERR_NF_MEM and VALIDLOG[1:0], REDMEMA and REDMEMB	See Figure 54, "DDR Error Recovery Scheme" on page 376.
M18	SPD protocol Error	The MCH detected an SPD interface error.	Corr	FERR_NF_MEM/ NERR_NF_MEM	Successive correction attempts performed by software.
M20	Rank-Spare Copy start	Triggered Spare copy	Corr	SPCPC[1:0]	Start spare copy
M21	Rank-Spare Copy complete	Spare copy completed normally	Corr	SPCPS[1:0]	No Action

Notes:
1. IO3 error logging in Intel® 5100 MCH Chipset has been defeatured due to *PCI Express* Base Specification*, Rev. 1.0a ECN (Dec. 2004). However, PEXLNKSTS.TERR provides training indication.
2. Aliased uncorrectable errors are uncorrectable errors that masquerades as correctable errors to the Memory Controller.

Figure 54. DDR Error Recovery Scheme



6.0 Electrical Characteristics

6.1 Absolute Maximum Ratings

Table 124 lists the maximum environmental stress ratings for the Intel® 5100 MCH Chipset. Functional operation at or exceeding the absolute maximum and minimum ratings is neither implied nor guaranteed.

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 124. Absolute Maximum Ratings

Symbol	Parameter	Min.	Max.	Unit
$T_{storage}$	Storage Temperature[1]	-45.0	75.0	°C
V_{CC}	MCH Supply Voltages 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

Notes:
1. The component storage temperature range (Pre-Board Assembly) is -45 °C to 75 °C for short-term exposure and -10 °C to 45 °C for sustained exposure. In addition to this storage temperature specification, compliance to the latest IPC/JEDEC J-STD-033B.1 joint industry standard is required for all Surface Mount Devices (SMDs). This document governs handling, packing, shipping, and use of moisture/reflow sensitive SMDs.

6.1.1 Thermal Characteristics

See Table 135, "Thermal Diode Parameters" for the Thermal Diode Parameters. For additional information on thermal characteristics, consult the *Intel® 5100 Memory Controller Hub Chipset for Communications, Embedded, and Storage Applications Thermal/Mechanical Design Guide*.

6.1.2 Power Characteristics

Table 125. Operating Condition Power Supply Rails

Symbol	Parameter	AC Min.	DC Min.	Nom.	DC Max.	AC Max.	Unit	Notes
V_{TT}	1.2 V Host AGTL+ Termination Voltage	1.140	1.164	1.20	1.236	1.260	V	1, 2, 8
V_{TT}	1.1 V Host AGTL+ Termination Voltage	1.045	1.067	1.10	1.133	1.155	V	1, 2, 8
V_{TT}	1.05 V Host AGTL+ Termination Voltage	0.998	1.019	1.05	1.081	1.102	V	1, 2, 8
I_{TT}	Host AGTL+ Termination Current			2.6	4.8		A	
V_{CC}	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			8	9.6		A	
VCCPE	1.5 V PCI Express* Supply Voltage	1.425	1.455	1.53	1.575	1.605	V	1, 2
ICCPE	1.5 V PCI Express* Supply Current			1.8	2.16		A	
VCCDDR	1.8 V DDR2 Supply Voltage	1.701	1.755	1.8	1.845	1.899	V	1, 2, 3
ICCDDR	1.8 V DDR2 Supply Current			1.7	2		A	4
V3REF	3.3 V Supply Voltage	3.119	3.218	3.3	3.382	3.481	V	
I3REF	3.3 V Supply Current			10			mA	9
FSBVCCA	Analog PLL Voltage	1.405		1.5	1.545		V	5
FSBICCA	Analog PLL Current				28.9		mA	6
COREVCCA	1.5 V Analog Supply Voltage	1.405		1.5	1.545		V	5
ICCA	1.5 V Analog Supply Current				28.9		mA	
PEVCCA	Analog PCI Express* Voltage	1.405		1.5	1.545			5
PEICCA	Analog PCI Express* Current				52		mA	7
PEVCCBG	Analog PCI Express* Bandgap Voltage		2.425	2.5	2.575		V	5
PEICCBG	Analog PCI Express* Bandgap Current				600		µA	

Notes:

1. Under no circumstances may the supply voltage go past the AC minimum/maximum window. The supply voltage may go outside the DC minimum/maximum window for transient events.
2. The supply voltage must stay within the DC minimum/maximum window in a static system (no active switching). The DC window only assumes voltage regulator ripple and motherboard induced noise.
3. There is a V_{TT} termination supply at VCCDDR/2 available on the DIMM, but it does not connect to the Intel® 5100 MCH Chipset.
4. Memory configuration of two channels using two DIMMs, each channel with two ranks per DIMM for a total implementation of eight ranks
5. The analog voltage is intended to be a filtered copy of its associated supply voltage. Refer to the *Quad-Core and Dual-Core Intel® Xeon® Processor 5000 Sequence with Intel® 5100 Memory Controller Hub Chipset for Communications, Embedded, and Storage Applications – Platform Design Guide* or *Intel® Core™2 Duo Processors T9400 and SL9400 and Intel® 5100 Memory Controller Hub Chipset for Communications and Embedded Applications – Platform Design Guide* for the recommended implementation and frequency response requirements of each filter.
6. Current consumption based upon an FSB operating rate of 1333 MT/s.
7. Current consumption is irrespective of the number of PCI Express* ports enabled.
8. The MCH supports multiple V_{TT} supplies, one at nominal 1.05 V, one at nominal 1.1 V and one at nominal 1.2 V, which are based upon the companion processors' V_{TT} requirements. VTT 1.05 V (Intel® Core™2 Duo Processor T9400) is only supported at 1066 MT/s, VTT 1.1 V is supported at 1066/1333 MT/s, and VTT 1.2 V is supported at 1066/1333 MT/s.
9. Values will be updated upon completion of device characterization.

6.2 DC Characteristics

This section documents the DC characteristics of the MCH. The specifications are split into seven sections:

- "Clocks"
- "Front Side Bus (FSB) Interface"
- "DDR2 Interface"
- "PCI Express*/ESI Interface"
- "SMBus Interfaces and Error Signals"
- "JTAG Interface"
- "Miscellaneous"

6.2.1 Clocks

Table 126. Clock DC Characteristics (Sheet 1 of 2)

Symbol	Parameter	Min.	Nom.	Max.	Unit	Notes
333 MHz FSB Clock (CORECLKN/CORECLKP)						
V_{IL}	Input Low Voltage	-0.150	0	0.150	V	
V_{IH}	Input High Voltage	0.660	0.700	0.850	V	
$V_{Cross(abs)}$	Absolute Crossing Point	0.250		0.550	V	1, 6
$V_{Cross(rel)}$	Relative Crossing Point	0.250+0.5 x(V_{Havg}-0.700)		0.550-0.5 x(0.700-V_{Havg})	V	6, 7
ΔV_{Cross}	Range of Crossing Points			0.140	V	
V_{OS}	Overshoot			V_{IH}+0.300	V	2
V_{US}	Undershoot	-0.300			V	3
V_{RBM}	Ringback Margin	0.200			V	4
V_{TR}	Threshold Region	V_{Cross}-0.100		V_{Cross}+0.100	V	5
266 MHz FSB Clock (CORECLKN/CORECLKP)						
V_{IL}	Input Low Voltage	-0.150	0	0.150	V	
V_{IH}	Input High Voltage	0.660	0.700	0.850	V	
$V_{Cross(abs)}$	Absolute Crossing Point	0.250		0.550	V	1, 6
$V_{Cross(rel)}$	Relative Crossing Point	0.250+0.5 x(V_{Havg}-0.700)		0.550-0.5 x(0.700-V_{Havg})	V	6, 7
ΔV_{Cross}	Range of Crossing Points			0.140	V	
V_{OS}	Overshoot			V_{IH}+0.300	V	2
V_{US}	Undershoot	-0.300			V	3

Notes:

1. Crossing voltage is defined as the instantaneous voltage when the rising edge of CORECLKP is equal to the falling edge of CORECLKN.
2. Overshoot is defined as the absolute value of the maximum voltage.
3. Undershoot is defined as the absolute value of the minimum voltage.
4. 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.
5. Threshold Region is defined as a region centered around the crossing point voltage in which the differential receiver switches. It includes input threshold hysteresis.
6. The crossing point must meet the absolute and relative crossing point specifications simultaneously.
7. V_{Havg} (the average of V_{IH}) can be measured directly using "Vtop" on Agilent* scopes and "High" on Tektronix* scopes.

Table 126. Clock DC Characteristics (Sheet 2 of 2)

Symbol	Parameter	Min.	Nom.	Max.	Unit	Notes
V_{RBM}	Ringback Margin	0.200			V	4
V_{TR}	Threshold Region	V_{Cross}-0.100		V_{Cross}+0.100	V	5
100 MHz PCI Express* Clock (PECLKN/PECLKP)						
V_{IL}	Input Low Voltage	-0.150	0		V	
V_{IH}	Input High Voltage	0.660	0.700	0.850	V	
$V_{Cross(abs)}$	Absolute Crossing Point	0.250		0.550	V	1, 6
$V_{Cross(rel)}$	Relative Crossing Point	0.250+0.5 x(V_{Havg}-0.700)		0.550+0.5 x(V_{Havg}-0.700)	V	6, 7
ΔV_{Cross}	Range of Crossing Points			0.140	V	1
V_{OS}	Overshoot			V_{IH}+0.300	V	2
V_{US}	Undershoot	-0.300			V	3
V_{RBM}	Ringback Margin	0.200			V	4
V_{TR}	Threshold Region	V_{Cross}-0.100		V_{Cross}+0.100	V	5
DDR2 Clock (CH{0/1}_DCLKN/CH{0/1}_DCLKP)						
V_{OH}	High Output Voltage	VCCDDR/2 + 0.35			V	
V_{OL}	Low Output Voltage			VCCDDR/2 - 0.35	V	
V_{OX}	Crossing Voltage Requirement	VCCDDR/2 - 0.100		VCCDDR/2 +0.100	V	
Slew	Slew-rate Requirement	2.2		3.2	V/ns	
Impedance	Output Impedance	13		20	Ω	

Notes:

1. Crossing voltage is defined as the instantaneous voltage when the rising edge of CORECLKP is equal to the falling edge of CORECLKN.
2. Overshoot is defined as the absolute value of the maximum voltage.
3. Undershoot is defined as the absolute value of the minimum voltage.
4. 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.
5. Threshold Region is defined as a region centered around the crossing point voltage in which the differential receiver switches. It includes input threshold hysteresis.
6. The crossing point must meet the absolute and relative crossing point specifications simultaneously.
7. V_{Havg} (the average of V_{IH}) can be measured directly using "Vtop" on Agilent* scopes and "High" on Tektronix* scopes.

6.2.2 Front Side Bus (FSB) Interface

Table 127. FSB DC Characteristics

Symbol	Parameter	Min.	Nom.	Max.	Unit	Notes
V_{IL}	Host AGTL+ Input Low Voltage	0		GTLREF-(0.1xV_{TT})	V	1, 2
V_{IH}	Host AGTL+ Input High Voltage	GTLREF+(0.1xV_{TT})		V_{TT}	V	1, 3
V_{OL}	Host AGTL+ Output Low Voltage			0.4	V	
V_{OH}	Host AGTL+ Output High Voltage	0.90xV_{TT}		V_{TT}	V	4
I_{OL}	Host AGTL+ Output Low Current			V_{TT}/(0.50x R_{tt_min}+R_{on_min})	mA	8
I_{LI}	Host AGTL+ Input Leakage Current	N/A		± 200	µA	5, 6
I_{LO}	Host AGTL+ Output Leakage Current	N/A		± 200	µA	5, 6
R_{on}	Buffer on Resistance	7		11	Ω	
GTLREF	Host Bus Reference Voltage	(0.98x0.67)xV_{TT}	0.67xV_{TT}	(1.02x0.67)xV_{TT}	V	1
R_{TT}	Host Termination Resistance Common Clock, Async on Stripline	45	50	55	Ω	7

Notes:
1. GTLREF is equivalent to FSB{0/1}VREF. GTLREF is generated from V_{TT} on the baseboard by a voltage divider of 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 Section 5.0, "Overshoot/Undershoot Tables" of the *Intel® 5100 Memory Controller Hub Chipset (embedded) – External Design Specification (EDS) Addendum*.
5. Leakage to Vss with land held at V_{TT}
6. Leakage to V_{TT} with land held at 300 mV
7. Use 50 Ω±15% for all microstrip.
8. I_{OL} is defined as current when the output is low. The formula computes the total current drawn by the driver from the Vr (Voltage Regulator). Half of the total current goes through R_{TT} on the chipset, and another half goes through the R_{TT} on the CPU (the end-bus-agency).

6.2.3 DDR2 Interface

Table 128. DDR2 DC Characteristics (Sheet 1 of 2)

Symbol	Parameter	Min.	Nom.	Max.	Unit	Notes
Single-ended Signals						
V_{IN}	Input Voltage	0	-	V_{DD}		
$V_{IH(dc)}$	DC High-level Input Voltage	VCCDDR/2+100	-	VCCDDR+300	mV	
$V_{IL(dc)}$	DC Low-level Input Voltage	-300	-	VCCDDR/2-100	mV	
V_{OTR}	Output Timing Measurement Reference Level	-	0.5x VCCDDR	-	V	
$I_{OH(dc)}$	Output Minimum Source DC Current	-13.8	-	-	mA	2
$I_{OL(dc)}$	Output Minimum Sink DC Current	13.8	-	-	mA	2

Notes:
1. Input voltage for all pins is limited to a maximum of 2.3 V.
2. VCCDDR/2=1.7/2=850 mV
3. C_{IO} is the input/output capacitance for DQ/DQS and output capacitance for CMD/ADDR/CLK.

Table 128. DDR2 DC Characteristics (Sheet 2 of 2)

Symbol	Parameter	Min.	Nom.	Max.	Unit	Notes
Differential Signals						
$V_{ID(dc)}$	DC Differential Input Voltage	0.2	-	-	V	
I_{IH}	Input Leakage Current (High)	-	-	10	µA	
I_{IL}	Input Leakage Current (Low)	-	-	10	µA	
C_{IO}	Input/Output Capacitance	2.0	-	2.5	pF	3

Notes:
1. Input voltage for all pins is limited to a maximum of 2.3 V.
2. VCCDDR/2=1.7/2=850 mV
3. C_{IO} is the input/output capacitance for DQ/DQS and output capacitance for CMD/ADDR/CLK.

6.2.4 PCI Express*/ESI Interface

Table 129. PCI Express*/ESI Differential Transmitter (TX) Output DC Characteristics

Symbol	Parameter	Min.	Nom.	Max.	Unit	Notes
$V_{TX-DIF-DC}$	Differential Peak-to-peak Output Voltage	0.8		1.2	V	2
$V_{TX-CM-DC-ACTIVE-IDLE-DELTA}$	Absolute Delta of DC Common Mode Voltage During L0 and Electrical Idle	0		100	mV	2
$V_{TX-CM-DC-LINE-DELTA}$	Absolute Delta of DC Common Mode Voltage between D+ and D-	0		25	mV	2
$V_{TX-IDLE-DIFFp}$	Electrical Idle Differential Peak Output Voltage			20	mV	2
$V_{TX-RCV-DETECT}$	Amount of Voltage Change Allowed during Receiver Detection			600	mV	
$V_{TX-DC-CM}$	TX DC Common Mode Voltage	0		3.6	V	2
$I_{TX-SHORT}$	Short Circuit Current Limit			90	mA	
$Z_{TX-DIFF-DC}$	DC Differential TX Impedance	80	100	120	Ω	
Z_{TX-DC}	Transmitter DC Impedance	40			Ω	

Notes:
1. A test load is not required to be associated with the values in the table.
2. Specified at the measurement point into a timing and voltage compliance test load and measured over any 250 consecutive TX UIs

Table 130. PCI Express*/ESI Differential Receiver (RX) Input DC Characteristics

Symbol	Parameter	Min.	Nom.	Max.	Unit	Notes
$Z_{RX-DIFF-DC}$	DC Differential Input Impedance	80	100	120	Ω	5
Z_{RX-DC}	DC Input Impedance	40	50	60	Ω	2, 3
$Z_{RX-HIGH-IMP-DC}$	Powerdown DC Input Common Mode Impedance	200			kΩ	6
$V_{RX-IDLE-DET-DIFFp}$	Electrical Idle Detect Threshold	65		175	mV	

Notes:
1. A test load is not required to be associated with the values in the table.
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=0.40 UI provides for a total sum of 0.60 UI deterministic and random jitter budget for the transmitter and interconnect collected over any 250 consecutive UIs. The TRX-EYE-MEDIAN-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 0.6 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 is 50 Ω to ground for both the D+ and D- line (that is, as measured by a Vector Network Analyzer with 50 Ω probes). Note that the series capacitors CTX is optional for the return loss measurement.
5. Impedance during all Link Training and Status State Machine (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.

6.2.5 SMBus Interfaces and Error Signals

Table 131. DC Characteristics (3.3 V OD)

Symbol	Parameter	Min.	Nom.	Max.	Unit	Notes
V_{IH}	Input High Voltage	2.1			V	
V_{IL}	Input Low Voltage			0.8	V	
V_{OL}	Output Low Voltage			0.4	V	1
I_{OL}	Output Low Current			4	mA	
I_{Leak}	Leakage Current			10	µA	
C_{Pad}	Pad Capacitance			10	pF	

Notes:
1. At V_{OL} maximum, I_{OL}=maximum
2. SMBus signals are OD when driving the signal. When not driving, the signals are inputs as specified.

6.2.6 JTAG Interface

Table 132. JTAG DC Characteristics (1.5 V)

Symbol	Parameter	Min.	Nom.	Max.	Unit	Notes
V_{IH}	Input High Voltage	0.9			V	
V_{IL}	Input Low Voltage			0.5	V	
V_{OL}	Output Low Voltage			0.4	V	
I_{Leak}	Leakage Current			2.9	µA	

6.2.7 Miscellaneous

Table 133. CMOS DC Characteristics (1.5 V)

Symbol	Parameter	Min.	Nom.	Max.	Unit	Notes
V_{IH}	Input High Voltage	1.0		1.6	V	
V_{IL}	Input Low Voltage	-0.2		0.5	V	
V_{OH}	Output High Voltage	1.1			V	
V_{OL}	Output Low Voltage			0.4	V	
I_{Leak}	Leakage Current			70	µA	
V_{ABS}	Input Damage Thresholds	-0.2		1.6	V	

Notes:
1. FSBSLWCTRL, PSEL[2:0], ASYNCRFSH, 48GB_Mode signals.

Table 134. CMOS DC Characteristics (3.3 V)

Symbol	Parameter	Min.	Nom.	Max.	Unit	Notes
V_{IH}	Input High Voltage	2.1			V	
V_{IL}	Input Low Voltage			0.8	V	
I_{Leak}	Leakage Current			10	µA	
V_{ABS}	Input Damage Thresholds	-0.3		3.5	V	

Notes:
1. RESETI#, PWRGOOD signals.

Table 135. Thermal Diode Parameters

The MCH incorporates an on-die diode that may be used to monitor the die temperature (junction temperature). A thermal sensor located on the motherboard or a stand-alone measurement kit may monitor the die temperature of the MCH for thermal management or characterization (see Table 136 for Thermal Diode signals).

Symbol	Parameter	Min.	Nom.	Max.	Unit	Notes
I_{fb}	Current Forward Bias	1	---	300	µA	1
n_ideality	Diode Ideality factor	1.0027	1.0032	1.0036		2, 4
ESR	Effective Series Resistance	2.3	2.7	2.9	Ω	3

Notes:
1. Intel does not support or recommend operation of the thermal diode under reverse bias.
2. At room temperature with a forward bias of 630 mV
3. ESR is needed for various thermal diode measurement tools.
4. The diode ideality factor is represented by the diode equation below, where
   i = Collector current (I_{fb})
   Is = saturation current
   v = voltage across transistor
   q = charge of an electron, $1.6021892 \times 10^{-19}$
   n = Diode Ideality factor (n_ideality)
   k = Boltzmann Constant, 1.380662×10^{-23} J/K x molecule
   T = temperature in degrees Kelvin (K).

$$i = I_s\left(e^{\frac{vq}{nkT}} - 1\right)$$

Table 136. Thermal Diode Interface

Signal	Signal Description
TDIOANODE	Diode Anode (p_junction)
TDIOCATHODE	Diode Cathode (n_junction)

7.0 Testability

7.1 JTAG Port

Each component in the Intel® 5100 MCH 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-in, for details reference the IEEE 1149.1 standard.

7.1.1 TAP Signals

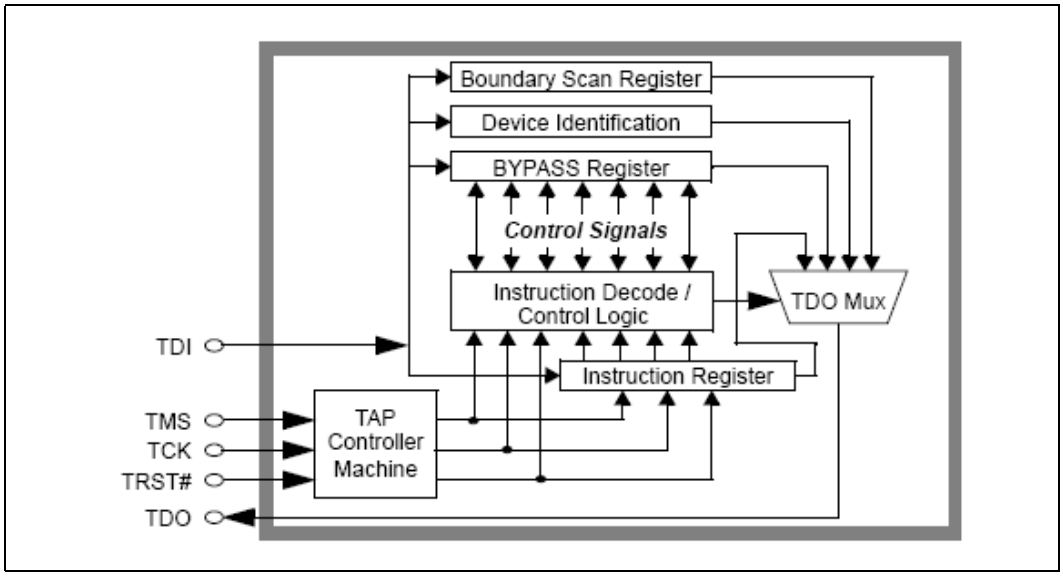
The TAP logic is accessed serially through five dedicated pins on each component as shown in Table 137, "TAP Signal Definitions."

Table 137. 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 Intel® 5100 MCH Chipset components is shown in Figure 55, "Simplified TAP Controller Block Diagram."

Figure 55. Simplified TAP Controller Block Diagram

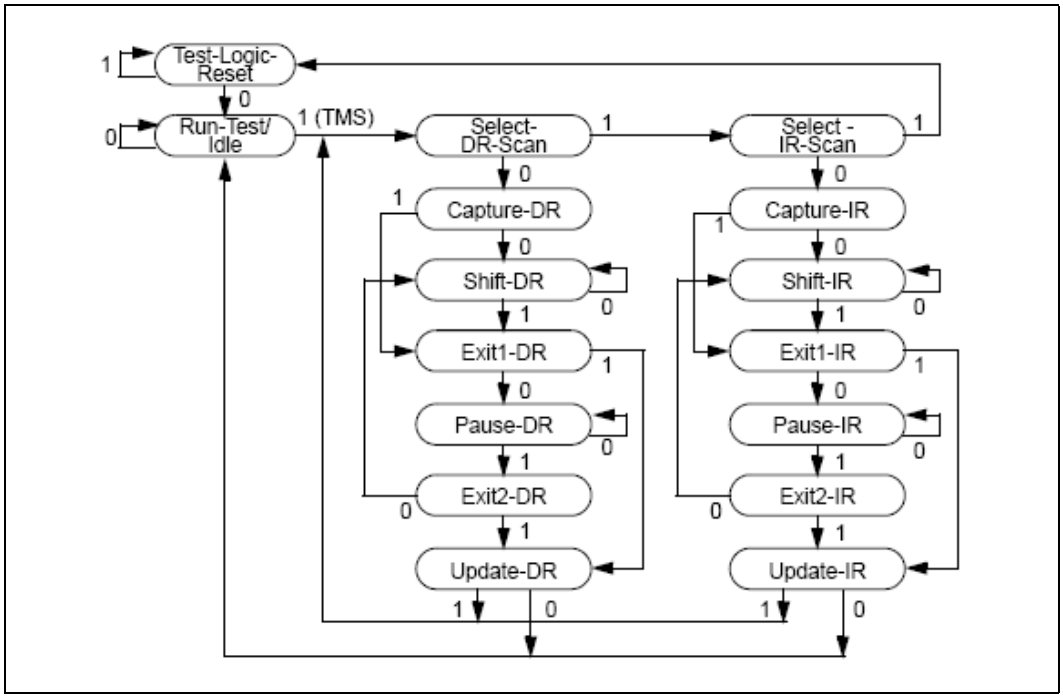


The TAP logic consists of a finite state machine controller, a serially-accessible instruction register, instruction decode logic and data registers. The set of data registers includes those described in the 1149.1 standard (the bypass register, device ID register, and so forth) plus the Intel® 5100 MCH Chipset-specific additions.

7.1.2 Accessing TAP Logic

The TAP is accessed through an 1149.1-compliant TAP controller finite state machine, which is illustrated in Figure 55, "Simplified TAP Controller Block Diagram." 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 56. 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.

7.1.3 Reset Behavior of 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 138, "TAP Reset Actions" are performed. The TAP is completely disabled upon reset (i.e., by resetting the TAP, the device will function as though the TAP did not exist).

Table 138. 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	Tristated	—

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 five 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.

7.1.4 Clocking TAP

There is no minimum frequency at which the Intel® 5100 MCH 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.

