Intel NetStructure® MPCBL0050 Single Board Computer

Technical Product Specification

September 2007

Order Number: 318146-001

INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL® PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN INTEL'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER, AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT. Intel products are not intended for use in medical, life saving, life sustaining, critical control or safety systems, or in nuclear facility applications.

Intel may make changes to specifications and product descriptions at any time, without notice.

Intel Corporation may have patents or pending patent applications, trademarks, copyrights, or other intellectual property rights that relate to the presented subject matter. The furnishing of documents and other materials and information does not provide any license, express or implied, by estoppel or otherwise, to any such patents, trademarks, copyrights, or other intellectual property rights.

The Intel NetStructure® MPCBL0050 Single Board Computer may contain design defects or errors known as errata which may cause the product to deviate from published specifications.

Contact your local Intel sales office or your distributor to obtain the latest specifications and before placing your product order.

Copies of documents which have an order number and are referenced in this document, or other Intel literature may be obtained by calling 1-800-548-4725 or by visiting Intel's website at http://www.intel.com.

Contents

Figures

Tables

Revision History

Date	Revision	Description
September 2007	001	Initial release of this document.

1.0 Introduction

1.1 Document Organization

This document gives technical specifications related to the Intel NetStructure® MPCBL0050 Single Board Computer (SBC). The MPCBL0050 is designed in accordance with the standards of the Advanced Telecommunications Compute Architecture (AdvancedTCA*) Design Guide for high availability, switched network computing. This document is intended for support during system product development and while sustaining a product. It specifies the architecture, design requirements, external requirements, board functionality, and design limitations of the MPCBL0050 SBC.

The following summarizes the focus of each section in this document.

Chapter 1.0, "Introduction," gives an overview of the information contained in the MPCBL0050 Technical Product Specification as well as a glossary of acronyms and important terms.

Chapter 2.0, "Feature Overview," introduces key features of the MPCBL0050.

Chapter 3.0, "Connectors and LEDs," includes an illustration of LEDs, connector locations, connector descriptions, and pinout tables.

Chapter 4.0, "Operating the Unit," provides specifics for configuring the MPCBL0050, including EFI configuration and jumper settings.

Chapter 5.0, "Hardware Management," provides a high-level overview of the IPMI implementation based on the PICMG* 3.0 and IPMI 2.0 specifications.

Chapter 6.0, "EFI BIOS Features," provides an introduction to the EFI, and the System Management EFI, stored in flash memory on the MPCBL0050 SBC.

Chapter 7.0, "EFI BIOS Setup," describes the interactive menu system of the EFI Setup program, which allows users to configure the EFI for a given system.

Chapter 8.0, "EFI BIOS Error Messages and Checkpoints," lists EFI error messages, Port 80h POST codes, and bus initialization checkpoints, and provides a brief description of each.

Chapter 9.0, "Serial Over LAN," describes the installation and configuration of Serial Over Lan (SOL), a specification for transmitting serial port data over an Ethernet connection, which allows the viewing of serial port data, thus providing a virtual remote terminal server for accessing a blade's serial port.

Chapter 10.0, "Firmware Update Utilities," describes how to use the board firmware utilities to update firmware on the board.

Chapter 11.0, "Specifications," contains the mechanical, environmental, and reliability specifications for the MPCBL0050.

Chapter 12.0, "Agency Information," Chapter 13.0, "Certifications," and Chapter 14.0, "Safety Warnings," document important safety precautions and describe regulatory requirements the MPCBL0050 is designed to meet.

Chapter 15.0, "Warranty Information," provides warranty information for Intel NetStructure® products.

Chapter 15.0, "Warranty Information," provides information on how to contact customer support.

Appendix A, "Supported IPMI Commands," lists the IPMI commands supported by the MPCBL0050.

Appendix B, "Reference Documents," lists related documentation.

1.2 Acronyms and Terms

Table 1. Acronyms and terms (Sheet 1 of 2)

Term	Definition
ACPI	Advanced Configuration and Power Interface
AdvancedMC	Advanced Mezzanine Card
AdvancedTCA	Advanced Telecommunications Compute Architecture
ASL	ACPI Source Language
Blade	An assembled PCB card that plugs into a chassis.
chassis ground	A chassis ground is also known as a shelf ground.
digital ground	A digital ground is also known as a logic ground.
DIMM	Dual Inline Memory Module
DMI	Desktop Management Interface
ECC	Error Correcting Code
EEPROM	Electrically Erasable Programmable Read-Only Memory
EFI BIOS	Extensible Firmware Interface Basic Input/Output Subsystem. ROM code that initializes the computer and performs some basic functions. In this document also referred as BIOS or EFI.
Fabric Board	A board capable of moving packet data between Node Boards via the ports of the backplane. This is sometimes referred to as a switch.
Fabric Slot	A slot supporting a link port connection to/from each Node Slot and/or out of the chassis.
FPGA	Field Programmable Gate Array
FRB	Fault Resilient Booting
FWH	Firmware Hub
GPIO	General Purpose I/O
Hyper-Threading Technology (HT Technology)	Allows a single (or dual) physical processor, to appear as two (or quad) logical processors to a HT Technology-aware operating system.
I2C	Inter-Integrated Circuit. A two-wire interface commonly used to carry management data.
IBA	Intel® Boot Agent. Software that allows your networked client computer to boot using a program code image supplied by a remote server.
ICH	I/O Controller Hub

Table 1. Acronyms and terms (Sheet 2 of 2)

Term	Definition
IDE	Integrated Device Electronics. A common, low-cost disk interface.
IPMB	Intelligent Platform Management Bus. A physical two-wire medium to carry IPMI information.
IPMC	Intelligent Platform Management Controller. A microcontroller on the baseboard responsible for low-level system management. Also referred to as BMC.
IPMI	Intelligent Platform Management Interface. A programming model for system management.
KCS	Keyboard Controller Style interface
logic ground	A digital ground is also known as a logic ground.
LPC Bus	Low Pin Count Bus. A legacy I/O bus that replaces ISA and X-bus. Refer to the Low Pin Count (LPC) Interface Specification.
MCH	Memory Controller Hub
MPCBL0050	A high-performance single board computer with an AdvancedMC slot.
MPRTM0040	A Rear Transition Module (RTM) that can be used with the MPCBL0050.
MPRTM0050	A Rear Transition Module (RTM) that can be used with the MPCBL0050. In addition to MPRTM0040 functionality, the MPRTM0050 provides Fibre Channel capability and rear access for MPCBL0050 Gigabit Ethernet ports.
MT/s	MegaTransfers per second - front side bus is quad pumped i.e. at one clock cycle 8 bytes are transferred, for 266MHz clock it results in 1066MT/s FSB.
MTBF	Mean Time Between Failure. A reliability measure based on the probability of failure.
NEBS	Network Equipment Building System. A set of telco standards for equipment emissions, thermal, shock, contaminants, and fire suppression requirements.
NMI	Non-Maskable Interrupt. A low-level PC interrupt.
Node Board	A board capable of providing and/or receiving packet data to/ from a Fabric Board via the ports of the networks. The term is used interchangeably with SBC in this document.
Node Slot	A slot supporting port connections to/from one or more Fabric slots. A Node slot is intended to accept a Node Board.
Physical Port	A port that physically exists. It is supported by one of many physical (PHY) type components.
POST	Power On Self Test
ROM	Read-Only Memory
RTM	Rear Transition Module. This term is used interchangeably with MPRTM0050 in this document.
SBC	Single Board Computer. This term is used interchangeably with Node Board and MPCBL0050 in this document.
SEL	System Event Log. Actions logged by the management controller.
shelf ground	A chassis ground is also known as a shelf ground.
SMBus	System Management Bus. Similar to I2C.
SMI	System Management Interrupt. A low-level PC interrupt which can be initiated by the chipset or management controller. Used to service IPMC or handle things like memory errors.
SMS	System Management Software or Standard Microsystems Corporation*
SOL	Serial Over LAN
USB	Universal Serial Bus. A general-purpose peripheral interconnect. USB 1.1 operates up to 12 Mbps. USB 2.0 operates up to 480 Mbps.

2.0 Feature Overview

2.1 Application

The AdvancedTCA* standards define open architecture modular computing components for carrier-grade, communications network infrastructure. The goals of the standards are to enable blade-based modular platforms to be:

- Cost effective
- High-density
- Highly available
- Scalable

These systems use a fabric I/O network for connecting multiple, independent processor boards, I/O nodes (for example, line cards), and I/O devices (for example, a storage subsystem).

2.2 MPCBL0050 Functional Description

This section describes the architecture of the MPCBL0050 SBC through functional block descriptions. Figure 1 shows the functional blocks of the MPCBL0050 SBC. The MPCBL0050 is a hot-swappable SBC with backplane connections to Gigabit Ethernet (GbE) ports on the base and fabric interface. The fabric interface of the MPCBL0050 board supports option 2 of the PICMG 3.1 specification.

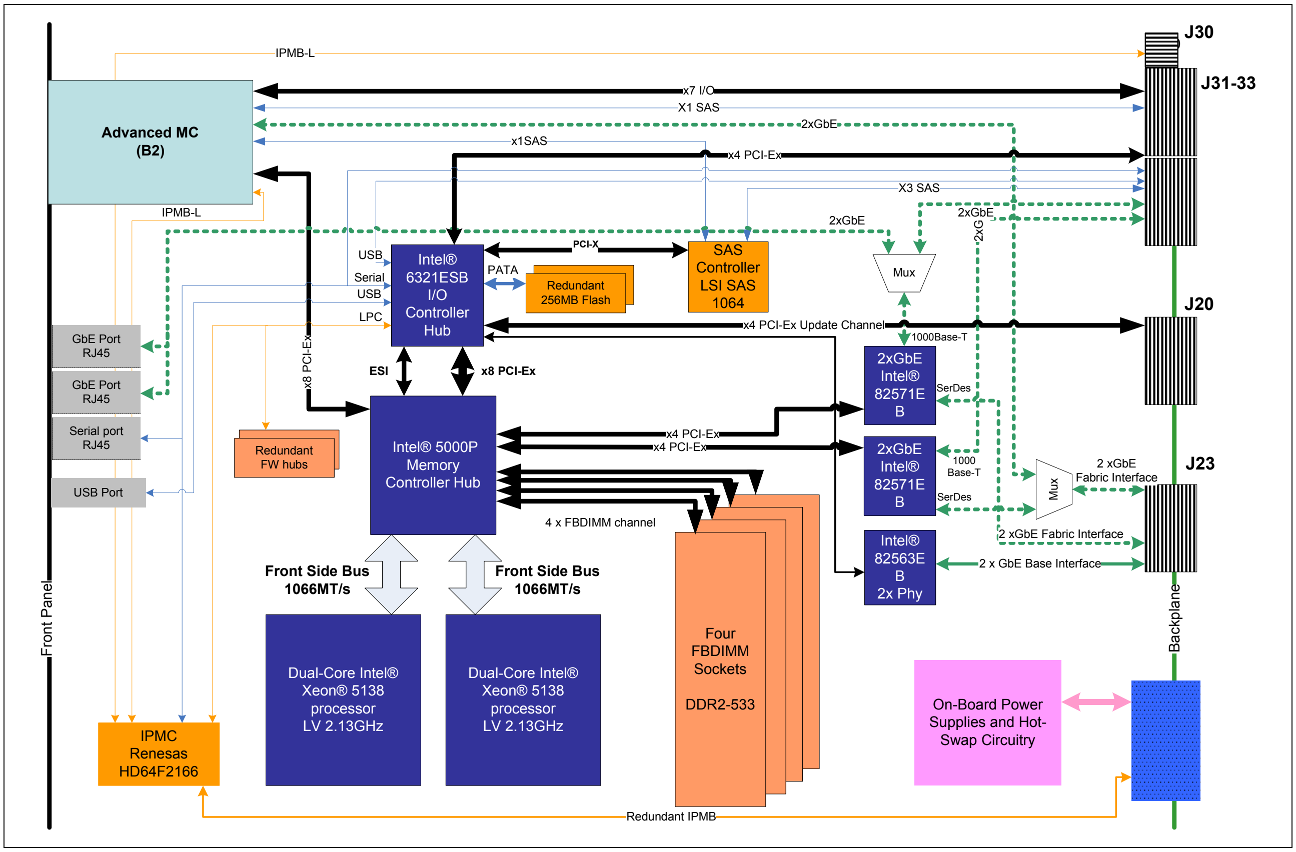
On the front panel, the MPCBL0050 offers an Mid-Size AdvancedMC* slot, one USB port, one serial console port and two Gigabit Ethernet ports.

For storage, the board itself supports two 256 MBytes of flash memory for user applications. Additional storage can be accessed via an on-board Serial Attached SCSI (SAS) controller providing SAS links to the RTM and AdvancedMC slot (See Section 2.2.6, "AdvancedMC Slot" for details).

The SBC incorporates an Intelligent Platform Management Controller (IPMC) that monitors critical hardware functionality of the board such as temperature and voltage, responds to commands from the shelf manager, and reports events.

Power is supplied to the MPCBL0050 SBC through two redundant -48 V power supply connections.

Figure 1. MPCBL0050 SBC block diagram



2.2.1 Rear Transition Module

The MPCBL0050 board supports Rear Transition Modules (RTMs). There are 2 RTMs available from Intel:

- Intel NetStructure® MPRTM0040
- Intel NetStructure® MPRTM0050

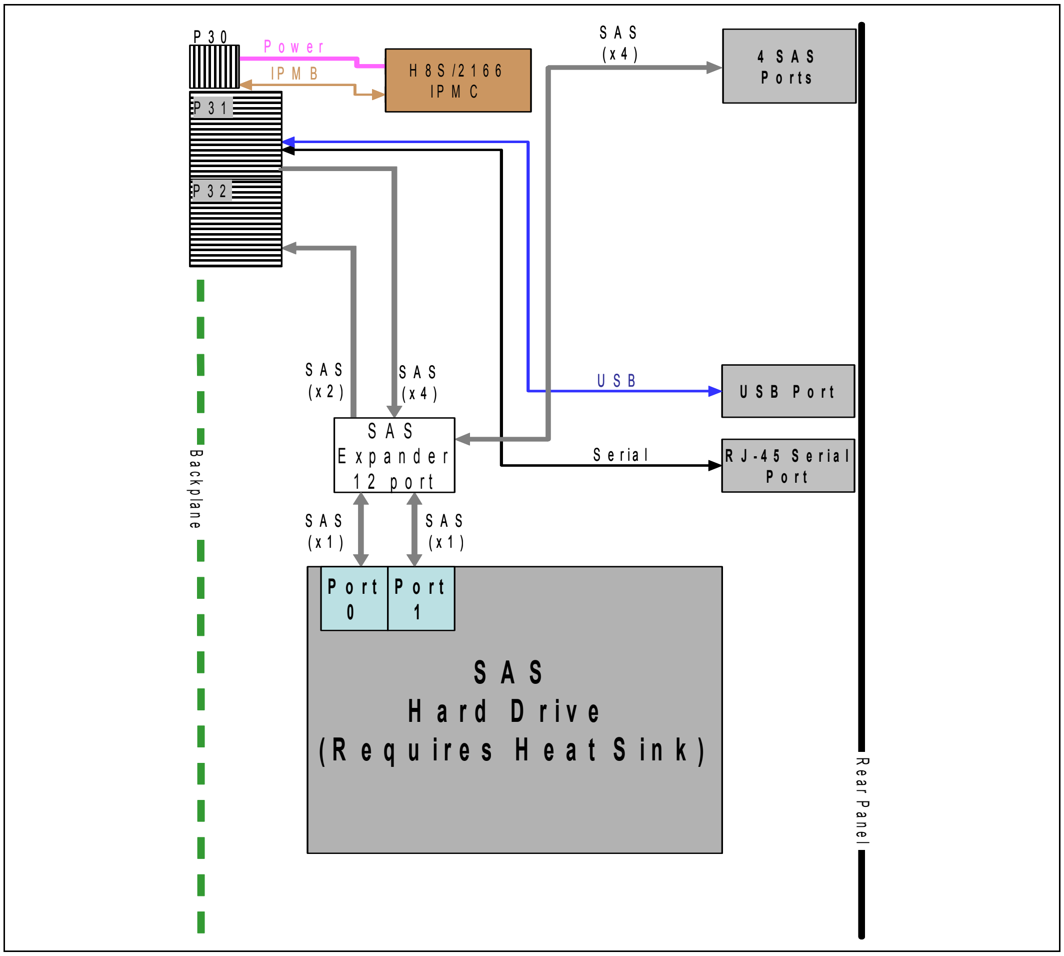
2.2.1.1 Intel NetStructure® MPRTM0040

The Intel NetStructure® MPRTM0040 Rear Transition Module (RTM) supports the following interfaces:

- One USB port
- One Serial console interface
- Four Serial Attached SCSI (SAS) ports for remote storage connectivity
- One on-board SAS hard drive (SAS HDD not included).

Figure 2 shows a block diagram of the MPRTM0040.

Figure 2. MPRTM0040 RTM block diagram



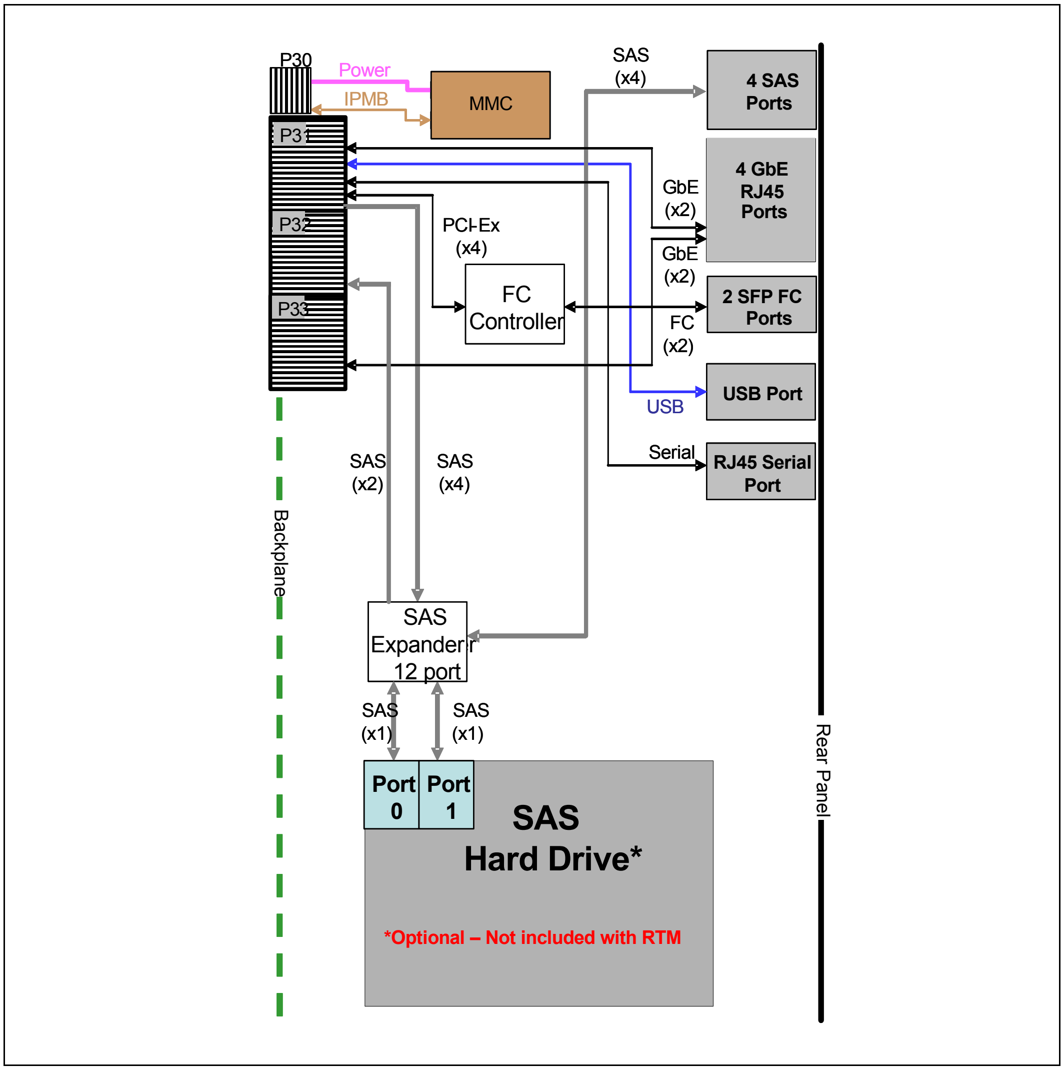
2.2.1.2 Intel NetStructure® MPRTM0050

The Intel NetStructure® MPRTM0050 Rear Transition Module (RTM) supports the following interfaces:

- One USB port
- One Serial console interface
- Four Serial Attached SCSI (SAS) ports for remote storage connectivity
- One on-board SAS hard drive (SAS HDD not included)
- Two Fiber Channel 1/2/4 Gb/s ports (SFP not included)
- Four GbE ports (available only if MPCBL0050 GbE ports are redirected to the RTM)

Figure 3 shows a block diagram of the MPRTM0050.

Figure 3. MPRTM0050 RTM block diagram



Note: The MPCBL0050 SBC provides only x3 SAS link from SAS controller to the RTM connector.

Note: Using a hard drive on the RTM in some environments may require the use of a heat sink in order to improve heat dissipation. The heat sink is still under development. Contact your Intel representative for further details.

Note: For detailed information about a Rear Transition Module, please refer to the Technical Product Specification for that product. Documents are available at http://www.intel.com/design/telecom/products/cbp/atca/mpcbl0050/techdocs.htm

2.2.2 Dual-Core Intel® Xeon® 5138 LV 2.13GHz Processor

The MPCBL0050 SBC supports two Dual-Core Intel® Xeon® 5138 LV 2.13 GHz processors with 1066 MHz front side bus with the following benefits:

- Dual processor support, power-optimized 1066 MT/s front-side bus (FSB), and a 4MB shared L2 cache per processor, that enables up to four high-performance cores per platform
- EM64T technology allowing to expand memory addressing space to 64bit.
- FSB address, data parity, and an enhanced error reporting mechanism through the MCA (Machine Check Architecture) that ensures reliability and data integrity

For further details, refer to the *Dual-Core Intel® Xeon® 5138 LV 2.13GHz processor processor Datasheet* available at http://www.intel.com.

2.2.3 Chipset

The MPCBL0050 uses the Intel® 5000P chipset which comprises the following major components:

- Intel® 5000P Memory Controller Hub (MCH)
- Intel® 6321ESB I/O Controller (ICH)
- Intel® 82571EB Gigabit Ethernet Controller

Although a brief overview is provided in this document, detailed component information can be found in the documentation for the respective devices. Please refer to the Intel web page: http://www.intel.com

2.2.3.1 Intel® 5000 Memory Controller Hub

The architecture of the Intel® 5000P Memory Controller Hub (MCH) provides the performance and feature set required for servers, with configuration options facilitating optimization of the platform for workloads characteristic of communication, presentation, storage, performance computation, or database applications. To accomplish this, the MCH has numerous RASUM (Reliability, Availability, Serviceability, Usability, and Manageability) features on multiple interfaces.

The Intel® 5000P chipset is designed for use in server systems based on the processor Dual-Core Intel Xeon 5000 Sequence processor. The Intel 5000P chipset supports two processors on dual independent point to point system buses operating at 266 MHz (1066 MTS). The theoretical bandwidth of the two processor busses is 21 GB/s for Dual-Core Intel Xeon 5100 series. The MCH supports 36 bit addressability for a total 64 GB of physical memory.

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

The MCH provides four channels of Fully Buffered DIMM (FB-DIMM) memory. Each channel can support up to 4 Dual Ranked FB-DIMM DDR2 DIMMs. FB-DIMM memory channels are organized into two branches for support of RAID 1 (mirroring). On the MPCBL0050, the maximum theoretical bandwidth between MCH and FB-DIMMs is 1GB/s per channel for DDR2 533 FB-DIMMs. With four FB-DIMM slots available, each connected to a separate channel, the total bandwidth available is 4GB/s.

The Intel® 5000P is compatible with the PCI Express* Interface Specification, Rev 1.0a. The MCH provides six x4 PCI Express interfaces which can be combined to three x8 ports. The MCH is a root class component as defined in the PCI Express Interface Specification, Rev1.0a.

The MCH interfaces with the Intel® 6321ESB ICH via a dedicated Enterprise South Bridge Interface (ESI) port together with x8 PCI Express port, providing appropriate bandwidth for I/O interfaces connected to to the ICH.

Table 2. PCI Express port mapping

Port	Function
Port 0 (ESI)	Enterprise South Bridge Interface (ESI) connects to Intel® 6321ESB ICH
Port 2,3	Connects to Intel® 6321ESB ICH
Port 4	Connects to Fabric Interface (Port 0, Channel 1 & 2) Gigabit Ethernet Controller 82571
Port 5	Connects to Fabric Interface (Port 1, Channel 1 & 2) Gigabit Ethernet Controller 82571
Port 6,7	Connects to AdvancedMC slot B2

2.2.3.2 Intel® 6321ESB I/O Controller Hub

The Intel® 6321ESB I/O Controller Hub component integrates bridge functionality for PCI Express, PCI-X, conventional PCI, LPC, USB, SATA, IDE and SMBus, and dual-Gigabit Ethernet MAC components, as well as numerous board management functions. It provides for all system I/O, allowing for simpler system board architectures and smaller board areas than if discrete components were used.

The 6321ESB integrates an Ultra ATA 100 controller, six Serial ATA host controller ports, one EHCI host controller, and four UHCI host controllers supporting eight external USB 2.0 ports, LPC interface controller, and flash.

The 6321ESB component provides the data buffering and interface arbitration required to ensure that system interfaces operate efficiently and provide the bandwidth necessary to enable the system to obtain peak performance.

The 6321ESB also contains two fully integrated Gigabit Ethernet Media Access Control (MAC). This provides a standard IEEE 802.3 Ethernet interface for 1000BASE-T, 1000BASE-X, 100BASE-TX, and 10BASE-T applications. Each port contains a Kumeran interface for connecting the ICH to the Intel® 82563EB 2x PHY Device.

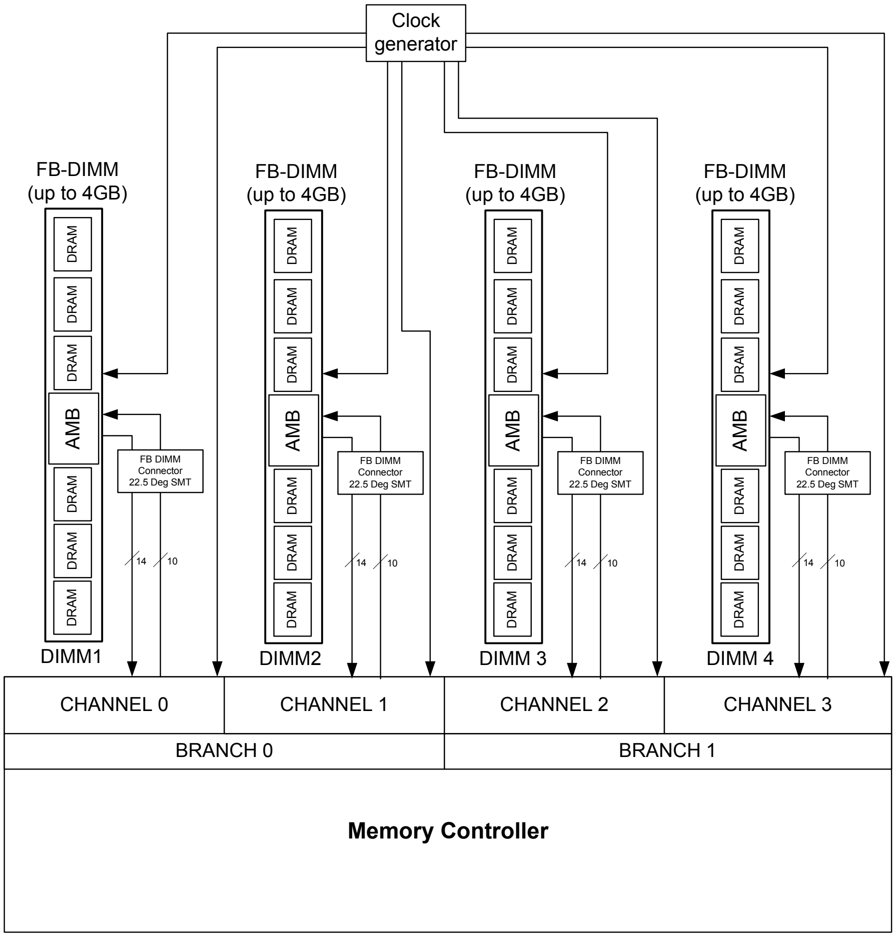
2.2.3.3 Intel® 82571EB Gigabit Ethernet Controller

The Intel® 82571EB Gigabit Ethernet Controller is a single, compact component with two fully integrated Gigabit Ethernet Media Access Control (MAC) and physical layer (PHY) ports. This device uses the PCI Express* architecture (Rev. 1.0a). The 82571EB provides two IEEE 802.3* Ethernet interfaces for 1000BASE-T, 100BASE-TX, and 10BASE-T applications. Both ports also integrate a Serializer-Deserializer (SerDes) to support Gigabit backplane applications. In addition to managing MAC and PHY Ethernet layer functions, the controller manages PCI Express packet traffic across its transaction, link, and physical/logical layers.

2.2.4 Memory

The MPCBL0050 MCH supports four Fully-Buffered DIMM (FB-DIMM) memory channels. FB-DIMM memory utilizes a narrow high-speed frame-oriented interface referred to as a channel. The four FB-DIMM channels are organized into two branches of two channels per branch. Each branch is supported by a separate Memory Controller (MC). The two channels on each branch operate in lockstep to increase FB-DIMM bandwidth. A branch transfers 16 bytes of payload/frame on Southbound lanes and 32 bytes of payload/ frame on Northbound lanes. Each FB-DIMM module, in addition to DRAM chips, contains an Advanced Memory Buffer (AMB) device that is responsible for communication between an FBDIMM memory module and MCH over the FBDIMM channel. See Figure 4 for details.

Figure 4. Fully Buffered DIMM channels



The key features of the MPCBL0050 FB-DIMM memory interface are:

- Four Fully Buffered DDR (FB-DIMM) memory channels.
- Branch channels paired together in lockstep to match FSB bandwidth requirement.
- Support for up to 4 dual-ranked FB-DDR2 4GB DIMMs (16GB of physical memory)

On the MPCBL0050, each FBDIMM channel provides 1 GB/s throughput. Full memory bandwidth of 4 GB/s can be achieved only by populating all four memory slots.

Table 3. DIMM memory features

Feature	Parameter
DIMM slots	Four
Rank Structure	Single or Dual
Max DIMM speed	533MHz (note that 667Mhz modules can be used with the MPCBL0050, but they will still operate at 533Mhz)
Device width	x4 or x8

Table 4. Supported memory configurations

Total Memory	U28 (DIMM1) Branch0 Channel0	U29 (DIMM2) Branch0 Channel1	U30 (DIMM3) Branch1 Channel2	U31 (DIMM4) Branch1 Channel3
4 GBytes	1 GByte FBDIMM DDR2	1 GByte FBDIMM DDR2	1 GByte FBDIMM DDR2	1 GByte FBDIMM DDR2
8 GBytes	2 GByte FBDIMM DDR2	2 GByte FBDIMM DDR2	2 GByte FBDIMM DDR2	2 GByte FBDIMM DDR2
16 GBytes	4 GByte FBDIMM DDR2	4 GByte FBDIMM DDR2	4 GByte FBDIMM DDR2	4 GByte FBDIMM DDR2

Refer to Section 4.5 for details on the memory installation procedure.

DIMM installation considerations:

- DIMMs must be installed in matching pairs.
- DIMMs must be identical in rank, size, device width, and memory timing.
- If only two DIMMs are used, they must be installed in slots **U28** and **U29** only. With only 2 DIMMs, only half the memory bandwidth can be achieved (2GB/s instead of 4GB/s).

Refer to the MPCBL0050 Compatibility Report for a list of DIMMs validated for the board.

2.2.5 I/O

2.2.5.1 Gigabit Ethernet

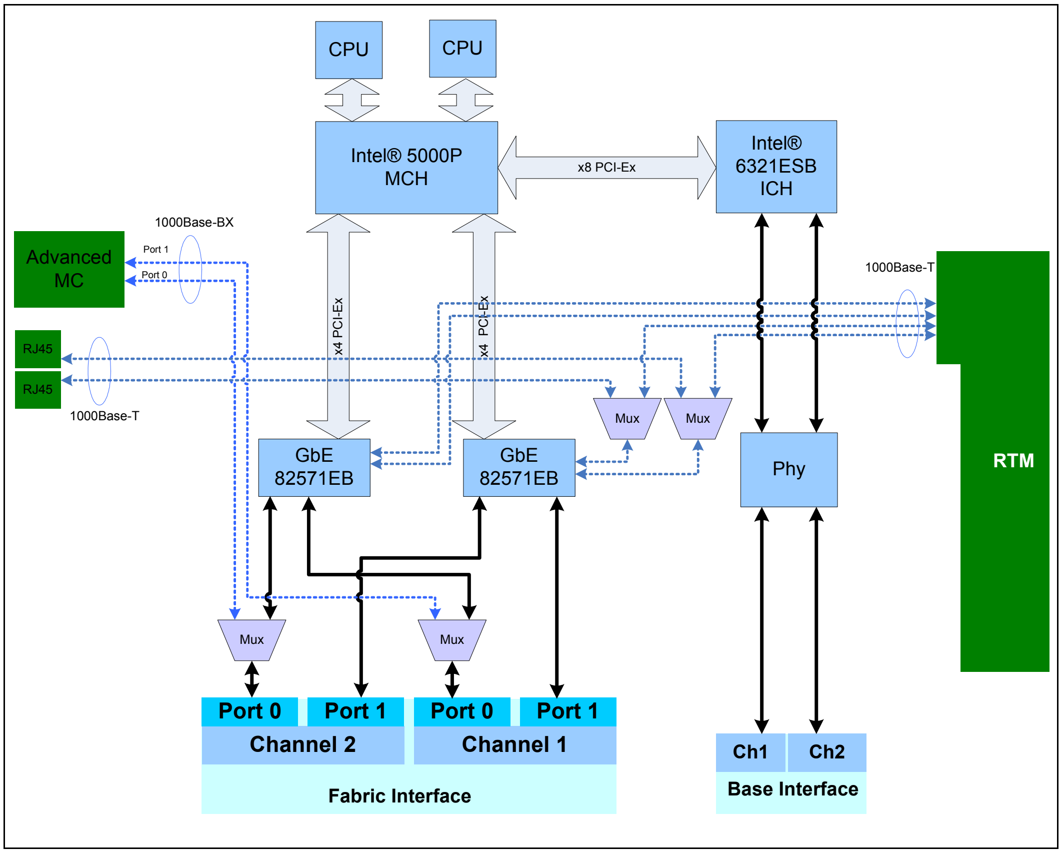
The MPCBL0050 SBC implements six Gigabit Ethernet (GbE) interfaces. Two of these are supported by Intel® 6321ESB and Intel® 82563EB PHY and are routed to the base interface on the AdvancedTCA backplane to support PICMG* 3.0 (base). The remaining four are supported with two Intel® 82571EB Gigabit Ethernet Controllers and are routed to the fabric interface on the AdvancedTCA backplane to support the 3.1 option 2 (fabric) specifications.

The user has the flexibility to route two of the fabric interface GbE ports (Port 1, Channel 1 & 2) to the front panel instead of the backplane fabric interface. It is also possible to route all four fabric interface GbE ports to the RTM connector for rear access external connectivity. This can be done by sending an OEM IPMI command via the Shelf Manager or by changing the direction in EFI BIOS. Both ways require a payload reset for the direction change to take effect.

In addition, when an AdvancedMC module providing GbE port (type AMC.2) is installed, these can be connected to the backplane fabric interface (Port 0, Channel 1 & 2) by changing the input of the the multiplexer (MUX).

Figure 5 shows detailed diagram of ALL possible GbE ports routing.

Figure 5. Gigabit Ethernet interface block diagram



The following diagrams show most common configurations of fabric ports:

- Figure 6: All fabric ports directed to the backplane (PICMG 3.1 option 2).
- Figure 7: Two fabric ports directed to the backplane, two remaining directed to the Front Panel.
- Figure 8: All four ports directed to the RTM RJ45 for rear connectivity. AMC GbE ports connected to the backplane.

For details on available redirection options, please refer to Section 7.5.1 in the EFI BIOS setup section.

Figure 6. Four GbE ports directed to the backplane (PICMG 3.1 Option 2)

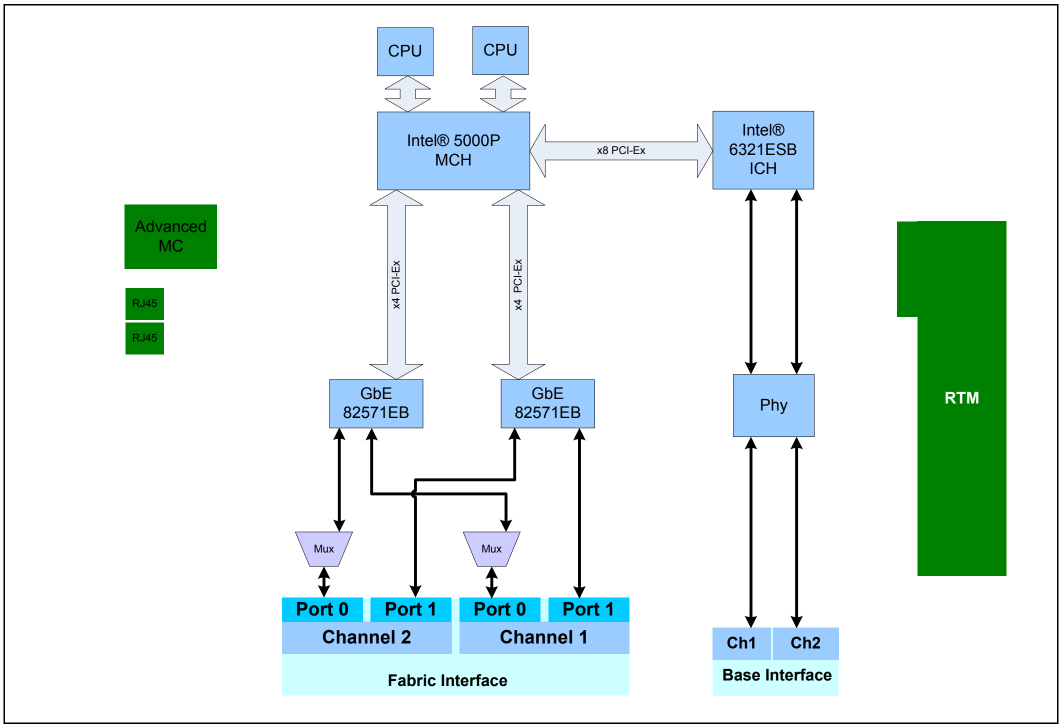


Figure 7. Two GbE ports directed to backplane, two GbE ports to front panel

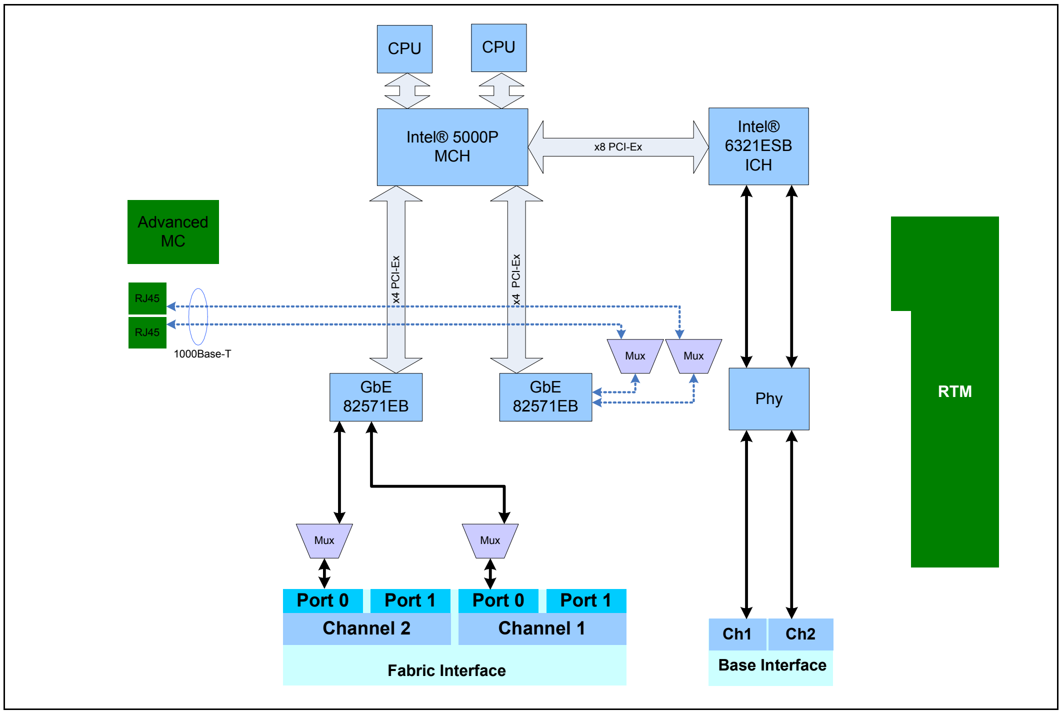
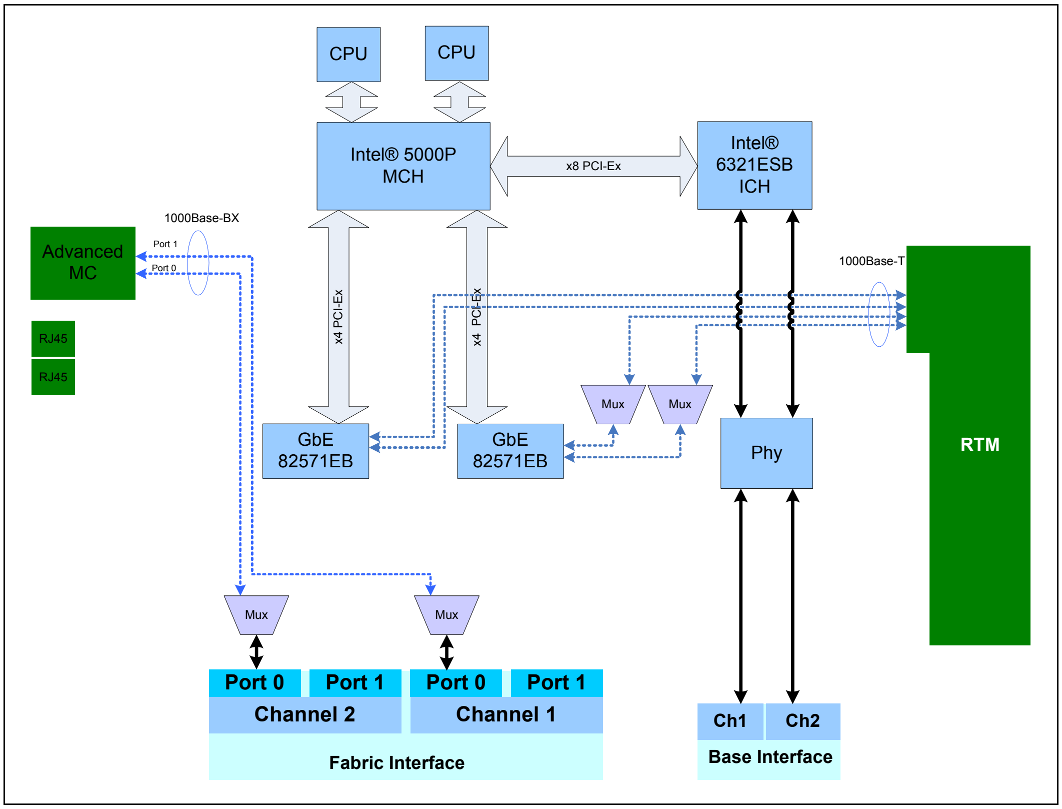


Figure 8. Four GbE ports directed to RTM, AMC GbE port connected to backplane



2.2.5.2 Serial Attached SCSI (SAS) Controller

The MPCBL0050 has a 4-port Serial Attached SCSI (SAS) controller that provides ports for AdvancedMC and RTM equipped with SAS HDD. The LSI1064 SAS controller is connected to the PCI-X bus of the Intel® 6321ESB I/O Controller Hub. All SAS ports have a serial point-to-point interface using a differential transmit/receive pair. The SAS controller has a flash device that is used to store its firmware.

The SAS ports are mapped as follows:

- Port 0: AdvancedMC Slot
- Port 1, 2,3: Rear Transition Module

There are four SAS activity LEDs on the front panel, one for each of the four SAS ports.

Note: MPCBL0050 board does not support an on board HDD. However, local storage can be provided with a SAS HDD mounted on the RTM connected to the blade.

2.2.5.3 USB 2.0

The MPCBL0050 SBC has one front panel USB connector that supports USB 2.0 and 1.1. The USB port supports Plug & Play* and hot swapping operations (OS level), which allows USB devices to be automatically attached, configured, and detached without rebooting. There is a second USB port routed to the RTM.

2.2.5.4 Serial Ports

The MPCBL0050 supports one serial port connected to the Intel® 6321ESB I/O Controller Hub. The serial port is routed to the front panel RJ-45 connector and the RTM RJ-45 connector. Since they are connected together, only one port can be used at a time (front panel or RTM).

Note: A cable connected to a serial port must always be terminated. Leaving a serial port cable unterminated may cause the board serial console or SBC to become unresponsive.

2.2.6 AdvancedMC Slot

The MPCBL0050 SBC provides one Mid-Size Single Wide AdvancedMC* slot in location B2. The MPCBL0050 SBC does not support Double Wide, Full-Size, or Compact options.

Below is an overview of the AdvancedMC slot connections:

- One x8 PCI Express port that connects to the MCH. The port may train with a link width of x8, x4, or x1. The PCI Express basic “x1” per pair link has a raw bit-rate of 5.0 Gbit/s bidirectional (2.5 Gbit/s each direction). This results in maximum bandwidth per pair of 250 MBytes/s given the 8b/10b encoding used to transmit data across this interface. For a x4 port, the maximum theoretical realized bandwidth is 1 GBytes/s in each direction or an aggregate of 2 GBytes/s.
- Two Gb Ethernet ports that connects to the fabric interface on the backplane through a multiplexer (MUX).
- One SAS port that connects to the on-board SAS controller.
- One SAS port that connects to the RTM (Zone 3).
- 7ports (13 to 15, 17 to 20) connect to the RTM connector on the rear of the board (Zone 3).

The MPCBL0050 SBC supports AdvancedMC modules with a maximum power consumption of 20 watts and has independent hot swap circuitry for +12 V and +3.3 V connections.

Table 5 shows the port mapping from the AdvancedMC connector.

Table 5. AdvancedMC connections (Sheet 1 of 2)

Link #	Function	Description
0	GEth0	AMC.2 Gigabit Ethernet connection to backplane fabric interface through a MUX: • GEth0 connects to channel 1, Port 0 • GEth1 connects to channel 2, Port 0
1	GEth1	
2	SAS	AMC.3 SAS/SATA connection to port 1 of the SAS controller on SBC
3	SAS	AMC.3 SAS/SATA connection to the RTM
4	x8 PCI- Ex	AMC.1 PCI Express connection to ports 6 and 7 on the Memory Controller Hub (MCH)
5		
6		
7		
8		
9		
10		
11		

Table 5. AdvancedMC connections (Sheet 2 of 2)

Link #	Function	Description
12		Not Connected
13		To/From Rear Transition Module (RTM) Connector J32
14		
15		
16		Not connected
17		To/From Rear Transition Module (RTM) Connector J32
18		
19		
20		

Note: Although the AdvancedMC connector provides access to an SAS port, Intel has not validated AdvancedMC cards supporting an SAS HDD. Due to thermal constraints, using such a card may not be possible in certain environments.

Caution: Never *ship* the MPCBL0050 SBC with any AdvancedMC modules installed.

Shipping the MPCBL0050 SBC with an AdvancedMC module installed may cause damage to the SBC or AdvancedMCs. Damage that occurs to the MPCBL0050 due to an AdvancedMC module installed during shipment will not be covered by the MPCBL0050 product warranty.

Caution: Always *operate* the MPCBL0050 SBC with the AdvancedMC filler panel or AdvancedMC module installed.

The AdvancedMC module slot should not be left open or uncovered when the MPCBL0050 SBC is in use. The MPCBL0050 SBC includes one AdvancedMC filler panel, which is provided to optimize cooling and radiated emissions for the SBC.

2.2.7 Firmware Hub for EFI BIOS

The MPCBL0050 SBC has two physically separate 1 MByte EFI (Extensible Firmware interface) flash devices:

- Primary EFI BIOS flash (FWH0)
- Recovery EFI BIOS flash (FWH1)

The flash is allocated for storing the binary code of the EFI.

The SBC boots from the primary flash FWH0 under normal circumstances. If booting EFI from primary flash FWH0 fails, a hardware mechanism automatically changes the flash device select logic to boot from the recovery flash FWH1.

For instructions on how to update the EFI BIOS, refer to Section 10.2. After completing the EFI update, the user must reset the system for the new EFI BIOS image to take effect.

2.2.7.1 FWH 0 (Main EFI)

BIOS executes code off of the flash device and performs checksum validation of its operational code. This checksum occurs in the boot block of BIOS. When the user performs a BIOS update, the BIOS image is stored in FWH0 only. FWH0 also stores the factory default and user-configured BIOS options (CMOS settings).

2.2.7.2 FWH 1 (Backup/Recovery EFI)

FWH 1 stores the recovery EFI BIOS image. In the event of a checksum failure on the primary BIOS operational code, BIOS requests the IPMC to switch the flash device so that the board is able to boot from FWH1 for recovery.

Note: FWH1 also stores its own configuration which is independent of the FWH0 settings.

2.2.7.3 EFI Backup Mechanism

The on-board Intelligent Platform Management Controller (IPMC) manages which of the two BIOS flash devices is selected during the boot process. The IPMC can change the BIOS flash device selection from FWH0 to FWH1 and reset the processor.

The default state of this control configures the primary Firmware Hub (FWH0) device ID to be the boot device; the secondary FWH1 is assigned the next ID. The secondary FWH1 responds to the address range just below the primary FWH0 in high memory.

The IPMC sets the ID for both FWH devices. Boot accesses are directed to the FWH with ID = 0000; unconnected ID pins are pulled low by the FWH device. In this way, the IPMC may select which flash device is used for the boot process.

Note: It is possible to force booting from FWH1 by changing on-board DIP switch setting. Refer to Section 3.4 for details.

2.2.8 On-board DC/DC converters

Power to the MPCBL0050 board is delivered from the two -48V input rails through the backplane power connector (P10).

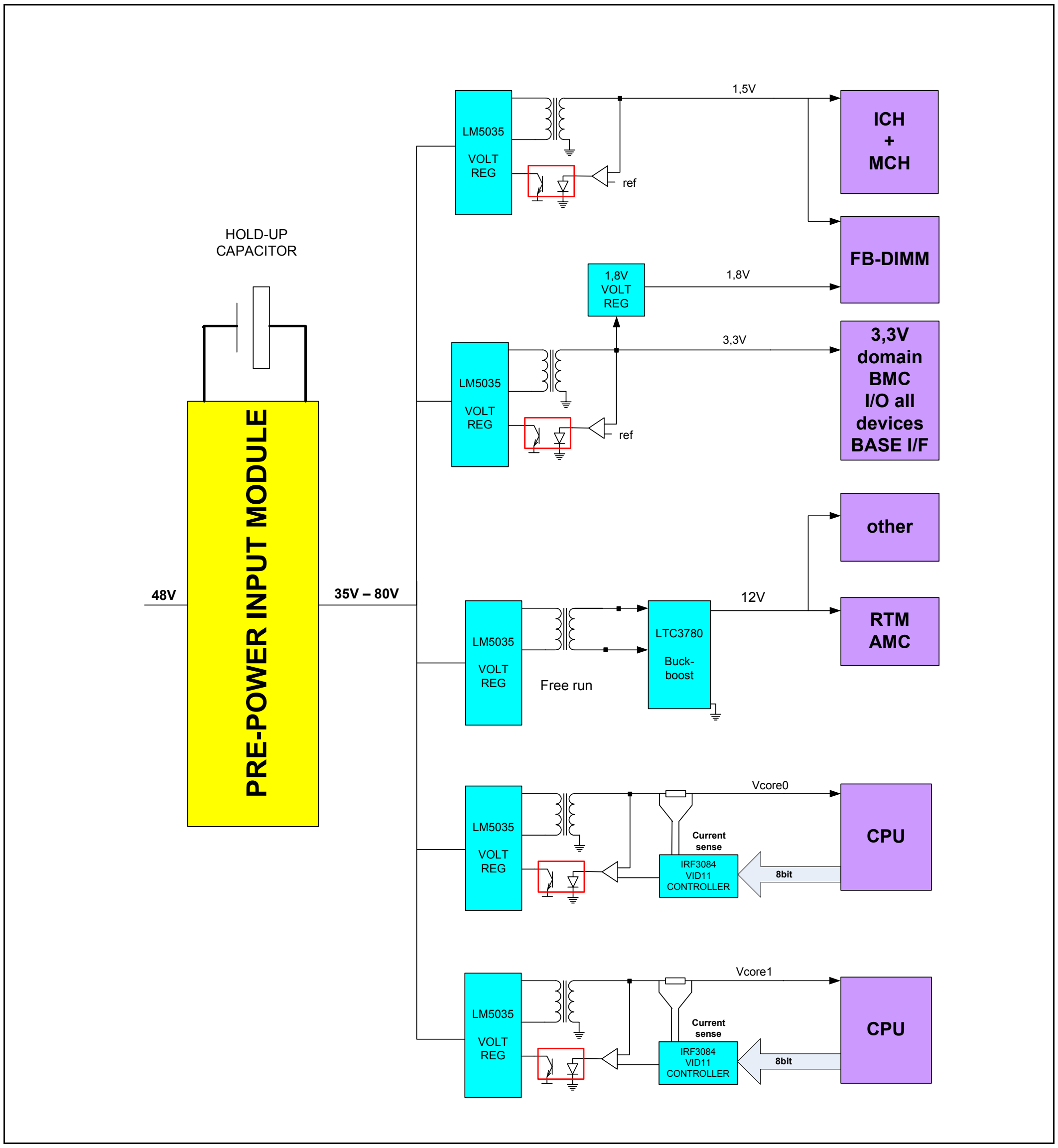
Most onboard voltages are obtained by a single-stage conversion. Primary and secondary side controllers are based on NSC devices. Transformers from Pulse Engineering* ensure basic insulation up to 1500V.

Voltage sequencing is implemented via CPLD which is part of IPMC circuitry. 3.3V for IPMC is provided independently from 3.3V to the payload. After E-keying, this voltage is forwarded further to other devices according to the power sequence.

The power module, which is part of the baseboard, contains all necessary support for meeting AdvancedTCA requirements such as fusing, ORing dual voltage feeds, hot-swap capability, in-rush current protection, and hold-up circuit.

Figure 9 shows high-level power distribution on the SBC.

Figure 9. Power conversion diagram



2.2.8.1 AdvancedTCA -48V Power Feed

The PRE-POWER block is used to interface directly to the AdvancedTCA power feeds and is equipped with all necessary devices to meet PICMG 3.0/3.1 requirements. The main features of this circuit are:

- Input fusing from the AdvancedTCA backplane
- 250W of -48 VDC, inrush protected and filtered per the PICMG 3.0 specification
- ORing function for the A & B feeds and their returns
- EMI filtering (designed to meet CISPR Class B) for conducted emissions

- Hot Swap devices comply to with the PCIMG 3.0 requirements
- 80 VDC charging current for holdup storage capacitors to meet the holdup requirement in the PICMG 3.0 spec
- Alarms for A/B feed loss and fuse failure through isolated outputs to IPMI
- Overtemperature protection

The entry block is able to turn off all of its devices when it receives a break signal from baseboard temperature sensors located at thermally sensible locations (close to large consumers of power such as main processors and DC-DC converters). This is protection-independent of IPMI devices and is a last stage for avoiding board damage if IPMI devices malfunction.

2.2.8.2 3.3V

This voltage is derived directly from 48V with single-stage conversion. Initially, it supplies early power for management circuitry and the base interface controller. After payload power is enabled, it also supplies other 3.3V domain devices.

2.2.8.3 1.5V

These voltages are derived directly from 48V with single-stage conversion. Voltage 1.5V is required for powering MCH, ICH, and FB-DIMMs.

2.2.8.4 1.8V

1.8V is derived from 3.3V using a standard buck converter and supplies FBDIMMs.

2.2.8.5 12V

This feed is used by the RTM, AMC, and others devices such as gate drivers. This supply is derived from a dual stage converter.

2.2.8.6 CPU VRD

For efficiency, core voltages are obtained directly from 48V. Converters for both CPUs are identical.

Voltages for processors depend on a digital word provided to special control logic which is able to tune it across the whole range of the VRM11 specification. Circuitry includes D/A converter to translate a VID word into the desired voltage, overcurrent protection, characteristic modeling, and logic circuit for enabling.

2.2.9 Autonomous Emergency Power-down Circuitry

To help protect the system from disaster, a number of thermal sensors are placed on the board. These sensors measure temperature in the most thermally stressed points on the board. In case of an uncontrolled temperature rise, the main power to the board will be cut off. The sensors used by this feature are not monitored by IPMC.

Note: During normal board operation, it is IPMC's responsibility to monitor onboard temperature and inform the Shelf Manager by logging SEL events. Autonomous power-down is last-resort protection intended to save the board in the event of IPMC or Shelf Manager failure.

2.2.10 Intelligent Platform Management Controller

The MPCBL0050 uses the Renesas* HD64F2166 processor, as the Intelligent Platform Management Controller (IPMC). The IPMC is a management subsystem providing monitoring, event logging, and recovery control. The IPMC serves as the gateway for management applications to access the platform hardware. Some of the key features are:

- Compliance with PICMG 3.0 and IPMI v2.0
- Automatic rollback capability if an operational image upgrade fails
- Upgradable from both IPMI interface (KCS and IPMB)
- Support for serial port redirection over LAN interface
- Supports the initiation of a graceful shutdown on the host CPU

The IPMC circuitry also utilizes a Xilinx* XC3S1000 FPGA and Lattice* ispMACH4512 for glue-logic and to control the power-up and power-down sequencing of the power supplies. It is powered by sustaining 3.3V and is clocked at 32.768 khz. The CPLD controls resets, the enabling and monitoring of power good signals from all the on-board power converters, and power sequencing to ensure that all of the converters power up in the correct order to prevent latch-up or damage to a device. The CPLD will be used for parallel loading of FPGA from BMC firmware.

The FPGA is also used to extend the GPIO interconnects required by the IPMC, and to monitor Port 80 POST codes during EFI BIOS execution.

The National Semiconductor* LM93 is used by the IPMC subsystem to monitor on-board power supplies and processor thermal diodes.

2.2.11 256 MByte Flash Drives

The board has two 256 MByte flash devices. Each flash is connected to the ICH via ATA Flash Disk Controller. Main characteristics of the controller are:

- Write performance up to 10.0 MB/sec
- Security protection for confidential information stored in the flash media
- WP_PD# pin to protect critical information stored in the flash media from unauthorized overwrites. (On board DIP switch allows locking the flash content.)
- One ATA controller working as master Flash HDD and the second as slave. (Done via hardware strap pin CSEL, controlled by IPMC)

The flashes are visible to the operating system as two IDE HDDs. Each of the 256 MByte flash drives can be used to store anything that can be kept on a normal hard drive. In addition, the ATA controller has a wear leveling algorithm to improve the longevity of the flash device.

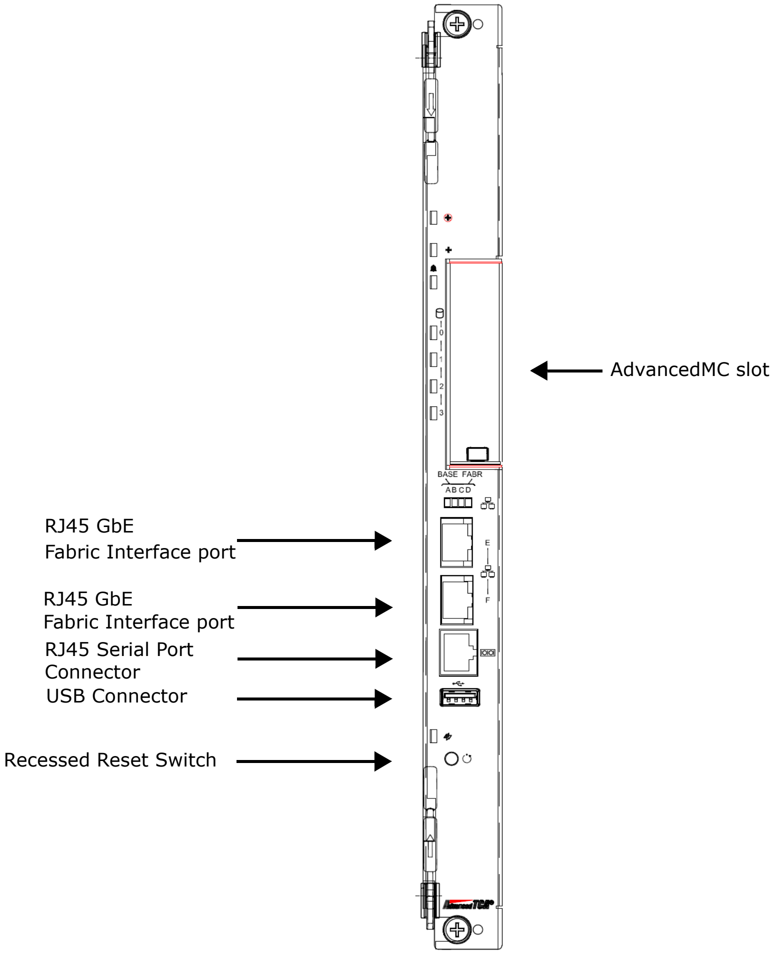
2.2.12 Real-Time Clock

The MPCBL0050 SBC real-time clock is integrated into the ICH. It is derived from a 32.768 kHz crystal. The real-time clock is powered by a large capacitor when main power is not applied to the board. If the capacitor fully discharges, the only effect will be loss of RTC time/date. The correct date/time will need to be set next time the board is inserted. The RTC clock is usually set using the Network Time Protocol once the operating system has loaded.

3.0 Connectors and LEDs

3.1 Front Panel Connectors

Figure 10. Front panel connectors



3.1.1 USB Connector

The MPCBL0050 SBC has one USB connector, J3, that supports USB 2.0 and USB 1.1. This connector is available through the front panel. Figure 11 shows the connector and Table 6 lists the pin assignments.

Figure 11. USB connector

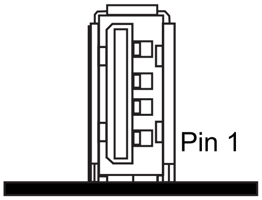


Table 6. USB connector pin assignments

Pin	Signal
1	+5 V
2	-DATA
3	+DATA
4	GND

3.1.2 Serial Port Connector

A single serial port interface is provided on the front edge of the card using an RJ-45 style shielded connector. The connector is an 8-pin RJ-45. Figure 12 shows the connector and Table 7 gives the pin assignments. Figure 13 shows the RJ-45 to DB-9 translation.

Figure 12. Serial port connector (J4)

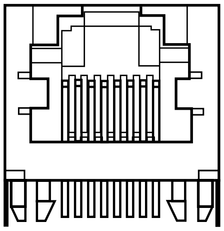
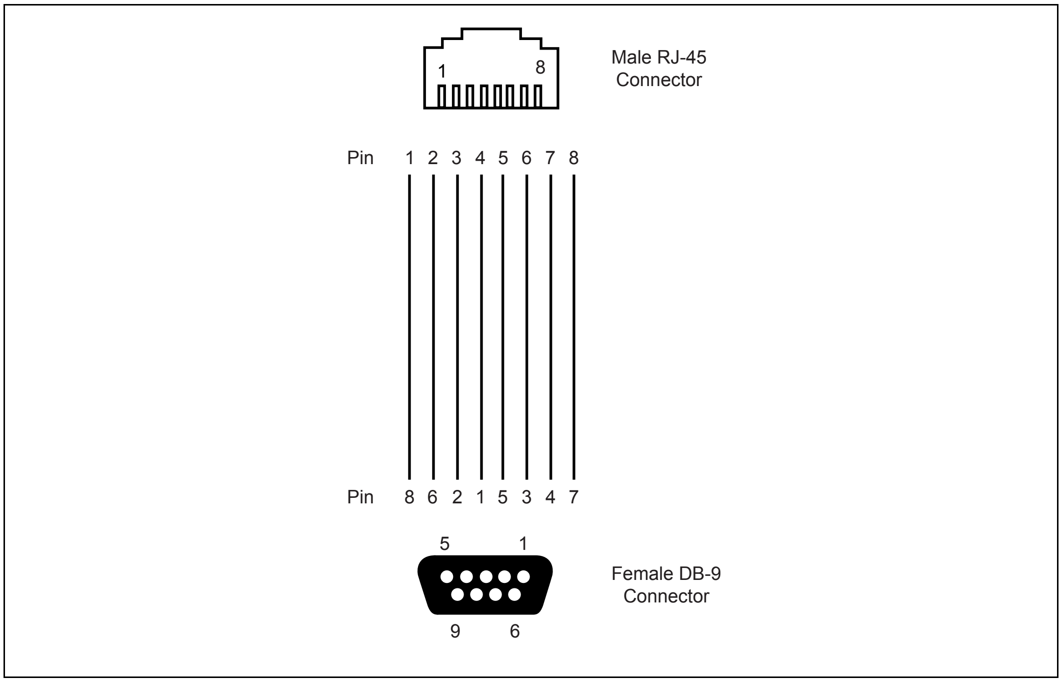


Table 7. Serial port connector (J4) pin assignments

Pin	Signal
1	RTS
2	DTR
3	TXD
4	GND
5	GND
6	RXD
7	DSR
8	CTS

Figure 13. DB-9 to RJ-45 pin translation



3.1.3 Ethernet 10/100/1000 Connectors

Two Ethernet ports are provided on the front edge of the board using an RJ-45 style shielded connector (Bel Fuse* L829-1BIT-43). Figure 14 shows the connector and Table 8 gives the pin assignments.

