BladeServer

Base Specification For Processor Blade Subsystems

Document Version: 2.41
Document Date: 12 March 2009

Confidential Document

The information in this document is IBM/Intel Confidential and is protected by a Confidential Disclosure Agreement when disclosed to individuals in other corporations or entities.

Table of Contents

1 Preface

1.1 Introduction

This document specifies the mechanical, electrical, logical, management and functional aspects of a Processor Blade for compatibility with BladeServer chassis platforms and related subsystems. The term "Processor Blade" is generic and may also be applicable to special function blades, storage blades or I/O blades.

1.2 Document Control

All approved levels are 1.x and higher. The document is only available in PDF format.

1.3 Version Levels

Version	Date	Reason
1.21	02/08/2005	Draft for approval
1.20	9/24/2004	Approved version

1.4 Document Change History

Document change history will be maintained for versions 1.x and greater.

Version	Date	Modified	Reason
1.21	02/08/2005	*	Add BIOS-BMC interface • Added section 2.9 Blade BIOS • Added section 2.10.1 BIOS and VPD fields • Added section 5.19.4 BIOS-BMC OEM IPMI commands
1.22	07/12/2005	*	• Added several sections in Blade BIOS to BMC interfaces • Updated various errors and clarifications • Added new OEM commands and Controller specific examples • Added grids on Board Failure LEDs and behavior table
1.23	11/20/2005	*	• Corrections to 'Get Throttle Status' Command • Corrections to 'Get Watt Usage' Command
2.04	12/2/2005	*	Added BladeServerH content • Added USB2.0 section 2.5.2 • Modified section 2.10 on Power Input Circuit • Modified Thermal in Chapter 3 • New mechanical drawings in Chapter 4 • Modified Mezzanine Card section 2.8.2 • Added section 2.7 cKVM
2.05	12/10/2005	*	Added section 3.3 Component Temperature

			Reporting Added 2.3.2 High Speed I/O Expansion Card interface to midplane
2.06	1/19/2006	*	Modified USB2.0, modified Thermal in Chapter 3, modified mechanical in Chapter 4
2.20	8/07/2007	DH/JH	Update to Table 3.1 and 3.3, Added Telco 12U (BladeServer HT) content to Section 2.1 and Chapter 3, Added maximum blade weight to Secion 4.3, Added connector tables 2.1 and 2.3, Updated Figure 4.1 to reference tables 2.1 and 2.3, Added Figure 4.2b and updated Figure 4.2a to reference Figure 4.2b, Updated Figure 2.3, Changed reference from "daughter card" to "I/O expansion card" in Sections 2.9, 5.1.2, 5.5-5.7.5, 5.11, Table 5.14, and Chapter 6,
2.20	8/07/2007	JEB	Added section 1.6. Chapter 5 updates for additional FLR, Entity Id's and OEM SDR's. Power thermal management EPOW handling. Table 5.28 IPMI OEM command descriptions updates and additions. Table 5.2 remove override.
2.20	8/07/2007	SV	Updated Section 2.3.1 for 11-row VHDM connector for Network Clock support. Updated telecom requirements section.
2.21	11/15/2007	JEB	Changed to version 2.21 Added changes for FRU data handling section 5.11. Fixed Section 5.27.2 Command Table. Removed one BMC capability bit.
2.30	5/13/2008	*	Added Sections 4.3.3 Reserving Label Space for End User, and 5.7.5 Power Restore Policy. Updated description of front panel switches. Updated section 2.5 with chassis that support each speed. Updated tables 2.1, 2.2, 2.3, 5.16, 5.22.
2.41	3/12/2009	HA	Added figure 3.4 and deleted note on table 5.17.12.

1.5 Change Frequency

This document will be updated to reflect changes and updates that are approved by the joint Intel/IBM Collaboration Architecture Review Board.

1.6 Vital Product Data Field for 'Base Specification Major and Minor Version'

Below is the information that must be initialized into the components Vital Product Data (VPD) field described as *'Base Specification Major and Minor Version*'. For a *Processor*

Blade component this is the equivalent of the information as specified in the ‘Document Version’ on the title page of this specification. See ‘Base Specification for VPD’ for additional information.

Table 1-1 Component VPD Field for 'Base Specification Major and Minor Version'

VPD Field Location	Hex Value in VPD	Displayed ASCII value on MM Management Interfaces
Block1 (Dynamic Block Controller Area) byte offset: 01B2h	0228h	‘2.41’

2 Processor Blade Design

2.1 Functional Overview

This document describes the architecture (mechanical and electrical) and interfaces of a Processor Blade to other subsystems within a BladeServer Chassis via the Mid-plane. A Processor Blade is a subset of an industry standard server that is implemented on a card with a protective cover and which plugs into a Mid-plane in the BladeServer (BC) chassis.

Each Processor Blade plugs into the Mid-plane via two 60-pin VHDM connectors and two 12V power connectors. In most cases, a blade is implemented as a single card and occupies one slot. In some cases, the Processor Blade may take up more than one slot and use more than two sets of connectors on the Mid-plane. Blades are installed into a rack-mounted chassis which is designed specifically for these cards. The Processor Blade subsystem can use any processor that complies with the mechanical and electrical interfaces of other sub-systems within a chassis and also meets the power and cooling requirements.

The Processor Blade has an enclosure, based on the InfiniBand specification, which houses the board and components. The enclosure is an InfiniBand double high (6 U), single width (29 mm enclosure. However, it has a custom depth of approximately 446 mm. Air flow is from front to back.

The BladeServer-E chassis is a 7U rack-mounted enclosure which provides fourteen (14) slots for 29mm single-wide Processor blades. The BladeServer-H chassis is a 9U rack-mounted enclosure which also supports fourteen (14) 29mm blades, but also provides four (4) 20mm bays for four (4) high-speed switches. The BladeServer-T chassis is an 8U rack-mounted enclosure which provides eight (8) slots for the processor blades while the BladeServer-HT chassis has slots for 12 blades, but also provides four (4) 20mm bays for four (4) high-speed switches. These chassis also provide the interconnections, via a midplane, between the blades, power supplies, blowers, Management Modules, high-speed communication ports, and shared resources such as front and rear customer interface LEDs, CD-ROM drive (not available on 12U chassis), and diskette drive (not available on 9U or 12U chassis). Although the FDD (diskette drive) is not available on all chassis, the devices in the media tray will still be referred to as USB attached devices.

Processor Blades interface with other subsystems in a chassis via the following:

RS485
Analog video
USB
High Speed SERDES (such as Gigabit Ethernet)
Miscellaneous control signals
12V continuous

These interfaces provide the ability to communicate with other subsystems in a BladeServer chassis, such as Management Modules, the Media tray and Switch modules (e.g. Ethernet, Fiber Channel or Infiniband). These interfaces are doubled on the Mid-plane to provide redundancy for single points of failure. Each Processor Blade has a front panel with status LEDs and multiple push button switches for power, media, KVM or cKVM, reset, and NMI.

2.2 Typical Front panel customer interface (Optional)

The front panel design defines the following LEDs and push button switches:

LEDs

- Power-on LED – Green
- Activity LED – Green
- Location LED – Blue
- Information LED – Amber
- Blade Server error (Fault) LED – Amber
- KVM LED (illuminates the push button) – Green
- Media LED (illuminates the push button) – Green

Push buttons

- Off/On (Power-control) Button
- KVM Select Button
- Media Select Button
- NMI (Reset) Button

The Power button should be protected from being accidentally pressed. The KVM and Media LEDs and Select Buttons should operate while the blade is powered off. The Location, Blade Fault and Information LEDs should illuminate when the blade is powered off (in standby).

The positions and icons for the LEDs and push buttons are as shown in Figure 2-1. If any LED or push button is not implemented, the relative positions of the remaining LEDs and push buttons should be maintained.

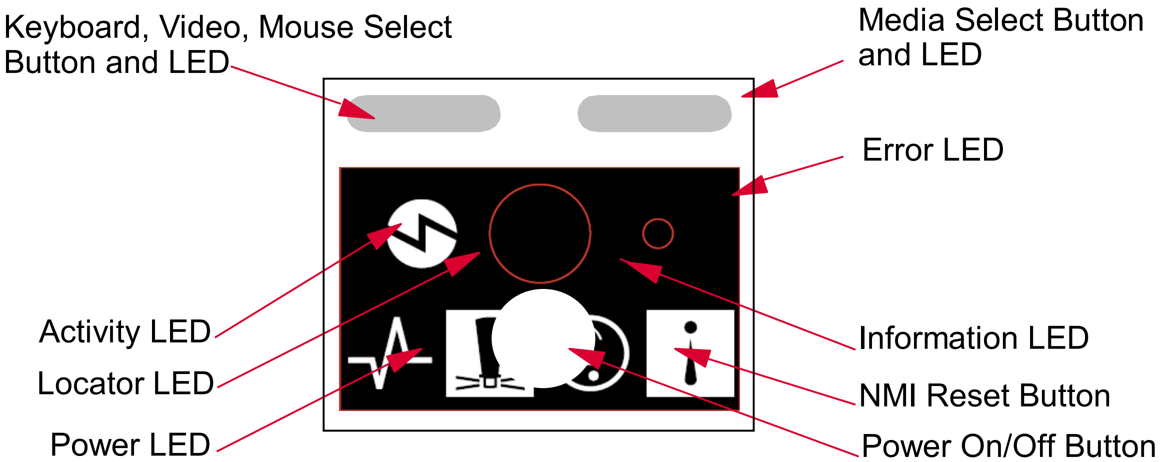


Figure 2-1 - Front panel layout

2.3 Mid-plane connector pinouts

2.3.1 Connectors to Mid-plane for legacy switch modules (Required)

For redundancy, the midplane supports two sets of signal and power connectors that are provided on the back of each blade. The signal connector, power connector, and their midplane mating connectors may be found in Table 2-1. The 10-row connectors (looking from the midplane to the back of the blade) are as shown in Figure 2-2; 11-row connectors are shown in Figure 2-3. The pinout descriptions for both are listed in Table 2-3.

Table 2-1 - Processor Blade Signal and Power Connectors (10-row)

Connector Description	Source A		Source B		Note
	Vendor	Part Number	Vendor	Part Number	
Blade Signal Connector	Molex	74030-9973	Amphenol TCS	AV600-00018	Part of Blade
Mating Blade Signal Connector on Midplane	Molex	74074-9987	Amphenol TCS	498-5010-022	Mating connector on Midplane
Blade Power Connector	Molex	87684-2001	--	--	Part of Blade
Mating Blade Power Connector on Midplane	Molex	87680-2001	FCI	10011570-001LF	Mating connector on Midplane
Note: Refer to *BladeServer_Connectors_20071211.pdf* located on the Blade Open Spec Support Center website (http://www-03.ibm.com/systems/bladecenter/open_specs.html) for most up to date connector list.					

The 11-row VHDM connector adds a row adjacent to the power connector to allow for network clock distribution between blade servers. The signals are identical on each blade slot and provide a shared clock distribution network. Each clock signal has a Positive and Negative pair. The midplane provides 100 ohm termination between the balanced pair signals.

Table 2-2 - Processor Blade Signal Connectors (11-row)

Connector Description	Source A		Source B		Note
	Vendor	Part Number	Vendor	Part Number	
Blade Signal Connector	Molex	74030-9871	--	--	Part of Blade
Mating Blade Signal Connector on Midplane	Molex	74074-9864	--	--	Mating connector on Midplane
Note: Refer to *BladeServer_Connectors_20071211.pdf* located on the Blade Open Spec Support Center website (http://www-03.ibm.com/systems/BladeServer/open_specs.html) for most up to date connector list.					

Note: The extra rows of pins do not change the relative position of the 10-row connector and the power connector or the guide pin.

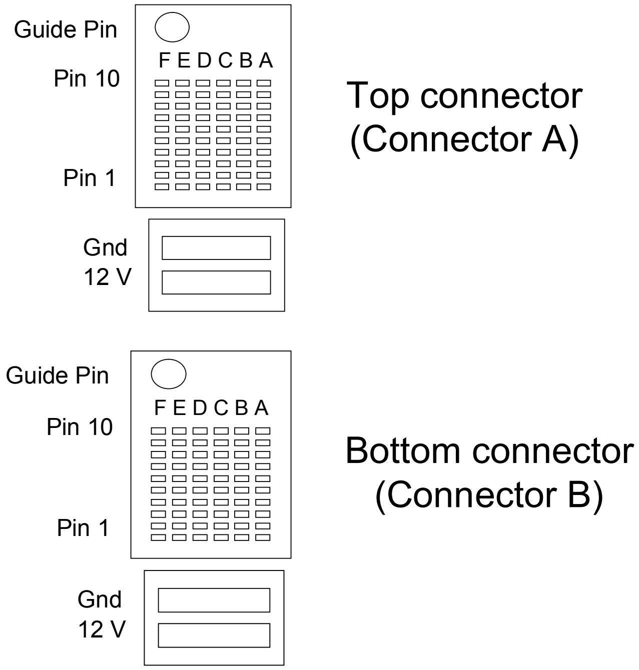


Figure 2-2 - Processor Blade Connector Layout (10-row)

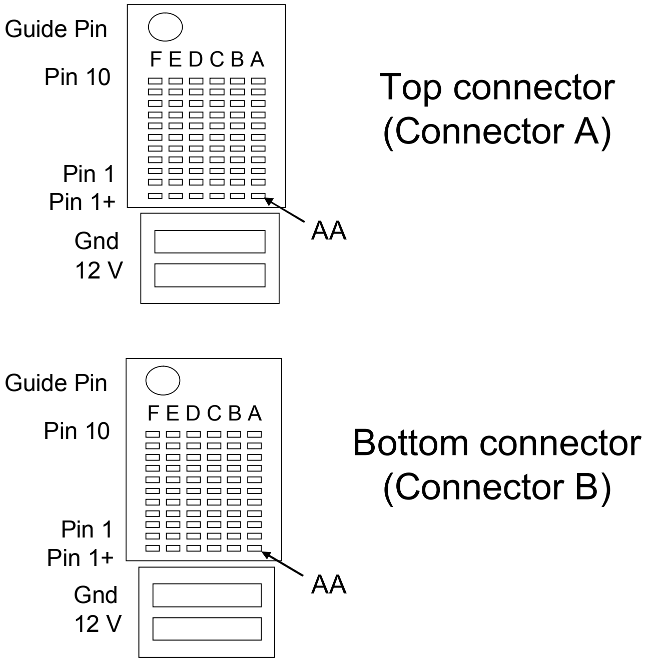


Figure 2-3 - Processor Blade Connector Layout (11-row)