7.1.5 Accessing Instruction Register

Figure 57, "TAP Instruction Register" 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 57. TAP Instruction Register

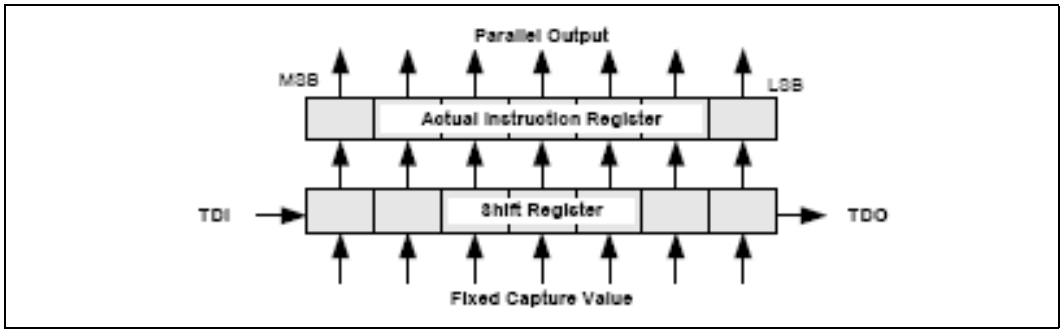


Figure 58, "TAP Instruction Register Operation" shows the operation of the instruction register during the Capture-IR, Shift-IR and Update-IR states. Shaded areas indicate the bits that are updated. In Capture-IR, the shift register portion of the instruction register is loaded in parallel with the fixed 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 58. TAP Instruction Register Operation

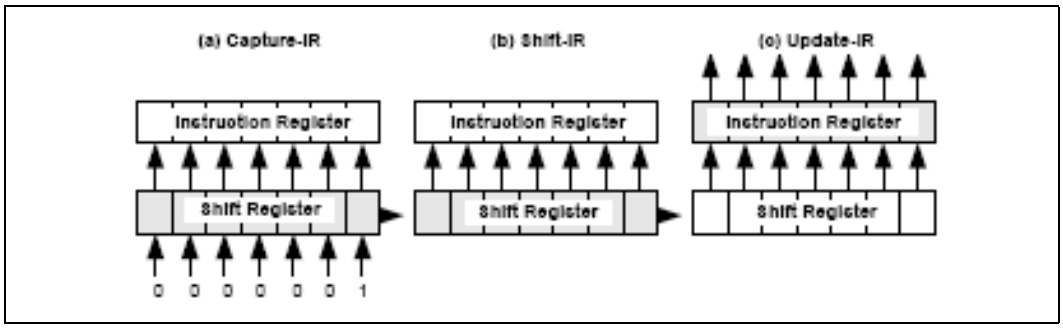
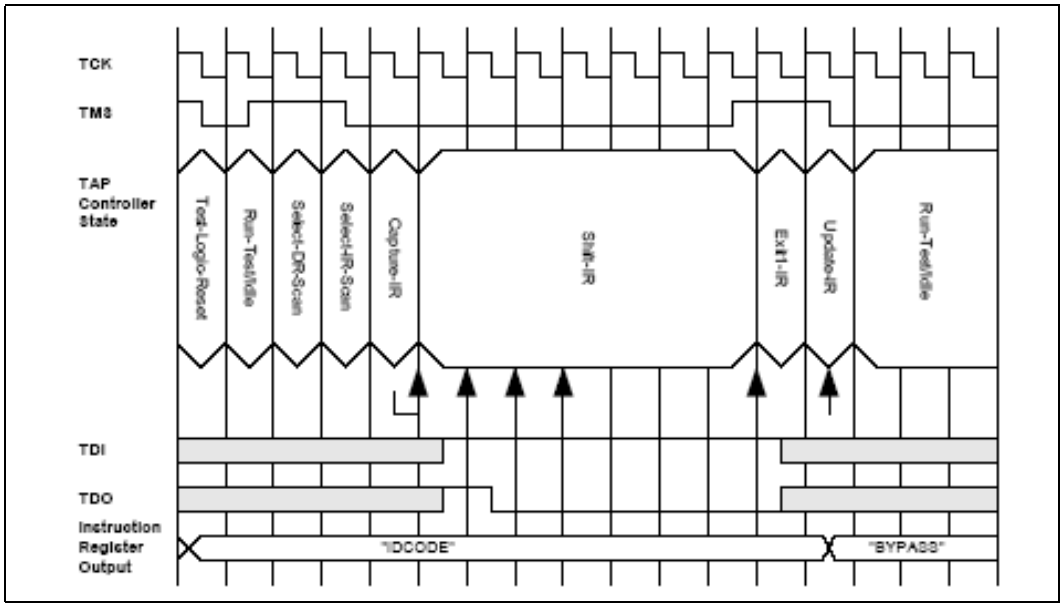


Figure 59, "TAP Instruction Register Access" illustrates the timing when loading the BYPASS instruction (opcode 1111111b) into the TAP instruction register. Vertical arrows in Figure 59, "TAP Instruction Register Access" show the specific clock edges on which the Capture-IR, Shift-IR and Update-IR actions 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 59. TAP Instruction Register Access



7.1.6 Accessing Data Registers

The test data registers in the Intel® 5100 MCH Chipset components are architected in the same way as the instruction register, with components (i.e., 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 Figure 56, "TAP Controller State Machine." 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.

7.1.7 Public TAP Instructions

Table 139, "Public TAP Instructions" contains descriptions of the encoding and operation of the public TAP instructions. There are four 1149.1-defined instructions implemented in Intel® 5100 MCH Chipsets. 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.

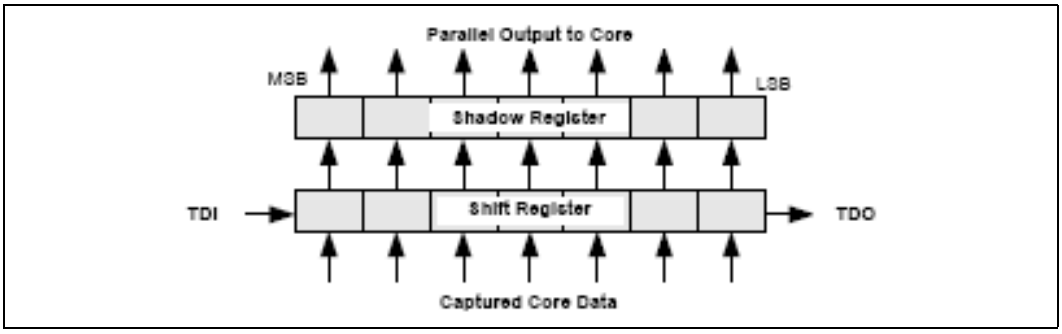
Table 139. Public TAP Instructions

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.

7.1.8 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 60, "TAP Data Register" which is the same as the instruction register.

Figure 60. TAP Data Register



7.1.9 Public Data Register Control

Table 140, "Actions of Public TAP Instructions During Various TAP States" 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 140. 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

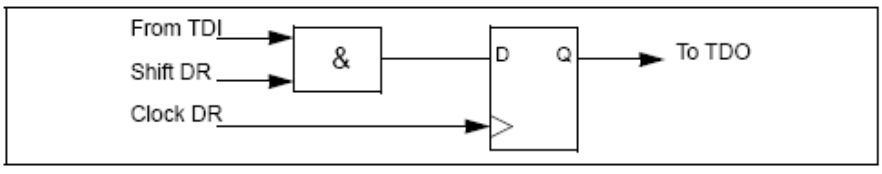
7.1.10 Bypass Register

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 61, "Bypass Register Implementation" for an implementation example.

Table 141. Bypass Register Definition

JTAG encode: 1111111			
Bit	**Attr**	**Default**	**Description**
1	RW	0	Bypass: a 1-bit register used to bypass the chip for board testing

Figure 61. Bypass Register Implementation



7.1.11 Device ID Register

This register contains the device identification code in the format shown in Table 142, "Intel® 5100 Memory Controller Hub Chipset Device ID Codes." 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 three

fields. The Product Segment field identifies if the component is intended for CPU, laptop, desktop, server, etc. Product Type further defines the component within a segment by stating it to be a CPU, memory, chipset, etc. 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 142. Intel® 5100 Memory Controller Hub Chipset Device ID Codes

Device	Version	Component Identification Fields			Manufacturing ID	"1"	Entire Code (hex)
		Product Segment	Product Type	Component Number			
Bits per field	4	6	5	5	11	1	32
Intel® 5100 MCH Chipset - A0	0000b	0001 00b	01 000b	0 1011b	0000 0001 001b	1b	0110B013h
Intel® 5100 MCH Chipset - B0	0100b	000100b	01000b	01011b	00000001001b	1b	4110B013h

7.1.11.1 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 two bits are the stepping number: 00 A-step; 01 B-step, 10 C-step, and 11 D-step. The least significant two bits is the revision within a stepping. Note: The SRID determines the Version number. The SRID is the combination of the two least significant bits of the Major Revision and the two least significant bits of the Minor Revision (see Section 3.8.1.3, "RID - Revision Identification Register").
27:22	R	000 100	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 etc.
21:17	R	01 000	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 etc.
16:12	R	Listed in next column	Component Number: Sequential listing based on request to database. Intel® 5100 MCH Chipset: 01011b
11:1	R	00000001001	Manufacturing ID: This number is assigned to Intel.
0	R	1	'1'

7.1.12 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, & 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 (i.e., 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.

7.2 Extended Debug Port (XDP)

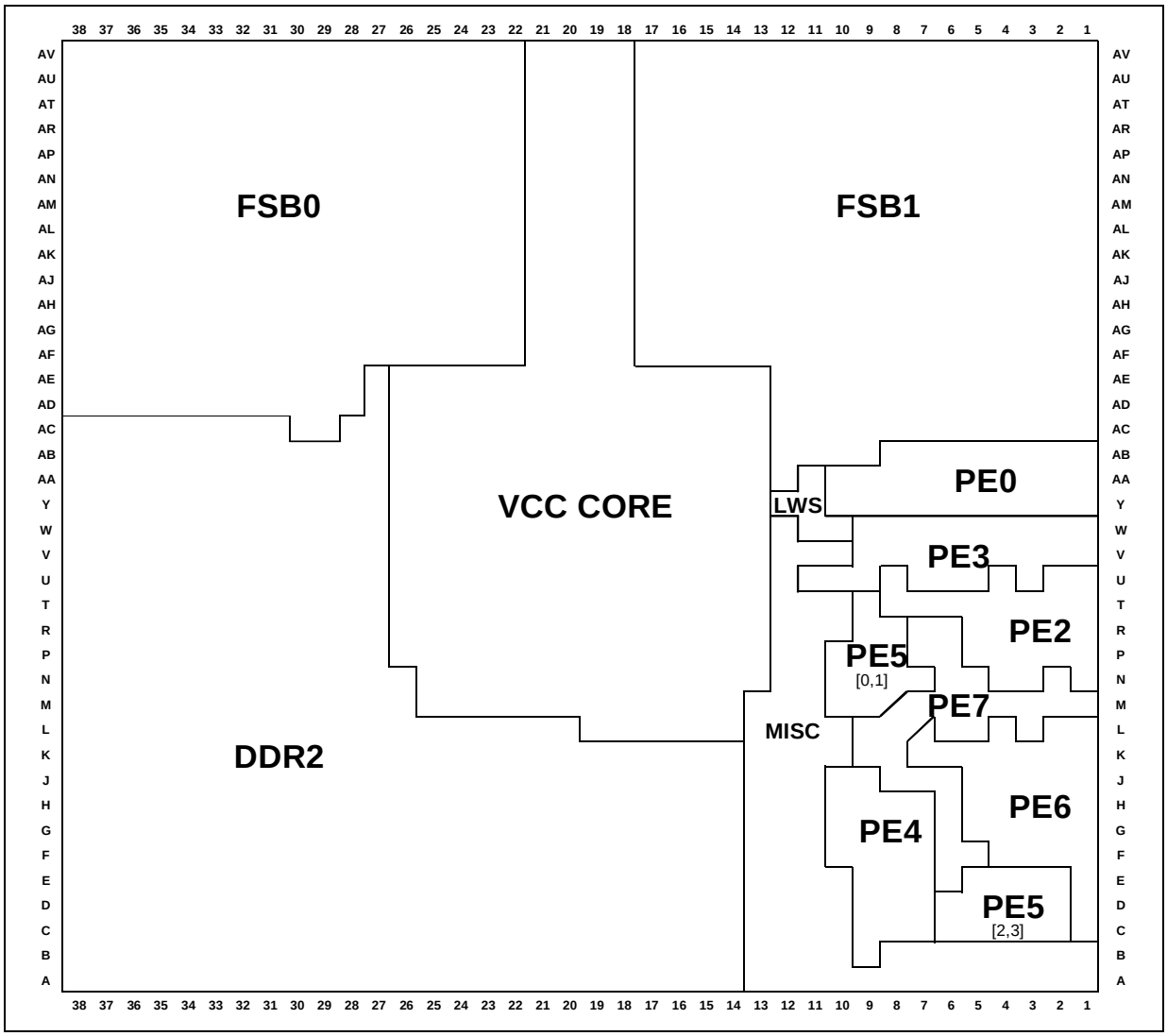
The Extended Debug Port is covered in the *Debug Port Design Guide for Intel® 5000 Series Chipset Based Platforms*.

8.0 Ballout and Package Information

8.1 Intel® 5100 Memory Controller Hub Chipset Ballout

The following section presents preliminary ballout information for the Intel® 5100 MCH Chipset. This ballout is subject to change and is to be used for informational purposes only. See Section 1.2, "Related Documents and Materials" for electronic pin list and ball map.

Figure 62. Intel® 5100 Memory Controller Hub Chipset Quadrant Map



8.2 Intel® 5100 Memory Controller Hub Chipset Ballout

Figure 63. Intel® 5100 Memory Controller Hub Chipset Ballout Left Side (Top View)

	38	37	36	35	34	33	32	31	30	29	28	27	26	
AV			VSS	VSS	FSB0 DEFER#	FSB0 HITM#	VSS	FSB0 RS[0]#	FSB0 BNR#	VSS	FSB0 A[17]#	RSVD	VSS	AV
AU		VSS	FSB0 D[21]#	FSB SLW CRES	FSB0 BPRI#	VSS	FSB0 HIT#	FSB0 RS[2]#	VSS	FSB0 ADS#	FSB0 A[21]#	VSS	FSB0 A[25]#	AU
AT	VSS	FSB0 D[22]#	FSB0 D[19]#	FSB CRES	VSS	FSB0 DP[1]#	FSB0 TRDY#	VSS	FSB0 LOCK#	FSB0 DRDY#	VSS	FSB0 A[23]#	FSB0 A[24]#	AT
AR	FSB0 DSTBP[1]#	FSB0 DSTBN[1]#	VSS	VSS	FSB ODT CRES	RSVD	VSS	FSB0 DP[0]#	FSB0 DBSY#	VSS	FSB0 BPM[4]#	FSB0 A[19]#	VSS	AR
AP	FSB0 D[27]#	FSB0 DBI[1]#	FSB0 D[23]#	FSB0 D[18]#	FSB0 D[20]#	VSS	VSS	FSB0 DP[2]#	VSS	FSB0 BPM[5]#	VSS	VSS	FSB0 A[7]#	AP
AN	VSS	VSS	FSB0 D[26]#	FSB0 D[24]#	VSS	FSB0 D[16]#	FSB0 D[5]#	VSS	FSB0 RESET#	FSB0 DP[3]#	VSS	FSB0 RSP#	FSB0 A[6]#	AN
AM	FSB0 D[28]#	FSB0 D[31]#	VSS	VSS	FSB0 D[29]#	FSB0 D[12]#	VSS	VSS	FSB0 VREF	VSS	FSB0 BREQ0#	FSB0 VREF	VSS	AM
AL	FSB0 D[50]#	FSB0 D[52]#	FSB0 D[51]#	FSB0 D[25]#	FSB0 D[17]#	VSS	FSB0 DBI[0]#	FSB0 D[7]#	VSS	FSB0 D[1]#	FSB0 RS[1]#	VSS	FSB0 REQ[0]#	AL
AK	VSS	VSS	FSB0 D[53]#	FSB0 D[30]#	VSS	FSB0 DSTBP[0]#	FSB0 DSTBN[0]#	VSS	FSB0 D[6]#	FSB0 D[3]#	VSS	FSB0 BINIT#	FSB0 AP[1]#	AK
AJ	FSB0 D[56]#	FSB0 D[55]#	VSS	VSS	FSB0 D[49]#	FSB0 D[15]#	VSS	FSB0 D[8]#	FSB0 D[10]#	VSS	FSB0 D[0]#	FSB0 MCERR#	VSS	AJ
AH	FSB0 D[57]#	FSB0 DBI[3]#	FSB0 D[61]#	FSB0 DSTBP[3]#	FSB0 DSTBN[3]#	VSS	FSB0 D[9]#	FSB0 D[11]#	VSS	FSB0 D[4]#	FSB0 D[2]#	VSS	FSB0 AP[0]#	AH
AG	VSS	VSS	FSB0 D[60]#	FSB0 D[54]#	VSS	FSB0 D[32]#	FSB0 D[36]#	VSS	FSB0 D[33]#	FSB0 D[13]#	VSS	FSB0 D[14]#	FSB0 BREQ1#	AG
AF	FSB0 D[59]#	FSB0 D[48]#	VSS	VSS	FSB0 VREF	FSB0 D[37]#	VSS	FSB0 D[35]#	FSB0 DBI[2]#	VSS	FSB0 D[45]#	VSS	VSS	AF
AE	FSB0 D[58]#	FSB0 D[63]#	FSB0 D[62]#	VSS	FSB0 D[34]#	VSS	FSB0 D[39]#	FSB0 D[40]#	VSS	FSB0 D[43]#	FSB0 D[47]#	VSS	VTT	AE
AD	VSS	VSS	VSS	FSB0 D[38]#	VSS	FSB0 DSTBP[2]#	FSB0 DSTBN[2]#	VSS	FSB0 D[41]#	FSB0 D[46]#	VSS	VSS	VCC	AD
AC	CH1_DQSP[7]	CH1_DQSN[7]	VCC DDR	CH1_DQ[63]	CH1_DQ[58]	CH1_DQ[59]	VSS	FSB0 D[44]#	FSB0 D[42]#	CH1_DQ[48]	VSS	VSS	VCC	AC
AB	CH1_DQSP[16]	CH1_DQSN[16]	CH1_DQ[56]	CH1_DQ[51]	VSS	CH1_DQ[57]	CH1_DQ[62]	CH1_DQSP[6]	CH1_DQSN[6]	CH1_DQ[53]	CH1_DQ[52]	VSS	VSS	AB
AA	CH1_DQ[61]	CH1_DQ[60]	VSS	CH1_DQ[54]	CH1_DQ[55]	CH1_DQ[50]	CH1_DQSN[15]	CH1_DQSP[15]	CH1_DQ[49]	VSS	CH0_DQ[62]	CH0_DQ[59]	VSS	AA
Y	CH0_DQ[61]	VSS	CH0_DQ[50]	CH0_DQ[56]	CH0_DQSP[16]	CH0_DQSN[16]	VSS	CH0_DQ[57]	CH0_DQSN[7]	CH0_DQSP[7]	CH0_DQ[63]	CH0_DQ[58]	VSS	Y
W	VSS	CH0_DQSP[6]	CH0_DQSN[6]	CH0_DQ[51]	CH0_DQ[60]	VSS	CH0_DQ[55]	VSS	CH0_DQ[44]	CH1_DQ[35]	VSS	CH1_DQ[34]	VSS	W
V	CH0_DQSN[15]	CH0_DQSP[15]	CH0_DQ[49]	CH0_DQ[53]	VSS	CH0_DQ[54]	CH1_DQ[43]	CH1_DQ[42]	VCC DDR	CH1_DQ[33]	CH1_DQ[32]	CH1_CB[4]	VSS	V
U	CH0_DQ[48]	CH0_DQ[52]	VCC DDR	VSS	CH0_DQ[45]	CH1_DQSP[5]	CH1_DQSN[5]	VSS	VSS	CH1_DQSP[8]	CH1_DQSN[8]	CH1_CB[5]	VSS	U
T	CH0_DQ[43]	CH0_DQ[42]	VSS	CH0_DQ[47]	CH1_DQ[46]	CH1_DQ[47]	VCC DDR	VSS	CH1_CB[2]	CH1_CB[7]	CH1_DQSP[17]	CH1_CB[0]	VSS	T
R	CH0_DQ[46]	VSS	CH0_DQSN[5]	CH0_DQSP[5]	CH1_DQ[40]	CH1_DQ[45]	VSS	CH1_DQSP[4]	CH1_DQ[36]	VSS	CH1_DQSN[17]	CH1_CB[1]	VSS	R
P	VSS	CH0_DQSN[14]	CH0_DQSP[14]	CH0_DQ[40]	CH0_DQ[34]	CH1_DQ[39]	CH1_DQSP[13]	CH1_DQSN[4]	CH1_DQ[37]	CH1_ODT[0]	CH1_CB[6]	CH1_ODT[3]	VSS	P
N	CH0_DQ[41]	CH1_DQSN[14]	CH1_DQSP[14]	VSS	CH0_DQSN[13]	CH1_DQ[38]	CH1_DQSN[13]	VSS	VCC DDR	CH1_CB[3]	CH1_ODT[1]	CH1_ODT[2]	CH0_A[14]	N
M	CH1_DQ[41]	CH0_DQ[35]	VSS	CH0_DQ[39]	CH0_DQSP[13]	CH0_DQ[33]	CH1_BA[1]	CH1_CAS#	CH1_RAS#	CH1_A[13]	VSS	CH0_BA[2]	CH1_CS[3]#	M
L	CH1_DQ[44]	CH0_DQSP[4]	VCC DDR	VCC DDR	CH1_CRES2	CH0_DQ[32]	VCC DDR	VSS	CH1_WE#	CH1_CS[1]#	CH0_A[15]/CH0_ODT[4]	CH0_A[9]	CH0_CB[3]	L
K	CH0_DQ[38]	CH0_DQSN[4]	CH1_CRES RET	CH0_DQ[37]	CH1_CRES1	CH0_DQ[36]	VSS	CH1_BA[0]	CH1_CS[2]#	VSS	CH0_A[7]	CH0_A[8]	CH0_DQSP[8]	K
J	VSS	CH1_DCLKP[1]	CH1_DCLKN[1]	VSS	CH1_A[14]	VSS	CH1_A[0]	CH1_A[10]	CH0_CKE[0]	CH0_A[11]	CH0_CB[6]	VSS	CH0_CB[1]	J
H	CH1_DCLKP[3]/CH1_CS[4]#	CH1_DCLKN[3]/CH1_CS[5]#	CH1_SLEW CRES	CH1_DRV CRES	VSS	CH1_CKE[2]	CH1_CKE[3]/CH1_ODT[5]	CH1_CS[0]#	CH0_A[12]	CH0_DQSN[17]	CH0_DQSP[17]	CH0_CB[0]	CH0_DRV CRES	H
G	CH1_DCLKN[0]	CH1_DCLKP[2]	CH1_DCLKN[2]	VSS	CH1_A[15]/CH1_ODT[4]	CH1_CKE[1]	CH0_CKE[3]/CH0_ODT[5]	CH0_CKE[2]	VSS	VCC DDR	CH0_SLEW CRES	CH0_DCLKN[1]	CH0_DCLKP[1]	G
F	CH1_DCLKP[0]	CH1_A[3]	VSS	CH1_A[6]	CH1_BA[2]	CH1_CKE[0]	CH0_A[6]	VSS	CH0_A[3]	CH0_DCLKN[3]/CH0_CS[5]#	CH0_DCLKN[0]	CH0_DCLKN[2]	CH0_DCLKP[2]	F
E	CH1_A[1]	VSS	CH1_A[9]	CH0_ODT[1]	CH0_CKE[1]	VCC DDR	VSS	CH0_A[5]	CH0_CRES RET	CH0_DCLKP[3]/CH1_CS[4]#	CH0_DCLKP[0]	VSS	VSS	E
D	CH1_A[2]	CH1_A[4]	CH1_A[11]	CH1_A[12]	CH0_CS[0]#	VSS	CH0_A[4]	CH0_A[2]	CH0_A[1]	VCC DDR	VSS	CH1_DQ[27]	CH1_DQ[26]	D
C	VSS	CH1_A[5]	CH1_A[7]	CH0_ODT[2]	VSS	CH0_CS[1]#	CH0_WE#	CH0_BA[0]	CH0_A[0]	VSS	CH1_DQSP[3]	CH1_DQSN[3]	CH1_DQ[29]	C
B		VSS	CH1_A[8]	VSS	CH0_CS[3]#	CH0_CS[2]#	CH0_CAS#	CH0_BA[1]	VSS	CH1_DQ[31]	CH1_DQSP[12]	CH1_DQ[25]	CH1_DQ[28]	B
A			VSS	CH0_ODT[3]	CH0_ODT[0]	CH0_A[13]	CH0_RAS#	VSS	CH0_A[10]	CH1_DQ[30]	CH1_DQSN[12]	CH1_DQ[24]	VCC DDR	A
	38	37	36	35	34	33	32	31	30	29	28	27	26	

Figure 64. Intel® 5100 Memory Controller Hub Chipset Ballout Center (Top View)