Figure 14. Gigabit Ethernet connector

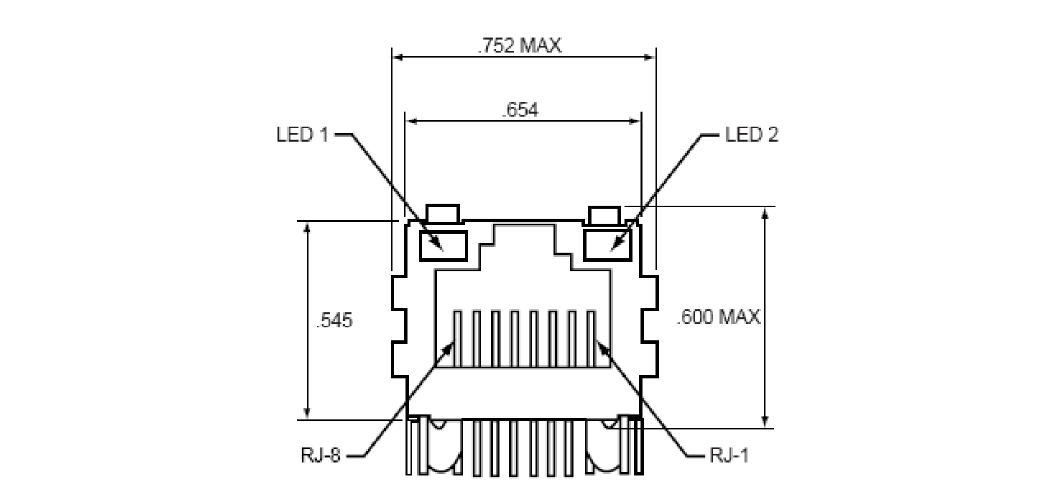


Table 8. Gigabit Ethernet connector pin assignments

Pin	Signal Name	Comments
1	BI_DA+	Bi-directional lane 1, positive polarity
2	BI_DA-	Bi-directional lane 1, negative polarity
3	BI_DB+	Bi-directional lane 2, positive polarity
4	BI_DC+	Bi-directional lane 3, positive polarity
5	BI_DC-	Bi-directional lane 3, negative polarity
6	BI_DB-	Bi-directional lane 2, negative polarity
7	BI_DD+	Bi-directional lane 4, positive polarity
8	BI_DD-	Bi-directional lane 4, negative polarity

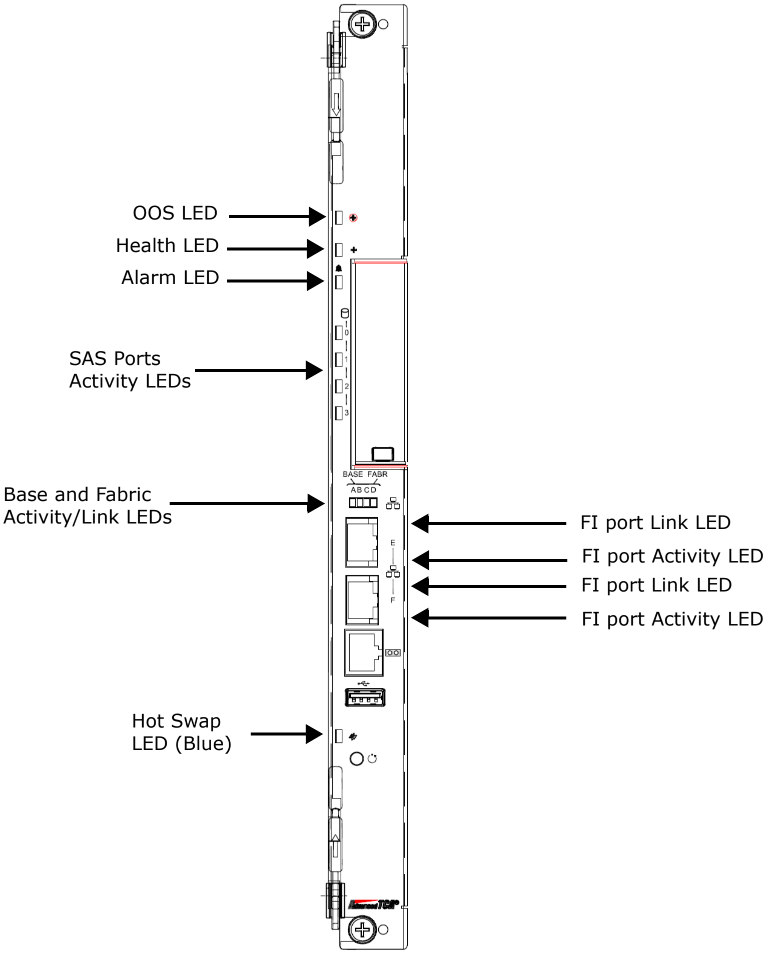
3.1.4 Front Panel Reset Switch

The reset switch is located in a small recessed hole below the USB connector. The reset button is an input to the IPMC to request a payload soft reset. There are IPMI commands to reset the board and change power states through the software. The reset button can be used only when a user is physically present at the chassis to operate the button. The reset button is available at the front panel as shown in Figure 10.

3.2 Front Panel LEDs

Figure 15 shows LEDs the MPCBL0050 SBC uses to indicate status.

Figure 15. Front panel LEDs



When these LEDs are lit, they indicate a status as defined in the Table 9.

Table 9. Front panel LED descriptions (Sheet 1 of 2)

LED	Function
Hot Swap	**Function:** Hot Swap as defined in the AdvancedTCA 3.0 specification It is also possible for a user to override the default behavior of the LED using AdvancedTCA FRU LED Control commands. **Possible States:** OFF / BLUE / SHORT BLINK / LONG BLINK
Out of Service	**Function:** Out of Service (AdvancedTCA LED 1) RED: The board is out of service. OFF: The board is running. It is possible for a user to override the default IPMC behavior of the LED using AdvancedTCA FRU LED Control commands. **Possible States:** OFF / RED / AMBER

Table 9. Front panel LED descriptions (Sheet 2 of 2)

LED	Function
Health ✚	**Function:** Health (AdvancedTCA LED 2). The SBC health is based on an aggregation of IPMI sensors, like board temperature and voltage. GREEN: The SBC is healthy. RED: The SBC is not healthy. It is possible for the user to override the default IPMC behavior of the LED using AdvancedTCA FRU LED Control commands. **Possible States:** OFF / GREEN / RED / AMBER
🔔	**Function:** Alarm LED (AdvancedTCA LED 3). This LED is user-defined and in off state by default. The LED's default IPMC behavior can be overridden with AdvancedTCA FRU LED Control commands. Possible States: OFF / AMBER
Ports A, B, C, D Link/Activity	There is one LED that indicates link and activity for the following ports: • A: Base Interface: Channel 1, Port 0 • B: Base Interface: Channel 2, Port 0 • C: Fabric Interface: Channel 1, Port 0 • D: Fabric Interface: Channel 2, Port 0 **Function:** Gigabit Ethernet Base/Fabric Interface Link and Activity OFF: No Link GREEN: Link GREEN-BLINK: Link and Activity
Ports E & F **Link Speed** 🖧	There is one LED that indicates link speed for the following ports: • E: Fabric Interface: Channel 1, Port 1 • F: Fabric Interface: Channel 2, Port 1 **Function:** Gigabit Ethernet Fabric Interface Link Speed OFF: 10 Mb/s GREEN: 100 Mb/s AMBER: 1000 Mb/s Note: This LED will be active even if Ethernet port is connected to the backplane fabric interface. This will show status of Fabric Interface ports on the backplane.
Ports E & F **Activity** 🖧	There is one LED that indicates activity for the following ports: • E: Fabric Interface: Channel 1, Port 1 • F: Fabric Interface: Channel 2, Port 1 **Function:** Gigabit Ethernet Fabric Interface Link Activity: GREEN-NO BLINKING: No traffic GREEN-BLINKING: Transmit or receive traffic is coming across the link Note: This LED will be active even if Ethernet port is connected to the backplane fabric interface.
SAS Ports Activity	**Function:** SAS controller ports activity. 0/1/2: SAS ports 0/1/2 - connected to the RTM 3: SAS Port 3- connected to AdvancedMC OFF: No traffic on SAS ports BLINKING GREEN: Traffic on SAS ports (read/write activity)

3.3 Backplane and On-Board Connectors, DIP Switch Location

Connectors along the rear edge of AdvancedTCA* server blades are divided into three distinct zones, as described in Section 2.3 of the PICMG 3.0 Specification:

- Zone 1 for system management and power distribution. (P10)
- Zone 2 for data fabric (J20, J23)
- Zone 3 for the rear transition module (RTM) (J30, J31, J32, J33)

Figure 16 shows the locations of the backplane and on-board connectors and Table 10 explains the function of each connector.

Figure 16. Backplane and on-board connector/DIP switch locations

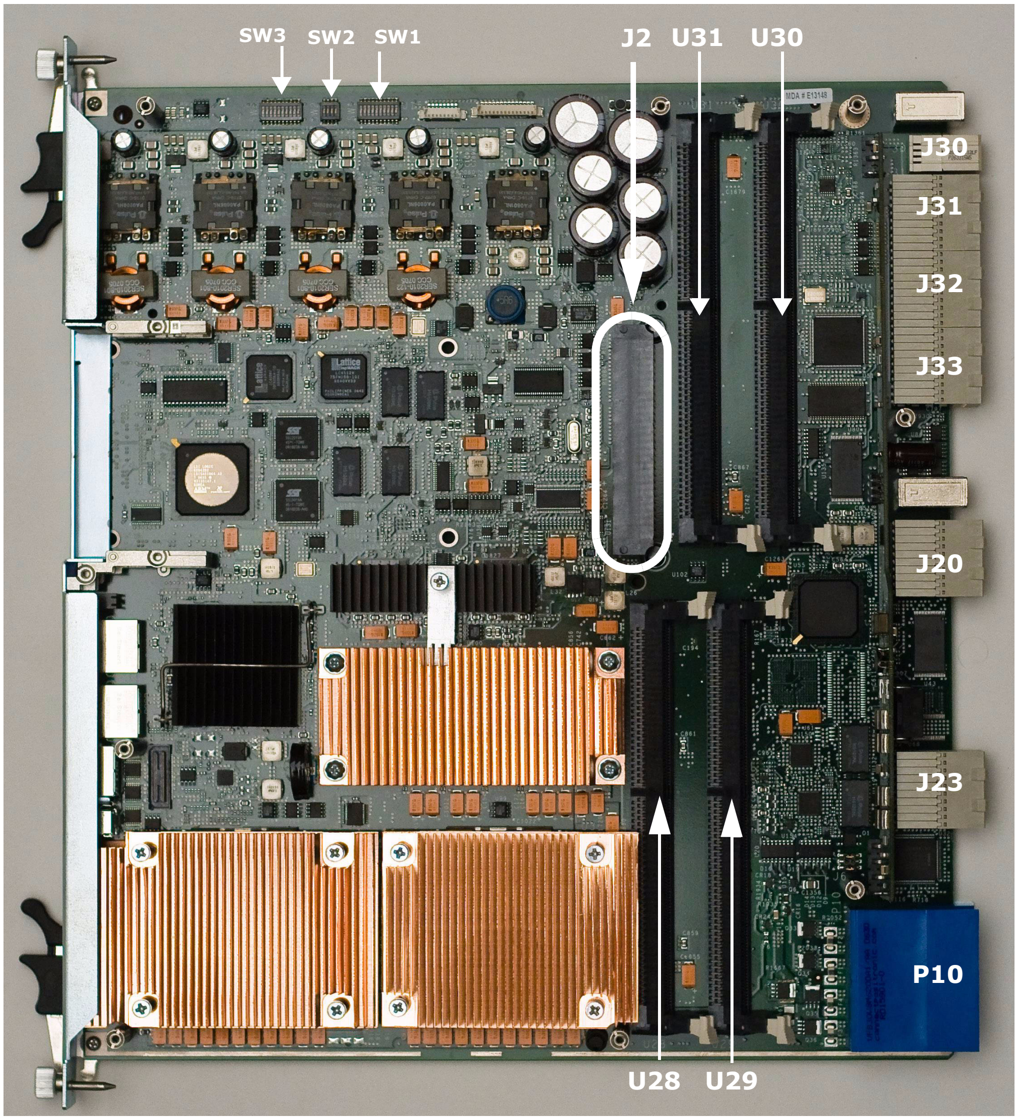


Table 10. Backplane and on-board connector assignments

Connector	Description
J2	AdvancedMC* connector
U28	FBDIMM1 connector (Branch0-Channel0)
U29	FBDIMM2 connector (Branch0-Channel1)
U30	FBDIMM3 connector (Branch1-Channel2)
U31	FBDIMM4 connector (Branch1-Channel3)
J20	AdvancedTCA data transport for Update Channel (Zone2)
J23	AdvancedTCA data transport for Base and Fabric Interface (Zone2)
J30	RTM Power connector (Zone 3)
J31	RTM Data and Control connector (Zone 3)
J32	RTM Data and Control connector (Zone 3)
J33	RTM Data and Control connector (Zone 3)
P10	Advanced TCA Power and IPMB
SW1-3	DIP Switches. Refer to the chapter "DIP Switches" on page 42

3.4 DIP Switches

The MPCBL0050 contains a number of DIP switches that allow the user to configure certain options not configurable through the EFI Setup utility. These DIP switches are used for diagnostic purposes. Users should take precautions when changing these switches. See Figure 16 for the locations of the DIP switches. The following tables describe DIP switches functionality.

Warning: Changing any of the "Reserved" switches to position other then default may render the MPCBL0050 instable or prevent the board from powering up. It is advised to check DIP switches for valid configuration always before inserting board into the chassis.

Note: The DIP switch descriptions presented below (Table 11, Table 12, and Table 13) apply only to the production version of the MPCBL0050 (only 3 DIP switch modules on the board: SW1, SW2, SW3). For details on preproduction boards, please contact your Intel representative.

Table 11. DIP switch SW1

Switch	Pos#	Signal Name	Switch setting	Default
SW1	1	N/A	RESERVED	Off
	2	N/A	RESERVED	Off

Table 11. DIP switch SW1

	3	N/A	RESERVED	Off
	4	PATA_WP_PD_N	On= Enable PATA flash write protect	Off
	5	SW_FWH_1_IDSEL	On= FWH1 selected (otherwise the IPMC selects the FWH)	Off
	6	FW_Rollback	On= Force IPMC Firmware Rollback	Off
	7	FRC_UPD	On= Force IPMC Firmware Update	Off
	8	FRB_DIS	On=Disable Fault Resilient Booting (BIOS)	Off
	9	SW_SPM_DISB_N	On= Serial Programming Mode enabled (Used for IPMC boot block update)	Off
	10	SW_PS_EN	On= Power payload without interaction of ShMC	Off

Table 12. DIP switch SW2

Switch	Pos#	Signal Name	Switch setting	Default
SW2	1	CLR_CMOS_N	On= Clear CMOS (load default EFI BIOS settings	Off
	2	CLR_PASSWD_N	On = Clear Password	Off
	3	N/A	RESERVED	Off
	4	N/A	RESERVED	Off

Table 13. DIP switch SW3

Switch	Pos#	Signal Name	Switch setting	Default
SW3	1	N/A	RESERVED	Off
	2 3	N/A	RESERVED	Off
	4	SW_AT_UART_SELECT	On = Enable IPMC serial port (required for IPMC boot block upgrade)	Off
	5	SW_IPMC_RST	On = Payload power on with IPMC in reset mode (Allows starting board without IPMC)	Off
	6	N/A	RESERVED	Off
	7 8 9 10	N/A	RESERVED	Off

3.5 On-Board Connectors

3.5.1 AdvancedMC Connector (J2)

The connectors and pinouts are defined by the industry standard specifications AMC.0 R1.0, AMC.1 R1.0, AMC.2 R1.0, and AMC.3 R1.0.

Refer to Table 5 for a high-level overview of the AdvancedMC port mapping. Pin assignments are shown in Table 14.

Table 14. AdvancedMC connector pin assignments (Sheet 1 of 5)

PIN	Signal	Comments
B1	GND1 - GND	Logic Ground
B2	PWR1 - 12V_AMC1	AMC main power
B3	PS1* - AMC_PRESENT1_N_B1	Presence 1
B4	MP - 3.3V_AMC1	3.3V management power
B5	GA0 -N.C.	Geo Address - Bit 0 <- ì1î
B6	ETH100RX - N.C.	Not Used (Reserved by AMC.0 R1.0)
B7	GND2 - GND	Logic Ground
B8	ETH100TX - N.C.	Not Used (Reserved by AMC.0 R1.0)
B9	PWR2 - 12V_AMC1	12V AMC main power
B10	GND3 - GND	Logic Ground
B11	Rx0+ - FIC_AMC1TX0_P	AMC Port 0 Tx - Ethernet link to ATCA backplane
B12	Rx0- - FIC_AMC1TX0_N	|
B13	GND4 - GND	Logic Ground
B14	Tx0+ - FIC_AMC1RX0_P	AMC Port 0 Rx - Ethernet link to ATCA backplane
B15	Tx0- - FIC_AMC1RX0_N	|
B16	GND5 - GND	
B17	GA1 - GND	Geo Address - Bit 1 <- 0
B18	PWR3 - 12V_AMC1	AMC main power
B19	GND6 - GND	
B20	Rx1+ - FIC_AMC1TX1_P	AMC Port 1 Tx - Ethernet link to ATCA backplane
B21	Rx1- - FIC_AMC1TX1_N	|
B22	GND7 - GND	Logic Ground
B23	Tx1+ - FIC_AMC1RX1_P	AMC Port 1 Rx - Ethernet link to ATCA backplane
B24	Tx1- - FIC_AMC1RX1_N	|
B25	GND8 - GND	Logic Ground
B26	GA2 - N.C.	Geo Address - Bit 2 <- 1
B27	PWR4 - 12V_AMC1	AMC main power
B28	GND9 - GND	Logic Ground
B29	Rx2+ - SAS_RX3_P	AMC Port 2 Tx - SAS/SATA link to port 0 of LSI 1064 SAS controller
B30	Rx2- - SAS_RX3_N	|
B31	GND10 - GND	Logic Ground

Table 14. AdvancedMC connector pin assignments (Sheet 2 of 5)

B32	Tx2+ - SAS_TX3_P_C	AMC Port 2 Rx - SAS/SATA link to port 0 of LSI 1064 SAS controller
B33	Tx2- - SAS_TX3_N_C	|
B34	GND11 - GND	Logic Ground
B35	Rx3+ - SAS_AMC_RTM_RX_P	AMC Port 3 Tx - SAS/SATA link to RTM
B36	Rx3- - SAS_AMC_RTM_RX_N	|
B37	GND12 - GND	Logic Ground
B38	Tx3+ - SAS_AMC_RTM_TX_P	AMC Port 3 Rx - SAS/SATA link from RTM
B39	Tx3- - SAS_AMC_RTM_TX_N	|
B40	GND13 - GND	Logic Ground
B41	ENABLE* - AMC_ENABLE_N_B1	AMC Enable
B42	PWR5 - 12V_AMC1	12V AMC main power
B43	GND14 - GND	Logic Ground
B44	Rx4+ - MCH_EXP6RXP<0>	AMC Port 4 Tx - PCI-Ex to MCH Port 6 Lane 0
B45	Rx4- - MCH_EXP6RXN<0>	|
B46	GND15 - GND	Logic Ground
B47	Tx4+ - MCH_EXP6TXP_C<0>	AMC Port 4 Rx - PCI-Ex from MCH port 6 Lane 0
B48	Tx4- - MCH_EXP6TXN_C<0>	|
B49	GND16 - GND	Logic Ground
B50	Rx5+ - MCH_EXP6RXP<1>	AMC Port 5 Tx - PCI-Ex to MCH Port 6 Lane 1
B51	Rx5- - MCH_EXP6RXN<1>	|
B52	GND17 - GND	Logic Ground
B53	Tx5+ - MCH_EXP6TXP_C<1>	AMC Port 5 Rx - PCI-Ex from MCH port 6 Lane 1
B54	Tx5- - MCH_EXP6TXN_C<1>	|
B55	GND18 - GND	Logic Ground
B56	SCL_L - AMC_SCL_B1	IPMB-L Clock
B57	PWR6 - 12V_AMC1	12V AMC main power
B58	GND19 - GND	Logic Ground
B59	Rx6+ - MCH_EXP6RXP<2>	AMC Port 6 Tx - PCI-Ex to MCH Port 6 Lane 2
B60	Rx6- - MCH_EXP6RXN<2>	|
B61	GND20 - GND	Logic Ground
B62	Tx6+ - MCH_EXP6TXP_C<2>	AMC Port 6 Rx - PCI-Ex from MCH port 6 Lane 2
B63	Tx6- - MCH_EXP6TXN_C<2>	|
B64	GND21 - GND	Logic Ground
B65	Rx7+ - MCH_EXP6RXP<3>	AMC Port 7 Tx - PCI-Ex to MCH Port 6 Lane 3
B66	Rx7- - MCH_EXP6RXN<3>	|
B67	GND22 - GND	Logic Ground
B68	Tx7+ - MCH_EXP6TXP_C<3>	AMC Port 7 Rx - PCI-Ex from MCH port 6 Lane 3
B69	Tx7- - MCH_EXP6TXN_C<3>	|
B70	GND23 - GND	Logic Ground
B71	SDA_L - AMC_SDA_B1	IPMB-L Data
B72	PWR7 - 12V_AMC1	12V AMC main power

Table 14. AdvancedMC connector pin assignments (Sheet 3 of 5)

B73	GND24 - GND	Logic Ground
B74	CLK1+ - AMC_X1_CLK1+	Not Used
B75	CLK1- - AMC_X1_CLK1-	|
B76	GND25 - GND	Logic Ground
B77	CLK2+ - AMC_X1_CLK2+	Not Used
B78	CLK2- - AMC_X1_CLK2-	|
B79	GND26 - GND	Logic Ground
B80	CLK3+ - CLK_100M_AMC_P_B1	Synchronization Clock 3 - PCI-Ex clock
B81	CLK3- - CLK_100M_AMC_N_B1	|
B82	GND27 - GND	Logic Ground
B83	PS0* - GND	Presence 0
B84	PWR8 - 12V_AMC1	12V AMC main power
B85	GND28 - GND	Logic Ground
B86	GND29 - GND	Logic Ground
B87	Tx8- - MCH_EXP7TXN_C<0>	AMC Port 8 Rx - PCI-Ex from MCH port 7 Lane 0
B88	Tx8+ - MCH_EXP7TXP_C<0>	|
B89	GND30 - GND	Logic Ground
B90	Rx8- - MCH_EXP7RXN<0>	AMC Port 8 Tx - PCI-Ex to MCH Port 7 Lane 0
B91	Rx8+ - MCH_EXP7RXP<0>	|
B92	GND31 - GND	Logic Ground
B93	Tx9- - MCH_EXP7TXN_C<1>	AMC Port 9 Rx - PCI-Ex from MCH port 7 Lane 1
B94	Tx9+ - MCH_EXP7TXP_C<1>	|
B95	GND32 - GND	Logic Ground
B96	Rx9- - MCH_EXP7RXN<1>	AMC Port 9 Tx - PCI-Ex to MCH Port 7 Lane 1
B97	Rx9+ - MCH_EXP7RXP<1>	|
B98	GND33 - GND	Logic Ground
B99	Tx10- - MCH_EXP7TXN_C<2>	AMC Port 10 Rx - PCI-Ex from MCH port 7 Lane 2
B100	Tx10+ - MCH_EXP7TXP_C<2>	|
B101	GND34 - GND	Logic Ground
B102	Rx10- - MCH_EXP7RXN<2>	AMC Port 10 Tx - PCI-Ex to MCH Port 7 Lane 2
B103	Rx10+ - MCH_EXP7RXP<2>	|
B104	GND35 - GND	Logic Ground
B105	Tx11- - MCH_EXP7TXN_C<3>	AMC Port 11 Rx - PCI-Ex from MCH port 7 Lane 3
B106	Tx11+ - MCH_EXP7TXP_C<3>	|
B107	GND36 - GND	Logic Ground
B108	Rx11- - MCH_EXP7RXN<3>	AMC Port 11 Tx - PCI-Ex to MCH Port 7 Lane 3
B109	Rx11+ - MCH_EXP7RXP<3>	|
B110	GND37 - GND	Logic Ground
B111	Tx12- - AMC_12_TXN_B1	not used
B112	Tx12+ - AMC_12_TXP_B1	not used
B113	GND38 - GND	Logic Ground
B114	Rx12- - AMC_12_RXN_B1	not used

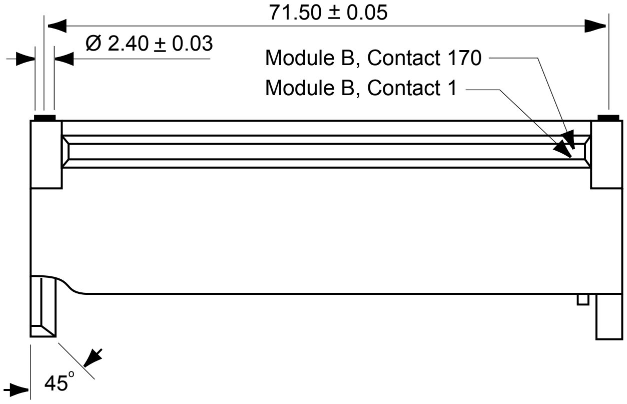
Table 14. AdvancedMC connector pin assignments (Sheet 4 of 5)

B115	Rx12+ - AMC_12_RXP_B1	not used
B116	GND39 - GND	Logic Ground
B117	Tx13- - AMC_13_TXN_B1	AMC Port 13 Rx - AMC I/O connection from RTM
B118	Tx13+ - AMC_13_TXP_B1	|
B119	GND40 - GND	Logic Ground
B120	Rx13- - AMC_13_RXN_B1	AMC Port 13 Tx - AMC I/O connection to RTM
B121	Rx13+ - AMC_13_RXP_B1	|
B122	GND41 - GND	Logic Ground
B123	Tx14- - AMC_14_TXN_B1	AMC Port 14 Rx - AMC I/O connection from RTM
B124	Tx14+ - AMC_14_TXP_B1	|
B125	GND42 - GND	Logic Ground
B126	Rx14- - AMC_14_RXN_B1	AMC Port 14 Tx - AMC I/O connection to RTM
B127	Rx14+ - AMC_14_RXP_B1	|
B128	GND43 - GND	Logic Ground
B129	Tx15- - AMC_15_TXN_B1	AMC Port 15 Rx - AMC I/O connection from RTM
B130	Tx15+ - AMC_15_TXP_B1	|
B131	GND44 - GND	Logic Ground
B132	Rx15- - AMC_15_RXN_B1	AMC Port 15 Tx - AMC I/O connection to RTM
B133	Rx15+ - AMC_15_RXP_B1	|
B134	GND45 - GND	Logic Ground
B135	Tx16- - AMC_16_TXN_B1	Not Used
B136	Tx16+ - AMC_16_TXP_B1	|
B137	GND46 - GND	Logic Ground
B138	Rx16- - AMC_16_RXN_B1	Not Used
B139	Rx16+ - AMC_16_RXP_B1	|
B140	GND47 - GND	Logic Ground
B141	Tx17- - AMC_17_TXN_B1	AMC Port 17 Rx - AMC I/O connection from RTM
B142	Tx17+ - AMC_17_TXP_B1	|
B143	GND48 - GND	Logic Ground
B144	Rx17- - AMC_17_RXN_B1	AMC Port 17 Tx - AMC I/O connection to RTM
B145	Rx17+ - AMC_17_RXP_B1	|
B146	GND49 - GND	Logic Ground
B147	Tx18- - AMC_18_TXN_B1	AMC Port 18 Rx - AMC I/O connection from RTM
B148	Tx18+ - AMC_18_TXP_B1	|
B149	GND50 - GND	Logic Ground
B150	Rx18- - AMC_18_RXN_B1	AMC Port 18 Tx - AMC I/O connection to RTM
B151	Rx18+ - AMC_18_RXP_B1	|
B152	GND51 - GND	Logic Ground
B153	Tx19- - AMC_19_TXN_B1	AMC Port 19 Rx - AMC I/O connection from RTM
B154	Tx19+ - AMC_19_TXP_B1	|
B155	GND52 - GND	Logic Ground
B156	Rx19- - AMC_19_RXN_B1	AMC Port 19 Tx - AMC I/O connection to RTM

Table 14. AdvancedMC connector pin assignments (Sheet 5 of 5)

B157	Rx19+ - AMC_19_RXP_B1	|
B158	GND53 - GND	Logic Ground
B159	Tx20- - AMC_20_TXN_B1	AMC Port 20 Rx - AMC I/O connection from RTM
B160	Tx20+ - AMC_20_TXP_B1	|
B161	GND54 - GND	Logic Ground
B162	Rx20- - AMC_20_RXN_B1	AMC Port 20 Tx - AMC I/O connection to RTM
B163	Rx20+ - AMC_20_RXP_B1	|
B164	GND55 - GND	Logic Ground
B165	TCLK - JTAG_TCLK_AMC_B1	JTAG
B166	TMS - JTAG_TMS_AMC_B1	|
B167	TRST* - JTAG_TRST_N_AMC_B1	|
B168	TDO - JTAG_TDO_AMC_B1	|
B169	TDI - JTAG_TDI_AMC_B1	|
B170	GND56 - GND	Logic Ground

Figure 17. AdvancedMC connector



3.6 Backplane Connectors

3.6.1 Power Distribution Connector (P10)

Zone 1 consists of P10, a 34-pin Positronic* header connector that provides the following signals:

- Two -48 VDC power feeds (four signals each; eight signals total)
- Two IPMB ports (two signals each, four signals total)
- Geographic address (eight signals)
- Two ground pins
- 12 unused (but physically present) pins

Figure 18 shows the mechanical drawing of the connector. The pin assignments are listed in Table 15.

Figure 18. Power distribution connector (Zone 1) P10

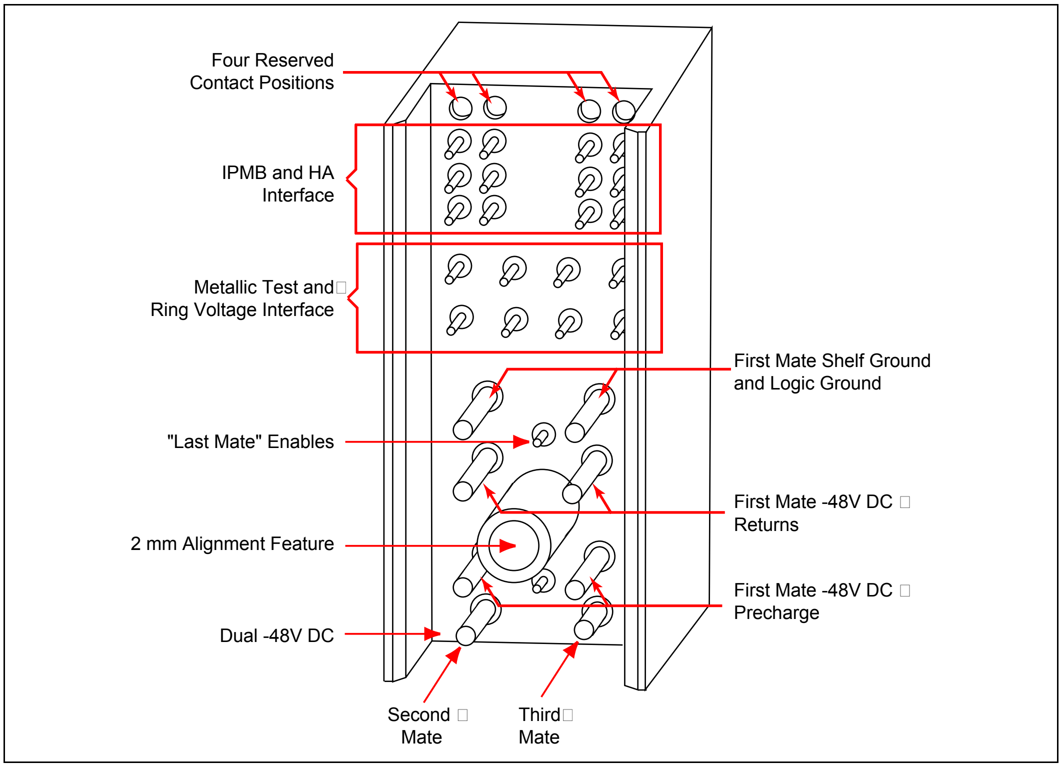


Table 15. Power distribution connector (Zone 1) P10 pin assignments

Pin	Signal	Description
1	Reserved	No Connect
2	Reserved	No Connect
3	Reserved	No Connect
4	Reserved	No Connect
5	GA0	Geographic Addr Bit 0
6	GA1	Geographic Addr Bit 1
7	GA2	Geographic Addr Bit 2
8	GA3	Geographic Addr Bit 3
9	GA4	Geographic Addr Bit 4
10	GA5	Geographic Addr Bit 5
11	GA6	Geographic Addr Bit 6
12	GA7/P	Geo Adr Bit 7 (Odd Parity)
13	IPMB_CLK_A	IPMB Bus A Clock
14	IPMB_DAT_A	IPMB Bus A Data
15	IPMB_CLK_B	IPMB Bus B Clock
16	IPMB_DAT_B	IPMB Bus B Data
17	Unused	No Connect
18	Unused	No Connect
19	Unused	No Connect
20	Unused	No Connect
21	Unused	No Connect
22	Unused	No Connect
23	Unused	No Connect
24	Unused	No Connect
25	EMI_GND	EMI Chassis Ground
26	LOGIC_GND	Gnd Ref for Card Logic
27	ENABLE_B	Enb DC-DC conv, B Feed
28	VRTN_A	-48 V Return, Feed A
29	VRTN_B	-48 V Return, Feed B
30	-48V_EARLY_A	No Connect
31	-48V_EARLY_B	No Connect
32	ENABLE_A	Enb DC-DC conv, A Feed
33	-48V_A	-48 V Input, Feed A
34	-48V_B	-48 V Input, Feed B

3.6.2 AdvancedTCA Data Transport Connectors (J20, J23)

Zone 2 consists of two 120-pin HM-Zd connector, labeled J20 and J23, with 40 differential pairs. J20 provides Update Channel connectivity, while J23 provides base interface and fabric interface GbE ports connectivity.

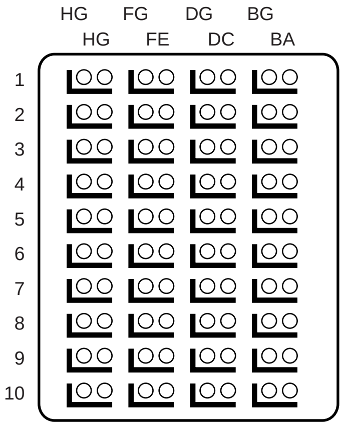
3.6.2.1 Update Channel connector (J20)

This connector provides following signals:

- One 100MHz PCI clock (UpdateCLK)
- x4 PCI-Ex from ICH (Intel® 6321ESB):
  Update Channel Port 0 - PCI-Ex form ESB2 - Lane 0 (TX0, RX0)
  Update Channel Port 1 - PCI-Ex form ESB2 - Lane 1 (TX1, RX1)
  Update Channel Port 2 - PCI-Ex form ESB2 - Lane 2 (TX2, RX2)
  Update Channel Port 3 - PCI-Ex form ESB2 - Lane 3 (TX3, RX3)

The connector used is an AMP*/Tyco* part number 1469001-1 (Intel part number A66621-005). Figure 19 shows a face view of the connector.

Figure 19. Data transport connector J20 (Zone 2)



B0899-01

— The BG, DG, FG, and HG (G for Ground) columns contain the ground shields for the four columns of differential pairs. They have been omitted from the pinout tables below for simplification. All pins in the BG, DG, FG, and HG columns are connected to logic ground.

Table 16. AdvancedTCA data transport connector (Zone 2) J20 pin assignments

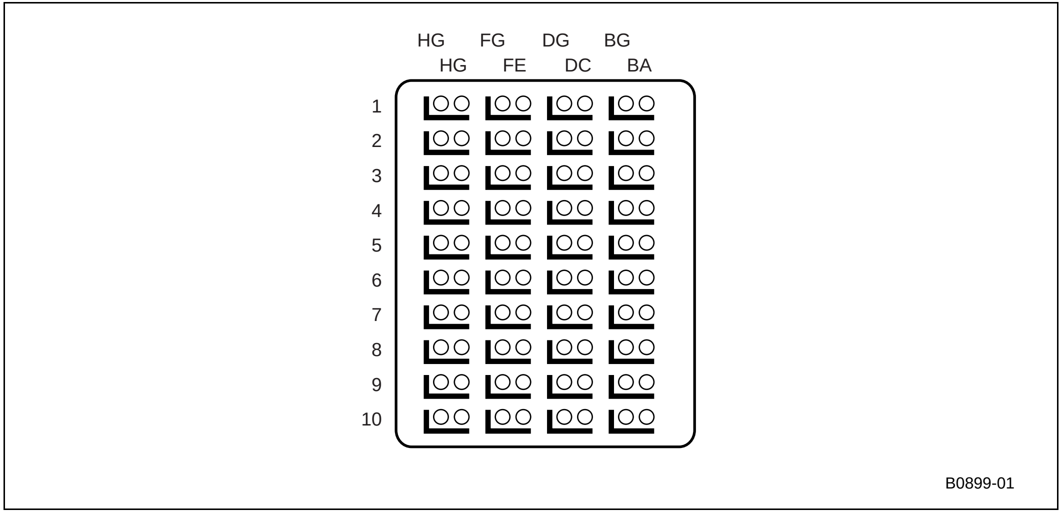
Pin	A	B	C	D	E	F	G	H
1	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
2	UpdateClk+	UpdateClk-	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
3	Tx2(UP+)	Tx2(UP-)	Rx2(UP+)	Rx2(UP-)	Tx3(UP+)	Tx3(UP-)	Rx3(UP+)	Rx3(UP-)
4	Tx0(UP+)	Tx0(UP-)	Rx0(UP+)	Rx0(UP-)	Tx1(UP+)	Tx1(UP-)	Rx1(UP+)	Rx1(UP-)
5	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect
6	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect
7	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect
8	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect
9	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect
10	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect

3.6.2.2 Base and Fabric Interface ports connector (J23)

The connector provides following signals.

- Two 10/100/1000BASE-T/TX Ethernet base channels (4 differential signal pairs each, 16 signals total)
- Two 1000BASE-BX Ethernet fabric channels (4 differential signal pairs each, 8 signals total)

Figure 20. Data transport connector J23 (Zone 2)



The BG, DG, FG, and HG (G for Ground) columns contain the ground shields for the four columns of differential pairs. They have been omitted from the pinout tables below for simplification. All pins in the BG, DG, FG, and HG columns are connected to logic ground.

Table 17. AdvancedTCA data transport connector (Zone 2) J23 pin assignments

Pin	A	B	C	D	E	F	G	H
1	No Connect	No Connect	Terminated	Terminated	No Connect	No Connect	Terminated	Terminated
2	F[2]Tx0+	F[2]Tx0-	F[2]Rx0+	F[2]Rx0-	F[2]Tx1+	F[2]Tx1-	F[2]Rx1+	F[2]Rx1-
3	No Connect	No Connect	Terminated	Terminated	No Connect	No Connect	Terminated	Terminated
4	F[1]Tx0+	F[1]Tx0-	F[1]Rx0+	F[1]Rx0-	F[1]Tx1+	F[1]Tx1-	F[1]Rx1+	F[1]Rx1-
5	BI_DA1+ (Tx1+)	BI_DA1- (Tx1-)	BI_DB1+ (Rx1+)	BI_DB1- (Rx1+)	BI_DC1+	BI_DC1-	BI_DD1+	BI_DD1-
6	BI_DA2 + (Tx2+)	BI_DA2- (Tx2-)	BI_DB1+ (Rx2+)	BI_DB1- (Rx2-)	BI_DC2+	BI_DC2-	BI_DD2+	BI_DD2-
7	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect
8	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect
9	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect
10	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect

░ Fabric interface (Gigabit Ethernet) ports
▒ Base interface (Gigabit Ethernet) ports

Note: All "Terminated" pins are grounded on the baseboard as defined in the PICMG 3.1, Release 1.0 specification.

The following naming convention describes the signals on this connector. Signal direction is defined from the perspective of the MPCBL0050 SBC.

For the base interface, the bi-directional 10/100/1000BASE-T data signals have the following conventions:

BI_Dr[c]p
r = differential pair (A, B, C, or D)
c = channel (1, 2)
p = polarity (+, -)

For the fabric interface, the 1000BASE-BX data signals have the following conventions:

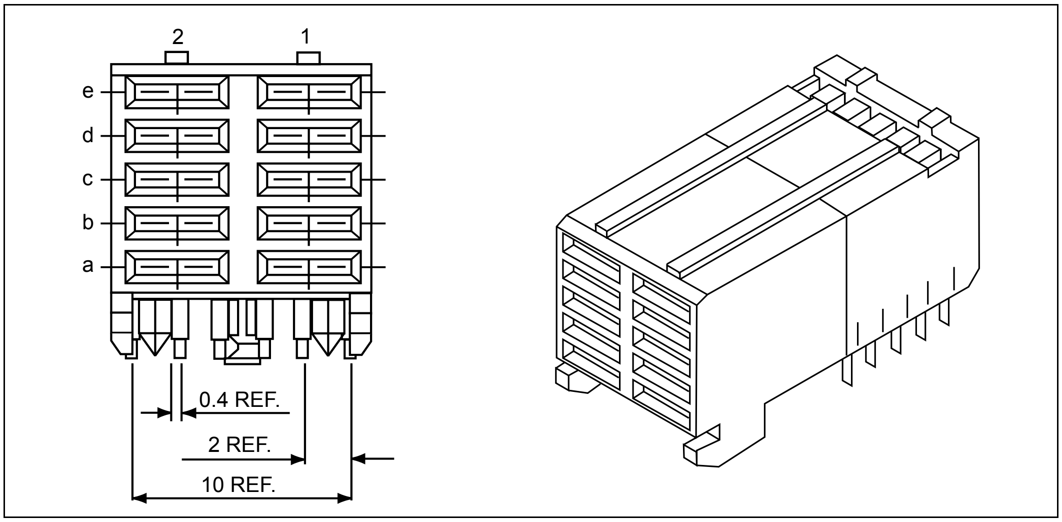
F[c]dnp
c = channel (1, 2)
d = direction (Tx = Transmit, Rx = Receive)
n = port number (0, 1)
p = polarity (+, -)

A port is two differential pairs; one Tx and one Rx.

3.6.3 Zone 3 Rear Transition Module Power Connector (J30)

The J30 connectors are bladed connectors originally developed for FutureBus* applications.

Figure 21. J30 connector



The signals are arranged as shown in Table 18.

Table 18. J30 pinout

Pin	Signal	Pin	Signal
E2(S)	ENABLE#	E1(S)	PS1#
D2(S)	12V	D1(S)	12V
C2(M)	IPMI_Sdata	C1(M)	IPMI_Sclk
B2(L)	+3.3V_MP	B1(L)	Logic_GND
A2(L)	Shelf_GND	A1(L)	Logic_GND

Table 19. J30 signal descriptions (Sheet 1 of 2)

Pin	Signal	Comments
A1	Logic_GND	Logic ground connection (long contact); provides return path for power and signal connections.
A2	Shelf_GND	Shelf ground connection (long contact); provides safety ground contact between SBC and RTM.
B1	Logic_GND	Logic ground connection (long contact); see A1 above.
B2	+3.3V_MP	Management power (long contact); provides up to 100 mA to power management system on RTM. Used exclusively for management power.
C1	IPMI_Sclk	IPMB/I^{2}C clock signal (medium contact); this provides the clock signal for the two-wire IPMB interface.
C2	IPMI_Sdata	IPMB/ I^{2}C data signal (medium contact); this provides the data signal for the two-wire IPMB interface.

Table 19. J30 signal descriptions (Sheet 2 of 2)

Pin	Signal	Comments
D1	12V	12V RTM payload power (short contact); provides power to active devices (other than management system) on the RTM. See additional requirements below.
D2	12V	
E1	PS1#	Presence Signal, active low (short contact); the RTM connects this signal to Logic_GND through a 100 Ohm resistor (to facilitate manufacturing test). The SBC reads this signal to understand if an RTM is fully inserted.
E2	ENABLE#	Module enable signal, active low (short contact); the SBC sets this signal high to reset the RMC (RTM Management Controller).

3.6.4 Zone 3 Rear Transition Module Data/Control Connector (J31)

The MPCBL0050 SBC implementation includes an RTM connector (J31) that mates directly to the RTM without connecting through the backplane. The Zone 3 connector J30 consists of one 120-pin HM-Zd connector with 40 differential pairs, which allows high-speed signals to be passed between the boards. The signals that are routed through J31 are the IPMC signals, IEEE 1149.1 JTAG signals, SAS storage ports, USB 2.0 signals, and serial port.

Table 20. AdvancedTCA RTM connector (Zone 3) J31 Pinouts

Pin	A	B	C	D	E	F	G	H
1	RMD_INT#	TRTST-	TCLK	TDI	TMS	TDO	Reserved	No Connect
2	SA[0]TX+	SA[0]TX-	SA[0]RX+	SA[0]RX-	SA[1]TX+	SA[1]TX-	SA[1]RX+	SA[1]RX-
3	SA[2]TX+	SA[2]TX-	SA[2]RX+	SA[2]RX-	No Connect	No Connect	No Connect	No Connect
4	Eth0_DA+	Eth0_DA-	Eth0_DB+	Eth0_DB-	Eth0_DC+	Eth0_DC-	Eth0_DD+	Eth0_DD-
5	Eth1_DA+	Eth1_DA-	Eth1_DB+	Eth1_DB-	Eth1_DC+	Eth1_DC-	Eth1_DD+	Eth2_DD-
6	Eth0_Link	Eth0_Act	Eth0_Spd1000	Eth1_Link	Eth1_Act	Eth1_Spd1000	PCI_INTA	PCI_RESET
7	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	PCIe_Clk_1+	PCIe_Clk_1-
8	P0_Tx+	P0_Tx-	P0_Rx+	P0_Rx-	P01_Tx+	P1_Tx-	P1_Rx+	P1_Rx-
9	P2_Tx+	P2_Tx-	P2_Rx+	P2_Rx-	P03_Tx+	P3_Tx-	P3_Rx+	P3_Rx-
10	USB[0]+	USB[0]-	DSR#	RXD#	RTS	TXD	CTS#	DTR

Table 21. J31 signal descriptions (Sheet 1 of 2)

Pin	Signal	Comments
A1	RMD_INT	Reserved
B1	TRST#	Test Reset signal as defined in JTAG (IEEE 1149.1). The SBC's main JTAG chain must connect to this signal.
C1	TCLK	Test Clock signal as defined in JTAG. Required on SBCs and RTMs with JTAG-enabled devices.
D1	TDI	Test Data In signal as defined in JTAG. SBCs must connect this signal into the test data chain (that is, in line with TDO connections from other chips), but must have a means to bypass this connection if an RTM is not installed.
E1	TMS	Test Mode State signal as defined in JTAG. Required on SBCs and RTMs with JTAG-enabled devices.
F1	TDO	Test Data Out signal as defined in JTAG. See TDI comments above. Output of RTM.
G1-H1	RSVD	Reserved.

Table 21. J31 signal descriptions (Sheet 2 of 2)

Pin	Signal	Comments
A2-D3	SA[x]TX+, SA[x]TX-, SA[x]RX+, SA[x]RX-	Serial Attached SCSI signals for transmit and receive portions of differential pairs. Three SAS ports are routed to the RTM.
A4 -H4	Eth0_D[x]	GbE portd C redirected from front board.
A5 -H5	Eth1_D[x]	GbE port D redirected from front board.
A6-C6	Eth0_Link,/Act/Spd	LEDs for GbE port C
D6-F6	Eth1_Link,/Act/Spd	LEDs for GbE port D
G6	PCI_INTA	PCI INT
H6	PCI_RESET#	PCI reset signal required on all SBCs supporting PCI Express, but optional for RTMs. This signal is used by devices that need to provide a PCI interrupt but cannot always rely on MSI (Message Signaled Interrupt) to be implemented properly.
G7-H7	PCIe_Clk_1	PCI-Ex x4 Clock
A8-H9	P[x]TX/Rx	PCI-Ex x4 lanes connected to ICH
A10-B10	USB[0]+, USB[0]-	USB data signals. Note that the RTM's 5 V power for the USB connections must be derived from the 12 V rail.
C10	DSR#	Data Set Ready signal for COM1 RS-232 connection.
D10	RXD#	Received Data signal for COM1 RS-232 connection.
E10	RTS#	Ready to Send signal for COM1 RS-232 connection.
F10	TXD#	Transmit Data signal for COM1 RS-232 connection.
G10	CTS#	Clear to Send signal for COM1 RS-232 connection.
H10	DTR#	Data Terminal Ready signal for COM1 RS-232 connection.

3.6.5 Zone 3 Rear Transition Module Data Connector (J32)

This J32 connector is used to route signals from the AdvancedMC slots to the RTM.

Table 22. AdvancedTCA RTM connector (Zone 3) J32 pinouts

Pin	A	B	C	D	E	F	G	H
1	AP1[0]TX+	AP1[0]TX-	AP1[0]RX+	AP1[0]RX-	AP1[1]TX+	AP1[1]TX-	AP1[1]RX+	AP1[1]RX-
2	AP1[2]TX+	AP1[2]TX-	AP1[2]RX+	AP1[2]RX-	AP1[3]TX+	AP1[3]TX-	AP1[3]RX+	AP1[3]RX-
3	No Connect	No Connect	No Connect	No Connect	AP1[5]TX+	AP1[5]TX-	AP1[5]RX+	AP1[5]RX-
4	AP1[6]TX+	AP1[6]TX-	AP1[6]RX+	AP1[6]RX-	AP1[7]TX+	AP1[7]TX-	AP1[7]RX+	AP1[7]RX-
5	No Connect	No Connect	No Connect	No Connect	SA_Tx[0]+	SA_Tx[0]-	SA_Rx[0]+	SA_Rx[0]-
6	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect
7	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect
8	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect
9	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect
10	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect

AP0[x] are signals routed from AdvancedMC to the RTM.

- AMC_Port 20 is mapped to AP0[0]
- AMC_Port19 to AP0[1]
- AMC_Port18 to AP0[2]
- AMC_Port17 to AP0[3]

- AMC_Port15 to AP0[5]
- AMC_Port14 to AP0[6]
- AMC_Port13 to AP0[7]

SA_Rx/Tx ports are mapped to AMC SAS port3.

3.6.6 Zone 3 Rear Transition Module Data/Control Connector (J33)

The MPCBL0050 SBC implementation includes an RTM connector (J31) that mates directly to the RTM without connecting through the backplane. The Zone 3 connector J30 consists of one 120-pin HM-Zd connector with 40 differential pairs, which allows high-speed signals to be passed between the boards. The signals that are routed through J31 are the IPMC signals, IEEE 1149.1 JTAG signals, SAS storage ports, USB 2.0 signals, and serial port.

Table 23. AdvancedTCA RTM connector (Zone 3) J33 pinout

Pin	A	B	C	D	E	F	G	H
1	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect
2	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect
3	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect
4	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect
5	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect
6	Eth2_DA+	Eth2_DA-	Eth2_DB+	Eth2_DB-	Eth2_DC+	Eth2_DC-	Eth2_DD+	Eth2_DD-
7	Eth3_DA+	Eth3_DA-	Eth3_DB+	Eth3_DB-	Eth3_DC+	Eth3_DC-	Eth3_DD+	Eth3_DD-
8	Eth2_Link	Eth2_Act	Eth2_Spd1000	Eth3_Link	Eth3_Act	Eth3_Spd1000	Reserved	Reserved
9	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect	No Connect
10	No Connect	No Connect	No Connect	No Connect	Reserved	Reserved	Reserved	Reserved

Table 24. J33 signal descriptions

Pin	Signal	Comments
A6 -H6	Eth2_D[x]	GbE port E redirected from front board.
A7 -H7	Eth1_D[x]	GbE port F redirected from front board.
C8-E8	Eth2_Link,/Act/Spd	LEDs for GbE port E
F8-H8	Eth3_Link,/Act/Spd	LEDs for GbE port F

3.6.7 Alignment Blocks

Tyco 1-1469373-1 (or equivalent). The Zone 3 alignment block (K2) is assigned a keying value of 73, and uses Tyco 7-1469373-3 (or equivalent).

The MPCBL0050 SBC implements the K1 and K2 alignment blocks at the top of Zone 2 and Zone 3, as required in Section 2.4.4 of the PICMG 3.0 Specification. The Zone 2 alignment block (K1) is assigned a keying value of 11, and uses Tyco* 1-1469373-1 (or equivalent). The Zone 3 alignment block (K2) is assigned a keying value of 73, and uses Tyco 7-1469373-3 (or equivalent).

4.0 Operating the Unit

4.1 Board Configuration DIP Switches

The MPCBL0050 provides several DIP switches that allow the user to configure certain options not configurable through the EFI BIOS setup utility. The MPCBL0050 is shipped pre-configured and jumper positions do not generally need to be altered. For detailed information on DIP switch configuration, see Section 3.4.

4.2 Installing and Extracting the SBC

The MPCBL0050 SBC has two ejector handles to help insert the board into and eject the board from the chassis.

4.2.1 Insertion

To insert the board into the chassis, open the ejector handles and push the board into the chassis. Once the ejector handles slide into the top and bottom notches in the chassis, push both ejector handles toward the faceplate of the board until the handles click into place. The ejector handles provide a positive cam action, which ensures that the blade is properly seated. Once the bottom ejector handle is closed, the Hot Swap switch is engaged and this starts the normal power-on sequence.

Note: Make sure that the plastic slider on the bottom ejector handle is in the low position after inserting the SBC. You can do that by pressing the slider down after board is fully inserted. This will assure proper enabling of hot swap microswitch, and board will start booting.

4.2.2 Ejection

To eject the board from the chassis, slide the bottom ejector handle mechanism so it releases from the faceplate and then gently pull the bottom ejector handle away from the faceplate. When the lower ejector handle is disengaged from the faceplate, the Hot Swap switch is released and this starts the normal board shutdown process. This power down process is identified by the blinking blue Hot Swap LED. Once the Hot Swap LED turns solid blue, slide the top ejector handle mechanism so it releases from the faceplate and use both ejector handles to disengage the board from the backplane.

See Section 5.10 for detailed information on the function and operation of the Hot Swap LED.

Warning: Removing the SBC before the Hot Swap LED is solid blue can lead to device corruption or failure.

4.3 AdvancedMC Module Installation

Install the AdvancedMC module as follows:

1. Install AdvancedMC module prior to board payload power-on or after the operating system is fully loaded.
2. Remove the AdvancedMC filler panel by pulling on the ejector handle until the module is removed.
3. Insert the AdvancedMC module into the slot and close the ejector handle by pushing it toward the faceplate of the board.

Note: AdvancedMC modules must be installed before the board payload powers on or after the operating system has fully loaded. If an AdvancedMC module is hot added after the initial EFI BIOS device discovery, but before the operating system is fully loaded, then the AdvancedMC module may not work properly and the SBC may stop booting. To correct this, a board payload reset or payload power cycle is required.

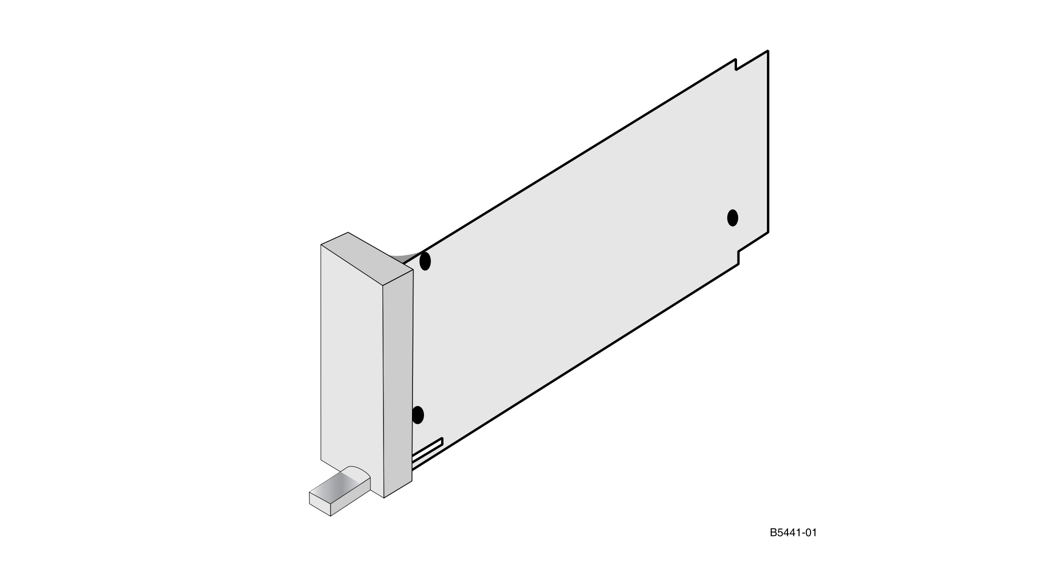
Note: AdvancedMC modules are not included with the MPCBL0050 board and must be purchased separately. Refer to the MPCBL0050 Compatibility Report for a list of devices validated.

4.4 AdvancedMC Filler Panels

AdvancedMC* filler panels are used to optimize cooling and reduce radiated emissions when AdvancedMC modules are not installed in theMPCBL0050 AdvancedMC module slots.

Caution: Do not operate the MPCBL0050 without filler panels or AdvancedMC modules installed. AdvancedMC module slots should not be left open or uncovered when the MPCBL0050 is in use.

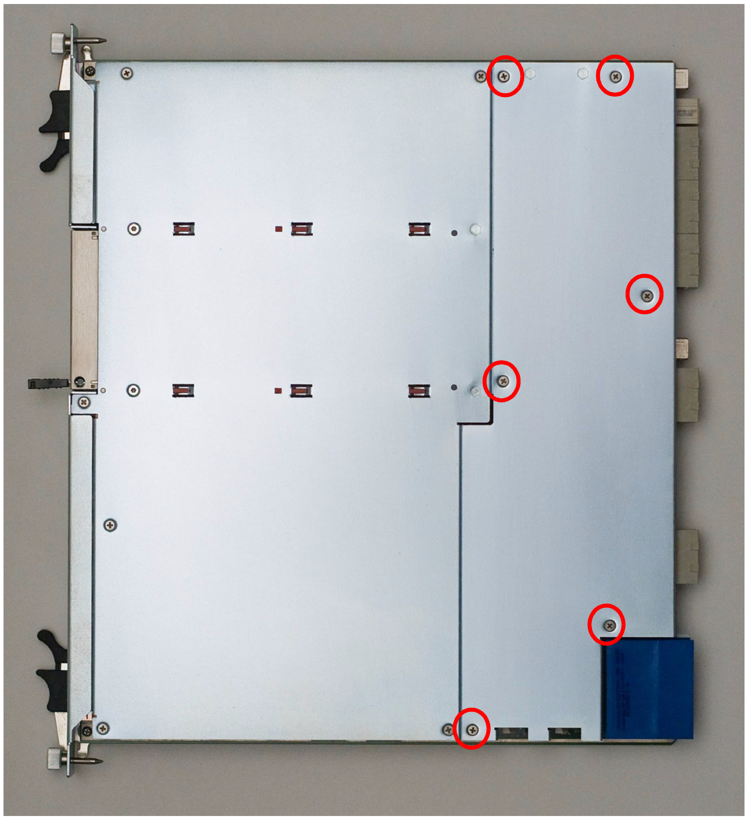
Figure 22. AdvancedMC filler panel



4.5 Memory (FBDIMM) Installation

The top cover of the board has a separate cover to allow easy access to FBDIMM modules. The screw locations are marked with red circles in Figure 23.

Figure 23. Top cover screw locations

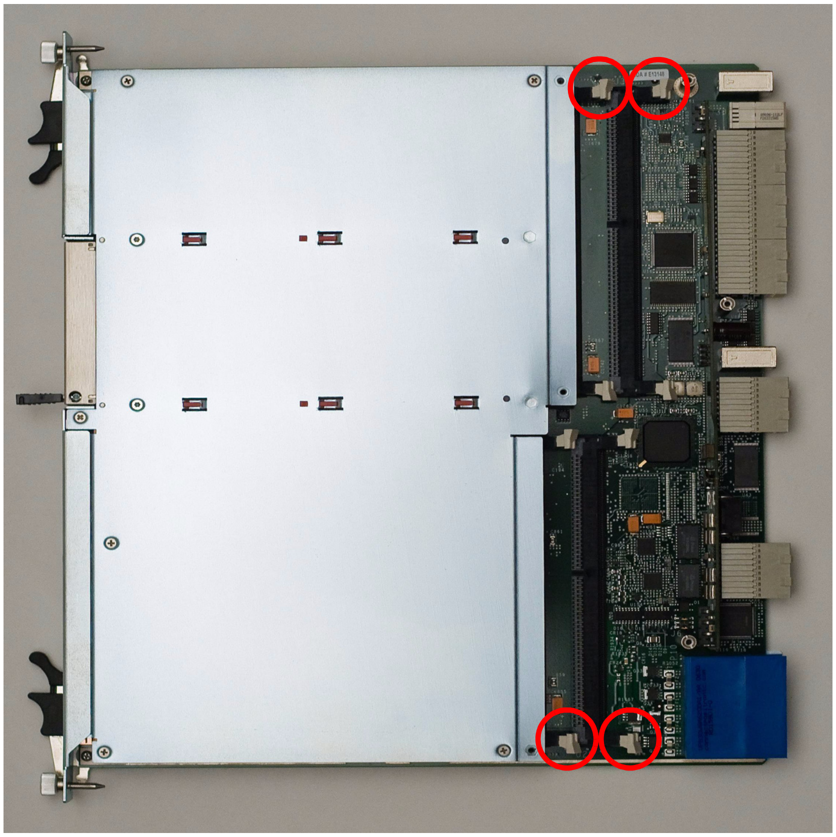


Install the FBDIMM modules as follows:

1. Unscrew the top metal cover where the FBDIMMs will be installed. Refer to Figure 23 for the location of the FBDIMM cover screws.
2. Open the FBDIMM ejector handles.
3. Install four FBDIMMs. FBDIMMs must be installed in matched pairs. FBDIMMs need to be identical in rank, size, device width, and memory timing. For details on memory population rules, see Table 4.
4. Re-attach the FBDIMM sheet metal cover. Torque the top cover screws to 4 in-lb (0.45N-m) using a torque screwdriver.

Caution: Before reattaching the top cover, ensure that the DIMM ejector latches are closed. If they are left open, installing of the top cover will not be possible. Figure 24 shows the latches left in the closed (correct) position.

Figure 24. Latches left in the closed (correct) position



Note: Memory modules are not included with the MPCBL0050 board and must be purchased separately. Refer to the MPCBL0050 Compatibility Report for a list of devices validated.

4.6 Rear Transition Module (RTM) Installation

In order to boot the MPCBL0050 from SAS HDD provided on MPRTM0050 install the Rear Transition module as follows:

1. Install the MPRTM0050 Rear Transition Module from the rear of the chassis.
2. Close the RTM ejector handles and tighten the RTM faceplate screws so the board is firmly seated in the chassis.
3. Install the MPCBL0050 SBC from the front of the chassis according to the installation notes in Section 4.2.

Note: The MPRTM0050 is not included with the MPCBL0050 board and must be purchased separately. Refer to the MPCBL0050 Compatibility Report for a list of devices validated.

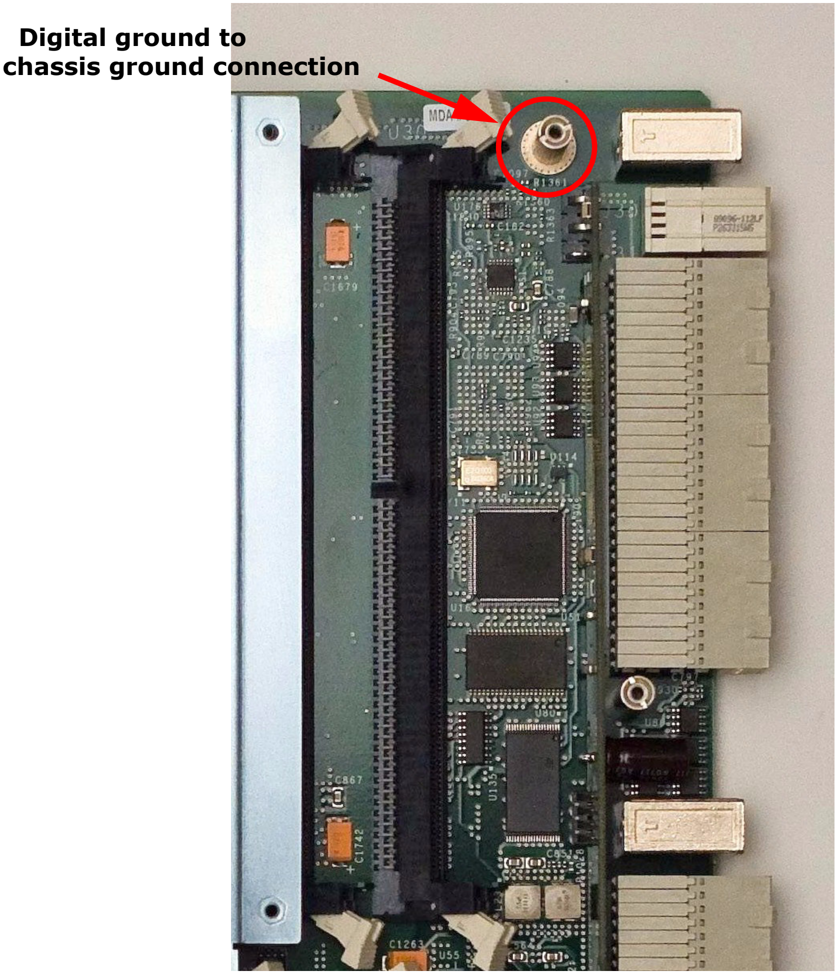
4.7 Digital Ground to Chassis Ground Connectivity

By default, the board is shipped with the digital ground connected to the chassis ground by a metal standoff.

Warning: To meet safety requirements, this configuration MUST be used if the board is operating in an environment with an operating voltage at or above -60V. For further explanation of this matter, refer to the *Official Customer Notification of Changes to the Default Grounding Configuration of MPCBL0050 Products* document included with the MPCBL0050.

If the board operates with a standard -48V power supply, however, this connection is not required and can be removed using the procedure below.

Figure 25. Digital ground to chassis ground standoff location



To disconnect the digital ground from the chassis ground:

1. Confirm that the board and service person are appropriately grounded prior to removing the board from the ESD bag.
2. Using a Philips head screwdriver, remove the screws from the memory top cover and then from the cover itself (see Section 4.5 for details on cover removal).
3. Using a flat-head screw driver, remove the standoff (metal cylinder) located beside the RTM power connector J3 under the FBDIMM cover, as shown in Figure 25.
   — To avoid confusion, now put the removed standoff away somewhere safely. It will not be needed further in this procedure. The removed standoff would only be needed again to reverse the current procedure (to once again connect the digital ground to the chassis ground).

4. From the plastic parts bag included in the box with the MPCBL0050, remove the replacement metal standoff and white plastic washer.
   — Be sure not to confuse the removed standoff with the replacement standoff. They are similar but not identical.
5. Slide the washer on to the threaded end of the replacement standoff.
6. With the plastic washer in place, carefully thread the new standoff into the board where the old standoff was removed in step 3. Using a flat head torque screwdriver, tighten the standoff/washer combination into place with a torque of 4 in-lbs.
   — With the white washer properly in place with the replacement standoff, the digital ground is now isolated from the chassis ground.
7. Replace the memory top cover that was removed in step 2.

Note: A digital ground is also known as a logic ground. A chassis ground is also known as a shelf ground.

4.8 EFI BIOS Configuration

In most cases, the EFI BIOS defaults provide the correct configuration for board use. However, a user that needs to make changes to the default EFI settings can refer to Chapter 7.0, "EFI BIOS Setup," for a complete description of the EFI options.

4.9 Remote Access Configuration

Console redirection to the front panel serial port is enabled by default. This setting redirects the text output of the EFI BIOS and operating system to the RJ-45 serial port on the MPCBL0050 faceplate. Remote access using serial console redirection allows users to monitor the boot process and run the EFI BIOS setup from a remote serial terminal. The default settings are 115200, N, 8, 1 with no flow control. Use these settings to access the console data. See Section 3.1.2 for the pin description of this interface.

4.10 Boot Devices

The EFI BIOS Setup program includes a choice of available boot devices, with each boot device having options for removable media (USB CD-ROM, USB flash disk, etc.), SAS hard drive, on-board PATA flash drives, or PXE boot through any of the six GbE adapters.