Table 2-3 - Processor Blade Connector Pin Assignments

Mid-plane Processor Blade Connector Pin Assignments			
Pin	**Name**	**I/O**	**Purpose**
AA1	CLK2_[A/B]_P	I/O	Network CLK2[A/B]: Sense P (Note 6)
A1	GND		Ground
A2	BLADE_TXP[PBnum]_[1/2]	O	SERDES transmit-positive signal for the transmit differential pair. This goes to switch Module [1/2] for the [Top/Bottom] connector.
A3	GND		Ground
A4	USB_KM_[A/B]	I/O	USB+ signal for Keyboard/Mouse USB Bus.
A5	NC		No Connection
A6	BMC_RS_485_RESET[A/B}_N	I	Signal from the Management Module to reset the Base Management Controller (e.g. H8). Implement circuit per reference schematic. See Note 4.
A7	ADDR3_[PBnum][A/B]	I	Slot Address Bit 3. See Slot Address Table 2.8
A8	ADDR0_[PBnum][A/B]	I	Slot Address Bit 0. See Slot Address Table 2.8
A9	MM_SELECT_[A/B]	I	Select line used to determine which set of I2C and RS-485 buses to use. See Note 4.
A10	GND		Ground
BB1	CLK2_[A/B]_N	I/O	Network CLK2[A/B]: Sense N (Note 6)
B1	GND		Ground
B2	BLADE_TXN[PBnum]_[1/2]	O	SERDES transmit- negative signal for the transmit differential pair. This goes to switch Module [1/2] for the [Top/Bottom] connector.
B3	BLADE_RXP[PBnum]_[1/2]	I	SERDES receive-positive signal for the receive differential pair. This goes to switch Module [1/2] for the [Top/Bottom] connector.
B4	USBN_KM_[A/B]	I/O	USB- signal for Keyboard/Mouse USB Bus
B5	ADDR2_[PBnum][A/B]	I	Slot Address Bit 2. See Slot Address Table 2.8
B6	RS_485_DEGATE[A/B]_N	I	Signal from Management Module to degate the blade from the RS485 bus. See Note 4.
B7	NC		See note 5.
B8	ADDR1_[PBnum][A/B]	I	Slot Address Bit 1. See Slot Address Table 2.8
B9	DDC_DAT_[A/B]_N	O	DDC Data signal for video monitor
B10	DDC_CLK_[A/B]_N	O	DDC Clock signal for video monitor
CC1	CLK1_[A/B]_P	I/O	Network CLK1[A/B]: Sense P (Note 6)
C1	GND		Ground
C2	GND		Ground
C3	BLADE_RXN[PBnum]_[1/2]	I	SERDES receive-negative signal for the receive differential pair. This goes to switch Module [1/2] for the [Top/Bottom] connector.
C4	GND		Ground
C5	GND		Ground
C6	NC		See note 5.
C7	EPOW_N		Emergency Power Off Warning
C8	VSYNC[PBnum]_[A/B]	O	Video vertical sync signal
C9	HSYNC[PBnum]_[A/B]	O	Video horizontal sync signal
C10	GND		Ground
DD1	CLK1_[A/B]_N	I/O	Network CLK1[A/B]: Sense N (Note 6)

D1	GND		Ground
D2	BLADE_TXP[PBnum]_[3/4]	O	I/O Expansion Card SERDES transmit-positive signal for the transmit differential pair. This goes to switch Module [3/4] for the [Top/Bottom] connector.
D3	GND		Ground
D4	USB_CDFDD[PBnum]_[A/B]	I/O	USB+ signal for Media USB bus
D5	485_[A/B]_RX_TX+	I/O	RS-485 RxTx+ signal
D6	NC		See note 5.
D7	GND		Ground
D8	GND		Ground
D9	GND		Ground
D10	CD_SELECT_[PBnum]_[A/B]_N	O	With permission from the Management Module (through RS485 bus), the Blade pulls this signal low to use the media tray. The FET switch it controls is on the Mid-plane. See Note 4.
EE1	CLK3_[A/B]_P	I/O	Network CLK3[A/B]: Sense P (Note 6)
E1	GND		Ground
E2	BLADE_TXN[PBnum]_[3/4]	O	I/O Expansion Card SERDES transmit-negative signal for the transmit differential pair. This goes to switch Module [3/4] for the [Top/Bottom] connector.
E3	BLADE_RXP[PBnum]_[3/4]	O	I/O Expansion Card SERDES receive-positive signal for the transmit differential pair. This goes to switch Module [3/4] for the [Top/Bottom] connector.
E4	USBN_CDFDD[PBnum]_[A/B]	I/O	USB- signal for Media USB bus
E5	485_[A/B]_RX_TX-	I/O	RS-485 RxTx- signal
E6	NC		See note 5.
E7	BLUE[PBnum]_[A/B]	O	Analog Video Blue signal
E8	GREEN[PBnum]_[A/B]	O	Analog Video Green signal
E9	RED[PBnum]_[A/B]	O	Analog Video Red signal
E10	GND		Ground
FF1	CLK3_[A/B]_N	I/O	Network CLK3[A/B]: Sense N (Note 6)
F1	GND		Ground
F2	GND		Ground
F3	BLADE_RXN[PBnum]_[3/4]	I	I/O Expansion Card SERDES receive-negative signal for the transmit differential pair. This goes to switch Module [3/4] for the [Top/Bottom] connector.
F4	GND		Ground
F5	GND		Ground
F6	PRES_BIT[PBnum][A/B]_N	O	Presence bit - pulled low on processor blade.
F7	GND		Ground
F8	GND		Ground
F9	GND		Ground
F10	VIDEO[PBnum]_SEL_[A/B]_N	O	With permission from the Management Module (through RS485 bus), the Blade pulls this signal low to use the KVM. See Note 4.

Notes:

1. For the [A/B], [1/2], and [3/4] designations, the top connector uses the first number/letter (A with 1 or 3) and the bottom connector the second (B with 2 or 4).
2. [PBnum] refers to the Processor Blade slot number for the connector.
3. Discrepancies between this table and the reference schematics are due to debug pins (i.e. listed in this table as NC, but named in the reference schematics).
4. The signals on pins A6, A9, B6, D10 and F10 are all pulled up to +5V on the midplane. If necessary, the blade should provide level conversion for these signals.
5. For the bottom connector only, these pins will be used for serial port signals: B7 is CTS, C6 is RTS, D6 is RXD, E6 is TXD.
6. AA1 through FF1 are optional. Used only on blades that implement network clocks.
   Use the 11-row VHDM connector for these applications (refer to Table 2-2 and Figure 2-3).

2.3.2 Connectors to Mid-plane for High Speed Switch Fabric (Mounted on I/O Expansion Card)

For redundancy, two sets of signal connectors are provided to the Mid-plane for the High Speed Switch Fabric. This signal connector and its mating connector are found in Table 2.4. The connectors (looking from the midplane to the back of the blade) are as shown in Figure 2.4, and the pinout descriptions are listed in Tables 2.5 and 2.6.

Table 2-4 - Processor Blade Connectors for High Speed Switch Fabric

Connector Description	Source A		Source B		Note
	Vendor	Part Number	Vendor	Part Number	
Mezzanine Card Conn., 54 pos. Rt. Angle Header (Midplane Connector)	FCI	10058229-101LF	--	--	Part of High Speed I/O Expansion Card
Mating Conn. For Mezzanine Card Conn.	FCI	10039850-101LF	--	--	Mating connector on Midplane
Note: Refer to *BladeServer_Connectors_20071211.pdf* located on the Blade Open Spec Support Center website (http://www-03.ibm.com/systems/bladecenter/open_specs.html) for most up to date connector list.					

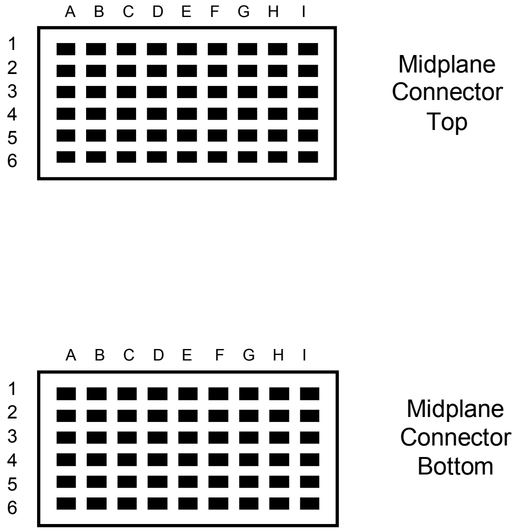


Figure 2-4 - Connectors to Mid-plane for High Speed Switch Fabric

Table 2-5– Top connector

Signal Name	Pin #	Type	Description
TX_HSSM1_LN1_P	H4	OUT	to HSSM1, Lane 1 (True)
TX_HSSM1_LN2_P	G3	OUT	to HSSM1, Lane 2 (True)
TX_HSSM1_LN3_P	H2	OUT	to HSSM1, Lane 3 (True)
TX_HSSM1_LN4_P	G1	OUT	to HSSM1, Lane 4 (True)
TX_HSSM1_LN1_N	I4	OUT	to HSSM1, Lane 1 (Complement)
TX_HSSM1_LN2_N	H3	OUT	to HSSM1, Lane 2 (Complement)
TX_HSSM1_LN3_N	I2	OUT	to HSSM1, Lane 3 (Complement)
TX_HSSM1_LN4_N	H1	OUT	to HSSM1, Lane 4 (Complement)
RX_HSSM1_LN1_P	B6	IN	from HSSM1, Lane 1 (True)
RX_HSSM1_LN2_P	A5	IN	from HSSM1, Lane 2 (True)
RX_HSSM1_LN3_P	B4	IN	from HSSM1, Lane 3 (True)
RX_HSSM1_LN4_P	A3	IN	from HSSM1, Lane 4 (True)
RX_HSSM1_LN1_N	C6	IN	from HSSM1, Lane 1 (Complement)
RX_HSSM1_LN2_N	B5	IN	from HSSM1, Lane 2 (Complement)
RX_HSSM1_LN3_N	C4	IN	from HSSM1, Lane 3 (Complement)
RX_HSSM1_LN4_N	B3	IN	from HSSM1, Lane 4 (Complement)
TX_HSSM3_LN1_P	H6	OUT	to HSSM3, Lane 1 (True)
TX_HSSM3_LN2_P	G5	OUT	to HSSM3, Lane 2 (True)
TX_HSSM3_LN3_P	D3	OUT	to HSSM3, Lane 3 (True)
TX_HSSM3_LN4_P	E2	OUT	to HSSM3, Lane 4 (True)
TX_HSSM3_LN1_N	I6	OUT	to HSSM3, Lane 1 (Complement)
TX_HSSM3_LN2_N	H5	OUT	to HSSM3, Lane 2 (Complement)
TX_HSSM3_LN3_N	E3	OUT	to HSSM3, Lane 3 (Complement)
TX_HSSM3_LN4_N	F2	OUT	to HSSM3, Lane 4 (Complement)
RX_HSSM3_LN1_P	D5	IN	from HSSM3, Lane 1 (True)
RX_HSSM3_LN2_P	E4	IN	from HSSM3, Lane 2 (True)
RX_HSSM3_LN3_P	B2	IN	from HSSM3, Lane 3 (True)
RX_HSSM3_LN4_P	A1	IN	from HSSM3, Lane 4 (True)
RX_HSSM3_LN1_N	E5	IN	from HSSM3, Lane 1 (Complement)
RX_HSSM3_LN2_N	F4	IN	from HSSM3, Lane 2 (Complement)
RX_HSSM3_LN3_N	C2	IN	from HSSM3, Lane 3 (Complement)
RX_HSSM3_LN4_N	B1	IN	from HSSM3, Lane 4 (Complement)

Total pins: 32

Signal Name	Pin #	Description
GND	A2, A4, A6	Ground/return pins
GND	C1, C3, C5	
GND	D2, D4, D6	
GND	F1, F3, F5	
GND	G2, G4, G6	
GND	I1, I3, I5	
KEY	D1, E1, E6, F6	NC:
Total pins: 22		

Table 2-6– **Bottom connector**

Signal Name	Pin #	Type	Description
TX_HSSM2_LN1_P	H4	OUT	to HSSM2, Lane 1 (True)
TX_HSSM2_LN2_P	G3	OUT	to HSSM2, Lane 2 (True)
TX_HSSM2_LN3_P	H2	OUT	to HSSM2, Lane 3 (True)
TX_HSSM2_LN4_P	G1	OUT	to HSSM2, Lane 4 (True)
TX_HSSM2_LN1_N	I4	OUT	to HSSM2, Lane 1 (Complement)
TX_HSSM2_LN2_N	H3	OUT	to HSSM2, Lane 2 (Complement)
TX_HSSM2_LN3_N	I2	OUT	to HSSM2, Lane 3 (Complement)
TX_HSSM2_LN4_N	H1	OUT	to HSSM2, Lane 4 (Complement)
RX_HSSM2_LN1_P	B6	IN	from HSSM2, Lane 1 (True)
RX_HSSM2_LN2_P	A5	IN	from HSSM2, Lane 2 (True)
RX_HSSM2_LN3_P	B4	IN	from HSSM2, Lane 3 (True)
RX_HSSM2_LN4_P	A3	IN	from HSSM2, Lane 4 (True)
RX_HSSM2_LN1_N	C6	IN	from HSSM2, Lane 1 (Complement)
RX_HSSM2_LN2_N	B5	IN	from HSSM2, Lane 2 (Complement)
RX_HSSM2_LN3_N	C4	IN	from HSSM2, Lane 3 (Complement)
RX_HSSM2_LN4_N	B3	IN	from HSSM2, Lane 4 (Complement)
TX_HSSM4_LN1_P	H6	OUT	to HSSM4, Lane 1 (True)
TX_HSSM4_LN2_P	G5	OUT	to HSSM4, Lane 2 (True)
TX_HSSM4_LN3_P	D3	OUT	to HSSM4, Lane 3 (True)
TX_HSSM4_LN4_P	E2	OUT	to HSSM4, Lane 4 (True)
TX_HSSM4_LN1_N	I6	OUT	to HSSM4, Lane 1 (Complement)
TX_HSSM4_LN2_N	H5	OUT	to HSSM4, Lane 2 (Complement)

TX_HSSM4_LN3_N	E3	OUT	to HSSM4, Lane 3 (Complement)
TX_HSSM4_LN4_N	F2	OUT	to HSSM4, Lane 4 (Complement)
RX_HSSM4_LN1_P	D5	IN	from HSSM4, Lane 1 (True)
RX_HSSM4_LN2_P	E4	IN	from HSSM4, Lane 2 (True)
RX_HSSM4_LN3_P	B2	IN	from HSSM4, Lane 3 (True)
RX_HSSM4_LN4_P	A1	IN	from HSSM4, Lane 4 (True)
RX_HSSM4_LN1_N	E5	IN	from HSSM4, Lane 1 (Complement)
RX_HSSM4_LN2_N	F4	IN	from HSSM4, Lane 2 (Complement)
RX_HSSM4_LN3_N	C2	IN	from HSSM4, Lane 3 (Complement)
RX_HSSM4_LN4_N	B1	IN	from HSSM4, Lane 4 (Complement)
Total pins:	32		

Signal Name	Pin #	Description
GND	A2, A4, A6	Ground/return pins
GND	C1, C3, C5	
GND	D2, D4, D6	
GND	F1, F3, F5	
GND	G2, G4, G6	
GND	I1, I3, I5	
KEY	D1, E1, E6, F6	NC:
Total pins:	22	

2.4 RS485 bus Interface

The RS485 bus is used for multi-point communications between the Management Module and Processor Blades. The BladeServer RS485 bus uses a Multi Master Architecture, where each slave unit has a unique address and responds only to packets addressed by the MM to that unit.

The standard RS485 interface has two versions: half-duplex (single twisted pair) and full-duplex (dual twisted pair). BladeServer architecture uses a half-duplex interface via the Mid-plane to interconnect all of the devices. Thus, all of the blades and the Management Module must have drivers with tri-state outputs with slew rate control equivalent to the Max3483E as shown in Section 2.4.1.