	25	24	23	22	21	20	19	18	17	16	15	14	13	
AV	FSB0 A[28]#	FSB0 A[31]#	VSS	FSB0 A[35]#	VTT	VTT	VTT	VTT	VSS	VSS	VSS	VSS	FSB SLW CTRL	AV
AU	FSB0 A[22]#	VSS	FSB0 A[29]#	FSB0 A[34]#	VTT	VTT	VTT	VTT	FSB VCCA	CORE VSSA	VSS	48GB_ Mode	VSS	AU
AT	VSS	FSB0 A[27]#	FSB0 A[30]#	VSS	VTT	VTT	VTT	VTT	CORE VCCA	VSS	VSS	FSB1 VREF	VSS	AT
AR	FSB0 A[26]#	FSB0 A[20]#	VSS	FSB0 A[33]#	VTT	VTT	VTT	VTT	VSS	VSS	VSS	VSS	FSB1 D[45]#	AR
AP	FSB0 A[18]#	VSS	FSB0 ADSTB [1]#	FSB0 A[32]#	VTT	VTT	VTT	VTT	CORE CLKN	FSB1 D[55]#	VSS	FSB1 D[47]#	FSB1 D[43]#	AP
AN	VSS	FSB0 A[4]#	RSVD	VSS	VTT	VTT	VTT	VTT	CORE CLKP	VSS	FSB1 D[53]#	FSB1 D[50]#	VSS	AN
AM	FSB0 A[5]#	RSVD	VSS	FSB0 A[9]#	VTT	VTT	VTT	VTT	VSS	FSB1 D[49]#	FSB1 D[51]#	VSS	FSB1 D[28]#	AM
AL	FSB0 A[3]#	VSS	FSB0 ADSTB [0]#	FSB0 A[12]#	VTT	VTT	VTT	VTT	VCC	FSB1 D[56]#	VSS	FSB1 D[52]#	FSB1 D[30]#	AL
AK	VSS	FSB0 REQ [2]#	RSVD	VSS	VTT	VTT	VTT	VTT	RSVD	VSS	FSB1 DSTBP [3]#	FSB1 DSTBN [3]#	VSS	AK
AJ	FSB0 REQ [4]#	FSB0 REQ [3]#	VSS	FSB0 A[11]#	VTT	VTT	VTT	VTT	VSS	FSB1 D[61]#	FSB1 D[60]#	VSS	FSB1 D[57]#	AJ
AH	FSB0 REQ [1]#	VSS	FSB0 A[8]#	FSB0 A[13]#	VTT	VTT	VTT	VTT	ASYNC RFSH	FSB1 D[54]#	VSS	FSB1 D[48]#	FSB1 DBI[3]#	AH
AG	VSS	FSB0 A[10]#	FSB0 A[15]#	VSS	VTT	VTT	VTT	VTT	VSS	VSS	FSB1 D[59]#	FSB1 D[62]#	VSS	AG
AF	FSB0 A[14]#	VSS	VSS	FSB0 A[16]#	VTT	VTT	VTT	VTT	VSS	FSB1 D[63]#	FSB1 D[58]#	VSS	FSB1 DP[3]#	AF
AE	VTT	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	VTT	AD
AC	VCC	VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCC	VTT	VTT	AC
AB	VCC DDR	VCC	VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCC	VCC	AB
AA	VCC DDR	VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCC	AA
Y	VCC DDR	VCC	VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCCPE	VCCPE	Y
W	VCC DDR	VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCC	VCCPE	VCCPE	W
V	VCC DDR	VCC	VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCCPE	VCCPE	V
U	VCC DDR	VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCC	VCCPE	VCCPE	U
T	VCC DDR	VCC	VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCCPE	VCCPE	T
R	VCC DDR	VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCC	VCCPE	VCCPE	R
P	VCC DDR	VCC	VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCC	VSS	VCCPE	VCCPE	P
N	VCC DDR	VCC DDR	VCC DDR	VCC DDR	VCC DDR	VCC DDR	VCC	VSS	VCC	VSS	VCCPE	VCCPE	VCCPE	N
M	VSS	VSS	VSS	VSS	VSS	VSS	VSS	VCC	VCC	VCC	VCCPE	VCCPE	GPIO SMB CLK	M
L	CH0_ CB[2]	CH0_ CB[7]	CH1_ DQ[16]	CH1_ DQSN [11]	CH1_ DQSP [11]	CH0_ DQ[27]	VCC	VCC	VCC	VCC	VCCPE	VCCPE	ERR[1]#	L
K	CH0_ DQSN [8]	CH0_ CB[5]	CH1_ DQSP [2]	VSS	CH1_ DQ[20]	VSS	CH1_ DQSP [9]	VSS	CH1_ DQ[0]	VSS	VSS	VSS	XDP ODT CRES	K
J	CH0_ CB[4]	VSS	CH1_ DQSN [2]	CH1_ DQ[17]	CH1_ DQ [21]	CH0_ DQ[26]	CH1_ DQSN [9]	CH1_ DQ[1]	CH1_ DQ[5]	CH1_ DQ[4]	CH0_ DQSP [10]	CH0_ DQ[14]	VSS	J
H	VSS	VSS	VSS	CH0_ DQ[30]	CH0_ DQ[31]	CH0_ DQ[29]	VSS	CH0_ DQ[11]	CH0_ DQ[10]	CH0_ DQ[15]	CH0_ DQSN [10]	VSS	XDP SLW CRES	H
G	VCC DDR	CH1_ DQ[23]	CH0_ DQSN [3]	CH0_ DQSP [3]	VCC DDR	VSS	CH1_ DQSP [0]	CH1_ DQSN [0]	CH0_ DQSP [1]	CH0_ DQSN [1]	VSS	CH0_ DQ[12]	XDP COM CRES	G
F	CH0_ CRES2	CH1_ DQ[22]	CH0_ DQSP [12]	CH0_ DQSN [12]	VSS	CH1_ DQSP [1]	CH1_ DQSN [1]	CH1_ DQ[2]	CH1_ DQ[6]	VSS	CH0_ DQ[9]	CH0_ DQ[13]	V3REF	F
E	CH0_ CRES1	CH0_ DQ[24]	CH0_ DQ[25]	VSS	CH0_ DQ[28]	CH1_ DQ[14]	CH1_ DQ[3]	CH1_ DQ[9]	VSS	CH0_ DQ[8]	CH0_ DQSP [9]	CH0_ DQ[4]	VSS	E
D	CH1_ DQ[19]	CH0_ DQ[19]	VSS	CH0_ DQ[18]	CH0_ DQ[21]	CH1_ DQ[15]	VCC DDR	VSS	CH1_ DQ[7]	CH0_ DQ[2]	CH0_ DQSN [9]	VSS	XDPD [12]#	D
C	CH1_ DQ[18]	VSS	CH0_ DQSN [2]	CH0_ DQSP [2]	CH0_ DQ[20]	CH1_ DQ[10]	VSS	CH1_ DQ[8]	VSS	CH0_ DQ[6]	VSS	CH0_ DQ[5]	XDPD [15]#	C
B	VSS	CH0_ DQ[23]	CH0_ DQSP [11]	CH0_ DQ[17]	VCC DDR	VSS	CH1_ DQSN [10]	CH1_ DQ[12]	CH0_ DQ[3]	VSS	CH0_ DQSP [0]	CH0_ DQ[0]	XDPD [14]#	B
A	VSS	CH0_ DQ[22]	CH0_ DQSN [11]	CH0_ DQ[16]	VSS	CH1_ DQ[11]	CH1_ DQSP [10]	CH1_ DQ[13]	VSS	CH0_ DQ[7]	CH0_ DQSN [0]	CH0_ DQ[1]	VSS	A
	25	24	23	22	21	20	19	18	17	16	15	14	13	

Figure 65. Intel® 5100 Memory Controller Hub Chipset Ballout Right Side (Top View)

	12	11	10	9	8	7	6	5	4	3	2	1	
AV	FSB1 D[40]#	VSS	FSB1 D[39]#	FSB1 D[37]#	VSS	FSB1 D[32]#	FSB1 D[9]#	VSS	FSB1 D[10]#	VSS			AV
AU	VSS	FSB1 DBI[2]#	FSB1 D[38]#	VSS	FSB1 D[33]#	FSB1 D[13]#	VSS	FSB1 D[15]#	FSB1 DSTBP[0]#	FSB1 DSTBN[0]#	VSS		AU
AT	FSB1 D[42]#	FSB1 D[41]#	VSS	FSB1 D[34]#	FSB1 D[36]#	VSS	FSB1 D[11]#	FSB1 D[8]#	VSS	FSB1 D[7]#	FSB1 D[6]#	VSS	AT
AR	FSB1 D[46]#	VSS	FSB1 DSTBP[2]#	FSB1 D[35]#	VSS	FSB1 D[14]#	FSB1 D[12]#	VSS	FSB1 D[5]#	FSB1 D[4]#	VSS	FSB1 D[3]#	AR
AP	VSS	FSB1 D[44]#	FSB1 DSTBN[2]#	VSS	FSB1 D[22]#	FSB1 DBI[0]#	VSS	FSB1 D[1]#	FSB1 D[2]#	VSS	FSB1 ADS#	FSB1 D[0]#	AP
AN	FSB1 D[25]#	FSB1 D[27]#	VSS	FSB1 D[24]#	FSB1 D[23]#	VSS	FSB1 D[18]#	FSB1 VREF	VSS	FSB1 BPM[5]#	FSB1 BPM[4]#	VSS	AN
AM	FSB1 D[26]#	VSS	FSB1 DSTBP[1]#	FSB1 D[21]#	VSS	VSS	FSB1 D[20]#	VSS	FSB1 DBSY#	FSB1 DRDY#	VSS	FSB1 BREQ1#	AM
AL	VSS	FSB1 D[29]#	FSB1 DSTBN[1]#	VSS	FSB1 D[17]#	FSB1 D[16]#	VSS	FSB1 RS[2]#	FSB1 LOCK#	VSS	FSB1 BREQ0#	FSB1 RS[1]#	AL
AK	FSB1 D[31]#	FSB1 DBI[1]#	VSS	FSB1 D[19]#	FSB1 HIT#	VSS	FSB1 TRDY#	FSB1 RS[0]#	VSS	FSB1 BNR#	FSB1 RSP#	VSS	AK
AJ	FSB1 DP[1]#	VSS	FSB1 BPRI#	FSB1 DEFER#	VSS	FSB1 HITM#	FSB1 REQ[2]#	VSS	FSB1 BINIT#	FSB1 A[26]#	VSS	RSVD	AJ
AH	VSS	FSB1 MCERR#	VSS	VSS	FSB1 A[6]#	RSVD	VSS	FSB1 A[19]#	FSB1 VREF	VSS	FSB1 A[24]#	FSB1 A[25]#	AH
AG	FSB1 AP[0]#	FSB1 DP[0]#	FSB1 AP[1]#	FSB1 REQ[0]#	FSB1 A[7]#	VSS	VSS	FSB1 A[18]#	VSS	FSB1 ADSTB[1]#	RSVD	VSS	AG
AF	FSB1 DP[2]#	VSS	VSS	FSB1 REQ[3]#	VSS	FSB1 A[8]#	FSB1 A[11]#	VSS	FSB1 A[17]#	FSB1 A[28]#	VSS	FSB1 A[27]#	AF
AE	VSS	FSB1 RESET#	FSB1 REQ[4]#	VSS	RSVD	FSB1 A[9]#	VSS	FSB1 A[21]#	FSB1 A[30]#	VSS	FSB1 A[31]#	FSB1 A[32]#	AE
AD	FSB1 A[3]#	FSB1 REQ[1]#	VSS	FSB1 A[4]#	FSB1 A[12]#	VSS	FSB1 A[20]#	FSB1 A[23]#	VSS	FSB1 A[29]#	FSB1 A[33]#	VSS	AD
AC	FSB1 A[5]#	VSS	FSB1 ADSTB[0]#	FSB1 A[13]#	VSS	FSB1 A[15]#	FSB1 A[22]#	VSS	FSB1 A[34]#	FSB1 A[35]#	VSS	PSEL[0]	AC
AB	VSS	FSB1 A[10]#	FSB1 A[14]#	VSS	PE0 RP[2]	PE0 RN[2]	VSS	PE0 TN[2]	PE0 TP[2]	VSS	PSEL[1]	PSEL[2]	AB
AA	FSB1 A[16]#	PE WIDTH[0]	VSS	PE0 TN[3]	PE0 TP[3]	VSS	PE0 RN[3]	PE0 RP[3]	VSS	PE0 TP[1]	PE0 TN[1]	VSS	AA
Y	PE WIDTH[1]	VSS	PE0 RP[0]	PE0 RN[0]	VSS	PE0 TP[0]	PE0 TN[0]	VSS	PE0 RP[1]	PE0 RN[1]	VSS	VSS	Y
W	VCCPE	PE WIDTH[2]	PE WIDTH[3]	VSS	PE3 RN[2]	PE3 RP[2]	VCCPE	PE3 TP[1]	PE3 TN[1]	VSS	PE3 RN[0]	VSS	W
V	VSS	VSS	VSS	PE3 TN[3]	PE3 TP[3]	VSS	PE3 RN[1]	PE3 RP[1]	VSS	PE3 TN[0]	PE3 RP[0]	VSS	V
U	VSS	VSS	PE3 RN[3]	PE3 RP[3]	VCCPE	PE3 TN[2]	PE3 TP[2]	VSS	PE2 TP[1]	PE3 TP[0]	VCCPE	PE2 RN[3]	U
T	VSS	VSS	VSS	VSS	PE2 TP[0]	PE2 TN[0]	VSS	PE2 RP[0]	PE2 TN[1]	VSS	PE2 TN[3]	PE2 RP[3]	T
R	PEI COMPI	PE VCCBG	VCCPE	PE5 RP[0]	PE5 RN[0]	VSS	VSS	PE2 RN[0]	VCCPE	PE2 RN[2]	PE2 TP[3]	VSS	R
P	PER COMPO	VSS	PE5 TP[0]	PE5 TN[0]	VSS	PE7 RP[1]	PE7 RN[1]	VSS	PE2 RN[1]	PE2 RP[2]	VSS	PE2 TN[2]	P
N	VCCPE	PE VSSBG	VCCPE	VSS	PE5 RP[1]	PE5 RN[1]	VCCPE	PE7 RN[2]	PE2 RP[1]	VSS	PE7 RN[3]	PE2 TP[2]	N
M	VSS	VCCPE	VSS	PE5 TP[1]	PE5 TN[1]	VSS	PE7 TN[2]	PE7 RP[2]	VSS	PE7 TN[3]	PE7 RP[3]	VSS	M
L	CFG SMB CLK	VSS	SPD0 SMB DATA	VSS	VCCPE	PE6 TN[2]	PE7 TP[2]	VSS	PE6 RN[3]	PE7 TP[3]	VCCPE	PE VSSA	L
K	VSS	TDIO CATHODE	ERR[0]#	VSS	PE7 TN[1]	PE6 TP[2]	VSS	PE6 TN[3]	PE6 RP[3]	VSS	PE CLKN	PE VCCA	K
J	ERR[2]#	TDIO ANODE	VCCPE	PE4 TP[0]	PE7 TP[1]	VSS	PE7 RN[0]	PE6 TP[3]	VSS	VSS	PE CLKP	VSS	J
H	GPIO SMB DATA	VSS	PE4 RP[0]	PE4 TN[0]	VSS	PE4 RN[3]	PE7 RP[0]	VSS	PE6 RN[2]	PE6 RP[2]	VSS	VSS	H
G	PWR GOOD	CFG SMB DATA	PE4 RN[0]	VSS	PE4 RN[2]	PE4 RP[3]	VSS	PE6 TP[0]	PE6 TN[0]	VSS	PE6 TP[1]	PE6 TN[1]	G
F	XDP RDY#	XDP D[4]#	VSS	PE4 TP[1]	PE4 RP[2]	VSS	PE7 TP[0]	PE7 TN[0]	VSS	PE6 RP[0]	PE6 RN[0]	VSS	F
E	XDP D[6]#	XDP D[5]#	XDP D[0]#	PE4 TN[1]	VSS	PE4 TN[3]	VCCPE	VSS	PE5 TP[3]	PE5 TN[3]	VSS	PE6 RN[1]	E
D	VSS	XDP DSTBN#	XDP D[3]#	VSS	PE4 TP[2]	PE4 TP[3]	VSS	PE5 RP[2]	PE5 RN[2]	VSS	VSS	PE6 RP[1]	D
C	XDP D[9]#	XDP DSTBP#	VSS	PE4 RP[1]	PE4 TN[2]	VSS	PE5 TP[2]	PE5 TN[2]	VSS	PE5 RP[3]	PE5 RN[3]	VSS	C
B	XDP D[10]#	VSS	XDP D[7]#	PE4 RN[1]	VSS	TRST#	RESETI#	VSS	TDO	TDI	VSS		B
A	XDP D[13]#	XDP D[11]#	XDP D[8]#	VSS	XDP D[1]#	XDP D[2]#	TMS	SPD0 SMB CLK	TCK	VSS			A
	12	11	10	9	8	7	6	5	4	3	2	1	

Table 143. Intel® 5100 Memory Controller Hub Chipset Signals By Ball (Sheet 1 of 16)

Ball No.	Signal Name	Buffer Type	Direction	Ball No.	Signal Name	Buffer Type	Direction
A3	VSS	Power/Other		B13	XDPD[14]#	XDP	I/O
A4	TCK	JTAG	I	B14	CH0_DQ[0]	DDR2	I/O
A5	SPD0SMBCLK	SMB	I/O	B15	CH0_DQSP[0]	DDR2	I/O
A6	TMS	JTAG	I	B16	VSS	Power/Other	
A7	XDPD[2]#	XDP	I/O	B17	CH0_DQ[3]	DDR2	I/O
A8	XDPD[1]#	XDP	I/O	B18	CH1_DQ[12]	DDR2	I/O
A9	VSS	Power/Other		B19	CH1_DQSN[10]	DDR2	I/O
A10	XDPD[8]#	XDP	I/O	B20	VSS	Power/Other	
A11	XDPD[11]#	XDP	I/O	B21	VCCDDR	Power/Other	
A12	XDPD[13]#	XDP	I/O	B22	CH0_DQ[17]	DDR2	I/O
A13	VSS	Power/Other		B23	CH0_DQSP[11]	DDR2	I/O
A14	CH0_DQ[1]	DDR2	I/O	B24	CH0_DQ[23]	DDR2	I/O
A15	CH0_DQSN[0]	DDR2	I/O	B25	VSS	Power/Other	
A16	CH0_DQ[7]	DDR2	I/O	B26	CH1_DQ[28]	DDR2	I/O
A17	VSS	Power/Other		B27	CH1_DQ[25]	DDR2	I/O
A18	CH1_DQ[13]	DDR2	I/O	B28	CH1_DQSP[12]	DDR2	I/O
A19	CH1_DQSP[10]	DDR2	I/O	B29	CH1_DQ[31]	DDR2	I/O
A20	CH1_DQ[11]	DDR2	I/O	B30	VSS	Power/Other	
A21	VSS	Power/Other		B31	CH0_BA[1]	DDR2	O
A22	CH0_DQ[16]	DDR2	I/O	B32	CH0_CAS#	DDR2	O
A23	CH0_DQSN[11]	DDR2	I/O	B33	CH0_CS[2]#	DDR2	O
A24	CH0_DQ[22]	DDR2	I/O	B34	CH0_CS[3]#	DDR2	O
A25	VSS	Power/Other		B35	VSS	Power/Other	
A26	VCCDDR	Power/Other		B36	CH1_A[8]	DDR2	O
A27	CH1_DQ[24]	DDR2	I/O	B37	VSS	Power/Other	
A28	CH1_DQSN[12]	DDR2	I/O	C1	VSS	Power/Other	
A29	CH1_DQ[30]	DDR2	I/O	C2	PE5RN[3]	PEX	I
A30	CH0_A[10]	DDR2	O	C3	PE5RP[3]	PEX	I
A31	VSS	Power/Other		C4	VSS	Power/Other	
A32	CH0_RAS#	DDR2	O	C5	PE5TN[2]	PEX	O
A33	CH0_A[13]	DDR2	O	C6	PE5TP[2]	PEX	O
A34	CH0_ODT[0]	DDR2	O	C7	VSS	Power/Other	
A35	CH0_ODT[3]	DDR2	O	C8	PE4TN[2]	PEX	O
A36	VSS	Power/Other		C9	PE4RP[1]	PEX	I
B2	VSS	Power/Other		C10	VSS	Power/Other	
B3	TDI	JTAG	I	C11	XDPDSTBP#	XDP	I/O
B4	TDO	JTAG	O	C12	XDPD[9]#	XDP	I/O
B5	VSS	Power/Other		C13	XDPD[15]#	XDP	I/O
B6	RESETI#	CMOS	I	C14	CH0_DQ[5]	DDR2	I/O
B7	TRST#	JTAG	I	C15	VSS	Power/Other	
B8	VSS	Power/Other		C16	CH0_DQ[6]	DDR2	I/O
B9	PE4RN[1]	PEX	I	C17	VSS	Power/Other	
B10	XDPD[7]#	XDP	I/O	C18	CH1_DQ[8]	DDR2	I/O
B11	VSS	Power/Other		C19	VSS	Power/Other	
B12	XDPD[10]#	XDP	I/O	C20	CH1_DQ[10]	DDR2	I/O

Table 143. Intel® 5100 Memory Controller Hub Chipset Signals By Ball (Sheet 2 of 16)

Ball No.	Signal Name	Buffer Type	Direction	Ball No.	Signal Name	Buffer Type	Direction
C21	CH0_DQ[20]	DDR2	I/O	D28	VSS	Power/Other	
C22	CH0_DQSP[2]	DDR2	I/O	D29	VCCDDR	Power/Other	
C23	CH0_DQSN[2]	DDR2	I/O	D30	CH0_A[1]	DDR2	O
C24	VSS	Power/Other		D31	CH0_A[2]	DDR2	O
C25	CH1_DQ[18]	DDR2	I/O	D32	CH0_A[4]	DDR2	O
C26	CH1_DQ[29]	DDR2	I/O	D33	VSS	Power/Other	
C27	CH1_DQSN[3]	DDR2	I/O	D34	CH0_CS[0]#	DDR2	O
C28	CH1_DQSP[3]	DDR2	I/O	D35	CH1_A[12]	DDR2	O
C29	VSS	Power/Other		D36	CH1_A[11]	DDR2	O
C30	CH0_A[0]	DDR2	O	D37	CH1_A[4]	DDR2	O
C31	CH0_BA[0]	DDR2	O	D38	CH1_A[2]	DDR2	O
C32	CH0_WE#	DDR2	O	E1	PE6RN[1]	PEX	I
C33	CH0_CS[1]#	DDR2	O	E2	VSS	Power/Other	
C34	VSS	Power/Other		E3	PE5TN[3]	PEX	O
C35	CH0_ODT[2]	DDR2	O	E4	PE5TP[3]	PEX	O
C36	CH1_A[7]	DDR2	O	E5	VSS	Power/Other	
C37	CH1_A[5]	DDR2	O	E6	VCCPE	Power/Other	
C38	VSS	Power/Other		E7	PE4TN[3]	PEX	O
D1	PE6RP[1]	PEX	I	E8	VSS	Power/Other	
D2	VSS	Power/Other		E9	PE4TN[1]	PEX	O
D3	VSS	Power/Other		E10	XDPD[0]#	XDP	I/O
D4	PE5RN[2]	PEX	I	E11	XDPD[5]#	XDP	I/O
D5	PE5RP[2]	PEX	I	E12	XDPD[6]#	XDP	I/O
D6	VSS	Power/Other		E13	VSS	Power/Other	
D7	PE4TP[3]	PEX	O	E14	CH0_DQ[4]	DDR2	I/O
D8	PE4TP[2]	PEX	O	E15	CH0_DQSP[9]	DDR2	I/O
D9	VSS	Power/Other		E16	CH0_DQ[8]	DDR2	I/O
D10	XDPD[3]#	XDP	I/O	E17	VSS	Power/Other	
D11	XDPDSTBN#	XDP	I/O	E18	CH1_DQ[9]	DDR2	I/O
D12	VSS	Power/Other		E19	CH1_DQ[3]	DDR2	I/O
D13	XDPD[12]#	XDP	I/O	E20	CH1_DQ[14]	DDR2	I/O
D14	VSS	Power/Other		E21	CH0_DQ[28]	DDR2	I/O
D15	CH0_DQSN[9]	DDR2	I/O	E22	VSS	Power/Other	
D16	CH0_DQ[2]	DDR2	I/O	E23	CH0_DQ[25]	DDR2	I/O
D17	CH1_DQ[7]	DDR2	I/O	E24	CH0_DQ[24]	DDR2	I/O
D18	VSS	Power/Other		E25	CH0_CRES1	DDR2	I
D19	VCCDDR	Power/Other		E26	VSS	Power/Other	
D20	CH1_DQ[15]	DDR2	I/O	E27	VSS	Power/Other	
D21	CH0_DQ[21]	DDR2	I/O	E28	CH0_DCLKP[0]	DDR2	O
D22	CH0_DQ[18]	DDR2	I/O	E29	CH0_DCLKP[3]/ CH0_CS[4]#	DDR2	O
D23	VSS	Power/Other		E30	CH0_CRESRET	DDR2	I
D24	CH0_DQ[19]	DDR2	I/O	E31	CH0_A[5]	DDR2	O
D25	CH1_DQ[19]	DDR2	I/O	E32	VSS	Power/Other	
D26	CH1_DQ[26]	DDR2	I/O	E33	VCCDDR	Power/Other	
D27	CH1_DQ[27]	DDR2	I/O	E34	CH0_CKE[1]	DDR2	O

Table 143. Intel® 5100 Memory Controller Hub Chipset Signals By Ball (Sheet 3 of 16)

Ball No.	Signal Name	Buffer Type	Direction	Ball No.	Signal Name	Buffer Type	Direction
E35	CH0_ODT[1]	DDR2	O	G4	PE6TN[0]	PEX	O
E36	CH1_A[9]	DDR2	O	G5	PE6TP[0]	PEX	O
E37	VSS	Power/Other		G6	VSS	Power/Other	
E38	CH1_A[1]	DDR2	O	G7	PE4RP[3]	PEX	I
F1	VSS	Power/Other		G8	PE4RN[2]	PEX	I
F2	PE6RN[0]	PEX	I	G9	VSS	Power/Other	
F3	PE6RP[0]	PEX	I	G10	PE4RN[0]	PEX	I
F4	VSS	Power/Other		G11	CFGSMBDATA	SMB	I/O
F5	PE7TN[0]	PEX	O	G12	PWRGOOD	CMOS	I
F6	PE7TP[0]	PEX	O	G13	XDPCOMCRES	Analog	
F7	VSS	Power/Other		G14	CH0_DQ[12]	DDR2	I/O
F8	PE4RP[2]	PEX	I	G15	VSS	Power/Other	
F9	PE4TP[1]	PEX	O	G16	CH0_DQSN[1]	DDR2	I/O
F10	VSS	Power/Other		G17	CH0_DQSP[1]	DDR2	I/O
F11	XDPD[4]#	XDP	I/O	G18	CH1_DQSN[0]	DDR2	I/O
F12	XDPRDY#	XDP	I/O	G19	CH1_DQSP[0]	DDR2	I/O
F13	V3REF	Analog		G20	VSS	Power/Other	
F14	CH0_DQ[13]	DDR2	I/O	G21	VCCDDR	Power/Other	
F15	CH0_DQ[9]	DDR2	I/O	G22	CH0_DQSP[3]	DDR2	I/O
F16	VSS	Power/Other		G23	CH0_DQSN[3]	DDR2	I/O
F17	CH1_DQ[6]	DDR2	I/O	G24	CH1_DQ[23]	DDR2	I/O
F18	CH1_DQ[2]	DDR2	I/O	G25	VCCDDR	Power/Other	
F19	CH1_DQSN[1]	DDR2	I/O	G26	CH0_DCLKP[1]	DDR2	O
F20	CH1_DQSP[1]	DDR2	I/O	G27	CH0_DCLKN[1]	DDR2	O
F21	VSS	Power/Other		G28	CH0_SLEWCRES	DDR2	I
F22	CH0_DQSN[12]	DDR2	I/O	G29	VCCDDR	Power/Other	
F23	CH0_DQSP[12]	DDR2	I/O	G30	VSS	Power/Other	
F24	CH1_DQ[22]	DDR2	I/O	G31	CH0_CKE[2]	DDR2	O
F25	CH0_CRES2	DDR2	I	G32	CH0_CKE[3]/CH0_ODT[5]	DDR2	O
F26	CH0_DCLKP[2]	DDR2	O	G33	CH1_CKE[1]	DDR2	O
F27	CH0_DCLKN[2]	DDR2	O	G34	CH1_A[15]/CH1_ODT[4]	DDR2	O
F28	CH0_DCLKN[0]	DDR2	O	G35	VSS	Power/Other	
F29	CH0_DCLKN[3]/ CH0_CS[5]#	DDR2	O	G36	CH1_DCLKN[2]	DDR2	O
F30	CH0_A[3]	DDR2	O	G37	CH1_DCLKP[2]	DDR2	O
F31	VSS	Power/Other		G38	CH1_DCLKN[0]	DDR2	O
F32	CH0_A[6]	DDR2	O	H1	VSS	Power/Other	
F33	CH1_CKE[0]	DDR2	O	H2	VSS	Power/Other	
F34	CH1_BA[2]	DDR2	O	H3	PE6RP[2]	PEX	I
F35	CH1_A[6]	DDR2	O	H4	PE6RN[2]	PEX	I
F36	VSS	Power/Other		H5	VSS	Power/Other	
F37	CH1_A[3]	DDR2	O	H6	PE7RP[0]	PEX	I
F38	CH1_DCLKP[0]	DDR2	O	H7	PE4RN[3]	PEX	I
G1	PE6TN[1]	PEX	O	H8	VSS	Power/Other	
G2	PE6TP[1]	PEX	O	H9	PE4TN[0]	PEX	O
G3	VSS	Power/Other		H10	PE4RP[0]	PEX	I