In every POST, the EFI BIOS detects all available boot devices and displays them on the boot order screen.

Refer to Section 7.6 for detailed information on selecting boot devices.

4.10.1 Booting from a SAS Hard Disk

If the MPRTM0050 RTM equipped with SAS HDD is installed the HDD can be used for booting the operating system. Refer to Section 7.6 for detailed information on selecting boot devices.

4.10.2 Booting from a PATA Flash (On-board)

Two Parallel ATA flashes are available on the board. Each of them can be used for booting operating system.

Note: By default PATA flash drives are disabled in EFI BIOS configuration. In order to enable it refer to Section 7.6.

4.10.3 Booting from a USB Device

The board can boot from a USB device that is plugged into the MPCBL0050 board or the MPRTM0050 RTM. The most common USB devices used for booting are USB hard drive, USB CD-ROM or USB flash disk.

If a USB device is attached to the board during boot up, the EFI BIOS detects the device and it appears in the boot device list on the EFI BIOS Setup menu.

If the USB device is non-bootable, the EFI attempts to boot from the next boot device as configured in the Boot Device Priority list.

Note: For information on how to create a bootable USB flash disk, search the web for "create bootable usb flash". There are multiple web sites that have information on how to do this.

Refer to Section 7.6 for detailed information on selecting boot devices.

4.10.4 Booting from a LAN (PXE Boot)

Any of the six on-board Gigabit Ethernet LAN adapters on the base and fabric interfaces can be used for the remote boot process. It is necessary to set up a PXE boot server on which to store the Linux* image. (Details on how to set up a PXE boot server are beyond the scope of this document.) Remember to configure the boot device priority in the EFI Setup to specify the Gigabit Ethernet LAN adapter(s) as the top priority.

An Ethernet switch must be installed in your AdvancedTCA* chassis in order to boot from LAN.

It is possible to PXE boot from AdvancedMC modules, but the AdvancedMC module installed into the MPCBL0050 must support the PXE boot agent.

Refer to Section 7.6 for detailed information on selecting boot devices. For a block diagram, see Figure 5.

Table 25. Ethernet port mapping

BIOS PXE Boot Port *	Identification on MPCBL0050 Faceplate	MAC Address	PCI Express Port	PICMG 3.0 Port Definition
1400	Port A	xxxxxxxxxxxN	ICH	Base Interface; Port 0, Channel 1,
1401	Port B	xxxxxxxxxxx (N+1)	ICH	Base Interface; Port 0, Channel 2
1700	Port C	xxxxxxxxxxx (N+2)	MCH Port 4	Fabric Interface; Port 0, Channel 1 (Ports C and D can be jointly selected to connect to the backplane or to the RJ45 connectors on the RTM (optional)

* BIOS PXE Boot Port xxyy refers to::

- xx = PCI bus number, for example 02 means bus 2.
- yy = Bits 7-3 is the PCI device number; bits 2-0 is the PCI function number. For example, 00 means device 0, function 0; 01 means device 0, function 1 etc.

Note: For the Ethernet operating system port, this is the default unless the default parameters are changed. For example, pci=xxx boot parameter.

Table 25. Ethernet port mapping

BIOS PXE Boot Port *	Identification on MPCBL0050 Faceplate	MAC Address	PCI Express Port	PICMG 3.0 Port Definition
1701	Port D	xxxxxxxxxxx (N+4)	MCH Port 4	Fabric Interface; Port 0, Channel 2 (Ports C and D can be jointly selected to connect to the backplane or to the RJ45 connectors on the RTM (optional))
1800	Port E	xxxxxxxxxxx (N+3)	MCH Port 5	Fabric Interface; Port 1, Channel 1 (Ports E and F can be jointly selected to connect to backplane (default setting) front panel RJ45 or RTM RJ45.)
1801	Port F	xxxxxxxxxxx (N+5)	MCH Port 5	Fabric Interface; Port 1, Channel 2 (Ports E and F can be jointly selected to connect to backplane (default setting) front panel RJ45 or RTM RJ45.)

* BIOS PXE Boot Port xxyy refers to::

- xx = PCI bus number, for example 02 means bus 2.
- yy = Bits 7-3 is the PCI device number; bits 2-0 is the PCI function number. For example, 00 means device 0, function 0; 01 means device 0, function 1 etc.

Note: For the Ethernet operating system port, this is the default unless the default parameters are changed. For example, pci=xxx boot parameter.

4.11 Identifying MPCBL0050 Ethernet MAC Addresses

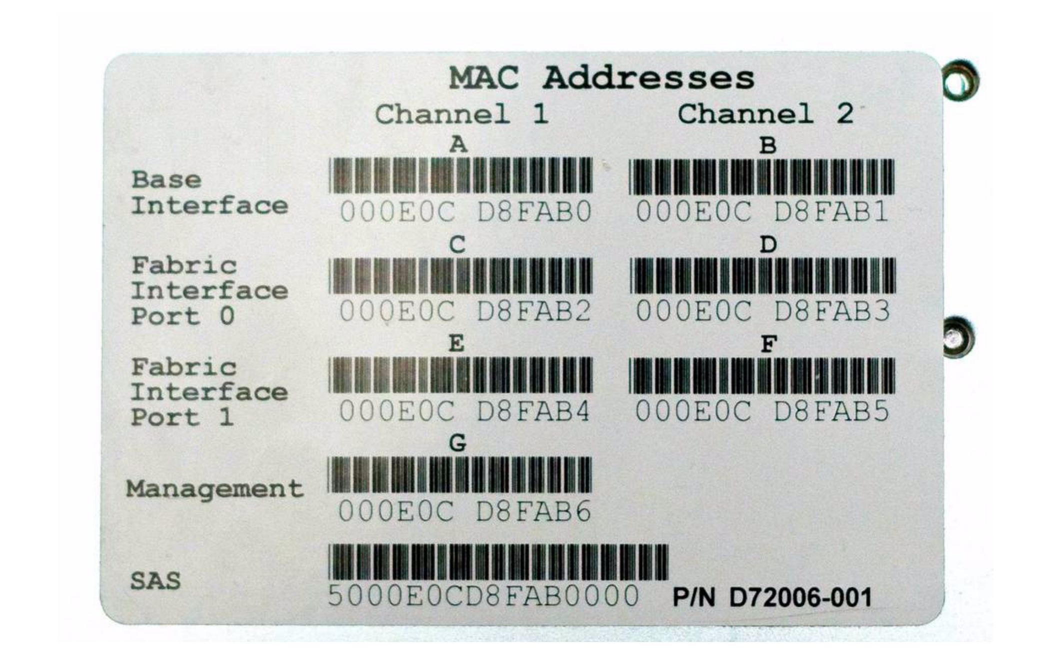
On the backing plate of the MPCBL0050, there is a MAC address label to assist a user in determining the MAC address of each Ethernet port on the SBC. The MAC address may be needed for PXE boot configuration, configuring Serial Over LAN (SOL) and for OS configuration.

Figure 26 shows the location of the MAC address label on the MPCBL0050 backing plate. Table 25 breaks down the mapping of the Ethernet interfaces to the labeling on the MPCBL0050 faceplate and in the operating system.

The first MAC address (Port A) can be retrieved remotely by using IPMI command. From that first MAC address (port A), each port increments by 0x01h.

In the IPMI specification (23.2), parameter #5 (MAC address) of the Get LAN Configuration Parameters Command will be populated automatically with the base interface MAC address. Alternatively, using ipmitool from the payload processor, the command is: “ipmitool lan print 1”

Figure 26. MAC address label on MPCBL0050 board



MAC Addresses

	Channel 1	Channel 2
	A	B
Base Interface	000E0C D8FAB0	000E0C D8FAB1
	C	D
Fabric Interface Port 0	000E0C D8FAB2	000E0C D8FAB3
	E	F
Fabric Interface Port 1	000E0C D8FAB4	000E0C D8FAB5
	G	
Management	000E0C D8FAB6	
SAS	5000E0CD8FAB0000	P/N D72006-001

4.12 Cable Information

For the front panel ports, here are details on the cables that need to be used with the board:

Table 26. Cable information

Type	Cable Description	Max Length
Ethernet Port	Intra-building (same building) only: Shielded Ethernet cable SFTP5e.	100m
Serial Port	Intra-building wiring (cabling) only, directly connects equipment within the same frame, cabinet or line-up and where equipment is separated by a distance of 6 m or less. Shielded Category 5e cable with DB-9 to RJ45 converter.	6m
USB Port	External USB shielded cable	5m

4.13 Firmware Updates

The major components that need updates are the EFI and IPMC firmware. See Section 10.0 for more information on firmware updates.

5.0 Hardware Management

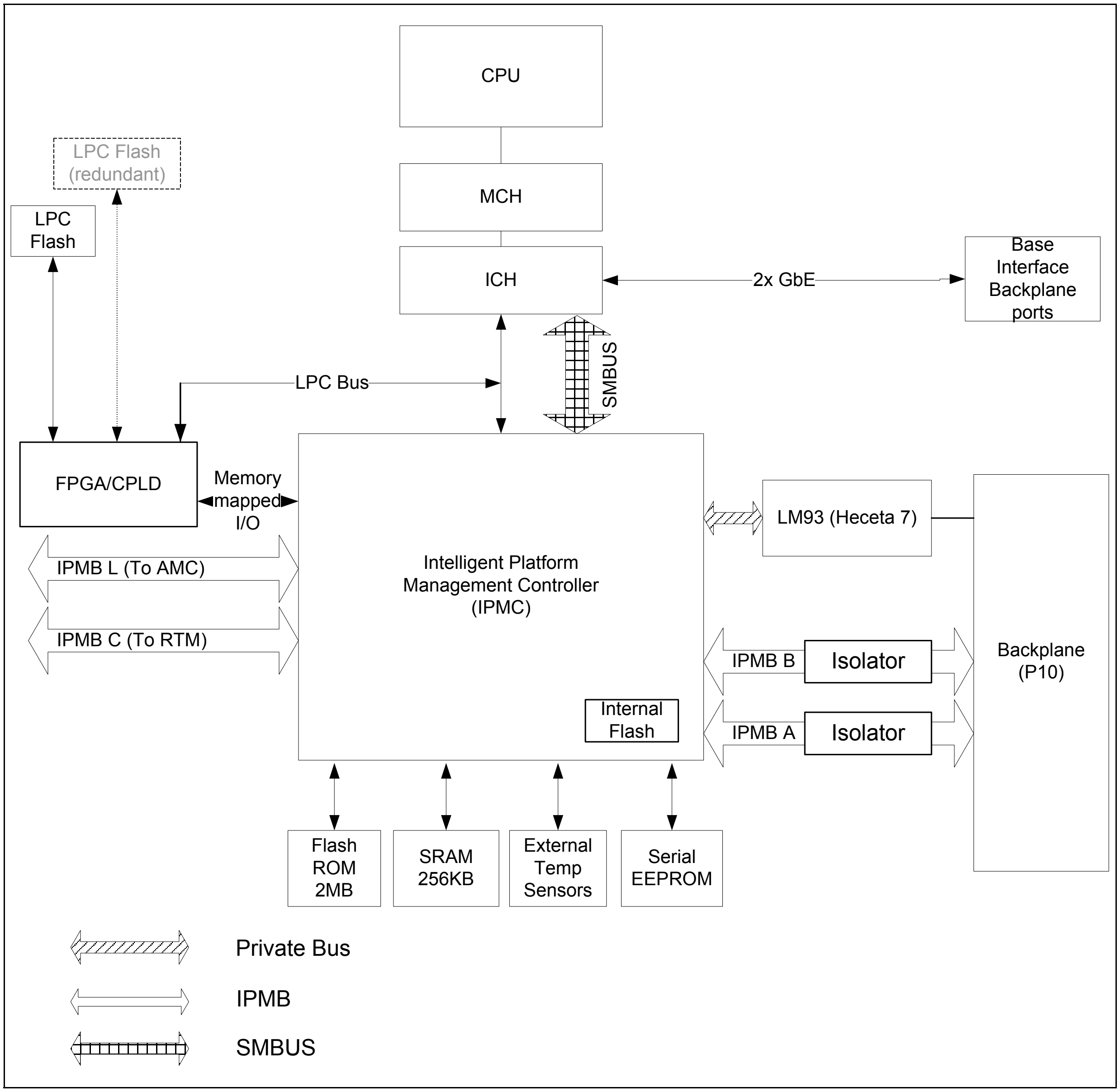
The Intelligent Platform Management Controller (IPMC) is based on the Renesas* microcontroller, model HD64F2166. This controller is a 16-bit processor with 40 KBytes of internal RAM and 512 kBytes of internal flash. It supports up to six I^2C buses (master/slave), three serial-ports, a low-pin count (LPC) interface, and an A/D and DAC. This microcontroller is also capable of addressing up to 16 MBytes and has external access to 2 MBytes of flash memory; 512 KBytes of SRAM and a 64 KByte serial EEPROM.

The IPMC implementation on the MPCBL0050 conforms to PICMG 3.0 R2.0 ECN002 specifications. IPMI defines a standardized, abstracted, message-based in-band and out-of-band interfaces between system management software and the platform management hardware. The manageability framework for the MPCBL0050 is developed per the IPMI v2.0 Specification, which includes the following:

- Intelligent Platform Management Controller (IPMC)
- IPMI messaging, commands, and abstractions
- IPMI channels, and sessions
- Sensors
- Sensor Data Records (SDRs) and SDR Repository
- FRU information
- Autonomous event logging
- System Event Log (SEL): holding at least 5,000 entries
- IPMI Hardware Watchdog Timer
- Platform Event Filtering (PEF)
- Serial Over LAN (SOL)— ability to redirect the system serial controller over LAN to a remote console.
- Proprietary management features including:
  - — Fault Resilient Booting
  - — EFI BIOS logging of power-on self-test (POST) progress and POST errors
  - — Snooping of port 80h POST code
  - — Advanced Configuration and Power Interface (ACPI)
  - — Serial port buffering (at least one screen worth of data)
  - — Rear Transition Module (RTM) management support
  - — AdvancedMC management support

The high-level architecture of the baseboard management for the MPCBL0050 is represented in Figure 27.

Figure 27. IPMC Block Diagram



The main processors communicate with the IPMC using the Keyboard Controller Style (KCS) interface. The EFI BIOS uses the SMS interface for normal communication and the SMM interface when executing code under Systems Management Mode (SMM). The base address of the LPC interface for SMS is 0xCA2/CA3 for the SMS and 0xCA8/CAC for SMM operation. The EFI BIOS is also able to communicate with the IPMC via the KCS interface for POST error logging purposes, fault resilient purposes and critical interrupts.

An external 256 KByte SRAM is used as a storage area for code when flash programming is under execution. The Field Replacement Unit (FRU) inventory information, SEL events, and SDR information is stored in an external Serial EEPROM. Having the SEL and logging functions managed by the IPMC helps ensure that "post-mortem" logging information is available even if the system processor becomes disabled.

IPMB isolators on both IPMB busses are used to switch and isolate a faulty IPMB bus on a board from the backplane IPMB bus connections. Where possible, the IPMC activates the redundant IPMB bus to re-establish system management communication to report the fault.

The on-board DC voltages are monitored by the LM93 device, manufactured by National Semiconductor*. The IPMC queries the LM93 over a local system management I^2C bus. External CPU thermal diodes and PROCHOT signals are connected to this device to ensure report thermal events.

The CPLD controls the enabling and monitoring of power good signals from all on-board power converters. It also controls power sequencing to ensure that all of the converters power up in the correct order to prevent any latch-up or damage to a device. The FPGA and CPLD are also used to expand the GPIO capabilities of the IPMC management circuitry due to the limited number of GPIO's supported by the IPMC. The LPC interface between the FPGA and ICH is used to monitor the Port 80 codes during power up. In the event of a board failing to power up, a user can query the last five Port 80 codes stored in the FPGA registers using an Intel OEM IPMI command.

To increase the reliability of the MPCBL0050 SBC, a watchdog timer is implemented. More details on watchdog timer operation and features are available in Section 5.14.1.

5.1 Supervision

Table 27 lists the main components that perform hardware monitoring of voltages and timers.

Table 27. Hardware monitoring components

Component	Function	Monitors
Intelligent Platform Management Controller	WDT #1	IPMI watchdog timer (monitors payload). This WDT is strobed by payload. If the timer expires (times out), it executes pre-determined action (payload hard reset, power down, power cycle or do nothing) and generates an IPMI SEL event is logged.
Intelligent Platform Management Controller	WDT #2	IPMI hardware watchdog timer (monitors IPMC). This WDT is strobed by IPMC firmware. It has a 1 second timeout with a 500ms strobe. If the WDT expires, it isolates the MPCBL0050 IPMB buses from the backplane , and resets the IPMC.
LM93	Voltage/ Temperature	On-board voltages/temperature, CPU "PROCHOT", and processor VID.
Various Devices	Temperature	Monitor on-board temperature. See Figure 28, "On-board temperature sensor locations" on page 100.

5.2 Sensor Data Record (SDR)

Sensor Data Records (SDRs) contain information about the type and number of sensors in the baseboard, sensor threshold support, event generation capabilities, and the types of sensor readings handled by system management firmware.

The MPCBL0050 management controller is set up as a satellite management controller (SMC). It supports sensor devices, whose population is static by nature. SDRs can be queried using Device SDR commands to the firmware.

Table 28 lists the sensor identification numbers and information regarding the sensor type, name, supported thresholds, assertion and deassertion information, and a brief description of the sensor purpose. See the Intelligent Platform Management Interface Specification, Version 2.0 and Intelligent Platform Management Interface Specification, Version 2.0 for sensor and event/reading-type table information. The following sections describe the information contained in Table 28 columns.

Sensor Type

The sensor type references the values enumerated in the Sensor Type Codes table in the IPMI 2.0 specification. It provides the context in which to interpret the sensor, such as the physical entity or characteristic that is represented by this sensor.

Event/Reading Type

The Event/Reading Type references values from the Event/Reading Type Code Ranges and Generic Event/Reading Type Codes tables in the IPMI 2.0 specification. Note that digital sensors are a specific type of discrete sensors, which have only two states.

Event Thresholds/Triggers

Event Thresholds are supported event generating thresholds for Threshold type sensors.

- [u,l][nr,c,nc]: upper nonrecoverable, upper critical, upper noncritical, lower nonrecoverable, lower critical, lower noncritical
- uc, lc: upper critical, lower critical

Event Triggers are supported event generating offsets for discrete type sensors. The offsets can be found in the Generic Event/Reading Type Codes or Sensor Type Codes tables in the IPMI specification, depending on whether the sensor event/reading type is generic or a sensor specific response

Event Data

This is the data that is included in an event message generated by the associated sensor.

Health LED On/Off

This indicates events that turn on or off the Health LED. The following are used to indicate how the LED is affected.

- +: Event turns on the LED. Payload power cycle is required to turn it off
- C/D: critical event assertion turns LED on. Deassertion turns it off
- A/D: Offset assertion turns health LED on. Deassertion turns LED off.

Event

This column contains description of the offsets for discrete sensors and supported thresholds for analog sensors.

Assertion/De-assertion Enables

Assertions and de-assertion indicators reveals what type of events this sensor can generate.

- As: Assertion
- De: Deassertion

- "-": No events generated

Readable Value/Offsets

Readable value indicates the type of value returned for threshold and other non-discrete type sensors. Readable offsets indicates the offsets for discrete sensors that are readable via the Get Sensor Reading command. Possible values:

- Analog: Indicates the analog value of sensor can be read.
- yes: Indicates that discrete offset can be read
- "-": Indicates that this is event only sensor/offset and cannot be read

Rearm Sensors

The "rearm" is a request for the event status for a sensor to be rechecked and updated upon a transition between good and bad states. Rearming the sensors can be done manually or automatically. This column indicates the type supported by the sensor. The following abbreviations are used in the comment column of Table 28 to describe a sensor:

- A: Auto rearm
- M: Manual rearm

Standby

Some sensors operate on standby power. These sensors may be accessed and/or generate events when the payload power is off, but DC power is present.

- "-" marks sensors that cannot be accessed on standby power
- "stby" marks sensors that can be accessed on standby power

Polling Time

The polling time is the interval time in seconds that the IPMC controller reads the sensor to determine the value of the sensor. For example, if the polling time is 5 seconds, the value of that sensor is read every 5 seconds and compared against the Sensor Device Record (SDR) thresholds. If the sensor value exceeds the threshold limits, then a SEL event is generated. Some sensors generate events asynchronously and do not need to be polled. Typically, only analog sensors require polling.

Managing Device

The managing device is the hardware device that a sensor is connected to.

Table 28. IPMC hardware sensor and events (Sheet 1 of 18)

Sensor Name	Sensor No.	Sensor Type	Event / Reading Type	Event Offset ED1 [3:0]	Event Data		Health Led On	Health Led Off	Event	Assert / De-assert Events	Readable Value / Offsets	Rearm	Standby	Polling Time, (seconds)
					Byte 2	Byte 3								
Power Unit Status (See Chapter 5.0, "Power Unit Sensor" for details)	01h	Power Unit 09h	Sensor Specific 6Fh	00h	Power Fault Reg #1 (see Table 30)	Power Fault Reg #2 (see Table 30)			Power Off	As	–	A	X	N/A
				01h					Power Cycle					
				05h			+		Soft Power Control Fault					
				06h			+		Power Unit Failure					
IPMC Watchdog	03h	Watchdog 23h	Sensor Specific 6Fh	00h	As per IPMI Spec.	FFh			Timer Expired	As & De	–	A	X	N/A
				01h					Hard Reset					
				02h					Power Down					
				03h					Power Cycle					
				08h					Timer Interrupt					
Platform Security Violation	04h	Platform Security Violation Attempt 06h	Sensor Specific 6Fh	05h	FFh	FFh			Out-of-band access password violation. Logged by BIOS if incorrect password is used.	As	–	A	X	N/A
POST Error	06h	System Firmware Progress 0Fh	Sensor Specific 6Fh	00h	Refer to Table 68, "EFI BIOS POST error messages" on page 157 for possible byte 2 and byte3 values.				Logged by EFI BIOS in case of post errors during board boot.	As	-	A	-	N/A
Critical Int	07h	Critical Interrupt 13h	Sensor Specific 6Fh	04h	Bus	Device & Function	+		PCI PERR	As	-	A	-	N/A
				05h			+		PCI SERR					
				07h	FFh	FFh	+		Bus Correctable Error					
				08h			+		Bus Uncorrectable Error					

Table 28. IPMC hardware sensor and events (Sheet 2 of 18)

Sensor Name	Sensor No.	Sensor Type	Event / Reading Type	Event Offset ED1 [3:0]	Event Data		Health Led On	Health Led Off	Event	Assert / De-assert Events	Readable Value / Offsets	Rearm	Standby	Polling Time, (seconds)
					Byte 2	Byte 3								
Memory	08h	Memory 0Ch	Sensor Specific 6Fh	00h	FFh	DIMMid: 00h - DIMM1 01h - DIMM2 02h - DIMM3 03h - DIMM4 0Fh - General Memory Error	+		Correctable ECC	As	-	A	-	N/A
				01h			+		Uncorrectable ECC					
Logging Disabled	09h	Event Logging Disabled 10h	Sensor Specific 6Fh	00h	DIMM id: 00h - DIMM1 01h - DIMM2 02h - DIMM3 03h - DIMM4	FFh			Correctable Memory Error Logging Disabled.	As	-	A	-	N/A
				02h	FFh	FFh			Log Area Reset/Cleared. SEL has been cleared.					
				04h	FFh	FFh			Sel Full					
				05h	FFh	FFh			Sel Almost Full (95%)					
Session Audit	0Ah	Session Audit 2Ah	Sensor Specific 6Fh	00h	FFh As per IPMI Spec.	FFh As per IPMI Spec.			Session Activation	As	-	A	X	N/A
				01h					Session Deactivation					

Table 28. IPMC hardware sensor and events (Sheet 3 of 18)

Sensor Name	Sens or No.	Sensor Type	Event / Reading Type	Event Offset ED1 [3:0]	Event Data		Health Led On	Health Led Off	Event	Assert / De-assert Events	Readable Value / Offsets	Rearm	Standby	Polling Time, (seconds)
					Byte 2	Byte 3								
Version Change	0Eh	Version Change 2Bh	Sensor Specific 6Fh	00h	FFh	FFh			Hardware change detected with associated Entity.	As	-	A	X	N/A
				01h	FFh	FFh			Firmware or software change detected with associated Entity.					
				002	FFh	FFh			Hardware incompatibility detected with associated Entity.					
				03h	FFh	FFh			Firmware or software incompatibility detected with associated Entity					
				04h	FFh	FFh			Entity is of an invalid or unsupported hardware version.					
				05h	FFh	FFh			Entity contains an invalid or unsupported firmware or software version.					
				06h	FFh	FFh			Hardware Change detected with associated Entity was successful.					
				07h	SW version change type	FFh			Software or F/W Change detected with associated Entity was successful					

Table 28. IPMC hardware sensor and events (Sheet 4 of 18)

Sensor Name	Sensor No.	Sensor Type	Event / Reading Type	Event Offset ED1 [3:0]	Event Data		Health Led On	Health Led Off	Event	Assert / De-assert Events	Readable Value / Offsets	Rearm	Standby	Polling Time, (seconds)
					Byte 2	Byte 3								
FW Update	0Fh	OEM C0h	OEM 72h	00h	Major FW Version	Minor FW Version			Roll-Back FW Image Captured	As	-	A	X	N/A
				01h	FFh	FFh			Roll-Back SDR Captured					
				02h	Major FW Version	Minor FW Version			Staged Image Registered					
				03h	FFh	FFh			Staged SDR Registered					
				04h	Major FW Version	Minor FW Version			Roll-Back via Switch					
				05h					Roll-Back via Command					
				06h					FW Roll-Back due to operation check failure					
				07h					Roll-Back Complete					
				08h					Staged FW Update Completed					
				09h					Direct FW Update Completed					
0.9V DDR	10h	Voltage 02h	Threshold 01h	R, T	–	–	C	D	[u,l][nr,c,nc]	As & De	Analog	A	–	5
1.2V SAS	11h	Voltage 02h	Threshold 01h	R, T	–	–	C	D	[u,l][nr,c,nc]	As & De	Analog	A	–	5
1.2V VTT	12h	Voltage 02h	Threshold 01h	R, T	–	–	C	D	[u,l][nr,c,nc]	As & De	Analog	A	–	5
1.5V DDR	13h	Voltage 02h	Threshold 01h	R, T	–	–	C	D	[u,l][nr,c,nc]	As & De	Analog	A	–	5
1.5	14h	Voltage 02h	Threshold 01h	R, T	–	–	C	D	[u,l][nr,c,nc]	As & De	Analog	A	-	5

Table 28. IPMC hardware sensor and events (Sheet 5 of 18)

Sensor Name	Sensor No.	Sensor Type	Event / Reading Type	Event Offset ED1 [3:0]	Event Data		Health Led On	Health Led Off	Event	Assert / De-assert Events	Readable Value / Offsets	Rearm	Standby	Polling Time, (seconds)
					Byte 2	Byte 3								
1.8V Early PHY	15h	Voltage 02h	Threshold 01h	R, T	–	–	C	D	[u,l][nr,c,nc]	As & De	Analog	A	X	5
1.8V DDR	16h	Voltage 02h	Threshold 01h	R, T	–	–	C	D	[u,l][nr,c,nc]	As & De	Analog	A	–	5
12V RTM	17h	Voltage 02h	Threshold 01h	R, T	–	–	C	D	[u,l][nr,c,nc]	As & De	Analog	A	–	5
3.3V Early	18h	Voltage 02h	Threshold 01h	R, T	–	–	C	D	[u,l][nr,c,nc]	As & De	Analog	A	X	5
5V	19h	Voltage 02h	Threshold 01h	R, T	–	–	C	D	[u,l][nr,c,nc]	As & De	Analog	A	–	5
5V Early	1Ah	Voltage 02h	Threshold 01h	R, T	–	–	C	D	[u,l][nr,c,nc]	As & De	Analog	A	X	5
12V Early	1Bh	Voltage 02h	Threshold 01h	R, T	–	–	C	D	[u,l][nr,c,nc]	As & De	Analog	A	–	5
1.5V Early ESB	1Ch	Voltage 02h	Threshold 01h	R, T	–	–	C	D	[u,l][nr,c,nc]	As & De	Analog	A	–	5
RTM 12V CURRENT	20h	Current 03h	Threshold 01h	R, T	–	–	C	D	[u,l][nr,c,nc]	As & De	Analog	A	–	5
AMC 12V CURRENT	21h	Current 03h	Threshold 01h	R, T	–	–	C	D	[u,l][nr,c,nc]	As & De	Analog	A	–	5
3.3V	22h	Voltage 02h	Threshold 01h	R, T	–	–	C	D	[u,l][nr,c,nc]	As & De	Analog	A	–	5
12V AMC	23h	Voltage 02h	Threshold 01h	R, T	–	–	C	D	[u,l][nr,c,nc]	As & De	Analog	A	–	5
5V USB	24h	Voltage 02h	Threshold 01h	R, T	–	–	C	D	[u,l][nr,c,nc]	As & De	Analog	A	–	5
1.2V Early PHY	25h	Voltage 02h	Threshold 01h	R, T	–	–	C	D	[u,l][nr,c,nc]	As & De	Analog	A	–	5
1.1V FIC (Fabric)	27h	Voltage 02h	Threshold 01h	R, T	–	–	C	D	[u,l][nr,c,nc]	As & De	Analog	A	–	5

Table 28. IPMC hardware sensor and events (Sheet 6 of 18)

Sensor Name	Sens or No.	Sensor Type	Event / Reading Type	Event Offset ED1 [3:0]	Event Data		Health Led On	Health Led Off	Event	Assert / De-assert Events	Readable Value / Offsets	Rearm	Standby	Polling Time, (seconds)
					Byte 2	Byte 3								
1.8 FIC (Fabric)	28h	Voltage 02h	Threshold 01h	R, T	–	–	C	D	[u,l][nr,c,nc]	As & De	Analog	A	–	5
CPU1 Current	29h	Current 03h	Threshold 01h	R, T	–	–	C	D	[u,l][nr,c,nc]	As & De	Analog	A	–	5
CPU2 Current	2Ah	Current 03h	Threshold 01h	R, T	–	–	C	D	[u,l][nr,c,nc]	As & De	Analog	A	–	5
MCH JUNCT TEMP	30h	Temp 01h	Threshold 01h	R, T	–	–	C	D	[u,l][nr,c,nc]	As & De	Analog	A	X	0.5
SAS PCBT TEMP	31h	Temp 01h	Threshold 01h	R, T	–	–	C	D	[u,l][nr,c,nc]	As & De	Analog	A	X	0.5
DC/DC PCBT TEMP	32h	Temp 01h	Threshold 01h	R, T	–	–	C	D	[u,l][nr,c,nc]	As & De	Analog	A	X	0.5
OUTLET PCBT TEMPT	33h	Temp 01h	Threshold 01h	R, T	–	–	C	D	[u,l][nr,c,nc]	As & De	Analog	A	X	1
CENTER PCBT TEMP	36h	Temp 01h	Threshold 01h	R, T	–	–	C	D	[u,l][nr,c,nc]	As & De	Analog	A	X	0.5
ICH PCBT TEMP	37h	Temp 01h	Threshold 01h	R, T	–	–	C	D	[u,l][nr,c,nc]	As & De	Analog	A	X	1
FRONT PCBB TEMP	38h	Temp 01h	Threshold 01h	R, T	–	–	C	D	[u,l][nr,c,nc]	As & De	Analog	A	X	1

Table 28. IPMC hardware sensor and events (Sheet 7 of 18)

Sensor Name	Sens or No.	Sensor Type	Event / Reading Type	Event Offset ED1 [3:0]	Event Data		Health Led On	Health Led Off	Event	Assert / De-assert Events	Readable Value / Offsets	Rearm	Standby	Polling Time, (seconds)
					Byte 2	Byte 3								
DIMM1	50h	Slot connector 21h	Sensor Specific 6Fh	00h	FFh	FFh	+		Fault Status Asserted Asserted on one of the two: -EFI BIOS detects an uncorrectable ECC error - EFI BIOS issues a Set DIMM State command indicating that a DIMM fault status should be asserted.	As	-	M	-	N/A
				02h					Device Installed If BIOS detects DIMM presence it sets the this state at system power up (EFI BIOS POST) time					
				08h					Disabled					
DIMM2	51h	Slot connector 21h	Sensor Specific 6Fh	00h	FFh	FFh	+		Fault Status Asserted	As	-	M	-	N/A
				01h					Device Installed					
				08h					Disabled					
DIMM3	52h	Slot connector 21h	Sensor Specific 6Fh	00h	FFh	FFh	+		Fault Status Asserted	As	-	M	-	N/A
				01h					Device Installed					
				08h					Disabled					
DIMM4	53h	Slot connector 21h	Sensor Specific 6Fh	00h	FFh	FFh	+		Fault Status Asserted	As	-	M	-	N/A
				01h					Device Installed					
				08h					Disabled					
EFI BIOS FWH0 Flash	54h	OEM C0h	Digital Discrete 03h	00h	-	-			State Deasserted	-	Yes	A	X	0.1
				01h	Change reason. Refer to Table 5.2.3	FFh			State Asserted (FWH0 is active)	As				

Table 28. IPMC hardware sensor and events (Sheet 8 of 18)

Sensor Name	Sens or No.	Sensor Type	Event / Reading Type	Event Offset ED1 [3:0]	Event Data		Health Led On	Health Led Off	Event	Assert / De-assert Events	Readable Value / Offsets	Rearm	Standby	Polling Time, (seconds)
					Byte 2	Byte 3								
EFI BIOS FWH1 Flash	55h	OEM C0h	Digital Discrete 03h	00h	-	-			State Deasserted	-	Yes	A	X	0.1
				01h	Change reason. Refer Table 5.2.3	FFh			State Asserted (FWH1 is active)	As				
OS Stop/ Shutdown. Note: All offsets of this sensor logged by OS and are defined in IPMI spec.	56h	OS Stop / Shutdown 20h	Sensor Specific 6Fh	00h	As per IPMI specification				Critical Stop during OS load / initialization.	As	-	N/A	X	N/A
				01h					Run-time Critical Stop (a.k.a. 'core dump', 'blue screen'). Note: after this event OS writes to SEL error message text as series of SEL OEM 0xF0 entries.					
				02h					OS Graceful Stop.					
				03h					OS Graceful Shutdown (system graceful power down by OS).					
				04h					Soft Shutdown initiated by PEF.					
				05h					Agent Not Responding. Graceful shutdown request to agent via IPMC did not occur due to missing or malfunctioning local agent.					

Table 28. IPMC hardware sensor and events (Sheet 9 of 18)

Sensor Name	Sensor No.	Sensor Type	Event / Reading Type	Event Offset ED1 [3:0]	Event Data		Health Led On	Health Led Off	Event	Assert / De-assert Events	Readable Value / Offsets	Rearm	Standby	Polling Time, (seconds)
					Byte 2	Byte 3								
ACPI State	82h	System ACPI Power State 22h	Sensor Specific 6Fh	00h					S0 / G0	As	Yes		X	N/A
				01h					S1					
				04h					S4					
				05h					S5 / G2					
				0Bh					Legacy On					
				0Ch					Legacy Off					
System Event	83h	System Event 12h	Sensor Specific 6Fh	01h	FFh	FFh			OEM System Boot Event	As	–	A	–	N/A
				04h	As per IPMI Spec.	FFh			PEF Action					
				05h	As per IPMI Spec.	As per IPMI Spec.			Timestamp Clock Synch - logged by BIOS on every system startup					
Button	84h	Button 14h	Sensor Specific 6Fh	00h	FFh	FFh			Power Button - logged when Front Panel reset button has been pressed for more than 5 seconds	As	–	A	X	N/A
				02h					Reset Button - logged when Front Panel reset button has been pressed for less than 5 seconds when payload is powered up					

Table 28. IPMC hardware sensor and events (Sheet 10 of 18)

Sensor Name	Sensor No.	Sensor Type	Event / Reading Type	Event Offset ED1 [3:0]	Event Data		Health Led On	Health Led Off	Event	Assert / De-assert Events	Readable Value / Offsets	Rearm	Standby	Polling Time, (seconds)
					Byte 2	Byte 3								
SMI Timeout	85h	OEM C0h	Digital Discrete 03h	00h	FFh	FFh			State Deasserted	As &De	–	A	-	N/A
				01h			+		State Asserted SMI has been asserted for more than 90 seconds. It means that BIOS/OS did not handle SMI interrupt.					
Forced Reset	86h	OEM C0h	Digital Discrete 03h	01h	Reset type: 00h - Hard reset 01h - Soft Reset (Keyboard)	Reset Cause See: Table			State Asserted - Reset signal state has been asserted and deasserted by IPMC Note: this sensor does not reflect state of reset signal. Events are logged only when IPMC asserts this signal. This is event-only sensor.	As	-	A	-	N/A
Forced NMI	87h	OEM C0h	Digital Discrete 03h	01h	NMI Cause	FFh			State Asserted - NMI signal state has been asserted and deasserted by IPMC Note: This sensor does not reflect state of CPU NMI input. Events are logged only when IPMC asserts CPU NMI signal. This is event-only sensor.	As	-	A	-	N/A
SMI State	88h	OEM C0h	Digital Discrete 03h	00h	FFh	FFh			State Deasserted	As & De	–	A	–	N/A
				01h					State Asserted					

Table 28. IPMC hardware sensor and events (Sheet 11 of 18)

Sensor Name	Sensor No.	Sensor Type	Event / Reading Type	Event Offset ED1 [3:0]	Event Data		Health Led On	Health Led Off	Event	Assert / De-assert Events	Readable Value / Offsets	Rearm	Standby	Polling Time, (seconds)
					Byte 2	Byte 3								
IPMC Watchdog Overflow	89h	OEM C0h	Digital Discrete 03h	01h	FFh	FFh			Last IPMC reset caused by internal watchdog circuit, integrated in IPMC microcontroller	As	–	–	–	N/A
				02h					Last IPMC reset caused by external watchdog and voltage monitoring device, connected to IPMC microcontroller via external address & data bus.					
SBC FRU Hot Swap	8Ah	PICMG Hot Swap Event F0h	Sensor Specific 6Fh	00h	FFh	FFh			M0 – FRU not installed	As		A		N/A
				01h					M1 – FRU inactive					
				02h					M2 – FRU activation request					
				03h					M3 - FRU activation in progress					
				04h					M4 - FRU active					
				05h					M5 – FRU deactivation request					
				06h					M6 - FRU deactivation in Progress					
				07h					M7 - Communication lost					

Table 28. IPMC hardware sensor and events (Sheet 12 of 18)

Sensor Name	Sensor No.	Sensor Type	Event / Reading Type	Event Offset ED1 [3:0]	Event Data		Health Led On	Health Led Off	Event	Assert / De-assert Events	Readable Value / Offsets	Rearm	Standby	Polling Time, (seconds)
					Byte 2	Byte 3								
RTM FRU Hot Swap	8Bh	PICMG Hot Swap Event F0h	Sensor Specific 6Fh	00h	FFh	FFh			M0 – FRU not installed	As	–	A	X	N/A
				01h					M1 – FRU inactive					
				02h					M2 – FRU activation request					
				03h					M3 - FRU activation in progress					
				04h					M4 - FRU active					
				05h					M5 – FRU deactivation request					
				06h					M6 - FRU deactivation in Progress					
				07h					M7 - Communication lost					
AdvancedMC FRU Hot Swap	8Ch	PICMG Hot Swap Event F0h	Sensor Specific 6Fh	00h	FFh	FFh			M0 – FRU not installed	As	–	A	X	N/A
				01h					M1 – FRU inactive					
				02h					M2 – FRU activation request					
				03h					M3 - FRU activation in progress					
				04h					M4 - FRU active					
				05h					M5 – FRU deactivation request					
				06h					M6 - FRU deactivation in Progress					
				07h					M7 - Communication lost					

Table 28. IPMC hardware sensor and events (Sheet 13 of 18)

Sensor Name	Sensor No.	Sensor Type	Event / Reading Type	Event Offset ED1 [3:0]	Event Data		Health Led On	Health Led Off	Event	Assert / De-assert Events	Readable Value / Offsets	Rearm	Standby	Polling Time, (seconds)
					Byte 2	Byte 3								
IPMB Link State	8Dh	PICMG Physical IPMB-0 Link F1h	Sensor Specific 6Fh	00h	FFh	FFh			IPMB A & B disabled	As	–	A	X	N/A
				01h					IPBM A enabled IPMB B disabled					
				02h					IPMB A disabled IPMB B enabled					
				03h					IPMB A & B enabled					
CPU 1 Status	90h	Processor 07h	Sensor Specific 6Fh	00h	FFh	FFh	+		IERR	As & De	Yes	M	X	N/A
				01h			+		Thermal Trip	As	Yes			
				02h					FRB1	As	No			
				03h					FRB2	As	No			
				04h					FRB3	As	No			
				05h					Config Error	As	No			
				07h					Presence	As & De	Yes			
				08h					Disabled	As	No			
				0Ah					Thermal throttle (prochot)	As	Yes			

Table 28. IPMC hardware sensor and events (Sheet 14 of 18)

Sensor Name	Sensor No.	Sensor Type	Event / Reading Type	Event Offset ED1 [3:0]	Event Data		Health Led On	Health Led Off	Event	Assert / De-assert Events	Readable Value / Offsets	Rearm	Standby	Polling Time, (seconds)
					Byte 2	Byte 3								
CPU 2 Status	91h	Processor 07h	Sensor Specific 6Fh	00h	FFh	FFh	+		IERR	As & De	Yes	M	X	N/A
				01h			+		Thermal Trip	As	Yes			
				02h					FRB1	As	No			
				03h					FRB2	As	No			
				04h					FRB3	As	No			
				05h					Config Error	As	No			
				07h					Presence	As & De	Yes			
				08h					Disabled	As	No			
				0Ah					Thermal throttle		Yes			
CPU 1 Junct Temp	92h	Temp 01h	Threshold 01h	R, T	–	–	C	D	[u,l][nr,c,nc] Note: This sensor reports negative values measured by PECI	As & De	Analog	A		0.5
CPU 2 Junct Temp	93h	Temp 01h	Threshold 01h	R, T	–	–	C	D	[u,l][nr,c,nc] Note: This sensor reports negative values measured by PECI	As & De	Analog	A		0.5
CPU 1 Vcc	94h	Voltage 02h	Threshold 01h	R, T	–	–	C	D	[u,l][nr,c,nc]	As & De	Analog	A	–	5
CPU 2 Vcc	95h	Voltage 02h	Threshold 01h	R, T	–	–	C	D	[u,l][nr,c,nc]	As & De	Analog	A	–	5

Table 28. IPMC hardware sensor and events (Sheet 15 of 18)

Sensor Name	Sensor No.	Sensor Type	Event / Reading Type	Event Offset ED1 [3:0]	Event Data		Health Led On	Health Led Off	Event	Assert / De-assert Events	Readable Value / Offsets	Rearm	Standby	Polling Time, (seconds)
					Byte 2	Byte 3								
CPU 1 ThermCtrl	96h	Temp 01h	Discrete 07h	00h	–	–			Transitioned to OK	–	–	M	–	N/A
				01h	FFh	FFh			Transitioned to Non-Critical from OK Percentage of time a processor has been throttling over the last 1.46 seconds was larger than 0% and previous state was "Transitioned to OK".	As & De				
CPU 2 ThermCtrl	97h	Temp 01h	Discrete 07h	00h	–	–			Transitioned to OK	–	–	M	–	N/A
				01h	FFh	FFh			Transitioned to Non-Critical from OK	As & De				
CPU 1 VRD Hot	98h	Temp 01h	Discrete 07h	00h	-	-			Transitioned to OK	-	-	A	–	N/A
				01h	FFh	FFh			Transitioned to Non-Critical from OK	As & De				
CPU 2 VRD Hot	99h	Temp 01h	Discrete 07h	00h	-	-			Transitioned to OK	-	-	A	–	N/A
				01h	FFh	FFh			Transitioned to Non-Critical from OK	As & De				
CPU Config Error	9Ah	Processor 07h	Generic 03h	01h	FFh	FFh			State Asserted: Logged by BIOS after detecting improper CPU configuration.	As & De	Discrete	A		N/A
FI1 Junc TEMP	9Ch	Temp 01h	THreshold 01h	R,T			C	D	[u,l][nr,c,nc]	As &De	Analog	A	X	0.5
FI2 Junc TEMP	9Dh	Temp 01h	THreshold 01h	R,T			C	D	[u,l][nr,c,nc]	As &De	Analog	A	X	0.5
CPU1 PCBT TEMP	9Eh	Temp 01h	THreshold 01h	R,T			C	D	[u,l][nr,c,nc]	As &De	Analog	A	X	0.5

Table 28. IPMC hardware sensor and events (Sheet 16 of 18)

Sensor Name	Sensor No.	Sensor Type	Event / Reading Type	Event Offset ED1 [3:0]	Event Data		Health Led On	Health Led Off	Event	Assert / De-assert Events	Readable Value / Offsets	Rearm	Standby	Polling Time, (seconds)
					Byte 2	Byte 3								
CPU2 PCBT TEMP	9Fh	Temp 01h	THreshold 01h	R,T			C	D	[u,l][nr,c,nc]	As &De	Analog	A	X	0.5
CPU2 PCBB TEMP Note: This is not a CPU sensor. It is located at the bottom of the PCB near CPU2	A0h	Temp 01h	THreshold 01h	R,T			C	D	[u,l][nr,c,nc]	As &De	Analog	A	X	0.5
INLET PCBB TEMP	A1h	Temp 01h	THreshold 01h	R,T			C	D	[u,l][nr,c,nc]	As &De	Analog	A	X	0.5
AMBIENT AIR TEMP	A2h	Temp 01h	THreshold 01h	R,T	-	-	C	D	[u,l][nr,c,nc]	As &De	Analog	A	X	0.5
DIMM 1 Size	B0h	OEM C0h	OEM C0h	00h	DIMM size low byte	DIMM size high byte			DIMM size, size in MB = (Event Data Byte 3) * 256 + (Event Data Byte 2) Offset logged by BIOS on startup	As	-	-	-	N/A
DIMM 2 Size	B1h	OEM C0h	OEM C0h	00h	DIMM size low byte	DIMM size high byte			DIMM size, size in MB	As	-	-	-	N/A
DIMM 3 Size	B2h	OEM C0h	OEM C0h	00h	DIMM size low byte	DIMM size high byte			DIMM size, size in MB	As	-	-	-	N/A
DIMM 4 Size	B3h	OEM C0h	OEM C0h	00h	DIMM size low byte	DIMM size high byte			DIMM size, size in MB	As	-	-	-	N/A
CPU 1 Type	B8h	Microcontroller / Coprocessor 16h	Sensor Specific 6Fh	00h	CPU Family ID	Bits: [7:4]=Model [3:0]=Stepping			CPU 1 Type Offset logged by BIOS on startup.	As	-	-	-	N/A

Table 28. IPMC hardware sensor and events (Sheet 17 of 18)

Sensor Name	Sensor No.	Sensor Type	Event / Reading Type	Event Offset ED1 [3:0]	Event Data		Health Led On	Health Led Off	Event	Assert / De-assert Events	Readable Value / Offsets	Rearm	Standby	Polling Time, (seconds)
					Byte 2	Byte 3								
CPU 2 Type	B9h	Microcontroller / Coprocessor 16h	Sensor Specific 6Fh	00h	CPU Family ID	Bits: [7:4]=Model [3:0]=Stepping			CPU 1 Type Offset logged by BIOS on startup.	As	-	-	-	
USB Overcurrent	E1h	OEM C0h	Digital Discrete 03h	00h	-	-			No USB overcurrent	-	-	A	X	1
				01h	FFh	FFh	A	D	USB overcurrent	As & De				
LAN Link A Sts	E3h	LAN 27h	Sensor Specific 6Fh	00h	-	-			Heartbeat Lost (Link Down)	-	-	A		1
				01h	FFh	FFh			Heartbeat (Link Up) Asserted when phy is linked up, Deasserted when phy goes down.	As & De				
LAN Link B Sts	E4h	LAN 27h	Sensor Specific 6Fh	00h	-	-			Heartbeat Lost (Link Down)	-	-	A		1
				01h	FFh	FFh			Heartbeat (Link Up)	As & De				
LAN Link C Sts	E5h	LAN 27h	Sensor Specific 6Fh	00h	-	-			Heartbeat Lost (Link Down)	-	-	A		1
				01h	FFh	FFh			Heartbeat (Link Up)	As & De				
LAN Link D Sts	E6h	LAN 27h	Sensor Specific 6Fh	00h	-	-			Heartbeat Lost (Link Down)	As & De	-	A		1
				01h	FFh	FFh			Heartbeat (Link Up)					
LAN Link E Sts	E7h	LAN 27h	Sensor Specific 6Fh	00h	-	-			Heartbeat Lost (Link Down)	-	-	A		1
				01h	FFh	FFh			Heartbeat (Link Up)	As & De				

Table 28. IPMC hardware sensor and events (Sheet 18 of 18)

Sensor Name	Sensor No.	Sensor Type	Event / Reading Type	Event Offset ED1 [3:0]	Event Data		Health Led On	Health Led Off	Event	Assert / De-assert Events	Readable Value / Offsets	Rearm	Standby	Polling Time, (seconds)
					Byte 2	Byte 3								
LAN Link F Sts	E8h	LAN 27h	Sensor Specific 6Fh	00h	-	-			Heartbeat Lost (Link Down)	-	-	A		1
				01h	FFh	FFh			Heartbeat (Link Up)	As & De				
Feed A Fail	E9h	OEM C1h	Digital Discrete 03h	00h	-	-			State Deasserted	-	Yes	A	X	1
				01h	FFh	FFh	A	D	State Asserted	As & De				
Feed BFail	EAh	OEM C1h	Digital Discrete 03h	00h	-	-			State Deasserted	-	Yes	A	X	1
				01h	FFh	FFh	A	D	State Asserted	As & De				
Fuse PlusA Fail	EBh	OEM C0h	Digital Discrete 04h	00h	-	-			State Deasserted	-	Yes	A	X	1
				01h	FFh	FFh	A	D	State Asserted	As & De				
Fuse MinusA Fail	ECh	OEM C0h	Digital Discrete 04h	00h	-	-			State Deasserted	-	Yes	A	X	1
				01h	FFh	FFh	A	D	State Asserted	As & De				
Fuse PlusB Fail	EDh	OEM C0h	Digital Discrete 04h	00h	-	-			State Deasserted	-	Yes	A	X	1
				01h	FFh	FFh	A	D	State Asserted	As & De				
Fuse MinusB Fail	EEh	OEM C0h	Digital Discrete 04h	00h	-	-			State Deasserted	-	Yes	A	X	1
				01h	FFh	FFh	A	D	State Asserted	As & De				

5.2.1 Power Unit Sensor

The power unit sensor indicates the status of the onboard power subsystem. If the power subsystem fails to turn on the payload power, as indicated by the power unit power good signal, a soft power control failure will be indicated. If the power unit power good signal is deasserted after power has been successfully turned on, a power unit failure will be indicated. This sensor is an IPMI sensor type Power Unit (09h). Table 29 describes sensor states/offsets and its descriptions.

Table 29. Power unit status sensor states

State	Assertion / Deassertion	Sensor-specific Offset	Description
Power Off/Down	Assertion and Deassertion	00h	Asserted when system power is off, Deasserted when power is on
Power Cycle	Assertion and Deassertion	01h	Asserted and then deasserted as a result of a Power Cycle request. Power Cycle condition is the situation when power is turned off, then immediately turned on.
Soft Power Control Failure	Assertion	05h	Asserted if power unit did not respond to request to turn on
Power Unit Failure detected	Assertion	06h	Asserted if power unit changes power state unexpectedly

When a power fault has been detected, the contents of the CPLD "Power Fault1" and "Power Fault2" registers are read and are stored as event data 2 (power fault 1 register) and event data 3 (power fault 1 register) bytes in the SEL. The contents of the event data are described in Table 30.

Table 30. Power unit status sensor event data

Event Data	Bit	Description
02h	0	1_2V_E_PHY Power Fault
	1	1_8V_E_PHY Power Fault
	2	1_5V_E_ESB Power Fault
	3	12V_E Power Fault
	4	5V Power Fault
	5	3_3V Power Fault
	6	1_2V_SAS Power Fault
	7	1_5V_ESB Power Fault
03h	0	1_5V_DDR Power Fault
	1	1_8V_DDR Power Fault
	2	0.9VTT Power Fault
	3	1_2VTT Power Fault
	4	CPU0_VRM Power fault
	5	CPU1_VRM Power fault
	6	0
	7	0

5.2.2 BIOS FWH State Change Sensors

The IPMC implements two BIOS firmware hub sensors (event/reading type 03h=Digital Discreet, sensor type 0xC0=EOM, sensor numbers 54h and 55h).

First sensor (0x54: "BIOS FWH0 Flash") indicates state of first firmware hub selection. Second sensor (0x55: "BIOS FWH1 Flash") indicates state of second firmware hub selection.

Table 31. FWH sensor states possible

Sensor 0x54 offset	Sensor 0x55 offset	Firmware hubs state
01h	00h	First firmware hub selected. Second firmware unselected. BIOS will boot from **first** firmware hub
00h	01h	Second firmware hub selected. First firmware unselected. BIOS will boot from **second** firmware hub

Table 32. FWH sensors byte2

Event Data 2 value	Cause of change	Notes
00h	Board Insertion	After board insertion IPMC detects firmware hub selection change.
01h	DIP-SWITCH	BIOS FWH change caused by DIP-SWITCH.
02h	“Reset BIOS Flash Type” command	BIOS FWH change caused by IPMI command “Reset BIOS Flash Type”.
03h	“Set Control State” command	BIOS FWH change caused by IPMI command “Set Control State”.
FFh	FRB	BIOS FWH change caused by FRB action

5.2.3 Reset Sensor

Every reset generated by IPMC is also logged to SEL. After reset signal assertion the IPMC logs SEL event from event-only sensor "Forced Reset". Only one offset is logged: offset 01 = "State Asserted".

Event data 2 contains Reset Type:

"Value 00h means Hard Reset

"Value 01h means Soft Reset (Keyboard Reset)

Event Data 3 contains reset cause, as defined in following Table .

Table 33. Reset causes

Value	Reset reason/cause
00h	Reserved
01h	Button Reset has been forced by the IPMC in response to front panel reset button.
02h	CPU Reset has been forced by the IPMC in response to "Set Processor Status" command.
03h	FRB Reset has been forced by the IPMC in FRB3 action failure.
04h	IPMI Watchdog Reset has been forced by the IPMC on expiration of IPMI watchdog When IPMI watchdog timer use flag is configured to hard reset.
05h	IPMI command Reset has been forced by the IPMC in response to one of following IPMI commands: - "Chassis Control" with action "Hard Reset" - PICMG "FRU control" with action "Hard Reset" or "Warm Reset" - "Reset BIOS Flash Type" – after BIOS firmware hub switching
06h	**PEF** filter action
07h	Reserved
08h	Reserved
09h	Reserved
0Ah	IERR Reset has been forced by the IPMC in response to IERR detection. This reset occurs when IERR action is configured to "Reset".

5.2.4 NMI Sensor/NMI Assertion from IPMC

The IPMC generates an NMI pulse under certain conditions. The IPMC-generated NMI pulse duration is at least 30 ms. Once an NMI has been generated by the IPMC, the IPMC will not generate another until the system has been reset or powered down except that enabling NMI via an NMI Enable/Disable command will re-arm the NMI.

The IPMC captures the NMI source(s) and makes that information available via a Get NMI Source command. Reading the NMI source information causes it to be cleared. The Set NMI Source command is available to other agents (e.g., BIOS SMI Handler) to register NMI sources when they detect NMI generating errors. OS NMI handlers that save system crash state can use the Get NMI Source command to determine and save the cause of the NMI.

IPMC NMI generation can be disabled by the NMI Enable/Disable command. The default state is enabled and the enabled/disabled state is volatile (not saved across AC power cycles).

The following may cause the IPMC to generate an NMI/INIT pulse:

- Receiving a Chassis Control command to pulse the Diagnostic Interrupt.
- A PEF table entry matching an event where the filter entry has the Diagnostic Interrupt action indicated.
- Watchdog timer pre-timeout expiration with NMI/Diagnostic Interrupt pre-timeout action enabled.

- Receiving a Set NMI Source command issued from one of the command interfaces.
- IERR action - when IERR action is configured to assert NMI the IPMC will generate the NMI signal on detection of IERR error from CPU.

After NMI assertion the IPMC logs SEL event from event-only sensor "Forced NMI". Only one offset is logged: offset 01 = "State Asserted". Event data 2 contains NMI cause, as defined in Table 34.

Table 34. NMI causes

Value	NMI cause
01h	IPMI watchdog pre-timeout action
02h	PEF action
03h	Chassis command "Pulse Diagnostic Interrupt"
04h	IERR action. Used and logged when NMI has been generated in response to IERR. Note: this requires configure IERR action as "NMI"

5.3 System Event Log (SEL)

The System Event Log (SEL) is the collection of events that are generated by the IPMC. Event logs are stored in non-volatile memory (Serial EEPROM). This resides on the board and allows better tracking of error conditions on the baseboard when it is moved from chassis to chassis. Having the SEL and logging functions managed by the IPMC helps ensure that post-mortem logging information is available should a failure occur that disables the system's processor(s).

The list of all IPMC generated SEL events can be found in Table 28. Refer to the columns with the titles "Event Offset" and "Event".

Management software (not provided by Intel) running on the host processor is responsible for ensuring that SEL storage has sufficient space for SEL logging. Events are normally forwarded to the shelf manager and logged to SEL on the board. If SEL storage on the board is full, new events are forwarded to the Shelf Manager, but are not logged in SEL on the board. The user needs to ensure on-board SEL log events are backed up (if desired) and SEL log cleared if the log is full.

SEL can be cleared by *Issue Reserve SEL* and *Issue Clear SEL* IPMI commands.

Events may be received while the SEL is being cleared. The IPMC implements an event message queue to avoid messages being lost. Messages are not overwritten once they are stored in the queue.

When a board is installed into a new system, the shelf manager will only log board SEL events that occur after the board is inserted into a chassis. There may be older SEL events stored in the board that may need to be retrieved for debug or troubleshooting purposes. To get all the SEL events stored on the board, issue a *Get SEL Info* IPMI command to find out the number of IPMC SEL entries. Then loop on the *Get SEL Entry* IPMI command for number of SEL entries obtained by the *Get SEL Info* command.

A set of IPMI commands (see Appendix A, "Supported IPMI Commands") allows the SEL to be read and cleared and allows events to be added to the SEL. The IPMI commands used for adding events to the SEL are *Platform Event Message*, *Add SEL* entry, and *Partial Add Entry*.

5.3.1 Analog Sensors

Temperature, voltage and current readings are critical parameters that ensure the MPCBL0050 is operating at its predefined threshold limits. The sensor thresholds, as defined by PICMG 3.0 are categorized as follows:

- Lower Non-Recoverable
- Lower Non-Critical
- Lower Critical
- Upper Non-Critical
- Upper Critical
- Upper Non-Recoverable

If a lower non-critical or upper non-critical threshold is exceeded, it raises a minor alarm. If a lower critical or upper critical threshold is exceeded, it raises a major alarm.

The health LED is turned solid red only when critical or non-recoverable thresholds (major alarm) are exceeded. However, for any of the categories above, the IPMC forwards the events to the shelf manager to log in the shelf manager’s SEL.

Temperature sensors naming:

- PCBT suffix - thermal diode on top surface of the PCB
- PCBB suffix - thermal diode on bottom surface of the PCB
- JUNCT - sensor embedded on the chip

Table 35. Analog sensors and thresholds for SDR v0.132 (Sheet 1 of 3)

Sensor Name	Description	Sensor Number	Nominal Value	Thresholds					
				Lower			Upper		
				Non- Recoverable	Critical	Non-Critical	Non-Critical	Critical	Non-Recoverable
0.9V DDR	DDR 0.9v power	10	0.90	0.80	0.80			0.80	1.00
1.2V SAS	Core Voltage for SAS controller	11	1.20	1.05	1.05			1.05	1.35
1.2V VTT	FSB 1.2 Terminating Voltage	12	1.20	1.05	1.05			1.05	1.35
1.5V DDR	DDR 1.5V power	13	1.50	1.30	1.30			1.30	1.70
1.5V	CPU, MCH and ICH 1.5V power supply	14	1.50	1.30	1.30			1.30	1.70
1.9V EARLY PHY	Early 1.9V for PHY	15	1.90	1.70	1.72			2.08	2.10
1.8V DDR	DDR 1.8V power supply	16	1.80	1.60	1.60			1.60	2.00

Table 35. Analog sensors and thresholds for SDR v0.132 (Sheet 2 of 3)

Sensor Name	Description	Sensor Number	Nominal Value	Thresholds					
				Lower			Upper		
				Non- Recoverable	Critical	Non-Critical	Non-Critical	Critical	Non-Recoverable
12V RTM	RTM 12V	17	12.00	11.00	11.00			11.00	13.00
3.3V EARLY	Early Power 3.3V	18	3.40	2.95	3.00			3.63	3.65
5V	Main 5V (USB &ICH)	19	5.00	4.60	4.60			4.60	5.50
5V EARLY	Early 5V	1A	5.00	4.60	4.60			4.60	5.50
12V EARLY	Main Early 12V	1B	12.00	11.00	11.00			11.00	13.00
1.5V EARLY ESB	ESB2 Early 1.5V	1C	1.50	0.00	1.43			1.43	1.58
RTM 12V CURR	Current for RTM 12V supply [A]	20					3.40	3.70	4.20
AMC 12V CURR	Current for RTM 12V supply [A]	21					2.50	2.80	3.00
3.3V	Main 3.3V	22	3.30	0.00	3.14			3.14	3.65
12V AMC	AdvancedMC 12V	23	12.00	11.00	11.00			11.00	13.00
5V USB	USB 5V	24	5.00	4.50	4.50			4.50	5.50
1.2V EARLY PHY	Early Phy supply	25	1.20	1.05	1.05			1.05	1.35
1.1V FIC	Fabric Interface 1.1V	27	1.10	0.95	0.95			0.95	1.25
1.8V FIC	Fabric Interface 1.8V	28	1.80	1.60	1.60			1.60	2.00
CPU1 CURRENT	CPU 1supply current	29	22.00					44.00	60.00
CPU2 CURRENT	CPU 2 supply current	2A	22.00					44.00	60.00
MCH JUNC TEMP	MCH Temperature	30	60C		0.00	20.00	70.00	90.00	112.00
SAS PCBT TEMP	Temp Sensor SAS Controller	31	45C		0.00	7.00	65.00	80.00	90.00
DC/DC PCBT TEMP	CPU1 voltage regulator temperature	32	60C		0.00	7.00	70.00	90.00	105.00
OUTLET PCBT TEMP	Temperature sensor	33	45C		0.00	7.00	55.00	70.00	90.00
CENTER PCBT TEMP	Temperature sensor	36	55C		0.00	7.00	65.00	75.00	93.00

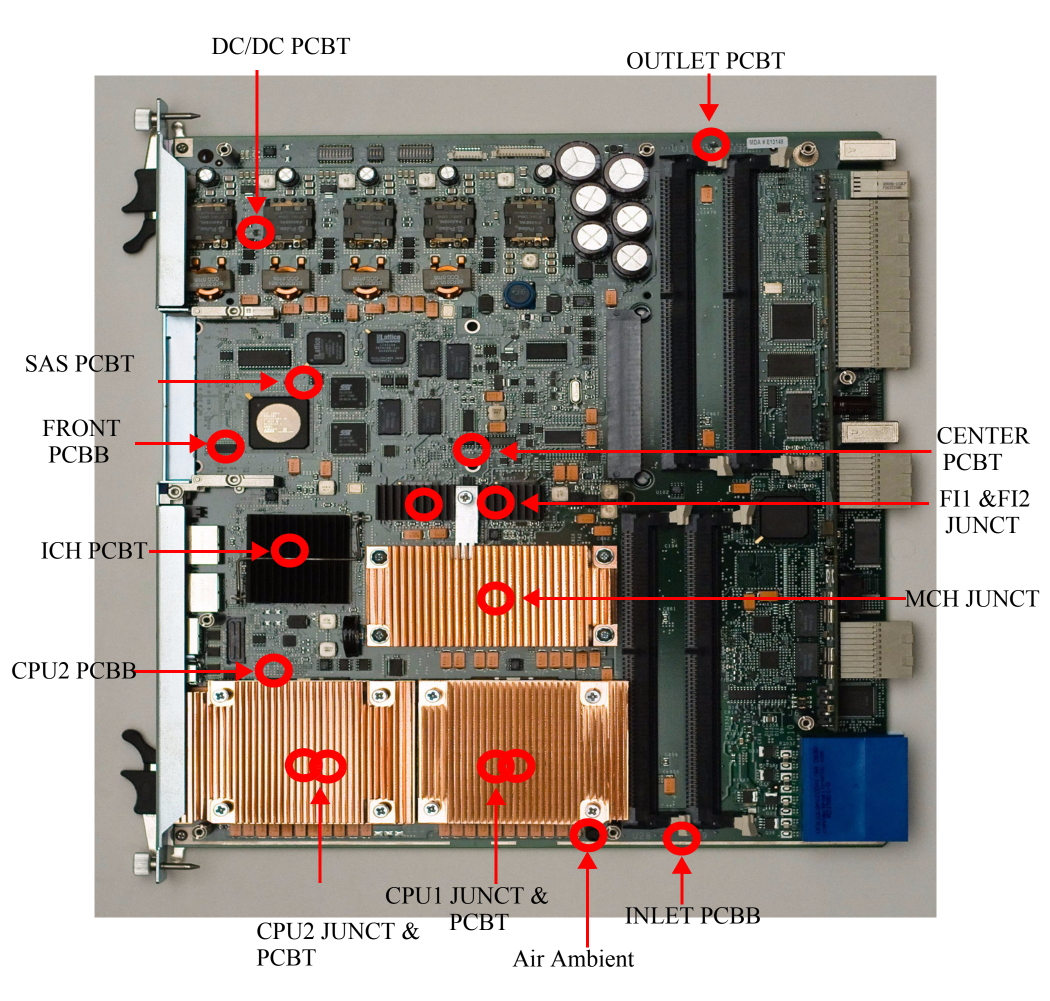
Table 35. Analog sensors and thresholds for SDR v0.132 (Sheet 3 of 3)

Sensor Name	Description	Sensor Number	Nominal Value	Thresholds					
				Lower			Upper		
				Non- Recoverable	Critical	Non-Critical	Non-Critical	Critical	Non-Recoverable
ICH PCBT TEMP	Temperature sensor close to ICH	37	45C		5.00	12.00	55.00	70.00	85.00
FRONT PCBB TEMP		38	40C		0.00	7.00	55.00	70.00	85.00
CPU1 JUNC TEMP	CPU1 PECI measurement	92	-45C		-85.00	-75.00	-30.00	-10.00	
CPU2 JUNC TEMP	CPU1 PECI measurement	93	-45C		-85.00	-75.00	-30.00	-10.00	
CPU 1 VCC	CPU1 Core voltage.	94	1.20		1.00			1.00	1.60
CPU 2 VCC	CPU2Core voltage.	95	1.20		1.00			1.00	1.60
FI1 JUNC TEMP	Fabric Interface GbE Controller Temp	9C	60C		0.00	10.00	70.00	85.00	100.00
FI2 JUNC TEMP	Fabric Interface GbE Controller Temp	9D	60C		0.00	10.00	70.00	85.00	100.00
CPU1 PCBT TEMP	Temperature sensor under CPU1	9E	45C		5.00	12.00	55.00	70.00	85.00
CPU2 PCBT TEMP	Temperature sensor under CPU1	9F	45C		5.00	12.00	55.00	70.00	85.00
CPU2 PCBB TEMP	Temperature sensor on bottom of PCB close to CPU2.	A0	45C		5.00	12.00	55.00	70.00	85.00
INLET PCBB TEMP	PCB temperature sensor at air inlet	A1	40C		-5.00	5.00	50.00	60.00	75.00
AMBIENT AIR TEMP	Inlet air temperature sensor	A2	40C		-5.00	5.00	50.00	60.00	75.00

5.3.2 Temperature Sensor Locations

The SBC is equipped with several temperature sensors. Refer to Figure 28 for the location of all the on-board temperature sensors.