A BladeServer chassis may have as many as 16 BMCs (SP[1]), one on each of the 14 Processor Blades and one on each of the two Management Modules. The communication between a Management Module and the Processor Blade SP (or BMC) is via one of two RS 485 buses (see Figure 2-5), with one Management module acting as the master or slave and the Processor Blade BMCs acting as slave or master.

[1] The SP (Service Processor) on the blade will be referred to as the BMC (Baseboard Management Controller) throughout this specification.

Management Modules communicate with Processor Blade BMCs via a command/response protocol over the RS485 bus. For redundancy, there are two RS485 buses (one RS485 bus per Blade connector) to communicate with Processor Blade BMCs. The Processor Blade is responsible for activating the correct interface to the top or bottom connector based on the state of the signals MM_SELECT_A and MM_SELECT_B.

The Management Module must be able to disable individual blades from utilizing the RS-485 bus when the Management Module determines that a blade is hanging this multi-drop communications bus. This allows the Management Module to recover control of the RS-485 bus for communication to other blades. For further information see section 5.17.6 RS485 Bus De-gating and BMC Reset.

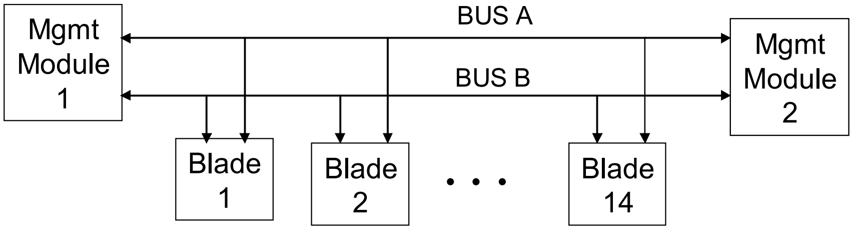


Figure 2-5 - Redundant RS485 busses

Since there are two Management Module slots per BladeServer chassis, a Processor Blade BMC is required to communicate to the Management Module on the appropriate RS485 Bus, A or B, as indicated by the select control signals in Table 2.7.   If both MM_SELECT_A and MM_SELECT_B signals are inactive (see Table 2.7), the Processor Blade BMC should deactivate the RS485 bus interface. If for any reason the Management Module is inactive or disabled, each Processor Blade is required to maintain status of all buses based on status of these signals. This is critical because Management Module failure or removal should not affect the operation of the Processor Blades. This implies that if the Management Module is inactive, the BladeServer chassis configuration cannot be changed.

MM_SELECT_A	MM_SELECT_B	Bus State
0	0	Degate Buses
1	0	Bus A active
0	1	Bus B active
1	1	Degate Buses

Table 2-7 - RS485 Bus Selection

Each Processor Blade BMC must be addressed based on fixed slot addresses. The addresses are hardwired by slot and are used by the Processor Blade BMC to determine which Processor Blade is being addressed on the bus and also the slot number they are plugged into.

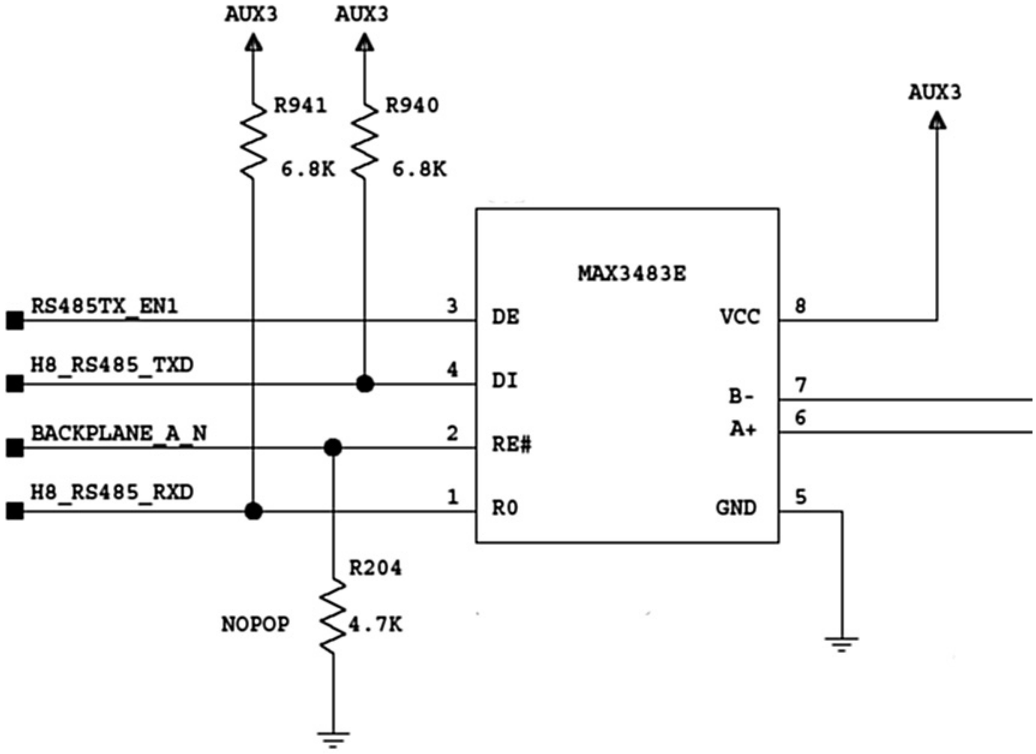
Note: Processor Blades that occupy more than one slot must respond on the RS485 bus with the lowest slot address of the slots occupied and only that address. For example a Processor Blade occupying slots 3, 4 and 5 must communicate on the RS485 bus using slot address 3.

Slot addresses for the Processor Blades are shown in Table 2.8.

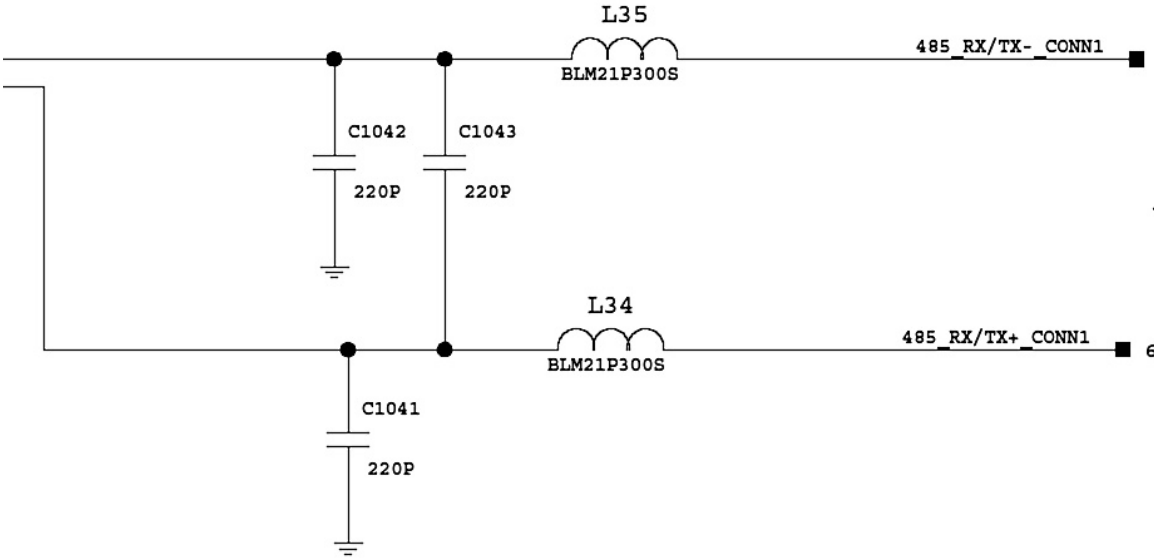
Slot Number	Address Pins A3 A2 A1 A0	Processor Blade
0	0000	Not used
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
10	1010	10
11	1011	11
12	1100	12
13	1101	13
14	1110	14
15	1111	Not Used

Table 2-8 - Processor Blade Slot Address Assignments

2.4.1 RS485 Reference Schematic



Continued from previous figure:



Refer to section 5.17.3 for more information.

2.5 USB (Optional)

2.5.1 USB 1.1

Chassis supporting USB 1.1: BC-E (Original).

The USB bus connects USB devices to a USB host. The physical interconnect is a tiered star topology. A USB hub is at the center of each star and it is point-to-point between host and hub or device or hub connected to another hub or device. There is only one host per USB subsystem. USB is capable of addressing up to 127 devices. The USB busses used within the BladeServer chassis conform to the Universal Serial Bus 1.1 specification and operate at 12 MB/sec. BladeServer uses the USB subsystem for shared resources such as Keyboard, Mouse, CD-ROM and Floppy Drive USB devices between the Processor Blades. These devices are connected to the Processor Blades via the Mid-plane.

The USB bus is a four-wire interface; Vbus, D+, D- and GND. The signaling occurs over two wires D+ and D- and Vbus is +5V Power. BladeServer implements two wire D+ and D- on the Mid-plane. Processor Blades do not provide power to USB devices because USB hubs are self-powered on the Management Module and Media Tray.

Processor Blades supporting USB must support at least four USB root ports, which are connected to 4 USB buses onto the Mid-plane via two connectors. There are two USB buses per each connector for redundancy. Two USB buses are routed to both Management Module slots on the Mid-plane from each Processor Blade (one bus per connector). The other two buses are routed to the Media tray, one per each connector from Processor Blade.

The USB bus routed to the Management Module is used for Keyboard and Mouse. This allows the Management Module to connect Keyboard/Mouse to the Processor Blade and control the routing of the Keyboard/Mouse inputs to one of fourteen Processor Blades. The other USB bus is routed to the Media Tray, which connects to the CD-ROM drive and diskette drive. Both USB buses are controlled by the Management Module independent of each other. This provides the capability to assign the CD and Floppy Disk drive to one Processor Blade while Keyboard/Mouse are assigned to another Processor Blade. However, this should not restrict the assignment of shared resources of both USB buses to a single Processor Blade.

The Management Module allows a user to request a shared resource in two different ways. One way is from an Administration console, and the other is by pressing a KVM and/or CD/FDD “Select” button on a Processor Blade. Pressing the “Select” button results in the Blade BMC requesting KVM and/or CD/FDD from the Management Module. The Management Module then issues commands to the Processor Blade BMC based on either the Administration console input or user selection. The Management Module deselects the current owner of the resource via a command before assigning the resource to a new owner.

The Processor Blade is responsible for activating the correct interface to top or bottom connector based the state of the signals MM_SELECT_A and MM_SELECT_B. If both select signals are inactive then Processor Blade should deactivate the USB bus interface. If for any reason the Management Module is inactive or disabled, each Processor Blade is required to maintain status of all buses based on status of these signals. This is critical because in event of Management Module failure or removal, it should not affect operation of the Processor Blades. This implies that if the Management Module is inactive, the BladeServer chassis configuration cannot be changed.

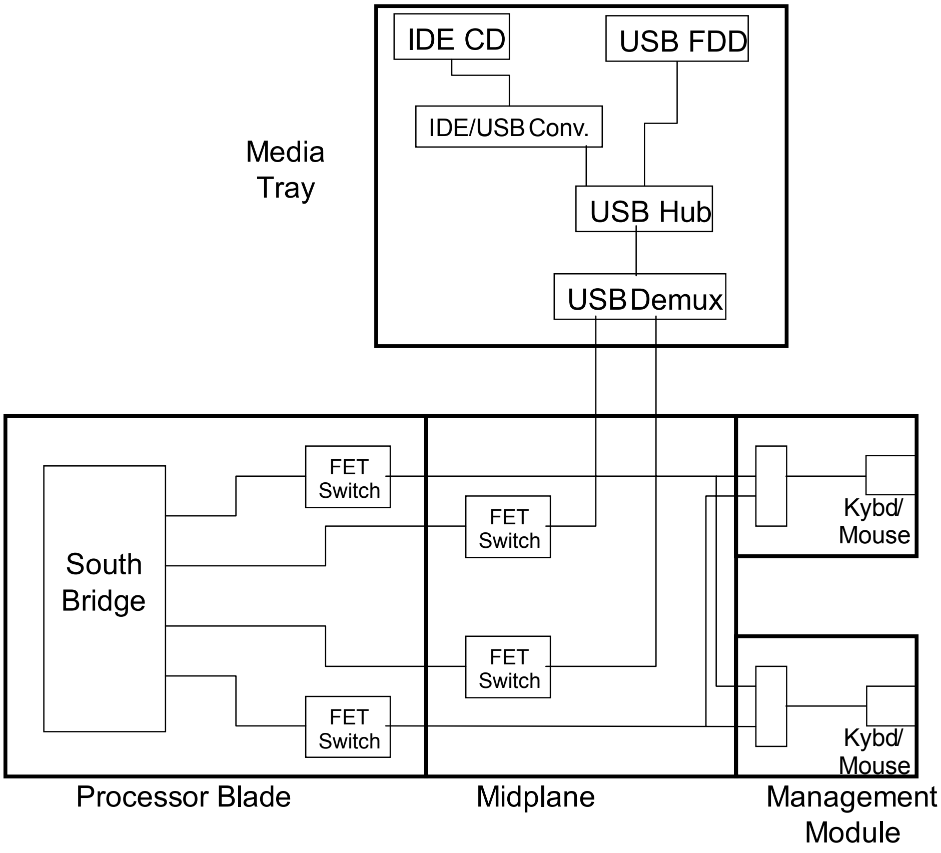


Figure 2-6 - Example of USB connections

2.5.2 USB 2.0

Chassis supporting USB 2.0 High Speed: BC-E (Refresh), BC-H, BC-HT.
Chassis supporting USB 2.0 Full Speed: BC-T (Refresh).

USB-2.0 is the next generation Universal Serial Bus that enables speeds up to 480 Mbps. This is approximately 40 times faster then USB-1.1 and is sometime referred to as high speed USB. USB-2.0 is backwards compatible with USB-1.1 and is also hot swappable. The blade can support up to two USB-2.0 interfaces. For redundancy, one USB-2.0 interface is routed to each midplane connector. The BCH midplane routes one USB-2.0 to each Management Module. Since only one MM can be active at a time, only one USB-2.0 will be active from the blade. Through the MM, the blade will have access to devices on the media tray, which includes the CD ROM and external USB Ports.

The USB bus is a four-wire interface; Vbus, D+, D- and GND. The signaling occurs over two wires D+ and D- and Vbus is +5V Power. BladeServer implements two wire D+ and D- on the

Mid-plane. Processor Blades do not provide power to USB devices, because USB hubs are self-powered on the Management Module.
USB-2.0 uses high speed current drivers to run at high speeds. When its 17.78ma driver is driving into a 45ohm termination, a 400mv differential is created. This is significantly lower than signals used in USB-1.1. Care must be taken when designing to these signal levels. The typical means of attaching a USB host to its devices is via well defined USB cables and connectors. The BladeServer uses its midplane's high speed FET switches and copper traces to perform this task. These high speed FET switches are configured to form a 14 to 1 mux and is controlled directly by the Management Module. Thus, some of the standard USB design rules do not apply, since there are no cables and connectors to use as a reference. To assure reliable operation at USB-2.0 speeds, the blades must adhere to the following design guidelines:

1. The USB-2.0 Driver/Receiver design must adhere to design guidelines as defined by chapter 7 of the Universal Serial Bus Specification (rev.2.0), unless otherwise stated.
2. Since the standard USB connectors are not used, all signal quality and compliance testing must be made close to the transceivers. This is referred to as TP1 and TP4 in the referenced USB spec.
3. Conformance to USB Templates 1,2,3 and 4 does not apply to the blade since standard USB cables and connectors are not used.
4. USB Template 5: Waveform requirements for hub transceiver measured at TP1 and device transceivers at TP4. Refer to the Chapter#7 of the USB specification for further details.
5. USB-2.0 Template 6: Receiver sensitivity requirement for hub transceiver measured at TP1 and device transceivers at TP4. Refer to the Chapter#7 of the USB specification for further details.
6. The maximum allowed trace length between the USB host and the connector must be less than **4.25** inches. The traces must be 90ohms differential impedance. Suggested path is 4 or 5 mil stripline.