Table 143. Intel® 5100 Memory Controller Hub Chipset Signals By Ball (Sheet 4 of 16)

Ball No.	Signal Name	Buffer Type	Direction	Ball No.	Signal Name	Buffer Type	Direction
H11	VSS	Power/Other		J18	CH1_DQ[1]	DDR2	I/O
H12	GPIOSMBDATA	SMB	I/O	J19	CH1_DQSN[9]	DDR2	I/O
H13	XDPSLWCRES	Analog		J20	CH0_DQ[26]	DDR2	I/O
H14	VSS	Power/Other		J21	CH1_DQ[21]	DDR2	I/O
H15	CH0_DQSN[10]	DDR2	I/O	J22	CH1_DQ[17]	DDR2	I/O
H16	CH0_DQ[15]	DDR2	I/O	J23	CH1_DQSN[2]	DDR2	I/O
H17	CH0_DQ[10]	DDR2	I/O	J24	VSS	Power/Other	
H18	CH0_DQ[11]	DDR2	I/O	J25	CH0_CB[4]	DDR2	I/O
H19	VSS	Power/Other		J26	CH0_CB[1]	DDR2	I/O
H20	CH0_DQ[29]	DDR2	I/O	J27	VSS	Power/Other	
H21	CH0_DQ[31]	DDR2	I/O	J28	CH0_CB[6]	DDR2	I/O
H22	CH0_DQ[30]	DDR2	I/O	J29	CH0_A[11]	DDR2	O
H23	VSS	Power/Other		J30	CH0_CKE[0]	DDR2	O
H24	VSS	Power/Other		J31	CH1_A[10]	DDR2	O
H25	VSS	Power/Other		J32	CH1_A[0]	DDR2	O
H26	CH0_DRVCRES	DDR2	I	J33	VSS	Power/Other	
H27	CH0_CB[0]	DDR2	I/O	J34	CH1_A[14]	DDR2	O
H28	CH0_DQSP[17]	DDR2	I/O	J35	VSS	Power/Other	
H29	CH0_DQSN[17]	DDR2	I/O	J36	CH1_DCLKN[1]	DDR2	O
H30	CH0_A[12]	DDR2	O	J37	CH1_DCLKP[1]	DDR2	O
H31	CH1_CS[0]#	DDR2	O	J38	VSS	Power/Other	
H32	CH1_CKE[3]/CH1_ODT[5]	DDR2	O	K1	PEVCCA	Analog	
H33	CH1_CKE[2]	DDR2	O	K2	PECLKN	Analog	
H34	VSS	Power/Other		K3	VSS	Power/Other	
H35	CH1_DRVCRES	DDR2	I	K4	PE6RP[3]	PEX	I
H36	CH1_SLEWCRES	DDR2	I	K5	PE6TN[3]	PEX	O
H37	CH1_DCLKN[3]/ CH1_CS[5]#	DDR2	O	K6	VSS	Power/Other	
H38	CH1_DCLKP[3]/CH1_CS[4]#	DDR2	O	K7	PE6TP[2]	PEX	O
J1	VSS	Power/Other		K8	PE7TN[1]	PEX	O
J2	PECLKP	Analog		K9	VSS	Power/Other	
J3	VSS	Power/Other		K10	ERR[0]#	CMOS	O
J4	VSS	Power/Other		K11	TDIOCATHODE	Analog	
J5	PE6TP[3]	PEX	O	K12	VSS	Power/Other	
J6	PE7RN[0]	PEX	I	K13	XDPODTCRES	Analog	
J7	VSS	Power/Other		K14	VSS	Power/Other	
J8	PE7TP[1]	PEX	O	K15	VSS	Power/Other	
J9	PE4TP[0]	PEX	O	K16	VSS	Power/Other	
J10	VCCPE	Power/Other		K17	CH1_DQ[0]	DDR2	I/O
J11	TDIOANODE	Analog		K18	VSS	Power/Other	
J12	ERR[2]#	CMOS	O	K19	CH1_DQSP[9]	DDR2	I/O
J13	VSS	Power/Other		K20	VSS	Power/Other	
J14	CH0_DQ[14]	DDR2	I/O	K21	CH1_DQ[20]	DDR2	I/O
J15	CH0_DQSP[10]	DDR2	I/O	K22	VSS	Power/Other	
J16	CH1_DQ[4]	DDR2	I/O	K23	CH1_DQSP[2]	DDR2	I/O
J17	CH1_DQ[5]	DDR2	I/O	K24	CH0_CB[5]	DDR2	I/O

Table 143. Intel® 5100 Memory Controller Hub Chipset Signals By Ball (Sheet 5 of 16)

Ball No.	Signal Name	Buffer Type	Direction	Ball No.	Signal Name	Buffer Type	Direction
K25	CH0_DQSN[8]	DDR2	I/O	L32	VCCDDR	Power/Other	
K26	CH0_DQSP[8]	DDR2	I/O	L33	CH0_DQ[32]	DDR2	I/O
K27	CH0_A[8]	DDR2	O	L34	CH1_CRES2	DDR2	I
K28	CH0_A[7]	DDR2	O	L35	VCCDDR	Power/Other	
K29	VSS	Power/Other		L36	VCCDDR	Power/Other	
K30	CH1_CS[2]#	DDR2	O	L37	CH0_DQSP[4]	DDR2	I/O
K31	CH1_BA[0]	DDR2	O	L38	CH1_DQ[44]	DDR2	I/O
K32	VSS	Power/Other		M1	VSS	Power/Other	
K33	CH0_DQ[36]	DDR2	I/O	M2	PE7RP[3]	PEX	I
K34	CH1_CRES1	DDR2	I	M3	PE7TN[3]	PEX	O
K35	CH0_DQ[37]	DDR2	I/O	M4	VSS	Power/Other	
K36	CH1_CRESRET	DDR2	I	M5	PE7RP[2]	PEX	I
K37	CH0_DQSN[4]	DDR2	I/O	M6	PE7TN[2]	PEX	O
K38	CH0_DQ[38]	DDR2	I/O	M7	VSS	Power/Other	
L1	PEVSSA	Analog		M8	PE5TN[1]	PEX	O
L2	VCCPE	Power/Other		M9	PE5TP[1]	PEX	O
L3	PE7TP[3]	PEX	O	M10	VSS	Power/Other	
L4	PE6RN[3]	PEX	I	M11	VCCPE	Power/Other	
L5	VSS	Power/Other		M12	VSS	Power/Other	
L6	PE7TP[2]	PEX	O	M13	GPIOSMBCLK	SMB	I/O
L7	PE6TN[2]	PEX	O	M14	VCCPE	Power/Other	
L8	VCCPE	Power/Other		M15	VCCPE	Power/Other	
L9	VSS	Power/Other		M16	VCC	Power/Other	
L10	SPD0SMBDATA	SMB	I/O	M17	VCC	Power/Other	
L11	VSS	Power/Other		M18	VCC	Power/Other	
L12	CFGSMBCLK	SMB	I/O	M19	VSS	Power/Other	
L13	ERR[1]#	CMOS	O	M20	VSS	Power/Other	
L14	VCCPE	Power/Other		M21	VSS	Power/Other	
L15	VCCPE	Power/Other		M22	VSS	Power/Other	
L16	VCC	Power/Other		M23	VSS	Power/Other	
L17	VCC	Power/Other		M24	VSS	Power/Other	
L18	VCC	Power/Other		M25	VSS	Power/Other	
L19	VCC	Power/Other		M26	CH1_CS[3]#	DDR2	O
L20	CH0_DQ[27]	DDR2	I/O	M27	CH0_BA[2]	DDR2	O
L21	CH1_DQSP[11]	DDR2	I/O	M28	VSS	Power/Other	
L22	CH1_DQSN[11]	DDR2	I/O	M29	CH1_A[13]	DDR2	O
L23	CH1_DQ[16]	DDR2	I/O	M30	CH1_RAS#	DDR2	O
L24	CH0_CB[7]	DDR2	I/O	M31	CH1_CAS#	DDR2	O
L25	CH0_CB[2]	DDR2	I/O	M32	CH1_BA[1]	DDR2	O
L26	CH0_CB[3]	DDR2	I/O	M33	CH0_DQ[33]	DDR2	I/O
L27	CH0_A[9]	DDR2	O	M34	CH0_DQSP[13]	DDR2	I/O
L28	CH0_A[15]/CH0_ODT[4]	DDR2	O	M35	CH0_DQ[39]	DDR2	I/O
L29	CH1_CS[1]#	DDR2	O	M36	VSS	Power/Other	
L30	CH1_WE#	DDR2	O	M37	CH0_DQ[35]	DDR2	I/O
L31	VSS	Power/Other		M38	CH1_DQ[41]	DDR2	I/O

Table 143. Intel® 5100 Memory Controller Hub Chipset Signals By Ball (Sheet 6 of 16)

Ball No.	Signal Name	Buffer Type	Direction	Ball No.	Signal Name	Buffer Type	Direction
N1	PE2TP[2]	PEX	O	P8	VSS	Power/Other	
N2	PE7RN[3]	PEX	I	P9	PE5TN[0]	PEX	O
N3	VSS	Power/Other		P10	PE5TP[0]	PEX	O
N4	PE2RP[1]	PEX	I	P11	VSS	Power/Other	
N5	PE7RN[2]	PEX	I	P12	PERCOMPO	Analog	
N6	VCCPE	Power/Other		P13	VCCPE	Power/Other	
N7	PE5RN[1]	PEX	I	P14	VCCPE	Power/Other	
N8	PE5RP[1]	PEX	I	P15	VSS	Power/Other	
N9	VSS	Power/Other		P16	VCC	Power/Other	
N10	VCCPE	Power/Other		P17	VSS	Power/Other	
N11	PEVSSBG	Analog		P18	VCC	Power/Other	
N12	VCCPE	Power/Other		P19	VSS	Power/Other	
N13	VCCPE	Power/Other		P20	VCC	Power/Other	
N14	VCCPE	Power/Other		P21	VSS	Power/Other	
N15	VCCPE	Power/Other		P22	VCC	Power/Other	
N16	VSS	Power/Other		P23	VSS	Power/Other	
N17	VCC	Power/Other		P24	VCC	Power/Other	
N18	VSS	Power/Other		P25	VCCDDR	Power/Other	
N19	VCC	Power/Other		P26	VSS	Power/Other	
N20	VCCDDR	Power/Other		P27	CH1_ODT[3]	DDR2	O
N21	VCCDDR	Power/Other		P28	CH1_CB[6]	DDR2	I/O
N22	VCCDDR	Power/Other		P29	CH1_ODT[0]	DDR2	O
N23	VCCDDR	Power/Other		P30	CH1_DQ[37]	DDR2	I/O
N24	VCCDDR	Power/Other		P31	CH1_DQSN[4]	DDR2	I/O
N25	VCCDDR	Power/Other		P32	CH1_DQSP[13]	DDR2	I/O
N26	CH0_A[14]	DDR2	O	P33	CH1_DQ[39]	DDR2	I/O
N27	CH1_ODT[2]	DDR2	O	P34	CH0_DQ[34]	DDR2	I/O
N28	CH1_ODT[1]	DDR2	O	P35	CH0_DQ[40]	DDR2	I/O
N29	CH1_CB[3]	DDR2	I/O	P36	CH0_DQSP[14]	DDR2	I/O
N30	VCCDDR	Power/Other		P37	CH0_DQSN[14]	DDR2	I/O
N31	VSS	Power/Other		P38	VSS	Power/Other	
N32	CH1_DQSN[13]	DDR2	I/O	R1	VSS	Power/Other	
N33	CH1_DQ[38]	DDR2	I/O	R2	PE2TP[3]	PEX	O
N34	CH0_DQSN[13]	DDR2	I/O	R3	PE2RN[2]	PEX	I
N35	VSS	Power/Other		R4	VCCPE	Power/Other	
N36	CH1_DQSP[14]	DDR2	I/O	R5	PE2RN[0]	PEX	I
N37	CH1_DQSN[14]	DDR2	I/O	R6	VSS	Power/Other	
N38	CH0_DQ[41]	DDR2	I/O	R7	VSS	Power/Other	
P1	PE2TN[2]	PEX	O	R8	PE5RN[0]	PEX	I
P2	VSS	Power/Other		R9	PE5RP[0]	PEX	I
P3	PE2RP[2]	PEX	I	R10	VCCPE	Power/Other	
P4	PE2RN[1]	PEX	I	R11	PEVCCBG	Analog	
P5	VSS	Power/Other		R12	PEICOMPI	Analog	
P6	PE7RN[1]	PEX	I	R13	VCCPE	Power/Other	
P7	PE7RP[1]	PEX	I	R14	VCCPE	Power/Other	

Table 143. Intel® 5100 Memory Controller Hub Chipset Signals By Ball (Sheet 7 of 16)

Ball No.	Signal Name	Buffer Type	Direction	Ball No.	Signal Name	Buffer Type	Direction
R15	VCC	Power/Other		T22	VCC	Power/Other	
R16	VSS	Power/Other		T23	VSS	Power/Other	
R17	VCC	Power/Other		T24	VCC	Power/Other	
R18	VSS	Power/Other		T25	VCCDDR	Power/Other	
R19	VCC	Power/Other		T26	VSS	Power/Other	
R20	VSS	Power/Other		T27	CH1_CB[0]	DDR2	I/O
R21	VCC	Power/Other		T28	CH1_DQSP[17]	DDR2	I/O
R22	VSS	Power/Other		T29	CH1_CB[7]	DDR2	I/O
R23	VCC	Power/Other		T30	CH1_CB[2]	DDR2	I/O
R24	VSS	Power/Other		T31	VSS	Power/Other	
R25	VCCDDR	Power/Other		T32	VCCDDR	Power/Other	
R26	VSS	Power/Other		T33	CH1_DQ[47]	DDR2	I/O
R27	CH1_CB[1]	DDR2	I/O	T34	CH1_DQ[46]	DDR2	I/O
R28	CH1_DQSN[17]	DDR2	I/O	T35	CH0_DQ[47]	DDR2	I/O
R29	VSS	Power/Other		T36	VSS	Power/Other	
R30	CH1_DQ[36]	DDR2	I/O	T37	CH0_DQ[42]	DDR2	I/O
R31	CH1_DQSP[4]	DDR2	I/O	T38	CH0_DQ[43]	DDR2	I/O
R32	VSS	Power/Other		U1	PE2RN[3]	PEX	I
R33	CH1_DQ[45]	DDR2	I/O	U2	VCCPE	Power/Other	
R34	CH1_DQ[40]	DDR2	I/O	U3	PE3TP[0]	PEX	O
R35	CH0_DQSP[5]	DDR2	I/O	U4	PE2TP[1]	PEX	O
R36	CH0_DQSN[5]	DDR2	I/O	U5	VSS	Power/Other	
R37	VSS	Power/Other		U6	PE3TP[2]	PEX	O
R38	CH0_DQ[46]	DDR2	I/O	U7	PE3TN[2]	PEX	O
T1	PE2RP[3]	PEX	I	U8	VCCPE	Power/Other	
T2	PE2TN[3]	PEX	O	U9	PE3RP[3]	PEX	I
T3	VSS	Power/Other		U10	PE3RN[3]	PEX	I
T4	PE2TN[1]	PEX	O	U11	VSS	Power/Other	
T5	PE2RP[0]	PEX	I	U12	VSS	Power/Other	
T6	VSS	Power/Other		U13	VCCPE	Power/Other	
T7	PE2TN[0]	PEX	O	U14	VCCPE	Power/Other	
T8	PE2TP[0]	PEX	O	U15	VCC	Power/Other	
T9	VSS	Power/Other		U16	VSS	Power/Other	
T10	VSS	Power/Other		U17	VCC	Power/Other	
T11	VSS	Power/Other		U18	VSS	Power/Other	
T12	VSS	Power/Other		U19	VCC	Power/Other	
T13	VCCPE	Power/Other		U20	VSS	Power/Other	
T14	VCCPE	Power/Other		U21	VCC	Power/Other	
T15	VSS	Power/Other		U22	VSS	Power/Other	
T16	VCC	Power/Other		U23	VCC	Power/Other	
T17	VSS	Power/Other		U24	VSS	Power/Other	
T18	VCC	Power/Other		U25	VCCDDR	Power/Other	
T19	VSS	Power/Other		U26	VSS	Power/Other	
T20	VCC	Power/Other		U27	CH1_CB[5]	DDR2	I/O
T21	VSS	Power/Other		U28	CH1_DQSN[8]	DDR2	I/O

Table 143. Intel® 5100 Memory Controller Hub Chipset Signals By Ball (Sheet 8 of 16)

Ball No.	Signal Name	Buffer Type	Direction	Ball No.	Signal Name	Buffer Type	Direction
U29	CH1_DQSP[8]	DDR2	I/O	V36	CH0_DQ[49]	DDR2	I/O
U30	VSS	Power/Other		V37	CH0_DQSP[15]	DDR2	I/O
U31	VSS	Power/Other		V38	CH0_DQSN[15]	DDR2	I/O
U32	CH1_DQSN[5]	DDR2	I/O	W1	VSS	Power/Other	
U33	CH1_DQSP[5]	DDR2	I/O	W2	PE3RN[0]	PEX	I
U34	CH0_DQ[45]	DDR2	I/O	W3	VSS	Power/Other	
U35	VSS	Power/Other		W4	PE3TN[1]	PEX	O
U36	VCCDDR	Power/Other		W5	PE3TP[1]	PEX	O
U37	CH0_DQ[52]	DDR2	I/O	W6	VCCPE	Power/Other	
U38	CH0_DQ[48]	DDR2	I/O	W7	PE3RP[2]	PEX	I
V1	VSS	Power/Other		W8	PE3RN[2]	PEX	I
V2	PE3RP[0]	PEX	I	W9	VSS	Power/Other	
V3	PE3TN[0]	PEX	O	W10	PEWIDTH[3]	CMOS	I
V4	VSS	Power/Other		W11	PEWIDTH[2]	CMOS	I
V5	PE3RP[1]	PEX	I	W12	VCCPE	Power/Other	
V6	PE3RN[1]	PEX	I	W13	VCCPE	Power/Other	
V7	VSS	Power/Other		W14	VCCPE	Power/Other	
V8	PE3TP[3]	PEX	O	W15	VCC	Power/Other	
V9	PE3TN[3]	PEX	O	W16	VSS	Power/Other	
V10	VSS	Power/Other		W17	VCC	Power/Other	
V11	VSS	Power/Other		W18	VSS	Power/Other	
V12	VSS	Power/Other		W19	VCC	Power/Other	
V13	VCCPE	Power/Other		W20	VSS	Power/Other	
V14	VCCPE	Power/Other		W21	VCC	Power/Other	
V15	VSS	Power/Other		W22	VSS	Power/Other	
V16	VCC	Power/Other		W23	VCC	Power/Other	
V17	VSS	Power/Other		W24	VSS	Power/Other	
V18	VCC	Power/Other		W25	VCCDDR	Power/Other	
V19	VSS	Power/Other		W26	VSS	Power/Other	
V20	VCC	Power/Other		W27	CH1_DQ[34]	DDR2	I/O
V21	VSS	Power/Other		W28	VSS	Power/Other	
V22	VCC	Power/Other		W29	CH1_DQ[35]	DDR2	I/O
V23	VSS	Power/Other		W30	CH0_DQ[44]	DDR2	I/O
V24	VCC	Power/Other		W31	VSS	Power/Other	
V25	VCCDDR	Power/Other		W32	CH0_DQ[55]	DDR2	I/O
V26	VSS	Power/Other		W33	VSS	Power/Other	
V27	CH1_CB[4]	DDR2	I/O	W34	CH0_DQ[60]	DDR2	I/O
V28	CH1_DQ[32]	DDR2	I/O	W35	CH0_DQ[51]	DDR2	I/O
V29	CH1_DQ[33]	DDR2	I/O	W36	CH0_DQSN[6]	DDR2	I/O
V30	VCCDDR	Power/Other		W37	CH0_DQSP[6]	DDR2	I/O
V31	CH1_DQ[42]	DDR2	I/O	W38	VSS	Power/Other	
V32	CH1_DQ[43]	DDR2	I/O	Y1	VSS	Power/Other	
V33	CH0_DQ[54]	DDR2	I/O	Y2	VSS	Power/Other	
V34	VSS	Power/Other		Y3	PE0RN[1]	PEX	I
V35	CH0_DQ[53]	DDR2	I/O	Y4	PE0RP[1]	PEX	I

Table 143. Intel® 5100 Memory Controller Hub Chipset Signals By Ball (Sheet 9 of 16)

Ball No.	Signal Name	Buffer Type	Direction	Ball No.	Signal Name	Buffer Type	Direction
Y5	VSS	Power/Other		AA12	FSB1A[16]#	Source Sync	I/O
Y6	PE0TN[0]	PEX	O	AA13	VCC	Power/Other	
Y7	PE0TP[0]	PEX	O	AA14	VSS	Power/Other	
Y8	VSS	Power/Other		AA15	VCC	Power/Other	
Y9	PE0RN[0]	PEX	I	AA16	VSS	Power/Other	
Y10	PE0RP[0]	PEX	I	AA17	VCC	Power/Other	
Y11	VSS	Power/Other		AA18	VSS	Power/Other	
Y12	PEWIDTH[1]	CMOS	I	AA19	VCC	Power/Other	
Y13	VCCPE	Power/Other		AA20	VSS	Power/Other	
Y14	VCCPE	Power/Other		AA21	VCC	Power/Other	
Y15	VSS	Power/Other		AA22	VSS	Power/Other	
Y16	VCC	Power/Other		AA23	VCC	Power/Other	
Y17	VSS	Power/Other		AA24	VSS	Power/Other	
Y18	VCC	Power/Other		AA25	VCCDDR	Power/Other	
Y19	VSS	Power/Other		AA26	VSS	Power/Other	
Y20	VCC	Power/Other		AA27	CH0_DQ[59]	DDR2	I/O
Y21	VSS	Power/Other		AA28	CH0_DQ[62]	DDR2	I/O
Y22	VCC	Power/Other		AA29	VSS	Power/Other	
Y23	VSS	Power/Other		AA30	CH1_DQ[49]	DDR2	I/O
Y24	VCC	Power/Other		AA31	CH1_DQSP[15]	DDR2	I/O
Y25	VCCDDR	Power/Other		AA32	CH1_DQSN[15]	DDR2	I/O
Y26	VSS	Power/Other		AA33	CH1_DQ[50]	DDR2	I/O
Y27	CH0_DQ[58]	DDR2	I/O	AA34	CH1_DQ[55]	DDR2	I/O
Y28	CH0_DQ[63]	DDR2	I/O	AA35	CH1_DQ[54]	DDR2	I/O
Y29	CH0_DQSP[7]	DDR2	I/O	AA36	VSS	Power/Other	
Y30	CH0_DQSN[7]	DDR2	I/O	AA37	CH1_DQ[60]	DDR2	I/O
Y31	CH0_DQ[57]	DDR2	I/O	AA38	CH1_DQ[61]	DDR2	I/O
Y32	VSS	Power/Other		AB1	PSEL[2]	CMOS	I
Y33	CH0_DQSN[16]	DDR2	I/O	AB2	PSEL[1]	CMOS	I
Y34	CH0_DQSP[16]	DDR2	I/O	AB3	VSS	Power/Other	
Y35	CH0_DQ[56]	DDR2	I/O	AB4	PE0TP[2]	PEX	O
Y36	CH0_DQ[50]	DDR2	I/O	AB5	PE0TN[2]	PEX	O
Y37	VSS	Power/Other		AB6	VSS	Power/Other	
Y38	CH0_DQ[61]	DDR2	I/O	AB7	PE0RN[2]	PEX	I
AA1	VSS	Power/Other		AB8	PE0RP[2]	PEX	I
AA2	PE0TN[1]	PEX	O	AB9	VSS	Power/Other	
AA3	PE0TP[1]	PEX	O	AB10	FSB1A[14]#	Source Sync	I/O
AA4	VSS	Power/Other		AB11	FSB1A[10]#	Source Sync	I/O
AA5	PE0RP[3]	PEX	I	AB12	VSS	Power/Other	
AA6	PE0RN[3]	PEX	I	AB13	VCC	Power/Other	
AA7	VSS	Power/Other		AB14	VCC	Power/Other	
AA8	PE0TP[3]	PEX	O	AB15	VSS	Power/Other	
AA9	PE0TN[3]	PEX	O	AB16	VCC	Power/Other	
AA10	VSS	Power/Other		AB17	VSS	Power/Other	
AA11	PEWIDTH[0]	CMOS	I	AB18	VCC	Power/Other	

Table 143. Intel® 5100 Memory Controller Hub Chipset Signals By Ball (Sheet 10 of 16)