Figure 28. On-board temperature sensor locations



5.3.3 Processor Events

The following processor events are supported:

- **Thermal trip -** A thermal trip error indicates that the processor junction temperature has reached a level where permanent silicon damage may occur. Upon THERMTRIP assertion, the IPMC powers down the boards. PROCHOT is asserted before THERMTRIP asserts.
- **IERR** - The processor asserts IERR as the result of an internal error. Assertion of IERR# is usually accompanied by a SHUTDOWN transaction on the processor system bus.
- **FRB-1 -** Built In Self Test (BIST) failed
- **FRB-2** - Board hang during EFI BIOS Post

- **FRB-3 timeout** - The FRB-3 algorithm is used to detect whether the boot strap processor is healthy and can run the EFI BIOS. The default FRB-3 timer is 10 seconds. The assumed initial condition is that both processors are enabled. The basic algorithm is followed on each power up or system reset:
  - At power up/reset, the IPMC starts an internal FRB-3 timer.
  - In a good system, the EFI BIOS issues the IPMI Set Watchdog Timer (WDT #1) command with the "timer use" byte configured for the EFI BIOS. When the IPMC receives this command, the IPMC starts the IPMI Watchdog Timer, and then stops the internal FRB3 timer. At this point, the FRB-2 phase starts.
  - In a failing system, the EFI does not issue the IPMI Set WDT command to the IPMC, and the IPMC FRB-3 timer expires.
  - The IPMC logs an FRB-3 failure event against the failing processor sensor, logs a processor disable event against that processor sensor, then causes the payload power to reset.
- **Processor configuration error -** EFI BIOS detected processor configuration error
- **Processor presence** - Indicates if the processor is present in the socket.
- **Processor disabled**
- **PROCHOT** - Indicates if the processor has entered automatic thermal throttling mode due to high CPU die temperature.

5.3.4 FB-DIMM Memory Events

Intel® 5000P MCH provides extensive logic built into the hardware to detect and correct correctable errors and detect uncorrectable errors. For either type of error, an SMI is generated to the processor so that the EFI BIOS code can take the appropriate actions. On the MPCBL0050, such actions include detecting the error and sending a memory error event message to the IPMC firmware over the KCS interface. IPMC also logs information events at board boot. These events provide information on memory configuration detected by BIOS.

Note: For details on memory detection and correction functionality, please refer to Intel® 5000P data sheet available at www.intel.com.

5.3.4.1 Correctable Errors

MCH provides detection and correction of any x4 or x8 DRAM device failure. When an error is detected, the chipset sends an SMI to BIOS. Then BIOS is responsible for sending an event to the IPMC. DIMM information is also sent to the IPMC as part of the event.

For each error occurence, the counter is incremented. To disregard intermittent errors, the counters are decremented by one every 6 days.

5.3.4.2 Logging Threshold

The system detects, corrects, and logs correctable errors as long as these errors occur infrequently (the system should continue to operate without a problem). Occasionally, correctable errors are caused by the persistent failure of a single component. Although these errors are correctable, continual calls to the error logger can affect system performance, preventing further useful work.

For this reason, the system counts correctable errors and disables reporting if errors occur too frequently. Error correction remains enabled, but calls to the error handler are disabled. This allows the system to continue running despite a persistent correctable failure.

BIOS initializes the correctable error counters to a value of 10 for correctable ECC errors. These counters are on a per-rank basis. A rank applies to a pair of FBDIMMs on adjacent channels functioning in lock-stepped mode.

Note: As there are 2 ranks per DIMM, logging errors for a single FB-DIMM may be disabled after detecting 10 to 19 ECC errors.

When the ECC counter for Rank reaches the threshold of 10 errors, BIOS stops logging records to SEL with sensor type 0Ch (Memory) and a single record is logged to SEL with sensor type 10h (Logging Disabled) with offset 00h (Correctable Memory Error Logging Disabled). BIOS adds an entry to the event log to indicate that logging for that type of error has been disabled. BIOS re-enables logging and SMIs the next time the system is rebooted.

The system BIOS implements this feature for correctable bus errors. Uncorrectable Errors

The MCH can also detect uncorrectable errors and generate SMIs to the BIOS. If the error is in a data area, BIOS generates an event to the IPMC.

If the error is in a code area, BIOS itself may be unresponsive and it may not be possible to detect whether an uncorrectable error has occurred. In that case, there are other functionalities in the management subsystem to log payload failure:

- The IPMC provides an additional SMI timeout sensor through which IPMC firmware continually monitors for SMI assertions and sets a 90-second timeout for clearing SMI assertions. If an SMI is not cleared within 90 seconds, the IPMC firmware generates an SMI timeout sensor event. The system manager can set policy for action to take on an SMI timeout, which can be pre-configured with the IPMC firmware to generate a blade power down, power cycle, cold reset, or warm reset.
- An OS or application executing on the processor can set up a watchdog timer with the IPMC. When it times out, a WDT timeout event is sent out and the IPMC takes the action that has been set up in the WDT (hard reset, power down, power cycle, or do nothing).

5.3.4.3 Other Memory Events

During board boot, BIOS logs several events about about memory configuration. The information contains data on slots populated and memory sizes for populated modules.

5.3.5 System Firmware Progress (POST Error)

The EFI BIOS performs a power-on self-test (POST) at initialization. POST examines all major board components and logs errors if detected to the SEL by generating a System Firmware Progress (sensor type 0Fh) event. Refer to Section 8.1 for the list of possible errors.

5.3.6 Port 80h POST Codes

As the EFI BIOS goes through its initialization process, it sends progress codes to port 80h. These codes are useful for test debug purposes. For remote management purposes, the IPMC firmware provides the capability to snoop and capture up to five consecutive codes written to port 80h. These codes are captured automatically by the IPMC firmware when the board goes through either cold or warm reset or by an explicit IPMI command received by the IPMC via the available interfaces (IPMB or KCS). Please refer to Table Note: for the POST codes list.

5.3.7 Critical Interrupt

MCH has build in capability to detect errors on the PCI and FSB interfaces. In case of the errors detection BIOS SMI code will be executed and log appropriate events to SEL.

5.3.8 IPMB Link Sensor

The MPCBL0050 provides two IPMB links to increase communication reliability with the shelf manager and other IPM devices on the IPMB bus. These IPMB links work together for increased throughput where both buses are actively used for communication at any point. A request might be received over IPMB Bus A, and the response is sent over IPMB Bus B. Any requests that time out are retried on the redundant IPMB bus. In the event of any link state changes, the events are written to the SEL. The IPMC monitors the bus for any link failure and isolates itself from the bus if it detects that it is causing errors on the bus. Events are sent to signify the failure of a bus or, conversely, the recovery of a bus.

5.3.9 FRU Hot Swap

The Hot Swap event message conveys the current state of the FRU, the previous state, and a cause of the state change as can be determined by the IPMC. Refer to the PICMG 3.0 specifications for further details on the hot swap state.

5.3.10 Ethernet Link Status

The IPMC firmware monitors the Ethernet link status (present/absent) of all available ports on the SBC (two Base interface, four Fabric interface). One sensor is provided for each of the six Ethernet links. Events are generated by the IPMC firmware when the link status of the ports change.

5.3.11 Power Feeds, Power Supply, and Fuses

Power Feeds A and B, fuses on power feeds, as well as power good signals from all onboard voltage converters monitored. In case of failure appropriate events are logged to SEL.

5.3.12 IPMC Watchdog Timer Reset

As per PICMG 3.0 requirements (section 3.2.4.6.3), the MPCBL0050 provides a watchdog timer that can reset the IPMC if the firmware hangs. After a reset, when IPMC restarts, an event is generated indicating that the IPMC was reset due to watchdog timer expiration.

5.4 Field Replaceable Unit (FRU) Information

The IPMC provides Field Replaceable Unit (FRU) information for the base board it manages and major replaceable modules on the SBC, such as the RTM. FRU information contained in the SBC includes data to describe the SBC as per the PICMG 3.0 Specification. Additional multirecords are provided to enable:

- the BIOS to write CPU information and BIOS version number to FRU data correctly;
- customers to write their custom inventory information that is not part of any inventory data provided by the MPCBL0050.

Inventory information on the MPCBL0050 is divided into different areas that are written to at manufacturing time. Some information may be written by the BIOS during initialization, which are identified where necessary.

The following sections are definitions for the multirecords implemented by the firmware as part of the FRU data.

5.4.1 Common Header

A common header is mandatory for all FRU Inventory Device implementations. It contains version information for the overall information and offsets to other information areas.

5.4.2 Board Area

The board area provides access to the following information, which is written when the board is manufactured:

- Manufacturing date and time
- Board manufacturer name
- Board product name
- Board serial number
- Board part number

5.4.3 Product Area

The product area provides access to following information, which is written when the board is manufactured:

- Manufacturer name (same as board manufacturer name)
- Product name (same as board product name)
- Product part/model number
- Product version
- Product serial number (same as board serial number)
- Asset tag

5.4.4 Multirecord Area

The multirecord area on the MPCBL0050 is further divided into nine different types:

- Board Point-to-Point Connectivity Record for E-Keying
- Carrier Information Table Record
- Carrier Activation and Current Management Record
- Carrier Point-to-Point Connectivity Record for E-Keying (AdvancedMC)
- Carrier Point-to-Point Connectivity Record for E-Keying (Rear Transition Module)
- AdvancedMC Point-to-Point Interface Record for E-Keying
- Rear Transition Module Point-to-Point Interface Record for E-Keying
- Base interface MAC address record
- Board UUID record
- General board info record with version info for BIOS, Firmware, CPU and RAM Information

5.4.4.1 BIOS, Firmware, CPU and RAM Information

An OEM record is provided that allows remote access to additional board-specific information that is not defined in the Platform Management FRU Information Storage Definition v1.0 Specification. Table 36 lists the information that is available on the MPCBL0050 SBC. Note that the table is an illustration. The actual structure that is implemented can be found using the FRU file itself.

Table 36. Additional board-specific information

Information	Size (Bytes)	Data	Type
Manufacturer ID (Intel IANA number)	3	0x000157 (LSB first)	Binary
Record Version	1	1	Binary
Type/Length	1	1	Binary
No. of CPUs	1	x (updated by BIOS on initialization)	Binary
Type/Length	1	2	Binary
RAM Information	2	X (in units of 1 MB)	Binary
Type/Length	1	0x10	Binary
NOT used on MPCBL0050	16	All zero	Binary
Type/Length	1	2	Binary
IPMC Boot block version	2	Major, minor	Binary
Type/Length	1	2	Binary
IPMC Firmware version	2	Major, minor	Binary
Type/Length	1	1	Binary
FPGA version	1	X (populated by IPMC)	Binary
Type/Length	1	1	Binary
Board HW revision	1	X (populated in the factory)	Binary
Type/Length	1	1	Binary
BIOS FWH Selected	1	0/1 (updated by IPMC)	Binary
Type/Length	1	0x3F	Binary
BIOS Version	63 (max)	Version of first BIOS image (as zero terminated string), updated by BIOS at power up or BIOS update utilities after successful BIOS update	ASCI
Type/Length	1	0x3F	Binary
BIOS Version	63 (max)	Version of first BIOS image (as zero terminated string), updated by BIOS at power up or BIOS update utilities after successful BIOS update	ASCI
End of Fields	1	0xC1	Binary

5.4.5 MPCBL0050 FRU Record

On the MPCBL0050 SBC, the FRU information pertaining to the board has been structured as below per the Platform Management FRU Information Storage Definition v1 specification.

The “Common”, “Board”, and “Product” fields are used by Intel during manufacturing and are used to program all the relevant board information into the FRU.

The “MULTIREC” is an Intel multi-record area allocated for use by the EFI BIOS to program the relevant version information to the FRU during power up.

The following FRU record is an illustration of typical FRU information for the MPCBL0050 and is subject to change. Please refer to the MPCBL0050 IPMC release package for the latest FRU file.

```
_LF_NAME       'MPCBL0050_101.fru' // Name for this load file

_LF_VERSION    '1.01'              // Version of this load file

_LF_FMT_VER    '1.50'// Version of the load file format

_IPMI_VERSION  '1.00'// IPMI format version (FRU spec version)


_FRU (

_FRU_TITLE     'CPU Board' // FRU Title

_START_ADDR    8000        // Start Address

_DATA_LEN      03A7        // Data Length

_NVS_TYPE      'IMBDEVICE' // Non-volatile Storage Type

_DEV_ADDRESS   20          // Device Address


_SEE_COMMON

   01                  // Common Header Format Version

   00                 // Internal Use Area Starting Offset (in multiples of 8 bytes)

   00                 // Chassis Info Area Starting Offset (in multiples of 8 bytes)

   01                  // Board Info Area Starting Offset (in multiples of 8 bytes)

   12                 // Product Info Area Starting Offset (in multiples of 8 bytes)

   27                  // MultiRecord Area Starting Offset (in multiples of 8 bytes)

   00                  // Pad

   C5                  // Common Header Checksum


_SEE_BOARD
```

```
   01                 // Board Info Area Format Version Bit Fields
   11                 // Board Info Area Length (in multiples of 8 bytes)
   00                 // Language Code
   000000             // Mfg. Date/Time
   D1                 // Board Manufacturer
   'Intel Corporation'
   C9                 // Board Product Name
   'MPCBL0050'
   CC                 // Board Serial Number
   '000000000000'
   CA                 // Board Part Number (PBA# number, for example "D49594-001")
   'DXXXXX-XXX'
   CC                 // FRU File ID
   'FRU Ver 1.01'
   D4                 // Customer Defined Field 1
   '                    '
   D4                 // Customer Defined Field 2
   '                    '
   D4                 // Customer Defined Field 3
   '                    '
   C1                 // End of Area tag

_SEE_PRODUCT
   01                 // Product Info Area Format Version Bit Fields
   15                 // Product Info Area Length (in multiples of 8 bytes)
   00                 // Language Code
   D1                 // Product Manufacturer Name
   'Intel Corporation'
   CD                 // Product Name (full product code, for example
"MPCBL0050B01A")
   'MPCBL0050XXXX'
   CA                 // Product Part Number (TA# number, example: "D71156-001")
   'DXXXXX-XXX'
   C0                 // Product Version (not used)
   CC                  // Product Serial Number
   '000000000000'
   D4                  // Asset Tag
   '00000000000000000000'
   D0                  // FRU File ID - available for customization
   '0000000000000000'
   D4                  // Customer Defined Field 1
   '                    '
   D4                  // Customer Defined Field 2
   '                    '
   D4                  // Customer Defined Field 3
   '                    '
   C1                  // End of Area tag


_SEE_MULTIREC
   // Board Point-to-Point Connectivity Record for E-Keying
   C0                  // Record Type ID
   02                  // Version Information
   32                  // Record Length
   6E                  // Record Checksum (zero checksum)
   9E                  // Header Checksum (zero checksum)
   5A3100              // Manufacturer ID (PICMG)
                       // As per PICMG3.0 v.90 Point-to-point Connectivity Record
   14                // PICMG Record ID For Board Point-to-point Connectivity Record
   00                  // Record Format Version
   01                  // OEM GUID Count
   37 CE 7B B5 E6 AB 43 77 91 A3 1C AA 90 C9 75 66 // 1st GUID: Kontron AT8400
   01 11 00 00 // Link Descriptor for Base Ethernet CH1
   02 11 00 00 // Link Descriptor for Base Ethernet CH2
   41 23 00 00                                // Link Descriptor for Fabric Mux 1
Port 0 & Ethernet 1 Port 1
   41 21 00 00                                // Link Descriptor for Fabric Mux 1
Port 0
   42 23 00 00                                // Link Descriptor for Fabric Mux 2
Port 0 & Ethernet 2 Port 1
```

```
   42 21 00 00                                // Link Descriptor for Fabric Mux 2
Port 0
   81 01 0F 00 // Link Descriptor for Update Channel, matches with 1st GUID


   // Carrier Information Table Record (AMC&RTM)
   C0                  // Record Type ID
   02                  // Version Information
   09                  // Record Length
   4A                  // Record Checksum (zero checksum)
   EB                  // Header Checksum (zero checksum)
   5A3100              // Manufacturer ID (PICMG)
                       // // As per PICMG3.0 v.90 Carrier Information Table Record
   1A                  // PICMG Record ID For Carrier Information Table
   00                  // Record Format Version
   01                  // AMC.0 Extension
   02                  // AMC Site Count
   05                  // AMC Site
   09                  // RTM Site


   // Carrier Activation and Current Management Record (AMC&RTM)
   C0                  // Record Type ID
   02                  // Version Information
   0F                  // Record Length
   ED                  // Record Checksum (zero checksum)
   42                  // Header Checksum (zero checksum)
   5A3100              // Manufacturer ID (PICMG)
                       // // As per PICMG3.0
   17                  // PICMG ID For Carrier Activation and Current Management
Record
   00                  // Record Format Version
   32 00 // = 5A   // Max Internal Current in 1/10 Amp Increments
   05                  // Activation Readiness in Seconds
   02                  // Module Descriptor Count
   // AMC Activation and Power Descriptors
   7A                  // IPMB-L address of the AMC Bay
   1B // = 2.7A       // max current, in tenths of Amps at 12V, that can be routed
to the AMC Bay

   FF                  // Field NOT USED

   // RTM Activation and Power Descriptors

   82                  // IPMB-L address of the AMC Bay

   23 // = 3.5         // max current, in tenths of Amps

   FF                  // Field NOT USED


   // Carrier Point-to-Point Connectivity Record for E-Keying (for AMC)

   C0                  // Record Type ID

   02                  // Version Information

   43                  // Record Length

   BA                  // Record Checksum (zero checksum)

   41                  // Header Checksum (zero checksum)

   5A3100              // Manufacturer ID (PICMG)

                       // // As per PICMG3.0 v.90

   18             // PICMG Record ID For Carrier Point-to-Point Connectivity Record

   00                  // Record Format Version

   05                  // Resource ID, bits: [7]=1 (AMC), [3:0]=5 (Site id 5),
[5:4]=rsvd

   14                  // Point to Point Port Count

   01 00 00 // AMC P0 -> FIC GbE port 0

   01 21 00 // AMC P1 -> FIC GbE port 1

   02 41 00 // AMC P2 -> Basebrd SAS cntrl

   89 69 00 // AMC P3 -> RTM SAS cntrl (P9)

   03 90 00 // AMC P4 -> PCI Ex x4 (MCH)

   03 B1 00 // AMC P5 -> PCI Ex x4 (MCH)

   03 D2 00// AMC P6 -> PCI Ex x4 (MCH)

   03 F3 00// AMC P7 -> PCI Ex x4 (MCH)

   03 14 01// AMC P8 -> PCI Ex x8 (MCH)

   03 35 01// AMC P9 -> PCI Ex x8 (MCH)

   03 56 01// AMC P10 -> PCI Ex x8 (MCH)

   03 77 01// AMC P11 -> PCI Ex x8 (MCH)

   89 A8 01// AMC P13 -> RTM P8

   89 C7 01// AMC P14 -> RTM P7
   89 E6 01// AMC P15 -> RTM P6

   89 05 02// AMC P16 -> RTM P5

   89 24 02// AMC P17 -> RTM P4

   89 43 02// AMC P18 -> RTM P3

   89 62 02// AMC P19 -> RTM P2

   89 81 02// AMC P20 -> RTM P1


   // Carrier Point-to-Point Connectivity Record for E-Keying (for RTM)

   C0                  // Record Type ID

   02                  // Version Information

   25                  // Record Length

   70                  // Record Checksum (zero checksum)

   A9                  // Header Checksum (zero checksum)

   5A3100              // Manufacturer ID (PICMG)

                       // // As per PICMG3.0 v.90

                     18 // PICMG Record ID For Carrier Point-to-Point Connectivity
Record

   00                  // Record Format Version

   09                  // Resource ID

   0A                  // Point to Point Port count

   0400 00  //  RTM->BMC conn (as in MPCBL0040)

   850D 01  //  RTM -> AMC connection P8->P13

   85EE 00  //  RTM -> AMC connection P7->P14

   85CF 00  //  RTM -> AMC connection P6->P15

   85B0 00  //  RTM -> AMC connection P5->P16

   8591 00  //  RTM -> AMC connection P4->P17

   8572 00  //  RTM -> AMC connection P3->P18

   8553 00  //  RTM -> AMC connection P2->P19

   8534 00  //  RTM -> AMC connection P2->P20

   8523 01  //  RTM P9 -> AMC P3 (SAS)


   // AMC Point-to-Point Interface Record for E-Keying

   C0                  // Record Type ID

   02                  // Version Information
   37                  // Record Length

   BD                  // Record Checksum (zero checksum)

   4A                  // Header Checksum (zero checksum)

   5A3100              // Manufacturer ID (PICMG)

                       // // As per PICMG3.0 v.90

                     19 // PICMG Record ID For AMC Point-to-Point Interface Record

   00                  // Record Format Version

   00                  // GUID Count

   05                                           // [7]=0(Carrier) [5:4]=0h(RSVD)
[3:0]=0x5h(ID B1)

   04                                             // Channel Count

   E0 FF FF                                     // AMC Channel Descriptor 0

   E1 FF FF                                     // AMC Channel Descriptor 1

   E2 FF FF                                     // AMC Channel Descriptor 2

   A4 98 F3                                     // AMC Channel Descriptor 3

   00 51 00 00 FC                               // AMC Channel 0

   01 51 00 00 FC                               // AMC Channel 1

   02 71 20 00 FC                               // AMC Channel 2

   03 2F 00 00 FC                               // AMC Channel 3 (Lane 0-3, Exact
match)

   03 21 00 00 FC                          // AMC Channel 3 (Lane 0, Exact match)

   03 2F 00 00 FE                            // AMC Channel 3 (Lane 0-3, Matches
with '01b')

   03 21 00 00 FE                               // AMC Channel 3 (Lane 0, Matches
with '01b')


   // RTM Point-to-Point Interface Record for E-Keying

   C0                  // Record Type ID

   02                  // Version Information

   74                  // Record Length

   D9                  // Record Checksum (zero checksum)

   F1                  // Header Checksum (zero checksum)

   5A3100              // Manufacturer ID (PICMG)

                       // // As per PICMG3.0 v.90

                     19 // PICMG Record ID For RTM Point-to-Point Interface Record

   00                  // Record Format Version
   05                 // OEM GUID Count
   FC FD 47 24 37 08 1A AB 67 4E 23 8E 97 5D FF A8 // GUID 1 (MPCBL0040)
   08 DA 5A 82 B1 FA 1C BA 53 45 55 75 90 74 CC BE   // GUID 2 (Tvl2)
   BA BB 6A 5A 6B E7 DC 87 15 43 DE 1A 3F 17 38 2C   // GUID 3 (RSVD)
   AC 76 75 52 50 0B BC B4 C1 44 FF 7F 64 95 48 A9   // GUID 4 (RSVD)
   66 66 B7 85 54 A7 1E AB 56 49 D2 3D EA 91 6A AA   // GUID 5 (RSVD)
   09                                            // [7]=0(Carrier) [5:4]=0h(RSVD)
[3:0]=0x9h(ID C1)
   01                                              // Channel Count
   E0 FF FF                                     // RTM Channel Descriptor 0
   00 01 0F 00 FC                             // RTM Channel 0
   00 11 0F 00 FC                             // RTM Channel 1
   00 21 0F 00 FC                             // RTM Channel 2
   00 31 0F 00 FC                             // RTM Channel 3
   00 41 0F 00 FC                             // RTM Channel 3
   //
   // Base Interface MAC Addresses Record (MPCBL0050 specific)
   //
   D0                 // Record Type ID -
   02                 // Version Information
   14                 // Record Length
   A0                 // Record Checksum (zero checksum)
   7A                 // Header Checksum (zero checksum)
   570100             // Manufacturer ID
   01                 // Record Version
   // MAC address for 1st channel of Base Interface
   03   // Interface Type
   00   // Channel number
   00 00 00 00 00 00 // Value of MAC address (6 bytes, binary encoded)
   // MAC address for 2nd channel of Base Interface
   03   // Interface Type
   01   // Channel number
   00 00 00 00 00 00 // Value of MAC address (6 bytes, binary encoded)
```

```
//
// Board UUID Record (MPCBL0050 specific)
//
D0                 // Record Type ID -
02                 // Version Information
14                 // Record Length
F8                 // Record Checksum (zero checksum)
22                 // Header Checksum (zero checksum)
570100             // Manufacturer ID
B0                 // Record Version
// UUID contents (16 bytes = 128 bits, every board has its unique UUID)
00 00 00 00
00 00 00 00
00 00 00 00
00 00 00 00

// BIOS Information Record (MPCBL0050 specific)
C0                 // Record Type ID
02                 // Version Information
A8                 // Record Length
C9                 // Record Checksum (zero checksum)
CD                 // Header Checksum (zero checksum)
570100             // Manufacturer ID
01                 // Record Version
01                 // Type/Length
02                 // CPU No.s
02                 // Type/Length
00 00              // RAM Info

10                 // Type/Length
00            // Number of PMCs (NOT USED in MPCBL0050, shall be 0)
00 00 00 00 00     // Reserved, shall be 0 (formely PMC1 info)
00 00 00 00 00     // Reserved, shall be 0 (formely PMC2 info)
00 00 00 00 00     // Reserved, shall be 0 (formely PMC3 info)
```

```
02       // Type/Length
00 00    // FW Boot Version (Major, Minor)
03       // Type/Length
00 00 00 // FW Version (Major, Minor, Build)

01                // Type/Length
00                // FPGA Version Number

01                // Type/Length
00                // Board HW Version Number

01                // Type/Length
00                // BIOS FWH Selected  0/1

00                 // Type/Length
00 00 00 00 00 00 00 00 // BIOS Version
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00
00                      // Type/Length
00 00 00 00 00 00 00 00 // BIOS Version
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00
C1 // End of Fields
```

```
                        // Utilities Info OEM record
    C0                  // Record Type ID
    82                  // Version Information
    0B                  // Record Length
    69                  // Record Checksum (zero checksum)
    4A                  // Header Checksum (zero checksum)
    570100              // Manufacturer ID (PICMG)
    04                  // Intel OEM record subtype
    01                  // Record Format Version
    0F 08               // Product ID
    20                  // Intel IPMI Device ID 20=IPMC, 30=AMC, 40=RTM
    01 01               // FRU file version visible by sbcupdate
    01                  // Flags

)
```

5.4.6 FRU Area for Customer-Specific Information

In the SBC FRU, an EEPROM is used to store all the FRU records. Customers can program user-specific information into the customer FRU MRA records. The FRU structure for the MPCBL0050 SBC is defined by the FRU file. Users can utilize the standard IPMI command to write this data.

There is room for customers to write data to the FRU after the Intel OEM-MRA records at the end of the FRU file.

Note: The FRU area has a maximum size of 512bytes so additional MRA records added by the customer shall not exceed this limit.

5.4.7 Writing to the Customer FRU MRA

The standard FRU IPMI read/write command is used to write to the customer FRU MRA.

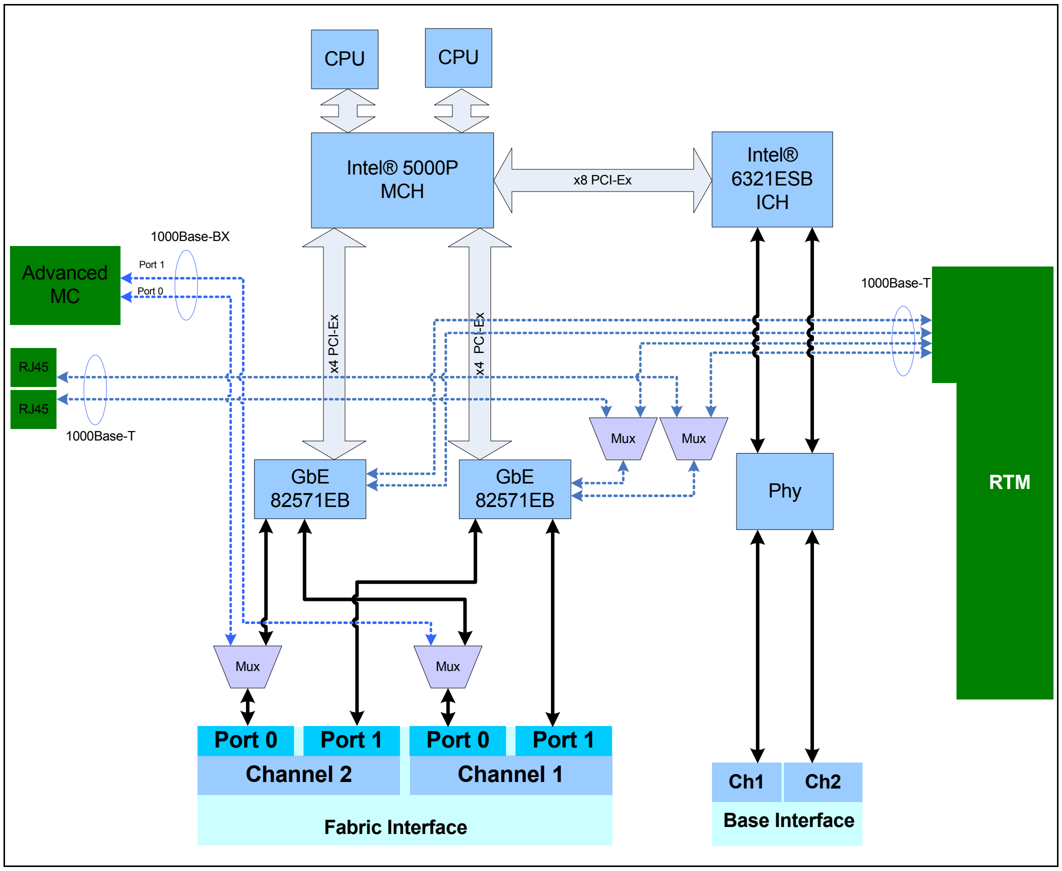
5.5 E-Keying

E-keying has been defined in the PICMG 3.0 Specification to prevent board damage, prevent misoperation, and verify fabric compatibility. The FRU data contains the board point-to-point connectivity record as described in Section 3.7.2.3 of the PICMG 3.0 Specification.

E-Keying is provided for connectivity between the MPCBL0050 board and the RTM and the MPCBL0050 and the AdvancedMC slot as described the in Section 3.9 and 3.7 of the AMC.0 RC.1.1 specification. The Set/Get AdvancedMC Port State IPMI commands defined by the AMC.0 specification are used for either granting or rejecting the E-keys.

Upon management power-on, the firmware sets the Gigabit Ethernet connectivity (fabric interface) to the backplane by default. Refer to Figure 29 for the fabric interface Ethernet routing. In the figure below, bolded lines to the backplane fabric interface show the default board settings.

Figure 29. Fabric interface Ethernet routing



The IPMC stores the configuration that determines where the fabric ports are routed; to the front panel or to the backplane. An OEM IPMI command can be sent by the EFI BIOS or external software to get/set the fabric port direction setting.

Note: The user can send this OEM IPMI command when the IPMC is operational, however changes to the routing are not applied immediately. The user **must reset the board to activate the settings.** This can be initiated separately via IPMI or AdvancedTCA-defined IPMI commands.

Table 37 lists connections to the base and fabric interfaces for E-keying purposes, and also lists the link descriptors for the two Gigabit Ethernet channels connected to the base interface and the four Gigabit Ethernet channels on the fabric interface. Table 38 lists AdvancedMC link descriptors.

Table 37. E-Keying board point-to-point connectivity link descriptor list

#	Link Descriptor	Link Grouping ID [31:24]	Link Type Extension [23:20]	Link Type [19:12]	Link Designator			Link Desc Value
					Port 0 - 3 Flags {11:8}	Interface {7:6}	Channel [5:0]	
1	Base Interface Ethernet Ch 1, Port 0	0	0000	00000001	0001	00	000001	0x00001101
2	Base Interface Ethernet Ch 2, Port 0	0	0000	00000001	0001	00	000010	0x00001102
3	Fabric Interface Ethernet Ch 2, Port 0 and Port 1	0	0000	00000010	0011	01	000001	0x00002341
4	Fabric Interface Ethernet Ch 2, Port 0	0	0000	00000010	0001	01	000001	0x00002141
5	Fabric Interface Ethernet Ch 1, Port 0 and Port 1	0	0000	00000010	0011	01	000010	0x00002342
6	Fabric Interface Ethernet Ch 1, Port 0	0	0000	00000010	0001	01	000010	0x00002142

Table 38. AdvancedMC (ID B2) link descriptors

#	Link Descriptor	Reserved	Asymmetrical Match	Link Group ID	Link Type Ext.	Link Type	Link Designator	Link Desc Value
		[39:34]	[33:32]	[31:24]	{23:20}	{19:12}	[11:0}	
1	AdvancedMC Channel 0	111111b	00b (Exact)	00h	0h	05h	100h	0xFC00005100
2	AdvancedMC Channel 1	111111b	00b (Exact)	00h	0h	05h	101h	0xFC00005101
3	AdvancedMC Channel 2	111111b	00b (Exact)	00h	2h	07h	102h	0xFC00207102
4	AdvancedMC Channel 3	111111b	00b (Exact)	00h	0h	02h	F03h	0xFC00002F03
5	AdvancedMC Channel 3	111111b	00b (Exact)	00h	0h	02h	103h	0xFC00002103
6	AdvancedMC Channel 3	111111b	10b (matches 01bt)	00h	0h	02h	F03h	0xFE00002F03
7	AdvancedMC Channel 3	111111b	10b (matches 01bt)	00h	0h	02h	103h	0xFE00002103

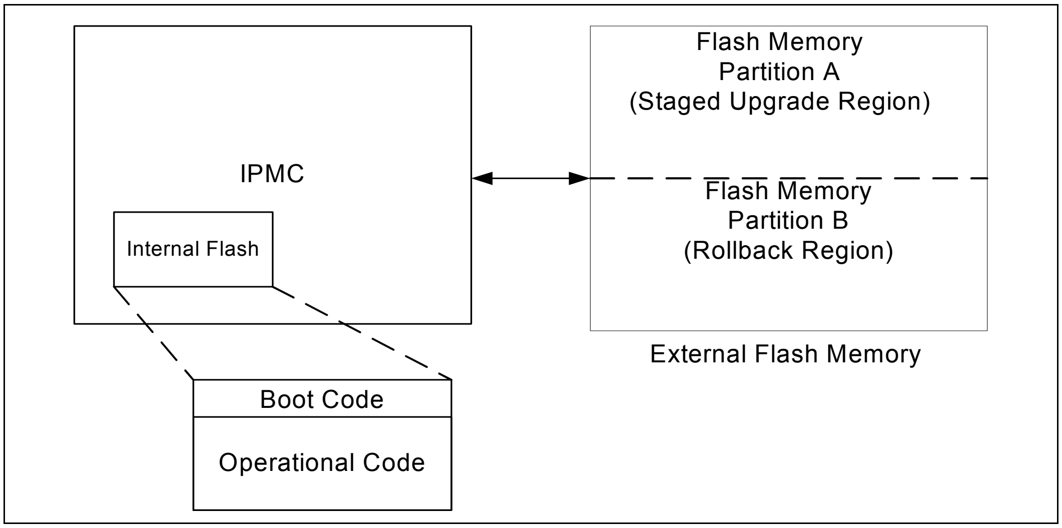
5.6 IPMC Platform Event Filtering (PEF)

The IPMC supports a maximum of 20 PEF entries. Initially, first 8 PEF table entries are reserved. Remaining entries (up to 20) are available for the user to configure. The only action available is to turn the Health LED to red.

5.7 IPMC Firmware Code Organization

Figure 30 shows the components of IPMC flash on the MPCBL0050.

Figure 30. IPMC flash hHigh-level block diagram



5.7.1 Functional Description

The IPMC flash comprises two components:

- **Internal Flash on IPMC** – 512 KB of memory. Used to store the primary firmware image. It consists of boot block and operational code (known as the execution image):
  - **Boot Block** – This area contains the code necessary to update the IPMC main operational code (opcode). This is a recovery block and not updated in the field.
  - **Operational Code** – This area contains the active operational code. Which is loaded to execution RAM when IPMC starts.
- **External Flash Memory** – External 2 MByte flash memory. The external flash device is partitioned into numerous regions and includes two regions for the Online Firmware Update feature.
  - **Partition A** (Staged Upgrade Region) is used to store the staged firmware image. For every staged firmware update initiated by a user, the staged image is copied to this location (known as the staging image).
  - **Partition B** (Rollback Region) is used to store a backup copy of the original operational code image from Internal IPMC flash before a staged update firmware takes place (known as the rollback image).

5.8 IPMC Firmware Updates

5.8.1 Update Modes

There are two update modes:

- **Direct Mode** – This upgrade procedure happens when the user updates the IPMC firmware directly to the internal flash on the IPMC. The user will not have the ability to back up the “old” operational image.

- **Staged Update Mode** – The Staged Update feature allows the IPMC operational code to be updated while the system is online (OS is running). After the new image has been staged, it is copied to the internal flash of the IPMC upon completion of the staged upgrade. If the switchover fails, or at the user's discretion, the firmware may be rolled back to the previous version.

  The advantage of running this mode are is the user can store the old firmware image to a rollback region (for redundancy or backup purposes).

Note: It is advised to always use staged Update Mode as it provides automatic recovery to operational IPMC code in case of upgrade failure.

See Section 10.6.2, "IPMC Operational Code Firmware Update Modes" on page 194 for more details firmware update modes and the procedure for updating IPMC firmware.

5.9 Ejector Mechanism

In addition to captive retaining screws, the MPCBL0050 SBC has two ejector mechanisms to provide a positive cam action, which ensures that the blade is properly seated and helps assist during blade extraction. The bottom ejector handle also has a micro switch that is connected to the IPMC to determine if the board has been properly inserted.

5.10 Hot Swap LED

The MPCBL0050 SBC supports one blue Hot Swap LED, mounted on the front panel. See Figure 15 for its location. This LED indicates when it is safe to remove the SBC from the chassis. The on-board IPMC drives this LED to indicate the Hot Swap state. See Table 39.

When the lower ejector handle is disengaged from the faceplate, the Hot Swap switch on the board asserts a signal to the IPMC, and the IPMC moves from the M4 state to the M5 state. At the M5 state, the IPMC asks the CMM (or Shelf Manager) for permission to move to the M6 state. The Hot Swap LED indicates this state by blinking on for about 100 milliseconds, followed by 900 milliseconds in the off state. This occurs as long as the SBC remains in the M5 state. Once permission is received from the CMM or higher level software, the SBC moves to the M6 state.

The CMM or higher level system software may be configured to reject the request to move to the M6 state. If this occurs, the Hot Swap LED returns to a solid off condition, indicating that the SBC has returned to M4 state.

If the SBC reaches the M6 state, either through an extraction request through the lower ejector handle or a direct command from higher-level software, and an a ACPI-enabled OS is loaded on the SBC, the IPMC communicates to the OS that the module must discontinue operation in preparation for removal. The Hot Swap LED continues to flash during this preparation time, just like it does in the M5 state. When the main board power is successfully removed from the SBC, the Hot Swap LED remains lit, indicating it is safe to remove the SBC from the chassis.

Warning: Removing the SBC prematurely can lead to corruption of files on the hard drive.

Table 39. Hot Swap LED

LED Status	Meaning
Off	Normal (active) status
Blinking Blue	Preparing for removal/insertion: Long blink indicates activation is in progress, short blink when deactivation is in progress.
Solid Blue	Ready for hot swap

5.11 ACPI

ACPI gives the operating system direct control over the power management and Plug and Play functions of a computer. The use of ACPI with the MPCBL0050 SBC requires an operating system that provides ACPI support. ACPI features include:

- Plug and Play (including bus and device enumeration) and APM support (normally contained in the BIOS)
- Power management control of individual devices, add-in boards (some AdvancedMC cards may require an ACPI-aware driver), and hard disk drives
- A soft-off feature that enables the operating system to power off the computer
- Support for an IPMC firmware command switch

For ACPI enabled operating systems, the IPMC is configured such that ACPI mode is enabled via the BIOS ACPI Source Language (ASL) code. When ACPI shut down request is received, the IPMC will pulse the ICH power button for 100ms which will start shutting down the OS. Once OS is shutdown, ICH will assert the SLP_S5# (sleep S5) signal. If the OS fails to shutdown after 3 minutes (indicated by the fact that the SLP_S5# has not been asserted), the IPMC PICMG State Machine will timeout and force the payload off with a 4-second power button override to the ICH.

For non-ACPI enabled operating systems, the IPMC will perform a 4-second power button override to the ICH.

Note: SLP_S5# is one of the power management signal from ICH. The signal is used to shut power off to all non-critical systems when in the S5 (Soft Off) state.

The IPMC management features are designed to work in conjunction with the ACPI EFI BIOS and hardware features of the baseboard. The following subsections illustrate these capabilities.

5.12 Reset Types

The following topics describe the two types of reset requests and the boot relationships among them. The two types of reset requests available on the MPCBL0050 are:

- Hard reset request (always results in a cold boot)
- Soft reset request (can result in either a warm or cold boot)

A hard reset request occurs whenever the processor Reset line is asserted, then deasserted. A soft reset occurs whenever an assertion occurs on the processor Init line. When a soft reset request occurs, the BIOS determines whether to initiate a warm boot while leaving main memory intact or a cold boot that clears memory. Table 40 summarizes hard and soft reset parameters.

Table 40. Reset requests

Reset Request	Signal Activated	Type	Description
Hard	Reset	Full reboot	The payload is reset via the SYSRESET* input signal of the I/O Controller Hub (ICH). This reset results in a PCI reset, and thus the entire memory region is tested and the lower 8 MB of RAM is initialized. References to a cold reset imply a hard reset.
Soft	Init	Partial reboot	The payload is reset via the RCIN* input signal to the ICH. The ICH then maps this onto the INIT signal to the FW-Hub and CPUs and is asserted for 16 PCI clocks. A PCI reset is not generated in this scenario and the contents of the lower 8MB of RAM is maintained. References to a warm reset imply a soft reset.

5.12.1 Reset Control Sources

Table 41 shows the sources for reset requests and the corresponding type of reset.

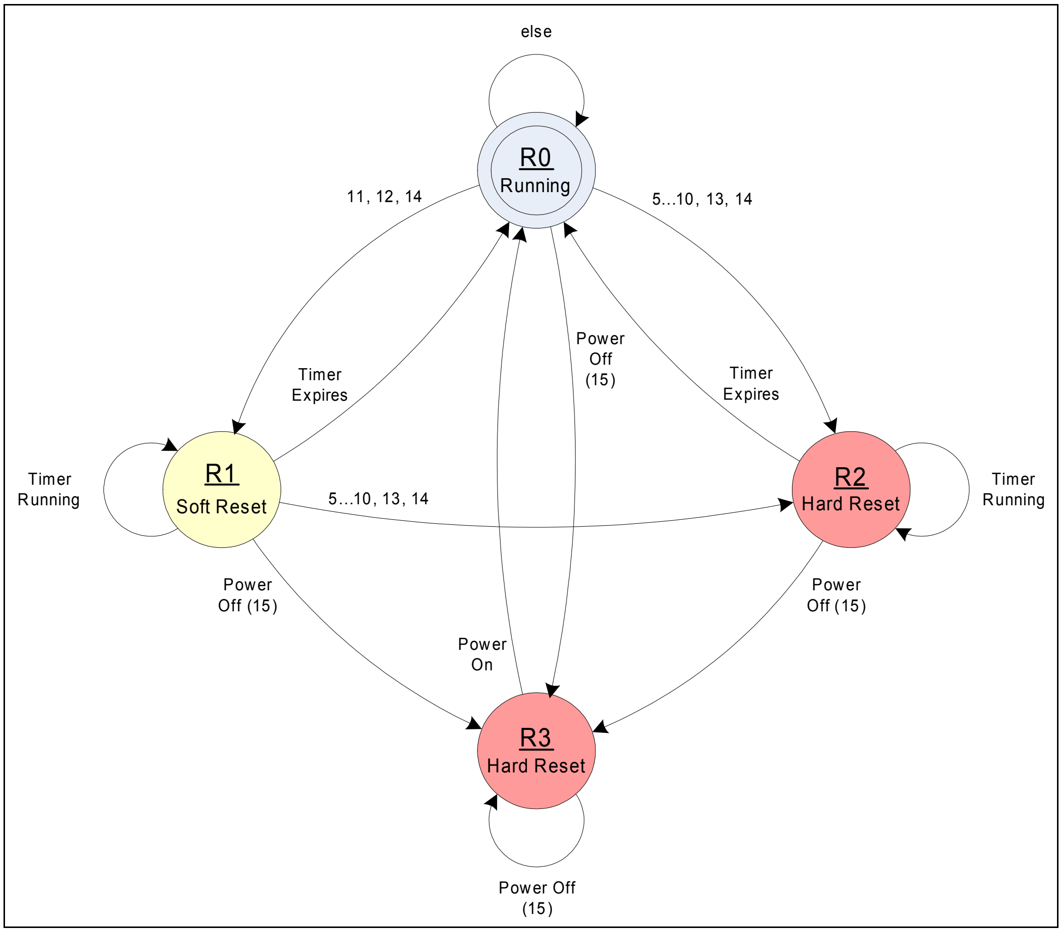
Table 41. System reset sources and actions

#	Reset Source	Payload Reset		IPMC Reset
		Soft	Hard	
1	Standby Power On Reset (SPOR)	--	--	Yes
2	IPMC Hardware Watch Dog Timer	No	No	Yes
3	IPMC Exit Firmware Update Mode	No	No	Yes
4	IPMI BMC Cold Reset command	No	No	Yes
5	IPMI Chassis Control command (hard reset)	No	Yes	No
6	IPMI Watchdog timer expiration (w/ action configured for payload reset)	No	Yes	No
7	IPMI PEF action	No	Yes	No
8	Fault Resilient Booting (FRB3 failure)	No	Yes	No
9	Set Processor State command (upon disabling a processor)	No	Yes	No
10	Reset EFI BIOS Flash Type	No	Yes	No
11	Front Panel Reset button	Yes	No	No
12	OS Warm Boot (restart)	Yes	No	No
13	PICMG Payload Activation (M4)	No	Yes	No
14	PICMG FRU Control command	Yes	Yes	No
15	Payload Power Off	--	Yes	No
--	FPGA image has been changed during online firmware update	No	Yes	No

5.12.2 Payload Reset Diagram

Figure 31 illustrates the reset state of the payload with respect to the possible source of the reset request given in Table 41.

Figure 31. Payload reset state diagram



5.12.3 Front Panel Payload Reset

The Reset button is a momentary contact button on the front panel. Its signal is routed through the front panel connector to the IPMC, which monitors and de-bounces it. The signal must be stable for at least 50 ms before a state change is recognized. The Front Panel reset is mapped to generate a hard reset to the payload.

5.12.4 IPMI Commanded Reset

The IPMI Chassis Control command is supported and can be used to generate a hard reset to the payload.

The Intel OEM Set Processor State command, which is used by the EFI BIOS during POST, also generates a hard reset when used to disable a processor as part of the Fault Resilient Booting (FRB) algorithm.

The PICMG FRU Control command can be used to generate either a hard or soft reset to the payload.

5.12.5 Watchdog Timer Expiration

The Watchdog Timer can be configured to cause a hard reset to the payload upon its expiration. Timeout and action can be configured in EFI BIOS. For more information, see the *Intelligent Platform Management Interface Specification*, Version 2.0.

5.12.6 FRB3 Failure

Simultaneous with resetting the payload, the IPMC starts an internal FRB3 timer. If the EFI BIOS does not start the IPMI Watchdog Timer with the usage set for EFI BIOS/ FRB2, the IPMC resets the system when the internal FRB3 timer expires.

5.13 IPMC Reset Control

Table 42 shows all the sources of IPMC resets and the actions by the system and the IPMC. The payload will not be reset by an IPMC reset except after a combined IPMC boot block and operational code update. In all other cases, the payload is not reset or powered down. The IPMC re-synchronizes itself to the state of the processor and power control signals it finds when it initializes.

Table 42. IPMC reset sources and actions

#	Reset Source	System Reset	IPMC Reset
1	Standby Power On Reset	No (payload not up yet)	Yes
2	IPMC Hardware WDT Expiration	No	Yes
3	IPMC Exit Firmware Update mode	No	Yes
4	IPMI BMC Cold Reset command	No	Yes
5	IPMC boot block update followed by IPMC operational code update	Yes	Yes
6	IPMC operational code update (without updating boot block)	No	Yes
7	IPMC operational code update with firmware that contains new FPGA load	Yes	Yes

5.13.1 Standby Power On Reset

Upon the SBC being inserted into a chassis, the IPMC is reset via the reset signal to the microcontroller. This is referred to as a Standby Power On Reset (SPOR). The firmware detects this situation programmatically, and initializes all of the GPIOs to a known state. At any time during normal operation, if the microcontroller reset input is asserted, the firmware initializes the GPIOs to a known state as if coming up from a SPOR event. This is only expected to occur if the standby power rail falls well below the regulation parameters of the power unit and the power monitoring hardware has asserted the reset to the microcontroller. If the standby power rail has failed, the payload power rails have also failed. Therefore, the payload has already been impacted, and it is therefore safe for the IPMC to initialize all its GPIOs to a known state.

5.13.2 IPMC Hardware Watch Dog Timer Expiration

Upon the IPMC firmware starting up, one of the very first actions performed by the firmware is to initialize the internal hardware watchdog timer to expire after one second. The IPMC firmware must reset the internal timer before the timer expires. If the firmware fails to reset the hardware timer, the IPMC core will be reset, and the firmware will restart from the reset vector. A failure of this nature is an indication that the firmware has failed in particular manor that it was unable to properly reset the hardware timer, and is considered to be faulty. If the hardware watchdog timer does expire, an event is logged into the IPMC System Event Log.

5.13.3 IPMC Exit Firmware Update Mode

The IPMC firmware can be updated using firmware transfer commands through the LPC or IPMB interface. The IPMC automatically enters Firmware Transfer Mode if it detects that the Force Update signal is asserted during initialization or if the operation code checksum fails. Upon exit from Firmware Transfer Mode, the IPMC resets itself.

5.13.4 IPMI BMC Cold Reset Command

The IPMC firmware supports the ability to reset the IPMC via the IPMI Cold Reset command. For more information, see the *Intelligent Platform Management Interface Specification*, Version 2.0.

5.13.5 IPMC Operational Code Update with New FPGA Load

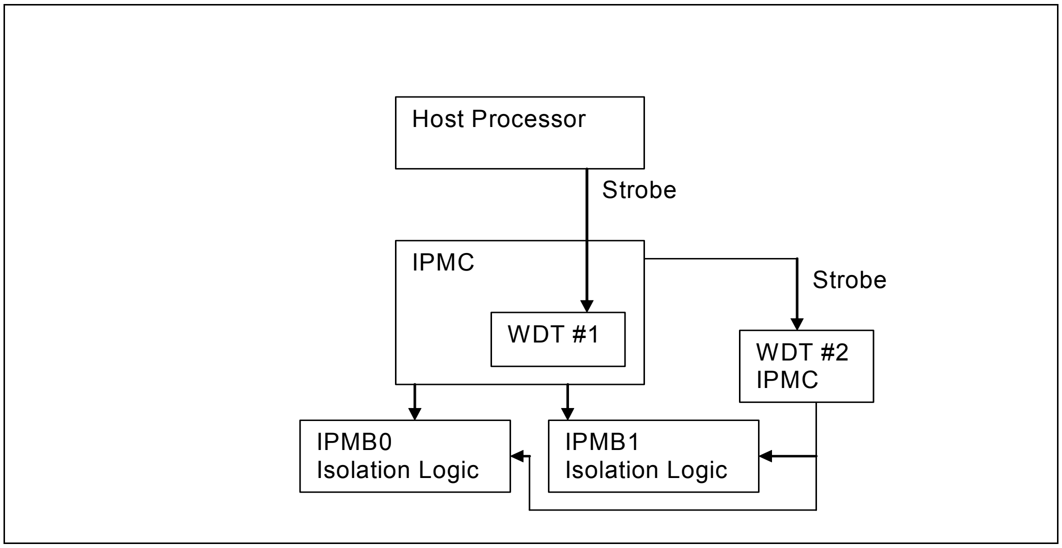
IPMC firmware image file that is used for firmware upgrade always contains image of the FPGA. If the FPGA version running on the SBC differs from the version of the FPGA loaded during firmware update the IPMC will reset payload in order to load the new FPGA image.

Warning: If the IPMC firmware you are planning to load contains new FPGA image please shut down the payload and use upgrade method via Shelf Manager.

5.14 Watchdog Timers (WDTs)

Figure 32 shows the relationship between the two watchdog timers (WDTs) on the MPCBL0050.

Figure 32. Watchdog timers



5.14.1 WDT #1 (IPMI Watchdog Timer)

WDT #1 is an IPMI Watchdog Timer. The host processor uses the IPMI “Set Watchdog Timer” message to configure WDT #1 and then the “Reset Watchdog Timer” message to strobe the timer over the KCS interface to the IPMC.

WDT #1 can be set in EFI BIOS. Choose Boot Menu then OS Load Timeout Timer.

WDT #1 can also be configured to take various actions prior to timing out (for example, SMI_N, NMI, nothing) or after timing out (for example, hard reset, power down, power cycle or do nothing). In addition, an event can be logged in the System Event Log whenever the watchdog timer expires. If WDT #1 expires, the IPMC is not impacted (that is, it is not reset).

WDT#1 operates as per the IPMI version 2.0 Specification as IPMI Watchdog Timer.

5.14.2 WDT #2 (IPMC Hardware Watch Dog Timer)

WDT #2 is a hardware timer internal to the IPMC and must be strobed by the IPMC firmware. When the IPMC firmware starts, one of the very first actions performed by the firmware is to initialize the internal hardware watchdog timer to expire after one second.

The IPMC firmware must reset the internal timer (programmed to do this every 500ms) before the timer expires. If the firmware fails to reset the hardware timer in 1 second, the IPMC core resets, and the firmware restarts from the reset vector. A failure of this nature is an indication that the firmware has failed in particular manner that it was unable to properly reset the hardware timer.

During the IPMC reset IPMB busses are isolated from the backplane. If AMC and/or RTM is installed, also their respective IPMB buses are isolated.

Any reset of the IPMC is completely transparent to the host processor with the possible exception that system management software attempting to communicate with the IPMC might time-out while the reset is in progress. There is no method for the processor to be explicitly notified that the IPMC is reset, but a SEL event will be logged upon next IPMC initialization cycle.

5.15 FRU Payload Control

The MPCBL0050 implements the FRU Control command as specified in the PICMG 3.0 Specification. Through this command, the payload can be reset, rebooted, or have its diagnostics initiated.

The FRU payload can be controlled by a command line via the CMM. The following Intel MPCMM0001/MPCMM0002 CMM commands are supported by the MPCBL0050. Equivalent commands from other shelf managers are available. Refer to the appropriate documentation for third party shelf managers.

Table 43. CMM commands for FRU control options

FRU Control Options	MPCMM0001 / MPCMM0002 command
Cold Reset	cmmset –l bladeN –d frucontrol –v 0 (N is chassis slot number)
Warm Reset	cmmset –l bladeN –d frucontrol –v 1 (N is chassis slot number)
Graceful Reboot	cmmset –l bladeN –d frucontrol –v 2 (N is chassis slot number)
Diagnostic Interrupt	cmmset –l bladeN –d frucontrol –v 3 (N is chassis slot number)

5.15.1 Cold Reset

When this command is initiated, the board performs a hard reset as described in Section 5.12.

5.15.2 Warm Reset

When this command is initiated, the board performs a soft reset as described in Section 5.12.

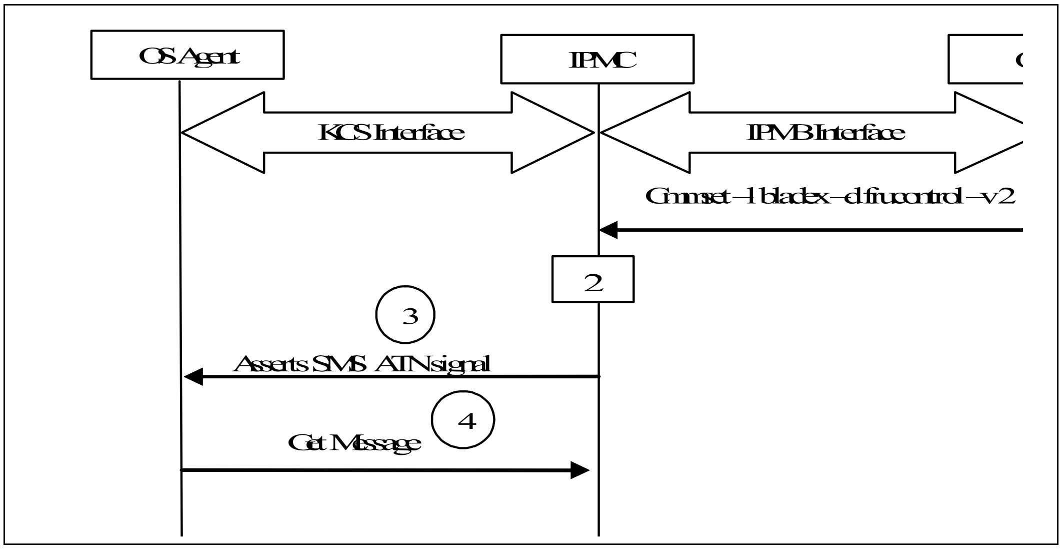
5.15.3 Graceful Reboot

This specific payload control command is implemented using the system interface messaging capability and the SMS_ATN bit of the KCS Status registers.

The Receive Message Queue is used to hold message data for system software until the system software can collect it, while the SMS_ATN bit is used to indicate that the IPMC requires attention from the system software.

The flow diagram shown in Figure 33 will assist users that are developing their system software to interact with this command.

Figure 33. Flow diagram for graceful reboot command



1. The MM sends a frucontrol=2 command to the IPMC, initiating a graceful reboot.
2. When the IPMC receives frucontrol=2, it formats a message into the send message queue and sets the SMS attention flag (SMS_ATN) on the KCS status register.
3. The OS agent polls for SMS_ATN using the Get Message Flags command.
4. The OS agent sends a Get Message command to the IPMC to retrieve the message from the receive message queue. The Get Message command returns the data in Table 44.

Table 44. Returned values from the Get Message command (Sheet 1 of 2)

Byte	Data	Value	Comments
1	Completion Code	00h	
2	Channel	40h	Administrator privilege, Channel 0 (IPMB 0)

Table 44. Returned values from the Get Message command (Sheet 2 of 2)

3	NetFN/rsLUN	C2h	NetFn=30h, Responder LUN=02h (SMS)
4	Header Checksum	3Eh	2's complement of the previous byte (chk1)
5	IPMC Address	(varies)	Board's IPMB address (depends on slot)
6	Sequence/rqLUN	04h	Sequence=01h, Requestor LUN=00h (IPMB)
7	Command	10h	Intel command for shutdown/reboot
8	Data	02h	Reboot action
9	Data Checksum	5F	2's complement of the sum of the previous 4 bytes (chk2)

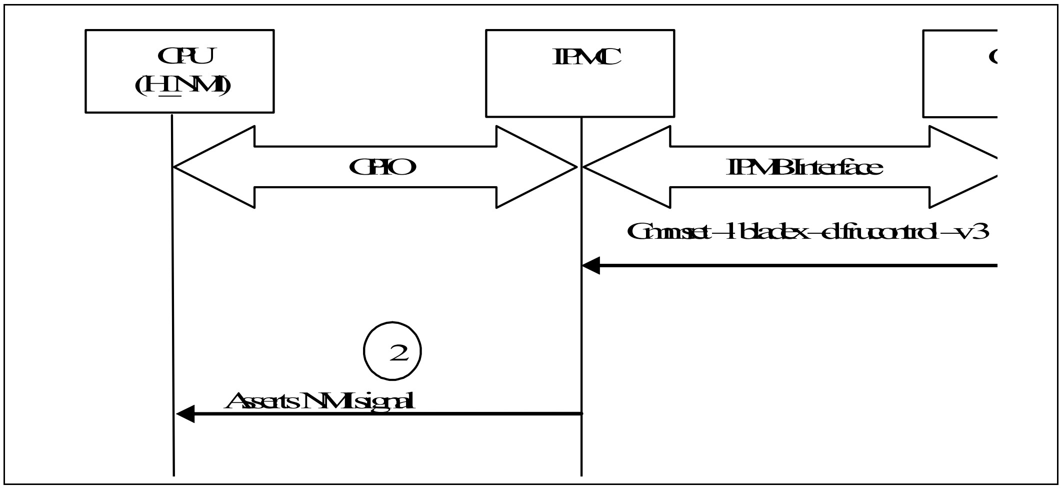
5.15.4 Diagnostic Interrupt

The following command provides the capability for an end user to issue a non-maskable interrupt (NMI) to the payload.

When issued, the NMI signal to the processor is asserted. To fully utilize the support of this command, the user needs to have an NMI handler installed.

The implementation details are shown in Figure 34.

Figure 34. Diagnostic interrupt command implementation



The sequence of actions is as follows:

1. The CMM sends a frucontrol=3 command to the IPMC initiating a diagnostic interrupt.
2. When the IPMC receives frucontrol=3, it asserts the NMI signal to the CPU via the GPIO pins connected to the NMI pin.

5.16 OEM IPMI Commands

This section documents the OEM style IPMI commands implemented and supported on the MPCBL0050. The commands are described in the following tables:

- Table 45, "Intel OEM commands (net function 0x06h)" on page 129
- Table 46, "Intel OEM commands (net function 0x30h)" on page 130
- Table 47, "Intel OEM commands (Net Function 0x32h)" on page 139
- Table 48, "Intel OEM commands (net function 0x3Ah)" on page 140

Table 45. Intel OEM commands (net function 0x06h)

Net Function = Application (0x06), LUN = 00			
Code	**Command**	**Request, Response Data**	**Description**
01h	Get Device ID	Per the IPMI 2.0 specification	Platform-specific response fields: Byte 2 (Device ID) – 0x20 Byte 3 (Device revision) – 0x01 Byte 4 (Firmware Revision 1) • bit 7 – Device Available: 0 – Normal operation 1 – Update mode • bits 6:0 – Major Firmware Revision Byte 5 (Firmware Revision 2) Byte 6 (IPMI version) – 0x02 Byte 7 (Additional Device Support) – 0x3F • bit 7: Chassis device • bit 6: Bridge • bit 5: IPMB Event Generator • bit 4: IPMB Event Receiver • bit 3: FRU Inventory Device • bit 2: SEL Device • bit 1: SDR Repository Device • bit 0: Sensor Device Bytes 8:10 (manufacturer ID) – 343 (57h, 01h, 00h) Byte 11: (product ID 1) – 0x0F (MPCBL0050) Byte 12: (product ID 2) – 0x08 (EID/MCPD Block) Byte 13: (Boot Block Revision 1) Byte 14: (Boot Block Revision 2) Byte 15: (Firmware Build Number) Byte 16: (Reserved) – 0x00

Table 46. Intel OEM commands (net function 0x30h) (Sheet 1 of 9)

Net Function = Intel® General Application (0x30), LUN = 00			
Code	**Command**	**Request, Response Data**	**Description**
01h	Change EFI BIOS boot Flash	**Request**: Byte 1 – EFI BIOS bank number • 00h – Selects main EFI BIOS flash bank and resets the board • 01h – Selects secondary EFI BIOS flash bank and resets the board **Response**: Byte 1 – Completion Code	Switch the boot EFI BIOS flash bank
05h	Set Control State	**Request**: Byte 1 – Control Number: • 00h – FWH Hub (for EFI BIOS bank information) • 01h – FWH0 Write Protect • 02h – FWH1 Write Protect • 03h – FWH0 Top Block Lock • 04h – FWH0 Top Block Lock Byte 2 – Control state: • 00h – Deasserted • 01h – Asserted **Response**: Byte 1 – Completion code	This command sets the state of a control signal. This command overrides the AUTO-state of the control. These control states are persistent through payload reset, but not through IPMC reset.
06h	Get Control State	**Request**: Byte 1 – Control Number • 00h – FWH Hub (for EFI BIOS bank information) • 01h – FWH0 Write Protect • 02h – FWH1 Write Protect • 03h – FWH0 Top Block Lock • 04h – FWH0 Top Block Lock **Response**: Byte 1 – Completion code Byte 2 – Control state • 00h – Deasserted • 01h – Asserted	This command gets the state of a control signal.