The Processor Blade is responsible for activating the correct interface to the top or bottom connector based the state of the signals MM_SELECT_A and MM_SELECT_B. If both select signals are inactive, then the Processor Blade should deactivate the USB bus interface. If for any reason the Management Module is inactive or disabled, each Processor Blade is required to maintain status of all buses based on the status of these signals. This is critical because, in event of Management Module failure or removal, it should not affect operation of the Processor Blades. This implies that, if the Management Module is inactive, the BladeServer chassis configuration cannot be changed.

2.6 Video subsystem (Optional)

The BladeServer video subsystem has three parts:

A) Processor Blade VGA sub-system
B) Mid-plane Video bus
C) Management Module (Remote console and KVM switch)

The VGA subsystem is used for Processor Blade management, configuration and debug of Processor Blades either locally or remotely via an Administrator console, and as an OS and application interface for the user. The VGA subsystem must provide analog video output to a Mid-plane video bus. There are two completely independent video buses, A and B, on the Mid-plane. Processor Blades connect to Mid-plane video buses A and B via the two Mid-plane connectors. Only one video output may be active at a time to drive either Bus A or B.

Analog output of the Video controller chip drives RGB and HSYNC/VSYNC to sets of buffers. These buffers must be located on each Processor Blade and these buffers control the routing of RGB and HSYNC/VSYNC to either Mid-plane video bus A or B.

The Processor Blade is responsible for activating correct interface to top or bottom connector based the state of the signals MM_SELECT_A and MM_SELECT_B. If both _SELECT_A and MM_SELECT_B are inactive, the Processor Blade should deactivate the Video bus interface. If for any reason the Management Module is inactive or disabled, each Processor Blade is required to maintain the status of all buses based on status of these signals. In the event of Management Module failure or removal, operation of the Processor Blades should not be affected. This implies that, if the Management Module is inactive, the BladeServer chassis configuration cannot be changed.

The RGB analog video signals are very sensitive to noise and changes in trace topology. Because of this, the goal in routing the video signals on the Mid-plane is to maintain a point-to-point connection between the selected blade and the management module. Therefore, MUXes are used on the Mid-plane to switch sections of traces in and out of the video nets, depending on which blade is selected. Regardless of which blade is selected, a single point-to-point connection is made between the selected blade and the management module.

The Management Module allows a user to request a shared resource in two different ways. One way is from an Administration console, and the other is by pressing a KVM “Select” button on a Processor Blade. The Management Module selects a Processor Blade to assign KVM shared resource by issuing a command to the Processor Blade BMC. The Management Module deselects the current owner of the resource via a command before assigning the resource to a new owner.

2.6.1 Video Wiring Requirements

Red, Green, Blue Analog Signals

- Zo=75ohms
- 75 ohm +/-1% pulldown to GND on each of these signals. Place each resistor close to the source (video chip)
- For redundant Mid-planes, the Red, Green, and Blue signals need to be buffered (e.g. - AD8075 part) before going to each path on the Mid-plane. Be sure to follow the placement and routing requirements specified in the buffer's datasheet and application notes.
- When routing the Red, Green, and Blue signals to the 2 buffers, route the traces in a daisy chain fashion, first to one buffer, then to the other.
- There can be no more than 1" length mismatch between the Red, Green, and Blue signals and Hsync and Vsync.

Hsync, Vsync

- Buffer Hsync and Vsync before sending them out to the Mid-plane. An acceptable buffer chip is the AHCT125.
- Place a 1 MegOhm pulldown resistor to GND at the input of each buffer.
- The single-ended trace impedance for each of these signals should be 50 ohms

- Add a 22 ohm series resistor to each of these signals between the buffer outputs and the VHDM connectors. Place the resistors as close to the buffers as possible/practical.

General

- Place the video chip as close as possible to the back of the blade next to the top or bottom Mid-plane connector.
- For the video chip, follow the placement, routing and decoupling guidelines as specified in the video chip data book and associated application notes (e.g. Design Guide).

2.7 cKVM (Optional)

Processor Blades are not required to implement cKVM. This section describes our implementation.

Concurrent keyboard, video and mouse (cKVM) is the use of a mezzanine card on the Processor Blade in order to enable concurrent remote control of all of the Blades in a BladeServer chassis using KVM over IP. The mezzanine card contains hardware video compression and the ability to emulate a keyboard, mouse and disk drives. In order to provide this functionality, the BladeServer system management communication infrastructure needs more bandwidth.

Today's BMC microcontrollers communicate through a shared system NIC. The physical interface between the BMC and the shared system NIC is the SMBus (System Management Bus). The SMBus is very slow, with a typical clock rate of 400 KHz and a few chips supporting the maximum clock rate of 1MHz. The SMBus communication is simplex in nature since only one device can talk at a time. The overhead in the SMBus protocol inhibits the performance. This architecture was originally designed for simple alerting, but has recently been expanded to Serial Over Lan (SoL). With SoL, a remote administrator can start a serial console session through the BMC.

Several NIC vendors are offering high speed alternatives to the SMBus. The two alternatives that have become available are MII (Media Independent Interface) and FML. MII is an industry standard Ethernet interface with a maximum speed of 100Mb/s. FML is a proprietary interface developed and licensed by Intel Corporation with a maximum bandwidth of 8Mb/s.

This document assumes the use of a Broadcom NIC to do this particular cKVM implementation, but blade designers are free to use any vendor's NICs as appropriate.

Figure 2.6 shows a block diagram of the cKVM mezzanine card.

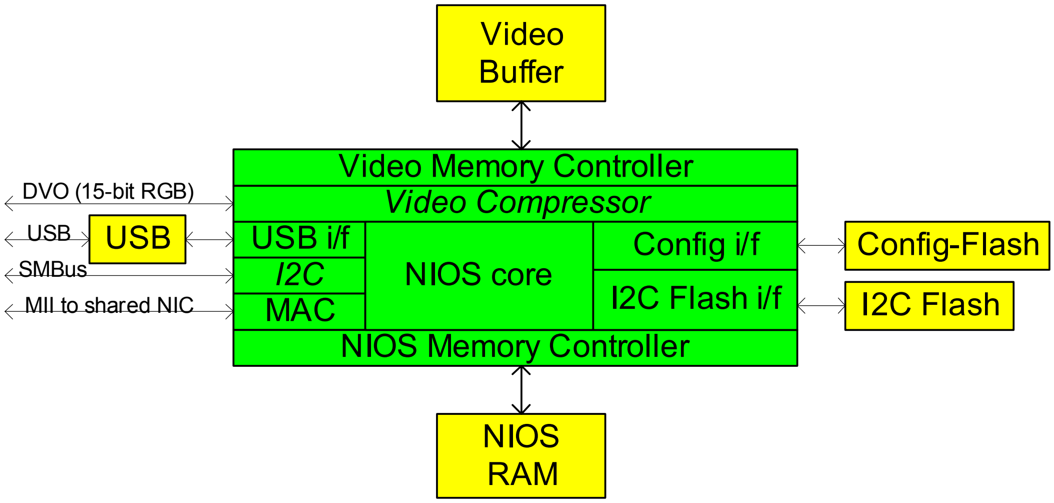


Figure 2-7 - cKVM Block Diagram

Table 2.9 lists the basic system functions on the mezzanine card.

Processor	32 bit NIOS softcore processor
Memory	SDRAM supports 128Mb FLASH 16Mb serial configuration EPCS16 (Stores logic and code)
I^{2}C	1 – I^{2}C slave connection to system 1 – Master for I^{2}C flash
MAC	10/100 Ethernet Media Access Controller core (MAC) with industry standard Media Independent Interface (MII)
USB Peripheral	USB composite device with three in and one out endpoint. The device will be able to emulate a keyboard interface, a mouse interface and a mass storage interface. The mass storage interface is capable of representing 16 mass storage devices by changing the value returned from the MAX_LUN command

Table 2-9 – **Basic System Functions**

2.7.1 Interfaces to the cKVM mezzanine card

cKVM has four communication busses (DVO, USB, MII and I^{2}C). These busses are shown in Figure 2.7.

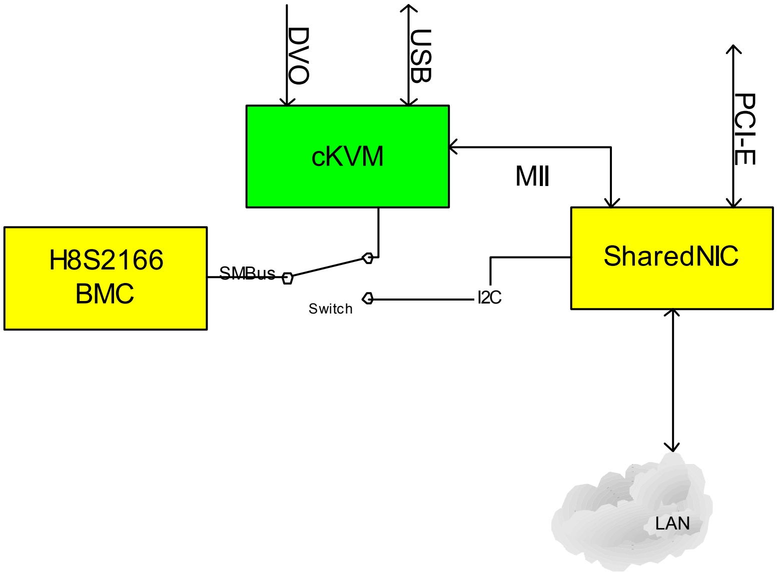


Figure 2-8 - cKVM interfaces

2.7.2 DVO (Digital Video Out)

The system video controller transmits video to the mezzanine card over the DVO bus. The DVO bus is a synchronous single direction 24-bit parallel bus. DVO uses raw RGB format with 8-bits per color [8-8-8]. The mezzanine card uses 15-bit color in [5-5-5] format.

2.7.3 USB

The mezzanine card contains a USB peripheral that connects to the server's USB host controller. The USB peripheral is configured as a composite device emulating Mass Storage and HID (Human Interface Device) devices.

2.7.4 MII

The acronym MII stands for Media Independent Interface. This interface is an industry standard that was developed to abstract the physical network from the Media Access Controller (MAC). The mezzanine card has an on board MAC that is connected to the shared NIC through an MII interface. The MII interface is connected as a MAC-to-MAC interface where the TX and RX signals are crossed. The shared NIC is an intelligent switch that provides the physical interface (PHY) to the LAN. The shared NIC accepts data that is addressed to either the host server or the mezzanine card and directs the data accordingly.

2.7.4.1 MII Fail Over

Each blade will contain two NICs. Both of the NICs will be on the same bus, identified by a two bit UMP address as defined in the "Universal Management Port Interface Command Specification For Broadcom Network Interface Controllers" document. The Broadcom document specifies a method for initializing and discovering each NIC. Only one MII interface is enabled for passing Ethernet traffic at a time.

2.7.4.2 MAC Address

The SMBus port on the Shared NIC has a dedicated MAC. The MAC address for the SMBus port is assigned by the BMC. The cKVM mezzanine card has its own MAC. The cKVM mezzanine card will be transparent to the BMC. During initialization cKVM must intercept the MAC address from the BMC.

2.7.5 I2C

I2C implements a master slave protocol where only a master can read and write other devices on the bus. In today's BMC architecture, the BMC is the master and the shared NIC is a slave. The BMC transmits data over the LAN by writing data to the NIC. When the shared NIC receives data for the BMC, the Shared NIC drives an interrupt signal into the BMC notifying that data is ready. The BMC can then read data from the NIC. The use of the interrupt is configurable. The I2C interface between the BMC and the mezzanine card will run at 400 kHz. When the mezzanine card is not plugged into the system, the interface to the NIC is SMBus instead of I2C.

2.7.5.1 I2C Fail Over

When the BMC fails over, it will address the secondary NIC on the I2C bus. When the mezzanine card is present, it will respond to the secondary NIC I2C slave address on the same bus. SMB_INTERRUPT1 is used by the mezzanine card when configured to alert the BMC of an available Ethernet packet at the secondary NIC I2C slave address.

2.7.6 cKVM Initialization

2.7.6.1 Shared NIC Initialization

The shared NIC requires different firmware in order to use the MII Port. The shared NIC vendor will provide an OEM configurable GPIO called MII_ACTIVE_N. This GPIO will be configured as an input with the following states.
High = I2C (default)
Low = MII Active
After POR (power on reset), the shared NIC will read the MII_ACTIVE_N and load the appropriate firmware. The MII_ACTIVE_N signal is driven by the mezzanine card CKVM_PRESENCE1_N pin and CKVM_PRESENCE2_N pin.

2.7.6.2 I2C Switch

The I2C switch will be activated by the CKVM_PRESENCEx_N pins. The BMC has the capability of switching the I2C bus when it disables the mezzanine card.

2.7.7 cKVM VPD

The MM requires the mezzanine card to emulate having at least 4KB of EEPROM storage for VPD. The BMC has access to this on aux power. The VPD data is accessed on the same I2C bus that is used for pass-through but using 0xA0 as the I2C address.

2.7.8 cKVM Connector options and Pin Descriptions

Table 2.10 shows the connector height options for the 140 pin connector.

Mated Height	Plug
5mm	Available upon request
6mm	IBM P/N 23k8778 / 41C4932 Pb Free

7mm	Available upon request
8mm	Available upon request
9mm	Available upon request
10mm	Available upon request

Table 2-10 – **Connector options**

Table 2.11 shows the pin descriptions for the 140 pin connector.