Ball No.	Signal Name	Buffer Type	Direction	Ball No.	Signal Name	Buffer Type	Direction
AB19	VSS	Power/Other		AC26	VCC	Power/Other	
AB20	VCC	Power/Other		AC27	VSS	Power/Other	
AB21	VSS	Power/Other		AC28	VSS	Power/Other	
AB22	VCC	Power/Other		AC29	CH1_DQ[48]	DDR2	I/O
AB23	VSS	Power/Other		AC30	FSB0D[42]#	Source Sync	I/O
AB24	VCC	Power/Other		AC31	FSB0D[44]#	Source Sync	I/O
AB25	VCCDDR	Power/Other		AC32	VSS	Power/Other	
AB26	VSS	Power/Other		AC33	CH1_DQ[59]	DDR2	I/O
AB27	VSS	Power/Other		AC34	CH1_DQ[58]	DDR2	I/O
AB28	CH1_DQ[52]	DDR2	I/O	AC35	CH1_DQ[63]	DDR2	I/O
AB29	CH1_DQ[53]	DDR2	I/O	AC36	VCCDDR	Power/Other	
AB30	CH1_DQSN[6]	DDR2	I/O	AC37	CH1_DQSN[7]	DDR2	I/O
AB31	CH1_DQSP[6]	DDR2	I/O	AC38	CH1_DQSP[7]	DDR2	I/O
AB32	CH1_DQ[62]	DDR2	I/O	AD1	VSS	Power/Other	
AB33	CH1_DQ[57]	DDR2	I/O	AD2	FSB1A[33]#	Source Sync	I/O
AB34	VSS	Power/Other		AD3	FSB1A[29]#	Source Sync	I/O
AB35	CH1_DQ[51]	DDR2	I/O	AD4	VSS	Power/Other	
AB36	CH1_DQ[56]	DDR2	I/O	AD5	FSB1A[23]#	Source Sync	I/O
AB37	CH1_DQSN[16]	DDR2	I/O	AD6	FSB1A[20]#	Source Sync	I/O
AB38	CH1_DQSP[16]	DDR2	I/O	AD7	VSS	Power/Other	
AC1	PSEL[0]	CMOS	I	AD8	FSB1A[12]#	Source Sync	I/O
AC2	VSS	Power/Other		AD9	FSB1A[4]#	Source Sync	I/O
AC3	FSB1A[35]#	Source Sync	I/O	AD10	VSS	Power/Other	
AC4	FSB1A[34]#	Source Sync	I/O	AD11	FSB1REQ[1]#	Source Sync	I/O
AC5	VSS	Power/Other		AD12	FSB1A[3]#	Source Sync	I/O
AC6	FSB1A[22]#	Source Sync	I/O	AD13	VTT	Power/Other	
AC7	FSB1A[15]#	Source Sync	I/O	AD14	VTT	Power/Other	
AC8	VSS	Power/Other		AD15	VTT	Power/Other	
AC9	FSB1A[13]#	Source Sync	I/O	AD16	VTT	Power/Other	
AC10	FSB1ADSTB[0]#	Source Sync	I/O	AD17	VTT	Power/Other	
AC11	VSS	Power/Other		AD18	VTT	Power/Other	
AC12	FSB1A[5]#	Source Sync	I/O	AD19	VTT	Power/Other	
AC13	VTT	Power/Other		AD20	VTT	Power/Other	
AC14	VTT	Power/Other		AD21	VTT	Power/Other	
AC15	VCC	Power/Other		AD22	VTT	Power/Other	
AC16	VSS	Power/Other		AD23	VTT	Power/Other	
AC17	VCC	Power/Other		AD24	VTT	Power/Other	
AC18	VSS	Power/Other		AD25	VTT	Power/Other	
AC19	VCC	Power/Other		AD26	VCC	Power/Other	
AC20	VSS	Power/Other		AD27	VSS	Power/Other	
AC21	VCC	Power/Other		AD28	VSS	Power/Other	
AC22	VSS	Power/Other		AD29	FSB0D[46]#	Source Sync	I/O
AC23	VCC	Power/Other		AD30	FSB0D[41]#	Source Sync	I/O
AC24	VSS	Power/Other		AD31	VSS	Power/Other	
AC25	VCC	Power/Other		AD32	FSB0DSTBN[2]#	Source Sync	I/O

Table 143. Intel® 5100 Memory Controller Hub Chipset Signals By Ball (Sheet 11 of 16)

Ball No.	Signal Name	Buffer Type	Direction	Ball No.	Signal Name	Buffer Type	Direction
AD33	FSB0DSTBP[2]#	Source Sync	I/O	AF2	VSS	Power/Other	
AD34	VSS	Power/Other		AF3	FSB1A[28]#	Source Sync	I/O
AD35	FSB0D[38]#	Source Sync	I/O	AF4	FSB1A[17]#	Source Sync	I/O
AD36	VSS	Power/Other		AF5	VSS	Power/Other	
AD37	VSS	Power/Other		AF6	FSB1A[11]#	Source Sync	I/O
AD38	VSS	Power/Other		AF7	FSB1A[8]#	Source Sync	I/O
AE1	FSB1A[32]#	Source Sync	I/O	AF8	VSS	Power/Other	
AE2	FSB1A[31]#	Source Sync	I/O	AF9	FSB1REQ[3]#	Source Sync	I/O
AE3	VSS	Power/Other		AF10	VSS	Power/Other	
AE4	FSB1A[30]#	Source Sync	I/O	AF11	VSS	Power/Other	
AE5	FSB1A[21]#	Source Sync	I/O	AF12	FSB1DP[2]#	Common Clk	I/O
AE6	VSS	Power/Other		AF13	FSB1DP[3]#	Common Clk	I/O
AE7	FSB1A[9]#	Source Sync	I/O	AF14	VSS	Power/Other	
AE8	RSVD			AF15	FSB1D[58]#	Source Sync	I/O
AE9	VSS	Power/Other		AF16	FSB1D[63]#	Source Sync	I/O
AE10	FSB1REQ[4]#	Source Sync	I/O	AF17	VSS	Power/Other	
AE11	FSB1RESET#	Common Clk	O	AF18	VTT	Power/Other	
AE12	VSS	Power/Other		AF19	VTT	Power/Other	
AE13	VTT	Power/Other		AF20	VTT	Power/Other	
AE14	VTT	Power/Other		AF21	VTT	Power/Other	
AE15	VTT	Power/Other		AF22	FSB0A[16]#	Source Sync	I/O
AE16	VTT	Power/Other		AF23	VSS	Power/Other	
AE17	VTT	Power/Other		AF24	VSS	Power/Other	
AE18	VTT	Power/Other		AF25	FSB0A[14]#	Source Sync	I/O
AE19	VTT	Power/Other		AF26	VSS	Power/Other	
AE20	VTT	Power/Other		AF27	VSS	Power/Other	
AE21	VTT	Power/Other		AF28	FSB0D[45]#	Source Sync	I/O
AE22	VTT	Power/Other		AF29	VSS	Power/Other	
AE23	VTT	Power/Other		AF30	FSB0DBI[2]#	Source Sync	I/O
AE24	VTT	Power/Other		AF31	FSB0D[35]#	Source Sync	I/O
AE25	VTT	Power/Other		AF32	VSS	Power/Other	
AE26	VTT	Power/Other		AF33	FSB0D[37]#	Source Sync	I/O
AE27	VSS	Power/Other		AF34	FSB0VREF	Power/Other	
AE28	FSB0D[47]#	Source Sync	I/O	AF35	VSS	Power/Other	
AE29	FSB0D[43]#	Source Sync	I/O	AF36	VSS	Power/Other	
AE30	VSS	Power/Other		AF37	FSB0D[48]#	Source Sync	I/O
AE31	FSB0D[40]#	Source Sync	I/O	AF38	FSB0D[59]#	Source Sync	I/O
AE32	FSB0D[39]#	Source Sync	I/O	AG1	VSS	Power/Other	
AE33	VSS	Power/Other		AG2	RSVD		
AE34	FSB0D[34]#	Source Sync	I/O	AG3	FSB1ADSTB[1]#	Source Sync	I/O
AE35	VSS	Power/Other		AG4	VSS	Power/Other	
AE36	FSB0D[62]#	Source Sync	I/O	AG5	FSB1A[18]#	Source Sync	I/O
AE37	FSB0D[63]#	Source Sync	I/O	AG6	VSS	Power/Other	
AE38	FSB0D[58]#	Source Sync	I/O	AG7	VSS	Power/Other	
AF1	FSB1A[27]#	Source Sync	I/O	AG8	FSB1A[7]#	Source Sync	I/O

Table 143. Intel® 5100 Memory Controller Hub Chipset Signals By Ball (Sheet 12 of 16)

Ball No.	Signal Name	Buffer Type	Direction	Ball No.	Signal Name	Buffer Type	Direction
AG9	FSB1REQ[0]#	Source Sync	I/O	AH16	FSB1D[54]#	Source Sync	I/O
AG10	FSB1AP[1]#	Common Clk	I/O	AH17	ASYNCRFSH	CMOS	I
AG11	FSB1DP[0]#	Common Clk	I/O	AH18	VTT	Power/Other	
AG12	FSB1AP[0]#	Common Clk	I/O	AH19	VTT	Power/Other	
AG13	VSS	Power/Other		AH20	VTT	Power/Other	
AG14	FSB1D[62]#	Source Sync	I/O	AH21	VTT	Power/Other	
AG15	FSB1D[59]#	Source Sync	I/O	AH22	FSB0A[13]#	Source Sync	I/O
AG16	VSS	Power/Other		AH23	FSB0A[8]#	Source Sync	I/O
AG17	VSS	Power/Other		AH24	VSS	Power/Other	
AG18	VTT	Power/Other		AH25	FSB0REQ[1]#	Source Sync	I/O
AG19	VTT	Power/Other		AH26	FSB0AP[0]#	Common Clk	I/O
AG20	VTT	Power/Other		AH27	VSS	Power/Other	
AG21	VTT	Power/Other		AH28	FSB0D[2]#	Source Sync	I/O
AG22	VSS	Power/Other		AH29	FSB0D[4]#	Source Sync	I/O
AG23	FSB0A[15]#	Source Sync	I/O	AH30	VSS	Power/Other	
AG24	FSB0A[10]#	Source Sync	I/O	AH31	FSB0D[11]#	Source Sync	I/O
AG25	VSS	Power/Other		AH32	FSB0D[9]#	Source Sync	I/O
AG26	FSB0BREQ1#	Common Clk	I/O	AH33	VSS	Power/Other	
AG27	FSB0D[14]#	Source Sync	I/O	AH34	FSB0DSTBN[3]#	Source Sync	I/O
AG28	VSS	Power/Other		AH35	FSB0DSTBP[3]#	Source Sync	I/O
AG29	FSB0D[13]#	Source Sync	I/O	AH36	FSB0D[61]#	Source Sync	I/O
AG30	FSB0D[33]#	Source Sync	I/O	AH37	FSB0DBI[3]#	Source Sync	I/O
AG31	VSS	Power/Other		AH38	FSB0D[57]#	Source Sync	I/O
AG32	FSB0D[36]#	Source Sync	I/O	AJ1	RSVD		
AG33	FSB0D[32]#	Source Sync	I/O	AJ2	VSS	Power/Other	
AG34	VSS	Power/Other		AJ3	FSB1A[26]#	Source Sync	I/O
AG35	FSB0D[54]#	Source Sync	I/O	AJ4	FSB1BINIT#	Common Clk	I/O
AG36	FSB0D[60]#	Source Sync	I/O	AJ5	VSS	Power/Other	
AG37	VSS	Power/Other		AJ6	FSB1REQ[2]#	Source Sync	I/O
AG38	VSS	Power/Other		AJ7	FSB1HITM#	Common Clk	I/O
AH1	FSB1A[25]#	Source Sync	I/O	AJ8	VSS	Power/Other	
AH2	FSB1A[24]#	Source Sync	I/O	AJ9	FSB1DEFER#	Common Clk	O
AH3	VSS	Power/Other		AJ10	FSB1BPRI#	Common Clk	O
AH4	FSB1VREF	Power/Other		AJ11	VSS	Power/Other	
AH5	FSB1A[19]#	Source Sync	I/O	AJ12	FSB1DP[1]#	Common Clk	I/O
AH6	VSS	Power/Other		AJ13	FSB1D[57]#	Source Sync	I/O
AH7	RSVD			AJ14	VSS	Power/Other	
AH8	FSB1A[6]#	Source Sync	I/O	AJ15	FSB1D[60]#	Source Sync	I/O
AH9	VSS	Power/Other		AJ16	FSB1D[61]#	Source Sync	I/O
AH10	VSS	Power/Other		AJ17	VSS	Power/Other	
AH11	FSB1MCERR#	Common Clk	I/O	AJ18	VTT	Power/Other	
AH12	VSS	Power/Other		AJ19	VTT	Power/Other	
AH13	FSB1DBI[3]#	Source Sync	I/O	AJ20	VTT	Power/Other	
AH14	FSB1D[48]#	Source Sync	I/O	AJ21	VTT	Power/Other	
AH15	VSS	Power/Other		AJ22	FSB0A[11]#	Source Sync	I/O

Table 143. Intel® 5100 Memory Controller Hub Chipset Signals By Ball (Sheet 13 of 16)

Ball No.	Signal Name	Buffer Type	Direction	Ball No.	Signal Name	Buffer Type	Direction
AJ23	VSS	Power/Other		AK30	FSB0D[6]#	Source Sync	I/O
AJ24	FSB0REQ[3]#	Source Sync	I/O	AK31	VSS	Power/Other	
AJ25	FSB0REQ[4]#	Source Sync	I/O	AK32	FSB0DSTBN[0]#	Source Sync	I/O
AJ26	VSS	Power/Other		AK33	FSB0DSTBP[0]#	Source Sync	I/O
AJ27	FSB0MCERR#	Common Clk	I/O	AK34	VSS	Power/Other	
AJ28	FSB0D[0]#	Source Sync	I/O	AK35	FSB0D[30]#	Source Sync	I/O
AJ29	VSS	Power/Other		AK36	FSB0D[53]#	Source Sync	I/O
AJ30	FSB0D[10]#	Source Sync	I/O	AK37	VSS	Power/Other	
AJ31	FSB0D[8]#	Source Sync	I/O	AK38	VSS	Power/Other	
AJ32	VSS	Power/Other		AL1	FSB1RS[1]#	Common Clk	O
AJ33	FSB0D[15]#	Source Sync	I/O	AL2	FSB1BREQ0#	Common Clk	I/O
AJ34	FSB0D[49]#	Source Sync	I/O	AL3	VSS	Power/Other	
AJ35	VSS	Power/Other		AL4	FSB1LOCK#	Common Clk	I/O
AJ36	VSS	Power/Other		AL5	FSB1RS[2]#	Common Clk	O
AJ37	FSB0D[55]#	Source Sync	I/O	AL6	VSS	Power/Other	
AJ38	FSB0D[56]#	Source Sync	I/O	AL7	FSB1D[16]#	Source Sync	I/O
AK1	VSS	Power/Other		AL8	FSB1D[17]#	Source Sync	I/O
AK2	FSB1RSP#	Common Clk	O	AL9	VSS	Power/Other	
AK3	FSB1BNR#	Common Clk	I/O	AL10	FSB1DSTBN[1]#	Source Sync	I/O
AK4	VSS	Power/Other		AL11	FSB1D[29]#	Source Sync	I/O
AK5	FSB1RS[0]#	Common Clk	O	AL12	VSS	Power/Other	
AK6	FSB1TRDY#	Common Clk	O	AL13	FSB1D[30]#	Source Sync	I/O
AK7	VSS	Power/Other		AL14	FSB1D[52]#	Source Sync	I/O
AK8	FSB1HIT#	Common Clk	I/O	AL15	VSS	Power/Other	
AK9	FSB1D[19]#	Source Sync	I/O	AL16	FSB1D[56]#	Source Sync	I/O
AK10	VSS	Power/Other		AL17	VCC	Power/Other	
AK11	FSB1DBI[1]#	Source Sync	I/O	AL18	VTT	Power/Other	
AK12	FSB1D[31]#	Source Sync	I/O	AL19	VTT	Power/Other	
AK13	VSS	Power/Other		AL20	VTT	Power/Other	
AK14	FSB1DSTBN[3]#	Source Sync	I/O	AL21	VTT	Power/Other	
AK15	FSB1DSTBP[3]#	Source Sync	I/O	AL22	FSB0A[12]#	Source Sync	I/O
AK16	VSS	Power/Other		AL23	FSB0ADSTB[0]#	Source Sync	I/O
AK17	RSVD			AL24	VSS	Power/Other	
AK18	VTT	Power/Other		AL25	FSB0A[3]#	Source Sync	I/O
AK19	VTT	Power/Other		AL26	FSB0REQ[0]#	Source Sync	I/O
AK20	VTT	Power/Other		AL27	VSS	Power/Other	
AK21	VTT	Power/Other		AL28	FSB0RS[1]#	Common Clk	O
AK22	VSS	Power/Other		AL29	FSB0D[1]#	Source Sync	I/O
AK23	RSVD			AL30	VSS	Power/Other	
AK24	FSB0REQ[2]#	Source Sync	I/O	AL31	FSB0D[7]#	Source Sync	I/O
AK25	VSS	Power/Other		AL32	FSB0DBI[0]#	Source Sync	I/O
AK26	FSB0AP[1]#	Common Clk	I/O	AL33	VSS	Power/Other	
AK27	FSB0BINIT#	Common Clk	I/O	AL34	FSB0D[17]#	Source Sync	I/O
AK28	VSS	Power/Other		AL35	FSB0D[25]#	Source Sync	I/O
AK29	FSB0D[3]#	Source Sync	I/O	AL36	FSB0D[51]#	Source Sync	I/O

Table 143. Intel® 5100 Memory Controller Hub Chipset Signals By Ball (Sheet 14 of 16)

Ball No.	Signal Name	Buffer Type	Direction	Ball No.	Signal Name	Buffer Type	Direction
AL37	FSB0D[52]#	Source Sync	I/O	AN6	FSB1D[18]#	Source Sync	I/O
AL38	FSB0D[50]#	Source Sync	I/O	AN7	VSS	Power/Other	
AM1	FSB1BREQ1#	Common Clk	I/O	AN8	FSB1D[23]#	Source Sync	I/O
AM2	VSS	Power/Other		AN9	FSB1D[24]#	Source Sync	I/O
AM3	FSB1DRDY#	Common Clk	I/O	AN10	VSS	Power/Other	
AM4	FSB1DBSY#	Common Clk	I/O	AN11	FSB1D[27]#	Source Sync	I/O
AM5	VSS	Power/Other		AN12	FSB1D[25]#	Source Sync	I/O
AM6	FSB1D[20]#	Source Sync	I/O	AN13	VSS	Power/Other	
AM7	VSS	Power/Other		AN14	FSB1D[50]#	Source Sync	I/O
AM8	VSS	Power/Other		AN15	FSB1D[53]#	Source Sync	I/O
AM9	FSB1D[21]#	Source Sync	I/O	AN16	VSS	Power/Other	
AM10	FSB1DSTBP[1]#	Source Sync	I/O	AN17	CORECLKP	Analog	I
AM11	VSS	Power/Other		AN18	VTT	Power/Other	
AM12	FSB1D[26]#	Source Sync	I/O	AN19	VTT	Power/Other	
AM13	FSB1D[28]#	Source Sync	I/O	AN20	VTT	Power/Other	
AM14	VSS	Power/Other		AN21	VTT	Power/Other	
AM15	FSB1D[51]#	Source Sync	I/O	AN22	VSS	Power/Other	
AM16	FSB1D[49]#	Source Sync	I/O	AN23	RSVD		
AM17	VSS	Power/Other		AN24	FSB0A[4]#	Source Sync	I/O
AM18	VTT	Power/Other		AN25	VSS	Power/Other	
AM19	VTT	Power/Other		AN26	FSB0A[6]#	Source Sync	I/O
AM20	VTT	Power/Other		AN27	FSB0RSP#	Common Clk	O
AM21	VTT	Power/Other		AN28	VSS	Power/Other	
AM22	FSB0A[9]#	Source Sync	I/O	AN29	FSB0DP[3]#	Common Clk	I/O
AM23	VSS	Power/Other		AN30	FSB0RESET#	Common Clk	O
AM24	RSVD			AN31	VSS	Power/Other	
AM25	FSB0A[5]#	Source Sync	I/O	AN32	FSB0D[5]#	Source Sync	I/O
AM26	VSS	Power/Other		AN33	FSB0D[16]#	Source Sync	I/O
AM27	FSB0VREF	Power/Other		AN34	VSS	Power/Other	
AM28	FSB0BREQ0#	Common Clk	I/O	AN35	FSB0D[24]#	Source Sync	I/O
AM29	VSS	Power/Other		AN36	FSB0D[26]#	Source Sync	I/O
AM30	FSB0VREF	Power/Other		AN37	VSS	Power/Other	
AM31	VSS	Power/Other		AN38	VSS	Power/Other	
AM32	VSS	Power/Other		AP1	FSB1D[0]#	Source Sync	I/O
AM33	FSB0D[12]#	Source Sync	I/O	AP2	FSB1ADS#	Common Clk	I/O
AM34	FSB0D[29]#	Source Sync	I/O	AP3	VSS	Power/Other	
AM35	VSS	Power/Other		AP4	FSB1D[2]#	Source Sync	I/O
AM36	VSS	Power/Other		AP5	FSB1D[1]#	Source Sync	I/O
AM37	FSB0D[31]#	Source Sync	I/O	AP6	VSS	Power/Other	
AM38	FSB0D[28]#	Source Sync	I/O	AP7	FSB1DBI[0]#	Source Sync	I/O
AN1	VSS	Power/Other		AP8	FSB1D[22]#	Source Sync	I/O
AN2	FSB1BPM[4]#	Common Clk	I/O	AP9	VSS	Power/Other	
AN3	FSB1BPM[5]#	Common Clk	I/O	AP10	FSB1DSTBN[2]#	Source Sync	I/O
AN4	VSS	Power/Other		AP11	FSB1D[44]#	Source Sync	I/O
AN5	FSB1VREF	Power/Other		AP12	VSS	Power/Other	

Table 143. Intel® 5100 Memory Controller Hub Chipset Signals By Ball (Sheet 15 of 16)

Ball No.	Signal Name	Buffer Type	Direction	Ball No.	Signal Name	Buffer Type	Direction
AP13	FSB1D[43]#	Source Sync	I/O	AR20	VTT	Power/Other	
AP14	FSB1D[47]#	Source Sync	I/O	AR21	VTT	Power/Other	
AP15	VSS	Power/Other		AR22	FSB0A[33]#	Source Sync	I/O
AP16	FSB1D[55]#	Source Sync	I/O	AR23	VSS	Power/Other	
AP17	CORECLKN	Analog	I	AR24	FSB0A[20]#	Source Sync	I/O
AP18	VTT	Power/Other		AR25	FSB0A[26]#	Source Sync	I/O
AP19	VTT	Power/Other		AR26	VSS	Power/Other	
AP20	VTT	Power/Other		AR27	FSB0A[19]#	Source Sync	I/O
AP21	VTT	Power/Other		AR28	FSB0BPM[4]#	Common Clk	I/O
AP22	FSB0A[32]#	Source Sync	I/O	AR29	VSS	Power/Other	
AP23	FSB0ADSTB[1]#	Source Sync	I/O	AR30	FSB0DBSY#	Common Clk	I/O
AP24	VSS	Power/Other		AR31	FSB0DP[0]#	Common Clk	I/O
AP25	FSB0A[18]#	Source Sync	I/O	AR32	VSS	Power/Other	
AP26	FSB0A[7]#	Source Sync	I/O	AR33	RSVD		
AP27	VSS	Power/Other		AR34	FSBODTCRES	Analog	
AP28	VSS	Power/Other		AR35	VSS	Power/Other	
AP29	FSB0BPM[5]#	Common Clk	I/O	AR36	VSS	Power/Other	
AP30	VSS	Power/Other		AR37	FSB0DSTBN[1]#	Source Sync	I/O
AP31	FSB0DP[2]#	Common Clk	I/O	AR38	FSB0DSTBP[1]#	Source Sync	I/O
AP32	VSS	Power/Other		AT1	VSS	Power/Other	
AP33	VSS	Power/Other		AT2	FSB1D[6]#	Source Sync	I/O
AP34	FSB0D[20]#	Source Sync	I/O	AT3	FSB1D[7]#	Source Sync	I/O
AP35	FSB0D[18]#	Source Sync	I/O	AT4	VSS	Power/Other	
AP36	FSB0D[23]#	Source Sync	I/O	AT5	FSB1D[8]#	Source Sync	I/O
AP37	FSB0DBI[1]#	Source Sync	I/O	AT6	FSB1D[11]#	Source Sync	I/O
AP38	FSB0D[27]#	Source Sync	I/O	AT7	VSS	Power/Other	
AR1	FSB1D[3]#	Source Sync	I/O	AT8	FSB1D[36]#	Source Sync	I/O
AR2	VSS	Power/Other		AT9	FSB1D[34]#	Source Sync	I/O
AR3	FSB1D[4]#	Source Sync	I/O	AT10	VSS	Power/Other	
AR4	FSB1D[5]#	Source Sync	I/O	AT11	FSB1D[41]#	Source Sync	I/O
AR5	VSS	Power/Other		AT12	FSB1D[42]#	Source Sync	I/O
AR6	FSB1D[12]#	Source Sync	I/O	AT13	VSS	Power/Other	
AR7	FSB1D[14]#	Source Sync	I/O	AT14	FSB1VREF	Power/Other	
AR8	VSS	Power/Other		AT15	VSS	Power/Other	
AR9	FSB1D[35]#	Source Sync	I/O	AT16	VSS	Power/Other	
AR10	FSB1DSTBP[2]#	Source Sync	I/O	AT17	COREVCCA	Power/Other	
AR11	VSS	Power/Other		AT18	VTT	Power/Other	
AR12	FSB1D[46]#	Source Sync	I/O	AT19	VTT	Power/Other	
AR13	FSB1D[45]#	Source Sync	I/O	AT20	VTT	Power/Other	
AR14	VSS	Power/Other		AT21	VTT	Power/Other	
AR15	VSS	Power/Other		AT22	VSS	Power/Other	
AR16	VSS	Power/Other		AT23	FSB0A[30]#	Source Sync	I/O
AR17	VSS	Power/Other		AT24	FSB0A[27]#	Source Sync	I/O
AR18	VTT	Power/Other		AT25	VSS	Power/Other	
AR19	VTT	Power/Other		AT26	FSB0A[24]#	Source Sync	I/O

Table 143. Intel® 5100 Memory Controller Hub Chipset Signals By Ball (Sheet 16 of 16)