Table 46. Intel OEM commands (net function 0x30h) (Sheet 2 of 9)

Net Function = Intel® General Application (0x30), LUN = 00			
Code	**Command**	**Request, Response Data**	**Description**
07h	Get Version Data	**Request**: Byte 1 – None **Response**: Byte 1 – Completion code Byte 2 – Type/Length • 02h – Two bytes Byte 3 – IPMC Boot Block Revision 1 (Binary) Byte 4 – IPMC Boot Block Revision 2 (BCD) Byte 5 – Type/Length • 03h – Three bytes Byte 6 – IPMC firmware (FW) revision 1 (Binary) Byte 7 – IPMC firmware revision 2 (BCD) Byte 8 – IPMC firmware patch level (Binary) Byte 9 – Type/Length • 01h – One byte Byte 10 – FPGA version/revision • bits 7:4 – Version bits • bits 3:0 – Revision bits Byte 11 – Type/Length • 01h – One byte Byte 12 – Board version • bits 7:4 – Version bits • bits 3:0 – Revision bits Byte 13 – Type/Length • 03h – Three bytes Byte 14 – Staged IPMC FW revision 1 (Binary) Byte 15 – Staged IPMC FW revision 2 (BCD) Byte 16 – Staged IPMC FW build (Binary) Byte 17 – Type/Length • 03h – Three bytes Byte 18 – Rollback IPMC FW revision 1 (binary) Byte 19 – Rollback IPMC FW revision 2 (BCD) Byte 20 – Rollback IPMC FW build (Binary)	This command returns the boot and operational, staged, rollback firmware versions and the FPGA version information in the format required by the FRU OEM MRA definition for this platform. Type/Length Format: Bits Value Meaning 7:6 00 Binary/unspecified 01 BCD 10 6 bit ASCII, packed 11 8 bit ASCII
0Ah-0Fh	Reserved	N/A	
18h-20h	Reserved	N/A	

Table 46. Intel OEM commands (net function 0x30h) (Sheet 3 of 9)

Net Function = Intel® General Application (0x30), LUN = 00			
Code	**Command**	**Request, Response Data**	**Description**
21h	Get DIMM State	**Request**: Byte 1 – DIMM Group ID **Response**: Byte 1 – Completion code Byte 2 – DIMM Group Presence • bits 7:1 – Reserved • bit 0 – Presence (1 = group present) Byte 3 – Bitmap of DIMM slot existence Byte 4 – Bitmap of DIMM failure state Byte 5 – Bitmap of DIMM disabled state Byte 6 – Reserved Byte 7 – Bitmap of DIMM presence state	This command allows the *presence, disabled*, *failure,* and *spared* state of a set of DIMMs to be obtained without having to know the IPMC IPMI sensor numbers of the associated DIMM sensors. The state is returned as bitmaps. A bit being set in a bitmap indicates assertion of the state. The DIMM that a bit offset refers to is platform dependent. The DIMM Group Selector value is platform dependent. Platforms with 8 or fewer DIMMs should always use 0 as the Group Selector value. On platforms with memory boards, the Group Selector maps to a board number which is 1 based.
22h	Set DIMM State	**Request**: Byte 1 – DIMM Group Selector • bits 7:1 – Group Id • bit 0: – Presence (1 = group present) Byte 2 – Bitmap of DIMM slot existence Byte 3 – Bitmap of DIMM failure state Byte 4 – Bitmap of DIMM disabled state Byte 5 – Reserved Byte 6 – Bitmap of DIMM presence state **Response**: Byte 1 – Completion code	This command allows the failure state of a set of DIMMs to be set. Executing this command causes the associated sensor offsets for affected DIMM sensors. The DIMM Group Id is defined the same as for the Get DIMM State command. The IPMC will accept a variable number of request bytes: as few as 1 bytes of this command (Group selector) up to the fully defined set. Note that the Group/Board Presence flag is ignored on platforms that don't implement removable groups (memory boards). Example of the Slot Existence bitmap: Byte 2 = 0Fh and byte 6 = 0Dh means DIMM 1,3 and 4 are present, the DIMM 2 slot is empty. DIMM slots 5,6,7 and 8 do not exist on this platform.
23h	ReArm DIMMs	**Request**: N/A **Response**: Byte 1 – Completion code	This command causes the DIMM failure and disabled state of all DIMM sensors to be reset.
25h	Reserved	**N/A**	

Table 46. Intel OEM commands (net function 0x30h) (Sheet 4 of 9)

Net Function = Intel® General Application (0x30), LUN = 00			
Code	**Command**	**Request, Response Data**	**Description**
28h	Set Processor State	**Request**: Byte 1 – Processor ID Valid values are 0:N-1, where N is the number of processors supported by the platform. Byte 2,3 – Processor state to set This is a bit mask identifying the sensor offsets to set in the associated processor status sensor maintained by the IPMC. The offsets are as defined in the IPMI 1.5 specification. The following offsets are supported: • 0 – IERR • 1 – Thermal Trip • 2 – FRB1/BIST Failure • 3 – FRB2/POST Hang Failure • 4 – FRB3/Processor Startup Failure • 5 – Configuration Error • 6 – SMEFI Uncorrectable CPU-complex error • 8 – Processor Disabled Byte 4 – Action This byte specifies the action to take after setting the processor status sensor state as requested. It is a bit-mask and multiple actions may be set. • bits 7:1 – Reserved • bit 0 – reset system **Response**: Byte 1 – Completion code	This command allows processor fault state to be asserted and an action to be taken afterwards. Asserting some fault states may cause the IPMC to generate SEL events (depending on the SDR configuration). Processor disabling will not take effect until the next reset.
29h	Get Processor State	**Request**: Byte 1 – Processor ID Valid values are 0:N-1, where N is the number of processors supported by the platform. **Response**: Byte 1 – Completion code Byte 2,3 – Processor state These bytes contain the associated Processor Status sensor's 2-byte event assertion status as defined in the IPMI 1.5 specification.	This command returns the current Processor Status sensor event assertion status for the requested processor. This command allows the caller to get processor state without having to know the IPMI sensor numbers of the Processor Status sensors.
2Ah	ReArm Processors	**Request**: N/A **Response**: Byte 1 – Completion code	This command clears all error and disabled state for all processors. Processor/terminator presence is not affected. Disabled processors are not actually run until the next system reset.
2Bh	Disable FRB3 Action	**Request**: N/A **Response**: Byte 1 – Completion code	This command disables resets associated with the watchdog timer expiring with an FRB3 reason.

Table 46. Intel OEM commands (net function 0x30h) (Sheet 5 of 9)

Net Function = Intel® General Application (0x30), LUN = 00			
Code	**Command**	**Request, Response Data**	**Description**
30h	Get Serial Port Capture Data	**Request:** Byte 1 - bits as following: [7:1] - reserved [0] - clear buffer after read Byte 2 - Data offset - Least Significant Byte Byte 3 - Data offset - Most Significant Byte **Response:** Byte 1 - completion code Byte 2 - size of data (max 16 for this implementation) = n Byte 3 to Byte (3+n) = captured characters	This command allows reading chunk of characters captured from the payload serial port console. In case of system failure it is possible to retrieve last full screen of console output. Use cases for Bytes 2-3: "To get captured characters from given offset use rule: Byte2=offset / 0FFh, Byte3=offset % 0FFh "To get captured characters from offset "0" use: Byte2=00h, Byte3=00h "To get characters from offset "5" use: Byte2=05h, Byte3=00h "To get characters from offset "255" use: Byte2=0FFh, Byte3=00h "To get characters from offset "256" use: Byte2=00h, Byte3=01h "To get characters from offset "257" use: Byte2=01h, Byte3=01h
31h	Get Serial Port Capture Configuratio n	**Request**: None **Response**: Byte 1 - completion code Byte 2 - Serial Port Buffering configuration, bits as following: [7:2] - reserved [1] - Buffering enable (1 = enabled, 0 = disabled) [0] - Buffering mode (1 = filtering mode, 0 = raw mode)	This command allows to read current status of serial port buffer configuration. Serial Port Buffering feature is designed to record last screen of payload serial port console. There are two modes of operation: "Raw mode - in this mode all characters sent by payload console are captured without any filtering. Captured data will contain also steering sequences like frames and ANSI control codes. "Filtering mode - characters will be filtered to remove ANSI control codes and frames. Captured data will contain only ASCII characters Byte 2 of the response contains provides information if the feature is enabled and which mode is chosen,

Table 46. Intel OEM commands (net function 0x30h) (Sheet 6 of 9)

Net Function = Intel® General Application (0x30), LUN = 00			
Code	**Command**	**Request, Response Data**	**Description**
32h	Set Serial Port Capture Configuration	**Request:** Byte 1 - bits as following: [7] - clear the buffer (1=clear, 0 = do not clear) [6:0] - reserved, shall be always 0 Byte 2 - bits as following: [7:6] - reserved [5] - set to 1 to enable or disable buffering [4] - set to 1 to change buffering mode [3:2] - reserved [1] - Buffering enable (1 = enable, 0 = disable) [0] - Buffering mode (1 = filtering mode, 0 = raw mode) Response: Byte 1 - completion code	This command allows setting the serial console buffer configuration. Bits [5:4] indicates which option will be changed. If user wants to enable or disable Serial Port Buffering ' bit 5 shall be set to "1". To change filtering mode, bit 4 shall be set to "1". Use cases for byte 2: "To enable buffering: set bit[5]=1, bit[1]=1. "To disable buffering: set bit[5]=1, bit[1]=0. "To enable filtering mode: set bit[4]=1, bit[0]=1. "To enable raw mode: set bit[4]=1, bit[0]=0. "To enable buffering and filtering: bit[5]=1, bit[4]=1, bit[1]=1, bit[0]=1. "To disable buffering and filtering: bit[5]=1, bit[4]=1, bit[1]=0, bit[0]=0. "
41h	Set System GUID	**Request**: Bytes 1-16 – System GUID **Response**: Byte 1 – Completion code	This command sets the System GUID retrieved per the format of the IPMI 1.5 *Get System GUID* command. This value is stored persistently.
47h	Reserved	N/A	
48h	Internal Platform Event Message	**Request**: Byte 1 – Generator ID Byte 2 – Event Message Revision (04h) Byte 3 – Sensor Type Byte 4 – Sensor Number Byte 5 – Event Dir / Event Type Byte 6 – Event Data 1 Byte 7 – Event Data 2 Byte 8 – Event Data 3 **Response**: Byte 1 – Completion code Byte 2 – Reserved Byte 3:4 – Record ID Byte 5:8 – Timestamp	This command provides a mechanism for the EFI BIOS to log event messages and retrieve the record ID and timestamp of the event. This command can be used to correlate EFI BIOS events with SEL events. The format of the request data is the same as for the Platform Event Message command (Sensor/Event net function, command number 02). See the IPMI 1.5 specification for details on the definition of the request data fields.
50h	Set SM Signal	**Request**: Byte 1 – Signal type: • 0Eh – Health LED • 0Fh – Out Of Service LED • 10h – Hot-Swap LED Byte 2 – Signal instance, zero based Byte 3 – Action: • 0 – Force De-asserted • 1 – Force Asserted • 2 – Revert Byte 4 – Optional value used by multi-value signals **Response**: Byte 1 – Completion code	This command allows the real-time state of certain output signals (for example, LEDs) to be set without losing the IPMC internal state associated with the signals. The command also allows the output signals to revert to their normal behavior. Certain signals take analog values or complex states. The fourth byte of the request is supported for these kinds of signals, for example, Fan Speed Control.

Table 46. Intel OEM commands (net function 0x30h) (Sheet 7 of 9)

Net Function = Intel® General Application (0x30), LUN = 00			
Code	**Command**	**Request, Response Data**	**Description**
51h	Get SM Signal	**Request**: Byte 1 – Signal type: • 13h – Reset Button • 14h – Hot-Swap Handle Switch Byte 2 – Signal instance, zero based Byte 3 – Action: • 0 – Sample • 1 – Ignore • 2 – Revert **Response**: Byte 1 – Completion code Byte 2 – Signal value: • 0 – De-asserted • 1 – Asserted	This command allows the real-time state of certain input signals to be polled without changes in the signals to be acted on by the IPMC. The command also allows the input signals to revert to their normal behavior. Not all signals are supported by all platforms.
52h	Get Self Test History	**Request**: Byte 1 – Action: • 0 – Get First • 1 – Get Next **Response**: Byte 1 – Completion code Byte 2 – First byte of test result • FFh – No more results Byte 3 – Second byte of test result	This command retrieves stored self-test failures. Normal use is to first call with action of "Get First" then use "Get Next" for subsequent calls. If byte 2 of the response is ever 0xFF, all the results have been retrieved.
60h	Get IPMI commands History	**Request:** Byte 1 - Unique Record ID. LS Byte. Byte 2 - Unique Record ID. MS Byte. Byte 3 - Header/Data flag. [7:1] - reserved [0] - Header/Data 0h - header 1h - data **Response (for header request):** Byte 1 - completion code Byte 2 - Unique Record ID for next record. LS Byte. Byte 3 - Unique Record ID for next record. MS Byte. Byte 4 - Channel of the captured IPMI message Byte 5 - Len of the captured IPMI message Byte 6 - NetFn of the captured IPMI message Byte 7 - Cmd of the captured IPMI message (Byte 8) - Data of the captured IPMI message (optional) **Response (for data request):** Byte 1 - completion code (Byte 2-25) - Data of the captured IPMI message (optional)	IPMC on MPCBL0050 collects the list of all IPMI commands that IPMC receives or sends (IPMI history). THe list is maximum 512 and is overwritten on round-robin basis *Get IPMI commands History* command allows to retrieve the commands from the list. The get request has two modes: -retrieve command header -retrieve command data (max 24 bytes) Byte 1 and 2 of the request contain index on the table of recorded IPMI commands. Each captured IPMI command can be fully retrieved using two 'IPMI commands history get' commands. An application that wishes to retrieve the full set of IPMI Commands History Records must first issue the Get Commands History starting with 0000h as the Record ID to get the first record. The Next Record ID is extracted from the response and this is then used as the Record ID in a Get IPMI Commands History request to get the next record. This is repeated until the 'Last Record ID' value (FFFFh) is returned in the 'Next Record ID' field of the response. Note: IPMI commands history feature doesn't capture own 'IPMI commands history get' commands.

Table 46. Intel OEM commands (net function 0x30h) (Sheet 8 of 9)

Net Function = Intel® General Application (0x30), LUN = 00			
Code	**Command**	**Request, Response Data**	**Description**
61h	IPMI commands history clear	Request: <empty> Response: Byte 1- completion code	This command clears the storage used to collect information about IPMI commands sent and received by SBC.
70h	Graceful OS Shutdown	**Request**: Byte 1 • bits 7:3 – Reserved • bits 2:0 – Shutdown Operation: 00h – No change (used to read shutdown command status) 01h – Power off using the OS Agent 02h – Reset using the OS Agent 03h-07h – Reserved **Response**: Byte 1 – Completion code Byte 2 – Only for command status • bits 7:2 – Reserved • bits 1:0 – Shutdown command status 00h – Operation Successful 01h – Operation Failed (no OS agent) 02h – Operation in progress 03h – Reserved	This command performs graceful shutdown using OS agent.
82h	Get ACPI Configuration Mode	**Request**: N/A **Response**: Byte 1 – Completion code Byte 2 – ACPI Configuration state • bits 7:1 – Reserved • bit 0 – ACPI Mode: 0 = IPMC is in Legacy mode 1 = IPMC is in ACPI mode	
83h	Set ACPI Configuration Mode	**Request**: Byte 1 – ACPI Configuration state Byte 2 – ACPI Configuration state • bits 7:1 – Reserved • bit 0 – ACPI Mode 0 = IPMC is in Legacy mode 1 = IPMC is in ACPI mode Byte 2 – ACPI Configuration Mask • bits 7:1 – Reserved • bit 0 – ACPI mode mask 1 = set ACPI mode **Response**: Byte 1 – Completion code	
86h thru 8Ch	Reserved		Reserved for internal IPMC use
A0h thru A8h	Reserved		Reserved for internal IPMC use

Table 46. Intel OEM commands (net function 0x30h) (Sheet 9 of 9)

Net Function = Intel® General Application (0x30), LUN = 00			
Code	**Command**	**Request, Response Data**	**Description**
E6h	Get NMI/ INIT Source	**Request**: N/A **Response**: Byte 1 – Completion code Byte 2 – NMI/INIT Source 1: • bits 7:6 – Reserved • bit 5 – Processor Thermal Trip • bit 4 – Processor IERR • bit 3 – Chassis Control Command • bit 2 – Event (PEF) • bit 1 – Watchdog NMI/Diagnostic Interrupt • bit 0 – Diagnostic Interrupt (FP NMI) Button Byte 3 – NMI/INIT Source 2: • bits 7:4 – Reserved • bit 3 – Chipset NMI • bit 2 – South Bridge NMI • bit 1 – PCI SERR/PERR • bit 0 – Multi-bit Memory Error	This command returns the IPMC's understanding of the source of the latest NMI/INIT assertion. The source information is a composite of IPMC detected sources (Source 1) and externally detected sources (Source2). Although multi-bit memory error is in the external source byte, on some platforms, this may be detected by the IPMC. The source 1 and 2 values are cleared when read and when the system is reset or powered off.
EDh	Set NMI/ INIT Source	**Request**: Byte 1 – • bits 7:4 – Reserved • bit 3 – Chipset NMI • bit 2 – South Bridge NMI • bit 1 – PCI SERR/PERR • bit 0 – Multi-bit Memory Error **Response**: Byte 1 – Completion code	This command merges the given values in with any NMI/INIT sources detected by the IPMC. The values given here will be read by the next *Get NMI/INIT Source* command. This command also causes the IPMC to generate an NMI/ INIT pulse for a supported source.
F7h	NMI/INIT Enable / Disable	**Request**: Byte 1 – NMI/INIT enable state • 0 = Disable IPMC NMI/INIT generation • 1 = Enable IPMC NMI/INIT generation **Response**: Byte 1 – Completion code	This command is the master control for the IPMC NMI/INIT generation. The default state for NMI/INIT generation is enabled. The state set by this command is volatile, that is, it is not saved across IPMC resets.
FAh	Get Latest Port80 Codes	**Request**: None **Response**: Byte 1 – Completion Code Byte 2 – Pre-Reset Port80 (byte n-4) Byte 3 – Pre-Reset Port80 (byte n-3) Byte 4 – Pre-Reset Port80 (byte n-2) Byte 5 – Pre-Reset Port80 (byte n-1) Byte 6 – Pre-Reset Port80 (last byte n) Byte 7 – Post-Reset Port80 (byte n-4) Byte 8 – Post-Reset Port80 (byte n-3) Byte 9 – Post-Reset Port80 (byte n-2) Byte 10 – Post-Reset Port80 (byte n-1) Byte 11 – Post-Reset Port80 (last byte n)	Returns two port80 signatures from before the last reset, and afterwards. The first signature contains the last 5 port80 bytes prior to the last payload reset event. The second signature contains the most current port80 bytes after the payload was reset.

Table 47. Intel OEM commands (Net Function 0x32h)

Net Function = Intel® Platform Specific (0x32), LUN = 00			
Code	**Command**	**Request, Response Data**	**Description**
01h	Get HW Info	**Request:** N/A **Response**: Byte 1 – Board version • bits 7:4 – Fab version • bits 3:2 – Fab revision • bits 1:0 – Reserved Byte 2 – FPGA version • bits 7:4 – Version • bits 3:0 – Revision	Provides the SBC board and FPGA versions from the FPGA itself.
02h	Get Power Unit Status	**Request:** N/A **Response:** Byte 1 – Power State • 00h – Power is ON (ACPI State S0) • 05h – Power is OFF (ACPI State S5) • 20h – Legacy ON • 21h – Legacy OFF • [xx] – All others reserved Byte 2 – Power Status Bits • bit 0 – Power Cycle • bit 1 – Control Fault • bits 7:2 – Reserved Byte 3 – Power Fault 1 Bits • bit 0 – 1_2V_E_PHY power fault • bit 1 – 1_8V_E_PHY Power Fault • bit 2 – 1_5V_E_ESB Power Fault • bit 3 – 12V_E Power Fault • bit 4 – 5V Power Fault • bit 5 – 3_3V Power Fault • bit 6 – 1_2V_SAS Power Fault • bit 7 – 1_5V_ESB Power Fault Byte 4 – Power Fault 2 Bits • bit 0 – 1_5V_DDR Power Fault • bit 1 – 1_8V_DDR Power Fault • bit 2 – 0.9VTT Power Fault • bit 3 – 1_2VTT Power Fault • bit 4 – CPU0_VRM Power fault • bit 5 – CPU1_VRM Power fault • bit 6 – 0 • bit 7 – 0	Provides the status of the payload power and corresponding power unit faults from the FPGA.
55h	Reserved		
57h	Reserved		

Table 48. Intel OEM commands (net function 0x3Ah)

Net Function = Application (0x3A), LUN = 00			
Code	**Command**	**Request, Response Data**	**Description**
26h	LAN Mux Settings	To set LAN configuration: Request: Byte 1 -80h Byte 2 - Mux Config, as defined in description Response: Byte 1 - completion code To get current LAN ports configuration: Request: Byte 1 - 00h Response: Byte 1 - Completion code Byte 2 - Mux Config, as defined in Description.	This commands allows changing the GbE ports direction. Refer to Figure 5 on page 25 for possible configurations. Mux config byte description: 14h - all ports to ATCA backplane 2Bh - all ports to RTM 1Bh - ports C/D to backplane, ports E/F to RTM 18h- ports C/D to backplane, ports E/F to Front panel. 24h -AMC GbE ports to backplane Channel1/2 port 0, ports C/D to RTM, ports E/F to backplane Channel1/2 port 1. NOTE: A board reboot is required for Ethernet port change to take effect.

6.0 EFI BIOS Features

6.1 Introduction

The Intel NetStructure® MPCBL0050 Single Board Computer uses a Aptio* EFI (Extensible Firmware Interface) developed by Intel and AMI*. The EFI BIOS resides in the Flash ROM. It provides hardware-specific initialization algorithms and standard PC-compatible basic input / output (I/O) services, and standard Intel® Server Board features. The Flash ROM also contains firmware for certain embedded devices. These images are supplied by the device manufacturers and are not specified in this document.

The EFI BIOS implementation is based on the Intel® Platform Innovation Framework for EFI architecture and is compliant with all Intel Platform Innovation Framework for EFI architecture specifications specified in the Extensible Firmware Interface Reference Specification, Version 1.1.

The EFI displays a message during POST identifying the type of EFI and a revision code.

6.2 EFI BIOS Flash Memory Organization

The MPCBL0050 contains two firmware hub (FWH) devices (see Figure 1, "MPCBL0050 SBC block diagram" on page 18). The first one is the primary FWH, which holds the EFI code that executes during POST. The second is the backup FWH, which recovers the system when the primary FWH is corrupted.

6.3 Redundant EFI BIOS Functionality

The MPCBL0050 hardware provides two flash devices for EFI BIOS where redundant copies are stored. Logic to select the active EFI BIOS device is connected to the IPMC. IPMC firmware selects the EFI BIOS device to boot from.

By default, the firmware selects EFI BIOS device FWH0. The EFI BIOS executes code from this flash and performs checksum validation of its operational code. This checksum occurs in the boot block of the EFI BIOS. If the boot block detects a checksum failure in the remainder of the EFI BIOS, it notifies the IPMC of the failure. In the event of failure, the IPMC firmware:

1. Asserts the RESET pin on the processor.
2. Switches the flash device.
3. Deasserts the RESET pin on the processor, allowing EFI BIOS to execute off the second flash device.
4. Logs a SEL event

6.4 Language Support

English is the only supported language.

6.5 Recovering EFI BIOS Data

Some types of failure can destroy the EFI BIOS. For example, the data can be lost if a power outage occurs while the EFI BIOS is being updated in flash memory. The EFI BIOS can be recovered from the backup EFI BIOS. Recovery mode activates when EFI BIOS checksum fails and EFI BIOS notifies the IPMC to failover. If EFI BIOS hangs before code calculating CRC32 is executed, expiration of FRB timer initiate booting from backup FWH1.

It is also possible to manually force EFI BIOS loading from redundant FWH1 by changing DIP switch. Refer to Section 3.4.

6.6 Complementary Metal-Oxide Semiconductor (CMOS) RAM

CMOS RAM is nonvolatile storage that stores data needed by the EFI BIOS. The data consists of certain onboard configurable settings. The settings in the EFI BIOS Setup menu are often called CMOS settings.

Note: The CMOS settings are not affected by the full discharge of the hold up capacitor as all settings are saved to non-volatile memory on the board. Only date and time values will be reset if the capacitor fully discharges.

6.7 Improving Booting Time

Loading option ROMs for bootable devices is one of the most time consuming procedures during EFI BIOS boot up. By default option ROMs for SAS storage devices and for PXE network boot will be loaded. In order to improve boot time of the system, disable Option - ROM(s) of the devices you do not need to boot. The SAS option ROM can be disabled in “EFI BIOS setup->Advanced->Mass Storage Controller”, while PXE option ROM can be disabled in “EFI BIOS setup->Advanced->PCI Configuration”. Please refer to Section 7.0 for details on EFI BIOS setup options.

6.8 EFI BIOS Security Features

The EFI BIOS uses passwords to prevent unauthorized tampering with the board setup. Both user and administrator passwords are supported by the EFI BIOS. An Administrator password must be entered in order to set the user password. The maximum length of the password is seven characters. The password cannot have characters other than alphanumeric (a-z, A-Z, 0-9). It is not case sensitive.

Once set, a password can be cleared by changing it to a null string. Entering the user password will allow the user to modify the time, date, and user password. Other setup fields can be modified only if the administrator password is entered. If only one password is set, this password is required to enter EFI BIOS Setup.

The administrator has control over all fields in EFI BIOS Setup, including the ability to clear the user password.

If the user or administrator enters an incorrect password three times in a row during the boot sequence, the system is placed into a halt state. A system reset is required to exit out of the halt state. This feature makes it difficult to break the password by guessing at it.

6.8.1 Password Clear DIP Switch

If the user and/or administrator password is lost or forgotten, both passwords may be cleared by moving the password clear DIP switch into the clear position. The EFI BIOS determines if the password clear jumper is in the clear position during EFI BIOS POST and clears any passwords if required. The password clear DIP switch must be restored to its original position before a new password will stay set.

6.9 Remote Access Configuration

Remote access using serial console redirection allows users to monitor the MPCBL0050 boot process and run the MPCBL0050 EFI BIOS Setup from a remote serial terminal. Connection is made directly through a serial port.

The console redirection feature is useful in cases where it is necessary to communicate with a processor board in an embedded application without video support.

Note: The default settings used for console redirection to the serial port are 115200, n, 8, 1, and no flow control.

Table 49 shows the escape code sequences that may be useful for things like EFI BIOS Setup if function keys cannot be directly sent from a terminal application.

Table 49. Function key escape code equivalents

Key	Escape Sequence	Notes
F2	ESC 2	Enter EFI BIOS setup utility.
F4	ESC 4	Enter EFI BIOS setup utility.
F10	ESC 0	To save and exit EFI BIOS Setup

6.10 Boot Device Priority

The boot device priority is stored in non volatile RAM (NVRAM), is static across reboots will not change except for in the following conditions:

- The user manually presses F4 during EFI BIOS to enter EFI BIOS Setup Menu, changes the Boot Device Priority and saves EFI BIOS settings.
- After a user performs a EFI BIOS update to the board and the user chooses to load the default EFI BIOS setup. This will erase the current boot order in NVRAM and EFI BIOS will automatically determine a new boot order upon next reboot of the board.
- If there is an error and no valid boot order stored in NVRAM.

As soon as the board boots one time, the boot order will be fixed until changed by the user. A manual save of boot order by user is not needed to fix the boot order.

Above mentioned NVRAM refers to a small portion of the FWH (Firmware Hub). When the NVRAM is empty or does not contain valid boot order information, the EFI BIOS automatically determines a boot order upon next reboot. If a USB device is connected to the board when there is no boot order saved, the USB device will automatically be placed at the top of the boot order.

Once the boot order is static, if a hard drive or any other boot device is added (such as USB boot device), it will be added at the bottom of the boot order list.

If a boot device needs to be added as a permanent boot device, the user should re-order the boot devices in EFI BIOS Boot Device Priority setup menu.

6.11 Progressive Boot Support

Progressive Boot is a feature added into the system to increase the availability of the system in the event of the primary boot device being corrupted or unbootable.

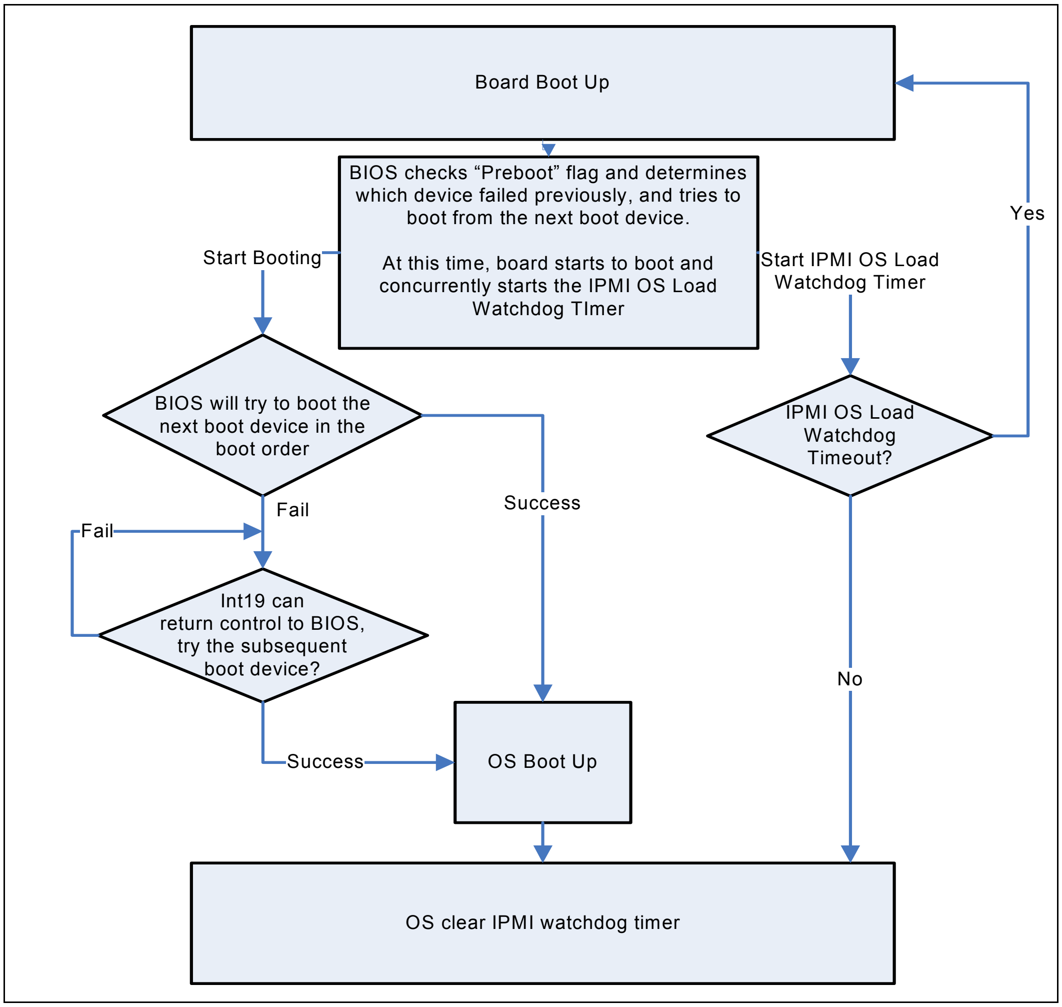
Taking into account that a user has configured the second/third boot device as a fail safe storage to store a recovery OS or a redundant OS image, the EFI BIOS attempts to boot from the subsequent boot device as configured under the “Boot Device Priority Submenu”. See Section 7.6, “Boot Options Menu” on page 154 for details.

6.11.1 Progressive Boot Mechanism

Prior to OS boot process, the EFI BIOS acquires the IPMI watchdog timer to determine whether the IPMI OS load flag has been expired. If the OS load flag has not been expired, then the IPMI watchdog timer for OS load will be turned on, and the EFI BIOS tries to boot the first boot device in the boot order. If an OS encounters a boot failure and returns to EFI BIOS, then the EFI BIOS can try the next boot device in the boot order. But if an OS encounters a boot failure and does not return to EFI BIOS, the IPMI watchdog timer times out and resets the board. In the next OS boot process, the EFI BIOS checks the IPMI watchdog timer to see whether the OS load flag has expired. If the OS load flag has expired, then the IPMI watchdog timer for OS load is turned on, and the EFI BIOS tries to boot the next boot device from the previous boot device that caused the IPMI watchdog timer to timeout.

Figure 35 shows the boot sequence.

Figure 35. Boot sequence


Board Boot Up
BIOS checks “Preboot” flag and determines which device failed previously, and tries to boot from the next boot device.
At this time, board starts to boot and concurrently starts the IPMI OS Load Watchdog TImer
Start Booting
Start IPMI OS Load Watchdog Timer
Yes
BIOS will try to boot the next boot device in the boot order
IPMI OS Load Watchdog Timeout?
Fail
Success
Fail
Int19 can return control to BIOS, try the subsequent boot device?
No
Success
OS Boot Up
OS clear IPMI watchdog timer

6.12 Pre-Defined Resources for AdvancedMC Modules

The AdvancedMC is hot swappable. In order to support AdvancedMC hot add, the MPCBL0050 SBC has to reserve resources that are not being used by a device when the SBC is booted. It is not possible for the SBC to know exactly what resources an AdvancedMC module that is hot added may require before it is inserted. If the resources reserved by the SBC for the AdvancedMC module are adequate, then the AdvancedMC hot add will be successful. If an AdvancedMC module needs more resources (memory, I/O space, etc.) than the SBC has reserved, then the AdvancedMC module will not work properly until the SBC is rebooted with the AdvancedMC module installed. If the hot added AdvancedMC module does not use all of the resources that are reserved by the SBC, then the reserved resources are left unused and cannot be used by another device because they are still reserved.

On the MPCBL0050 board, the EFI BIOS reserves the following resources for the AdvancedMC hot plug slot:

- Bus = 5
- I/O space = 8 KB

- Prefetchable memory = 32MB
- Non-prefetchable memory = 32MB

If an AdvancedMC module is installed before power up and recognized by the EFI BIOS, then the resources required for that AdvancedMC module is determined and that information is available to the operating system upon boot.

Warning: AdvancedMC modules shall only be installed before board payload powers on or after the operating system has fully loaded. If an AdvancedMC module is hot added after the initial EFI BIOS device discovery, but before the operating system is fully loaded, then the AdvancedMC module may not work properly. To correct this, a board payload reset or payload power cycle is required.

6.13 USB Support

6.13.1 Native USB Support

During the power-on self-test (POST), the EFI BIOS initializes and configures the USB subsystem in accordance with chapter 14 of the Extensible Firmware Interface Reference Specification, Version 1.1. The EFI BIOS is capable of initializing and using the following types of USB devices:

- USB Specification-compliant keyboards
- USB Specification-compliant mice
- USB Specification-compliant storage devices that utilize bulk-only transport mechanism

USB devices are scanned to determine if they are required for booting.

The EFI BIOS supports USB 1.1-compliant devices and host controllers. The EFI BIOS configures the USB 2.0-compliant host controller and USB 2.0-compliant devices in USB 1.1 mode because all USB 2.0 devices are required to support USB 1.1 mode. Although USB 1.1 mode is slower than USB 2.0 mode, the difference in speed is not significant during the pre-boot phase. The operating system can reconfigure the USB devices in USB 2.0 mode as required. The EFI BIOS configures the USB 2.0 host controller (EHCI) so the operating system can use it.

During the pre-boot phase, the EFI BIOS automatically supports the hot addition and hot removal of USB devices. For example, if a USB device is hot plugged, the EFI BIOS detects the device insertion, initializes the device, and makes it available to the user. Only onboard USB controllers are initialized by EFI BIOS. This does not prevent the operating system from supporting any available USB controllers, including add-in cards.

6.13.2 Legacy USB Support

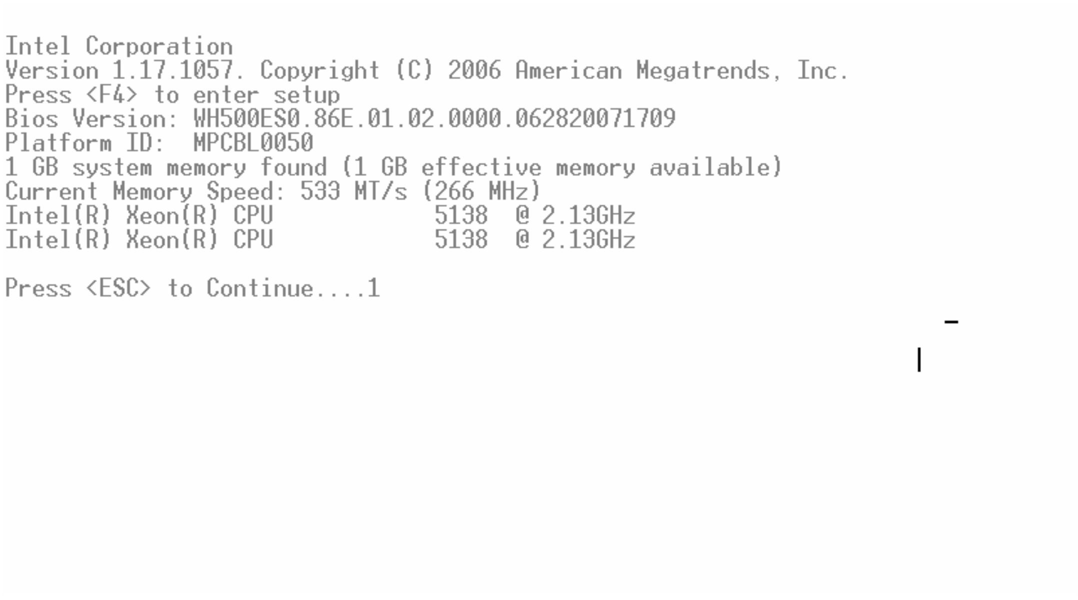
The EFI BIOS supports PS/2 emulation of USB keyboards and mice. During POST, the EFI BIOS initializes and configures the root hub ports and then searches for a keyboard and/or a mouse on the USB hub and then enables them.

7.0 EFI BIOS Setup

7.1 Introduction

The Aptio EFI BIOS Setup utility can be used to view and change the EFI settings for the SBC. The EFI BIOS Setup program is accessed by pressing the F2 key (Esc-2) or F4 key or Delete key (if connected via local USB keyboard) when the EFI booting screen appears. See Figure 36.

Figure 36. EFI BIOS welcome screen



```
Intel Corporation
Version 1.17.1057. Copyright (C) 2006 American Megatrends, Inc.
Press <F4> to enter setup
Bios Version: WH500ES0.86E.01.02.0000.062820071709
Platform ID:  MPCBL0050
1 GB system memory found (1 GB effective memory available)
Current Memory Speed: 533 MT/s (266 MHz)
Intel(R) Xeon(R) CPU            5138  @ 2.13GHz
Intel(R) Xeon(R) CPU            5138  @ 2.13GHz

Press <ESC> to Continue....1
```

Note: After payload power is applied to the boards (board entered M4), EFI BIOS starts the POST procedure and there will be only POST codes displayed in top right corner of the screen. It may take several tens of seconds before the welcome screen appears. This is normal board operation mode.

Note: By default EFI BIOS waits 10s for user input before continuing the boot procedure. To speed up the boot process, the user can change that value in the boot option menu.

Table 50. EFI setup program menu bar

Main	Advanced	Security	Server Mgmt	Boot Options	Boot Mgr	Error Mgr	Exit
General system information page	Processor, Chipset, Memory and I/O configuration	Configures EFI BIOS Passwords	Configures other server options	Configures Boot devices order	Allows boot from chosen device	Error Log	Saves or discards changes to Setup program options

7.2 Main Menu

To access this menu, select **Main** on the menu bar at the top of the screen. The Main menu options are described in Table 51.

Table 51. Main menu screen and options

Feature	Options	Description
Version	Info Only	EFI BIOS version ID
BIOS Build date	Info only	EFI BIOS build date
Processor	Info only	Displays processor type, speed, and count
Total Memory	Size	Displays system memory size of recognized DIMMs
System Date	Can set date	Allows setting the current date
System Time	Can set Time	Specifies the current time

7.3 Advanced Menu

To access this menu, select **Advanced** on the menu bar at the top of the screen.

Table 52 describes the Advanced menu, which sets advanced chipset features.

Table 52. Advanced menu options

Feature	Options	Description
Processor	Select to display submenu	Display CPU details, configure Intel SpeedStep® Mode and CPU features
Memory	Select to display submenu	Displays system memory configuration detected during POST
ATA Controller	Select to display submenu	Displays on-board P-ATA Flash storage configuration
Mass Storage Configuration	Select to display submenu	Enable/disable SAS controller
Serial Port	Select to display submenu	Serial port settings
USB Configuration	Select to display submenu	Enable/disable USB devices and configure USB2.0 support
PCI Configuration	Select to display submenu	Configure on-board PCI device settings (Gigabit Ethernet for base and fabric interface)

7.3.1 Processor Submenu

To access this submenu, select **Advanced** on the menu bar, then **Processor.** The CPU configuration options are given in Table 53.

Table 53. Processor submenu options

Feature	Options	Description
Core Frequency	Info Only	Display CPU frequency
System Bus Frequency	Info Only	Display front side bus speed
Enhanced Speedstep* Technology	**Enabled**/Disable	Enables Speedstep Technology Enhanced Intel SpeedStep® Technology allows the system to dynamically adjust processor voltage and core frequency, which can result in decreased average power consumption and decreased average heat production.
Core Multiprocessing	**Enabled**/Disable	Core Multi-processing sets the state of logical processor cores in a package. [Disabled] sets only logical processor core 0 as enabled in each processor package.
Virtualization Technology	Enabled/**Disabled**	Intel® Virtualization Technology allows a platform to run multiple operating systems and applications in independent partitions. Note: A change to this option requires the system to be powered off and then back on before the setting will take effect.
Execute Disable Bit	**Enabled**/Disabled	Execute Disable Bit can help prevent certain classes of malicious buffer overflow attacks. When disabled, force the XD feature flag to always return 0
Hardware Prefetcher	**Enabled**/Disabled	Hardware Prefetcher is a speculative prefetch unit within the processor(s)
Adjacent Cache Line Prefetch	Enabled/**Disabled**	[Enabled] - Cache lines are fetched in pairs (even line + odd line). [Disabled] - Only the current cache line required is fetched.
Processor 1 Information	Select to display submenu	Displays detailed information about CPU1
Processor 2 Information	Select to display submenu	Displays detailed information about CPU2
Note: **Bold** text indicates default setting.		

7.3.2 Memory Submenu

To access this submenu, select **Advanced** on the menu bar, then **Memory** submenu. The Memory submenu options are given in Table 54.

Table 54. Memory submenu options (Sheet 1 of 2)

Feature	Options	Description
Total Memory	Info Only	Displays the total amount of system memory
Effective Memory	Info Only	Size of the memory visible for the operating system
Current Configuration	Info Only	Displays the information on current memory config.

Table 54. Memory submenu options (Sheet 2 of 2)

Feature	Options	Description
Current Memory Speed	Info Only	Displays current memory bus speed. MPCBL0050 supports only 533MT/s memory speed
Configure Memory RAS and Performance	Select to display submenu	Displays the RAS configuration information. Note: MPCBL0050 does provide support for Memory Mirroring and Memory Sparing.
DIMM Information	Info Only	Displays FBDIMM slot population.
Note: **Bold** text indicates default setting.		

7.3.3 ATA Controller Submenu

To access this submenu, select **Advanced** on the menu bar, then **ATA Controller** submenu. The ATA Controller options are given in Table 55.

Table 55. ATA controller submenu options

Feature	Options	Description
Onboard PATA Controller	Enabled/ **Disabled**	Enables onboard Parallel ATA controller.
Primary IDE Master	Info only	By default the PATA controller is disabled therefore no IDE device is detected. When PATA controller is enabled this item will display information about flash storage provided on the board.
Primary IDE Master	Info only	By default the PATA controller is disabled therefore no IDE device is detected. When PATA controller is enabled this item will display information about flash storage provided on the board

7.3.4 Mass Storage Submenu

To access this submenu, select **Advanced** on the menu bar, then **Mass Storage** submenu. The mass storage options are given in Table 56.

Table 56. Mass storage submenu options

Feature	Options	Description
SAS Option ROM	**Enabled**/ Disabled	Enable or Disable the onboard Serial Attached SCSI (SAS) Controller.

7.3.5 Serial Port Submenu

To access this submenu, select **Advanced** on the menu bar, then **Serial Port** submenu. The Serial Port options are given in Table 57.

Table 57. Serial port submenu options

Feature	Options	Description
Address	Info Only	Displays base I/O address for serial port
IRQ	**Info Only**	Displays IRQ used for the port

7.3.6 USB Configuration Submenu

To access this submenu, select **Advanced** on the menu bar, then **USB Configuration**. The USB configuration options are given in Table 58.

Table 58. USB configuration submenu options

Feature	Options	Description
USB Devices Enabled	Info Only	Displays the number of USB devices detected by EFI BIOS
USB Controller	**Enabled**/ Disabled	Enable USB host controller
Legacy USB Support	**Enabled/** Auto	PS/2 emulation for USB keyboard and USB mouse devices. [Auto] - Legacy USB support will be enabled if a USB device is attached.
Device Reset Timeout	10 sec 20 sec 30 sec 40 sec	USB Mass storage device Start Unit command timeout
USB 2.0 Controller	**Enabled/** Disabled	Onboard USB ports will be enabled to support USB 2.0 mode. USB devices will operate in USB 1.1 mode during POST.
Note: **Bold** text indicates default setting.		

7.3.7 PCI Configuration Submenu

To access this submenu, select **Advanced** on the menu bar, then **PCI** Submenu. The PCI configuration options are given in Table 59.

Table 59. On-board PCI device settings (Sheet 1 of 2)

Feature	Options	Description
PCI memory mapped I/O	1.5GB 1.75GB 2.00GB 2.25GB 2.5GB 2.75GB 3.00GB 3.25GB 3.50GB	Select the start of the reserved memory region for PCI memory mapped I/O space that ends at 4GB. Warning: Depending on the system configuration, this option may impact the amount of system memory detected by an OS without Physical Address Extension (PAE) support. For all PAE (Physical Address Extension) aware Operating Systems, 2.5GB should be selected. The system will remap memory and the OS will detect all memory installed in the system. If the installed OS does not support PAE, the maximum memory size detected is linked to the setup option selected. For example, if 2.5GB is selected, only 2.5GB will be detected by the OS.
Memory Mapped I/O above 4GB	Enabled **/** **Disabled**	Enable or disable memory mapped I/O of 64-bit PCI devices to 4GB or greater address space.
Onboard ports A and B option ROM	**Enabled/** Disabled	Enables Option ROM for base interface (PICMG) Gigabit Ethernet LAN Controller [Ports A & B)
Onboard ports C/D/E and F option ROM	**Enabled/** Disabled	Enables Option ROM for fabric interface (PICMG) Gigabit Ethernet LAN Controller [Ports C, D, E & F)
NIC MAC Address	Info Only	Displays MAC addresses of all available Gigabit Ethernet LAN ports

Table 59. On-board PCI device settings (Sheet 2 of 2)

AMC Link Width	**PCI-Ex x8/**PCI-Ex x4	Choose the PCI-Ex link with between MCH and AdvancedMC ports.
IO Acceleration Technology*	**Enabled/** Disabled	Intel® I/O Acceleration Technology (I/OAT*) accelerates TCP/IP processing for onboard NICs, delivers data-movement efficiencies across the entire server platform, and minimizes system overhead. To utilize this feature OS driver must support new DMA engine embedded into the MCH.
Note: **Bold** text indicates default setting.		

7.4 Security Menu

To access this menu, select **Security** from the menu bar at the top of the screen. The options are given in Table 60.

Table 60. Security menu options

Feature	Options	Description
Administrator Password	Info Only	Display the Supervisor Password status Installed/not Installed
User Password	Info Only	Display the Supervisor Password status Installed/not Installed
Admin Password		Administrator password is used to control change access in EFI BIOS Setup Utility. Only alphanumeric characters can be used. Maximum length is 7 characters. It is not case sensitive. Note: Administrator password must be set in order to use the user account.
User Password		User password is used to control entry access to EFI BIOS Setup Utility. Only alphanumeric characters can be used. Maximum length is 7 characters. It is not case sensitive. Note: Removing the administrator password will also automatically remove the user password.
Note: **Bold** text indicates default setting.		

7.5 Server Management Menu

To access this menu, select **Server Management** on the menu bar at the top of the screen. Table 61 describes the Advanced menu, which sets advanced chipset features.

Table 61. Server Management options (Sheet 1 of 2)

Feature	Options	Description
Assert NMI on SERR	**Enabled**/Disabled	On SERR, generate an NMI and log an error. Note: [Enabled] must be selected for the Assert NMI on PERR setup option to be visible.
Assert NMI on PERR	**Enabled**/Disabled	On PERR, generate an NMI and log an error. Note: This option is only active if the Assert NMI on SERR option is [Enabled] selected."
Action on IERR	**No Action**/Hard Reset/ Assert NMI	This option allows to choose which action will be taken by IPMC on detection of IERR signal,
Clear System Event Log	Enabled/**Disabled**	Clears the System Event Log. All current entries will be lost. Note: This option will be reset to [Disabled] after a reboot.

Table 61. Server Management options (Sheet 2 of 2)

Feature	Options	Description
FRB-2 Enable	**Enabled**/Disabled	Fault Resilient Boot (FRB). EFI BIOS programs the BMC watchdog timer for approximately 6 minutes. If EFI BIOS does not complete POST before the timer expires, the BMC will reset the system.
O/S Boot Watchdog Timer	Enabled/**Disabled**	EFI BIOS programs the watchdog timer with the timeout value selected. If the OS does not complete booting before the timer expires, the BMC will reset the system and an error will be logged. When booting next time EFI BIOS will try booting from subsequent device on the booting list. Refer to Section 6.11.1, "Progressive Boot Mechanism" on page 144 for details. Note: Requires OS support or Intel Management Software
O/S Boot Watchdog Timer Policy	**Power Off**/Reset	If the OS watchdog timer is enabled, this is the system action taken if the watchdog timer expires. [Reset] - System performs a reset. [Power Off] - System powers off.
O/S Boot Watchdog Timer Timeout	5 minutes 10 minutes 15 minutes 20 minutes	If the OS watchdog timer is enabled, this is the timeout value EFI BIOS will use to configure the watchdog timer.
Fabric Interface Ports Options	Select to display submenu	Configures Fabric Interface ports routing
System Information	Select to display submenu	Displays System Information
Console Redirection	Select to display submenu	View/Configure console redirection information and settings.

7.5.1 Fabric Interface Ports Options Submenu

To access this submenu, select **Advanced** on the menu bar, then **Fabric Interface Ports Options Options submenu**. The Fabric Interface Ports Options options are presented in Table 62.

Table 62. Fabric Interface Ports Options submenu options

Feature	Options	Description
Ports C/D redirection	To RTM **To Backplane**	Configures routing of Gigabit Ethernet ports C and D. 1. Note: When ports C/D are redirected to the RTM, AdvancedMC GbE ports will be automatically connected to the Backplane.
Ports E/F redirection	Both to Front Panel (FP) Both to RTM **To backplane** Port 1 to FP Port 2 to RTM Port 1 to RTM, Port2 to FP	Configures routing of Gigabit Ethernet ports E and F

Note: The settings are applied only after board reset for this option.

7.5.2 Console Redirection Submenu

To access this submenu, select **Server Management** on the menu bar, then **Console Redirection**. The Console Redirections configuration options are given in Table 63.

Table 63. Remote Access Configuration submenu options

Feature	Options	Description
Flow Control	**None** Hardware Software	Select flow control for console redirection
Baud Rate	**115200** 57600 38400 19200 9600	Serial port settings. Default is 115200, 8, n, 1
Terminal Type	ANSI **VT 100** VT-UTF8	Select the target terminal type
Legacy OS Redirection	Enabled/ **DIsabled**	This option will enable legacy OS redirection (i.e., DOS) on serial port. If it is enabled the associated serial port will be hidden from the legacy OS
Note: **Bold** text indicates default setting.		

7.6 Boot Options Menu

To access this menu, select **Boot** from the menu bar at the top of the screen. Table 64 describes the Advanced menu, which sets advanced chipset features.

Table 64. Boot Options menu

Feature	Options	Description
Boot Timeout	Seconds	The number of seconds EFI BIOS will pause at the end of POST to allow the user to press the [F2] key for entering the EFI BIOS Setup Utility. Valid values are 0-65535. Zero is the default. A value of 65535 will cause the system to go to the Boot Manager menu and wait for user input for every system boot.
Boot Option #X	List of devices	Set system boot order by selecting the boot option for this position.
Quite Boot	Enable/Disable	Disables logging Post messages to the Console
POST Error Pause	Enable/Disable	When enabled, EFI BIOS will pause on any errors that occurs during POST
Hard Disk Order	Select to display submenu	Set hard disk boot order by selecting the boot option for this position. Appears when more than 1 hard disk drive is in the system.
CDROM Order	Select to display submenu	Set CDROM boot order by selecting the boot option for this position. Appears when more than 1 CDROM drive is in the system.
Network Device Order	Select to display submenu	Set network device boot order by selecting the boot option for this position. This option appears when PXE Boot option ROM is enabled for onboard interfaces

7.6.1 Hard Disk Order Submenu

To access this submenu, select **Boot Options** on the menu bar, then **Hard Disk Order**. The boot settings configuration options are given in Table 65.

Table 65. Hard Disk Order submenu options

Feature	Options	Description
Hard Disk #1	List of available HDDs	Set hard disk boot order by selecting the boot option for this position.
Hard Disk #2	List of available HDDs	Set hard disk boot order by selecting the boot option for this position.
Hard Disk #3	List of available HDDs	Set hard disk boot order by selecting the boot option for this position.
Note: **Bold** text indicates default setting.		

7.6.2 Network Device Order Submenu

To access this submenu, select **Boot Options** on the menu bar, then **Network Device Order**. The boot settings configuration options are given in Table 66.

Table 66. Network Device submenu options

Feature	Options	Description
Network Device #1	List of available Network bootable devices	Choose first network device
Network Device #2	List of available Network bootable devices	Choose second network device
Network Device #3	List of available Network bootable devices	Choose third network device
Network Device #4	List of available Network bootable devices	Choose fourth network device
Network Device #5	List of available Network bootable devices	Choose fifth network device
Network Device #6	List of available Network bootable devices	Choose sixth network device
Note: **Bold** text indicates default setting.		

7.7 Boot Manager Menu

To access this menu, select **Boot** from the menu bar at the top of the screen. This menu allows you to immediately boot the chosen boot device. Just select the device and press enter. Using this menu for booting does not change boot order set in boot options menu.

7.8 Error Manager

This screen only displays System errors that occurred during POST.

7.9 Exit Menu

To access this menu, select **Exit** from the menu bar at the top of the screen. The options are given in Table 67.

Table 67. Exit menu options

Feature	Description
Save Changes and Exit	Exit EFI BIOS Setup Utility after saving changes. The system will reboot if required. The [F10] key can also be used.
Discard Changes and Exit	Exit system Setup without saving changes Similar to pressing ESC
Save Changes	Save changes without exiting EFI BIOS Setup Utility. Note: Saved changes may require a system reboot before taking effect.
Discard Changes	Discard changes without exiting
Restore Defaults	Load factory default values for all EFI BIOS Setup Utility options. The [F9] key can also be used.
Save As User Default Values	Save current EFI BIOS Setup Utility values as custom user default values. If needed, the user default values can be restored via the Load User Default Values option below. Note: Clearing CMOS or NVRAM will cause the user default values to be reset to the factory default values.
Load User Default Values	Restores previously stored user configuration

8.0 EFI BIOS Error Messages and Checkpoints

8.1 EFI BIOS POST Error Messages

Table 68 lists the EFI BIOS error messages supported by the MPCBL0050 SBC. The listed error codes are logged to POST error sensor events.

Table 68. EFI BIOS POST error messages (Sheet 1 of 3)

Error Code	Error Message
0012	CMOS date / time not set
0048	Password check failed
0108	Keyboard component encountered a locked error
0109	Keyboard component encountered a stuck key error
0140	PCI component encountered a PERR error
0141	PCI component encountered a resource conflict
0146	Insufficient memory to shadow PCI ROM
0192	L3 cache size mismatch
0193	Processor stepping mismatch
0194	CPUID, processor family are different
0195	Front side bus mismatch
0196	Processor model mismatch
0197	Processor speed mismatched
0198	Processor family is unsupported
5220	CMOS/NVRAM Configuration Cleared
5221	Passwords cleared by jumper
5224	Password clear jumper is set
8110	Processor 01 internal error (IERR) on last boot
8111	Processor 02 internal error (IERR) on last boot
8120	Processor 01 thermal trip error on last boot
8121	Processor 02 thermal trip error on last boot
8130	Processor 01 disabled
8131	Processor 02 disabled
8140	Processor 01 Failed FRB-3 Timer
8141	Processor 02 Failed FRB-3 Timer
8160	Processor 01 unable to apply Microcode update
8161	Processor 02 unable to apply Microcode update
8170	Processor 01 failed Self Test (BIST)
8171	Processor 02 failed Self Test (BIST)

Table 68. EFI BIOS POST error messages (Sheet 2 of 3)

Error Code	Error Message
8180	Processor 01 BIOS does not support the current stepping for processor
8181	Processor 02 BIOS does not support the current stepping for processor
8190	Watchdog timer failed on last boot
8198	Operating system boot watchdog timer expired on last boot
8300	Baseboard management controller failed self-test
84F2	Baseboard management controller failed to respond
84F3	Baseboard management controller in update mode
84F4	Sensor data record empty
84FF	System event log full
8500	Memory component was not detected
8520	DIMM1 failed Self Test (BIST)
8522	DIMM2 failed Self Test (BIST)
8524	DIMM3 failed Self Test (BIST)
8526	DIMM4 failed Self Test (BIST)
8540	DIMM1 Memory component has been disabled
8541	DIMM2 Memory component has been disabled
8542	DIMM3 Memory component has been disabled
8543	DIMM4 Memory component has been disabled
8560	DIMM1 Memory component encountered serial presence detection (SPD) error
8562	DIMM2 Memory component encountered serial presence detection (SPD) error
8564	DIMM3 Memory component encountered serial presence detection (SPD) error
8566	DIMM4 Memory component encountered serial presence detection (SPD) error
8580	DIMM1 Memory component has encountered correctable ECC error
8582	DIMM2 Memory component has encountered correctable ECC error
8584	DIMM3 Memory component has encountered correctable ECC error
8586	DIMM4 Memory component has encountered correctable ECC error
85A0	DIMM1 Memory component has encountered un-correctable ECC error
85A2	DIMM2 Memory component has encountered un-correctable ECC error
85A4	DIMM3 Memory component has encountered un-correctable ECC error
85A6	DIMM4 Memory component has encountered un-correctable ECC error
85E0	Memory component has encountered a non specific error
85FC	Fatal Memory component was not detected
8601	Jumper to boot from backup FWH1 is set
8602	Watch Dog timer expired
8603	BIOS checksum fail, system halted
8604	Chipset Reclaim of non critical variables complete
9226	Keyboard component encountered a controller error
9246	Mouse component encountered a controller error
9266	Local console component encountered a controller error
9268	Local console component encountered an output error

Table 68. EFI BIOS POST error messages (Sheet 3 of 3)

Error Code	Error Message
9269	Local console component encountered a resource conflict error
9286	Remote console component encountered a controller error
9287	Remote console component encountered an input error
9288	Remote console component encountered an output error
92A3	Serial port component was not detected
92A6	Serial port component encountered a controller error
92A7	Serial port component encountered an input error
92A8	Serial port component encountered an output error
92A9	Serial port component encountered a resource conflict error
9426	PCI component encountered a controller error
9427	PCI component encountered a read error
9428	PCI component encountered a write error
94C6	LPC component encountered a controller error
94C9	LPC component encountered a resource conflict error
9667	PEI Module component encountered an illegal software state error
9687	DXE Core component encountered an illegal software state error
96A0	DXE boot services driver component has encountered a non specific error
96A7	DXE boot services driver component encountered an illegal software state error
96AB	DXE boot services driver component encountered invalid configuration
96C7	DXE RT driver component encountered an illegal software state error
96E7	SMM driver component encountered an illegal software state error
A022	Processor component encountered a mismatch error
A044	BaseBoard Management Controller in Update Mode
A421	PCI component encountered a SERR error
A5A0	PCI Express component encountered a PERR error
A5A1	PCI Express component encountered a SERR error
A5A4	PCI Express IBIST error

8.2 Port 80h POST Codes

During the POST, the EFI BIOS generates diagnostic progress codes (POST-codes) to I/O port 80h. If the POST fails, execution stops and the last POST code generated is left at port 80h. This code is useful for determining the point where an error occurred.

Port 80h POST codes can be retrieved from IPMC with an OEM IPMI command. Refer to Table 46, "Intel OEM commands (net function 0x30h)" on page 130. In the case of SBC hang, Port 80h can be retrieved remotely from the chassis management module.

Table 69 lists all possible POST codes generated by the BIOS.

Note: POST codes are also displayed on the serial console and provide information on the boot progress.

Table 69. POST error codes (Sheet 1 of 3)

Group	Code	Description
Host Processor related codes	0x10	Power On Init of Boot Strap Processor
	0x11	BSP Cache Initialization
	0x12	AP processor initialization
	0x13	SMM initialization
Chipset/ Memory related codes	0x21	Chipset initialization
	0x22	Reading SPD data from DIMM
	0x23	Detecting memory presence
	0x24	Programming timing parameters in the memory controller and the DIMMs
	0x25	Configuring memory
	0x26	Optimizing memory settings
	0x27	Initializing memory, such as ECC init
	0x28	Testing memory
Recovery related codes	0x30	Crisis recovery has been initiated because of a user request
	0x31	Crisis recovery has been initiated by software
	0x34	Loading crisis recovery capsule
	0x35	Handing off control to the crisis recovery capsule
IO Bus related codes	0x50	Enumerating PCI busses
	0x51	Allocating resources to PCI bus
	0x52	Hot Plug PCI controller initialization
	0x58	Resetting USB bus
	0x5A	Resetting PATA/SATA bus and all devices
	0x5C	Resetting SMBUS
Output device codes	0x70	Resetting the VGA controller
	0x71	Disabling the VGA controller
	0x72	Enabling the VGA controller
	0x78	Resetting the console controller
	0x79	Disabling the console controller
	0x7A	Enabling the console controller

Table 69. POST error codes (Sheet 2 of 3)

Group	Code	Description
Input device codes	0x90	Resetting the keyboard
	0x91	Disabling the keyboard
	0x92	Detecting the presence of the keyboard
	0x93	Enabling the keyboard
	0x94	Clearing keyboard input buffer
	0x95	Instructing keyboard controller to run Self Test (PS2 only)
	0x98	Resetting mouse
	0x99	Detecting presence of mouse
	0x9A	Enabling mouse
Boot device codes	0xB0	Resetting fixed media
	0xB1	Disabling fixed media
	0xB2	Detecting presence of a fixed media (IDE hard drive detection etc.)
	0xB3	Enabling/configuring a fixed media
	0xB8	Resetting removable media
	0xB9	Disabling removable media
	0xBA	Detecting presence of a removable media (IDE CD-ROM drive detection etc.)
	0xBB	Enabling/configuring a removable media
BDS codes	0xD0	Booting from boot option 0
	0xD1	Booting from boot option 1
	0xD2	Booting from boot option 2
	0xD3	Booting from boot option 3
	0xD4	Booting from boot option 4
	0xD5	Booting from boot option 5
	0xD6	Booting from boot option 6
	0xD7	Booting from boot option 7
	0xD8	Booting from boot option 8
	0xD9	Booting from boot option 9
	0xDA	Booting from boot option A
	0xDB	Booting from boot option B
	0xDC	Booting from boot option C
	0xDD	Booting from boot option D
	0xDE	Booting from boot option E
	0xDF	Booting from boot option F

Table 69. POST error codes (Sheet 3 of 3)

Group	Code	Description
Software codes	0xE2	Permanent memory found
	0xE4	Entered EFI driver execution phase (DXE)
	0xE6	Started connecting drivers
	0xE7	Waiting for user input
	0xE8	Checking password
	0xE9	Entering BIOS setup
	0xEA	Flash Update
	0xEE	Calling Int 19
	0xF4	Entering Sleep state
	0xF5	Exiting Sleep state
	0xF8	OS has requested EFI to close boot services
	0xF9	OS has switched to virtual address mode
	0xFA	OS has requested the system to reset
	0x00	End of post-switch off post status LEDs

9.0 Serial Over LAN

Serial over LAN (SOL) is a packet format and protocol defined in the IPMI v2.0 specification for transmitting serial port data over Ethernet using IPMI over LAN (RMCP+) messages. This two-way redirection of a blade’s serial port data over Ethernet is independent of the operating system or any applications executing on it. The EFI BIOS also supports redirection of its console over a serial port, which can be redirected over the network for remote access.

The SOL mechanism, coupled with an SOL client utility executing on a remote node, allows the viewing of serial port data from any IPMI v2.0 based, SOL-enabled blade, thus providing a virtual remote terminal server for accessing the blade’s serial port character stream.

9.1 References

- *Intelligent Platform Management Interface Specification v2.0*, dated June 1, 2004
- *AES - Advanced Encryption Standard,* http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf

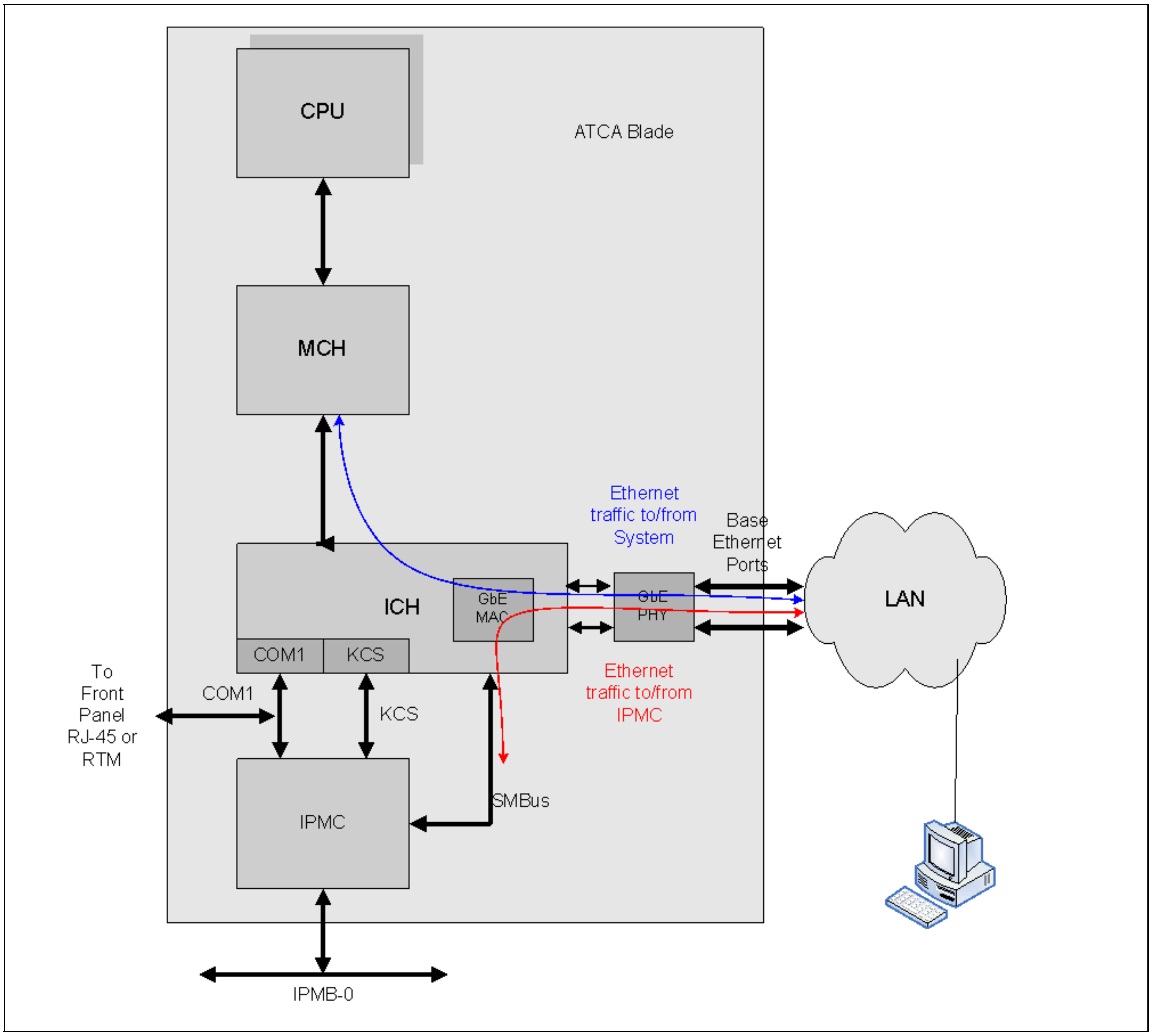
9.2 SOL Architecture

The SOL implementation on the Intel NetStructure® MPCBL0050 blade is based on the definition in Section 15 of the IPMI v2.0 specification.

Serial over LAN (SOL) enables suitably designed blades and servers to transparently redirect a serial character stream of a baseboard UART to/from a remote client via LAN over RMCP+ sessions. This enables users at remote consoles to access the serial port of a blade/server and interact with a text-based EFI BIOS console, operating system, command line interfaces, and serial text-based applications.

Figure 37 is a block diagram of the SOL implementation on the blade.