Pin	Net Name	Type	Description
1		GROUND	
2	DVO_RED6	INPUT	DVO RGB DATA
3		GROUND	
4	DVO_RED7	INPUT	DVO RGB DATA
5		GROUND	
6	DVO_RED3	INPUT	DVO RGB DATA
7		GROUND	
8	DVO_RED4	INPUT	DVO RGB DATA
9		GROUND	
10	DVO_RED5	INPUT	DVO RGB DATA
11		GROUND	
12	DVO_GREEN6	INPUT	DVO RGB DATA
13		GROUND	
14	DVO_GREEN7	INPUT	DVO RGB DATA
15		GROUND	
16	DVO_BLUE7	INPUT	DVO RGB DATA
17		GROUND	
18	DVO_HSYNC	INPUT	DVO Horizontal Sync
19		GROUND	
20	DVO_VSYNC	INPUT	DVO Vertical Sync
21		GROUND	
22	DVO_DE	INPUT	DVO Data Enable
23		GROUND	
24	DVO_CLK	INPUT	DVO Clock
25		GROUND	
26	DVO_GREEN3	INPUT	DVO RGB DATA
27		GROUND	
28	DVO_GREEN4	INPUT	DVO RGB DATA
29		GROUND	
30	DVO_GREEN5	INPUT	DVO RGB DATA
31		GROUND	
32	DVO_BLUE4	INPUT	DVO RGB DATA
33		GROUND	
34	DVO_BLUE3	INPUT	DVO RGB DATA
35		GROUND	
36	DVO_BLUE6	INPUT	DVO RGB DATA
37		GROUND	
38	DVO_BLUE5	INPUT	DVO RGB DATA
39		GROUND	
40	USB_P	I/O	USB D+
41	USB_N	I/O	USB D-

42		GROUND	
43	SMB_INTERRUPT0	OUTPUT	This is an open collector signal. The BMC can configure if this signal is used to indicate that the CKVM card / NIC has an Ethernet frame on the pass-through interface. Pull this up on the planar.
44		3.3VC	
45	I2C_SDA	I/O	I2C Data connected to the bmc bus for communicating with the NIC. Pull up resistor on the planar.
46	I2C_SCL	I/O	I2C Clock connected to the bmc bus for communicating with the NIC. Pull up resistor on the planar.
47		3.3VC	
48		3.3VC	
49		3.3VC	
50		3.3VC	
51	NIC_RESET_N	INPUT	So we can know when the NIC has been reset.
52		GROUND	
53	SMB_INTERRUPT1	OUTPUT	This is an open collector signal. The BMC can configure if this signal is used to indicate that the CKVM card / NIC has an Ethernet frame on the pass-through interface. Pull this up on the planar
54	RESERVED		
55	RESERVED		
56	RESERVED		
57	RESERVED		
58	RESERVED		
59	RESERVED		
60	RESERVED		
61	RESERVED		
62	RESERVED		
63	RESERVED		
64	RESERVED		
65	RESERVED		
66	RESERVED		
67	RESERVED		
68	RESERVED		
69	RESERVED		
70	CKVM_PRESENCE2_N		Connected on the card to CKVM_PRESENCE1_N.
71		3.3VC	
72	CKVM_PRESENCE1_N		Connected on the card to CKVM_PRESENCE2_N.
73	FPGA_DCLK	OUTPUT	
74		3.3VC	
75	FPGA_CONF_DONE	I/O	
76	FPGA_NCONFIG	INPUT	This signal is pulled up on the mezzanine card and should remain so when the system is turned off.
77		3.3VC	
78	FPGA_DATA0	INPUT	
79	FPGA_ASDO	I/O	
80		3.3VC	

81	FPGA_CE_N	INPUT	
82	FPGA_CS0_N	OUTPUT	
83		3.3VC	
84	MII_CD	OUTPUT	MII: Collision Detection
85	MII_CRS	OUTPUT	MII: Carrier Sense
86		3.3VC	
87	MII_MDC	I/O	
88	MII_MDIO	I/O	
89		3.3VC	
90	MII_RXDV	INPUT	MII: Receive data valid
91	MII_RXERR	INPUT	MII: Receive error
92		3.3VC	
93	MII_TXEN	OUTPUT	MII: Transmit data enable
94	MII_TXERR	OUTPUT	MII:
95		3.3VC	
96	EXT_PAUSE_IN	OUTPUT	
97		GROUND	
98	MII_RXD0	INPUT	
99		GROUND	
100	MII_RXD1	INPUT	
101		GROUND	
102	MII_RXD2	INPUT	
103		GROUND	
104	MII_RXD3	INPUT	
105		GROUND	
106	MII_RXCLK	INPUT	
107		GROUND	
108	MII_TXCLK	INPUT	
109		GROUND	
110	MII_TXD0	OUTPUT	
111		GROUND	
112	MII_TXD1	OUTPUT	
113		GROUND	
114	MII_TXD2	OUTPUT	
115		GROUND	
116	MII_TXD3	OUTPUT	
117		GROUND	
118	MDINT_N	OUTPUT	
119		3.3VC	
120		3.3VC	
121		3.3VC	
122		3.3VC	
123	RESERVED		
124		GROUND	
125	CKVM_FAULT_N	OUTPUT	This signal is normally held high by the FPGA. When there is a fault a pull down resistor on the planar will pull the signal low.
126	RESERVED		
127	RESERVED		

128	RESERVED		
129	RESERVED		
130	RESERVED		
131	RESERVED		
132	RESERVED		
133	RESERVED		
134	NIOS_RESET_N	I/O	Pull this signal low to hold mezzanine card in reset. A Pull up resistor is on the mezzanine card. This signal is also used to drive the I2C switch logic on the planar. To prevent power on resets from affecting the switching logic an open drain buffer should be placed between the switching logic and the mezzanine card.
135	RESERVED		
136	PWR_GOOD	INPUT	The mezzanine card uses this signal to determine when the blade is powered on
137	RESERVED		
138	RESERVED		
139		GROUND	
140	RESERVED		

Table 2-11 – **Pin Descriptions for cKVM connector**

2.7.9 cKVM Power Requirements

Table 2.12 shows the power supply requirements.

Voltage	Regulation	Min.	Max.	Max. Peak to Peak Ripple & Noise
3.3VC	+/- 5%	3.135V 0.0A	3.465V 1.5A	33mVp-p

Table 2-12 – **Power Supply Requirement**

2.8 Network Interface Topology

The BladeServer chassis supports four high speed communication ports per Processor Blade, as shown in Figure 2.8. The Mid-plane is wired to route this interface to four Switch Module slots. These high speed links are intended use for Ethernet, Fibre Channel and in future Infiniband network connections based on Processor Blade and its corresponding slot configuration. Each Switch module connects to one high speed communication port from each Processor Blade.

The following partial listing illustrates the routing:

Processor blade 1 PORT 1 -> Switch Module 1 input 1
Processor blade 1 PORT 2 -> Switch Module 2 input 1
Processor blade 1 PORT 3 -> Switch Module 3 input 1
Processor blade 1 PORT 4 -> Switch Module 4 input 1

Processor blade 2 PORT 1 -> Switch Module 1 input 2
Processor blade 2 PORT 2 -> Switch Module 2 input 2
Processor blade 2 PORT 3 -> Switch Module 3 input 2
Processor blade 2 PORT 4 -> Switch Module 4 input 2

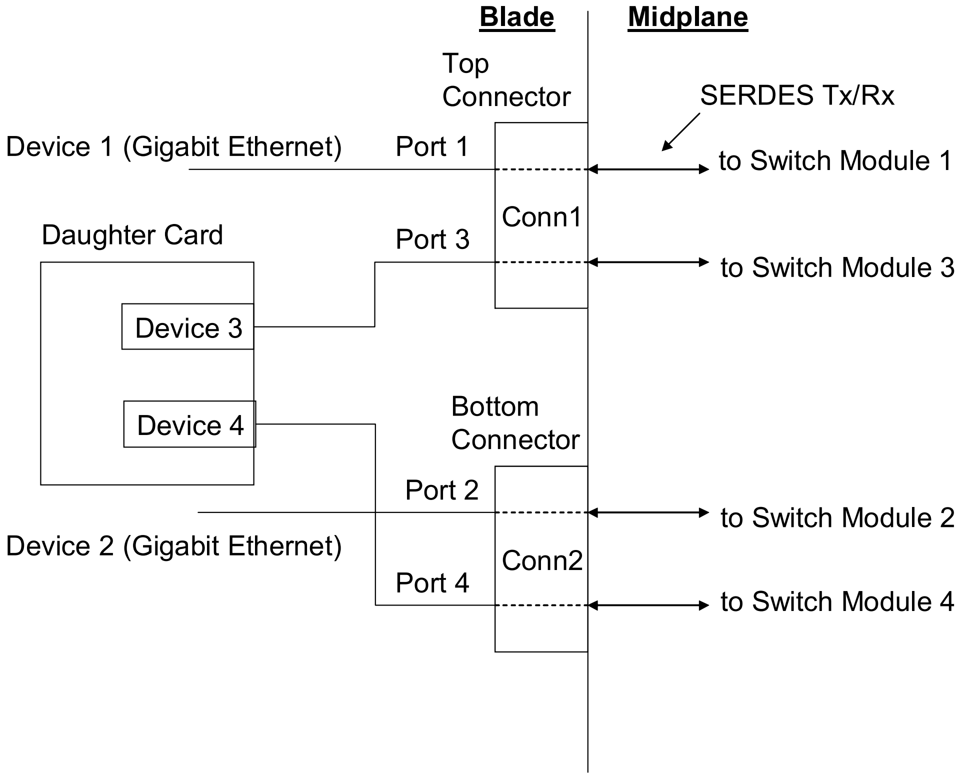


Figure 2-9 - Network routing

2.8.1 Ethernet Network Interface

The BladeServer Chassis is designed to accommodate several combinations of server blades, I/O expansion cards, and switch modules. The Ethernet NIC interface on the server blades must be flexible with regard to connectivity options. The internal blade NIC interface for compatible products has the following requirements:

1. The dual NIC interfaces to switch module slots 1 and 2 must support a 1 Gbps SERDES electrical interface as defined in the "BladeServer Base Specification and Design Guide for SERDES High-Speed Electrical Signaling."
2. The NIC interfaces must be compatible with internal switch interfaces that operate in Link Auto-negotiation mode as defined in Clause 37 of the IEEE 802.3 Standard. Clause 37 defines the protocol to select duplex mode, link speed and pause support. The blade NIC interface speed is fixed at 1 Gbps and full-duplex mode.
3. The NIC interfaces must also be compatible with internal switch interfaces that do not support Link Auto-negotiation. Such interfaces can be assumed to support a fixed-mode, full-duplex, 1 Gbps SERDES (per Clause 39).
4. The NIC interface is expected to establish connectivity if either the switch or the blade is initially installed and active. The MM controls power to the blade or switch module once it is properly inserted and recognized as valid.

The NIC interface may accomplish Steps 2 and 3 in either sequence, such as:

1. The NIC may attempt to establish link in Auto mode and will follow the protocol outlined in Clause 37 to establish a 1-Gbps, full-duplex link when a compatible interface is detected. Incompatible interfaces (e.g., 100 Mbps only) are rejected.
2. The NIC may detect the presence of an active switch link via the K28.5 link pulse activity on the receive link while in Auto mode (see Clause 39, section 39.6). If so, the NIC should respond by forcing connection in fixed-mode, full-duplex at 1Gbps.
3. Optionally, the NIC may attempt to establish link in fixed-mode initially. Failure to connect in fixed mode after a fixed time period will result in automatically toggling to Auto-negotiation mode.
4. Failure to connect after a fixed time period will propagate "no connection status" to higher level software. It is expected that a switch module may be powered on at a later time, therefore it is required that connectivity must be periodically retried.

2.8.2 I/O Expansion Card Interface

A Processor Blade can optionally support a custom expansion slot for additional functions. This slot is designed to be used to add additional interfaces such as LAN, Fibre channel or InfiniBand. Refer to the "BladeServer Base Specification For I/O Expansion Cards" for information on the placement, connectors and power requirements. If the design is for the I/O Expansion Card (EC), Small Form Factor I/O Expansion Card (SFF EC), or the Combo Form Factor V I/O Expansion Card (CFFv), the network signals will be routed through the Processor Blade, and the "BladeServer Base Specification and Design Guide for SERDES High-Speed Electrical Signaling" should be consulted.

2.9 BMC (BaseBoard Management Controller) (Required)

Each Processor Blade is required to have a local BMC that provides basic environmental monitoring capabilities. It typically interfaces with the Processor Blade CPU via an I^2C or LPC bus, which allows In-band monitoring capabilities of a Processor Blade within a BladeServer chassis. The BMC also provides out-of-band management capabilities of Processor Blades via the Management Module in a BladeServer chassis and is the control point for each blade.

The BMC interfaces with the Processor Blade local environmental sensor hardware via a local I^2C bus. Device drivers provide the interface between local OS monitoring applications and the BMC. The BMC must be designed to monitor voltages, temperatures, control the fault LEDs and allow selection of the shared media. The BMC also controls the following functions on a given Process Blade: local power On/Off, selection of shared media devices like CD-ROM and FDD, and switching on/off the USB and Video buses to the Mid-plane. In addition, the BMC is also required to maintain the current status of these buses. The Management Module additionally controls several policy settings at the BMC. This includes items like local power control, wake-on-lan settings and others. Host applications must never change or override settings previously set by the Management Module. The BMC must be responsive to the MM. All properly formed commands issued to the BMC from the MM must be executed immediately.

The BMC provides an IPMI KCS (*Keyboard Controller Style*) host interface[2] to support BIOS and System Management Software (SMS). The KCS interface is fully described in the IPMI specification.

The BMC must also provide an RS485 interface to allow it to be connected to chassis Management Modules via the Mid-Plane. This path provides the only out-of-band management of the server blade within the BladeServer chassis.

The BMC will provide the following:

- Watchdog Timer(s):
    - BMC contains watchdog timer support that can detect:
        - BIOS/POST hangs
        - OS Loader hangs
        - OS Hang
- VPD support (for more information see the 'Base Specification for VPD')
- PFA for VRM/CPU/Memory/HD
- SEL based Error log
- RS485 interconnect support – see Section 2.4
- Vendor specific Environmental querying/monitoring and Alerts (ex. TEMP/Voltages)
- Remote Power On/Off
- Local LED control – see Section 5.13
- Host Comm port serial redirection supporting Serial Over LAN – see Section 5.16
- I/O Expansion Card support (see BladeServer Base Specification For I/O Expansion Cards)

[2] The IPMI specification describes three possible choices for what the implementation may use for the host system interface. The assumption in this section and in section 5.18 that details the BMC Communication interface to BIOS is that a KCS system interface is used. However the specific system interface choice used to transfer IPMI messages is implementation specific.

The BMC will connect to two RS485 interfaces, one on the upper mid-plane connector and one on the lower mid-plane connector. This allows the BMC to communicate with the active Management Module. The onboard BMC will coordinate with the Management Module to accomplish the following:

- Remote power/power down of the Processor Blade
- Error logging
- Power and Thermal Management
- Remote systems management
- Selection/deselection of correct Processor Blade KVM and USB for CDROM and FDD
- Serial Redirection
- Local Policy Control
- Environmental Control
- BMC firmware updates and BMC resets

2.10 Reference BIOS (Basic Input/Output System) Example

Processor Blades that make use of a BMC typically need an interface from the BMC to a host processor on the blade. This interface is used to communicate control and status between the BMC and the HW placed in power domain 2. The following section is described using the term BIOS for convenience but other implementations of firmware execution on the host processor is also allowed.

In BladeServer, each Processor Blade requires firmware called BIOS (Basic Input Output System). The BIOS of the processor blade is contained in a non-volatile device such as flash memory. When a processor blade is powered on (PD2) or reset the processor will begin execution of the BIOS firmware. The BIOS initially executes a Power-On Self Test (POST) of the processor blade and validates that all the hardware components are functioning properly. Once POST has validated the hardware, the BIOS will attempt to boot and load an Operating System (OS). In this document the term OS will be used to indicate software which controls the HW located in PD2. The OS may be resident on a local hard-drive of the processor blade, on a device attached to the media tray (for example CD or floppy drive) or possibly on a network attached device. Once an OS loads, the BIOS may remain resident in memory to assist in some of the functions provided for systems management as well as providing some low level functions to control various I/O devices on the processor blade.