Ball No.	Signal Name	Buffer Type	Direction	Ball No.	Signal Name	Buffer Type	Direction
AT27	FSB0A[23]#	Source Sync	I/O	AU35	FSBSLWCRES	Analog	
AT28	VSS	Power/Other		AU36	FSB0D[21]#	Source Sync	I/O
AT29	FSB0DRDY#	Common Clk	I/O	AU37	VSS	Power/Other	
AT30	FSB0LOCK#	Common Clk	I/O	AV3	VSS	Power/Other	
AT31	VSS	Power/Other		AV4	FSB1D[10]#	Source Sync	I/O
AT32	FSB0TRDY#	Common Clk	O	AV5	VSS	Power/Other	
AT33	FSB0DP[1]#	Common Clk	I/O	AV6	FSB1D[9]#	Source Sync	I/O
AT34	VSS	Power/Other		AV7	FSB1D[32]#	Source Sync	I/O
AT35	FSBCRES	Analog		AV8	VSS	Power/Other	
AT36	FSB0D[19]#	Source Sync	I/O	AV9	FSB1D[37]#	Source Sync	I/O
AT37	FSB0D[22]#	Source Sync	I/O	AV10	FSB1D[39]#	Source Sync	I/O
AT38	VSS	Power/Other		AV11	VSS	Power/Other	
AU2	VSS	Power/Other		AV12	FSB1D[40]#	Source Sync	I/O
AU3	FSB1DSTBN[0]#	Source Sync	I/O	AV13	FSBSLWCTRL	CMOS	I
AU4	FSB1DSTBP[0]#	Source Sync	I/O	AV14	VSS	Power/Other	
AU5	FSB1D[15]#	Source Sync	I/O	AV15	VSS	Power/Other	
AU6	VSS	Power/Other		AV16	VSS	Power/Other	
AU7	FSB1D[13]#	Source Sync	I/O	AV17	VSS	Power/Other	
AU8	FSB1D[33]#	Source Sync	I/O	AV18	VTT	Power/Other	
AU9	VSS	Power/Other		AV19	VTT	Power/Other	
AU10	FSB1D[38]#	Source Sync	I/O	AV20	VTT	Power/Other	
AU11	FSB1DBI[2]#	Source Sync	I/O	AV21	VTT	Power/Other	
AU12	VSS	Power/Other		AV22	FSB0A[35]#	Source Sync	I/O
AU13	VSS	Power/Other		AV23	VSS	Power/Other	
AU14	48GB_Mode	CMOS		AV24	FSB0A[31]#	Source Sync	I/O
AU15	VSS	Power/Other		AV25	FSB0A[28]#	Source Sync	I/O
AU16	COREVSSA	Power/Other		AV26	VSS	Power/Other	
AU17	FSBVCCA	Power/Other		AV27	RSVD		
AU18	VTT	Power/Other		AV28	FSB0A[17]#	Source Sync	I/O
AU19	VTT	Power/Other		AV29	VSS	Power/Other	
AU20	VTT	Power/Other		AV30	FSB0BNR#	Common Clk	I/O
AU21	VTT	Power/Other		AV31	FSB0RS[0]#	Common Clk	O
AU22	FSB0A[34]#	Source Sync	I/O	AV32	VSS	Power/Other	
AU23	FSB0A[29]#	Source Sync	I/O	AV33	FSB0HITM#	Common Clk	I/O
AU24	VSS	Power/Other		AV34	FSB0DEFER#	Common Clk	O
AU25	FSB0A[22]#	Source Sync	I/O	AV35	VSS	Power/Other	
AU26	FSB0A[25]#	Source Sync	I/O	AV36	VSS	Power/Other	
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	O				
AU32	FSB0HIT#	Common Clk	I/O				
AU33	VSS	Power/Other					
AU34	FSB0BPRI#	Common Clk	O				

Table 144. Intel® 5100 Memory Controller Hub Chipset Signals By Name (Sheet 1 of 16)

Ball No.	Signal Name	Buffer Type	Direction	Ball No.	Signal Name	Buffer Type	Direction
AU14 AH17	48GB_Mode ASYNCRFSH	CMOS CMOS	I I	F29	CH0_DCLKN[3]/ CH0_CS[5]#	DDR2	O
L12	CFGSMBCLK	SMB	I/O	E28	CH0_DCLKP[0]	DDR2	O
G11	CFGSMBDATA	SMB	I/O	G26	CH0_DCLKP[1]	DDR2	O
C30	CH0_A[0]	DDR2	O	F26	CH0_DCLKP[2]	DDR2	O
D30	CH0_A[1]	DDR2	O	E29	CH0_DCLKP[3]/ CH0_CS[4]#	DDR2	O
A30	CH0_A[10]	DDR2	O	B14	CH0_DQ[0]	DDR2	I/O
J29	CH0_A[11]	DDR2	O	A14	CH0_DQ[1]	DDR2	I/O
H30	CH0_A[12]	DDR2	O	H17	CH0_DQ[10]	DDR2	I/O
A33	CH0_A[13]	DDR2	O	H18	CH0_DQ[11]	DDR2	I/O
N26	CH0_A[14]	DDR2	O	G14	CH0_DQ[12]	DDR2	I/O
L28	CH0_A[15]/CH0_ODT[4]	DDR2	O	F14	CH0_DQ[13]	DDR2	I/O
D31	CH0_A[2]	DDR2	O	J14	CH0_DQ[14]	DDR2	I/O
F30	CH0_A[3]	DDR2	O	H16	CH0_DQ[15]	DDR2	I/O
D32	CH0_A[4]	DDR2	O	A22	CH0_DQ[16]	DDR2	I/O
E31	CH0_A[5]	DDR2	O	B22	CH0_DQ[17]	DDR2	I/O
F32	CH0_A[6]	DDR2	O	D22	CH0_DQ[18]	DDR2	I/O
K28	CH0_A[7]	DDR2	O	D24	CH0_DQ[19]	DDR2	I/O
K27	CH0_A[8]	DDR2	O	D16	CH0_DQ[2]	DDR2	I/O
L27	CH0_A[9]	DDR2	O	C21	CH0_DQ[20]	DDR2	I/O
C31	CH0_BA[0]	DDR2	O	D21	CH0_DQ[21]	DDR2	I/O
B31	CH0_BA[1]	DDR2	O	A24	CH0_DQ[22]	DDR2	I/O
M27	CH0_BA[2]	DDR2	O	B24	CH0_DQ[23]	DDR2	I/O
B32	CH0_CAS#	DDR2	O	E24	CH0_DQ[24]	DDR2	I/O
H27	CH0_CB[0]	DDR2	I/O	E23	CH0_DQ[25]	DDR2	I/O
J26	CH0_CB[1]	DDR2	I/O	J20	CH0_DQ[26]	DDR2	I/O
L25	CH0_CB[2]	DDR2	I/O	L20	CH0_DQ[27]	DDR2	I/O
L26	CH0_CB[3]	DDR2	I/O	E21	CH0_DQ[28]	DDR2	I/O
J25	CH0_CB[4]	DDR2	I/O	H20	CH0_DQ[29]	DDR2	I/O
K24	CH0_CB[5]	DDR2	I/O	B17	CH0_DQ[3]	DDR2	I/O
J28	CH0_CB[6]	DDR2	I/O	H22	CH0_DQ[30]	DDR2	I/O
L24	CH0_CB[7]	DDR2	I/O	H21	CH0_DQ[31]	DDR2	I/O
J30	CH0_CKE[0]	DDR2	O	L33	CH0_DQ[32]	DDR2	I/O
E34	CH0_CKE[1]	DDR2	O	M33	CH0_DQ[33]	DDR2	I/O
G31	CH0_CKE[2]	DDR2	O	P34	CH0_DQ[34]	DDR2	I/O
G32	CH0_CKE[3] \CH0_ODT[5]	DDR2	O	M37	CH0_DQ[35]	DDR2	I/O
E25	CH0_CRES1	DDR2	I	K33	CH0_DQ[36]	DDR2	I/O
F25	CH0_CRES2	DDR2	I	K35	CH0_DQ[37]	DDR2	I/O
E30	CH0_CRESRET	DDR2	I	K38	CH0_DQ[38]	DDR2	I/O
D34	CH0_CS[0]#	DDR2	O	M35	CH0_DQ[39]	DDR2	I/O
C33	CH0_CS[1]#	DDR2	O	E14	CH0_DQ[4]	DDR2	I/O
B33	CH0_CS[2]#	DDR2	O	P35	CH0_DQ[40]	DDR2	I/O
B34	CH0_CS[3]#	DDR2	O	N38	CH0_DQ[41]	DDR2	I/O
F28	CH0_DCLKN[0]	DDR2	O	T37	CH0_DQ[42]	DDR2	I/O
G27	CH0_DCLKN[1]	DDR2	O	T38	CH0_DQ[43]	DDR2	I/O
F27	CH0_DCLKN[2]	DDR2	O	W30	CH0_DQ[44]	DDR2	I/O

Table 144. Intel® 5100 Memory Controller Hub Chipset Signals By Name (Sheet 2 of 16)

Ball No.	Signal Name	Buffer Type	Direction	Ball No.	Signal Name	Buffer Type	Direction
U34	CH0_DQ[45]	DDR2	I/O	B23	CH0_DQSP[11]	DDR2	I/O
R38	CH0_DQ[46]	DDR2	I/O	F23	CH0_DQSP[12]	DDR2	I/O
T35	CH0_DQ[47]	DDR2	I/O	M34	CH0_DQSP[13]	DDR2	I/O
U38	CH0_DQ[48]	DDR2	I/O	P36	CH0_DQSP[14]	DDR2	I/O
V36	CH0_DQ[49]	DDR2	I/O	V37	CH0_DQSP[15]	DDR2	I/O
C14	CH0_DQ[5]	DDR2	I/O	Y34	CH0_DQSP[16]	DDR2	I/O
Y36	CH0_DQ[50]	DDR2	I/O	H28	CH0_DQSP[17]	DDR2	I/O
W35	CH0_DQ[51]	DDR2	I/O	C22	CH0_DQSP[2]	DDR2	I/O
U37	CH0_DQ[52]	DDR2	I/O	G22	CH0_DQSP[3]	DDR2	I/O
V35	CH0_DQ[53]	DDR2	I/O	L37	CH0_DQSP[4]	DDR2	I/O
V33	CH0_DQ[54]	DDR2	I/O	R35	CH0_DQSP[5]	DDR2	I/O
W32	CH0_DQ[55]	DDR2	I/O	W37	CH0_DQSP[6]	DDR2	I/O
Y35	CH0_DQ[56]	DDR2	I/O	Y29	CH0_DQSP[7]	DDR2	I/O
Y31	CH0_DQ[57]	DDR2	I/O	K26	CH0_DQSP[8]	DDR2	I/O
Y27	CH0_DQ[58]	DDR2	I/O	E15	CH0_DQSP[9]	DDR2	I/O
AA27	CH0_DQ[59]	DDR2	I/O	H26	CH0_DRVCRES	DDR2	I
C16	CH0_DQ[6]	DDR2	I/O	A34	CH0_ODT[0]	DDR2	O
W34	CH0_DQ[60]	DDR2	I/O	E35	CH0_ODT[1]	DDR2	O
Y38	CH0_DQ[61]	DDR2	I/O	C35	CH0_ODT[2]	DDR2	O
AA28	CH0_DQ[62]	DDR2	I/O	A35	CH0_ODT[3]	DDR2	O
Y28	CH0_DQ[63]	DDR2	I/O	A32	CH0_RAS#	DDR2	O
A16	CH0_DQ[7]	DDR2	I/O	G28	CH0_SLEWCRES	DDR2	I
E16	CH0_DQ[8]	DDR2	I/O	C32	CH0_WE#	DDR2	O
F15	CH0_DQ[9]	DDR2	I/O	J32	CH1_A[0]	DDR2	O
A15	CH0_DQSN[0]	DDR2	I/O	E38	CH1_A[1]	DDR2	O
G16	CH0_DQSN[1]	DDR2	I/O	J31	CH1_A[10]	DDR2	O
H15	CH0_DQSN[10]	DDR2	I/O	D36	CH1_A[11]	DDR2	O
A23	CH0_DQSN[11]	DDR2	I/O	D35	CH1_A[12]	DDR2	O
F22	CH0_DQSN[12]	DDR2	I/O	M29	CH1_A[13]	DDR2	O
N34	CH0_DQSN[13]	DDR2	I/O	J34	CH1_A[14]	DDR2	O
P37	CH0_DQSN[14]	DDR2	I/O	G34	CH1_A[15]/CH1_ODT[4]	DDR2	O
V38	CH0_DQSN[15]	DDR2	I/O	D38	CH1_A[2]	DDR2	O
Y33	CH0_DQSN[16]	DDR2	I/O	F37	CH1_A[3]	DDR2	O
H29	CH0_DQSN[17]	DDR2	I/O	D37	CH1_A[4]	DDR2	O
C23	CH0_DQSN[2]	DDR2	I/O	C37	CH1_A[5]	DDR2	O
G23	CH0_DQSN[3]	DDR2	I/O	F35	CH1_A[6]	DDR2	O
K37	CH0_DQSN[4]	DDR2	I/O	C36	CH1_A[7]	DDR2	O
R36	CH0_DQSN[5]	DDR2	I/O	B36	CH1_A[8]	DDR2	O
W36	CH0_DQSN[6]	DDR2	I/O	E36	CH1_A[9]	DDR2	O
Y30	CH0_DQSN[7]	DDR2	I/O	K31	CH1_BA[0]	DDR2	O
K25	CH0_DQSN[8]	DDR2	I/O	M32	CH1_BA[1]	DDR2	O
D15	CH0_DQSN[9]	DDR2	I/O	F34	CH1_BA[2]	DDR2	O
B15	CH0_DQSP[0]	DDR2	I/O	M31	CH1_CAS#	DDR2	O
G17	CH0_DQSP[1]	DDR2	I/O	T27	CH1_CB[0]	DDR2	I/O
J15	CH0_DQSP[10]	DDR2	I/O	R27	CH1_CB[1]	DDR2	I/O

Table 144. Intel® 5100 Memory Controller Hub Chipset Signals By Name (Sheet 3 of 16)

Ball No.	Signal Name	Buffer Type	Direction	Ball No.	Signal Name	Buffer Type	Direction
T30	CH1_CB[2]	DDR2	I/O	D27	CH1_DQ[27]	DDR2	I/O
N29	CH1_CB[3]	DDR2	I/O	B26	CH1_DQ[28]	DDR2	I/O
V27	CH1_CB[4]	DDR2	I/O	C26	CH1_DQ[29]	DDR2	I/O
U27	CH1_CB[5]	DDR2	I/O	E19	CH1_DQ[3]	DDR2	I/O
P28	CH1_CB[6]	DDR2	I/O	A29	CH1_DQ[30]	DDR2	I/O
T29	CH1_CB[7]	DDR2	I/O	B29	CH1_DQ[31]	DDR2	I/O
F33	CH1_CKE[0]	DDR2	O	V28	CH1_DQ[32]	DDR2	I/O
G33	CH1_CKE[1]	DDR2	O	V29	CH1_DQ[33]	DDR2	I/O
H33	CH1_CKE[2]	DDR2	O	W27	CH1_DQ[34]	DDR2	I/O
H32	CH1_CKE[3]/CH1_ODT[5]	DDR2	O	W29	CH1_DQ[35]	DDR2	I/O
K34	CH1_CRES1	DDR2	I	R30	CH1_DQ[36]	DDR2	I/O
L34	CH1_CRES2	DDR2	I	P30	CH1_DQ[37]	DDR2	I/O
K36	CH1_CRESRET	DDR2	I	N33	CH1_DQ[38]	DDR2	I/O
H31	CH1_CS[0]#	DDR2	O	P33	CH1_DQ[39]	DDR2	I/O
L29	CH1_CS[1]#	DDR2	O	J16	CH1_DQ[4]	DDR2	I/O
K30	CH1_CS[2]#	DDR2	O	R34	CH1_DQ[40]	DDR2	I/O
M26	CH1_CS[3]#	DDR2	O	M38	CH1_DQ[41]	DDR2	I/O
G38	CH1_DCLKN[0]	DDR2	O	V31	CH1_DQ[42]	DDR2	I/O
J36	CH1_DCLKN[1]	DDR2	O	V32	CH1_DQ[43]	DDR2	I/O
G36	CH1_DCLKN[2]	DDR2	O	L38	CH1_DQ[44]	DDR2	I/O
H37	CH1_DCLKN[3]/ CH1_CS[5]#	DDR2	O	R33	CH1_DQ[45]	DDR2	I/O
F38	CH1_DCLKP[0]	DDR2	O	T34	CH1_DQ[46]	DDR2	I/O
J37	CH1_DCLKP[1]	DDR2	O	T33	CH1_DQ[47]	DDR2	I/O
G37	CH1_DCLKP[2]	DDR2	O	AC29	CH1_DQ[48]	DDR2	I/O
H38	CH1_DCLKP[3]/CH1_CS[4]#	DDR2	O	AA30	CH1_DQ[49]	DDR2	I/O
K17	CH1_DQ[0]	DDR2	I/O	J17	CH1_DQ[5]	DDR2	I/O
J18	CH1_DQ[1]	DDR2	I/O	AA33	CH1_DQ[50]	DDR2	I/O
C20	CH1_DQ[10]	DDR2	I/O	AB35	CH1_DQ[51]	DDR2	I/O
A20	CH1_DQ[11]	DDR2	I/O	AB28	CH1_DQ[52]	DDR2	I/O
B18	CH1_DQ[12]	DDR2	I/O	AB29	CH1_DQ[53]	DDR2	I/O
A18	CH1_DQ[13]	DDR2	I/O	AA35	CH1_DQ[54]	DDR2	I/O
E20	CH1_DQ[14]	DDR2	I/O	AA34	CH1_DQ[55]	DDR2	I/O
D20	CH1_DQ[15]	DDR2	I/O	AB36	CH1_DQ[56]	DDR2	I/O
L23	CH1_DQ[16]	DDR2	I/O	AB33	CH1_DQ[57]	DDR2	I/O
J22	CH1_DQ[17]	DDR2	I/O	AC34	CH1_DQ[58]	DDR2	I/O
C25	CH1_DQ[18]	DDR2	I/O	AC33	CH1_DQ[59]	DDR2	I/O
D25	CH1_DQ[19]	DDR2	I/O	F17	CH1_DQ[6]	DDR2	I/O
F18	CH1_DQ[2]	DDR2	I/O	AA37	CH1_DQ[60]	DDR2	I/O
K21	CH1_DQ[20]	DDR2	I/O	AA38	CH1_DQ[61]	DDR2	I/O
J21	CH1_DQ[21]	DDR2	I/O	AB32	CH1_DQ[62]	DDR2	I/O
F24	CH1_DQ[22]	DDR2	I/O	AC35	CH1_DQ[63]	DDR2	I/O
G24	CH1_DQ[23]	DDR2	I/O	D17	CH1_DQ[7]	DDR2	I/O
A27	CH1_DQ[24]	DDR2	I/O	C18	CH1_DQ[8]	DDR2	I/O
B27	CH1_DQ[25]	DDR2	I/O	E18	CH1_DQ[9]	DDR2	I/O
D26	CH1_DQ[26]	DDR2	I/O	G18	CH1_DQSN[0]	DDR2	I/O

Table 144. Intel® 5100 Memory Controller Hub Chipset Signals By Name (Sheet 4 of 16)

Ball No.	Signal Name	Buffer Type	Direction	Ball No.	Signal Name	Buffer Type	Direction
F19	CH1_DQSN[1]	DDR2	I/O	AT17	COREVCCA	Power/Other	
B19	CH1_DQSN[10]	DDR2	I/O	AU16	COREVSSA	Power/Other	
L22	CH1_DQSN[11]	DDR2	I/O	K10	ERR[0]#	CMOS	O
A28	CH1_DQSN[12]	DDR2	I/O	L13	ERR[1]#	CMOS	O
N32	CH1_DQSN[13]	DDR2	I/O	J12	ERR[2]#	CMOS	O
N37	CH1_DQSN[14]	DDR2	I/O	AG24	FSB0A[10]#	Source Sync	I/O
AA32	CH1_DQSN[15]	DDR2	I/O	AJ22	FSB0A[11]#	Source Sync	I/O
AB37	CH1_DQSN[16]	DDR2	I/O	AL22	FSB0A[12]#	Source Sync	I/O
R28	CH1_DQSN[17]	DDR2	I/O	AH22	FSB0A[13]#	Source Sync	I/O
J23	CH1_DQSN[2]	DDR2	I/O	AF25	FSB0A[14]#	Source Sync	I/O
C27	CH1_DQSN[3]	DDR2	I/O	AG23	FSB0A[15]#	Source Sync	I/O
P31	CH1_DQSN[4]	DDR2	I/O	AF22	FSB0A[16]#	Source Sync	I/O
U32	CH1_DQSN[5]	DDR2	I/O	AV28	FSB0A[17]#	Source Sync	I/O
AB30	CH1_DQSN[6]	DDR2	I/O	AP25	FSB0A[18]#	Source Sync	I/O
AC37	CH1_DQSN[7]	DDR2	I/O	AR27	FSB0A[19]#	Source Sync	I/O
U28	CH1_DQSN[8]	DDR2	I/O	AR24	FSB0A[20]#	Source Sync	I/O
J19	CH1_DQSN[9]	DDR2	I/O	AU28	FSB0A[21]#	Source Sync	I/O
G19	CH1_DQSP[0]	DDR2	I/O	AU25	FSB0A[22]#	Source Sync	I/O
F20	CH1_DQSP[1]	DDR2	I/O	AT27	FSB0A[23]#	Source Sync	I/O
A19	CH1_DQSP[10]	DDR2	I/O	AT26	FSB0A[24]#	Source Sync	I/O
L21	CH1_DQSP[11]	DDR2	I/O	AU26	FSB0A[25]#	Source Sync	I/O
B28	CH1_DQSP[12]	DDR2	I/O	AR25	FSB0A[26]#	Source Sync	I/O
P32	CH1_DQSP[13]	DDR2	I/O	AT24	FSB0A[27]#	Source Sync	I/O
N36	CH1_DQSP[14]	DDR2	I/O	AV25	FSB0A[28]#	Source Sync	I/O
AA31	CH1_DQSP[15]	DDR2	I/O	AU23	FSB0A[29]#	Source Sync	I/O
AB38	CH1_DQSP[16]	DDR2	I/O	AL25	FSB0A[3]#	Source Sync	I/O
T28	CH1_DQSP[17]	DDR2	I/O	AT23	FSB0A[30]#	Source Sync	I/O
K23	CH1_DQSP[2]	DDR2	I/O	AV24	FSB0A[31]#	Source Sync	I/O
C28	CH1_DQSP[3]	DDR2	I/O	AP22	FSB0A[32]#	Source Sync	I/O
R31	CH1_DQSP[4]	DDR2	I/O	AR22	FSB0A[33]#	Source Sync	I/O
U33	CH1_DQSP[5]	DDR2	I/O	AU22	FSB0A[34]#	Source Sync	I/O
AB31	CH1_DQSP[6]	DDR2	I/O	AV22	FSB0A[35]#	Source Sync	I/O
AC38	CH1_DQSP[7]	DDR2	I/O	AN24	FSB0A[4]#	Source Sync	I/O
U29	CH1_DQSP[8]	DDR2	I/O	AM25	FSB0A[5]#	Source Sync	I/O
K19	CH1_DQSP[9]	DDR2	I/O	AN26	FSB0A[6]#	Source Sync	I/O
H35	CH1_DRVCRES	DDR2	I	AP26	FSB0A[7]#	Source Sync	I/O
P29	CH1_ODT[0]	DDR2	O	AH23	FSB0A[8]#	Source Sync	I/O
N28	CH1_ODT[1]	DDR2	O	AM22	FSB0A[9]#	Source Sync	I/O
N27	CH1_ODT[2]	DDR2	O	AU29	FSB0ADS#	Common Clk	I/O
P27	CH1_ODT[3]	DDR2	O	AL23	FSB0ADSTB[0]#	Source Sync	I/O
M30	CH1_RAS#	DDR2	O	AP23	FSB0ADSTB[1]#	Source Sync	I/O
H36	CH1_SLEWCRES	DDR2	I	AH26	FSB0AP[0]#	Common Clk	I/O
L30	CH1_WE#	DDR2	O	AK26	FSB0AP[1]#	Common Clk	I/O
AP17	CORECLKN	Analog	I	AK27	FSB0BINIT#	Common Clk	I/O
AN17	CORECLKP	Analog	I	AV30	FSB0BNR#	Common Clk	I/O

Table 144. Intel® 5100 Memory Controller Hub Chipset Signals By Name (Sheet 5 of 16)