Figure 37. SOL block diagram


CPU
ATCA Blade
MCH
Ethernet traffic to/from System
Base Ethernet Ports
LAN
ICH
GbE MAC
GbE PHY
COM1
KCS
Ethernet traffic to/from IPMC
To Front Panel RJ-45 or RTM
COM1
KCS
SMBus
IPMC
IPMB-0

This architecture requires the following components to perform Serial over LAN operations:

- SOL-capable firmware executing on the Intelligent Platform Management Controller (IPMC). The IPMC also provides a dedicated SMBus connection to the base interface Ethernet controller. This connectivity is not shared with IPMB-0 or any other I^{2}C/ SMBus/IPMB connections that the IPMC may provide on the blade for hardware management.
- The base interface Ethernet controller, which provides a sideband interface port to the IPMC over which SOL traffic is redirected. The Ethernet controller also filters packets based on MAC address, RMCP port number, or IP address, and forwards the packets to the IPMC over the sideband interface.

- Client software running on any remote node that has LAN access to the blade whose serial port data is to be accessed. The IPMC is responsible for controlling the serial hardware MUX, the transformation of serial data to and from network packets, and the transmission and reception of SOL network packets through the Ethernet controller sideband interface port.

9.2.1 Architectural Components

9.2.1.1 IPMC

As shown in the block diagram in Figure 37, the IPMI controller on the blade provides a UART interface to the blade's serial port (COM1). This interface is used by the IPMC firmware to write data to the blade's serial port or to receive the blade's serial port data written by the EFI BIOS or the operating system. The serial port may be connected to either to the IPMI controller or to RJ-45 connector(s) on the front panel and the RTM (if available). The switching of the serial port between front-panel/RTM and the IPMC's UART port is controlled by the IPMC firmware.

The IPMC also provides a dedicated SMBus connection to the Ethernet controller, whose ports are connected to the base Ethernet interface.

The KCS interface is used for interaction between the IPMC firmware and software executing on the OS (for example, OpenIPMI or OpenHPI) by sending/receiving IPMI messages, and does not play a role during SOL communication. The KCS interface however, may be used for SOL-related IPMC configuration, as described below.

9.2.1.2 Ethernet Controller

The Ethernet controller provides an advanced pass-through mode of operation where the controller allows the on-board IPMC to communicate over the Ethernet ports using a sideband interface port.

The Ethernet controller embedded in ICH is available with standby (management) power, so that it is possible to view the initial serial port data written by the EFI BIOS or the OS.

9.3 Theory of Operation

9.3.1 Front Panel Serial Port or RTM

By default, the serial console is connected to the serial port connector(s) on the front panel and RTM (if available).

If serial cables are connected to both the front panel and RTM connectors, both connections will be active. However, only one user is allowed to use the serial session.

9.3.2 Serial Over LAN

IPMC firmware is pre-configured at manufacturing time with default serial port settings (baud rate, parity bits, data bits, stop bits, flow control), user name and password for RMCP+ sessions. The SOL feature, however, is disabled by default.

The IP address to be used by IPMC can be configured during initial setup of the blade in the system.

The IP address, once configured by the reference script provided, is stored in a non-volatile memory and is persistent across IPMC update and payload resets. The IP addresses are assigned to the IPMC independently of the host (OS) IP addresses and they need not match. The IP addresses used by the OS are not visible to the IPMC.

To start SOL communication, the user invokes the SOL client utility with the IP address of the blade and a series of authentication parameters (username, password, privilege level, cipher suite, etc.). The IPMI v2.0 specification allows for AES encryption algorithms for encryption of payload data sent over the network, including AES-128, which uses 128-bit cipher keys.

The SOL client utility initializes an RMCP+ session with the blade and activates SOL. When authentication is successfully completed, the IPMC firmware collects serial port data from the blade's serial port, formats it into network packets and forwards it to the SOL client utility over the SOL session. The SMBus sideband interface port between the IPMC and base interface Ethernet controller is used for this purpose. The SOL client utility receives the packets, extracts the serial port data, and displays it on the screen. The IPMC extracts the serial port data received from the SOL client utility and writes it to the serial port of the blade. This allows network redirection of blade's serial port data stream that is independent of the host OS or EFI BIOS. The Ethernet controller plays a critical role in redirecting the packets meant for the IPMC, based on receive filters.

The default SOL baud rate is 9.6 kbps.

Note: The EFI BIOS default baud rate is 115.200 kbps. If SOL is configured for a different baud rate, the EFI BIOS output is not seen using the SOL client until the EFI BIOS baud rate is set to match the SOL client baud rate.

9.4 Serial Over LAN Client

The SOL client establishes an IPMI-over-LAN connection with the IPMC on the blade. It then activates SOL, which effectively switches the board hardware to redirect serial traffic to the IPMC instead of the serial port. Any outbound characters from the UART are now packetized by the IPMC and sent over the network to the SOL client via the sideband interface port. Conversely, any input on the SOL client is packetized by the client and sent over the network to the IPMC, which is responsible for conveying it to the UART.

SOL data is carried over the network in UDP datagrams. IPMI 2.0 defines the specification of packet formats and protocols for SOL. As per the IPMI 2.0 specification, RMCP+ is the packet format with the Payload Type set to "SOL". Authentication, integrity, and encryption for SOL are part of the RMCP+ specification.

The ipmitool client (version 1.8.7 or higher) is required. The ipmitool software is open source. For more information, see Section 9.8.2.2.

This client needs to be downloaded and compiled on the Linux* operating system of your choice.

9.5 Reference Configuration Script

Intel provides an SOL reference script (reference_cfg) that sets up the various parameters required for SOL operation.

The SOL configuration reference script (reference_cfg) sends a sequence of IPMI commands to configure an SBC to enable the SOL feature. This script can be executed on a payload CPU for local configuration, or on a node that has network connectivity to the target SBC. Without this IPMC configuration, the SOL client utility is not able to communicate with the SBC. This script is provided to customers as an example of a semi-automated method of configuring systems and is not meant for use in production environments.

The ipmitool utility enables a user to establish an RMCP+ session with an SBC's management controller and activate two-way SOL packet communication.

It is important to note that while the ipmitool is a supported utility, reference_cfg is provided as an unsupported reference to be modified by customers to suit their specific environments and integration needs.

9.6 Supported Usage Model

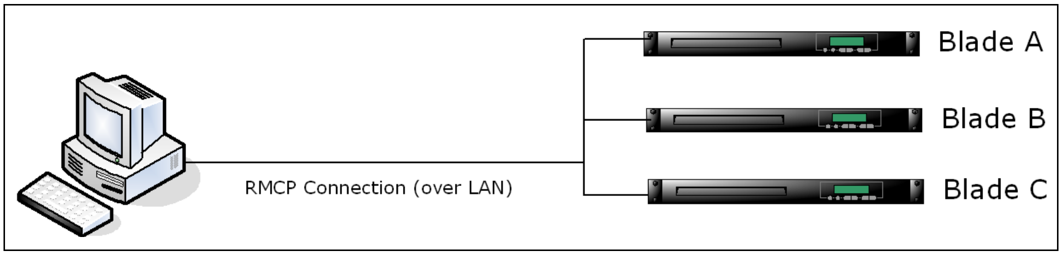
Customers are expected to use SOL to accomplish the following:

- EFI BIOS console redirection
- Remote terminal access for OS setup and viewing text console output

The ipmitool utility runs on a remote network node and communicates over the LAN interface. The remote node and the target SBC may be on the same subnet or on different subnets.

The reference script can be run on the remote node.

Figure 38. Reference script running on remote node, communicating over LAN



Note: The machine or "remote node" running ipmitool may also be an MPCBL0050 SBC within the same chassis.

9.6.1 Configuring the Blade for SOL

To configure a blade for SOL, the machine on which the configurator is installed (typically the remote node) needs to establish an RMCP connection with the SBC. The configurator sends commands and configuration settings to the SBC to configure and enable SOL operation. To configure a blade for SOL, reference_cfg needs to run either on the same blade as the IPMC and communicate via the KCS interface, or on a remote node. In the latter case, the script sends IPMI messages over the LAN to the SBC's shelf manager, which in turn bridges data to the IPMC's IPMB-0 interface.

The minimal per-blade configuration that must be set up includes the following:

- An IP/MAC address, subnet mask and default gateway
- ARP configuration
- User ID and password to authenticate access
- Channel, user, payload, and SOL privilege levels

The configuration utility is referring to the reference_cfg script described above.

9.7 Reference Script (reference_cfg)

9.7.1 SOL Configuration Reference Script (reference_cfg)

The reference script can run with no special setup. The script uses built-in `bash` commands as well as `grep` and `awk`. The environment in which the script runs must have `bash` installed at `/bin/bash` (or a symbolic link at that location), and must include `grep` and `awk` in the path.

The reference script is implemented as two separate bash files: reference_cfg, which contains the necessary IPMI commands, and `reference_func`, a library of supporting functions. When reference_cfg runs, it looks for the library in the following paths in the order listed:

1. the current working directory
2. /usr/lib/sbcutils
3. /home/scripts

If reference_cfg cannot find the library in any of these locations, it terminates with an error message.

When running the reference script on a remote node over RMCP via a shelf manager, RMCP to IPMB bridging must be enabled on the shelf manager. In the case of the Intel NetStructure® MPCMM0001/0002 Chassis Management Module, the following command enables RMCP:

```
    # cmmset -d rmcpenable -v 1
```

RMCP and KCS communication requires the OpenIPMI application library, version 1.4 or later. KCS communication further requires the OpenIPMI driver.

9.7.2 Default Behavior

To configure a blade for SOL communications, many items need configuration (for example, user information, channel parameters, LAN parameters and SOL parameters). Most of the values used for configuration appear as hard-coded default values.

9.7.3 SOL User Information

SBCs from Intel implement four different users, User1 through User4. User1 has a null username which is not editable. The script configures User2 as specifically enabled for SOL payloads.

The user name is “solusername”, zero-padded to a length of 16 bytes as per the IPMI 2.0 specification. The password is “soluserpassword”, zero-padded to 20 bytes as per the IPMI 2.0 extension.

9.7.4 LAN Parameters

The reference script configures IPMI channel one. The IPMI channel number used by reference_cfg can be changed to any IPMI LAN channel supported by the target SBC.

The configured channel must be a base interface, not a fabric interface. On the MPCBL0050 SBC, IPMC channel 1 corresponds to the Ethernet interface eth4.

Since reference_cfg uses IPMI channel one, this means that, for any SBC, eth0 is routed to a switch in slot seven (this may vary on different chassis implementations).

IP and MAC addresses supplied to the IPMC are specified on the command-line as per Table 70. The IP source is set to "static" and the subnet mask is set for a class C subnet. The gateway IP and MAC addresses should be specified with the command to enable RMCP communication across subnets. If the IP or MAC address options are missing from the command line, those parameters are not changed on the IPMC.

Note: The IP, MAC, gateway, and gateway MAC parameters are optional. If these settings have been previously configured and have not changed, it is not necessary to supply them on the command line every time the scripts is executed.

MAC address for the management interface can be found on the SBC label. It can also be easily calculated by adding 6 to the last byte of base interface port A MAC address.

Gratuitous ARPs generated by the BMC are enabled and sent every three seconds.

The `md5` authentication type is enabled for all privilege levels.

9.7.5 SOL Parameters

The SOL payload type is enabled with a privilege level of operator. Neither payload encryption nor payload authentication is forced. Serial characters are accumulated for as long as 40 milliseconds or as many as 50 characters. SOL packets are retried every 100 milliseconds, up to seven times. The default SOL baud rate is 9600 bps.

9.7.6 Channel Parameters

Channel 1 is configured as "always available", meaning that all RMCP traffic to the SBC LAN adaptor is routed exclusively to the IPMC and not to system software.

For channel 1, User2 is enabled for general IPMI messaging (as opposed to SOL only messaging). User2 has a privilege level of operator on channel 1.

9.7.7 Command Line Options

The available command-line options for the reference script are given in Table 70.

Table 70. SOL configuration reference script command-line options

Option	Meaning
-h	"help" - Display usage, version number and a list of options. When this option is specified, all other options are ignored.
-l	"location" - Specifies which blade to configure. When running on the MPCMM0001, this value should be one of ["blade1" to "blade14"]. When running on a remote node, this value should be the IPMB address. When running on the local processor, this option is ignored.
-i	"IP" - Specifies which IP address should be used to configure the blade. The IP address should be given in the form `###.###.###.###` where "###" is a decimal number from 0 to 255.
-j	Specifies the IP address of the subnet's gateway.
-n	Specifies the MAC address of the subnet's gateway.
-g	"Gratuitous ARPs" - Turns on gratuitous ARPs. If this switch is not supplied, then generated ARP responses are enabled instead.
-V	"Version" - Displays the version and quits.
-I	"Interface" - Specifies the interface used to communicate to the IPMC. Must be one of: kcs, lan or ipmb.
-m	"MAC" - Specifies which MAC address should be used to configure the blade. MAC address is given in the form of `##:##:##:##:##:##`
-H	RMCP-IP - IP address or host name of the shelf manager used to bridge RMCP messages to IPMB.

Table 70. SOL configuration reference script command-line options (Continued)

Option	Meaning
-U	"User" - Specifies the username for establishing the RMCP sessions. If not specified, the default value root is used.
-P	"Password" - Password for establishing RMCP sessions. If not specified, the default value cmmrootpass is used.
-A	"Authorization" - Authorization type for establishing the RMCP sessions. One of {none|straight|md2|md5}.
-?	Print the message and quit.

9.8 Setting up a Serial Over LAN Session

9.8.1 Target Blade Setup

The target blade is the SOL blade that sends the serial data to the client.

Ensure that the MPCBL0050 SBC has the following firmware/OS versions loaded:

- IPMC Firmware 1.02.00 or later
- EFI BIOS 1.02.00 or later
- `sbcutilities` 1.3.6.6 or later
- MontaVista* 4.0 LSP, Red Hat* RHEL 4.0 U4

9.8.1.1 EFI BIOS Configuration

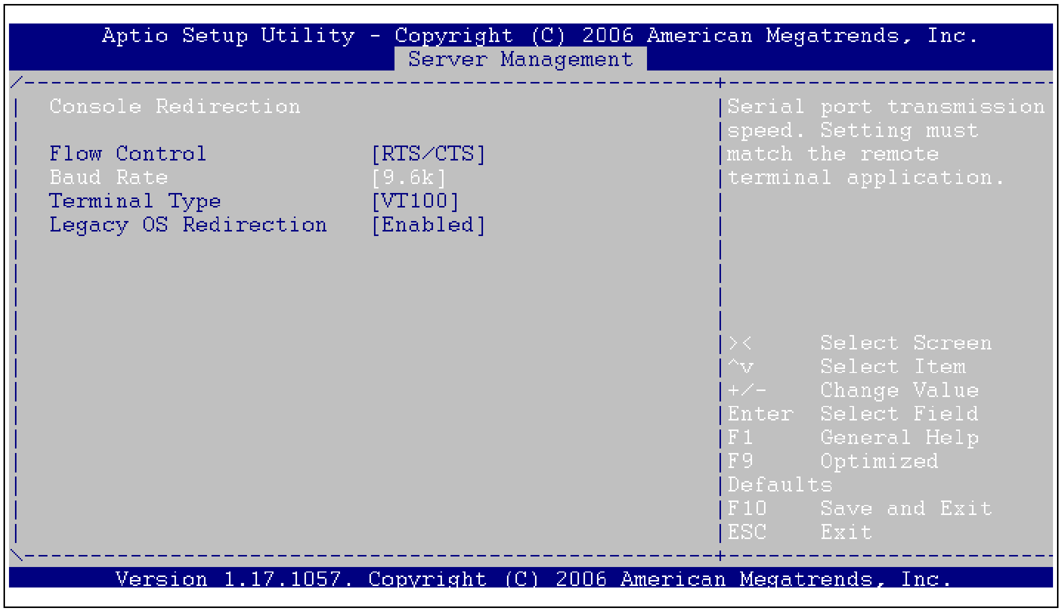
Configure the target blade EFI BIOS baud rate so that it is set to 9600. All examples in this document reflect a baud rate of 9600.

Configure the EFI BIOS on the SOL Target Blade as follows (refer also to Figure 39):

1. When the blade starts booting, press F4 from HyperTerminal (or equivalent terminal program) to enter EFI BIOS setup menu.
2. Choose the **Server Management** menu.
3. Choose the **Console Redirection** menu.
   Change the **Flow Control** parameter to "RTS/CTS".
4. Change **Serial Port Mode** to the 9600 baud rate if it is not already set to that speed.
5. Press **Esc**.
6. Choose **Save and Exit**.

Note: If the default EFI BIOS baud rate is changed to a baud rate other than 9600, then the reference_cfg script will need to be changed to match the same baud rate.

Figure 39. EFI BIOS configuration of SOL target blade



```
        Aptio Setup Utility - Copyright (C) 2006 American Megatrends, Inc.
                                   Server Management
/-----------------------------------------------------+----------------------------
|  Console Redirection                                |Serial port transmission
|                                                     |speed. Setting must
|  Flow Control             [RTS/CTS]                 |match the remote
|  Baud Rate                [9.6k]                    |terminal application.
|  Terminal Type            [VT100]                   |
|  Legacy OS Redirection    [Enabled]                 |
|                                                     |
|                                                     |
|                                                     |
|                                                     |
|                                                     |><     Select Screen
|                                                     |^v     Select Item
|                                                     |+/-    Change Value
|                                                     |Enter  Select Field
|                                                     |F1     General Help
|                                                     |F9     Optimized
|                                                     |Defaults
|                                                     |F10    Save and Exit
|                                                     |ESC    Exit
\-----------------------------------------------------+----------------------------
         Version 1.17.1057. Copyright (C) 2006 American Megatrends, Inc.
```

9.8.1.2 Operating System Configuration

Configure the operating system baud rate to match the EFI BIOS baud rate:

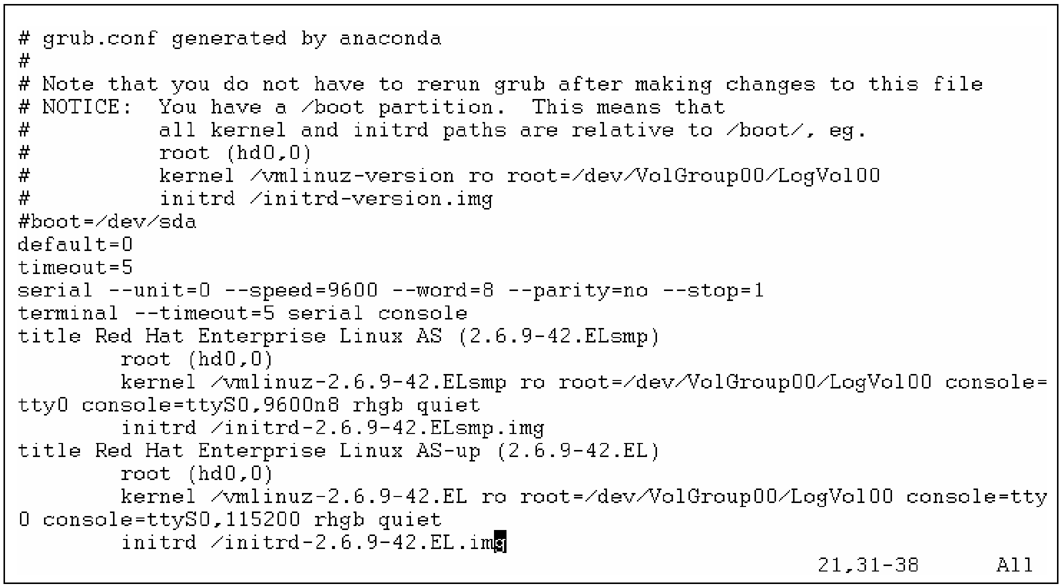
1. **When using MontaVista:**
   a. Edit the /etc/lilo.conf file
   b. Type `vi /etc/lilo.conf`
   c. Set the bootloader baud rate by adding or modifying a line as follows:
      `serial=0,9600n8r`
   d. Change the append line to read as follows:
      `append="ip=off console=ttyS0,9600n8r panic=5"`

Note: Hardware flow control is required (hence the `r` option)

   e. Once completed, save the lilo.conf file using the `wq!` command
   f. On the serial console, type `lilo`

2. **When using Red Hat RHEL 4 U4:**
   a. Edit the /boot/grub/grub.conf file
   b. Type `vi /boot/grub/grub.conf` (see Figure 40).
   c. Change the serial line to read:
      `serial --unit=0 --speed=9600 --word=8 --parity=no --stop=1`
   d. Change the kernel line to read:
      kernel /vmlinuz-2.6.XX-XX.ELsmp ro root=LABEL=/ console=ttyS0,
      9600n8 rhgb quiet
      Here you are adding `console=ttyS0,9600n8 rhgb quiet` to the end of the kernel line, if it does not already exist.
   e. Type `:wq!` to save the changes.

Figure 40. Configuration for RHEL



```
# grub.conf generated by anaconda
#
# Note that you do not have to rerun grub after making changes to this file
# NOTICE:  You have a /boot partition.  This means that
#          all kernel and initrd paths are relative to /boot/, eg.
#          root (hd0,0)
#          kernel /vmlinuz-version ro root=/dev/VolGroup00/LogVol00
#          initrd /initrd-version.img
#boot=/dev/sda
default=0
timeout=5
serial --unit=0 --speed=9600 --word=8 --parity=no --stop=1
terminal --timeout=5 serial console
title Red Hat Enterprise Linux AS (2.6.9-42.ELsmp)
        root (hd0,0)
        kernel /vmlinuz-2.6.9-42.ELsmp ro root=/dev/VolGroup00/LogVol00 console=
tty0 console=ttyS0,9600n8 rhgb quiet
        initrd /initrd-2.6.9-42.ELsmp.img
title Red Hat Enterprise Linux AS-up (2.6.9-42.EL)
        root (hd0,0)
        kernel /vmlinuz-2.6.9-42.EL ro root=/dev/VolGroup00/LogVol00 console=tty
0 console=ttyS0,115200 rhgb quiet
        initrd /initrd-2.6.9-42.EL.img
                                                             21,31-38      All
```

3. Ensure that at least one `agetty` process is running on the serial port. To do this, modify the /etc/inittab file. Issue the `vi /etc/inittab` command and change the system console by adding or modifying the “co” service as follows:
   a. For MontaVista:
      `co:2345:respawn:/sbin/agetty ttyS0 CON9600 vt102`
   b. For Red Hat RHEL:
      `co:2345:respawn:/sbin/agetty ttyS0 9600 vt100-nav`
4. Reboot the blade.
5. Change HyperTerminal to 9600, 8, n, 1, n to make sure the EFI BIOS, bootloader and OS come up at 9600 baud.
6. Optionally, if needed, configure the host OS IP address. This IP address can be the same or different as the IP address that will be assigned to the IPMC controller on the blade. The IP address of the host OS and IPMC should be on the same subnet. For the MPCBL0050 blade, use Eth4 for base interface port A.
   — For MontaVista:
      `# vi /etc/network/interfaces`
   — For Red Hat RHEL:
      `# vi /etc/sysconfig/network-scripts/ifcfg-eth`

9.8.1.3 sbcutils RPM Installation

The sbcutils RPM is installed via the following procedure. For complete details on sbcutils installation, refer to the sbcutilities RPM install procedure on the MPCBL0050 product page at http://www.intel.com.

1. Copy the RPM to the target blade. Ensure that the RPM copied is for the particular OS installed on the target blade.
2. Check the version of `sbcutils` installed: `rpm -q sbcutils`
3. If a previous version of the `sbcutils` RPM is installed, remove it by using this command: `rpm -e sbcutils`
4. Install a new version of `sbcutils` as follows:

— For MontaVista 4.0:

```
rpm -ivh sbcutils-1.X.X-X.i386-mv40.rpm
```

— For Red Hat RHEL 4U4:

```
rpm -ivh sbcutils-1.X.X-X.i386-rhel4.rpm
```

— For Red Hat RHEL 5:

```
rpm -ivh sbcutils-1.X.X-X.i386-rhel4.rpm
```

9.8.1.4 Execute the reference_cfg Script

1. For Red Hat Enterprise Linux only: Before using the reference_cfg script, start the IPMI drivers. For MontaVista, the IPMI drivers start automatically.

   Start IPMI driver by issuing the following commands:

   — `# /etc/init.d/ipmi start` (this starts ipmi drivers for this particular session only)

   — `# chkconfig ipmi on` (this starts ipmi drivers by default on the next reboot)

Note: The following apply to commands in the three procedures in step 2, below:

— When the SOL client and SOL target are behind the same gateway, the "<Gateway MAC Addr>" and "<Gateway IP Addr>" parameters can be omitted.

— The "-l <SOL Target IPMB Addr>" parameter needs to be entered as "-l 0xNN", where NN is the IPMB address of the target blade. The IPMB address depends upon the location of the target blade in the chassis. Refer to Table 79, "Mapping of physical slot to IPMB address in Intel NetStructure® MPCHC0001 14U Shelf" on page 232 for the IPMB address for each physical slot in an MPCHC0001 chassis. Other chassis may have different IPMB addresses.

2. Choose one of the three interfaces below to execute the reference_cfg script. It does not matter which interface is chosen. The reference_cfg script can be executed through any of these interfaces.

   — **Script executed on the local SOL target blade payload**. Communication is from host processor to local IPMC through the KCS interface.

   Execute this command to configure SOL on the target blade:

   ```
   reference_cfg -I kcs -g -i <SOL Target IP Addr>
   ```

   — **Script executed on the Intel CMM**. Communication is from CMM to target blade through IPMB. This requires the MPCMM0001 or MPCMM0002 CMM and firmware version 6.1.0.2779 or later.

   a. FTP the /usr/bin/reference_cfg and /usr/lib/sbcutils/reference_funcs files to the CMM /home/scripts directory. The CMM default IP address is 10.90.90.91. If you cannot FTP to the CMM, at the CMM prompt, type `vi /etc/ftpusers` and comment out the root (# root). Then type `:wq!` to save the file.

   b. Execute the following command to the change the reference_cfg file attributes: `chmod 777 reference_cfg`

   c. Execute the following command to configure SOL on the target blade:

   ```
   reference_cfg -I ipmb -g -i <SOL Target IP Addr>
       -j <Gateway IP Addr> -n <Gateway MAC Addr>
       -l <SOL Target IPMB Addr>
   ```

   — **Script executed on a remote computer**. Communication is from the remote computer to the CMM through a LAN interface (RMPC), which is then bridged to the target blade's IPMC through the IPMB.

a. Transfer the /usr/bin/reference_cfg and /usr/lib/sbcutils/reference_funcs files to the remote computer using FTP.

b. Transfer the /usr/share/doc/sbcutils/cmdPrivilege.ini file to both the active and standby CMMs. Place the cmdPrivelege.ini file in the /etc/ directory. This file is needed to set up RMCP.

c. Reboot both CMMs.

d. Once the CMMs have booted, from the remote computer execute the following command:

```
    reference_cfg -I lan  -g -i <SOL Target IP Addr>
        -j <Gateway IP Addr> -n <Gateway MAC Addr>
        -l <SOL Target IPMB Addr> -H <CMM IP Addr>
        -U root -P cmmrootpass -A md5
```

3. If the configuration script is successful, an output similar to the following is displayed on the console. Once the configuration has been successfully written to the SOL target blade, it is ready for a user to activate the SOL session from the client blade.

```
cmmget -l blade14 -t raw -d "0x06 0x41 1 0x80"
reference_cfg: Data response "0x2A 0x03"
reference_cfg: Success

cmmget -l blade14 -t raw -d "0x0C 0x21 1 0x01 0x01"
reference_cfg: Success

cmmget -l blade14 -t raw -d "0x0C 0x21 1 0x02 0x03"
reference_cfg: Success

cmmget -l blade14 -t raw -d "0x0C 0x21 1 0x03 0x07 0x2A"
reference_cfg: Success

cmmget -l blade14 -t raw -d "0x0C 0x21 1 0x04 0x03 0x0A"
reference_cfg: Success

cmmget -l blade14 -t raw -d "0x0C 0x21 1 0x05 0x06"
reference_cfg: Success

cmmget -l blade14 -t raw -d "0x0C 0x21 1 0x06 0x06"
reference_cfg: Success
```

9.8.2 Client Blade Setup

The client blade is the SOL blade that will activate SOL on the target and receive serial output from the target. The output is displayed on the client console.

Ensure that the MPCBL0050 SBC has the following firmware/OS versions loaded:

- IPMC Firmware 1.02.00 or later
- EFI BIOS 1.02.0.. or later
- System must be running a Linux* OS
- ipmitool 1.8.7 or later running on the client blade

Note: ipmitool needs to be installed on a Linux* computer and that computer can be located anywhere in the network. In the example described in this section, ipmitool is running on another blade in the same chassis.

9.8.2.1 Configure Ethernet Port

For MontaVista 4.0:

1. Configure the IP address of the Ethernet port using:
   `vi /etc network interfaces`

```
auto lo ethN
iface lo inet loopback
auto ethN
iface ethN inet static
    address 192.168.0.42
    network 192.168.0.0
    netmask 255.255.255.0
    broadcast 192.168.0.255
    gateway 192.168.0.1
```

2. Restart the network using the command:
   `/etc/init.d/networking restart`

For Red Hat RHEL:

1. Configure IP address of Ethernet port using:
   `vi /etc/sysconfig/network-scripts/ifcfg-ethN`

```
ifcfg-eth4
BOOTPROTO=static
IPADDR=10.90.90.113
NETMASK=255.0.0.0
ONBOOT=yes
TYPE=Ethernet
DHCP_HOSTNAME=rhel4u3
```

2. Execute this command to restart the network: `service network restart`

9.8.2.2 Installing ipmitool

1. Download ipmitool version 1.8.7 or later from http://ipmitool.sourceforge.net/
2. Install the ipmitool on the client blade. The ipmitool provides the SOL client interface.
3. Type: `tar zxvf  ipmitool-1.8.7.tar.gz`
4. Change directory to the ipmitool directory created after tar:
   `cd ipmitool-1.8.7`
5. Type: `./configure`
6. Type: `make install`
7. For Red Hat RHEL only: Before using ipmitool, start the IPMI drivers. In MontaVista, the IPMI drivers start automatically.
   — Start the IPMI driver by issuing the following commands:
     `# /etc/init.d/ipmi start`
     (this command starts the IPMI drivers for this particular session only)

     `# chkconfig ipmi on`
     (this command starts the IPMI drivers by default on the next reboot)
8. If the computer that ipmitool was just installed on has a local IPMC, the ipmitool installation can be tested by typing: `ipmitool raw 6 1`. If ipmitool is running correctly, the response should be in a format similar to:

```
20 81 01 03 02 3f 57 01 00 0c 08 01 05 01 00
```

9.8.2.3 Starting an SOL Session

Before starting an SOL session, first make sure you can `ping` the target blade.

On the client blade, execute this command:

```
     # ipmitool -I lanplus -L operator -H <SOL Target IP addr>
         -U solusername -P  soluserpassword sol activate
```

If the SOL session is activated successfully, the following message is displayed:

```
    [SOL Session operational.  Use ~? for help]
```

Press enter and you should see the target output on the console.

If the SOL session does not activate, run this command to see how ipmitool is configured:

```
     # ipmitool -I lanplus -L operator -H <SOL Target IP Addr>
         -U solusername -P  soluserpassword sol info
```

9.8.2.4 Checking SOL Configuration

To check the SOL configuration, the following command can be issued.

Note: This requires ipmitool to be installed on the SOL target blade also.

Use `ipmitool lan print 1` to display configuration for the 1st eth channel (this Ethernet port is connected to Ethernet switch located in Slot #7 on the MPCH0001 chassis) and `ipmitool lan print 2` to display configuration for the 2nd eth channel (Ethernet switch located in Slot #8).

```
root@DRBlade14:/usr/bin# ipmitool lan print 1
Set in Progress         : Set Complete
Auth Type Support       : NONE MD2 MD5 PASSWORD
Auth Type Enable        : Callback :
                        : User     :
                        : Operator :
                        : Admin    :
                        : OEM      :
IP Address Source       : Unspecified
IP Address              : 10.90.90.14
Subnet Mask             : 255.255.255.0
MAC Address             : 00:0e:0c:98:55:6e
SNMP Community String   :
IP Header               : TTL=0x40 Flags=0x40 Precedence=0x00 TOS=0x10
BMC ARP Control         : ARP Responses Disabled, Gratuitous ARP Disabled
Gratituous ARP Intrvl   : 2.0 seconds
Default Gateway IP      : 0.0.0.0
Default Gateway MAC     : 00:00:00:00:00:00
Backup Gateway IP       : 0.0.0.0
Backup Gateway MAC      : 00:00:00:00:00:00
RMCP+ Cipher Suites     : None
Cipher Suite Priv Max   : XXXXXXXXXXXXXXX
                        :     X=Cipher Suite Unused
                        :     c=CALLBACK
                        :     u=USER
                        :     o=OPERATOR
                        :     a=ADMIN
```

9.8.2.5 Ending an SOL Session

To use the front panel serial console port on the target blade, end the SOL session.

To end the SOL session:

1. Type: **~.** (Note: After pressing ~ just once, this symbol does not appear on the console screen. This is done at the login prompt.)
2. After typing **~** the SOL session deactivates on the SOL target blade. To make sure the target blade SOL session is deactivated, type the following command:

```
# ipmitool -I lanplus -L operator -H <SOL Target IP Addr>
    -U solusername -P soluserpassword sol deactivate
```

SOL is stopped completely and the serial console port is now redirected to the front panel of the SOL target blade.

9.8.2.6 Recovering SOL Session if ipmitool Segfaults

If ipmitool segfaults while in a SOL session and another SOL session is immediately activated, ipmitool will not be able to establish a new session because no SOL close session occurred. Ipmitool is still receiving encrypted packets from the previous SOL session and is expecting to receive encrypted packets that are responses to the new SOL session establishment commands.

There are two ways to recover:

- Wait for 60 seconds + 3 seconds (approximately) before establishing new SOL session. The IPMC firmware takes 60 seconds (session timeout) to clear the session.
- Using ipmitool, do an explicit Deactivate and then Activate a new SOL session:

Deactivate the SOL session:

```
# ipmitool -I lanplus -L operator -H <SOL Target IP Addr>
    -U solusername -P soluserpassword sol deactivate
```

Reactivate the SOL session:

```
# ipmitool -I lanplus -L operator -H <SOL Target IP Addr>
    -U solusername -P soluserpassword sol activate
```

9.9 Operating Systems for SOL Client (ipmitool)

The SOL client utility (ipmitool) can be compiled to work with any Linux* OS.

10.0 Firmware Update Utilities

10.1 Firmware Locations

The current firmware release bundle can be found on the following web page http://www.intel.com/design/telecom/products/cbp/atca/mpcbl0050/software.htm

10.2 EFI BIOS Image Updates

At times, new EFI BIOS images will be released to add features or fix issues.

- The EFI BIOS image needs to be executed locally on-board.
- The EFI BIOS image update utilities will only work under Linux.
- If doing updates remotely, transfer update utility and the MPCBL0050_EFI_BIOS_XXXXXX.CAP file to the board (e.g. using FTP) and use telnet to login to execute the EFI BIOS updates.

Note: Ensure that binary mode is used when transferring the MPCBL0050_EFI_BIOS_XXXXXX.CAP file using FTP.

10.2.1 Updating EFI BIOS under Linux* (Interactive mode)

This method requires operator/human intervention. The following is a step-by-step procedure to update the EFI BIOS under Linux.

1. Copy the flash utility (flashlnx) and the MPCBL0050_EFI_BIOS_XXXXXX.CAP file to the SBC (via FTP/etc.). Ensure both the flash utility and MPCBL0050_EFI_BIOS_XXXXXX.CAP file are in the same directory.
2. Issue the command: *./flashlnx -b MPCBL0050_EFI_BIOS_XXXXXX.CAP*
3. Enter "Y" to overwrite the EFI BIOS on the board.
4. Enter "Y" to clear the current CMOS settings on the board (if desired)
5. Enter "Y" to reboot the system after the EFI BIOS has been upgraded successfully.

10.2.2 Updating EFI BIOS under Linux* (Quiet Mode)

The EFI BIOS update utility supports quiet mode, where messages are not sent to the screen. This mode can be used for automatic (programmatic) invocations of the update utility.

To update EFI BIOS automatically without human intervention or responses to prompts, execute the following command:

./flashlnx -q -b MPCBL0050_EFI_BIOS_XXXXXX.CAP".

10.2.3 Synchronizing EFI BIOS Image and Settings from FWH0 (Main) to FWH1 (Backup)

Prior to upgrading the main EFI BIOS (FWH0), a user can create a mirror image where it will copy all the operational codes and CMOS settings to redundant EFI BIOS Flash device. It is suggested the user preserves a copy of the old EFI BIOS image prior to updating the main EFI BIOS in case FWH0 update fails.

The syntax "./flashlnx –m" can be used to initiate this transfer. Refer the suggested method in Table 71.

Table 71. Suggested EFI BIOS image synchronization method prior to EFI BIOS upgrade

EFI BIOS Image	Command	Behavior
FWH0 Image N FWH1 Image N-1		- This is the original FWH images before an upgrade. - FWH0 has Image N installed, which is a newer image than what is installed in FWH1, which is Image N-1.
FWH0 Image N FWH1 Image N	./flashlnx –m	User can initiate a EFI BIOS update while the OS is running. When this command is executed, the Image N in FWH0 (EFI BIOS codes + CMOS settings) is synchronized to FWH1. Image N has now been copied to the backup FWH1 EFI BIOS image No reboot is needed for this operation.
FWH0 Image N+1 FWH1 Image N	./flashlnx –b MPCBL0050_EFI_BIOS_XXXXXX.CAP	When this command is initiated, the FWH0 image will be updated to the latest version (Image N+1). The latest version of the EFI BIOS will take effect after the user initiates a reset. If a checksum error is detected on FWH0 after a reboot, it will automatically switch to FWH1 and regain normal operation.

Note: N = EFI BIOS version

10.2.4 Copying and Saving EFI BIOS (Including CMOS Settings)

The EFI BIOS settings (CMOS), together with the EFI BIOS binary image, can be copied to a file with a file name specified by the user. This feature is useful if you wish to use one standard configuration of EFI BIOS settings across multiple MPCBL0050 boards.

10.2.4.1 Copying BIOS.bin from the SBC

1. Copy the flashlnx utility to the SBC. This SBC is the one with custom BIOS options that will be used to update other SBCs.

2. Issue the `flashlnx -r -affe00000 -s2097152 BIOS.cap` command to copy BIOS with the customized settings to the same directory from which the flashlnx utility is executed. All user preferred settings (including the BIOS image) are saved in the file named BIOS.bin.

Note: BIOS.bin is a generic file name used here to illustrate the command line used to perform the operation. You may wish to use a filename that reflects the BIOS version (for example, EFI_BIOS_XXXXXX.bin) instead of BIOS.cap.

10.2.4.2 Saving BIOS.bin to the SBC

1. Copy the flashlnx utility and `BIOS.cap` to the SBC Linux.
2. Execute `chmod +x flashlnx` to change the file attribute to an executable form.
3. Execute `./flashlnx -b -zc BIOS.cap` to load the `BIOS.cap` file to the FWH0.
4. Upon completion, perform a reset to ensure that the new settings and BIOS are loaded.

Caution: To ensure that the BIOS.bin file is not corrupted, Intel strongly suggests performing these steps before major deployment of any SBCs running in a live network environment.

10.2.5 EFI BIOS Update Utility Command Line Options

Table 72 lists the command line parameter switches and features supported by the BIOS flash utility.

Table 72. Flashdos utility command line options

Command Line Parameter	Description
-b [option] bios_image where possible [option] values are: -z : do not clear the CMOS -zc : update the CMOS from image	Program a EFI BIOS image to primary fimware hub (FWH0)
-i [bios_image]	Display EFI BIOS system information
-r [options] bin_image where possible [option] values are: -aAddress :physical address in hex -pPage :page number in decimal -sSize :image size in decimal	Read the flash image and store to a file
-m	Mirror image where all the operational codes and CMOS settings are copied from FWH0 to FHW1 (redundant BIOS flash device)
-bc cmos_image	Backup current CMOS settings to a file
-rc cmos_image	Restore CMOS settings from a file
-q	Force non-interactive mode (assumes “yes” for all prompts)
Note: The command line parameters shown reflect flashdos usage, where command line options use the “/” symbol. When using flashlnx, replace “/” with “-”.	

10.2.6 EFI BIOS Update Utility error codes

Table 73 lists error codes flashlnx utility may return in case of update failure.

Table 73. FlashInx error code list (Sheet 1 of 3)

Error Number	Message Label	Message Text
0	MSG_NONE	"No error"
-1000	MSG_INVALID_ARG_COUNT	"Invalid argument count"
-999	MSG_INVALID_ARGUMENT	"Invalid argument"
-999	MSG_INVALID_SWITCH	"Invalid switch"
-998	MSG_FILE_OPEN	"Failed to open file '%s'"
-998	MSG_MAP_PHYSICAL_MEMORY	"Failed to map physical memory"
-997	MSG_BSIS_NOT_FOUND	"System Information Structure not found in BIOS"
-997	MSG_FSIS_NOT_FOUND	"System Information Structure not found in file"
-996	MSG_UNSUPPORTED_FLASH	"Unsupported flash programming algorithm"
-996	MSG_UNSUPPORTED_MECHANISM	"Unsupported flash mechanism"
-995	MSG_IOPERM_OPEN	"No permission to open port"
-995	MSG_NOT_ENOUGH_MEMORY	"Not enough memory"
-994	MSG_VIRT_TO_PHYS	"Failed to translate virtual address"
-994	MSG_INVALID_LOADADDR	"Invalid load address"
-993	MSG_INVALID_ENTRYPT	"Invalid entry point"
-993	MSG_INVALID_COPYSIZE	"Invalid copy size"
-992	MSG_ERASE_FLASH	"Error encountered while erasing flash"
-992	MSG_PROGRAM_FLASH	"Error encountered while programming flash"
-991	MSG_READ_FLASH	"Error encountered while reading flash"
-991	MSG_USER_ABORTED	"User aborted"
-990	MSG_INVALID_ADDR	"Invalid address"
-990	MSG_INVALID_PAGE	"Invalid page number"
-989	MSG_INVALID_SIZE	"Invalid size"
-989	MSG_INVALID_INDEX	"Invalid binary index"
-988	MSG_NOT_ALIGN_TO_PARA	"Entry point does not align to paragraph"
-988	MSG_OS_OVERLAPS_BIOS	"OS image overlaps with BIOS image"
-987	MSG_REQUIRE_VER103	"System Information Structure 1.03+ required"

Table 73. Flashlnx error code list (Sheet 2 of 3)

Error Number	Message Label	Message Text
-987	MSG_UNDEFINED_UB	"Undefined BIOS-defined binary"
-986	MSG_UB_OVERLAPS_OS	"Binary image overlaps with OS image"
-986	MSG_UB_OVERLAPS_BIOS	"Binary image overlaps with BIOS image"
-985	MSG_UNKNOWN	"Unknown error (%d)"
-985	MSG_INVALID_FILE	"This file is for a %s"
-984	MSG_ROOT_ACCESS	"No port access, please run as 'root'"
-984	MSG_OPEN_SCMAN	"No access rights to Service Control Manager"
-983	MSG_OPEN_SERVICE	"No access rights to Service Database"
-983	MSG_INVALID_SERVICE	"Invalid service name"
-982	MSG_START_DRIVER	"Driver could not be started"
-982	MSG_COPY_DRIVER	"Driver could not be installed"
-981	MSG_NOT_AMIBIOS	"AMIBIOS signature not found"
-981	MSG_INVALID_CHKSUM	"Invalid BIOS checksum"
-980	MSG_MODULE_HEADER	"Module header not found"
-980	MSG_LINK_PRESENT	"Cannot update a module with link present"
-979	MSG_MODULE_SPACE	"No space to accommodate new module"
-979	MSG_MODULE_NOT_FOUND	"Module not found"
-978	MSG_LOGO_NOT_SUPPORTED	"OEM logo is not supported by the BIOS"
-978	MSG_CMOS_CHECKSUM_BAD	"CMOS buffer has bad checksum"
-977	MSG_CBR_NOT_SUPPORTED	"CMOS Backup/Restore is not supported"
-977	MSG_DATAID_NOT_FOUND	"Data ID not found"
-976	MSG_POINTER_IS_NULL	"Pointer is NULL"
-976	MSG_RESOURCE_NOT_FOUND	"Resource (%s) not found"
-975	MSG_BIOSBIN_SIZE_NOT_MATCH	"BIOS binary size does not match"
-975	MSG_ISIS_INVALID_SIZE	"Invalid ISIS size"
-974	MSG_ISIS_INVALID_CHECKSUM	"Invalid ISIS checksum"
-974	MSG_UNRECOGNIZED_SYSID	"Unrecognized sysid (0x%x)"
-973	MSG_DIAG_NOT_FOUND	"DIAG header not found"

Table 73. FlashInx error code list (Sheet 3 of 3)

Error Number	Message Label	Message Text
-973	MSG_INSUFFICIENT_SPACE	"Insufficient space to fit image"
-972	MSG_GDT_NOT_FOUND	"DIAG image doesn't contain GDT"
-972	MSG_GDT_INVALID_SIZE	"Invalid GDT size"

10.3 Updating IPMC Firmware

The following subsections describe step-by-step procedures for updating the IPMC boot block and IPMC operational code.

IPMC firmware updates are divided into two phases:

- Boot block update - not typically needed
- Operational code update - most common form of update

The procedure for updating the boot block is in the following section, "Updating IPMC Boot Block".

The procedures for updating the operational code are located in Section 10.6.7 and Section 10.6.8.

10.3.1 Updating IPMC Boot Block

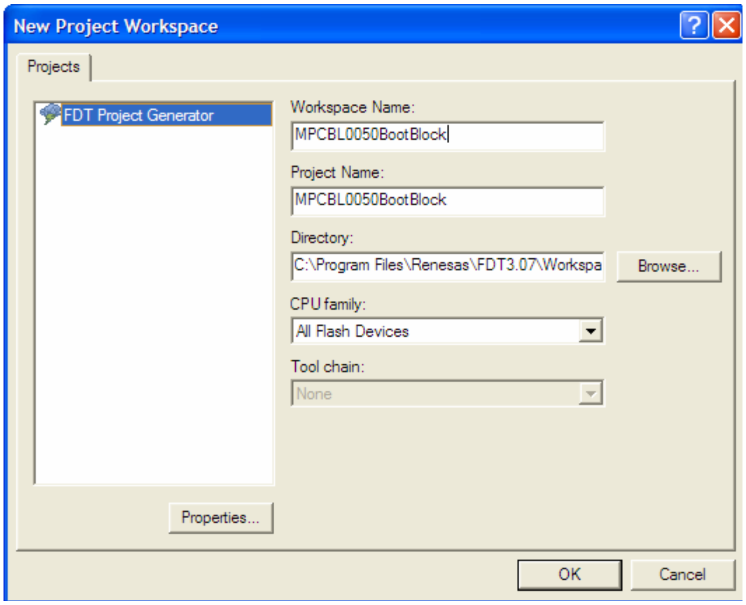
It is recommended that the IPMC Boot Block update is done on a computer using a COM port. It is possible to update the boot block from a computer with a USB to serial adapter, but some USB to serial adapters have found to be less reliable and using this USB to serial adapter may cause some intermittent communication or timeout errors when doing the update.

1. Update the IPMC firmware using the staged firmware update mode. Refer to Section 10.6.8, "Staged Firmware Update" on page 203. This step is performed first to ensure a valid IPMC operational code is saved into the into the rollback area. Once the boot block is updated, the IPMC operational code is erased and the next time the board boots, the IPMC operational code that is in the rollback area will be automatically copied to the IPMC and the board will boot.
2. On the MPCBL0050 board, change the DIP switch positions as follows:
   a. Set SW1.9 to ON (COM port to IPMC)
   b. Set SW3.4 to ON (enable IPMC serial update mode)

Note: All other DIP switches should be in the OFF state.

3. Reinsert the blade into the chassis.
4. Connect a serial cable to the front panel serial port.
5. Start the Renesas* Flash Development Toolkit. The program to download is the latest version of the "[Evaluation Software] Flash Development Toolkit V.X.XX". This Renesas toolkit can be downloaded from: http://www.renesas.com/fmwk.jsp?cnt=/download_search_results.jsp&fp=/support/downloads/download_results&layerId=1050
6. Create a new project workspace named "MPCBL0050BootBlock". See Figure 41.

Figure 41. Creating a new project workspace



New Project Workspace

Workspace Name: MPCBL0050BootBlock

Project Name: MPCBL0050BootBlock

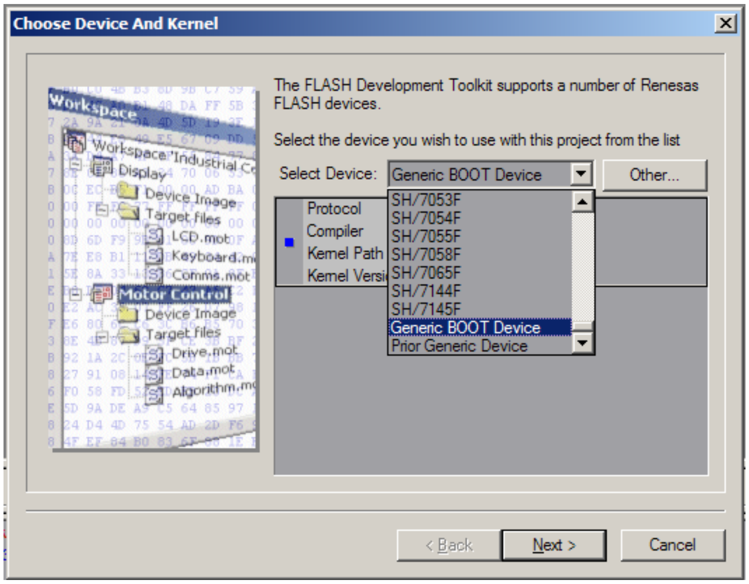
Directory: C:\Program Files\Renesas\FDT3.07\Workspa

CPU family: All Flash Devices

Tool chain: None

a. Select "Generic Boot Device" at the bottom of the pull-down menu. See Figure 42.

Figure 42. Selecting a boot device



Choose Device And Kernel

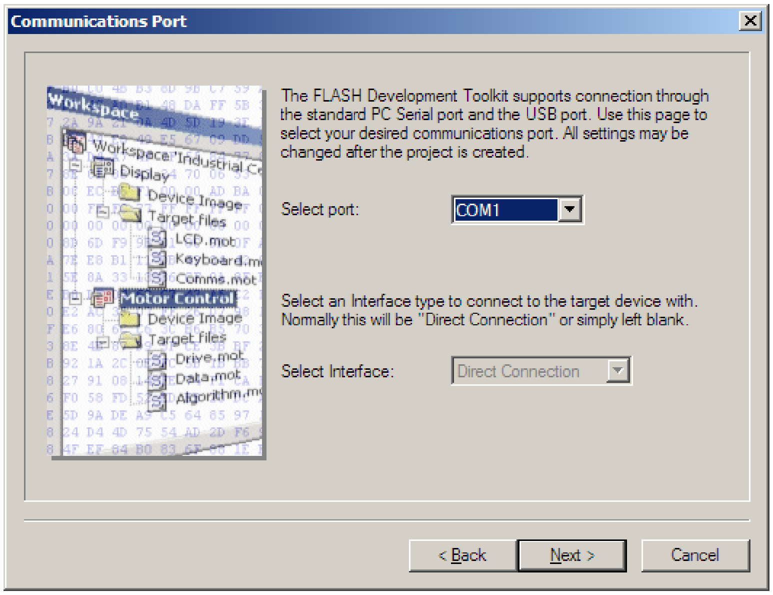
The FLASH Development Toolkit supports a number of Renesas FLASH devices.

Select the device you wish to use with this project from the list

Select Device: Generic BOOT Device

b. On the next screen, select the **COM** port connected to the board. See Figure 43.

Figure 43. Selecting a COM port



Communications Port

The FLASH Development Toolkit supports connection through the standard PC Serial port and the USB port. Use this page to select your desired communications port. All settings may be changed after the project is created.

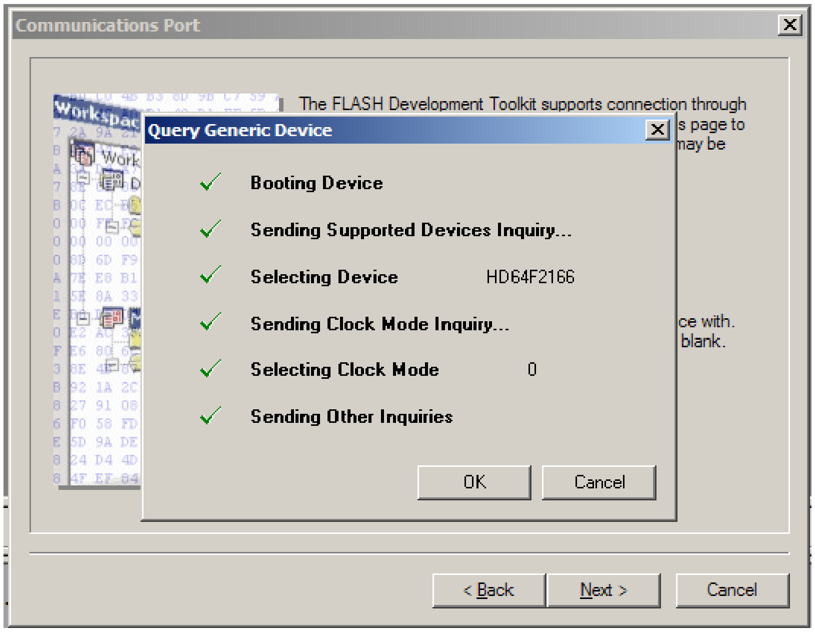
Select port: COM1

Select an Interface type to connect to the target device with. Normally this will be "Direct Connection" or simply left blank.

Select Interface: Direct Connection

c. After selecting a COM port, Renesas* attempts to query the device. The following dialog is displayed. See Figure 44.

Figure 44. Query device



Query Generic Device

✓ Booting Device

✓ Sending Supported Devices Inquiry...

✓ Selecting Device HD64F2166

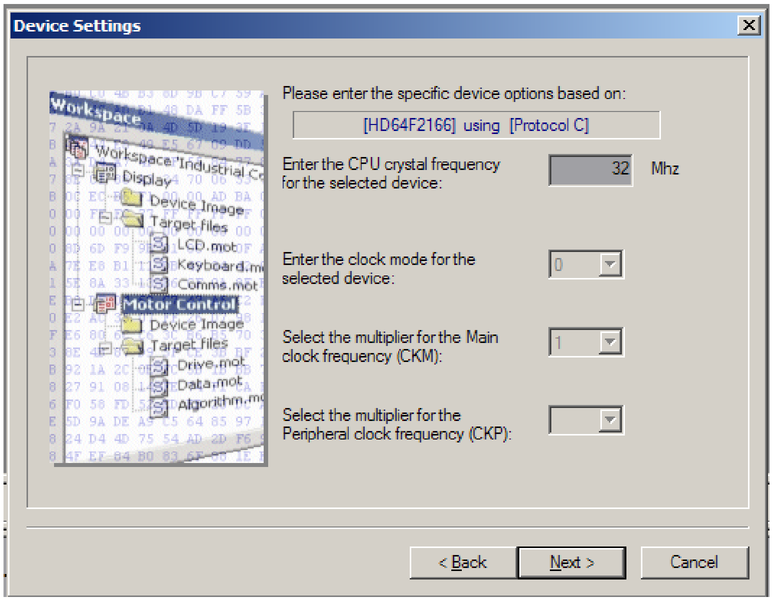
✓ Sending Clock Mode Inquiry...

✓ Selecting Clock Mode 0

✓ Sending Other Inquiries

d. In the **Device Settings** window, change the CPU crystal frequency to 32 MHz. See Figure 45.

Figure 45. Changing CPU crystal frequency



Device Settings

Please enter the specific device options based on:

[HD64F2166] using [Protocol C]

Enter the CPU crystal frequency for the selected device: 32 Mhz

Enter the clock mode for the selected device: 0

Select the multiplier for the Main clock frequency (CKM): 1

Select the multiplier for the Peripheral clock frequency (CKP):

e. Use the defaults for the remaining device settings.

f. Click **Next>** until you reach the end of the section.

7. Under the **Project** menu, right-click the project name and choose **Add Files**.
8. Add the file MPCBL0050_BB_XXX.mot (IPMC bootloader)
9. Right-click the MPCBL0050_BB_XXX.mot filename and choose **User Boot Flash**.

Warning: Make sure that you have done step 9, and selected the User BootFlash option. The common mistake is not to check this option which leads to IPMC image corruption,

10. Right-click the MPCBL0050_BB_XXX.mot filename again and choose **Download File to [User Boot Flash]**.

    The following text (or similar) should appear at the bottom of the screen after successful completion of the update procedure:

```
Loaded the Write operation module
Writing image to device... [0x00000000 - 0x000009FF]
Writing image to device... [0x00001F80 - 0x00001FFF]
Data programmed at the following positions:
 0x00000000 - 0x000009FF      Length : 0x00000A00
 0x00001F80 - 0x00001FFF      Length : 0x00000080
2.73 K programmed in 1 seconds
Image successfully written to device
```

11. After the boot block updates successfully, confirm matching checksums by right clicking on the filename “MPCBL0050_BB_XXX.mot” , then “Compare File-> Device Checksum”. Verify the checksum of the file matches the checksum of the device.
12. Exit the program.
13. Eject blade or power down (including standby power) the SBC.
14. Change the DIP switches to default position - ALL OFF

15. Reinsert the blade into the chassis.
16. If there is a IPMC operational code in the rollback area, this will be copied automatically to the IPMC and the board will boot. Once the board boots, then update the IPMC firmware using the staged firmware update mode. Refer to Section 10.6.8, "Staged Firmware Update" on page 203. The rest of the steps in this procedure do not need to be followed.
17. If the board doesn't boot, there was not valid IPMC operational code image in the rollback area. If this happens, press and release the front panel reset button. This will force the board payload power to turn on. Then follow the remaining steps in this section.

Note: The board does not boot up until the front panel reset button is pressed and released.

18. Once the board boots, the IPMC operational code must be updated using the direct firmware update mode. Staged firmware update mode does not work the first time after updating the boot block. Refer to Section 10.6.7, "Direct Firmware Update" on page 201.
19. After the direct IPMC firmware update, the board payload power will be cycled and the board will automatically reboot. This reboot only occurs after the boot block is updated followed by an IPMC operational code update. All subsequent IPMC firmware updates do not impact payload power.

10.4 SBC Utilities Installation Procedure

The SBC utilities and the supporting libraries are packaged together and released as a single RPM package manager archive (RPM). This rpm file is included as part of the MPCBL0050 IPMC Update Utilities package. The filename is "MPCBL0050-sbcutils-w.x.y-z.i386-mv40.rpm", where MPCBL0050 refers to the Single Board Computer. This utility is specifically created for this board only.

- *x* refers to the software generation number.
- *y* refers to the Interim Product Release number. This is typically changed when the release includes bug fixes.
- *z* refers to the build number of the software.

10.4.1 Contents of SBC Utilities RPM Package

The SBC utilities rpm includes the following:

- sbcupdate - a binary tool that is used to perform an IPMC firmware upgrade. It also updates FRU and SDR information on the SBC and the RTM.
- reference_cfg - a reference configuration script that contains all the necessary IPMI commands to enable and configure the SBC for SOL functionality on the MPCBL0050SBC.
- reference_funcs - a support file that contains functions used by the reference_cfg script.

Note: The SOL client for the SBC is ipmitool. This tool is not part of the SBC utilities RPM package, but is discussed in this chapter.

10.4.2 Remove Existing Packages

Before installing the RPM on a host system (Single Board Computer), verify that a different version of the sbcutils RPM is not currently installed.

To check whether a package has been installed on the system before, execute this command:

```
    # rpm -q sbcutils
```

Either the rpm tool returns the version of the installed package or it reports that no package is installed.

If a version of the sbcutils package is already installed, the user must remove it before attempting to install the latest version. If the user attempts to install the latest version of the sbcutils package before resolving this conflict, the rpm tool will fail with the error message similar to:

```
Error: Failed Dependencies

sbcutils conflicts with sbcutils-1.1.0.1
```

To remove the existing installation, execute the following command:

```
    # rpm -e sbcutils
```

10.4.3 Installing the sbcutils Package

Once older packages have been removed from the host system, execute the following command to install the sbcutils package:

```
    # rpm -ivh sbcutils-X.X.X-X.i386-mv40.rpm


Preparing...        ########################################### [100%]
   1:sbcutils       ########################################### [100%]
```

- -i specifies an installation operation.
- -v requests verbose output.
- -h causes hash characters to be displayed to indicate the progress of the installation.

10.5 System Configuration

10.5.1 Introduction

If the sbcupdate utility is to be used with RMCP, it is essential that communication between the target IPMC and a remote network node via a shelf manager can be established.

The shelf manager must bridge IPMI messages between the LAN and IPMB bus. If this feature can be disabled, make sure that bridging is in the enabled state before attempting RMCP bridge communication. Furthermore, if the shelf manager supports filtering and privilege settings for individual IPMI commands, make sure that the IPMI commands listed in Table 74 are not filtered and are set to the indicated privilege levels. Table 74 is the list of commands that must be FORWARDED from the LAN to the IPMB.

Table 74. Required IPMI command privileges for RMCP bridging (Sheet 1 of 2)

IPMI Command	Net Function	Command	Privilege Required
Get Device ID	06h	01h	User
Cold Reset	06h	02h	Admin
Set Channel Access	06h	40h	Admin
Get Channel Access	06h	41h	User
Set User Name	06h	45h	Admin
Set User Access	06h	43h	Admin
Set User Password	06h	47h	Admin
Set User Payload Access	06h	4Ch	Admin
Enter Firmware Transfer Mode	08h	00h	Admin
Firmware Program	08h	01h	Admin
Firmware Read	08h	02h	Admin
Exit Firmware Transfer Mode	08h	04h	Admin
Set Program Segment	08h	05h	Admin
Get FRU Inventory Information	0Ah	10h	User
Read FRU Data	0Ah	11h	User
Write FRU Data	0Ah	12h	Operator
Get SDR Repository Info	0Ah	20h	User
Reserve SDR Repository	0Ah	22h	User
Get SDR	0Ah	23h	User
Partial Add SDR	0Ah	25h	Operator
Clear SDR Repository	0Ah	27h	Operator
Set LAN Configuration Parameters	0Ch	01h	Admin
Set Serial/Modem configuration	0Ch	10h	Admin
Set SOL Configuration Parameters	0Ch	21h	Admin
Get PICMG Properties	2Ch	00h	User
Get Address Info	2Ch	01h	User
Get Serial Buffer [1]	30h	30h	User
Set Serial Buffer Configuration [2]	30h	32h	User
Online Update Prepare For Update	30h	A0h	Admin
Online Update Open Area	30h	A1h	Admin
Online Update Write Area	30h	A2h	Admin
Online Update Close Area	30h	A3h	Admin
Online Update Register Update	30h	A4h	Admin
Online Update Capture Rollback Image	30h	A5h	Admin

Table 74. Required IPMI command privileges for RMCP bridging (Sheet 2 of 2)

IPMI Command	Net Function	Command	Privilege Required
Online Update Get Status	30h	A6h	Admin
Online Update Get Capabilities	30h	A7h	Admin
Notes: 1. This command is needed only when using the reference_sbr script. Not all SBCs support this feature. 2. This command is needed only if Serial Buffer Reading configuration is enabled in the reference_cfg script. Not all SBCs support this feature.			

10.5.2 Configuring the Intel NetStructure® MPCMM0001/0002 CMM

This section describes the procedure to configure the Intel NetStructure® MPCMM0001/0002 Chassis Management Module (CMM) to support RMCP bridging.

The CMM filters individual IPMI commands for bridging based on the /etc/cmdPrivillege.ini CMM configuration file. (Note that the double "l" in the filename is spelled this way on purpose.) Each combination of net function and command number can be given a custom privilege level. Each combination can also be disabled, in which case the command is not bridged between the LAN and IPMB. This is an example cmdPrivillege.ini file, which can be modified if needed. Even without modifications, the file is working and can be used as is.

Copy cmdPrivillege.ini File

1. After the sbcutils installation, copy the cmdPrivillege.ini file from the sbcutils installation path (/usr/share/doc/sbcutils) directory to the CMM's /etc path. Copy the cmdPrivillege.ini file to both the active and standby CMMs.
2. After any change to this configuration file, both CMM RMCP server's must be restarted to enable the changes to take effect. To stop and then start the RMCP server, execute the following commands in sequence. Note, this must be done on both CMMs at the same time (within 10 to 15 seconds of each other):

```
# cmmset -d rmcpenable -v 0
Success
# cmmset -d rmcpenable -v 1
Success
```

Note: On some versions of the CMM firmware, enabling the RMCP server results in an error for the first 10 to 15 attempts before the server is successfully restarted.

```
# cmmset -d rmcpenable -v 1
IMB ERROR Completion Code Error.
# cmmset -d rmcpenable -v 1
IMB ERROR Completion Code Error.
# cmmset -d rmcpenable -v 1
IMB ERROR Completion Code Error.
# cmmset -d rmcpenable -v 1
IMB ERROR Completion Code Error.
# cmmset -d rmcpenable -v 1
IMB ERROR Completion Code Error.
# cmmset -d rmcpenable -v 1
IMB ERROR Completion Code Error.
# cmmset -d rmcpenable -v 1
IMB ERROR Completion Code Error.
# cmmset -d rmcpenable -v 1
Success
```

3. To verify the RMCP enable state, use the following command. A "0" indicates disabled and "1" indicates enabled.

```
# cmmget -d rmcpenable
1
```

Details of cmdPrivillege.ini File

The first section in the cmdPrivillege.ini file lists the net functions to be configured and gives each net function value a mnemonic string equivalent.
For example:

```
[IPMI]
NumNetFunc=25
NetFunc00=Chassis
NetFunc02=Bridge
NetFunc04=S_E
NetFunc06=App
NetFunc08=Firmware
NetFunc0A=Storage
NetFunc0C=Transport
NetFunc2C=PICMG
NetFunc30=Platform
```

The “NumNetFunc” field should be set to the value of the largest net function (30h hexadecimal or 48 decimal in the above example), divided by two, plus 1. This field must be set with a decimal number.

Each subsequent section of this file starts with a net function string equivalent, followed by the number of commands in the section, followed by an ordered, zero-based list of command numbers and privilege levels. For example, the section that configures commands of net function 0Ch starts this way:

```
[Transport]
NumCMD=35
00=D
01=A
02=O
03=D
04=D
```

The “NumCMD” field should be set to the highest command number of the section plus one and written in decimal format. Each command number is assigned a one character code to indicate the desired privilege level and enabled state. Any command number that is not a defined command should be disabled.

The character codes used in the cmdPrivillege.ini file are as follows:

- D - Disabled
- U - Enabled with user privilege
- O - Enabled with operator privilege
- A - Enabled with administrator privilege

10.6 SBC Update Utility

This section is an overview of the SBC Update tool used to update the IPMC firmware on Single Board Computer. This tool also can update FRU and SDR information on the SBC and the RTM.

10.6.1 Communication Interfaces

Graphical examples of the communication interfaces used for firmware updates are provided in this section.

The term "payload" is used in following graphics and refers to the CPU(s) of a SBC that perform user-specific computational tasks.