Optionally, two additional utilities are provided by the processor blade BIOS:

1. A user setup utility that can be invoked during POST. This setup utility might allow for a system administrator to view and configure certain features of the BIOS on the processor blade. The following list represents possible functions provided by a setup utility:

    - Specific information about the hardware configuration and firmware components on the ServerBlade
    - The ability to enable and disable certain functions or behaviors within the hardware configuration of the ServerBlade
    - The ability to set the RTC (Real Time Clock) to a specific date and time
    - Security settings like power on passwords
    - Ability to view/change the method and sequence of the devices used to boot the OS (Wake On LAN, hard drive(s), floppy drive, network, CDROM)
    - Information and functions provided by the BMC on the processor blade (Watchdogs, BMC configuration and SEL (System Event Log) viewer)
    - Ability to set up the serial ports for control of the *Serial Over LAN* function. In addition the ability to view the SOL LAN configuration.

2. A BIOS firmware flash update utility to allow for upgrades of the BIOS and any ROM based Diagnostic firmware

For additional information on BIOS interactions with the BMC see section 5.18.

2.11 Power Input Circuit (Required)

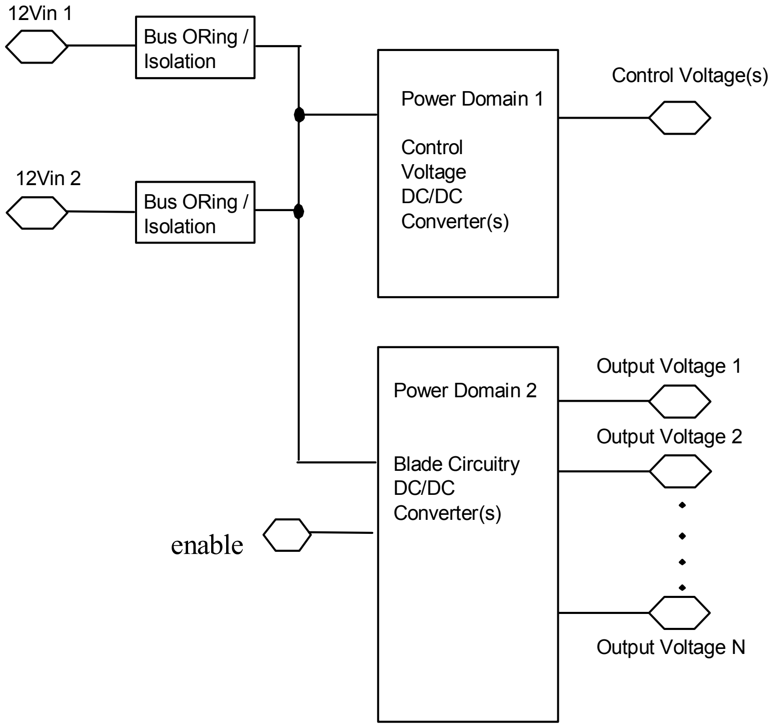
2.11.1 Functional Overview

This circuit (or its equivalent), present on all cards, accepts power from two redundant +12V inputs from the system Mid-plane and is designed to provide the following functions:

- Hot plugging of the card into an energized backplane.
- Auxiliary (continuous) voltage to onboard service processor or card control circuitry.
- ORing of the two +12V input voltages (allows continued power to the card if one of the inputs is removed due to a fault).
- The circuit must have a fast response to reverse currents which indicate a shorted +12V input source in order to minimize the effect on the redundant voltage source.
- Main power to the card, which is controlled by an Enable signal.
- Soft start to minimize impact on inrush when card is turned on.
- Current sensing and limiting. This circuit must react quickly to an overload on the card to minimize adverse affects to the rest of the system. The circuit latches card main power off when detected.
- The maximum average power draw allowed is 250 Watts per blade slot in BladeServerE and 350 Watts per blade slot in BladeServerH*. This maximum average power draw can be exceeded with the following restriction:
    - A maximum average power draw per slot of up to 308W is allowed in BladeServerE and 430W in BladeServerH as long as power management functions are implemented to reduce power to at least the maximum average stated above and as architected in this specification. The power reduction value will be contained in the OEM IPMI command “Set Pre-Set Watts” upon loss of +12V input power redundancy. This feature requires independent monitoring of each of the +12V inputs as specified in section 5.7.5 Power Throttle & Cooling Management. All blades must support the power management interface to the MM as specified in Section 5.7.

*Note 1: Blades that span multiple slots, connect to multiple slots on the backplane and exceed the single slot power limit must draw power from each occupied slot. The power load and 12V distribution must be segmented to keep each individual slot’s power connections isolated from each other. Also, Blades designed for the higher BladeServerH power limits may not be allowed to power on in a BladeServerE chassis.

*Note 2: The midplane power distribution is divided into two power domains such that multi-slot blades can span this power boundary. Blade designs that span more than one blade slot should take this into account if power is needed from multiple slots. The multi-slot blade should detect if both midplane domains are powered before applying main power to the blade.



* Voltage and number of DC Outputs required depends on Blade implementation

Figure 2-10 - Blade Power Input Block Diagram

2.11.2 Control and Fault Signals

- Enable_N - Open collector input, active LOW, controls main power to the card. O/C Fault - Output, active LOW, indicates an over-current condition occurred on main 12V to the card.
- O/C_FAULT_N -standard TTL output, active LOW, indicates an overload has occurred. Threshold depends on the host blade or module (set to 20A+/-5% in schematic for PB, below).
- O/C_Reset_N - standard TTL 5V or 3.3V input, momentary (1us to 5ms) active LOW, resets an over-current latch-off, allows reapplication of main 12V input if fault is removed.
- PWR_CRKT_FAULT_N - standard TTL output, active LOW, indicates possible input circuit control failure. Once the card is removed, it should not be re-inserted.
- VRM_OVP – standard TTL 5V or 3.3V input active HIGH indicating an over-voltage from the CPU dc to dc converter was detected. This signal will remove main power to the card.

2.11.3 Blade Hot-Plug Capabilities

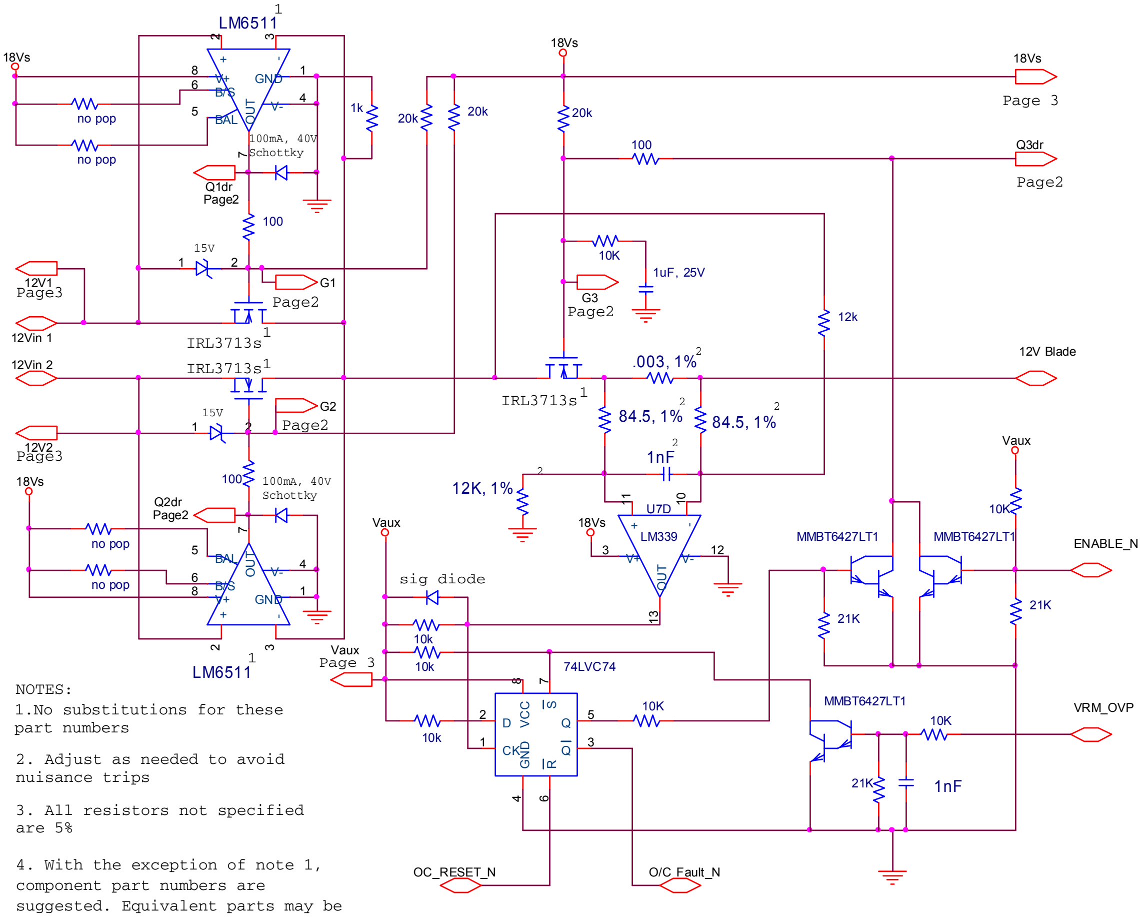
This circuit is designed so the host card can be plugged into a backplane energized with 12V. Auxiliary power will automatically be provided through a soft-start circuit thus enabling the I2C and any service processor circuitry in power domain 1, but main power to the card will not be applied until an Enable signal is received. The preferred maintenance procedure shall be as follows:

- Card Removal
  - The onboard SP will issue an ENABLE signal HIGH to remove main input power to the card.
  - Observe that the POWER APPLIED? LED is not lit.
  - Unlatch the card using the card handle.
  - Remove the card from system housing.
- Card Insertion
  - Plug in the new card into the system housing.
  - Latch the card handle.
  - Allow the onboard SP or management module to control power on.

In the event that no management module is installed, or there is a management module fault, the card can be turned on and off using the power-on push button switch.

The reference circuits for BladeServer shown in Figures 2.10a ,b, and c shows power domain 1 circuitry (standby power and protections) on POWER INPUT CIRCUITRY PAGE 3 and power domain 2 circuitry (discrete implementation of the MOSFET OR'ing of the two input sources, domain 2 enable and input protection circuitry) on POWER INPUT CIRCUITRY PAGE 1 and PAGE 2. Power input circuitry for BladeServerH applications must take into account the higher power loading requirements.

POWER INPUT CIRCUITRY PAGE 1



NOTES:
1.No substitutions for these part numbers

2. Adjust as needed to avoid nuisance trips

3. All resistors not specified are 5%

4. With the exception of note 1, component part numbers are suggested. Equivalent parts may be used.

Figure 2.11a Input ORing and Main Power Control

An acceptable alternate OR’ing MOSFET control implementation can be obtained using two Maxim MAX8585EUA controllers.

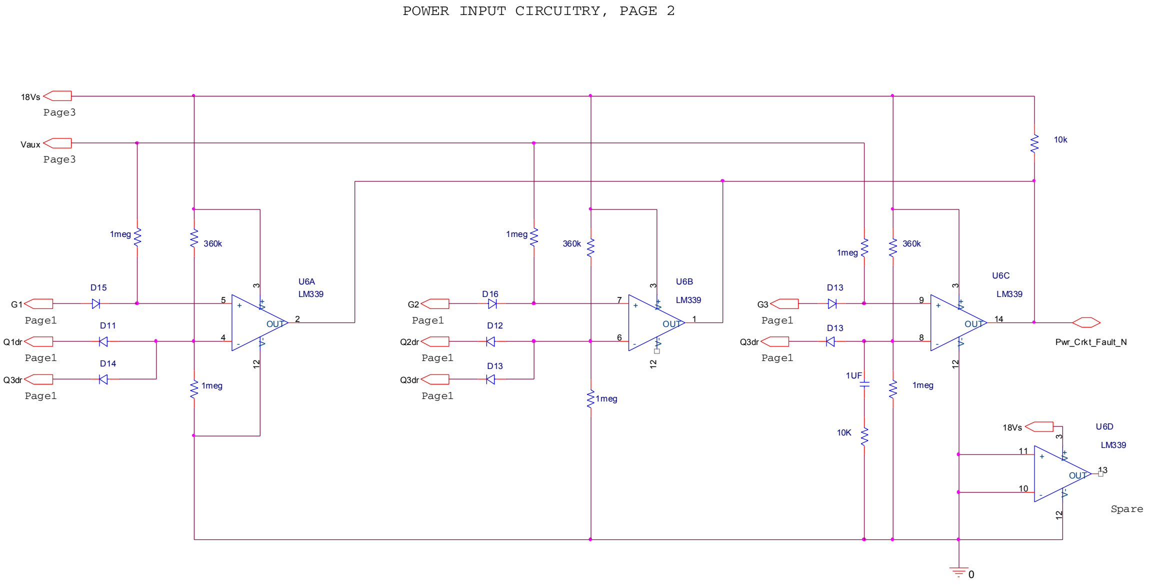


Figure 2.11b Input OR’ing diagnostics and fault control

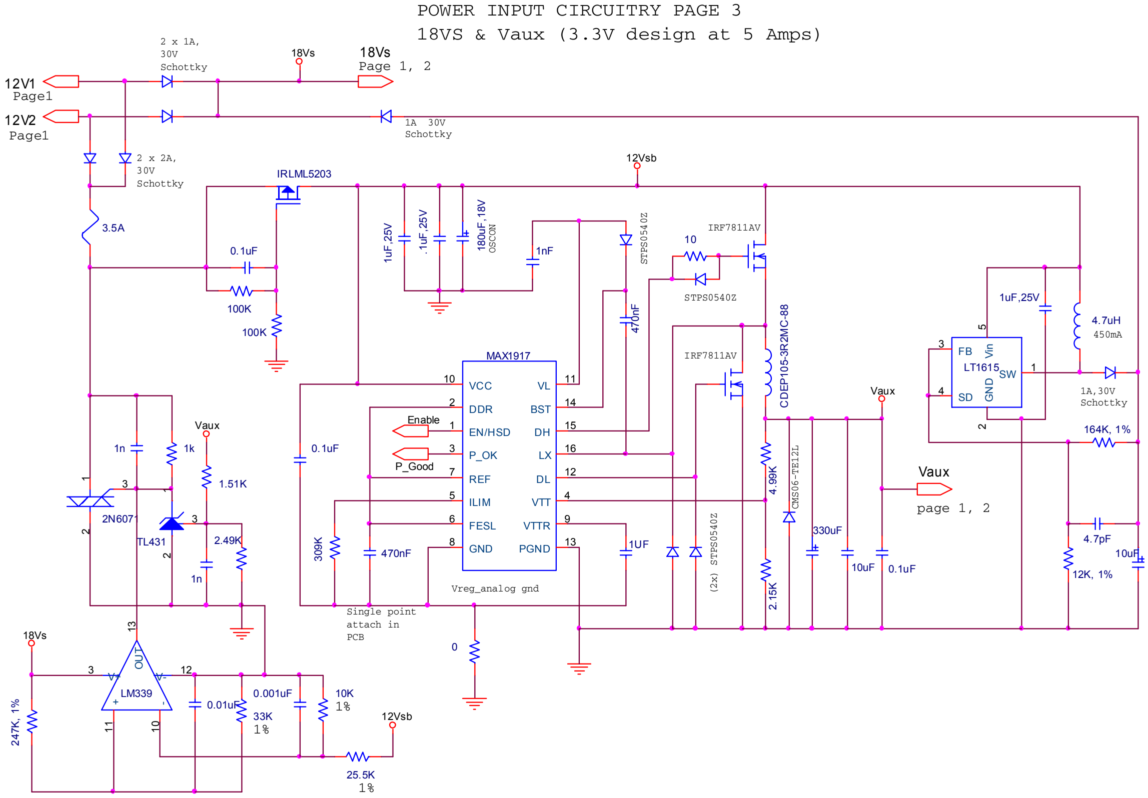


Figure 2.11c Power Domain 1: Standby power

3 Processor Blade Cooling

This section describes the thermal design requirements that will enable a Processor Blade to be properly cooled within a BladeServer Chassis.