Ball No.	Signal Name	Buffer Type	Direction	Ball No.	Signal Name	Buffer Type	Direction
AR28	FSB0BPM[4]#	Common Clk	I/O	AF28	FSB0D[45]#	Source Sync	I/O
AP29	FSB0BPM[5]#	Common Clk	I/O	AD29	FSB0D[46]#	Source Sync	I/O
AU34	FSB0BPRI#	Common Clk	O	AE28	FSB0D[47]#	Source Sync	I/O
AM28	FSB0BREQ0#	Common Clk	I/O	AF37	FSB0D[48]#	Source Sync	I/O
AG26	FSB0BREQ1#	Common Clk	I/O	AJ34	FSB0D[49]#	Source Sync	I/O
AJ28	FSB0D[0]#	Source Sync	I/O	AN32	FSB0D[5]#	Source Sync	I/O
AL29	FSB0D[1]#	Source Sync	I/O	AL38	FSB0D[50]#	Source Sync	I/O
AJ30	FSB0D[10]#	Source Sync	I/O	AL36	FSB0D[51]#	Source Sync	I/O
AH31	FSB0D[11]#	Source Sync	I/O	AL37	FSB0D[52]#	Source Sync	I/O
AM33	FSB0D[12]#	Source Sync	I/O	AK36	FSB0D[53]#	Source Sync	I/O
AG29	FSB0D[13]#	Source Sync	I/O	AG35	FSB0D[54]#	Source Sync	I/O
AG27	FSB0D[14]#	Source Sync	I/O	AJ37	FSB0D[55]#	Source Sync	I/O
AJ33	FSB0D[15]#	Source Sync	I/O	AJ38	FSB0D[56]#	Source Sync	I/O
AN33	FSB0D[16]#	Source Sync	I/O	AH38	FSB0D[57]#	Source Sync	I/O
AL34	FSB0D[17]#	Source Sync	I/O	AE38	FSB0D[58]#	Source Sync	I/O
AP35	FSB0D[18]#	Source Sync	I/O	AF38	FSB0D[59]#	Source Sync	I/O
AT36	FSB0D[19]#	Source Sync	I/O	AK30	FSB0D[6]#	Source Sync	I/O
AH28	FSB0D[2]#	Source Sync	I/O	AG36	FSB0D[60]#	Source Sync	I/O
AP34	FSB0D[20]#	Source Sync	I/O	AH36	FSB0D[61]#	Source Sync	I/O
AU36	FSB0D[21]#	Source Sync	I/O	AE36	FSB0D[62]#	Source Sync	I/O
AT37	FSB0D[22]#	Source Sync	I/O	AE37	FSB0D[63]#	Source Sync	I/O
AP36	FSB0D[23]#	Source Sync	I/O	AL31	FSB0D[7]#	Source Sync	I/O
AN35	FSB0D[24]#	Source Sync	I/O	AJ31	FSB0D[8]#	Source Sync	I/O
AL35	FSB0D[25]#	Source Sync	I/O	AH32	FSB0D[9]#	Source Sync	I/O
AN36	FSB0D[26]#	Source Sync	I/O	AL32	FSB0DBI[0]#	Source Sync	I/O
AP38	FSB0D[27]#	Source Sync	I/O	AP37	FSB0DBI[1]#	Source Sync	I/O
AM38	FSB0D[28]#	Source Sync	I/O	AF30	FSB0DBI[2]#	Source Sync	I/O
AM34	FSB0D[29]#	Source Sync	I/O	AH37	FSB0DBI[3]#	Source Sync	I/O
AK29	FSB0D[3]#	Source Sync	I/O	AR30	FSB0DBSY#	Common Clk	I/O
AK35	FSB0D[30]#	Source Sync	I/O	AV34	FSB0DEFER#	Common Clk	O
AM37	FSB0D[31]#	Source Sync	I/O	AR31	FSB0DP[0]#	Common Clk	I/O
AG33	FSB0D[32]#	Source Sync	I/O	AT33	FSB0DP[1]#	Common Clk	I/O
AG30	FSB0D[33]#	Source Sync	I/O	AP31	FSB0DP[2]#	Common Clk	I/O
AE34	FSB0D[34]#	Source Sync	I/O	AN29	FSB0DP[3]#	Common Clk	I/O
AF31	FSB0D[35]#	Source Sync	I/O	AT29	FSB0DRDY#	Common Clk	I/O
AG32	FSB0D[36]#	Source Sync	I/O	AK32	FSB0DSTBN[0]#	Source Sync	I/O
AF33	FSB0D[37]#	Source Sync	I/O	AR37	FSB0DSTBN[1]#	Source Sync	I/O
AD35	FSB0D[38]#	Source Sync	I/O	AD32	FSB0DSTBN[2]#	Source Sync	I/O
AE32	FSB0D[39]#	Source Sync	I/O	AH34	FSB0DSTBN[3]#	Source Sync	I/O
AH29	FSB0D[4]#	Source Sync	I/O	AK33	FSB0DSTBP[0]#	Source Sync	I/O
AE31	FSB0D[40]#	Source Sync	I/O	AR38	FSB0DSTBP[1]#	Source Sync	I/O
AD30	FSB0D[41]#	Source Sync	I/O	AD33	FSB0DSTBP[2]#	Source Sync	I/O
AC30	FSB0D[42]#	Source Sync	I/O	AH35	FSB0DSTBP[3]#	Source Sync	I/O
AE29	FSB0D[43]#	Source Sync	I/O	AU32	FSB0HIT#	Common Clk	I/O
AC31	FSB0D[44]#	Source Sync	I/O	AV33	FSB0HITM#	Common Clk	I/O

Table 144. Intel® 5100 Memory Controller Hub Chipset Signals By Name (Sheet 6 of 16)

Ball No.	Signal Name	Buffer Type	Direction	Ball No.	Signal Name	Buffer Type	Direction
AT30	FSB0LOCK#	Common Clk	I/O	AH8	FSB1A[6]#	Source Sync	I/O
AJ27	FSB0MCERR#	Common Clk	I/O	AG8	FSB1A[7]#	Source Sync	I/O
AL26	FSB0REQ[0]#	Source Sync	I/O	AF7	FSB1A[8]#	Source Sync	I/O
AH25	FSB0REQ[1]#	Source Sync	I/O	AE7	FSB1A[9]#	Source Sync	I/O
AK24	FSB0REQ[2]#	Source Sync	I/O	AP2	FSB1ADS#	Common Clk	I/O
AJ24	FSB0REQ[3]#	Source Sync	I/O	AC10	FSB1ADSTB[0]#	Source Sync	I/O
AJ25	FSB0REQ[4]#	Source Sync	I/O	AG3	FSB1ADSTB[1]#	Source Sync	I/O
AN30	FSB0RESET#	Common Clk	O	AG12	FSB1AP[0]#	Common Clk	I/O
AV31	FSB0RS[0]#	Common Clk	O	AG10	FSB1AP[1]#	Common Clk	I/O
AL28	FSB0RS[1]#	Common Clk	O	AJ4	FSB1BINIT#	Common Clk	I/O
AU31	FSB0RS[2]#	Common Clk	O	AK3	FSB1BNR#	Common Clk	I/O
AN27	FSB0RSP#	Common Clk	O	AN2	FSB1BPM[4]#	Common Clk	I/O
AT32	FSB0TRDY#	Common Clk	O	AN3	FSB1BPM[5]#	Common Clk	I/O
AF34	FSB0VREF	Power/Other		AJ10	FSB1BPRI#	Common Clk	O
AM27	FSB0VREF	Power/Other		AL2	FSB1BREQ0#	Common Clk	I/O
AM30	FSB0VREF	Power/Other		AM1	FSB1BREQ1#	Common Clk	I/O
AB11	FSB1A[10]#	Source Sync	I/O	AP1	FSB1D[0]#	Source Sync	I/O
AF6	FSB1A[11]#	Source Sync	I/O	AP5	FSB1D[1]#	Source Sync	I/O
AD8	FSB1A[12]#	Source Sync	I/O	AV4	FSB1D[10]#	Source Sync	I/O
AC9	FSB1A[13]#	Source Sync	I/O	AT6	FSB1D[11]#	Source Sync	I/O
AB10	FSB1A[14]#	Source Sync	I/O	AR6	FSB1D[12]#	Source Sync	I/O
AC7	FSB1A[15]#	Source Sync	I/O	AU7	FSB1D[13]#	Source Sync	I/O
AA12	FSB1A[16]#	Source Sync	I/O	AR7	FSB1D[14]#	Source Sync	I/O
AF4	FSB1A[17]#	Source Sync	I/O	AU5	FSB1D[15]#	Source Sync	I/O
AG5	FSB1A[18]#	Source Sync	I/O	AL7	FSB1D[16]#	Source Sync	I/O
AH5	FSB1A[19]#	Source Sync	I/O	AL8	FSB1D[17]#	Source Sync	I/O
AD6	FSB1A[20]#	Source Sync	I/O	AN6	FSB1D[18]#	Source Sync	I/O
AE5	FSB1A[21]#	Source Sync	I/O	AK9	FSB1D[19]#	Source Sync	I/O
AC6	FSB1A[22]#	Source Sync	I/O	AP4	FSB1D[2]#	Source Sync	I/O
AD5	FSB1A[23]#	Source Sync	I/O	AM6	FSB1D[20]#	Source Sync	I/O
AH2	FSB1A[24]#	Source Sync	I/O	AM9	FSB1D[21]#	Source Sync	I/O
AH1	FSB1A[25]#	Source Sync	I/O	AP8	FSB1D[22]#	Source Sync	I/O
AJ3	FSB1A[26]#	Source Sync	I/O	AN8	FSB1D[23]#	Source Sync	I/O
AF1	FSB1A[27]#	Source Sync	I/O	AN9	FSB1D[24]#	Source Sync	I/O
AF3	FSB1A[28]#	Source Sync	I/O	AN12	FSB1D[25]#	Source Sync	I/O
AD3	FSB1A[29]#	Source Sync	I/O	AM12	FSB1D[26]#	Source Sync	I/O
AD12	FSB1A[3]#	Source Sync	I/O	AN11	FSB1D[27]#	Source Sync	I/O
AE4	FSB1A[30]#	Source Sync	I/O	AM13	FSB1D[28]#	Source Sync	I/O
AE2	FSB1A[31]#	Source Sync	I/O	AL11	FSB1D[29]#	Source Sync	I/O
AE1	FSB1A[32]#	Source Sync	I/O	AR1	FSB1D[3]#	Source Sync	I/O
AD2	FSB1A[33]#	Source Sync	I/O	AL13	FSB1D[30]#	Source Sync	I/O
AC4	FSB1A[34]#	Source Sync	I/O	AK12	FSB1D[31]#	Source Sync	I/O
AC3	FSB1A[35]#	Source Sync	I/O	AV7	FSB1D[32]#	Source Sync	I/O
AD9	FSB1A[4]#	Source Sync	I/O	AU8	FSB1D[33]#	Source Sync	I/O
AC12	FSB1A[5]#	Source Sync	I/O	AT9	FSB1D[34]#	Source Sync	I/O

Table 144. Intel® 5100 Memory Controller Hub Chipset Signals By Name (Sheet 7 of 16)

Ball No.	Signal Name	Buffer Type	Direction	Ball No.	Signal Name	Buffer Type	Direction
AR9	FSB1D[35]#	Source Sync	I/O	AM3	FSB1DRDY#	Common Clk	I/O
AT8	FSB1D[36]#	Source Sync	I/O	AU3	FSB1DSTBN[0]#	Source Sync	I/O
AV9	FSB1D[37]#	Source Sync	I/O	AL10	FSB1DSTBN[1]#	Source Sync	I/O
AU10	FSB1D[38]#	Source Sync	I/O	AP10	FSB1DSTBN[2]#	Source Sync	I/O
AV10	FSB1D[39]#	Source Sync	I/O	AK14	FSB1DSTBN[3]#	Source Sync	I/O
AR3	FSB1D[4]#	Source Sync	I/O	AU4	FSB1DSTBP[0]#	Source Sync	I/O
AV12	FSB1D[40]#	Source Sync	I/O	AM10	FSB1DSTBP[1]#	Source Sync	I/O
AT11	FSB1D[41]#	Source Sync	I/O	AR10	FSB1DSTBP[2]#	Source Sync	I/O
AT12	FSB1D[42]#	Source Sync	I/O	AK15	FSB1DSTBP[3]#	Source Sync	I/O
AP13	FSB1D[43]#	Source Sync	I/O	AK8	FSB1HIT#	Common Clk	I/O
AP11	FSB1D[44]#	Source Sync	I/O	AJ7	FSB1HITM#	Common Clk	I/O
AR13	FSB1D[45]#	Source Sync	I/O	AL4	FSB1LOCK#	Common Clk	I/O
AR12	FSB1D[46]#	Source Sync	I/O	AH11	FSB1MCERR#	Common Clk	I/O
AP14	FSB1D[47]#	Source Sync	I/O	AG9	FSB1REQ[0]#	Source Sync	I/O
AH14	FSB1D[48]#	Source Sync	I/O	AD11	FSB1REQ[1]#	Source Sync	I/O
AM16	FSB1D[49]#	Source Sync	I/O	AJ6	FSB1REQ[2]#	Source Sync	I/O
AR4	FSB1D[5]#	Source Sync	I/O	AF9	FSB1REQ[3]#	Source Sync	I/O
AN14	FSB1D[50]#	Source Sync	I/O	AE10	FSB1REQ[4]#	Source Sync	I/O
AM15	FSB1D[51]#	Source Sync	I/O	AE11	FSB1RESET#	Common Clk	O
AL14	FSB1D[52]#	Source Sync	I/O	AK5	FSB1RS[0]#	Common Clk	O
AN15	FSB1D[53]#	Source Sync	I/O	AL1	FSB1RS[1]#	Common Clk	O
AH16	FSB1D[54]#	Source Sync	I/O	AL5	FSB1RS[2]#	Common Clk	O
AP16	FSB1D[55]#	Source Sync	I/O	AK2	FSB1RSP#	Common Clk	O
AL16	FSB1D[56]#	Source Sync	I/O	AK6	FSB1TRDY#	Common Clk	O
AJ13	FSB1D[57]#	Source Sync	I/O	AH4	FSB1VREF	Power/Other	
AF15	FSB1D[58]#	Source Sync	I/O	AN5	FSB1VREF	Power/Other	
AG15	FSB1D[59]#	Source Sync	I/O	AT14	FSB1VREF	Power/Other	
AT2	FSB1D[6]#	Source Sync	I/O	AT35	FSBCRES	Analog	
AJ15	FSB1D[60]#	Source Sync	I/O	AR34	FSBODTCRES	Analog	
AJ16	FSB1D[61]#	Source Sync	I/O	AU35	FSBSLWCRES	Analog	
AG14	FSB1D[62]#	Source Sync	I/O	AV13	FSBSLWCTRL	CMOS	I
AF16	FSB1D[63]#	Source Sync	I/O	AU17	FSBVCCA	Power/Other	
AT3	FSB1D[7]#	Source Sync	I/O	M13	GPIOSMBCLK	SMB	I/O
AT5	FSB1D[8]#	Source Sync	I/O	H12	GPIOSMBDATA	SMB	I/O
AV6	FSB1D[9]#	Source Sync	I/O	Y9	PE0RN[0]	PEX	I
AP7	FSB1DBI[0]#	Source Sync	I/O	Y3	PE0RN[1]	PEX	I
AK11	FSB1DBI[1]#	Source Sync	I/O	AB7	PE0RN[2]	PEX	I
AU11	FSB1DBI[2]#	Source Sync	I/O	AA6	PE0RN[3]	PEX	I
AH13	FSB1DBI[3]#	Source Sync	I/O	Y10	PE0RP[0]	PEX	I
AM4	FSB1DBSY#	Common Clk	I/O	Y4	PE0RP[1]	PEX	I
AJ9	FSB1DEFER#	Common Clk	O	AB8	PE0RP[2]	PEX	I
AG11	FSB1DP[0]#	Common Clk	I/O	AA5	PE0RP[3]	PEX	I
AJ12	FSB1DP[1]#	Common Clk	I/O	Y6	PE0TN[0]	PEX	O
AF12	FSB1DP[2]#	Common Clk	I/O	AA2	PE0TN[1]	PEX	O
AF13	FSB1DP[3]#	Common Clk	I/O	AB5	PE0TN[2]	PEX	O

Table 144. Intel® 5100 Memory Controller Hub Chipset Signals By Name (Sheet 8 of 16)

Ball No.	Signal Name	Buffer Type	Direction	Ball No.	Signal Name	Buffer Type	Direction
AA9	PE0TN[3]	PEX	O	H9	PE4TN[0]	PEX	O
Y7	PE0TP[0]	PEX	O	E9	PE4TN[1]	PEX	O
AA3	PE0TP[1]	PEX	O	C8	PE4TN[2]	PEX	O
AB4	PE0TP[2]	PEX	O	E7	PE4TN[3]	PEX	O
AA8	PE0TP[3]	PEX	O	J9	PE4TP[0]	PEX	O
R5	PE2RN[0]	PEX	I	F9	PE4TP[1]	PEX	O
P4	PE2RN[1]	PEX	I	D8	PE4TP[2]	PEX	O
R3	PE2RN[2]	PEX	I	D7	PE4TP[3]	PEX	O
U1	PE2RN[3]	PEX	I	R8	PE5RN[0]	PEX	I
T5	PE2RP[0]	PEX	I	N7	PE5RN[1]	PEX	I
N4	PE2RP[1]	PEX	I	D4	PE5RN[2]	PEX	I
P3	PE2RP[2]	PEX	I	C2	PE5RN[3]	PEX	I
T1	PE2RP[3]	PEX	I	R9	PE5RP[0]	PEX	I
T7	PE2TN[0]	PEX	O	N8	PE5RP[1]	PEX	I
T4	PE2TN[1]	PEX	O	D5	PE5RP[2]	PEX	I
P1	PE2TN[2]	PEX	O	C3	PE5RP[3]	PEX	I
T2	PE2TN[3]	PEX	O	P9	PE5TN[0]	PEX	O
T8	PE2TP[0]	PEX	O	M8	PE5TN[1]	PEX	O
U4	PE2TP[1]	PEX	O	C5	PE5TN[2]	PEX	O
N1	PE2TP[2]	PEX	O	E3	PE5TN[3]	PEX	O
R2	PE2TP[3]	PEX	O	P10	PE5TP[0]	PEX	O
W2	PE3RN[0]	PEX	I	M9	PE5TP[1]	PEX	O
V6	PE3RN[1]	PEX	I	C6	PE5TP[2]	PEX	O
W8	PE3RN[2]	PEX	I	E4	PE5TP[3]	PEX	O
U10	PE3RN[3]	PEX	I	F2	PE6RN[0]	PEX	I
V2	PE3RP[0]	PEX	I	E1	PE6RN[1]	PEX	I
V5	PE3RP[1]	PEX	I	H4	PE6RN[2]	PEX	I
W7	PE3RP[2]	PEX	I	L4	PE6RN[3]	PEX	I
U9	PE3RP[3]	PEX	I	F3	PE6RP[0]	PEX	I
V3	PE3TN[0]	PEX	O	D1	PE6RP[1]	PEX	I
W4	PE3TN[1]	PEX	O	H3	PE6RP[2]	PEX	I
U7	PE3TN[2]	PEX	O	K4	PE6RP[3]	PEX	I
V9	PE3TN[3]	PEX	O	G4	PE6TN[0]	PEX	O
U3	PE3TP[0]	PEX	O	G1	PE6TN[1]	PEX	O
W5	PE3TP[1]	PEX	O	L7	PE6TN[2]	PEX	O
U6	PE3TP[2]	PEX	O	K5	PE6TN[3]	PEX	O
V8	PE3TP[3]	PEX	O	G5	PE6TP[0]	PEX	O
G10	PE4RN[0]	PEX	I	G2	PE6TP[1]	PEX	O
B9	PE4RN[1]	PEX	I	K7	PE6TP[2]	PEX	O
G8	PE4RN[2]	PEX	I	J5	PE6TP[3]	PEX	O
H7	PE4RN[3]	PEX	I	J6	PE7RN[0]	PEX	I
H10	PE4RP[0]	PEX	I	P6	PE7RN[1]	PEX	I
C9	PE4RP[1]	PEX	I	N5	PE7RN[2]	PEX	I
F8	PE4RP[2]	PEX	I	N2	PE7RN[3]	PEX	I
G7	PE4RP[3]	PEX	I	H6	PE7RP[0]	PEX	I

Table 144. Intel® 5100 Memory Controller Hub Chipset Signals By Name (Sheet 9 of 16)

Ball No.	Signal Name	Buffer Type	Direction	Ball No.	Signal Name	Buffer Type	Direction
P7	PE7RP[1]	PEX	I	A6	TMS	JTAG	I
M5	PE7RP[2]	PEX	I	B7	TRST#	JTAG	I
M2	PE7RP[3]	PEX	I	F13	V3REF	Analog	
F5	PE7TN[0]	PEX	O	L16	VCC	Power/Other	
K8	PE7TN[1]	PEX	O	L17	VCC	Power/Other	
M6	PE7TN[2]	PEX	O	L18	VCC	Power/Other	
M3	PE7TN[3]	PEX	O	L19	VCC	Power/Other	
F6	PE7TP[0]	PEX	O	M16	VCC	Power/Other	
J8	PE7TP[1]	PEX	O	M17	VCC	Power/Other	
L6	PE7TP[2]	PEX	O	M18	VCC	Power/Other	
L3	PE7TP[3]	PEX	O	N17	VCC	Power/Other	
K2	PECLKN	Analog		N19	VCC	Power/Other	
J2	PECLKP	Analog		P16	VCC	Power/Other	
R12	PEICOMPI	Analog		P18	VCC	Power/Other	
P12	PERCOMPO	Analog		P20	VCC	Power/Other	
K1	PEVCCA	Analog		P22	VCC	Power/Other	
R11	PEVCCBG	Analog		P24	VCC	Power/Other	
L1	PEVSSA	Analog		R15	VCC	Power/Other	
N11	PEVSSBG	Analog		R17	VCC	Power/Other	
AA11	PEWIDTH[0]	CMOS	I	R19	VCC	Power/Other	
Y12	PEWIDTH[1]	CMOS	I	R21	VCC	Power/Other	
W11	PEWIDTH[2]	CMOS	I	R23	VCC	Power/Other	
W10	PEWIDTH[3]	CMOS	I	T16	VCC	Power/Other	
AC1	PSEL[0]	CMOS	I	T18	VCC	Power/Other	
AB2	PSEL[1]	CMOS	I	T20	VCC	Power/Other	
AB1	PSEL[2]	CMOS	I	T22	VCC	Power/Other	
G12	PWRGOOD	CMOS	I	T24	VCC	Power/Other	
B6	RESETI#	CMOS	I	U15	VCC	Power/Other	
AE8	RSVD			U17	VCC	Power/Other	
AG2	RSVD			U19	VCC	Power/Other	
AH7	RSVD			U21	VCC	Power/Other	
AJ1	RSVD			U23	VCC	Power/Other	
AK17	RSVD			V16	VCC	Power/Other	
AK23	RSVD			V18	VCC	Power/Other	
AM24	RSVD			V20	VCC	Power/Other	
AN23	RSVD			V22	VCC	Power/Other	
AR33	RSVD			V24	VCC	Power/Other	
AV27	RSVD			W15	VCC	Power/Other	
A5	SPD0SMBCLK	SMB	I/O	W17	VCC	Power/Other	
L10	SPD0SMBDATA	SMB	I/O	W19	VCC	Power/Other	
A4	TCK	JTAG	I	W21	VCC	Power/Other	
B3	TDI	JTAG	I	W23	VCC	Power/Other	
J11	TDIOANODE	Analog		Y16	VCC	Power/Other	
K11	TDIOCATHODE	Analog		Y18	VCC	Power/Other	
B4	TDO	JTAG	O	Y20	VCC	Power/Other	

Table 144. Intel® 5100 Memory Controller Hub Chipset Signals By Name (Sheet 10 of 16)

Ball No.	Signal Name	Buffer Type	Direction	Ball No.	Signal Name	Buffer Type	Direction
Y22	VCC	Power/Other		T32	VCCDDR	Power/Other	
Y24	VCC	Power/Other		U25	VCCDDR	Power/Other	
AA13	VCC	Power/Other		U36	VCCDDR	Power/Other	
AA15	VCC	Power/Other		V25	VCCDDR	Power/Other	
AA17	VCC	Power/Other		V30	VCCDDR	Power/Other	
AA19	VCC	Power/Other		W25	VCCDDR	Power/Other	
AA21	VCC	Power/Other		Y25	VCCDDR	Power/Other	
AA23	VCC	Power/Other		AA25	VCCDDR	Power/Other	
AB13	VCC	Power/Other		AB25	VCCDDR	Power/Other	
AB14	VCC	Power/Other		AC36	VCCDDR	Power/Other	
AB16	VCC	Power/Other		E6	VCCPE	Power/Other	
AB18	VCC	Power/Other		J10	VCCPE	Power/Other	
AB20	VCC	Power/Other		L2	VCCPE	Power/Other	
AB22	VCC	Power/Other		L8	VCCPE	Power/Other	
AB24	VCC	Power/Other		L14	VCCPE	Power/Other	
AC15	VCC	Power/Other		L15	VCCPE	Power/Other	
AC17	VCC	Power/Other		M11	VCCPE	Power/Other	
AC19	VCC	Power/Other		M14	VCCPE	Power/Other	
AC21	VCC	Power/Other		M15	VCCPE	Power/Other	
AC23	VCC	Power/Other		N6	VCCPE	Power/Other	
AC25	VCC	Power/Other		N10	VCCPE	Power/Other	
AC26	VCC	Power/Other		N12	VCCPE	Power/Other	
AD26	VCC	Power/Other		N13	VCCPE	Power/Other	
AL17	VCC	Power/Other		N14	VCCPE	Power/Other	
A26	VCCDDR	Power/Other		N15	VCCPE	Power/Other	
B21	VCCDDR	Power/Other		P13	VCCPE	Power/Other	
D19	VCCDDR	Power/Other		P14	VCCPE	Power/Other	
D29	VCCDDR	Power/Other		R4	VCCPE	Power/Other	
E33	VCCDDR	Power/Other		R10	VCCPE	Power/Other	
G21	VCCDDR	Power/Other		R13	VCCPE	Power/Other	
G25	VCCDDR	Power/Other		R14	VCCPE	Power/Other	
G29	VCCDDR	Power/Other		T13	VCCPE	Power/Other	
L32	VCCDDR	Power/Other		T14	VCCPE	Power/Other	
L35	VCCDDR	Power/Other		U2	VCCPE	Power/Other	
L36	VCCDDR	Power/Other		U8	VCCPE	Power/Other	
N20	VCCDDR	Power/Other		U13	VCCPE	Power/Other	
N21	VCCDDR	Power/Other		U14	VCCPE	Power/Other	
N22	VCCDDR	Power/Other		V13	VCCPE	Power/Other	
N23	VCCDDR	Power/Other		V14	VCCPE	Power/Other	
N24	VCCDDR	Power/Other		W6	VCCPE	Power/Other	
N25	VCCDDR	Power/Other		W12	VCCPE	Power/Other	
N30	VCCDDR	Power/Other		W13	VCCPE	Power/Other	
P25	VCCDDR	Power/Other		W14	VCCPE	Power/Other	
R25	VCCDDR	Power/Other		Y13	VCCPE	Power/Other	
T25	VCCDDR	Power/Other		Y14	VCCPE	Power/Other	

Table 144. Intel® 5100 Memory Controller Hub Chipset Signals By Name (Sheet 11 of 16)