Figure 46. KCS - local communication from CPU to local IPMC

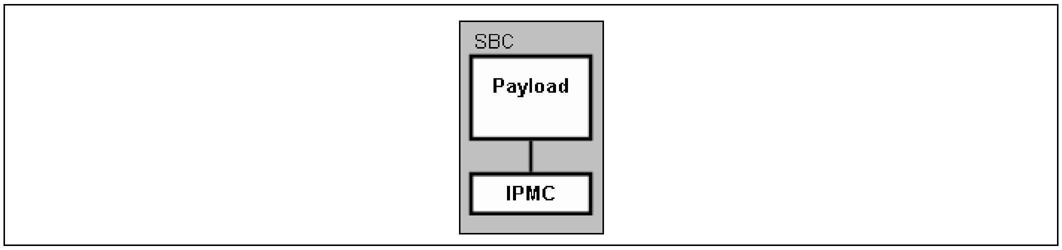


Figure 47. KCS bridge - local communication from CPU to local IPMC bridged to RTM

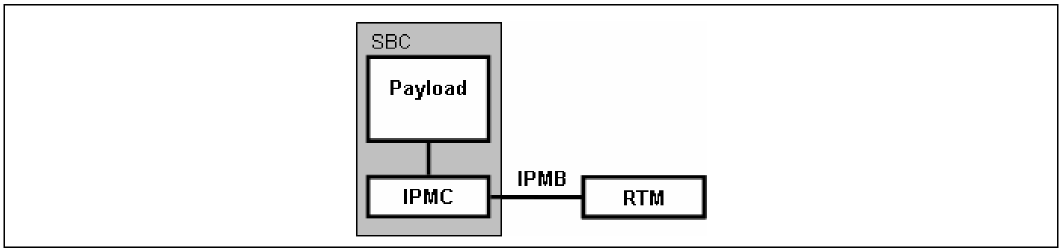


Figure 48. RMCP bridge - remote communication to shelf manager bridged to IPMC

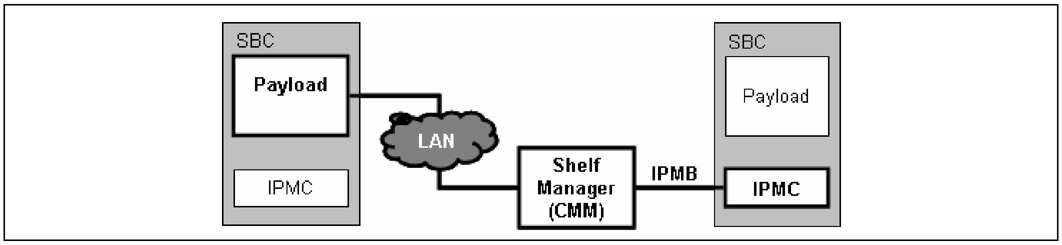


Figure 49. RMCP double bridge - remote communication to shelf manager bridged to IPMC, then bridged to the RTM

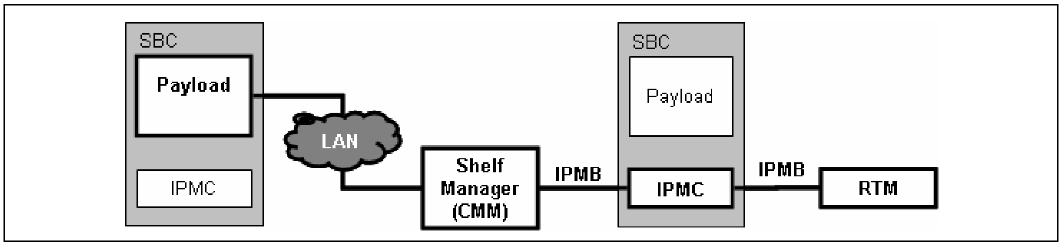


Figure 50. IPMB - remote communication from shelf manager to IPMC

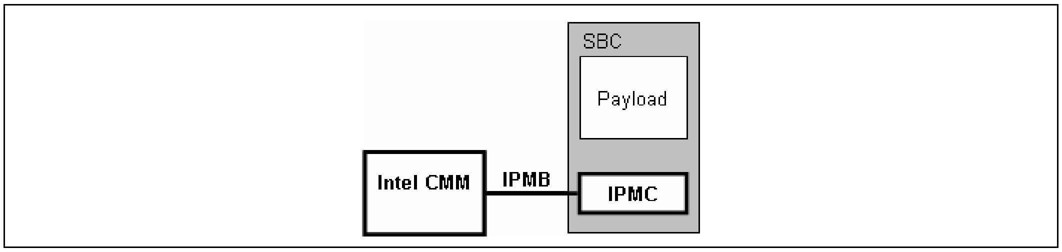
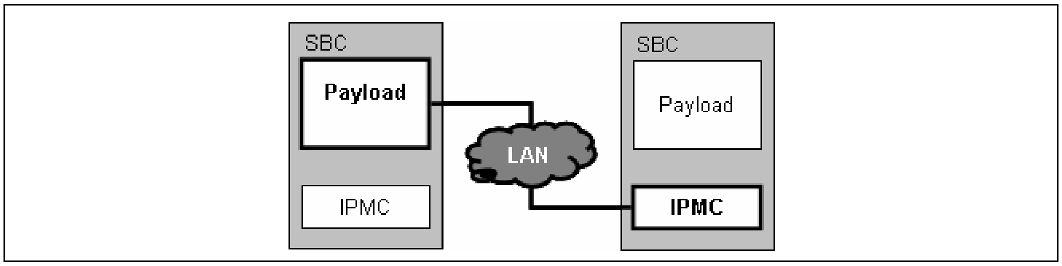


Figure 51. RPMC + direct - remote communication to IPMC directly through LAN interface



10.6.2 IPMC Operational Code Firmware Update Modes

There are two update modes:

- **Direct Update Mode** - This upgrade procedure updates IPMC firmware directly to the internal flash on the IPMC. The user does not have the ability to back up the “old” operational image.
- **Staged Update Mode** - The Staged Update feature allows the IPMC operational code to be updated while the system is online (OS is running). After saving copy of currently running IPMC image, the new image is loaded to stage area. Upon completion of the upgrade teh image is copied to the internal flash of the IPMC. If the switchover fails, or at the user's discretion, the firmware may be rolled back to the previous version.

  The advantage of running this mode is that the user can store the old firmware image to a rollback region (for redundancy purposes). A pending firmware update may also be created allowing the user to continue to use the current firmware before the actual update.

The following section provides an overview of the staged update process.

Warning: IPMC firmware image delivered in the firmware bundle also contains the image of onboard FPGA. If during the update procedure IPMC detects that newly loaded IPMC firmware image contains newer FPGA version then the one running on the board, the new FPGA will be loaded. Due to FPGA serving major functionalities on the payload the board will be automatically rebooted in order to assure proper board operation.

10.6.3 Staged Update Process

Table 75 explains the process of an online staged update. The diagrams in the table are color-coded to show old, new, or no firmware images, as indicated.

Table 75. Staged update process (Sheet 1 of 2)

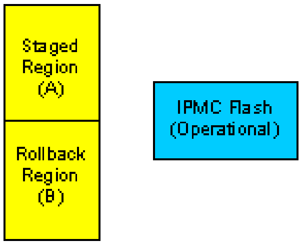
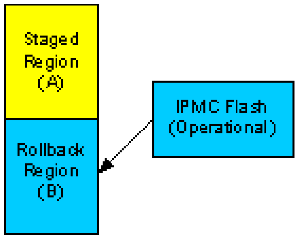
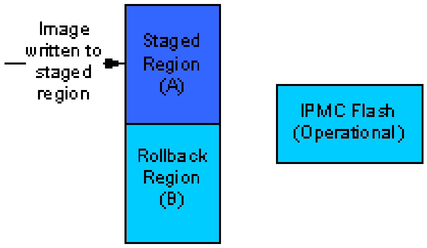
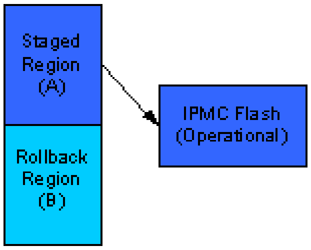
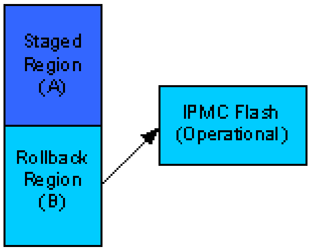
Scenario	Behavior	Actions
Fresh from production/ factory	 Staged Region (A) Rollback Region (B) IPMC Flash (Operational)	This is the original firmware composition when a board is shipped from a factory. Both external flash partitions are empty.
Initiating online staged update (step 1)	 Staged Region (A) Rollback Region (B) IPMC Flash (Operational)	When a staged firmware update is initiated, depending upon user specified command options, the sbcupdate utility first copies the current operational image into the rollback region(B).
Copying new image to the staged region (step 2)	 Image written to staged region Staged Region (A) Rollback Region (B) IPMC Flash (Operational)	The new firmware image is written to Staged Region (A) of the external flash. Note that the IPMC continues to operate with the original operational firmware image, even after the online update has been successfully staged. The IPMC needs to be reset for the new firmware image to be transferred to the IPMC from Staged Region (A).
Legend: 🟨 No Image 🟦 Old Image 🟦 New Image		

Table 75. Staged update process (Sheet 2 of 2)

Scenario	Behavior	Actions
Activating the staged image (step 3)	 Staged Region (A) Rollback Region (B) IPMC Flash (Operational)	Upon completion of the staged update, the new firmware image from staged region is written to operational image (internal flash of IPMC).
Initiating a rollback (optional)	 Staged Region (A) Rollback Region (B) IPMC Flash (Operational)	User can initiate a rollback by invoking the rollback option with SBC update utility. Upon IPMC reset, an automatic rollback recovery can occur if the IPMC boot block detects incomplete update or checksum error with the IPMC firmware image. User can initiate a rollback by invoking the rollback option with SBC update utility. If the force rollback option is used and there isn't a valid rollback image, the IPMC will enter direct firmware update mode.
Legend: No Image Old Image New Image		

10.6.4 Firmware Naming Scheme

Firmware for both the IPMC of the MPCBL0050 SBC and the RTM has a standard naming scheme. The letter "N" is given with the file name which refers to the version of the update. In all instances, replace the sequence of "N"s with the number of your update file.

IPMC naming:

- MPCBL0050_FW_NNNNNN.hex - IPMC firmware update (this image includes also FPGA image automatically loaded if older version is found on the board)
- MPCBL0050_NNN.sdr - SDR update
- MPCBL0050_NNN.fru - FRU update

RTM naming:

- MPRTM0050_FW_NNNNNN.hex - RTM firmware update
- MPRTM0050_NNN.sdr - RTM SDR update
- MPRTM0050_NNN.fru - RTM FRU update

For example, here is the IPMC firmware update for version 1.02.00:

- MPCBL0050_010200.hex

Here is an example IPMC FRU update for version 1.01:

- MPCBL0050_101.fru

10.6.5 Utility Invocation

The general format for utility invocation is:

```
sbcupdate [<options>] [<input-files>]
```

Each option has a long-form name, which consists of two dashes followed by a multi-character string of letters, numbers, and dashes. For long-form options that take arguments, the argument values are separated from the option name by one or more spaces or the "=" operator.

For example, this command uses the long form of an option with no arguments:

```
# sbcupdate --mode
```

The following command uses the long form of an option that takes an argument:

```
# sbcupdate --mode staged
```

or

```
# sbcupdate --mode=staged
```

Many options also have an alternate short form name, which follows the POSIX conventions for command-line options, namely, a single dash followed by a single letter. For short-form options that take arguments, the argument values are either concatenated to the end of the option name or separated by one or more spaces.

For example, this command uses the short version of an option with no arguments:

```
# sbcupdate -M
```

This command uses the short version of an option that takes an argument:

```
# sbcupdate -M staged
```

This command uses the short version without a space separating the option and the argument:

```
# sbcupdate -Mstaged
```

Throughout the chapter, the short form syntax is used. Any short form option may be substituted with the long form. If the long form is preferred, please refer to Table 76, "Common command line options and arguments" on page 198.

After any options and option arguments supplied on the command line, any remaining strings are interpreted as input files. Tools that accept multiple file formats as input determine the format of each file based on the file extension used in the file name. The file extensions recognized are:

- .fru—ASCII FRU file
- .bfru—binary FRU file
- .sdr—ASCII SDR file
- .bsdr—binary SDR file
- .hex—IPMC firmware file
- .cfg—configuration metadata file

Most options are common to all utilities. These options and arguments are described in Table 76.

Note: All commands and options are case sensitive.

Table 76. Common command line options and arguments (Sheet 1 of 2)

Short Option Name	Long Option Name	Possible Argument	Description
-h | -?	--help	none	Output a text description of each option.
none	--usage	none	A usage message displaying options by their short and long names.
-V	--version	none	Output the version number.
-I	--interface	kcs | lan | ipmb	Interface type. • kcs denotes the local KCS interface on the SBC. Use this interface only if the executable is being run on the SBC and targets the IPMC on the same SBC. • lan denotes IPMI over LAN using RMCP. Use this interface only if the executable is being run on a remote node with a LAN connection to a shelf manager that is configured with an RMCP bridge. • ipmb denotes IPMI over the IPMB bus. Use this interface only if the executable is being run on the shelf manager (Intel NetStructure® MPCMM0001/002 Chassis Management Module). If omitted, kcs is assumed.
-U	--user	*<string>*	User name. May be up to 16 characters. Valid only with lan interface.
-P	--password	*<string>*	User password. May be up to 20 characters. Valid only with lan interface.
-L	--privilege	callback | user | operator | admin	Privilege level. Valid only with lan interface.
-A	--auth	none | md2 | md5 | password	Authorization algorithm type. Valid only with lan interface.
-H	--host	*<ip-address>* | *<hostname>*	IP address or name of the host to update over RMCP.
-M	--mode	auto | staged | direct | cancel | rollback | info | get | get <file>	• auto (default) means that staged is attempted first, and direct is used as a fall back. • staged starts a staged update and does not fall back to a direct update. • direct bypasses the initial check for staged capability and starts immediate update. • cancel stops any pending staged firmware update. • rollback forces a rollback on the next reset. • info requests various pieces of information regarding the version of the SDR and FRU packages, and sends the information to the standard output. • get used with no arguments puts the utility into the mode to retrieve the FRU and SDRs from the targeted device. • get used with a <file> argument displays the contents of the file to the standard output. The file extension must be .fru, .bfru, .sdr, or .bsdr and must contain ASCII FRU data, binary FRU data, ASCII SDR data, or binary SDR data, respectively. • get can be used with the –o option to save FRU or SDR data to a file in binary or ASCII format. See the –o option in this table for more information.

Table 76. Common command line options and arguments (Sheet 2 of 2)

Short Option Name	Long Option Name	Possible Argument	Description
-o	--output	*<file>*	Use with –M get to save the contents of the FRU or the SDR to a file. The name of the file is specified as the argument to -o. The file extension given at the end of the file name determines whether FRU or SDR data is saved and whether the data is stored in ASCII or binary form as follows: *<file>*.fru – Save FRU data in ASCII *<file>*.bfru – Save FRU data in binary *<file>*.sdr – Save SDR data in ASCII *<file>*.bsdr – Save SDR data in binary
-t	--target	**KCS interface** 0x20 | rtm | 0x20:rtm **LAN interface** 0xNN[:rtm] | bladeN[:rtm]	**KCS interface** • 0x20 denotes the local SBC and targets the local IPMC. • rtm or 0x20:rtm denotes the RTM on the local SBC. • Omitting the –t option is equivalent to specifying –t 0x20. **LAN interface** • The value of NN in 0xNN is the hexadecimal IPMB address of the SBC. The IPMC on that SBC is targeted. • The value of N in bladeN can range from 1–14, inclusive determining which SBC to use in the chassis. • The IPMB to physical slot mapping used in the Intel NetStructure® MPCHC0001 chassis is assumed. See Table 79, “Mapping of physical slot to IPMB address in Intel NetStructure® MPCHC0001 14U Shelf” on page 232. The IPMC on the specified SBC is targeted. • If :rtm is appended to either target, the RTM on the specified SBC is targeted.
none	--no-capture-rb	*none*	Applies only to staged firmware updates. This option does not capture a rollback image, thus preserving the contents of the rollback area.
none	--force-id	*none*	Forces the firmware update to occur. This is needed to override product ID or manufacturer ID mismatches. (This happens if the SBC is in fail safe mode.)
none	--force-fw-update	*none*	This option is needed to downgrade the firmware version or if the target is in firmware update mode and version information is not available. This forces the firmware update in these cases.
none	--force-fpga-update	*none*	Sbcupdate will automatically request this option if the version of FPGA currently running on the SBC is different from the version included in the new IPMC FW image that is being updated. Otherwise, this option will not be required. • Note that upgrade of the FPGA automatically initiates payload hard reset.
none	--force-fru-update	*none*	This option is needed to downgrade the FRU version, if the FRU version information is missing from the system or from the input file, or also performing a full FRU update. The option forces the FRU update in these cases.
none	--force-sdr-update	*none*	This option is needed to downgrade the SDR version, if SDR information is missing either from the system or from the input file, or if the target has a pending staged firmware update. This option forces the SDR update in these cases.
none	--no-reset	*none*	Applies only to staged firmware updates. This option stops a reset of the target IPMC from being performed after performing a staged firmware update. This should not affect payload operation.

10.6.6 Automatic Firmware Update

While performing a firmware update via the KCS interface or an RMCP bridge, invoking auto mode will automatically negotiate between a direct or staged firmware update. A staged firmware update will be attempted first, while the direct firmware update is used as a fall back. If the target supports staged updates and it is in a state that allows staged updates, a staged update will be performed. If staged updates are not possible, a direct firmware update will be attempted. Note that auto mode is the default mode, by omitting the mode option (-M), auto is assumed.

10.6.6.1 Automatic Firmware Update using KCS Interface

To perform an auto firmware update over the KCS interface, you must be logged into the target SBC as root.

The syntax of the command is:

```
# sbcupdate -I kcs -M auto <path to firmware image/filename>
```

The following is an example of an auto firmware update via the KCS interface:

```
# sbcupdate -I kcs -M auto MPCBL0050_FW_NNNNNNN.hex
```

The above command line can be abbreviated since the KCS interface and the auto mode are both default options. The command can be abbreviated as follows:

```
# sbcupdate MPCBL0050_FW_NNNNNNN.hex
```

10.6.6.2 Automatic Firmware Update via RMCP Bridging

In order for RMCP bridging to work correctly, the Shelf Manager must be connected to the network in order to transmit TCP/IP traffic to and from your client. You must also specify the interface option to be lan for RMCP bridging. Auto mode is default so by omitting the -M option, auto is assumed.

When using the sbcupdate utility over RMCP bridge with Intel NetStructure MPCMM0001/0002 Chassis Management Module:

- Check that the CMM firmware version is 6.1.1.x or higher.
- Transfer the /usr/share/doc/sbcutils/cmdPrivilege.ini file to both the active and standby CMMs. Place cmdPrivelege.ini in the /etc/ directory on the CMMs. Then, restart the RMCP servers. More details on this can be found in Section 10.5.2, "Configuring the Intel NetStructure® MPCMM0001/0002 CMM" on page 191.

When using the sbcupdate utility over an RMCP bridge with a 3rd party Shelf Manager, it is essential that communication between the target IPMC and a remote network node via a shelf manager can be established. Refer to Table 74, "Required IPMI command privileges for RMCP bridging" on page 190 for more information on the required RMCP commands that must be forwarded to IPMB by the Shelf Manager.

The syntax of the command is:

```
# sbcupdate -I lan -M auto -H <shelf_manager_ip_address>
    -U <RMCP_username> -P <RMCP_password> -t <target>
    <path to firmware image/filename>
```

An example of auto firmware update via RMCP bridging is:

```
# sbcupdate -I lan -M auto -H 10.90.90.91 -U root
    -P cmmrootpass -t 0x8e MPCBL0050_FW_NNNNNNN.hex
```

10.6.7 Direct Firmware Update

Note: The user must not initiate a direct firmware update while a direct firmware update is in progress. Doing so results in the sbcupdate utility exiting because of write/verify failures. If this does happen, exit all invocations of sbcupdate, then the sbcupdate utility can be run again to update the firmware.

10.6.7.1 Direct Firmware Update Using the KCS Interface

To perform a direct firmware update over the KCS interface, you must be logged into the target SBC as root. Note that the KCS is the default interface if the -I option is omitted, so the command line can be abbreviated.

The syntax of the command using the short form for the options is:

```
    # sbcupdate -I kcs -M direct <path to firmware image/filename>
```

The syntax of the command using the long form for the options is:

```
    # sbcupdate --interface kcs --mode direct
        <path to firmware image/filename>
```

The output from this command should be similar to the following:

```
Entering Direct Firmware Update mode.
Entered Direct Firmware Update mode.
Erasing the current firmware area.
The firmware area has been erased.
Updating firmware.
0% completed
10% completed
20% completed
30% completed
40% completed
50% completed
60% completed
70% completed
80% completed
90% completed
100% completed
Exiting direct firmware update mode.
Firmware successfully updated.
Program exiting!
```

10.6.7.2 Direct Firmware Update to RTM via KCS

To perform a direct firmware update to an RTM, you must specify the target, -t option, to be the RTM. By omitting the -I option, the interface defaults to KCS.

For target information, refer to Table 76, “Common command line options and arguments” on page 198.

The following is an example of an RTM direct firmware update with the -I option omitted:

```
    # sbcupdate -M direct -t rtm MPRTM0050_NNNNNN.hex
```

10.6.7.3 Direct Firmware Update via RMCP Bridging

For RMCP bridging to work correctly, the Shelf Manager must be connected to the network so that it can transmit TCP/IP traffic to and from the client. The interface option must also be set to lan for RMCP bridging.

When using the sbcupdate utility over an RMCP bridge with the Intel NetStructure® MPCMM0001/0002 Chassis Management Module:

- Check for a CMM firmware version of 6.1.1.x or higher.
- Transfer the /usr/share/doc/sbcutils/cmdPrivilege.ini file to both the active and standby CMMs. Place cmdPrivelege.ini in the /etc/ directory on the CMMs. Then, restart the RMCP servers. More details on this can be found in Section 10.5.2, "Configuring the Intel NetStructure® MPCMM0001/0002 CMM" on page 191.

When using the sbcupdate utility over an RMCP bridge with a 3rd party Shelf Manager, it is essential that communication between the target IPMC and a remote network node via a shelf manager can be established. Refer to Table 74, "Required IPMI command privileges for RMCP bridging" on page 190 for more information on the required RMCP commands that must be forwarded to IPMB by the Shelf Manager.

The syntax of the command is:

```
# sbcupdate -I lan -M direct -H <shelf_manager_ip_address>
    -U <RMCP_username> -P <RMCP_password> -t <target>
    <path to firmware image/filename>
```

An example command for direct firmware update via RMCP bridging is:

```
# sbcupdate -I lan -M direct -H 10.90.90.91 -U root
    -P cmmrootpass -t 0x8e MPCBL0050_FW_NNNNNN.hex
```

10.6.7.4 Direct Firmware Update to RTM via RMCP Bridging

To perform a direct firmware update to an RTM via RMCP bridging, you must add the RTM to your target along with the target blade.

For RMCP bridging to work correctly, the Shelf Manager must be connected to the network so that it can transmit TCP/IP traffic to and from the client. The interface option must also be set to lan for RMCP bridging.

When using the sbcupdate utility over an RMCP bridge with the Intel NetStructure® MPCMM0001/0002 Chassis Management Module:

- Check for a CMM firmware version of 6.1.1.x or higher.
- Transfer the /usr/share/doc/sbcutils/cmdPrivilege.ini file to both the active and standby CMMs. Place cmdPrivelege.ini in the /etc/ directory on the CMMs. Then, restart the RMCP servers. More details on this can be found in Section 10.5.2, "Configuring the Intel NetStructure® MPCMM0001/0002 CMM" on page 191.

When using the sbcupdate utility over an RMCP bridge with a 3rd party Shelf Manager, it is essential that communication between the target IPMC and a remote network node via a shelf manager can be established. Refer to Table 74, "Required IPMI command privileges for RMCP bridging" on page 190 for more information on the required RMCP commands that must be forwarded to IPMB by the Shelf Manager.

The syntax of the command is:

```
# sbcupdate -I lan -M direct -H <shelf_manager_ip_address>
   -U <RMCP_username> -P <RMCP_password>
   -t <target_blade>:rtm <path to RTM firmware image/filename>
```

An example of a command for direct firmware update to RTM via RMCP bridging is:

```
    # sbcupdate -I lan -M direct -H 10.90.90.91 -U root
        -P cmmrootpass -t 0x8e:rtm MPRTM0050_NNNNNN.hex
```

10.6.8 Staged Firmware Update

10.6.8.1 Staged Firmware Update Using KCS Interface

The staged firmware update mode allows the option of creating a backup copy of the current operational image into a rollback partition area of flash memory, and then writes the new operational firmware image to a staging area of flash memory. This process happens while the IPMC is running normally. After completion of a staged update, the IPMC resets automatically and the new firmware is loaded to the IPMC (unless the -noreset-mmc option is used).

To perform a staged firmware update over the KCS interface, you must be logged into the target SBC as root.

The syntax of the command is:

```
    # sbcupdate -I kcs -M staged <path to firmware image/filename>
```

An example of a staged mode update is:

```
    # sbcupdate -I kcs -M staged MPCBL0050_FW_NNNNNN.hex
```

The output from this command should be similar to the following:

```
Entering Staged Firmware Update mode.
Preparing the staging flash area.
Staging flash area preparation complete.
Capturing the rollback image.
Rollback image captured successfully.
Uploading the new stage firmware image to the staging area.
0% completed
10% completed
20% completed
30% completed
40% completed
50% completed
60% completed
70% completed
80% completed
90% completed
100% completed
New staged image successfully uploaded.
Registering the staged firmware update in flash.
Staged firmware update successfully registered in flash.
The target management controller is being reset.
sbcupdate (W606) Please ignore any select timeout warnings.  They are issued by the
linux diver due to the target reset, and may be ignored.
IPMI message handler: BMC returned incorrect response, expected netfn 7 cmd 2, got
netfn31 cmd a2

The target management controller has been reset.
Staged Firmware Update complete!
Program exiting!
```

Note: The “IPMI message handler: BMC returned incorrect response.” error message is to be expected and may be ignored. This is a timeout error in response to warning (W606) as seen in the output above.

10.6.8.2 Creating a Pending Staged Firmware Update

To create a pending staged firmware update, the update command needs to be invoked with the --no-reset option, otherwise the target resets by default. This allows further operation with the current firmware with an update pending whenever a reset is initiated. The target can be reset with the --reset option.

The following is an example of creating a pending staged firmware update:

```
# sbcupdate -I kcs -M staged --no-reset MPCBL0050_FW_NNNNNN.hex
```

The following command resets the target initiating the firmware update:

```
# sbcupdate --reset
```

10.6.8.3 Staged Firmware Update via RMCP Bridging

For RMCP bridging to work correctly, the Shelf Manager must be connected to the network in order to transmit TCP/IP traffic to and from the client. The interface option must also be set to lan for RMCP bridging.

When using the sbcupdate utility over an RMCP bridge with the Intel NetStructure® MPCMM0001/0002 Chassis Management Module:

- Check for a CMM firmware version of 6.1.1.x or higher.
- Transfer the /usr/share/doc/sbcutils/cmdPrivilege.ini file to both the active and standby CMMs. Place cmdPrivelege.ini in the /etc/ directory on the CMMs. Then, restart the RMCP servers. More details on this can be found in Section 10.5.2, "Configuring the Intel NetStructure® MPCMM0001/0002 CMM" on page 191.

When using the sbcupdate utility over an RMCP bridge with a 3rd party Shelf Manager, it is essential that communication between the target IPMC and a remote network node via a shelf manager can be established. Refer to Table 74, "Required IPMI command privileges for RMCP bridging" on page 190 for more information on the required RMCP commands that must be forwarded to IPMB by the Shelf Manager.

After completion of a staged update, the IPMC resets automatically and the new firmware is loaded to the IPMC.

The syntax of the command is:

```
# sbcupdate -I lan -M staged -H <shelf_manager_ip_address
    or DNS_host_name> -U <username> -P <password>
    -t <target> <path_to_firmware_image/filename>
```

An example of staged firmware update via RMCP is:

```
# sbcupdate -I lan -M staged -H 10.90.90.91 -U root
    -P cmmrootpass -t 0x8e /tmp/MPCBL0050_FW_NNNNNN.hex
```

10.6.9 Canceling Staged Firmware Update

10.6.9.1 Cancel Staged Firmware Update via KCS Interface

To cancel a staged firmware update, a valid staged firmware update must already be pending.

To check if there is a pending staged firmware update, execute the -M info option.

The following is an example of how to check for pending updates:

```
    # sbcupdate -M info

Current firmware version: 1.02.00
Current bootblock version: 1.01
Current staged firmware version: 1.02.00 (pending)
Current rollback firmware version: 1.02.00
```

Note: This example output is only the beginning of the actual output. Notice that the “Current staged firmware version” is pending.

Execute the following command to cancel any pending staged firmware updates:

```
    # sbcupdate -M cancel
```

The output of this command is:

```
Canceling any pending staged firmware updates.
All pending staged firmware updates have been canceled.
Program exiting!
```

Execute the -M info option again to see that the staged firmware updated has canceled:

```
    # sbcupdate -M info

Current firmware version: 1.02.00
Current bootblock version: 1.01
Current staged firmware version: <None>
Current rollback firmware version: 1.02.00
```

10.6.9.2 Cancel Staged Firmware Update via RMCP Bridging

To cancel a staged firmware update, a valid staged firmware update must already be pending.

For RMCP bridging to work correctly, the Shelf Manager must be connected to the network so that it can transmit TCP/IP traffic to and from the client. The interface option must also be set to lan for RMCP bridging.

When using the sbcupdate utility over an RMCP bridge with the Intel NetStructure® MPCMM0001/0002 Chassis Management Module:

- Check for a CMM firmware version of 6.1.1.x or higher.
- Transfer the /usr/share/doc/sbcutils/cmdPrivilege.ini file to both the active and standby CMMs. Place cmdPrivelege.ini in the /etc/ directory on the CMMs. Then, restart the RMCP servers. More details on this can be found in Section 10.5.2, “Configuring the Intel NetStructure® MPCMM0001/0002 CMM” on page 191.

When using the sbcupdate utility over an RMCP bridge with a 3rd party Shelf Manager, it is essential that communication between the target IPMC and a remote network node via a shelf manager can be established. Refer to Table 74, “Required IPMI command privileges for RMCP bridging” on page 190 for more information on the required RMCP commands that must be forwarded to IPMB by the Shelf Manager.

The syntax of the command is:

```
    # sbcupdate -I lan -M cancel -H <shelf_manager_ip_address>
        -U <username> -P <password> -t <target>
```

For example:

```
    # sbcupdate -I lan -M cancel -H 10.90.90.91 -U root
        -P rootpass -t 0x8e
```

10.6.10 Setting Manual Rollback of the Firmware

10.6.10.1 Manual Rollback via KCS Interface

To force a manual rollback of the firmware, a valid rollback image must already be present on the target management controller. If there is no firmware image in the rollback area, the IPMC will go into direct update mode.

To run the command over the KCS interface, execute the following command:

```
# sbcupdate -I kcs -M rollback
```

Note: If no interface is specified, KCS is assumed.

The output from this command should be similar to the following:

```
Registering a manual rollback of the firmware on the next IPMC reset.
The target management controller is being reset.
sbcupdate (W606) Please ignore any select timeout warnings. They are issued by
the Linux driver due to target reset and may be ignored.
IPMI message handler: BMC returned incorrect response. expected netfn 7 cmd 2.
got netfn 31 cmd a4
The target management controller has been reset.
Manual rollback of the firmware for the next IMPC reset has been registered.
Program exiting!
```

Note: The "IPMI message handler: BMC returned incorrect response." error message is to be expected and may be ignored. This is a timeout error in response to warning (W606) as seen in the output above.

10.6.10.2 Manual Rollback via RMCP Bridging

To force a manual rollback of the firmware, a valid rollback image must already be present on the target management controller. If there is no firmware image in the rollback area, the IPMC will go into direct update mode.

For RMCP bridging to work correctly, the Shelf Manager must be connected to the network so that it can transmit TCP/IP traffic to and from the client. The interface option must also be set to lan for RMCP bridging.

When using the sbcupdate utility over RMCP bridge with Intel NetStructure MPCMM0001/0002 Chassis Management Module:

- Check for a CMM firmware version of 6.1.1.x or higher.
- Transfer the /usr/share/doc/sbcutils/cmdPrivilege.ini file to both the active and standby CMMs. Place cmdPrivelege.ini in the /etc/ directory on the CMMs. Then, restart the RMCP servers. More details on this can be found in Section 10.5.2, "Configuring the Intel NetStructure® MPCMM0001/0002 CMM" on page 191.

When using the sbcupdate utility over an RMCP bridge with a 3rd party Shelf Manager, it is essential that communication between the target IPMC and a remote network node via a shelf manager can be established. Refer to Table 74, "Required IPMI command privileges for RMCP bridging" on page 190 for more information on the required RMCP commands that must be forwarded to IPMB by the Shelf Manager.

The syntax of the command is:

```
# sbcupdate -I lan -M rollback
    -H <ip address of shelf manager> -U <RMCP_username>
    -P <RMCP_password> -t <target>
```

For example:

```
    # sbcupdate -I lan -M rollback -H 10.90.90.91 -U root
        -P cmmrootpass -t 0x8e
```

10.6.11 Retrieving FRU and SDR Data

Both FRU data and SDR data can be retrieved from a device using the get mode. More details and some examples using this option are provided in the remainder of this section.

10.6.11.1 Retrieving FRU and SDR via KCS Interface

In order to retrieve FRU and SDR data from a device via KCS, specify the get mode (-M get).

To run the command over the KCS interface, execute the following command:

```
    # sbcupdate -I kcs -M get
```

10.6.11.2 Retrieving FRU and SDR Information with -o Option

Both FRU and SDR information can be retrieved separately and stored in either binary or ASCII files. The -o option in conjunction with the -M get option saves FRU and SDR information to a specified file.

The syntax for using the -o option is:

```
    # sbcupdate -M get -o <filename.ext>
```

The .ext represents the extension of the filename. The filename can be one of the following:

- Binary FRU: <filename>.bfru
- Binary SDR: <filename>.bsdr
- ASCII FRU: <filename>.fru
- ASCII SDR: <filename>.sdr

An example of the command line for ASCII FRU information retrieval is as follows:

```
    # sbcupdate -M get -o getfru.fru
```

The output of this command should be similar to the following:

```
Successful retrieval of data from the target.
Program exiting!
```

The FRU data has been written to the getfru.fru file.

10.6.11.3 Retrieving FRU and SDR via RMCP Bridging

To retrieve the FRU and SDR contents of a management controller via an RMCP bridge, specify the get mode (-M get).

For RMCP bridging to work correctly, the Shelf Manager must be connected to the network so that it can transmit TCP/IP traffic to and from the client. The interface option must also be set to lan for RMCP bridging.

When using the sbcupdate utility over an RMCP bridge with the Intel NetStructure® MPCMM0001/0002 Chassis Management Module:

- Check for a CMM firmware version of 6.1.1.x or higher.

- Transfer the /usr/share/doc/sbcutils/cmdPrivilege.ini file to both the active and standby CMMs. Place cmdPrivelege.ini in the /etc/ directory on the CMMs. Then, restart the RMCP servers. More details on this can be found in Section 10.5.2, "Configuring the Intel NetStructure® MPCMM0001/0002 CMM" on page 191.

When using the sbcupdate utility over an RMCP bridge with a 3rd party Shelf Manager, it is essential that communication between the target IPMC and a remote network node via a shelf manager can be established. Refer to Table 74, "Required IPMI command privileges for RMCP bridging" on page 190 for more information on the required RMCP commands that must be forwarded to IPMB by the Shelf Manager.

The syntax of the command is:

```
# sbcupdate -I lan -M get -H <shelf_manager_ip_address>
    -U <RMCP_username> -P <RMCP_password>  -t <target>
```

For example, you might execute the following command:

```
# sbcupdate -I lan -M get -H 10.90.90.91 -U root
    -P cmmrootpass -t 0x8e
```

10.6.12 Updating FRU Data

To update FRU data on either the IPMC or the RTM, sbcupdate needs to know about the fields that exist in the FRU as well as the actual data to be written to the FRU.

Information about the fields in the FRU is contained in the /usr/share/doc/sbcutils/master.cfg file. This file applies to FRU information on both IPMCs and RTMs.

Information about the actual data to be written to the FRU is contained in a .fru file that is specific both to the SBC and to the FRU (either IPMC or RTM).

There are two possible FRU updates, full and incremental FRU updates. It is recommended to use the incremental FRU update method only.

- **Incremental FRU update** - The sbcutils package comes with a configuration file called /usr/share/doc/sbcutils/master.cfg that controls what system data to preserve. The supplied configuration file also directs sbcupdate to replace all PICMG and Intel records on the system with the input file PICMG and Intel records. Any other MRA record (i.e. customer created records) will not be altered during incremental FRU updates.
- **Full FRU update** - All system FRU information is removed from the system and replaced with FRU data from a file. Any modifications or additions performed by users, such as product number/name change, asset tag change or customer MRA record additions, are overwritten (lost).

Warning: Full FRU update erases all numbers specific to the particulat board (Serial Number, Top Assembly number, etc). It is advised always to use incremental FRU update

To perform FRU updates in certain circumstances, the --force-fru-update option may be necessary in the syntax. The following are examples of when to invoke this option:

- Downgrading an FRU version.
- The FRU version information is missing either from the system or from the input file.
- Performing a full FRU update.

Note: Using the --force-fru-update option will erase all FRU data and will not maintain any variable data in the FRU

The --force-id option may also be a necessary option in the following circumstances:

- The Product ID in the input file does not match the product ID of the system.
- The FRU version information is missing either from the system or from the input file.

10.6.12.1 Incremental FRU Update via KCS

To perform an incremental FRU update, use the supplied FRU file from the release package for your target IPMC along with the supplied /usr/share/doc/sbcutils/master.cfg file provided with the SBC utilities package. Staged FRU updates are not supported, therefore it is not necessary to specify direct mode (-M direct).

Here is an example of a full FRU update:

```
# sbcupdate -I kcs /usr/share/doc/sbcutils/master.cfg MPCBL0050_NNN.fru
```

To update the RTM FRU over the KCS interface on the MPCBL0050 SBC, execute the following command:

```
# sbcupdate -I kcs -t rtm /usr/share/doc/sbcutils/master.cfg MPRTM0050_NNN.fru
```

The output from this command should be similar to the following:

```
Beginning update of the FRU.
Writing FRU - Addr(0x20) ID(0) Bus(255)
Writing FRU Board Info Area
Writing FRU Product Info Area
Writing FRU Multi Record Area
Verifying contents of FRU
The FRU has been successfully updated.
Program exiting!
```

10.6.12.2 Incremental FRU Update via RMCP Bridging

To perform an incremental FRU update, use the supplied FRU file from the release package for your target IPMC along with the supplied /usr/share/doc/sbcutils/master.cfg file provided with the sbc utilities package. Staged FRU updates are not supported, therefore it is not necessary to specify direct mode (-M direct).

For RMCP bridging to work correctly, the Shelf Manager must be connected to the network so that it can transmit TCP/IP traffic to and from the client. The interface option must also be set to lan for RMCP bridging.

When using the sbcupdate utility over an RMCP bridge with the Intel NetStructure® MPCMM0001/0002 Chassis Management Module:

- Check for a CMM firmware version of 6.1.1.x or higher.
- Transfer the /usr/share/doc/sbcutils/cmdPrivilege.ini file to both the active and standby CMMs. Place cmdPrivelege.ini in the /etc/ directory on the CMMs. Then, restart the RMCP servers. More details on this can be found in Section 10.5.2, “Configuring the Intel NetStructure® MPCMM0001/0002 CMM” on page 191.

When using the sbcupdate utility over an RMCP bridge with a 3rd party Shelf Manager, it is essential that communication between the target IPMC and a remote network node via a shelf manager can be established. Refer to Table 74, “Required IPMI command privileges for RMCP bridging” on page 190 for more information on the required RMCP commands that must be forwarded to IPMB by the Shelf Manager.

The command to update the IPMC FRU on the MPCBL0050 has the following syntax:

```
    # sbcupdate -I lan -H <shelf_manager_ip_address>
        -U <RMCP_username> -P <RMCP_password> -t <target>
        /usr/share/doc/sbcutils/master.cfg MPCBL0050_NNN.fru
```

The command to update the RTM FRU on the MPRTM0050 has the following syntax:

```
    # sbcupdate -I lan -H <shelf_manager_ip_address>
        -U <RMCP_username> -P <RMCP_password>
        -t <target>:rtm /usr/share/doc/sbcutils/master.cfg MPRTM0050_NNN.fru
```

An example of updating the IPMC FRU:

```
    # sbcupdate -I lan -H 10.90.90.91 -U root -P cmmrootpass
        -t 0x8e /usr/share/doc/sbcutils/master.cfg MPCBL0050_NNN.fru
```

The following is an example of updating the RTM FRU:

```
    # sbcupdate -I lan -H 10.90.90.91 -U root -P cmmrootpass
        -t 0x8e:rtm /usr/share/doc/sbcutils/master.cfg MPRTM0050_NNN.fru
```

The output from this command should be similar to the following:

```
Beginning update of the FRU.
Writing FRU - Addr(0x20) ID(0) Bus(255)
Writing FRU Internal Use Area
Writing FRU Board Info Area
Writing FRU Product Info Area
Writing FRU Multi Record Area
Verifying contents of FRU
The FRU has been successfully updated.
Program exiting!
```

10.6.13 Full FRU Update via KCS

To perform a full FRU update, use the supplied FRU file for your target IPMC. Staged FRU updates are not supported, therefore it is not necessary to specify direct mode (-M direct). To invoke full FRU update requires the --force-fru-update command.

Caution: Full FRU updates remove all system FRU information and replaces it with FRU data from a file. All variable data and modifications/additions performed by users, such as product number or name change, asset tag change or customer MRA record additions are overwritten.

Here is an example of a full FRU update:

```
    # sbcupdate -I kcs --force-fru-update MPCBL0050_NNN.fru
```

The output from this command should be similar to the following:

```
Beginning update of the FRU.
Writing FRU - Addr(0x20) ID(0) Bus(255)
Writing FRU Board Info Area
Writing FRU Product Info Area
Writing FRU Multi Record Area
Verifying contents of FRU
The FRU has been successfully updated.
Program exiting!
```

The following is an example of when the FRU version information is missing from the system; the --force-id option must now be invoked:

```
# sbcupdate -I kcs --force-fru-update --force-id
    MPCBL0050_NNN.fru
```

To update the RTM FRU over the KCS interface on the MPCBL0050 SBC, execute the following command:

```
# sbcupdate -I kcs -t rtm --force-fru-update MPRTM0050_NNN.fru
```

10.6.14 Full FRU Update via RMCP Bridging

To perform a full FRU update, use the supplied FRU file for your target IPMC. Staged FRU updates are not supported, therefore it is not necessary to specify direct mode (-M direct). To invoke full FRU update requires the --force-fru-update command.

Caution: Full FRU updates remove all system FRU information and replace it with FRU data from a file. All variable data and modifications/additions performed by users, such as product number or name change, asset tag change or customer MRA record additions are overwritten.

For RMCP bridging to work correctly, the Shelf Manager must be connected to the network so that it can transmit TCP/IP traffic to and from the client. The interface option must also be set to lan for RMCP bridging.

When using the sbcupdate utility over an RMCP bridge with the Intel NetStructure® MPCMM0001/0002 Chassis Management Module:

- Check for a CMM firmware version of 6.1.1.x or higher.
- Transfer the /usr/share/doc/sbcutils/cmdPrivilege.ini file to both the active and standby CMMs. Place cmdPrivelege.ini in the /etc/ directory on the CMMs. Then, restart the RMCP servers. More details on this can be found in Section 10.5.2, "Configuring the Intel NetStructure® MPCMM0001/0002 CMM" on page 191.

When using the sbcupdate utility over an RMCP bridge with a 3rd party Shelf Manager, it is essential that communication between the target IPMC and a remote network node via a shelf manager can be established. Refer to Table 74, "Required IPMI command privileges for RMCP bridging" on page 190 for more information on the required RMCP commands that must be forwarded to IPMB by the Shelf Manager.

The command to update the IPMC FRU on the MPCBL0050 has the following syntax:

```
# sbcupdate -I lan -H <shelf_manager_ip_address>
    -U <RMCP_username> -P <RMCP_password> -t <target>
    --force-fru-update MPCBL0050_NNN.fru
```

The command to update the RTM FRU on the MPCBL0050 has the following syntax:

```
# sbcupdate -I lan -H <shelf_manager_ip_address>
    -U <RMCP_username> -P <RMCP_password>
    -t <target>:rtm --force-fru-update MPRTM0050_NNN.fru
```

An example of updating the IPMC FRU:

```
# sbcupdate -I lan -H 10.90.90.91 -U root -P cmmrootpass
    -t 0x8e --force-fru-update MPCBL0050_NNN.fru
```

The following is an example of updating the RTM FRU:

```
# sbcupdate -I lan -H 10.90.90.91 -U root -P cmmrootpass
    -t 0x8e:rtm --force-fru-update MPRTM0050_NNN.fru
```

The output from this command should be similar to the following:

```
Beginning update of the FRU.
Writing FRU - Addr(0x20) ID(0) Bus(255)
Writing FRU Internal Use Area
Writing FRU Board Info Area
Writing FRU Product Info Area
Writing FRU Multi Record Area
Verifying contents of FRU
The FRU has been successfully updated.
Program exiting!
```

10.6.15 Writing SDR Data

10.6.15.1 SDR Update via KCS

To perform an SDR update via KCS, use the supplied SDR file from the release package for your target IPMC. Staged SDR updates are not supported, therefore it is not necessary to specify direct mode (-M direct).

Note: If no interface is specified, KCS is assumed.

The output of this command should be similar to the following:

```
Beginning update of the SDRs.
Upgrading the SDR from version unkown to version 0.1.
The SDRs have been successfully updated.
The proxied target management controller is being reset.
The porxied target management controller has been reset.
The target management controller is being reset.
sbcupdate (W606) Please ignore any select timeout warnings. They are issued by
the Linux driver due to target reset and may be ignored.
IPMI message handler: BMC returned incorrect response. Expected netfn 7 cmd 2.
got netfn 7 cmd 33
The target  management controller has been reset.
Program exiting.
```

Note: The “IPMI message handler: BMC returned incorrect response.” error message is to be expected and may be ignored. This is a timeout error in response to warning (W606) as seen in the output above. This exact error message may not appear and it may not always be the same error message.

When performing an SDR update, in certain circumstances it may be necessary to invoke the --force-sdr-update option. The following are instances of when to invoke this option:

- Downgrading an SDR version.
- The SDR version information is missing either from the system or from the input file.
- The target has a pending staged firmware update or with RTM/AMC targets, the carrier board has a pending update.

An example of the use of the --force-sdr-update option is:

```
# sbcupdate -I kcs --force-sdr-update MPCBL0050_NNN.sdr
```

10.6.15.2 RTM SDR Update via KCS

To perform an RTM SDR update, you must specify the target (-t option) to be the RTM. You will use the supplied SDR file from the release package for your target RTM.

The following is an example of updating an RTM SDR:

```
# sbcupdate -I kcs -t rtm MPRTM0050_NNN.sdr
```

10.6.15.3 SDR Update via RMCP Bridge

To perform an SDR update via an RMCP bridge, use the supplied SDR file from the release package for your target ICMP. To run the command over an RMCP bridge, you must have a shelf manager (an Intel NetStructure® MPCMM0001/0002 Chassis Management Module) that bridges IPMB traffic. The shelf manager must be connected to the network so that it can transmit TCP/IP traffic to and from the client. The interface selected must be lan, since the KCS interface is assumed otherwise.

The syntax of the command used to execute an MPCBL0050 SDR update is as follows:

```
# sbcupdate -I lan -H <shelf_manager_ip_address>
    -U <RMCP_username> -P <RMCP_password> -t <target>
    MPCBL0050_NNN.sdr
```

For example:

```
# sbcupdate -I lan -H 10.90.90.91 -U root -P cmmrootpass
    -t 0x8e MPCBL0050_NNN.sdr
```

10.6.16 RTM SDR Update via RMCP Bridge

To perform an RTM SDR update via RMCP bridge, you must specify the target blade as well as the RTM.

The syntax of RTM SDR update via RMCP bridge command is as follows:

```
# sbcupdate -I lan -H <shelf_manager_IP_address>
    -U <RMCP_username> -P <RMCP_password>
    -t <target blade>:rtm MPRTM0050_NNN.sdr
```

For example:

```
# sbcupdate -I lan -H 10.90.90.91 -U root -P cmmrootpass
    -t 0x8e:rtm MPRTM0050_NNN.sdr
```

10.6.17 Writing Asset Tag Information

To write asset tag information, execute the following command:

```
sbcupdate [-i <interface>] [-t <target>]
    /usr/share/doc/sbcutils/asset.cfg
```

You are prompted for an asset number, which consists of one to 20 alphanumeric characters. This string is written to the device targeted in the command (IPMC or RTM) over the specified interface (or over KCS if no interface is specified).

The following is an example of writing asset tag information:

```
# sbcupdate -I kcs /usr/share/doc/sbcutils/asset.cfg
```

The output from the command should be similar to the following:

```
Enter value for PRODUCT Asset Tag (Maximum length allowed is 20 characters):
-Your data entered-
Writing FRU - ADDR(0x20) ID(0) Bus(0)
Writing FRU Product Info Area
Verifying contents of FRU
Program exiting!
```

10.6.18 Output from the –M info Option

The following is example output from the -M info option. See Table 76, “Common command line options and arguments” on page 198 for more information about using the -M info option. The version in the staged area and the version in the rollback area are both included the output.

For example:

```
    # sbcupdate -I kcs -M info
```

The output from this command should be similar to the following:

```
root@demo:~/utils# sbcupdate -M info
sbcupdate - Intel SBC Utilities 1.3.2.2
Running with the invocation: -M info
Current firmware version: 1.02.00
Current bootblock version: 1.02
Current staged firmware version: <None>
Current rollback firmware version: <None>

Manufacturer ID: 343 (0x00000157)
Product ID: 2063 (0x080f)
Device type: IPMC
FPGA revision: C.0B
Board revision: 0x70

FRU device ID: 0
FRU version: 1.01
FRU Asset Tag: 00000000000000000000
BIOS Primary Version: WH500ES0.86E.01.02.0000.062820071709

SDR version: SDR File 1.04
SDR Package version: SDR Package 1.04

Successful retrieval of data from the target.
Program exiting!
```

An example of the -M info option for the RTM is as follows:

```
    # sbcupdate -M info -t rtm
```

The output from this command should be similar to the following:

```
Current firmware version: 1.02.00
Current bootblock version: 1.02

Manufacturer ID: 343 (0x00000157)
Product ID: 2226 (0x08b2)
Device type: RTM
FPGA revision: <Not Available>
Board revision: <Not Available>

FRU device ID: 1
FRU version: 1.01

FRU Asset Tag: <Blank>

SDR version: SDR File 1.04

SDR Package version: SDR Package 1.04

Successful retrieval of data from the target.
Program exiting!
```

10.7 Sensor Device Record (SDR) Threshold Management

The SBC update tool can be use to manage the SDR thresholds setting of all sensors. The SDR threshold change feature is for customers that want to change the default values of Intel defined SDR thresholds. The primary use of this is to change upper non-critical temperature thresholds so that the blade temperature thresholds can be set for different temperature environments. The types of thresholds that can be managed are follows:

- The ACTIVE thresholds, which are the current thresholds value of the operational FRU
- The SDR thresholds, which store sensor threshold values that are used to set the active sensor thresholds when the FRU transitions from the M0 state to the M1 state
- The factory default thresholds. The ACTIVE and SDR thresholds can be restored to these values without an external configuration file. The ACTIVE and SDR thresholds values are not allowed to be set outside of the range defined by non-recoverable thresholds in the factory default setting.

The following subsections describe step-by-step procedure to read, change and restore the SDR thresholds setting.

Caution: Changes to voltage and current thresholds can lead to device corruption or failure.

Note: Intel is not responsible for customer changes of thresholds that fall outside the Intel-defined factory default settings.

10.7.1 Reading Thresholds Using KCS Interface

To read the threshold from a device via KCS, specify the threshold mode using `–M threshold` and add options to specify types of threshold to be read.

The syntax of the command is:

```
# sbcupdate -I kcs -M threshold [--get-thr-factory]
      [--get-thr-active] [--get-thr-sdr] [--get-thr] -s 0xYZ
```

where:

- `--get-thr-factory` reads FACTORY settings of thresholds
- `--get-thr-active` reads ACTIVE settings of thresholds
- `--get-thr-sdr` reads thresholds settings from SDR repository
- `--get-thr` reads all of the above
- `0xYZ` specifies the sensor number and accepts the following value:
  - — `0x00` to 0xFE to read thresholds settings of single sensor
  - — `0xFF` to read threshold settings of all sensors

The following is an example to read the threshold of a sensor threshold:

```
# sbcupdate -I kcs -M threshold --get-thr-factory --get-thr-active
--get-thr-sdr --get-thr -s 0x93
```

The output from this command should be similar to the following:

```
Reading thresholds settings.
SENSOR_NUMBER                          : 93
_ACTIVE_LOWER_NON_CRITICAL_THRESHOLD   : B0
_ACTIVE_LOWER_CRITICAL_THRESHOLD       : AB
_ACTIVE_LOWER_NON_RECOVERABLE_THRESHOLD : A1
```

```
_ACTIVE_UPPER_NON_CRITICAL_THRESHOLD      : EC
_ACTIVE_UPPER_CRITICAL_THRESHOLD          : F9
_ACTIVE_UPPER_NON_RECOVERABLE_THRESHOLD   : FE
_ACTIVE_POSITIVE_GOING_HYSTERESIS         :  2
_ACTIVE_NEGATIVE_GOING_HYSTERESIS         :  2
_SDR_LOWER_NON_CRITICAL_THRESHOLD         : B0
_SDR_LOWER_CRITICAL_THRESHOLD             : AB
_SDR_LOWER_NON_RECOVERABLE_THRESHOLD      : A1
_SDR_UPPER_NON_CRITICAL_THRESHOLD         : EC
_SDR_UPPER_CRITICAL_THRESHOLD             : F9
_SDR_UPPER_NON_RECOVERABLE_THRESHOLD      : FE
_SDR_POSITIVE_GOING_HYSTERESIS            :  2
_SDR_NEGATIVE_GOING_HYSTERESIS            :  2
_FACTORY_LOWER_NON_CRITICAL_THRESHOLD     : B0
_FACTORY_LOWER_CRITICAL_THRESHOLD         : AB
_FACTORY_LOWER_NON_RECOVERABLE_THRESHOLD  : A1
_FACTORY_UPPER_NON_CRITICAL_THRESHOLD     : EC
_FACTORY_UPPER_CRITICAL_THRESHOLD         : F9
_FACTORY_UPPER_NON_RECOVERABLE_THRESHOLD  : FE
_FACTORY_POSITIVE_GOING_HYSTERESIS        :  2
_FACTORY_NEGATIVE_GOING_HYSTERESIS        :  2
Program exiting!
```

Note: Only specified sensor threshold values will appear in the output. The output will be displayed on the console or redirected to the output file specified by `<-o filename.thr>` in the command line.

The syntax of the command to redirect output to a .thr file:

```
# sbcupdate [-I kcs] -M threshold [--get-thr-factory]
      [--get-thr-active] [--get-thr-sdr] [--get-thr]
      -s 0xYZ [-o filename.thr]
```

Below is the example of the content of the filename.thr:

```
SENSOR_NUMBER                              : 93
_ACTIVE_LOWER_NON_CRITICAL_THRESHOLD       : B0
_ACTIVE_LOWER_CRITICAL_THRESHOLD           : AB
_ACTIVE_LOWER_NON_RECOVERABLE_THRESHOLD    : A1
_ACTIVE_UPPER_NON_CRITICAL_THRESHOLD       : EC
_ACTIVE_UPPER_CRITICAL_THRESHOLD           : F9
_ACTIVE_UPPER_NON_RECOVERABLE_THRESHOLD    : FE
_ACTIVE_POSITIVE_GOING_HYSTERESIS          :  2
_ACTIVE_NEGATIVE_GOING_HYSTERESIS          :  2
_SDR_LOWER_NON_CRITICAL_THRESHOLD          : B0
_SDR_LOWER_CRITICAL_THRESHOLD              : AB
_SDR_LOWER_NON_RECOVERABLE_THRESHOLD       : A1
_SDR_UPPER_NON_CRITICAL_THRESHOLD          : EC
_SDR_UPPER_CRITICAL_THRESHOLD              : F9
_SDR_UPPER_NON_RECOVERABLE_THRESHOLD       : FE
_SDR_POSITIVE_GOING_HYSTERESIS             :  2
_SDR_NEGATIVE_GOING_HYSTERESIS             :  2
_FACTORY_LOWER_NON_CRITICAL_THRESHOLD      : B0
_FACTORY_LOWER_CRITICAL_THRESHOLD          : AB
_FACTORY_LOWER_NON_RECOVERABLE_THRESHOLD   : A1
_FACTORY_UPPER_NON_CRITICAL_THRESHOLD      : EC
_FACTORY_UPPER_CRITICAL_THRESHOLD          : F9
_FACTORY_UPPER_NON_RECOVERABLE_THRESHOLD   : FE
_FACTORY_POSITIVE_GOING_HYSTERESIS         :  2
_FACTORY_NEGATIVE_GOING_HYSTERESIS         :  2
```

10.7.2 Reading Thresholds via RMCP Bridging

In order for RMCP bridging to work correctly, the shelf manager must be connected to the network in order to transmit TCP/IP traffic to and from the client. The user must also specify the `-I lan` interface option for RMCP bridging.

To read the threshold from a device via RMCP bridge, specify the threshold mode (`–M threshold`) and add options to specify types of threshold to be read.

The syntax of the command is:

```
# sbcupdate -I lan -H <ip address of shelf manager> -U <username>
      -P <password> -t <IPMB address> -M threshold
      [--get-thr-factory] [--get-thr-active] [--get-thr-sdr]
      [--get-thr] -s 0xYZ
```

For example, users would execute the following command:

```
# sbcupdate -I lan -H 10.90.90.91 -U root -P cmmrootpass -t 0x8e
      -M threshold --get-thr-factory --get-thr-active
      --get-thr-sdr --get-thr -s 0x93
```

10.7.3 Changing Thresholds Values Using KCS Interface

To change the threshold values using the KCS interface, regular ".thr" file needs to be specified in the command line.

The syntax of the command is:

```
# sbcupdate -I kcs <path to change threshold file/filename>
```

The following is the example to change only selected thresholds using the change.thr file:

```
# sbcupdate -I kcs change.thr
```

The output from this command should be similar to the following:

```
Changing thresholds settings.
Program exiting!
```

Note: The change.thr file is a file of the same format as the regular .thr file created by the sbcupdate tool when reading thresholds to a file. Each record in the .thr file should begin with _SENSOR_NUMBER line followed by predefined THRESHOLDS lines. Only thresholds listed in records are affected in SBC.

An example to change SBC Temp 1 Upper Non-critical thresholds is as follow:

1. Create a change.thr file with content as below:
   ```
   _SENSOR_NUMBER                              : 30
   _ACTIVE_UPPER_NON_CRITICAL_THRESHOLD        : 45
   ```
2. Run the change threshold command.
   ```
   # sbcupdate -I kcs change.thr
   ```
3. Read the sensor threshold using a command as in Section 10.7.1, "Reading Thresholds Using KCS Interface" or the cmmget command to make sure the SBC Temp 1 Upper Non-critical thresholds is changed to the value stated in the change.thr file.
   ```
   # sbcupdate -I kcs -M threshold --get-thr-factory
   --get-thr-active --get-thr-sdr --get-thr -s 0x30
   ```
   or
   ```
   # cmmget -l blade12 -t "SBC Temp 1" -d thresholdsall
   ```

10.7.4 Changing Thresholds Values via RMCP Bridging

For RMCP bridging to work correctly, the shelf manager must be connected to the network in order to transmit TCP/IP traffic to and from the client. The user must also specify the interface option to be "lan" for RMCP bridging.

To change the threshold values, the regular .thr file needs to be specified in the command line.

The syntax of the command is:

```
# sbcupdate -I lan -H <ip address of shelf manager>
      -U <username> -P <password> -t <IPMB address>
      <path to restore threshold file/filename>
```

For example, user would execute the following command:

```
# sbcupdate -I lan -H 10.90.90.91 -U root -P cmmrootpass
      -t 0x8e change.thr
```

10.7.5 Restoring Factory Default Thresholds Values Using KCS Interface

To restore the factory default values of ACTIVE and SDR threshold types using the KCS interface, specify the threshold type to be restored.

The syntax of the command is:

```
# sbcupdate -I kcs [--restore-thr-active] [--restore-thr-sdr]
      [--restore-thr] -s 0xYZ
```

where:

- `--restore-thr-active` restores ACTIVE settings of thresholds
- `--restore-thr-sdr` restores thresholds settings from SDR repository
- `--restore-thr` restores all of the above
- `0xYZ` specifies the sensor number and accepts the following value:
    - 0x00 to 0xFE to read thresholds settings of single sensor
    - 0xFF to read threshold settings of all sensors

The sbcudate utility also supports restoring only selected thresholds values using the KCS interface by providing a special restore.thr file in command line.

The syntax of the command is:

```
# sbcupdate -I kcs <path to restore threshold file/filename>
```

The following example shows how to restore only selected thresholds using the restore.thr file:

```
# sbcupdate -I kcs restore.thr
```

The output from this command should be similar to the following:

```
Restoring factory default thresholds settings.
Program exiting!
```

Note: The restore.thr file is a file of the same format as the regular .thr file created by the sbcupdate tool when reading thresholds to a file. Because factory default settings can not be restored, any _FACTORY_ lines that appear in restore.thr will be ignored and a warning will be displayed on console.

10.7.6 Restoring Factory Default Thresholds Values via RMCP Bridging

For RMCP bridging to work correctly, the shelf manager must be connected to the network in order to transmit TCP/IP traffic to and from the client. The user must also specify the interface option to be lan for RMCP bridging.

To restore the factory default values of ACTIVE and SDR threshold types via RMCP bridge, specify the threshold type to be restored.

The syntax of the command is:

```
# sbcupdate -I lan -H <ip address of shelf manager>
      -U <username> -P <password> -t <IPMB address>
      [--restore-thr-active] [--restore-thr-sdr]
      [--restore-thr] -s 0xYZ
```

The sbcudate utility also supports restoring only selected thresholds values using RMCP bridge by providing a special restore.thr file in command line.

The syntax of the command is:

```
# sbcupdate -I lan -H <ip address of shelf manager>
      -U <username> -P <password> -t <IPMB address>
      <path to restore threshold file/filename>
```

For example, user would execute the following command:

```
# sbcupdate -I lan -H 10.90.90.91 -U root -P cmmrootpass -t
      0x8e restore.thr
```

10.8 Displayed Messages

Warning and error messages are formatted with the name of the utility generating the message, the message number and the message text. For example, if sbcupdate is given a firmware file intended for a product that is not the same as the target MMC, the following message is displayed:

```
sbcupdate (E525) Product ID mismatch!  The supplied input file is not intended for
this target.
```

Table 77 lists possible informational messages and Table 78 lists the possible warnings and errors.

Table 77. Informational messages (Sheet 1 of 3)

Message	Description
%1 - Intel SBC Utilities %2	Version information displayed with the ¬¬ version option. %1 is the name of the utility and %2 is the version of the SBC Utilities package.
Entering Staged Firmware Update mode.	This informational message is displayed when the utility begins a Staged Firmware Update.
Preparing the staging flash area.	This informational message is displayed when the staging area in flash memory is being erased, making room for the new image to be uploaded. This operation can take a number of seconds to complete, depending on system activity.
Staging flash area preparation complete.	This informational message is displayed once the staging area has been erased.
Capturing the rollback image.	This informational message is displayed when the rollback image is being copied. This operation includes erasing the rollback area prior to capturing a new rollback image. This operation can take a number of seconds to complete, depending on system activity.
Rollback image captured successfully.	This informational message is displayed when the rollback image has been successfully captured.
No rollback is being captured.	This informational message is only displayed when the no capture rb command-line option is included with the update. Messages I503 and I504 are not displayed if this message is displayed.
Uploading the new staged firmware image to the staging area.	This informational message is displayed once the new staging image supplied in the Intel HEX file is being written to flash memory.

Table 77. Informational messages (Sheet 2 of 3)

Message	Description
New staged firmware image successfully uploaded.	This informational message is displayed once the image has finished being written to flash memory. This does not imply the update is complete.
Forced firmware update in progress, ignoring product and manufacturer IDs.	This informational message is displayed when the force id command-line option is included in the update. This overrides comparisons between product and manufacturer IDs between the supplied HEX file from Intel and what is on the SBC. This would be used in the event the firmware is corrupt and is reporting false information about the SBC; use this only when directed.
Registering the staged firmware update in flash.	This informational message is displayed at the conclusion of the staged update. It indicates that the staged image was successfully written and verified. A bit is set in flash memory telling the IPMC to write the new staged image to the operational area on the next IPMC reset. This operation may take a few seconds, depending on system activity.
Staged firmware update successfully registered in flash.	This informational message is displayed when the registration of the valid staging image has been completed.
Staged Firmware Update complete!	This informational message is displayed when all staged update operations have completed successfully. This indicates the update has been completed.
Entering Direct Firmware Update mode.	This informational message is displayed when a direct firmware update is being initiated. This indicates the operational code in the IPMC is being halted, and all commands will be serviced by the boot block. This operation may take a few seconds, depending on the number of outstanding transactions being processed by the IPMC.
Entered Direct Firmware Update mode.	This informational message is displayed after the transition from operational firmware to boot block has been completed. This is only displayed once a GetDeviceID shows the lowest bit of byte 4 on, per the IPMI spec (Is Device Busy).
Updating firmware.	This informational message is displayed before the writing of the firmware is initiated.
Firmware successfully updated.	This informational message is displayed after the new firmware has been written and verified.
Exiting direct firmware update mode.	This informational message is displayed when the transition from the boot block back to the new operational code is initiated. This message implies that the IPMC is being reset.
Program exiting!	This informational message is displayed when the utility has completed all operations and is completing execution.
Mismatch at offset %1 (length %2 bytes).	This informational message is displayed if the write verification during a direct firmware update finds a mismatch. %1 is replaced with the byte offset that the difference was located at and %2 is replaced with the number of bytes found to be different.
Got: %1 Expected: %2	This informational message is displayed if the write verification during a direct firmware update finds a mismatch. %1 is replaced with the data byte read from flash memory, and %2 is replaced with the data byte that should have been in flash memory.
%1%% completed	This informational message is displayed during a direct firmware update for every 10% of the firmware image written. %1 is replaced with the percentage completed. It ranges from 0% to 100%.
The %1 target management controller is being reset.	This informational message is displayed after the utility is about to sent a cold reset command to the target MMC.
The target management controller has been reset	This informational message is displayed after the utility has sent a cold reset command to the target MMC and no error completion codes have been received.