As described above, there is more than one type of BladeServer chassis that will support Processor Blades depending on the customer environment, such as Enterprise or Telecommunications (Telco). Each BladeServer chassis is based on 29 mm single wide Processor Blades. However, there are two types of airflow paths based on the chassis type. The Enterprise 7U chassis and the Telco 8U chassis have similar airflow paths. The Enterprise 9U chassis and the Telco 12U chassis with high speed switch function provide a slightly different airflow path for the blades. A blade designed to be cooled in an Enterprise 7U or Telco 8U will be sufficiently cooled when installed in an Enterprise 9U or Telco 12U chassis. The opposite may not be true and is primarily dependent on the blade wattage.

3.1 Enterprise 7U & Telco 8U BladeServer Chassis

The air flow paths through a Processor Blade intended to be installed in an Enterprise 7U and a Telco 8U BladeServer chassis are shown in Figure 3.1. All air enters the blade through the front of the blade as shown by air flow path “B”. Place components on the blade board such that the air impedance across the board provides the required air flow (CFM) through the “XYZ” paths of the blade enclosure, where at least 20% of the CFM exits via path X, at least 20% of the CFM exits via path Z and the remainder exits via path Y. The dimensions of the blade enclosure air flow path openings are defined in the Processor Blade Enclosure Mechanical Drawing section below.

Tables 3.1 and 3.2 define the major thermal operating conditions of the system. The air flow rates listed are values for a specific blade design in the chassis. Air flow through other blade designs may vary slightly depending on the actual blade air flow impedance. Under all operating conditions for the Enterprise 7U system expressed in Table 3.1, the blade exit air temperature must not exceed 52 degrees C at any exhaust point of the blade.

Enterprise 7U

Blower Condition	**Low Speed**	**High Speed**	**High Speed**	**Blower Failure**
Ambient	25 °C	32 °C	35 °C	35 °C
Elevation	7000 ft	7000 ft	3000 ft	3000 ft
Airflow	13.7 CFM	28.9 CFM	28.9 CFM	15.2 CFM

Table 3.1 Enterprise Environment and Air Flow Specifications

Telco 8U

Blower Condition	**Low Speed**	**High Speed**	**High Speed - Blower Failure**	**High Speed – Air Conditioner Failure**
Ambient	25 °C	40 °C	40 °C	55 °C *
Elevation	5905 ft	5905 ft	5905 ft	5905 ft
Airflow	17 CFM	38 CFM	28.5 CFM	38 CFM

*55 °C spec. when equipment is tested outside of a rack full of equipment. 50 °C is the spec. when in the rack.

Table 3.2 Telco Environment and Air Flow Specifications

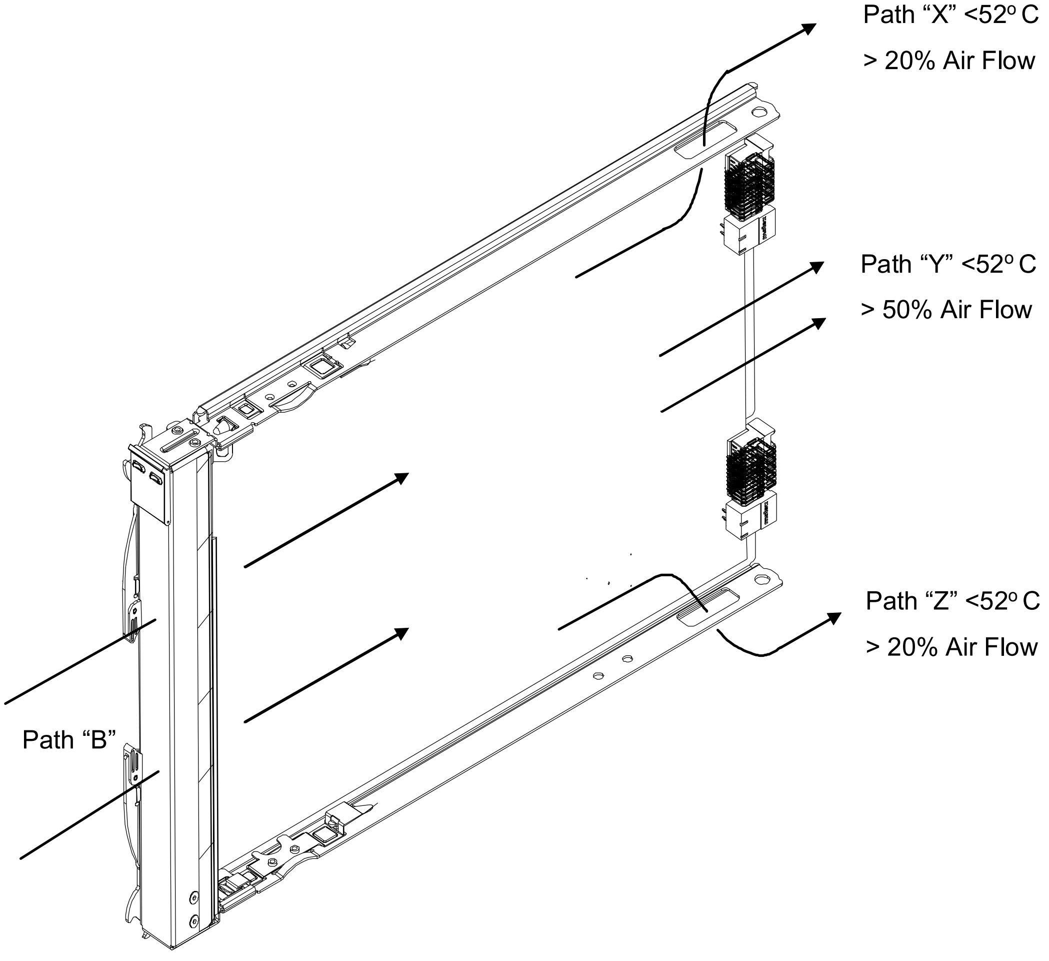


Figure 3.1 Air flow Path – Processor Blade for an Enterprise 7U & Telco 8U BladeServer Chassis

3.2 Enterprise 9U & Telco 12U BladeServer Chassis

The air flow path through a blade when installed in an Enterprise 9U and a Telco 12U BladeServer chassis is different than when installed in an Enterprise 7U & Telco 8U chassis. In this case, fresh airflow (i.e. ambient) may be brought into the blade at three locations, as shown in Figure 3.2. The dimensions of the blade enclosure air flow path openings are defined in the Processor Blade Enclosure Mechanical Drawing section below.

Major thermal operating conditions of the Enterprise 9U system are defined in Table 3.3 and the major thermal operating conditions of the Telco 12U system are defined in Table 3.4. The air flow rates listed for the three inlet regions are values for a specific blade design in the chassis. Air flow through other blade designs may vary slightly depending on the actual blade air flow impedance.

Note that for blades intended for the Enterprise 9U and the Telco 12U chassis, air inlet into the blade through path A and path C is optional. It is acceptable to exclude the chassis openings for paths A and C and to bring all of the air into the blade through path B.

Enterprise 9U

Blower Condition	**Low Speed**	**High Speed**	**High Speed**	**Blower Failure**
Ambient	25 °C	32 °C	35 °C	35 °C
Elevation	7000 ft	7000 ft	3000 ft	3000 ft
Airflow Inlet “B”	19.0 CFM	33.0 CFM	33.0 CFM	23.5 CFM
Airflow Sides Both “A & C”	2.5 CFM	3.5 CFM	3.5 CFM	3.0 CFM
Airflow Total	24.0 CFM	40.0 CFM	40.0 CFM	29.5 CFM

Table 3.3 Enterprise 9U Environment and Air Flow Specifications

Telco 12U

Blower Condition	**Low Speed**	**High Speed**	**High Speed - Blower Failure**	**High Speed – Air Conditioner Failure**
Ambient	25 °C	40 °C	40 °C	55 °C *
Elevation	5905 ft	5905 ft	5905 ft	5905 ft
Airflow Inlet “B”	19 CFM	40 CFM	35 CFM	40 CFM
Airflow Sides Both “A & C”	2.5 CFM	5 CFM	4.5 CFM	5 CFM
Airflow Total	24 CFM	50 CFM	44 CFM	50 CFM

*55 °C spec. when equipment is tested outside of a rack full of equipment. 50 °C is the spec. when in the rack.

Table 3.4 Telco 12U Environment and Air Flow Specifications

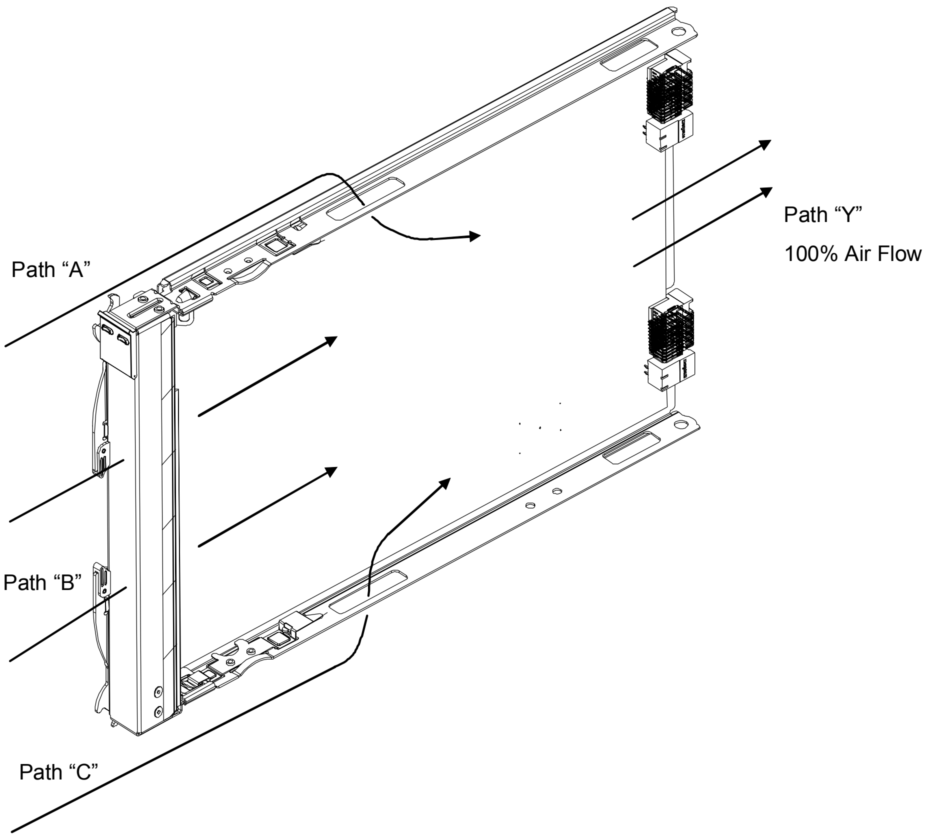


Figure 3.2 Air flow Path – Processor Blade for an Enterprise 9U & Telco 12U BladeServer Chassis

Path "Y" is restricted by an opening in the system midplane. The minimum opening is 34.25mm tall and its bottom edge is located 113.6mm from the bottom edge of the Blade chassis as oriented in Figure 3.2

Component layout should take this restriction into account to allow minimally restricted airflow in this area.

3.3 Processor Blade Air Flow Impedance

In addition to each of the above stated airflow rates for the various operating modes, it is required that the total pressure drop across the defined blade enclosure, with the blade board and electronic components, be maintained between the limits shown by Figure 3.3 and defined by Equations 3.1 and 3.2. For example, at 18 CFM, the blade impedance should be between 0.191 and 0.211 inches H_2O. The impedance curves apply to all blades, including single and double wide blades (or wider).

Equation 3.1: $\Delta P = 0.00196 * CFM^{1.62}$ (Upper Limit)

Equation 3.2: $\Delta P = 0.00196 * CFM^{1.584}$ (Lower Limit)

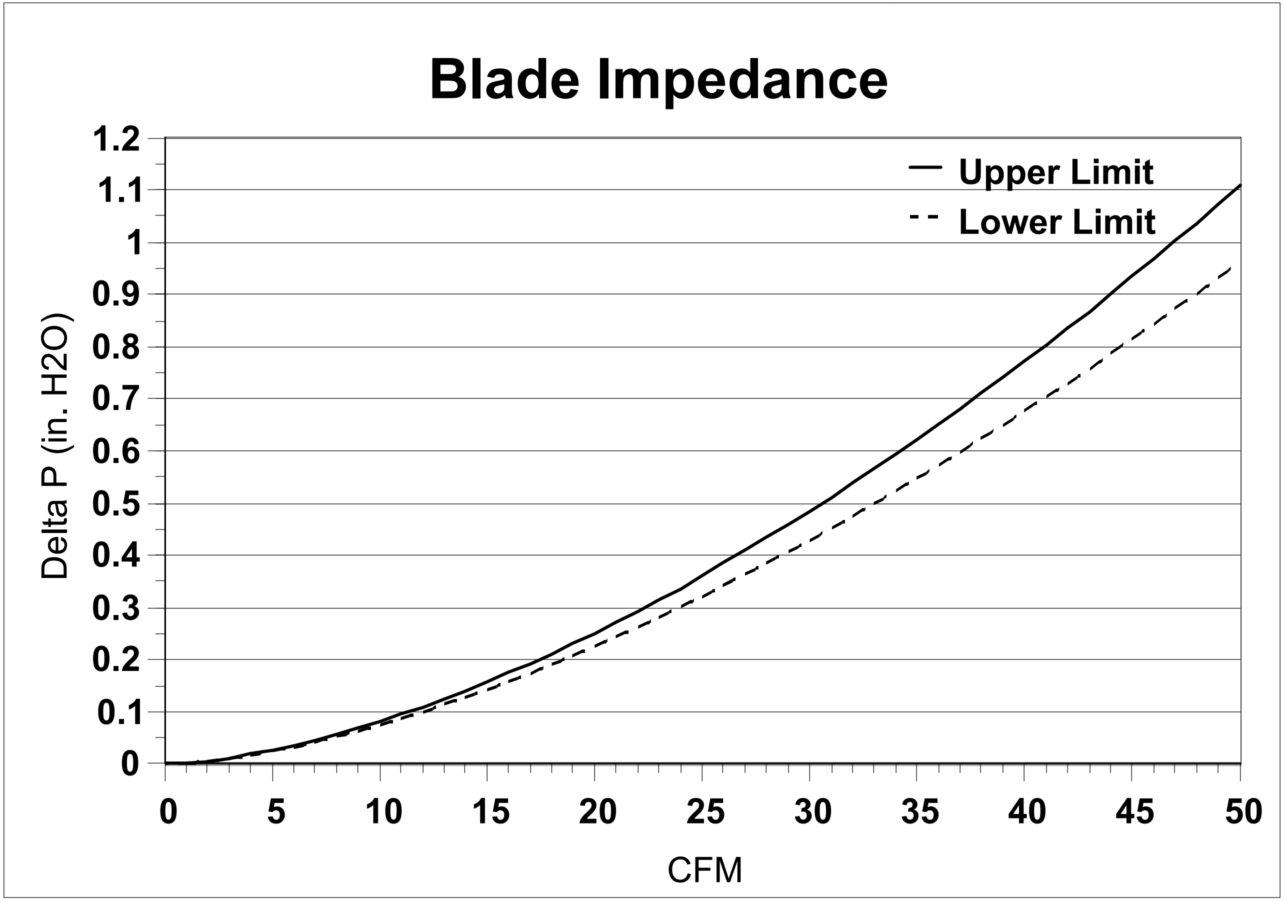


Figure 3.3 – Processor Blade Impedance

A gasket is used between individual Blades, primarily for the purpose of preventing EMC problems, but it also provides an air seal between blades. If an alternative type of EMC shielding is used any air leakage past the shield must be taken into account as part of the overall Blade impedance. The gasket material specified in Figure 4.9 should be used in order to minimize air by-pass.