Ball No.	Signal Name	Buffer Type	Direction	Ball No.	Signal Name	Buffer Type	Direction
A3	VSS	Power/Other		E26	VSS	Power/Other	
A9	VSS	Power/Other		E27	VSS	Power/Other	
A13	VSS	Power/Other		E32	VSS	Power/Other	
A17	VSS	Power/Other		E37	VSS	Power/Other	
A21	VSS	Power/Other		F1	VSS	Power/Other	
A25	VSS	Power/Other		F4	VSS	Power/Other	
A31	VSS	Power/Other		F7	VSS	Power/Other	
A36	VSS	Power/Other		F10	VSS	Power/Other	
B2	VSS	Power/Other		F16	VSS	Power/Other	
B5	VSS	Power/Other		F21	VSS	Power/Other	
B8	VSS	Power/Other		F31	VSS	Power/Other	
B11	VSS	Power/Other		F36	VSS	Power/Other	
B16	VSS	Power/Other		G3	VSS	Power/Other	
B20	VSS	Power/Other		G6	VSS	Power/Other	
B25	VSS	Power/Other		G9	VSS	Power/Other	
B30	VSS	Power/Other		G15	VSS	Power/Other	
B35	VSS	Power/Other		G20	VSS	Power/Other	
B37	VSS	Power/Other		G30	VSS	Power/Other	
C1	VSS	Power/Other		G35	VSS	Power/Other	
C4	VSS	Power/Other		H1	VSS	Power/Other	
C7	VSS	Power/Other		H2	VSS	Power/Other	
C10	VSS	Power/Other		H5	VSS	Power/Other	
C15	VSS	Power/Other		H8	VSS	Power/Other	
C17	VSS	Power/Other		H11	VSS	Power/Other	
C19	VSS	Power/Other		H14	VSS	Power/Other	
C24	VSS	Power/Other		H19	VSS	Power/Other	
C29	VSS	Power/Other		H23	VSS	Power/Other	
C34	VSS	Power/Other		H24	VSS	Power/Other	
C38	VSS	Power/Other		H25	VSS	Power/Other	
D2	VSS	Power/Other		H34	VSS	Power/Other	
D3	VSS	Power/Other		J1	VSS	Power/Other	
D6	VSS	Power/Other		J3	VSS	Power/Other	
D9	VSS	Power/Other		J4	VSS	Power/Other	
D12	VSS	Power/Other		J7	VSS	Power/Other	
D14	VSS	Power/Other		J13	VSS	Power/Other	
D18	VSS	Power/Other		J24	VSS	Power/Other	
D23	VSS	Power/Other		J27	VSS	Power/Other	
D28	VSS	Power/Other		J33	VSS	Power/Other	
D33	VSS	Power/Other		J35	VSS	Power/Other	
E2	VSS	Power/Other		J38	VSS	Power/Other	
E5	VSS	Power/Other		K3	VSS	Power/Other	
E8	VSS	Power/Other		K6	VSS	Power/Other	
E13	VSS	Power/Other		K9	VSS	Power/Other	
E17	VSS	Power/Other		K12	VSS	Power/Other	
E22	VSS	Power/Other		K14	VSS	Power/Other	

Table 144. Intel® 5100 Memory Controller Hub Chipset Signals By Name (Sheet 12 of 16)

Ball No.	Signal Name	Buffer Type	Direction	Ball No.	Signal Name	Buffer Type	Direction
K15	VSS	Power/Other		R16	VSS	Power/Other	
K16	VSS	Power/Other		R18	VSS	Power/Other	
K18	VSS	Power/Other		R20	VSS	Power/Other	
K20	VSS	Power/Other		R22	VSS	Power/Other	
K22	VSS	Power/Other		R24	VSS	Power/Other	
K29	VSS	Power/Other		R26	VSS	Power/Other	
K32	VSS	Power/Other		R29	VSS	Power/Other	
L5	VSS	Power/Other		R32	VSS	Power/Other	
L9	VSS	Power/Other		R37	VSS	Power/Other	
L11	VSS	Power/Other		T3	VSS	Power/Other	
L31	VSS	Power/Other		T6	VSS	Power/Other	
M1	VSS	Power/Other		T9	VSS	Power/Other	
M4	VSS	Power/Other		T10	VSS	Power/Other	
M7	VSS	Power/Other		T11	VSS	Power/Other	
M10	VSS	Power/Other		T12	VSS	Power/Other	
M12	VSS	Power/Other		T15	VSS	Power/Other	
M19	VSS	Power/Other		T17	VSS	Power/Other	
M20	VSS	Power/Other		T19	VSS	Power/Other	
M21	VSS	Power/Other		T21	VSS	Power/Other	
M22	VSS	Power/Other		T23	VSS	Power/Other	
M23	VSS	Power/Other		T26	VSS	Power/Other	
M24	VSS	Power/Other		T31	VSS	Power/Other	
M25	VSS	Power/Other		T36	VSS	Power/Other	
M28	VSS	Power/Other		U5	VSS	Power/Other	
M36	VSS	Power/Other		U11	VSS	Power/Other	
N3	VSS	Power/Other		U12	VSS	Power/Other	
N9	VSS	Power/Other		U16	VSS	Power/Other	
N16	VSS	Power/Other		U18	VSS	Power/Other	
N18	VSS	Power/Other		U20	VSS	Power/Other	
N31	VSS	Power/Other		U22	VSS	Power/Other	
N35	VSS	Power/Other		U24	VSS	Power/Other	
P2	VSS	Power/Other		U26	VSS	Power/Other	
P5	VSS	Power/Other		U30	VSS	Power/Other	
P8	VSS	Power/Other		U31	VSS	Power/Other	
P11	VSS	Power/Other		U35	VSS	Power/Other	
P15	VSS	Power/Other		V1	VSS	Power/Other	
P17	VSS	Power/Other		V4	VSS	Power/Other	
P19	VSS	Power/Other		V7	VSS	Power/Other	
P21	VSS	Power/Other		V10	VSS	Power/Other	
P23	VSS	Power/Other		V11	VSS	Power/Other	
P26	VSS	Power/Other		V12	VSS	Power/Other	
P38	VSS	Power/Other		V15	VSS	Power/Other	
R1	VSS	Power/Other		V17	VSS	Power/Other	
R6	VSS	Power/Other		V19	VSS	Power/Other	
R7	VSS	Power/Other		V21	VSS	Power/Other	

Table 144. Intel® 5100 Memory Controller Hub Chipset Signals By Name (Sheet 13 of 16)

Ball No.	Signal Name	Buffer Type	Direction	Ball No.	Signal Name	Buffer Type	Direction
V23	VSS	Power/Other		AB12	VSS	Power/Other	
V26	VSS	Power/Other		AB15	VSS	Power/Other	
V34	VSS	Power/Other		AB17	VSS	Power/Other	
W1	VSS	Power/Other		AB19	VSS	Power/Other	
W3	VSS	Power/Other		AB21	VSS	Power/Other	
W9	VSS	Power/Other		AB23	VSS	Power/Other	
W16	VSS	Power/Other		AB26	VSS	Power/Other	
W18	VSS	Power/Other		AB27	VSS	Power/Other	
W20	VSS	Power/Other		AB34	VSS	Power/Other	
W22	VSS	Power/Other		AC2	VSS	Power/Other	
W24	VSS	Power/Other		AC5	VSS	Power/Other	
W26	VSS	Power/Other		AC8	VSS	Power/Other	
W28	VSS	Power/Other		AC11	VSS	Power/Other	
W31	VSS	Power/Other		AC16	VSS	Power/Other	
W33	VSS	Power/Other		AC18	VSS	Power/Other	
W38	VSS	Power/Other		AC20	VSS	Power/Other	
Y1	VSS	Power/Other		AC22	VSS	Power/Other	
Y2	VSS	Power/Other		AC24	VSS	Power/Other	
Y5	VSS	Power/Other		AC27	VSS	Power/Other	
Y8	VSS	Power/Other		AC28	VSS	Power/Other	
Y11	VSS	Power/Other		AC32	VSS	Power/Other	
Y15	VSS	Power/Other		AD1	VSS	Power/Other	
Y17	VSS	Power/Other		AD4	VSS	Power/Other	
Y19	VSS	Power/Other		AD7	VSS	Power/Other	
Y21	VSS	Power/Other		AD10	VSS	Power/Other	
Y23	VSS	Power/Other		AD27	VSS	Power/Other	
Y26	VSS	Power/Other		AD28	VSS	Power/Other	
Y32	VSS	Power/Other		AD31	VSS	Power/Other	
Y37	VSS	Power/Other		AD34	VSS	Power/Other	
AA1	VSS	Power/Other		AD36	VSS	Power/Other	
AA4	VSS	Power/Other		AD37	VSS	Power/Other	
AA7	VSS	Power/Other		AD38	VSS	Power/Other	
AA10	VSS	Power/Other		AE3	VSS	Power/Other	
AA14	VSS	Power/Other		AE6	VSS	Power/Other	
AA16	VSS	Power/Other		AE9	VSS	Power/Other	
AA18	VSS	Power/Other		AE12	VSS	Power/Other	
AA20	VSS	Power/Other		AE27	VSS	Power/Other	
AA22	VSS	Power/Other		AE30	VSS	Power/Other	
AA24	VSS	Power/Other		AE33	VSS	Power/Other	
AA26	VSS	Power/Other		AE35	VSS	Power/Other	
AA29	VSS	Power/Other		AF2	VSS	Power/Other	
AA36	VSS	Power/Other		AF5	VSS	Power/Other	
AB3	VSS	Power/Other		AF8	VSS	Power/Other	
AB6	VSS	Power/Other		AF10	VSS	Power/Other	
AB9	VSS	Power/Other		AF11	VSS	Power/Other	

Table 144. Intel® 5100 Memory Controller Hub Chipset Signals By Name (Sheet 14 of 16)

Ball No.	Signal Name	Buffer Type	Direction	Ball No.	Signal Name	Buffer Type	Direction
AF14	VSS	Power/Other		AJ36	VSS	Power/Other	
AF17	VSS	Power/Other		AK1	VSS	Power/Other	
AF23	VSS	Power/Other		AK4	VSS	Power/Other	
AF24	VSS	Power/Other		AK7	VSS	Power/Other	
AF26	VSS	Power/Other		AK10	VSS	Power/Other	
AF27	VSS	Power/Other		AK13	VSS	Power/Other	
AF29	VSS	Power/Other		AK16	VSS	Power/Other	
AF32	VSS	Power/Other		AK22	VSS	Power/Other	
AF35	VSS	Power/Other		AK25	VSS	Power/Other	
AF36	VSS	Power/Other		AK28	VSS	Power/Other	
AG1	VSS	Power/Other		AK31	VSS	Power/Other	
AG4	VSS	Power/Other		AK34	VSS	Power/Other	
AG6	VSS	Power/Other		AK37	VSS	Power/Other	
AG7	VSS	Power/Other		AK38	VSS	Power/Other	
AG13	VSS	Power/Other		AL3	VSS	Power/Other	
AG16	VSS	Power/Other		AL6	VSS	Power/Other	
AG17	VSS	Power/Other		AL9	VSS	Power/Other	
AG22	VSS	Power/Other		AL12	VSS	Power/Other	
AG25	VSS	Power/Other		AL15	VSS	Power/Other	
AG28	VSS	Power/Other		AL24	VSS	Power/Other	
AG31	VSS	Power/Other		AL27	VSS	Power/Other	
AG34	VSS	Power/Other		AL30	VSS	Power/Other	
AG37	VSS	Power/Other		AL33	VSS	Power/Other	
AG38	VSS	Power/Other		AM2	VSS	Power/Other	
AH3	VSS	Power/Other		AM5	VSS	Power/Other	
AH6	VSS	Power/Other		AM7	VSS	Power/Other	
AH9	VSS	Power/Other		AM8	VSS	Power/Other	
AH10	VSS	Power/Other		AM11	VSS	Power/Other	
AH12	VSS	Power/Other		AM14	VSS	Power/Other	
AH15	VSS	Power/Other		AM17	VSS	Power/Other	
AH24	VSS	Power/Other		AM23	VSS	Power/Other	
AH27	VSS	Power/Other		AM26	VSS	Power/Other	
AH30	VSS	Power/Other		AM29	VSS	Power/Other	
AH33	VSS	Power/Other		AM31	VSS	Power/Other	
AJ2	VSS	Power/Other		AM32	VSS	Power/Other	
AJ5	VSS	Power/Other		AM35	VSS	Power/Other	
AJ8	VSS	Power/Other		AM36	VSS	Power/Other	
AJ11	VSS	Power/Other		AN1	VSS	Power/Other	
AJ14	VSS	Power/Other		AN4	VSS	Power/Other	
AJ17	VSS	Power/Other		AN7	VSS	Power/Other	
AJ23	VSS	Power/Other		AN10	VSS	Power/Other	
AJ26	VSS	Power/Other		AN13	VSS	Power/Other	
AJ29	VSS	Power/Other		AN16	VSS	Power/Other	
AJ32	VSS	Power/Other		AN22	VSS	Power/Other	
AJ35	VSS	Power/Other		AN25	VSS	Power/Other	

Table 144. Intel® 5100 Memory Controller Hub Chipset Signals By Name (Sheet 15 of 16)

Ball No.	Signal Name	Buffer Type	Direction	Ball No.	Signal Name	Buffer Type	Direction
AN28	VSS	Power/Other		AU9	VSS	Power/Other	
AN31	VSS	Power/Other		AU12	VSS	Power/Other	
AN34	VSS	Power/Other		AU13	VSS	Power/Other	
AN37	VSS	Power/Other		AU15	VSS	Power/Other	
AN38	VSS	Power/Other		AU24	VSS	Power/Other	
AP3	VSS	Power/Other		AU27	VSS	Power/Other	
AP6	VSS	Power/Other		AU30	VSS	Power/Other	
AP9	VSS	Power/Other		AU33	VSS	Power/Other	
AP12	VSS	Power/Other		AU37	VSS	Power/Other	
AP15	VSS	Power/Other		AV3	VSS	Power/Other	
AP24	VSS	Power/Other		AV5	VSS	Power/Other	
AP27	VSS	Power/Other		AV8	VSS	Power/Other	
AP28	VSS	Power/Other		AV11	VSS	Power/Other	
AP30	VSS	Power/Other		AV14	VSS	Power/Other	
AP32	VSS	Power/Other		AV15	VSS	Power/Other	
AP33	VSS	Power/Other		AV16	VSS	Power/Other	
AR2	VSS	Power/Other		AV17	VSS	Power/Other	
AR5	VSS	Power/Other		AV23	VSS	Power/Other	
AR8	VSS	Power/Other		AV26	VSS	Power/Other	
AR11	VSS	Power/Other		AV29	VSS	Power/Other	
AR14	VSS	Power/Other		AV32	VSS	Power/Other	
AR15	VSS	Power/Other		AV35	VSS	Power/Other	
AR16	VSS	Power/Other		AV36	VSS	Power/Other	
AR17	VSS	Power/Other		AC13	VTT	Power/Other	
AR23	VSS	Power/Other		AC14	VTT	Power/Other	
AR26	VSS	Power/Other		AD13	VTT	Power/Other	
AR29	VSS	Power/Other		AD14	VTT	Power/Other	
AR32	VSS	Power/Other		AD15	VTT	Power/Other	
AR35	VSS	Power/Other		AD16	VTT	Power/Other	
AR36	VSS	Power/Other		AD17	VTT	Power/Other	
AT1	VSS	Power/Other		AD18	VTT	Power/Other	
AT4	VSS	Power/Other		AD19	VTT	Power/Other	
AT7	VSS	Power/Other		AD20	VTT	Power/Other	
AT10	VSS	Power/Other		AD21	VTT	Power/Other	
AT13	VSS	Power/Other		AD22	VTT	Power/Other	
AT15	VSS	Power/Other		AD23	VTT	Power/Other	
AT16	VSS	Power/Other		AD24	VTT	Power/Other	
AT22	VSS	Power/Other		AD25	VTT	Power/Other	
AT25	VSS	Power/Other		AE13	VTT	Power/Other	
AT28	VSS	Power/Other		AE14	VTT	Power/Other	
AT31	VSS	Power/Other		AE15	VTT	Power/Other	
AT34	VSS	Power/Other		AE16	VTT	Power/Other	
AT38	VSS	Power/Other		AE17	VTT	Power/Other	
AU2	VSS	Power/Other		AE18	VTT	Power/Other	
AU6	VSS	Power/Other					

Table 144. Intel® 5100 Memory Controller Hub Chipset Signals By Name (Sheet 16 of 16)

Ball No.	Signal Name	Buffer Type	Direction	Ball No.	Signal Name	Buffer Type	Direction
AE19	VTT	Power/Other		AR19	VTT	Power/Other	
AE20	VTT	Power/Other		AR20	VTT	Power/Other	
AE21	VTT	Power/Other		AR21	VTT	Power/Other	
AE22	VTT	Power/Other		AT18	VTT	Power/Other	
AE23	VTT	Power/Other		AT19	VTT	Power/Other	
AE24	VTT	Power/Other		AT20	VTT	Power/Other	
AE25	VTT	Power/Other		AT21	VTT	Power/Other	
AE26	VTT	Power/Other		AU18	VTT	Power/Other	
AF18	VTT	Power/Other		AU19	VTT	Power/Other	
AF19	VTT	Power/Other		AU20	VTT	Power/Other	
AF20	VTT	Power/Other		AU21	VTT	Power/Other	
AF21	VTT	Power/Other		AV18	VTT	Power/Other	
AG18	VTT	Power/Other		AV19	VTT	Power/Other	
AG19	VTT	Power/Other		AV20	VTT	Power/Other	
AG20	VTT	Power/Other		AV21	VTT	Power/Other	
AG21	VTT	Power/Other		G13	XDPCOMCRES	Analog	
AH18	VTT	Power/Other		E10	XDPD[0]#	XDP	I/O
AH19	VTT	Power/Other		A8	XDPD[1]#	XDP	I/O
AH20	VTT	Power/Other		B12	XDPD[10]#	XDP	I/O
AH21	VTT	Power/Other		A11	XDPD[11]#	XDP	I/O
AJ18	VTT	Power/Other		D13	XDPD[12]#	XDP	I/O
AJ19	VTT	Power/Other		A12	XDPD[13]#	XDP	I/O
AJ20	VTT	Power/Other		B13	XDPD[14]#	XDP	I/O
AJ21	VTT	Power/Other		C13	XDPD[15]#	XDP	I/O
AK18	VTT	Power/Other		A7	XDPD[2]#	XDP	I/O
AK19	VTT	Power/Other		D10	XDPD[3]#	XDP	I/O
AK20	VTT	Power/Other		F11	XDPD[4]#	XDP	I/O
AK21	VTT	Power/Other		E11	XDPD[5]#	XDP	I/O
AL18	VTT	Power/Other		E12	XDPD[6]#	XDP	I/O
AL19	VTT	Power/Other		B10	XDPD[7]#	XDP	I/O
AL20	VTT	Power/Other		A10	XDPD[8]#	XDP	I/O
AL21	VTT	Power/Other		C12	XDPD[9]#	XDP	I/O
AM18	VTT	Power/Other		D11	XDPDSTBN#	XDP	I/O
AM19	VTT	Power/Other		C11	XDPDSTBP#	XDP	I/O
AM20	VTT	Power/Other		K13	XDPODTCRES	Analog	
AM21	VTT	Power/Other		F12	XDPRDY#	XDP	I/O
AN18	VTT	Power/Other		H13	XDPSLWCRES	Analog	
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					

8.3 Package Information

See Section 1.2, "Related Documents and Materials" for RoHS compliance reference documents. For information on product marking and manufacturing steppings, please refer to the *Intel® 5100 Memory Controller Hub Chipset Specification Update*.

Figure 66. Bottom View

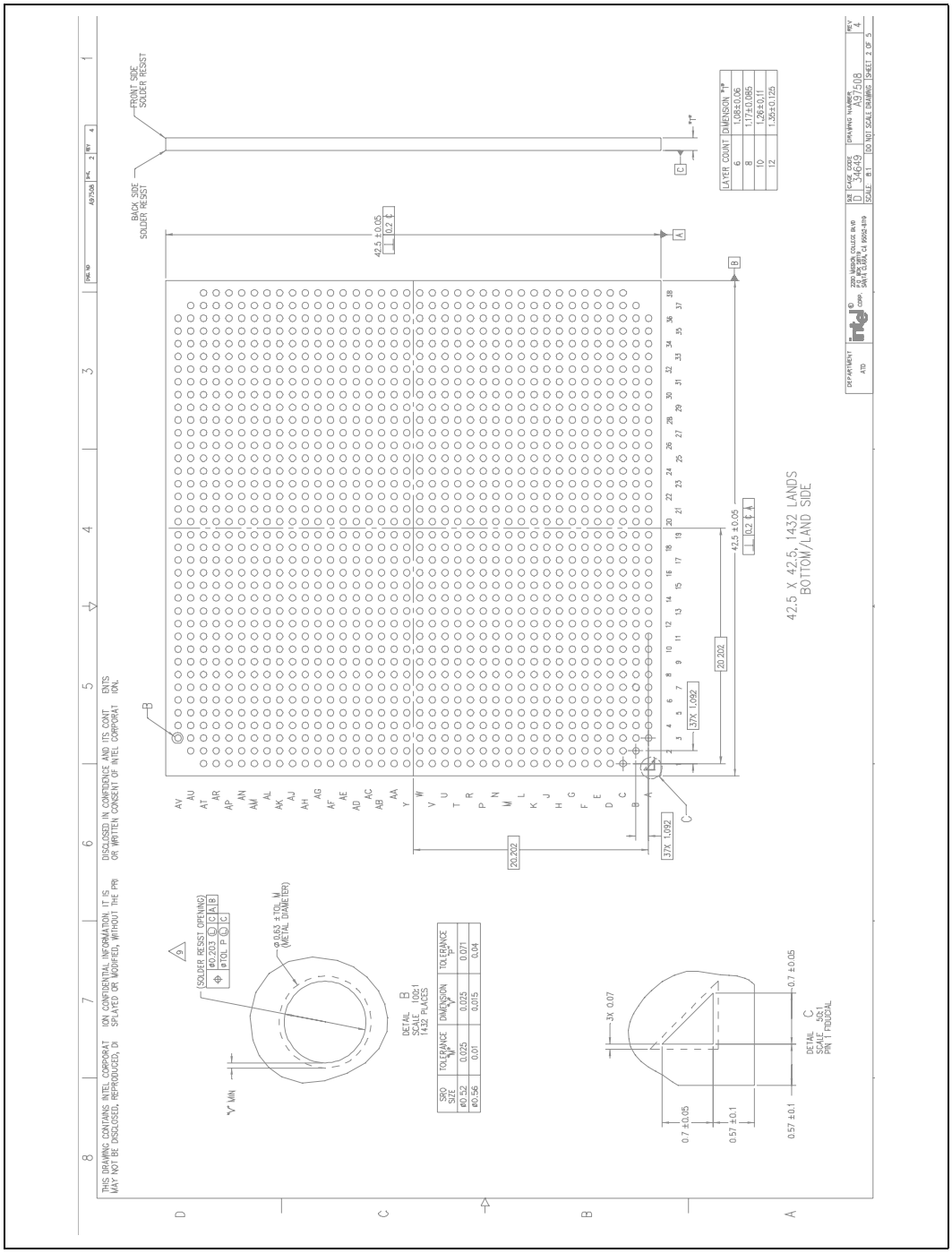


Figure 67. Top View

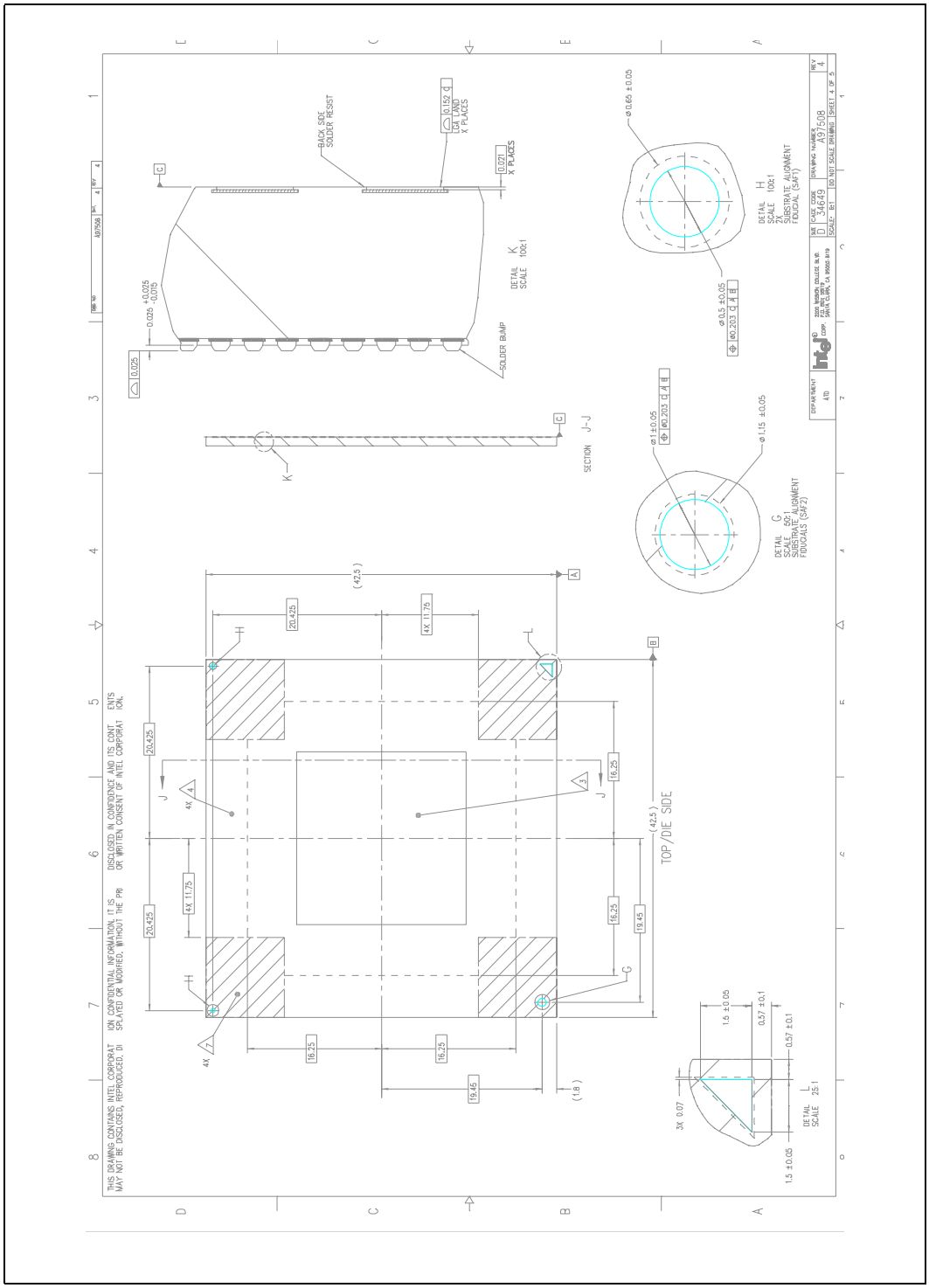


Figure 68. Bottom View with Package Height