Table 77. Informational messages (Sheet 3 of 3)

Message	Description
Running with the invocation: %1	Displays a copy of all arguments, given exactly as they were supplied on the command line.
%1 in progress.	This verbose message is a level 1 verbose message, and is displayed during CaptureRollback or EraseStagingArea operations. It can be viewed by passing -v 1 or --verbose 1 on the command-line. Note that higher levels of verbosity still activate this message. %1 is replaced with the current function being executed. This message is only available in staged updates.
Canceling any pending staged firmware updates.	This informational message is displayed when the 'cancel' mode is invoked. It indicates the start of the process to cancel a pending staged firmware update.
All pending staged firmware updates have been canceled.	This informational message is displayed when the cancel mode has been completed successfully.
Registering a manual rollback of the firmware on the next IPMC reset.	This informational message is displayed when the 'rollback' mode is invoked. It indicates the start of the process to force a rollback to happen on the next IPMC reset.
Manual rollback of the firmware for the next IPMC reset has been registered.	This informational message is displayed when the manual rollback registration has been completed.
Successful retrieval of the FRU.	This informational message is displayed when the FRU get operation has completed successfully.
The FRU has been successfully updated.	This informational message is displayed when the FRU has been successfully written to the target FRU device.
Successful retrieval of data from the target.	A read operation from the system has completed with no errors.
The SDRs have been successfully updated.	This informational message is displayed when the SDR has been successfully written to the target device.
Beginning update of the FRU.	This information message is displayed to indicate the start of the FRU update.
Beginning update of the SDRs.	This information message is displayed to indicate the start of the SDR update.
No firmware version to report.	This message is displayed when using the info mode to obtain version information of a board that is not provided by Intel in firmware transfer mode. Once the board (not provided by Intel) exits firmware transfer mode, the firmware version can be read.
Upgrading the FRU from version %1 to version %2.	This information message is displayed to indicate the direction of FRU update, as an upgrade from one version to another. %1 is replaced with the existing system FRU version. %2 is replaced with the newer version that is being used for the update.
Upgrading the SDR from version %1 to version %2.	This information message is displayed to indicate the direction of SDR update, as an upgrade from one version to another. %1 is replaced with the existing system SDR version. %2 is replaced with the newer version that is being used for the update.
Erasing the current firmware area.	In versions of the boot block that support one shot full-firmware erasures, this message is displayed to signal the start of erasure.
The firmware area has been erased.	This message is displayed to signal the end of firmware erasure. It is only displayed if the target boot block supports one shot full-firmware erasure.

Table 78. Warning and error messages (Sheet 1 of 11)

Number	Message	Description
E102	Unsupported interface “%1”. Must be one of {kcs | lan}.	The user has supplied the interface option with an argument other than “kcs” or “lan”. %1 is replaced with the offending interface name.
E103	Unsupported authorization type “%1”. Must be one of {none | md2 | md5 | password}.	The user has supplied the authtype option with an argument other than “none”, “md2”, “md5”, or “password”. %1 is replaced with the offending authorization type.
E104	Unsupported privilege level “%1”. Must be one of {callback | user | operator | admin}.	The user has supplied the privilege option with an argument other than “callback”, “user”, “operator” or “admin”. %1 is replaced with the offending privilege level.
E105	Illegal IPMB address: 0x%1.	The user has supplied an out-of-range IPMB address. The range enforced by sbcupdate is [0x02, 0xFE], but the chassis in use is likely to require a subset of this range.
E106	Parse error: %1.	This message is displayed as a result of a number of different errors in invocation syntax such as, unknown option, missing option argument, string supplied for a numerical option argument, etc. %1 is replaced with further detail on the parse problem.
E107	AMC site number %1 is out-of-range (must be between %2 and %3).	From the AMC specification, the AMC site numbers must be between 1 and 8, inclusive.
E108	IPMC site number %1 is out-of-range (must be between %2 and %3).	The user has supplied an out-of-range IPMC site number. The range enforced by sbcupdate is [0x01, 0xFF], but the chassis in use is likely to require a subset of this range.
E109	Syntax error in target argument: “%1”	The argument to the target option is malformed.
E110	When communicating locally, it is not possible to specify a target blade other than 0x20.	For “direct to the blade” operation (that is, KCS or RMCP+ direct), if the target argument includes an IPMB address, it must be 0x20 (for example: --target 0x20:rtm or --target rtm). SBCs and CMMs from Intel do not support bridging from one SBC to another to from the SBC to the ShMC.
E150	IPMI communication problem while reading FRU address info.	During a FRU operation, this message is displayed if there is a problem sending messages to the carrier SBC, for example, a timeout or error completion code. The three commands send to the carrier are Get Device ID, Get PICMG Properties and Get Address Info.
E151	The specified system does not support FRUs of type "%1".	The user specified an FRU operation to a target that is not supported by the SBC. For example, this message is displayed if the target argument is “amc” and the SBC in use does not have an AdvancedMC* module.
E152	The specified target is ambiguous. There is more than one possible FRU of type "%1".	If the target’s carrier board supports more than one sub-FRU of the given type, then the sub-FRU’s site number must be supplied on the command line. For example, on an MPCBL0050 SBC, it is fine to use “--target amc” because only a single amc is supported. Using the same target argument for a multi-AMC SBC is ambiguous. In that case, it is required to use something like “--target amc6”.
E153	There is no response from any %1 at the specified location.	A sub-FRU was specified but is not currently present in the system.

Table 78. Warning and error messages (Sheet 2 of 11)

Number	Message	Description
E154	IPMI communication error while checking for the target presence.	While attempting to check for the presence of a sub-FRU, there was an IPMI communication error, such as a timeout. No completion code was returned, successful or otherwise, in response to the query.
E155	IPMI communication problem while determining the IPMB-0 address of IPMC site %1.	While attempting to find the IPMB address of the target (such as the Get Address Info command) there was an IPMI communication error, such as a timeout. No completion code was returned, successful or otherwise, in response to the query.
E156	The specified system does not support IPMCs at site %1.	The target site specified is supported by the chassis, but not for an AdvancedTCA board. For example, slots 15 and 16 of an MPCHC0001 contain PEM and fan trays, but not SBCs.
E157	Cannot bridge to the target from a %1. Did you mean to use the --target/-t option?	Cannot communicate to a second SBC while running locally on a different SBC (for example, the KCS interface).
E158	Cannot contact the %1 while its carrier board is in firmware transfer mode.	It is not possible to communicate with a sub-FRU (for example, an RTM or AMC) while the carrier board (for example, SBC) is in firmware transfer mode. Try again later when the carrier is no longer in transfer mode, or bring the carrier out of transfer mode with the Intel OEM Exit FW Transfer mode command.
E204	RMCP connection to "%1" failed	An RMCP/RMCP+ session could not be activated with the target because of an earlier problem such as a bad username or password.
E206	Invalid Username	An RMCP/RMCP+ session can not be established because the supplied username was not accepted by the target.
E207	Invalid Password	An RMCP/RMCP+ session can not be established because the supplied password was not accepted by the target.
E213	The authentication type: %1 is unavailable for the selected privilege level	An RMCP/RMCP+ session can not be established because the desired authentication algorithm (for example, MD2, MD5, etc.) was refused by the target for the supplied privilege level. If no privilege is specified, the default privilege is administrator. If no privilege is specified, it is the same as typing –L administrator.
E214	No valid authentication type available	For the given user and channel, the target does not support MD5, MD2, straight password or none. This could mean the target is misconfigured, or that the target only supports an OEM authentication algorithm.
E218	Unable to find suitable KCS Driver	No IPMI request can be made using any known IPMI device driver. For operating systems that have a compiled-in IPMI driver, this indicates that the KCS interface is in a bad state and may require a power cycle of the SBC and/or any attached sub-FRUs. For OSs that have loadable IPMI driver modules, the KCS interface could be broken as above, or else the IPMI driver service needs to be started with the /etc/init.d/ipmi start command.
E222	No slot available to establish session	For direct LAN communication to an SBC (for example, RMCP+ direct), the SBC already has its maximum number of RMCP+ sessions active and cannot accept a new session. For an RMCP bridge, it is the ShMC that cannot accept an additional RMCP/RMCP+ session.
E226	A LAN failure occurred	A LAN (that is RMCP/RMCP+) error occurred that is not described by an other situation in this table.

Table 78. Warning and error messages (Sheet 3 of 11)

Number	Message	Description
E546	Requested information not supported on the target	This message should only be displayed if there is an internal code error in which a FRU library operation is called without a specification of what operation is to be performed.
E700	Unsupported firmware update mode "%1". It must be one of %2	This error message is displayed when an unsupported mode is supplied on the command line. The command-line option is -M or --mode. %1 is replaced with what was supplied, and %2 is replaced with the mode types that are supported.
E701	You must supply exactly one input filename	This error message is displayed when a HEX file provided by Intel was not supplied on the command line, or if there is more than one file or option specified at the end of the command line.
E702	Manufacturer ID mismatch! The supplied input file is not intended for this target.	This error message is displayed when the supplied HEX file provided by Intel has a different manufacturer ID for the target SBC. This can be displayed whether performing a direct or staged firmware update.
E703	Product ID mismatch! The supplied input file is not intended for this target.	This error message is displayed when the supplied HEX file supplied by Intel has a different product ID for the target SBC. This can be displayed whether performing a direct or staged firmware update.
E704	Staged firmware updates are not available at this time. Either the target does not have that capability, or the target is in direct firmware transfer mode. Use -M direct or --mode direct to perform a firmware upgrade.	This error message is displayed when a staged firmware update was requested, but the target IPMC does not support them. This can only be displayed when specifically requesting a staged update with mode staged or -M staged.
E705	Firmware rollback updates are not supported for the specified target.	This error message is displayed if the IPMC does not support capturing a rollback image. This is more of a safety measure, and can be overridden with nocapturerb. This is only displayed if attempting a staged firmware update.
E706	Preparation of the staging flash area failed!	This error message is displayed if an error occurred while erasing the flash memory where the staging image is to be written. Failure cases can be a flash error, no permission, or a refused operation due to another operation being performed on the same area. The underlying library should supply a more descriptive message indicating what specifically failed. This is only displayed if attempting a staged firmware update.
E707	Capture of the rollback image failed!	This error message is displayed if an error occurred while capturing the rollback image. Failure cases can be a flash error, no permission, or a refused operation due to another operation being performed on the same area. The underlying library should supply a more descriptive message indicating what specifically failed. This is only displayed if attempting a staged firmware update.
E708	Upload of the new staged image failed!	This error message is displayed if an error occurred while writing the new staging image to the staging area of flash memory. Failure cases can be a flash error, no permission, or a refused operation due to another operation being performed on the same area. The underlying library should supply a more descriptive message indicating what specifically failed. This is only displayed if attempting a staged firmware update.

Table 78. Warning and error messages (Sheet 4 of 11)

Number	Message	Description
E709	Registration of staged firmware update in flash failed!	This error message is displayed if an error occurred while writing to flash memory indicating the IPMC needs to perform a staged update on reset. Failure cases can be a flash error, no permission, or a refused operation due to another operation being performed on the same area. The underlying library should supply a more descriptive message indicating what specifically failed. This is only displayed if attempting a staged firmware update.
E710	Error updating the firmware!	This error message is displayed if any error occurred during a direct firmware update. Possible causes are communication failures during RMCP-related updates, flash errors, and no permission.
E711	Cannot cancel pending staged firmware updates because the target is in firmware transfer mode.	This error message is displayed if the cancel mode is requested, but the blade is in firmware transfer mode.
E712	Cannot register a manual rollback of the firmware because the target is in firmware transfer mode.	This error message is displayed if the rollback mode is requested, but the blade is in firmware transfer mode.
E713	Cannot cancel pending staged firmware updates because staged firmware updates are not supported by this target.	This error message is displayed if the cancel mode is requested, but staged firmware updates are not supported by the currently running firmware.
E714	Cannot register a manual rollback of the firmware because staged firmware updates are not supported by this target.	This error message is displayed if the rollback mode is requested, but staged firmware updates are not supported by the currently running firmware.
E720	Exit direct firmware update mode operation refused by target.	This error message is displayed if the Exit Firmware Transfer mode IPMI command is refused by the IPMC. This can happen if the IPMC is busy and is unable to switch states. This only happens during direct firmware updates.
E721	Write direct firmware update area operation refused by target.	This error message is displayed if a write command is refused by the IPMC. This can happen if the flash area is busy and cannot be accessed. This only happens during direct firmware updates.
E722	Verify direct firmware update area operation refused by target.	This error message is displayed if a read command is refused by the IPMC during write verification. This can happen if the flash area is busy and cannot be accessed. This only happens during direct firmware updates.
E723	Flash error occurred during an exit direct firmware update mode operation.	This error message is displayed if a flash memory error is detected after attempting to exit Firmware Transfer Mode. This only happens during direct firmware updates.
E724	Flash error occurred during a write direct firmware update area operation.	This error message is displayed if a flash memory error is detected during a write operation. This only happens during direct firmware updates.
E725	Flash error occurred during a verify direct firmware update area operation.	This error message is displayed if a flash memory error is detected during a read operation while verification of the previous write. This only happens during direct firmware updates.

Table 78. Warning and error messages (Sheet 5 of 11)

Number	Message	Description
E726	Communication error while attempting to exit direct firmware update mode.	This error message is displayed if an IPMI communication failure occurs while sending the Exit Firmware Transfer Mode request. Communication failures can be caused by lossy networks if using RMCP or an RMCP bridge, excessive IPMB traffic within a chassis, or an overrun shelf manager (RMCP bridge only). This only happens during direct firmware updates.
E727	Communication error while attempting to verify direct firmware update area.	This error message is displayed if an IPMI communication failure occurs when sending a read request during verification. Communication failures can be caused by lossy networks if using RMCP or an RMCP bridge, excessive IPMB traffic within a chassis, or an overrun shelf manager (RMCP bridge only). This only happens during direct firmware updates.
E728	Communication error while attempting to set the write segment.	This error message is displayed if an IPMI communication failure occurs when attempting to set the current write segment. Communication failures can be caused by lossy networks if using RMCP or an RMCP bridge, excessive IPMB traffic within a chassis, or an overrun shelf manager (RMCP bridge only). This only happens during direct firmware updates.
E729	The target reports a previous staged firmware update failure.	This warning message is currently not displayed during a staged update. It is part of the GetStatus command. It would be displayed if a non-successful IPMI completion code was returned from the GetStatus command.
E730	IPMI error while attempting to register staged firmware update (completion code 0x%1).	This error message is displayed when an unknown, non-successful IPMI completion code is returned when registering the staged update in flash. This can only be displayed during a staged update. %1 is replaced with the IPMI completion code, displayed in hexadecimal format.
E731	IPMI error while attempting to close staging area (completion code 0x%1).	This error message is displayed when an unknown, non-successful IPMI completion code is returned when closing the staging area after an update has been performed. This can only be displayed during a staged update. %1 is replaced with the IPMI completion code, displayed in hexadecimal format.
E732	IPMI error while attempting to erase staging area (completion code 0x%1).	This error message is displayed when an unknown, non-successful IPMI completion code is returned when erasing the staging area prior to an update being performed. This can only be displayed during a staged update. %1 is replaced with the IPMI completion code, displayed in hexadecimal format.
E733	IPMI error while attempting to capture rollback (completion code 0x%1).	This error message is displayed when an unknown, non-successful IPMI completion code is returned when capturing a rollback image. This can only be displayed during a staged update. %1 is replaced with the IPMI completion code, displayed in hexadecimal format.
E734	IPMI error while attempting to get staged update status (completion code 0x%1).	This error message is displayed when an unknown, non-successful IPMI completion code is returned when getting the status of an update. This is currently not being used. This can only be displayed during a staged update. %1 is replaced with the IPMI completion code, displayed in hexadecimal format.
E735	Register staged firmware update operation refused by target.	This error message is displayed when a register update operation is refused by the IPMC. This can happen if the firmware currently has that part of flash in use. This can only be displayed during staged updates.

Table 78. Warning and error messages (Sheet 6 of 11)

Number	Message	Description
E736	Close staging area operation refused by target.	This error message is displayed if the staged area is attempted to be closed and is refused by the IPMC. This can happen if the firmware currently has that part of flash in use. This can only be displayed during staged updates.
E737	Write staging area operation refused by target.	This error message is displayed if a write operation is refused by the IPMC during a staged update. This can happen if the firmware currently has that part of flash in use. This can only be displayed during staged updates.
E738	Erase staging area operation refused by target.	This error message is displayed if erasing the staging area is refused by the IPMC. This can happen if the firmware currently has that part of flash in use. This can only be displayed during staged updates.
E739	Capture rollback operation refused by target.	This error message is displayed if capturing the rollback image is refused by the IPMC. This can happen if the firmware currently has that part of flash in use. This can only be displayed during staged updates.
E740	Flash error occurred during a register staged firmware update operation.	This error message is displayed if a flash error occurs while registering the staged update in flash memory. This can only be displayed during staged updates.
E741	Flash error occurred during a close staging area operation.	This error message is displayed if a flash error occurs when closing the staging area. This can only be displayed during staged updates.
E742	Flash error occurred during an erase staging area operation.	This error message is displayed if a flash error occurs while the staging area of flash memory is being erased. This can only be displayed during staged updates.
E743	Communication error while attempting to write to the staging firmware area.	This error message is displayed if an IPMI communication failure occurs while writing a staged update. Communication failures can be caused by lossy networks if using RMCP or an RMCP bridge, excessive IPMB traffic within a chassis, or an overrun shelf manager (RMCP bridge only). This can only be displayed during staged updates.
E744	Capture rollback operation timed out, aborting staged firmware update.	This error message is displayed if the capture rollback operation does not abort with an error, but takes too long to complete. This can be caused by excessive load on the IPMC, or heavy flash memory access on the IPMC. This can only be displayed during staged updates.
E745	The target reports that the staging area image is corrupt.	This error message is displayed if the Get Status command finds that the staging image is corrupt. Possible causes are undetected write errors or failing flash memory. This message is not displayed in the current version, since checks are performed during an update that satisfy this function.
E746	The target reports that the rollback area image is corrupt.	This error message is displayed if the Get Status command finds that the rollback image is corrupt. Possible causes are undetected write errors or failing flash memory. This message is not displayed in the current version, since checks are performed during an update that satisfy this function.

Table 78. Warning and error messages (Sheet 7 of 11)

Number	Message	Description
E749	File %1 cannot be opened: %2.	This error message is displayed if the provided filename cannot be opened. Examples are the file is missing or permissions are insufficient. %1 is replaced with the supplied filename, and %2 is replaced with the error.
E751	Unknown hex record type encountered at line %1.	This error message is displayed if the supplied file contains a record not conforming to the HEX record standard for Intel. This could indicate that the supplied file is corrupt. %1 is replaced by the line number in the file where the bad record was found.
E752	Hex record checksum incorrect at line %1.	This error message is displayed if the two's complement checksum of the hex record line does not match the line's checksum. This format is defined by the HEX record standard for Intel. This could indicate the supplied file is corrupt. %1 is replaced by the line number in the file where the bad record was found.
E753	Unable to allocate memory while parsing file.	This error message is displayed when the supplied filename is being read into memory, and the operating system runs out of available memory. This indicates that the client machine is running a large program, or may be running a program with a memory leak.
E754	No EOF hex record found. Supplied file appears to be corrupt.	This error message is displayed when the supplied file has no HEX EOF (end-of-file) record. This is defined by the HEX record standard for Intel. This could indicate the supplied file is corrupt.
E755	Unknown open failure for file %1.	This error message is displayed when an unknown error occurs while attempting to open the supplied file. This message is the last catch-all message to detect unknown failures. %1 is replaced with the supplied filename.
E756	The supplied file has an unsupported opcode header version (%1).	This error message is displayed if the supplied filename contains a data format for the operational code that is not supported. The data format could differ when major enhancements are made to the operational image. %1 is replaced with the opcode header version that was found in the supplied file.
E757	The supplied file does not contain a valid operational code header.	This error message is displayed if the header portion of the operational code image cannot be found. This could indicate the supplied file is corrupt.
E758	Invalid record length encountered at line %1.	This error message is displayed if the HEX record specifies a certain length for a record, but the data does not match that length. This could indicate the supplied file is corrupt.
E759	Errors occurred while running prechecks. Aborting.	This error message is displayed if all other known checks and messages cannot catch the error. This message should not be seen in practical operation.
E760	The specified file %1 is an unknown file type.	This error message is displayed if a file is specified that has an unknown file extension. Valid file extensions are: .hex, .cfg, .fru, .bsdr, and.bfru.
E761	Staged FRU updates are not supported.	This error message is displayed if an FRU update is being performed, and the mode is staged.
E762	Cannot downgrade the firmware from version %1 to version %2 without the --force-fw-update flag.	This flag is required to show the user's intention to downgrade the firmware. If this is really the intended operation, supply the switch and this error no longer appears.

Table 78. Warning and error messages (Sheet 8 of 11)

Number	Message	Description
E764	Cannot perform an SDR update with a pending staged firmware image. Please cancel the staged update, or use --force-sdr-update or --no-reset.	If there is a pending staged firmware update, it is not possible to perform an SDR update that would reset the target. If the target were to reset after the SDR update (which is the default behavior), then the FW version would change also. To do the SDR update, either commit the pending firmware update by resetting the target, or prevent target reset after the SDR update with the --no-reset flag, or else give the --force-sdr-update flag to show that the pending firmware update should be committed at the same time.
E765	IPMI error when trying to contact the carrier board for the proxied target.	Before communicating with a sub-FRU such as an RTM or AMC, the carrier board (for example, an SBC) must be queried to see if it is in a mode that would prevent sub-FRU communication. This message is displayed if there is a timeout to this query.
E766	Cannot perform requested operations against the requested target. The target's carrier board is in firmware transfer mode, and cannot forward requests.	If a carrier board is in firmware transfer mode, then none of its sub-FRUs are contactable. It is not possible, for example, to update RTM firmware while its carrier is busy in firmware transfer mode.
E767	A timeout occurred waiting for the target management controller to reset.	After sbcupdate resets a target, it waits until the target is able to respond to a get device ID and the response indicates the target is no longer in firmware transfer mode.
E768	Error performing the requested operation	An operation failed with an unknown error. This is a generic catch-all and should not be displayed in practice.
E769	Requested operation not supported on the target.	A user attempted to get a type of information (for example, FRU, SDR, etc.) from a target that does not support that type.
E770	IPMI error during a Get Device Id transfer.	A non-successful completion code was returned for a Get Device ID command request.
E772	IPMI error during a Staged Get Capabilities transfer.	A non-successful completion code was returned for a Staged Get Capabilities command request.
E773	IPMI error during a Staged Get Version Data transfer.	A non-successful completion code was returned for a Staged Get Version command request.
E774	No valid staging or rollback images were found to register. Aborting.	A user attempted to use the rollback mode when there was no valid staged or rollback firmware image available.
E775	Cannot perform a full FRU upgrade without specifying --force-fru-update. Performing a full FRU update will overwrite all data stored in the FRU such as serial numbers, part numbers, manufacture date, etc. Please supply a .cfg file name to perform an incremental FRU update.	This error message is displayed when a full FRU update is attempted without specifying the force flag on the command line. It also briefly states the impact of a full FRU update.
E776	Cannot downgrade the FRU from version %1 to version %2 without the --force-fru-update flag.	This error message is displayed on a FRU downgrade attempt without the force flag. The FRU downgrade attempt explicitly requires the –-force-fru-update flag.
E777	Cannot compare FRU Product ID. System FRU Product ID is missing. Use --force-id flag to update	This error message is displayed when the Product ID information in the existing system FRU is missing. Due to this, the Product ID in the supplied input FRU file and system FRU cannot be compared.
E778	Cannot downgrade the SDR from version %1 to version %2 without the --force-sdr-update flag.	This error message is displayed on an SDR downgrade attempt without the force flag. The SDR downgrade attempt explicitly requires the –force-sdr-update flag.

Table 78. Warning and error messages (Sheet 9 of 11)

Number	Message	Description
E779	Cannot compare SDR versions. Both System and File information is missing. Use --force-sdr-update flag to update	This error message is displayed when the SDR version information is missing from both the system SDR and the supplied input SDR file. Hence SDR version information cannot be compared.
E780	Cannot retrieve FRU/SDR data since the target is in firmware transfer mode.	This error message is displayed when FRU or SDR data is queried but the target is already in the firmware transfer mode.
E781	Cannot perform FRU update since the target is in firmware transfer mode.	This error message is displayed when a FRU update is attempted, but the target is already in the firmware transfer mode.
E782	Cannot perform SDR update since the target is in firmware transfer mode.	This error message is displayed when a SDR update is attempted but the target is already in the firmware transfer mode.
E784	Cannot compare both FRU Product ID and version. System information is missing. Use --force-id and --force-fru-update flag to update	This error message is displayed when both FRU product ID and version data from the system FRU is missing. In such a case, the user is directed to use the appropriate force flags to be able to perform the operation.
E785	Cannot compare both FRU Product ID and version. File information is missing. Use --force-id and --force-fru-update flag to update	This error message is displayed when both FRU product ID and version data from the supplied input FRU file is missing. In such a case, the user is directed to use the appropriate force flags to be able to perform the operation.
E786	Cannot compare FRU versions. System FRU version information is missing. Use --force-fru-update flag to update	This error message is displayed when the system FRU version information is missing and hence the FRU version data cannot be compared with the corresponding value in the supplied input FRU file. User is directed to use the appropriate force flag to be able to perform the operation.
E787	Cannot compare FRU versions. File FRU version information is missing. Use --force-fru-update flag to update	This error message is displayed when the FRU version information in the supplied input FRU file is missing and therefore the FRU version data cannot be compared with the corresponding value in the system FRU. The user is directed to use the appropriate force flag in order to perform the operation.
E788	Cannot compare FRU Product ID. File FRU Product ID is missing. Use --force-id flag to update	This error message is displayed when the FRU Product ID information in the supplied input FRU file is missing and therefore this data cannot be compared with the corresponding value in the system FRU. The user is directed to use the appropriate force flag in order to perform the operation.
E789	Cannot compare FRU versions. Both File and System FRU version information is missing. Use --force-fru-update flag to update	This error message is displayed when the FRU version information is missing in both the supplied input FRU file and the system FRU. The user is directed to use the appropriate force flag in order to perform the operation.
E790	Cannot compare FRU Product ID. Both File and System FRU Product ID is missing. Use --force-id flag to update	This error message is displayed when the FRU Product ID information is missing in both the supplied input FRU file and the system FRU. The user is directed to use the appropriate force flag in order to perform the operation.
E791	Cannot compare both FRU Product ID and version. Both System and File information is missing. Use --force-id and --force-fru-update flag to update	This error message is displayed when both the FRU product ID and version data is missing from both the supplied input FRU file and the system FRU. The user is directed to use the appropriate force flags in order to perform the operation.
E792	Cannot compare SDR versions. File information is missing. Use --force-sdr-update flag to update	This error message is displayed when the SDR version information is missing from the supplied input SDR file. Therefore, this data cannot be compared with the system SDR version information. The user is directed to use the appropriate force flags in order to perform the operation.

Table 78. Warning and error messages (Sheet 10 of 11)

Number	Message	Description
E793	Cannot compare SDR versions. System information is missing. Use --force-sdr-update flag to update	This error message is displayed when the SDR version information is missing from the system SDR. Therefore, this data cannot be compared with the version information in the supplied input SDR file. The user is directed to use the appropriate force flags in order to perform the operation.
E800	Target is in firmware transfer mode. You must force the firmware update with --force-fw-update since another firmware update may be running right now.	When a target is in firmware transfer mode, it either means that a second user is actively updating the firmware, or that the target is in firmware transfer mode with no active update in progress. In the latter case, use the --force-fw-update flag to initiate an update.
W201	Bad hostname or IPv4 address "%1"	The argument to the --H/--host option contains an illegal character such as "~".
W202	Socket connection has been dropped	The socket connection is no longer open due either to network problems or the remote RMCP node (usually the ShMC) not responding.
W203	The selected user does not have %1 privilege level access	This message is displayed if, during RMCP/RMCP+ communication, the user given on the command line does not have administrative privilege, or does not have the privilege specified with the "-L" option. Either use a different user or configure the target so that the desired user has the required privilege.
W221	KCS Driver Failure	The IPMI driver is responding with an error such as a timeout.
W225	Non-standard RMCP bridging detected.	If, during an RMCP bridge, the ShMC responds to a send message request with data from the embedded send message request, this message is displayed. This type of send message response does not conform to the IPMI 2.0 specification. This is only a warning and not an error because sbcupdate is designed to work this way with a shelf manager that does not implement RMCP bridging per the specification.
W600	Upgrading with the same version of the firmware, version %1.	This warning message is displayed when the major, minor, and build version numbers of the firmware are the same between the SBC and supplied firmware image. This message can be displayed on both staged and direct firmware updates.
W601	Retrying direct firmware update.	This warning message is displayed when any failure is detected during a direct firmware update, and the update is being retried. The retry is only attempted once, and implies that the whole image will be rewritten. For example, if the update is 90% complete and an error occurs, the retry begins at 0%.
W602	Target management controller currently in direct firmware update mode, cannot compare firmware versions.	This warning message is displayed when the target IPMC is in firmware transfer mode. When in this mode, the version of the current firmware is meaningless; therefore a comparison cannot be done between the supplied file and the target management controller. This message can be displayed on both staged and direct firmware updates.
W603	Downgrading the firmware from version %1 to version %2.	This warning message is displayed when the firmware is being downgraded. %1 is the current version, %2 is the version attempting to be programmed to the target IPMC.
W605	The target management controller needs to be manually reset for the SDR update to take effect.	This warning message is displayed when an SDR update is attempted specifying a -–no-reset flag (no reset) and there is a pending staged firmware image. For the SDR update to take effect, the target controller needs to be manually reset.

Table 78. Warning and error messages (Sheet 11 of 11)

Number	Message	Description
W606	Please ignore any select timeout warnings. They are issued by the Linux driver due to the target reset, and may be ignored.	When a target resets, for example after a firmware or SDR update, it may not be able to respond to the Linux IPMI driver's presence pings. If the driver does not receive a ping in time, the driver issues warning messages. These can safely be ignored.
W607	Downgrading the FRU from version %1 to version %2.	This warning message is displayed to indicate the direction of FRU update, as a downgrade from one version to another. %1 is replaced with the existing system FRU version. %2 is replaced with the older version that is being used as input for the update.
W608	Upgrading with the same version of the FRU, version %1.	This warning message is displayed to indicate the direction of FRU update, as a upgrade, but with the same version as exists on the system. %1 is replaced with the existing FRU version.
W609	Downgrading the SDR from version %1 to version %2.	This warning message is displayed to indicate the direction of SDR update, as a downgrade from one version to another. %1 is replaced with the existing system SDR version. %2 is replaced with the older version that is being used as input for the update.
W610	Upgrading with the same version of the SDR, version %1.	This warning message is displayed to indicate the direction of SDR update, as a upgrade but with the same version as exists on the system. %1 is replaced with the existing FRU version.
W611	Target is in firmware transfer mode, hence cannot display all information	This warning message is displayed to indicate that on the "Info" query, not all information can be displayed since the target is in firmware transfer mode.

Table 79. Mapping of physical slot to IPMB address in Intel NetStructure® MPCHC0001 14U Shelf

Physical Slot	Logical Slot	Hardware Address	IPMB Address
1	13	4Dh	9Ah
2	11	4Bh	96h
3	9	49h	92h
4	7	47h	8Eh
5	5	45h	8Ah
6	3	43h	86h
7	1	41h	82h
8	2	42h	84h
9	4	44h	88h
10	6	46h	8Ch
11	8	48h	90h
12	10	4Ah	94h
13	12	4Ch	98h
14	14	4Eh	9Ch

11.0 Specifications

This chapter provides the MPCBL0050 mechanical, environmental, and reliability specifications.

11.1 Mechanical Specifications

11.1.1 Board Outline

The MPCBL0050 form factor is mechanically compliant with the PICMG 3.0 Specification which stipulates dimensions of 322.25 mm x 280.00 mm (12.687" x 11.024"). The board pitch is 6HP, and the PCB thickness is 2.36 mm (±0.23 mm).

Figure 52 shows the locations of major components of the MPCBL0050 SBC. Table 80 lists the components shown in the figure.

Figure 52. Component layout

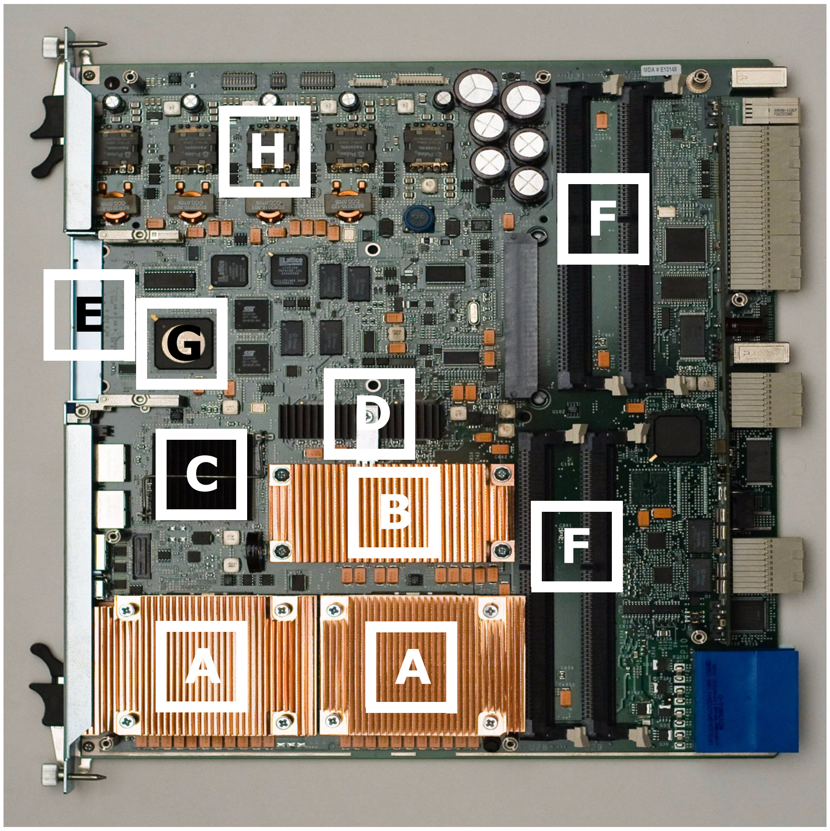


Table 80. Board components

Label (Figure 52)	Component/Function
A	Dual-Core Intel® Xeon® 5138 LV 2.13GHz processors
B	Intel® 5000P Memory Controller Hub (MCH)
C	Intel® 6321ESB I/O Controller Hub (ICH)
D	Intel® 82571EB Gigabit Ethernet Controller (Fabric Interface)
E	Advanced Mezzanine Card slot
F	Fully Buffered DIMMs slots
G	LSI* 1064 SAS Controller
H	Power conversion circuitry

11.1.2 Backing Plate and Primary Side Top Cover

The MPCBL0050 SBC has a rugged metal backing plate that forms a single-piece faceplate. This backing plate is made of zinc-plated commercial-quality cold-rolled steel. The backing plate and integral faceplate are nominally 1.0 mm thick. The top cover is made of zinc-plated commercial-quality cold-rolled steel and is nominally 0.60 mm thick.

There is a removable part of the top cover for easy access to memory modules. The solid backing plate and top cover provide PCB stiffening, enhanced EMI protection from adjacent boards, and protection during flame tests.

Caution: Removing the backing plate can damage the components on the board and may void the warranty if any hardware is damaged during the removal/reattachment of the backing plate. No user-serviceable parts are available under the PCB. Do not remove the faceplate or backing plate.

11.2 Environmental Specifications

The test methodology is a combination of Intel and NEBS test requirements with the intent that the product will pass pure system-level NEBS testing. The MPCBL0050 has passed board level tests that are applicable to NEBS.

Table 81 summarizes environmental limits, both operating and nonoperating.

Table 81. Environmental specifications

Parameter	Conditions	Detailed Specification
Temperature (Ambient)	Operating (Normal	5 to 40°C
	Operating (Short term)	-5 to 55°C
	Storage	-40 to 70°C
Humidity	Operating	15%-90% (non-condensing) at 55°C
	Storage	5%-95% (non-condensing) at 40°C
Altitude	Operating	4,000 m (13,100 ft.) Note: may require additional cooling above 1800 m (5,900 ft.)
Unpackaged Vibration	Operating	Sine sweep: 5 to 100 Hz: 1G @ 0.25 Octave/minute
Shock	Storage	50G, 170 inches/second trapezoidal

11.3 Reliability Specifications

11.3.1 Mean Time Between Failure (MTBF) Specifications

Calculation Type:	MTBF/FIT Rate
Standard:	Telcordia* Standard SR-332 Issue 1
Methods:	Method I, Case I, Quality Level II

The calculation results were generated using the references and assumptions listed. This report and its associated calculations supersede all other released Mean Time Between Failures (MTBF) and Failure in Time (FIT) calculations of earlier report dates. The reported failure rates do not represent catastrophic failure. Catastrophic failure rates will vary based on application environment and features critical to the intended function.

Table 82. Reliability estimate data

Failure Rate (FIT)	11,772 failures in 10^9 hours
MTBF	84,947 hours

Note: The MTBF data is calculated without AdvancedMC and DIMMs installed.

11.3.1.1 Environmental Assumptions

- Failure rates are based on a 40° C ambient temperature.
- Applied component stress levels are 50% (voltage, current, and/or power).
- Ground, fixed, controlled environment with an environmental adjustment factor equal to 1.0.

11.3.1.2 General Assumptions

- Component failure rates are constant.
- Board-to-system interconnects included within estimates.
- Non-electrical components (screws, mechanical latches, labels, covers, etc.) are not included within estimations.
- Printed circuit board is considered to have a 0 FIT rate.

11.3.1.3 General Notes

- Method I, Case I = Based on parts count. Equipment failure is estimated by totaling device failures rates and quantities used.
- Quality Level II = Devices purchased to specifications, qualified devices, vendor lot-to-lot controls for AQLs and DPMs.

Where available, direct component supplier predictions or actual FIT rates have been used.

The SBC MTBF does not include addition of the AMCs, hard drives or memory. Please contact the manufacturer for specific component and relevant operational MTBF information.

11.4 Power Consumption

The power consumed by the MPCBL0050 SBC is dependent on the configuration (AdvancedMC type, Memory Modules type, RTM population) and workload. Table 83 shows operating voltage ranges and Table 84 gives total measured power values for selected board configurations.

Table 83. Operating voltage ranges

Operating Modes	Voltage
Normal	-38 VDC to -72 VDC
Non-Operating	0 VDC to < -38 VDC, -72 VDC to -75 VDC

Note: These voltages assume a 1 V round trip drop on power signals between shelf power input terminals and board/module slots.

Power measurements configuration:

- Chassis: Schroff 14-slot, P/N 11592452
- MPCBL0050 Single board computer
- FB-DIMM modules: 4GB, Micron* MT36HTF51272FY-667E1D4
- MPRTM0050 Rear Transition Module
- Fiber CHannel SFP: Intel TXN31115, 4Gbps Fiber Channel SFP
- SFF SAS HDD:HP* 146GB, 431954-003
- Set of load application both for CPU and I/O stressing.
- Ambient Temperature: 25C
- Supply voltage: -48V

Table 84. Total measured powe

Configuration	Idle	Under load
Four 4 GByte FB-DIMMs No AdvancedMC No RTM	105W	164W
Four 4 GByte FB-DIMMs No AdvancedMC MPRTM0050 with Fiber Channel SFPs, NO HDD	114W	173W
Four 4 GByte FB-DIMMs No AdvancedMC MPRTM0050 with Fiber Channel SFPs, 146GB SFF SAS HDD	122W	178W

Note: AdvancedMC power consumption is not included in above measurements. Maximum power budget available for AdvancedMC is 20W.

11.5 Board Layer Specifications

- Material: HTG FR4
- Layers: 18
- Copper:
  - — Outer layers 1 and 16 are 0.5 oz. copper plated.
  - — Internal layers 3, 5, 7, 12, 14 and 16 are 0.5 oz. copper.

- Middle planes 9 an 10 are 2 oz. copper.
- All others are 1 oz. copper.

11.6 Cooling Requirements

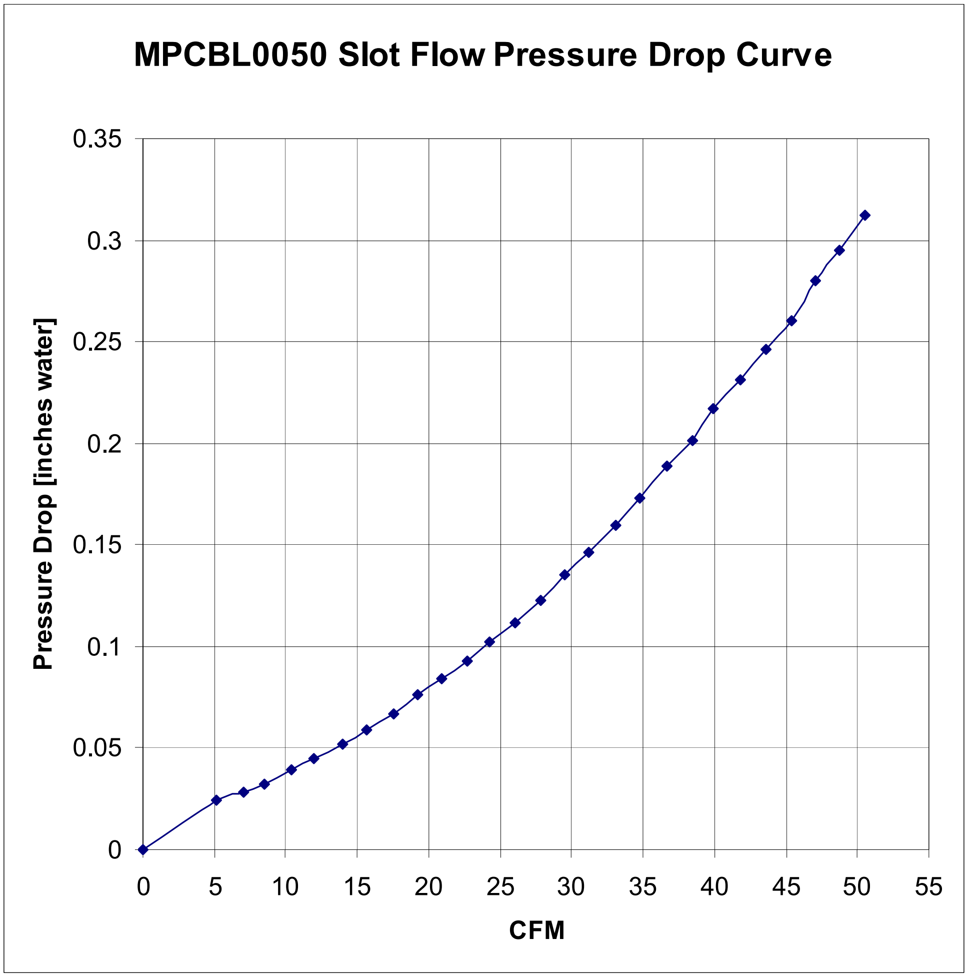
The MPCBL0050 SBC should be installed vertically in a chassis, with bottom-to-top airflow. Airflow is expected to be distributed across the bottom edge of the installed MPCBL0050 blade and to maintain at least 30 cubic feet per minute (CFM) airflow.

- Most components on the MPCBL0050 blade are specified to operate with a localized ambient temperature up to 70° C and do not require heat sinks.
- The MPCBL0050 blade uses custom heat sinks designed by Intel. The following are the on-board components that have heat sinks installed:
  - Processors
  - GbE Controllers (Intel® 82571EB Gigabit Ethernet Controller)
  - Memory Controller Hub (Intel® E7520 MCH Chipset)
  - I/O Controller Hub (Intel® 6321ESB ICH Chipset)
- The rate of airflow specified above is critical to ensuring that the blade operates as designed.

11.7 Thermals

The curve of the pressure drop versus the flow rate in Figure 53 represents the flow impedance of the slot. This information is provided in accordance with Section 5 of the PICMG 3.0 Specification to aid system integrators in using the Intel NetStructure® MPCBL0050 Single Board Computer in various AdvancedTCA* shelves.

Figure 53. Pressure vs. flow rate



11.8 Weight

The weight of the baseboard is 3.50 kg (7.72 lbs) with no DIMMs, no packaging materials and one AdvancedMC filler panel installed.

Each DIMM adds up to approximately 0.045 kg (0.10 lbs). For the most accurate memory weight, please check with the manufacturer for the particular make/model of interest.

The packaged weight of the MPCBL0050 is 4.45 kg (9.81 lbs). Packaged weight includes baseboard, packaging materials and one AdvancedMC filler panel. DIMMs do not ship with the board are not included in the package weight.

11.9 Compliance

The MPCBL0050 product is compatible with the following specifications:

- AdvancedTCA 3.0 R2.0 and ECN002 (AdvancedTCA base specification)

- AdvancedTCA 3.1 R1.0 (Ethernet/Fibre Channel over AdvancedTCA)
- AdvancedMC AMC.0 R1.0 (AdvancedMC base specification)
- AdvancedMC AMC.1 R1.0 (PCI Express and Advanced Switching)
- AdvancedMC AMC.2 R1.0 (Gb Ethernet)

IPMI 2.0 (Intelligent Platform Management Interface)

12.0 Agency Information

12.1 North America (FCC Class A)

Federal Communications Commission (FCC) Part 15 Rules

This equipment has been tested and found to comply with the limits for a Class A digital device, pursuant to Part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment. This equipment generates, uses, and can radiate radio frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. Operation of this equipment in a residential area is likely to cause harmful interference, in which case the user will be required to correct the interference at his own expense.

12.2 Canada – Industry Canada (ICES-003 Class A)

Industry Canada ICES-003 Issue 3

This Class A digital apparatus meets all requirements of the Canadian Interference-Causing Equipment Regulations.

Cet appareil numérique de la classe A respecte toutes les exigences du Règlement sur le matérial brouilleur du Canada.

12.3 European Union

The products covered by this notice meet the following European Directives:

- 73/23/EEC Low Voltage Directive
- 89/336/EEC EMC Directive

To achieve CE compliance, be sure to select a host that already meets the EMC and Low Voltage Directives before the addition of any optional board. Remember that the use of option boards declared compliant with the Directives by their manufacturer only gives “presumption of compliance” for the whole system. It is the responsibility of the system supplier to verify that the requirements of the listed Directives are still met by the final system, as supplied to the end-user. System integrators should take notice of further conditions expressed in the sections below and the Safety Information sheet supplied with each board.

Warning: This is a class A product. In a domestic environment this product may cause radio interference in which case the user may be required to take adequate measures.

Compliance with the R&TTE Directive

The R&TTE Directive includes its own safety and EMC requirements. Although equipment declared compliant to the R&TTE Directive does not require explicit declaration of conformity to EMC and Low Voltage Directives, above conditions must also be met to satisfy the safety and EMC requirements of the R&TTE Directive.

Intel Declarations of Conformity for the products covered by this notice can be found under the “Network Building Blocks” heading at http://developer.intel.com/design/litcentr/ce_docs

Manufacturer's office in European Union:

Intel Corporation (UK) Ltd.
Pipers Way
Swindon, Wiltshire SN3 1RJ
UK
Tel: +44 (0)1793 403000
Fax: +44 (0)1793 641440

13.0 Certifications

Safety:

- IEC60950-1
- EN60950
- UL/CSA 60950-1

Hazardous substances:

- The Intel NetStructure® MPCBL0050N01Q has been verified to be compliant with the European Directive 2002/95/EC, officially titled “The Restriction on the Use of Hazardous Substances (RoHS) in Electrical and Electronic Equipment” or RoHS. Specifically, this product uses only RoHS compliant parts and Pb-free solder and may take advantage of certain exemptions referenced within the Directive.
- The Material Declaration Datasheets (MDDS) is available on request.

Electromagnetic Compatibility (EMC) emissions:

- CISPR22/EN55022 Class A
- EN300386
- FCC Rules CFR 47 Part 15B Class A
- ICES-003 Class A

Electromagnetic Compatibility (EMC) immunity:

- CISPR24/EN55024
- EN300386

Network Equipment Building System (NEBS) compliance:

- The test methodology is a combination of Intel and NEBS test requirements with the intent that the product will pass pure system-level NEBS testing.
- Intel has performed NEBS testing from GR-1089-CORE and GR-63-CORE that is applicable at the board level. To comply with GR-1089, the following statement must be placed on the system: “Caution: This system contains ESD sensitive device. Care must be applied during installation or removal of component.”
- The Ethernet ports on the front panel of the MPCBL0050 SBC must used shielded Ethernet cable and both ends of the shield must be grounded.
- Intel has performed testing from ETSI 300 019 that is applicable at the board level.

United States Export Classification:

- 5D002 Unrestricted

14.0 Safety Warnings

14.1 Safety Precautions

Review the following precautions to avoid personal injury and prevent damage to this product or products to which it is connected. To avoid potential hazards, use the product only as specified.

Read all safety information and understand the precautions associated with safety symbols, written warnings, and cautions before accessing parts or locations within the unit.

SYSTEM FOR RESTRICTED ACCESS USE ONLY!

Warning: To avoid the risk of electrical shock hazard, special measures and precautions must be taken when using these products:

- Access to this equipment must be restricted by locating this equipment where access can only be gained by SERVICE PERSONNEL who have been informed about the reasons for the restrictions applied to the location and about any precautions that shall be taken. Access is through the use of a TOOL, lock and key, or other means of security and is controlled by the authority responsible for the location.
- This product should only be used by SERVICE PERSONNEL who have the knowledge and training required to work with products of this type.
- To avoid shock, ensure that the chassis power cables are connected to a properly wired and grounded receptacle.
- The system containing these boards should not be operated with the faceplates, blank panels, or covers removed. Some voltages, that are on the board and inside the chassis, present an electrical shock and/or energy hazard to the user. Keep hands out of the chassis when power is applied or when performing hot swap of the boards.

Warning: To meet safety requirements, the digital ground must be connected to the chassis ground if the board is operating in an environment with an operating voltage at or above -60V. For further explanation of this matter, refer to the *Official Customer Notification of Changes to the Default Grounding Configuration of MPCBL0050 Products* document included with the MPCBL0050.

Caution: This equipment is designed to permit the connection of the earthed conductor of the d.c. supply circuit to the earthing conductor at the equipment. See chassis installation instructions.

15.0 Warranty Information

15.1 Intel NetStructure® Compute Boards and Platform Products Limited Warranty

Intel warrants to the original owner that the product delivered in this package will be free from defects in material and workmanship for two (2) year(s) following the latter of: (i) the date of purchase only if you register by returning the registration card as indicated thereon with proof of purchase; or (ii) the date of manufacture; or (iii) the registration date if by electronic means provided such registration occurs within 30 days from purchase. This warranty does not cover the product if it is damaged in the process of being installed. Intel recommends that you have the company from whom you purchased this product install the product.

THE ABOVE WARRANTY IS IN LIEU OF ANY OTHER WARRANTY, WHETHER EXPRESS, IMPLIED OR STATUTORY, INCLUDING, BUT NOT LIMITED TO, ANY WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, ANY WARRANTY OF INFRINGEMENT OF ANY OTHER PARTY'S INTELLECTUAL PROPERTY RIGHTS, OR ANY WARRANTY ARISING OUT OF ANY PROPOSAL, SPECIFICATION OR SAMPLE.

This warranty does not cover replacement of products damaged by abuse, accident, misuse, neglect, alteration, repair, disaster, improper installation or improper testing. If the product is found to be otherwise defective, Intel, at its option, will replace or repair the product at no charge except as set forth below, provided that you deliver the product along with a return material authorization (RMA) number (see below) either to the company from whom you purchased it or to Intel. If you ship the product, you must assume the risk of damage or loss in transit. You must use the original container (or the equivalent) and pay the shipping charge. Intel may replace or repair the product with either a new or reconditioned product, and the returned product becomes Intel's property. Intel warrants the repaired or replaced product to be free from defects in material and workmanship for a period of the greater of: (i) ninety (90) days from the return shipping date; or (ii) the period of time remaining on the original two (2) year warranty.

This warranty gives you specific legal rights and you may have other rights which vary from state to state. All parts or components contained in this product are covered by Intel's limited warranty for this product. The product may contain fully tested, recycled parts, warranted as if new.

15.2 Returning a Defective Product (RMA)

Before returning any product, contact an Intel Customer Support Group to obtain either a Direct Return Authorization (DRA) or Return Material Authorization (RMA). Return Material Authorizations are only available for products purchased within 30 days.

The following subsections contain return contact information by geography.

15.2.1 For the Americas

Return Material Authorization (RMA) credit requests e-mail address: requests.rma@intel.com

Direct Return Authorization (DRA) repair requests e-mail address: uspss.repair@intel.com

DRA on-line form: http://support.intel.com/support/motherboards/draform.htm

Intel Business Link (IBL): http://www.intel.com/ibl

Telephone No.: 1-800-INTEL4U or 480-554-4904

Office Hours: Monday - Friday 0700-1700 MST Winter / PST Summer

15.2.2 For Europe, Middle East, and Africa (EMEA)

Return Material Authorization (RMA) e-mail address EMEA.Returns@Intel.com

Direct Return Authorization (DRA) for repair requests e-mail address: EMEA.Returns@Intel.com

Intel Business Link (IBL): http://www.intel.com/ibl

Telephone No.: 00 44 1793 403063

Fax No.: 00 44 1793 403109

Office Hours: Monday - Friday 0900-1700 UK time

15.2.3 For Asia and Pacific (APAC)

RMA/DRA requests e-mail address: apac.rma.front-end@intel.com

Telephone No.: 604-859-3111 or 604-859-3325

Fax No.: 604-859-3324

Office Hours: Monday - Friday 0800-1700 Malaysia time

Return Material Authorization (RMA) requests e-mail address: rma.center.jpss@intel.com

Telephone No.: 81-298-47-0993 or 81-298-47-5417

Fax No.: 81-298-47-4264

Direct Return Authorization (DRA) for repair requests, contact the JPSS Repair center.

E-mail address: sugiyamakx@intel.co.jp

Telephone No.: 81-298-47-8920

Fax No.: 81-298-47-5468

Office Hours: Monday - Friday 0830-1730 Japan time

If the Customer Support Group verifies that the product is defective, they will have the Direct Return Authorization/Return Material Authorization Department issue you a DRA/RMA number to place on the outer package of the product. Intel cannot accept any product without a DRA/RMA number on the package. Limitation of Liability and Remedies

INTEL SHALL HAVE NO LIABILITY FOR ANY INDIRECT OR SPECULATIVE DAMAGES (INCLUDING, WITHOUT LIMITING THE FOREGOING, CONSEQUENTIAL, INCIDENTAL AND SPECIAL DAMAGES) ARISING FROM THE USE OF OR INABILITY TO USE THIS PRODUCT, WHETHER ARISING OUT OF CONTRACT, NEGLIGENCE, TORT, OR UNDER ANY WARRANTY, OR FOR INFRINGEMENT OF ANY OTHER PARTY'S INTELLECTUAL PROPERTY RIGHTS, IRRESPECTIVE OF WHETHER INTEL HAS ADVANCE NOTICE OF THE POSSIBILITY OF ANY SUCH DAMAGES, INCLUDING, BUT NOT LIMITED TO LOSS OF USE, BUSINESS INTERRUPTIONS, AND LOSS OF PROFITS. NOTWITHSTANDING THE FOREGOING, INTEL'S TOTAL LIABILITY FOR ALL CLAIMS UNDER THIS AGREEMENT SHALL NOT EXCEED THE PRICE PAID FOR THE PRODUCT. THESE LIMITATIONS ON POTENTIAL LIABILITIES WERE AN ESSENTIAL ELEMENT IN SETTING THE PRODUCT PRICE. INTEL NEITHER ASSUMES NOR AUTHORIZES ANYONE TO ASSUME FOR IT ANY OTHER LIABILITIES.

Some states do not allow the exclusion or limitation of incidental or consequential damages, so the above limitations or exclusions may not apply to you.

16.0 Customer Support

16.1 Customer Support

This chapter offers technical and sales assistance information for this product. Information on returning an Intel NetStructure® product for service is in the following chapter.

16.2 Technical Support and Return for Service Assistance

For all product returns and support issues, please contact your Intel product distributor or Intel Sales Representative for specific information.

16.3 Sales Assistance

If you have a sales question, please contact your local Intel NetStructure® Sales Representative or the Regional Sales Office for your area. Address, telephone and fax numbers, and additional information is available at Intel's web site located at:

http://www.intel.com/buy/networking/telecom.htm

16.4 Product Code Summary

Table 85 presents the product codes for the MPCBL0050 and companion Rear Transition Module (RTM).

Table 85. Product Code Summary

Product Code	MM#	Description
MPCBL0050N02Q	887727	Intel NetStructure® MPCBL0050 Single Board Computer with two Dual-Core Intel® Xeon® 5138 LV 2.13GHz processor (memory not included)
MPCBL0050S01Q	887996	Intel NetStructure® MPCBL0050 Single Board Computer with two Dual-Core Intel® Xeon® 5138 LV 2.13GHz processor (memory not included), NO FRONT PANEL MYLAR SKU
MPRTM0050S00Q	887726	Rear Transition Module for the Intel NetStructure® MPCBL0050 Single Board Computer (HDD not included)
MPRTM0050S01Q	887993	Rear Transition Module for the Intel NetStructure® MPCBL0050 Single Board Computer (HDD not included) NO FRONT PANEL MYLAR SKU

Appendix A Supported IPMI Commands

Table 86 shows the IPMI commands supported by the MPCBL0050 SBC.

Table 86. Supported IPMI commands (Sheet 1 of 6)

Command	NetFn	Cmd
IPM Device "Global" Commands		
Get Device ID	App	1
Cold Reset	App	2
Get Self Test Results	App	4
Set ACPI Power State	App	6
Get ACPI Power State	App	7
BMC Watchdog Timer Commands		
Reset Watchdog Timer	App	22h
Set Watchdog Timer	App	24h
Get Watchdog Timer	App	25h
BMC Device and Messaging Commands		
Set BMC Global Enables	App	2Eh
Get BMC Global Enables	App	2Fh
Clear Message Flags	App	30h
Get Message Flags	App	31h
Get Message	App	33h
Send Message	App	34h
Read Event Message Buffer	App	35h
Get System GUID	App	37h
Get Channel Authentication Capabilities	App	38h
Get Session Challenge (through LAN interface only)	App	39h
Activate Session (through LAN interface only)	App	3Ah
Set Session Privilege Level (through LAN interface only)	App	3Bh
Close Session (through LAN interface only)	App	3Ch
Get Session Info (through LAN interface only)	App	3Dh
Get AuthCode	App	3Fh

Table 86. Supported IPMI commands (Sheet 2 of 6)

Command	NetFn	Cmd
Set Channel Access	App	40h
Get Channel Access	App	41h
Get Channel Info	App	42h
Set User Access	App	43h
Get User Access	App	44h
Set User Name	App	45h
Get User Name	App	46h
Set User Password	App	47h
Activate Payload	App	48h
Deactivate Payload	App	49h
Get Payload Activation Status	App	4Ah
Get Payload Instance Info	App	4Bh
Set User Payload Access	App	4Ch
Get User Payload Access	App	4Dh
Get Chan Payload Support	App	4Eh
Get Chan Payload Version	App	4Fh
Get Chan OEM Payload Info	App	50h
Master Write Read I2C	App	52h
Get Chan Cipher Suites	App	54h
Suspend Payload Encryption	App	55h
Set Channel Security Keys	App	56h
Reserved	App	E0h
Chassis Device Commands – 00h		
Get Chassis Status	Chassis	01h
Chassis Control	Chassis	02h
Get System Restart Cause	Chassis	07h
Set System Boot Options	Chassis	08h
Get System Boot Options	Chassis	09h
Event Commands – 04h		
Set Event Receiver	S/E	00h
Get Event Receiver	S/E	01h
Platform Event (a.k.a. “Event Message”)	S/E	02h
PEF and Alerting Commands – 04h		
Get PEF Capabilities	S/E	10h
Arm PEF Postpone Timer	S/E	11h
Set PEF Configuration Parameters	S/E	12h
Get PEF Configuration Parameters	S/E	13h
Set Last Processed Event ID	S/E	14h
Get Last Processed Event ID	S/E	15h

Table 86. Supported IPMI commands (Sheet 3 of 6)

Command	NetFn	Cmd
Sensor Device Commands – 04h		
Get Device SDR Info	S/E	20h
Get Device SDR	S/E	21h
Reserve Device SDR Repository	S/E	22h
Set Sensor Hysteresis	S/E	24h
Get Sensor Hysteresis	S/E	25h
Set Sensor Threshold	S/E	26h
Get Sensor Threshold	S/E	27h
Set Sensor Event Enable	S/E	28h
Get Sensor Event Enable	S/E	29h
Re-arm Sensor Events	S/E	2Ah
Get Sensor Event Status	S/E	2Bh
Get Sensor Reading	S/E	2Dh
Intel OEM Commands (INTEL = 08h)		
Reserved	FWTR	00 thru 07h
FRU Device Commands – 0Ah		
Get FRU Inventory Area Info	Storage	10h
Read FRU Data	Storage	11h
Write FRU Data	Storage	12h
SDR Repository Commands – 0Ah		
Get SDR Repository Info	Storage	20h
Get SDR Repository Allocation Info	Storage	21h
Reserve SDR Repository	Storage	22h
Get SDR	Storage	23h
Add SDR	Storage	24h
Partial Add SDR	Storage	25h
Delete SDR	Storage	26h
Clear SDR Repository	Storage	27h
Get SDR Repository Time	Storage	28h
Enter SDR Repository Update Mode	Storage	2Ah
Exit SDR Repository Update Mode	Storage	2Bh
Run Initialization Agent	Storage	2Ch
SEL Device Commands – 0Ah		
Get SEL Info	Storage	40h
Get SEL Allocation Info	Storage	41h
Reserve SEL	Storage	42h
Get SEL Entry	Storage	43h
Add SEL Entry	Storage	44h
Partial Add SEL Entry	Storage	45h

Table 86. Supported IPMI commands (Sheet 4 of 6)

Command	NetFn	Cmd
Delete SEL Entry	Storage	46h
Clear SEL	Storage	47h
Get SEL Time	Storage	48h
Set SEL Time	Storage	49h
LAN Device Commands – 0Ch		
Set LAN Configuration Parameters	Transport	01h
Get LAN Configuration Parameters	Transport	02h
Suspend BMC ARPs	Transport	03h
Get IP/UDP/RMCP Statistics	Transport	04h
Serial/Modem Device Commands – 0Ch		
Set Serial/Modem Configuration	Transport	10h
Get Serial/Modem Configuration	Transport	11h
Set Serial/Modem Mux	Transport	12h
Serial/Modem Connection Active	Transport	18h
Callback	Transport	19h
Set SOL 2.0 Configuration	Transport	21h
Get SOL 2.0 Configuration	Transport	22h
AdvancedTCA™ - 2Ch		
Get PICMG Properties	PICMG*	00h
Get Address Info	PICMG	01h
FRU Control	PICMG	04h
Get FRU LED Properties	PICMG	05h
Get LED Color Capabilities	PICMG	06h
Set FRU LED State	PICMG	07h
Get FRU LED State	PICMG	08h
Set IPMB State	PICMG	09h
Set FRU Activation Policy	PICMG	0Ah
Get FRU Activation Policy	PICMG	0Bh
Set FRU Activation	PICMG	0Ch
Get Device Locator Record ID	PICMG	0Dh
Set Port State	PICMG	0Eh
Get Port State	PICMG	0Fh
Compute Power Properties	PICMG	10h
Set Power Level	PICMG	11h
Get Power Level	PICMG	12h
Get Fan Speed Properties	PICMG	14h
Get IPMB Link Info	PICMG	18h
Set AMC Port State (AMC.0)	PICMG	19h
Get AMC Port State (AMC.0)	PICMG	1Ah

Table 86. Supported IPMI commands (Sheet 5 of 6)

Command	NetFn	Cmd
Intel OEM Commands (INTEL = 30h)		
Change BIOS Boot Flash	INTEL	01h
Reserved	INTEL	05h
Reserved	INTEL	06h
Get Version Data	INTEL	07h
Reserved	INTEL	0Bh thru 0Fh
Reserved	INTEL	18h thru 20h
Get DIMM State	INTEL	21h
Set DIMM State	INTEL	22h
ReArm DIMMs	INTEL	23h
Sync SMBus Arbitration	INTEL	25h
Set Processor Status	INTEL	28h
Get Processor Status	INTEL	29h
ReArm Processors	INTEL	2Ah
Disable FRB Action	INTEL	2Bh
Get Serial Port Capture Data	INTEL	30h
Get Serial Port Capture Configuration	INTEL	31h
Set Serial Port Capture Configuration	INTEL	32h
Set System GUID	INTEL	41h
SetAuxChannelInfo	INTEL	42h
SetGetSDRTransChans	INTEL	43h
Log Post Code	INTEL	47h
SEL Internal Platform Event	INTEL	48h
Set SM Signal	INTEL	50h
Get SM Signal	INTEL	51h
Get Self Test History	INTEL	52h
Manufacturing Test Mode	INTEL	54h
IPMI Commands History Get	INTEL	60h
IPMI Commands History Clear	INTEL	61h
Graceful OS Shutdown	INTEL	70h
Init Agent Started	INTEL	80h
Init Agent End	INTEL	81h
Get ACPI Configuration	INTEL	82h
Set ACPI Configuration	INTEL	83h
Reserved	INTEL	86h thru 8Ch
Reserved	INTEL	A0h
Reserved	INTEL	A1h
Reserved	INTEL	A2h
Reserved	INTEL	A3h
Reserved	INTEL	A4h

Table 86. Supported IPMI commands (Sheet 6 of 6)

Command	NetFn	Cmd
Reserved	INTEL	A5h
Reserved	INTEL	A6h
Reserved	INTEL	A7h
Reserved	INTEL	A8h
Get NMI Source	INTEL	E6h
Set NMI Source	INTEL	EDh
NMI Enable Disable	INTEL	F7h
Get Latest Port80 Codes	INTEL	FAh
Other Intel OEM Commands: PLAT=32h		
Get HW Info	PLAT	01h
Get Power Unit Status	PLAT	02h
Reserved	PLAT	03h
Reserved	PLAT	55h
Reserved	PLAT	57h
Other Intel OEM Commands: PLAT=3Ah		
LAN Mux Settings	PLAT	26h

Appendix B Reference Documents

The following documents should be available when using this specification. Documents that are not available on web sites may be obtained from your Intel Business Link (IBL) account, or contact your Intel Field Sales Engineer (FSE) or Field Application Engineer (FAE).

- PICMG AdvancedTCA Specification (http://www.advancedtca.org)
- PICMG Advanced Mezzanine Card Specification (http://www.picmg.org)
- Renesas H8S/2168 Group Product Specification (http://www.renesas.com/fmwk.jsp?cnt=h8s2168_root.jsp&fp=/products/mpumcu/h8s_family/h8s2100_series/h8s2168_group/)

The following Intel Corporation documents may be required for more detailed information:

- Intel® 82571EB Gigabit Ethernet Controller Datasheet (http://www.intel.com/design/network/products/lan/controllers/82571eb.htm)
- Intel® Boot Agent. (http://www.intel.com/support/network/adapter/pro100/bootagent/userguide3/index_dm.htm)
- Intel NetStructure® MPCHC0001 14U Shelf Technical Product Specification (http://www.intel.com/design/network/products/cbp/atca/MPCHC0001.htm)
- Intel NetStructure® MPCMM0001 Chassis Management Module Hardware Technical Product Specification (http://www.intel.com/design/network/products/cbp/atca/mpcmm0001.htm)
- Intel NetStructure® MPCMM0001 Chassis Management Module Software Technical Product Specification (http://www.intel.com/design/network/products/cbp/atca/mpcmm0001.htm)
- Intel products for AdvancedTCA* (http://developer.intel.com/technology/atca/)
- Intelligent Platform Management Interface v2.0 Specification (http://developer.intel.com/design/servers/ipmi/spec.htm)
- Intelligent Platform Management Interface Implementer's Guide (http://developer.intel.com/design/servers/ipmi/spec.htm)
- ITP700 Debug Port Design Guide (http://www.intel.com/design/xeon/documentation.htm)
- Extensible Firmware Interface (EFI): http://www.intel.com/technology/efi/