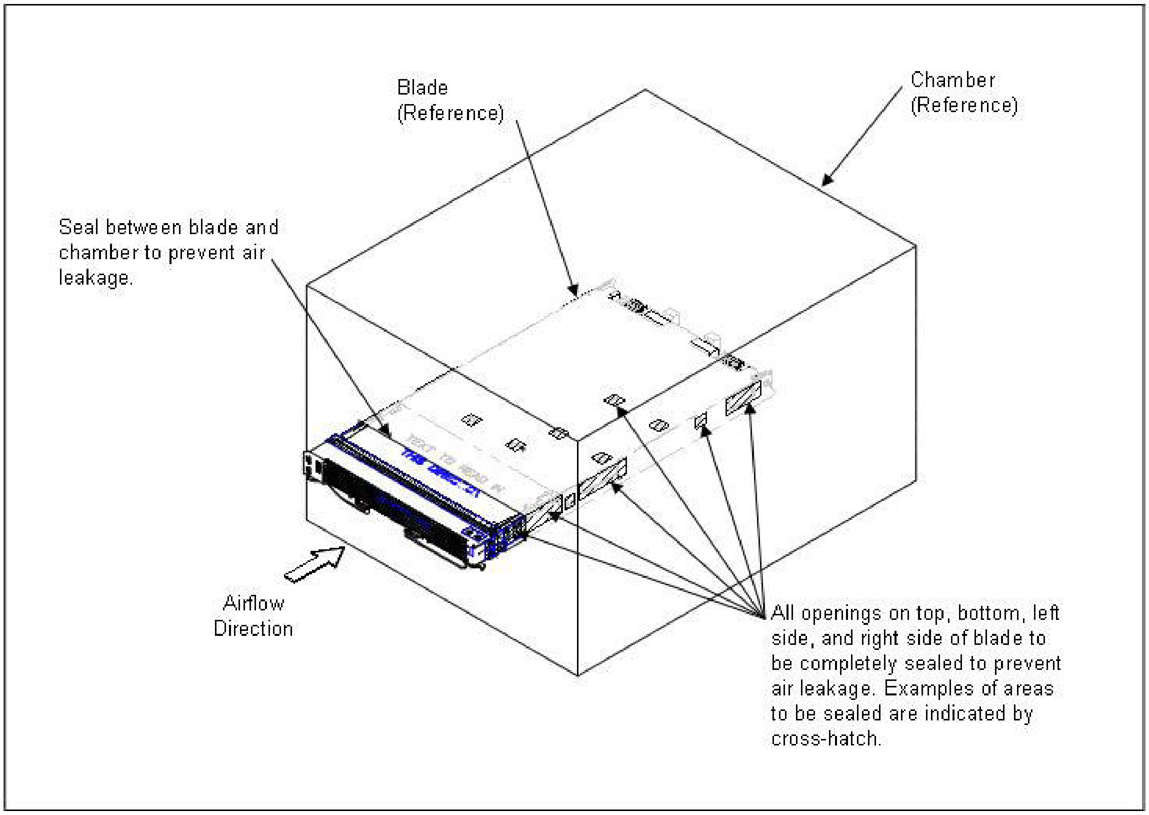

Blade
(Reference)
Chamber
(Reference)
Seal between blade and chamber to prevent air leakage.
Airflow
Direction
All openings on top, bottom, left side, and right side of blade to be completely sealed to prevent air leakage. Examples of areas to be sealed are indicated by cross-hatch.

Figure 3.4 – Processor Blade Impedance Test Setup Requirement

Figure 3.4 is representative of the test setup required to measure the impedance of the blade. The airflow protocol through the blade for this test will be uniquely different from what you will have in the actual application within the chassis, as shown in figures 3.1 & 3.2.

4 Processor Blade Mechanical Drawings

4.1 Processor Blade Mechanical Design Kit

The Processor Blade Mechanical Design Kit is available at the following URL:

http://www-03.ibm.com/systems/bladecenter/open_specs.html

This kit will provide data necessary to design the mechanical enclosure and printed circuit card for the Processor Blade. It is recommended that the construction technique described by the design data be followed; however, it may be modified per the application. The mechanical design kit consists of a “Readme” file, a set of 3D CAD models, and PDF files which provide a base for designing a blade enclosure. The 3D CAD models are in Pro/ENGINEER design software, IGES, STEP, and an EMN file format. The EMN file of the printed circuit card will define the card outline, connector location, component keep-out areas and component height restrictions. The 2D drawings are provided in Adobe® PDF file format. A bill of material is included in the design kit “Readme” file. The mechanical design information in the design kits shall take precedence over the following information. Information describing 240 VA keep-out areas is stated exclusively in this document, not in the Mechanical Design Kits.

4.2 Processor Blade Board

All dimensions shown are critical to function. The mechanical design must meet all required National and International applicable safety and marketing requirements such as, but not limited to, 240VA limiting and EMC and ESD Standards.

The following notes pertain to the drawings below.

1) All dimensions shown are mm over inch.

2) Linear tolerance of +/- 0.076mm (0.003") for routed and hole dimensions. All others to be +/- 0.25mm (0.010").

3) No components shall protrude past the edge of the card, other than the Blade signal connectors and Blade power connectors as defined in Table 2.1 and Table 2.2. Violation of this may result in incompatibility with a chassis.

4) Drawings not to scale.

Figure 4.1 defines the primary blade board connectors and the mezzanine card connector. Note that the mezzanine connectors don't mount to the blade board, but rather on a secondary card that is intended to mount parallel to and 18 mm from the blade board.

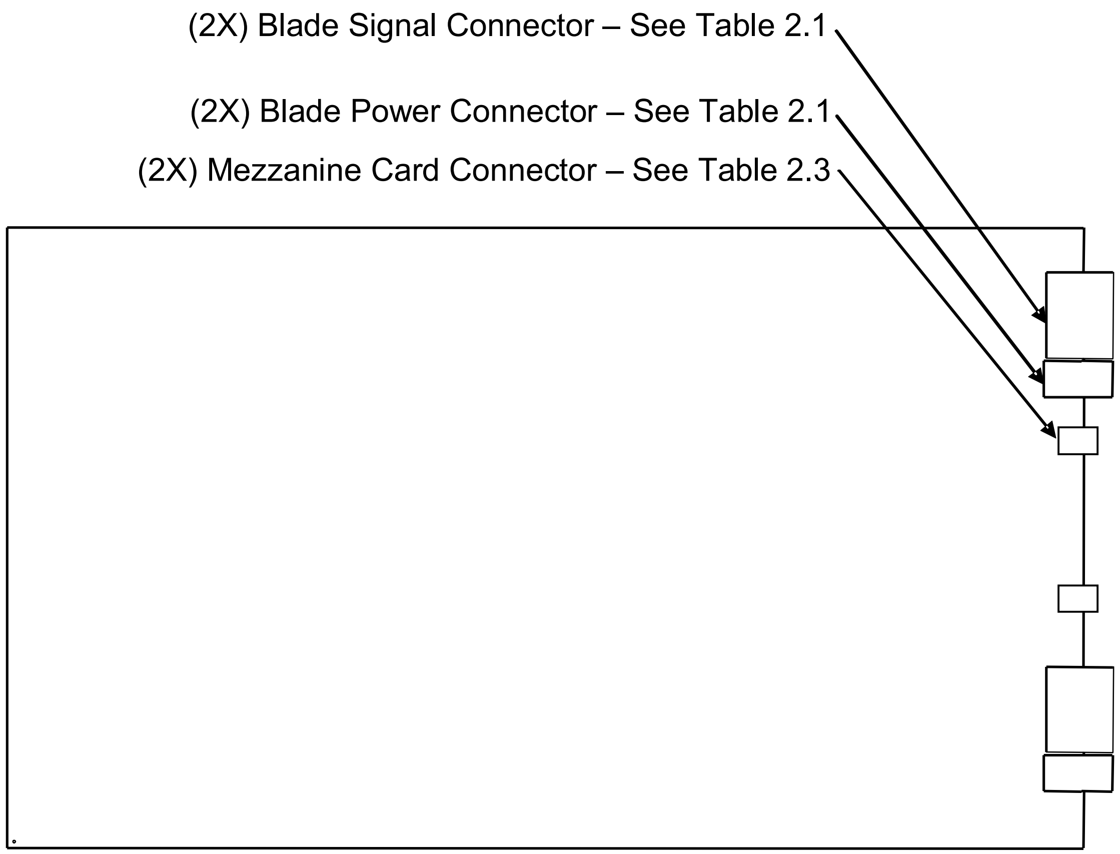

(2X) Blade Signal Connector – See Table 2.1
(2X) Blade Power Connector – See Table 2.1
(2X) Mezzanine Card Connector – See Table 2.3

Figure 4.1 – Processor Blade Board Primary Connector and Mezzanine Card Connector Definition

The board size (227 x 394.2 mm) shown in Figure 4.2 may vary per the Processor Blade requirements. The primary connector locations and the 1.7 mm dimension from the edge of the board to Datum A must not vary.

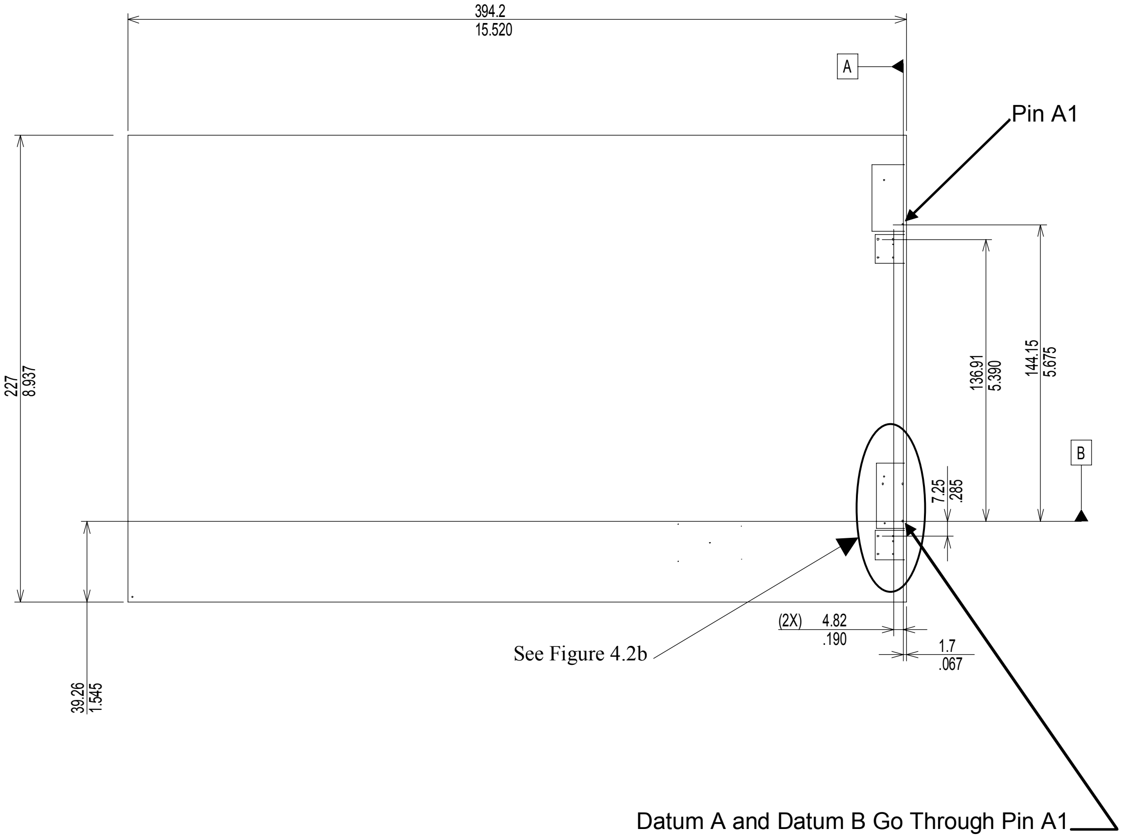


Figure 4.2a – Processor Blade Board Size and Primary Connector Location Definition

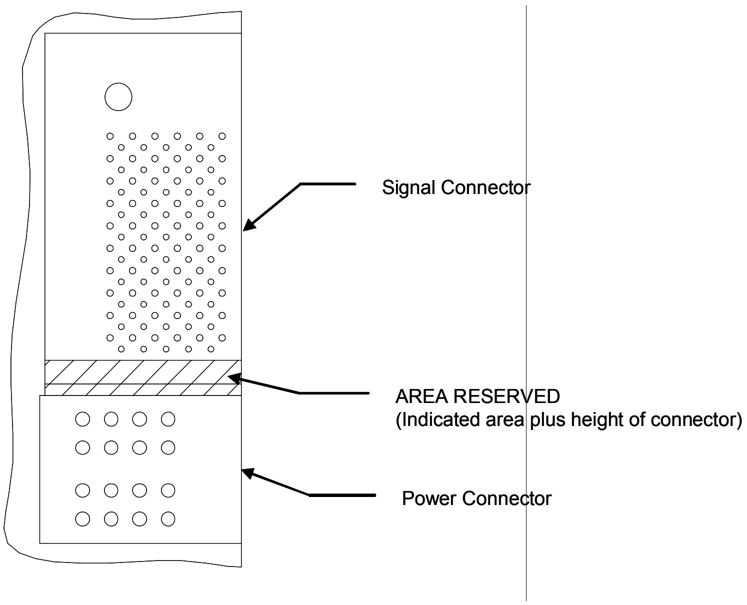


Figure 4.2b – Reserved area around the signal and power connectors (applies to both locations)

Figure 4.3 defines the location of the mezzanine card connectors relative to the Blade Board datum A and B and the blade board top surface. As can be seen from the figure, these two connectors are intended to be mounted on a card that is located parallel to the blade board with a distance of 18 mm between boards.

The card profile and mounting technique for the Mezzanine card is beyond the scope of this specification.

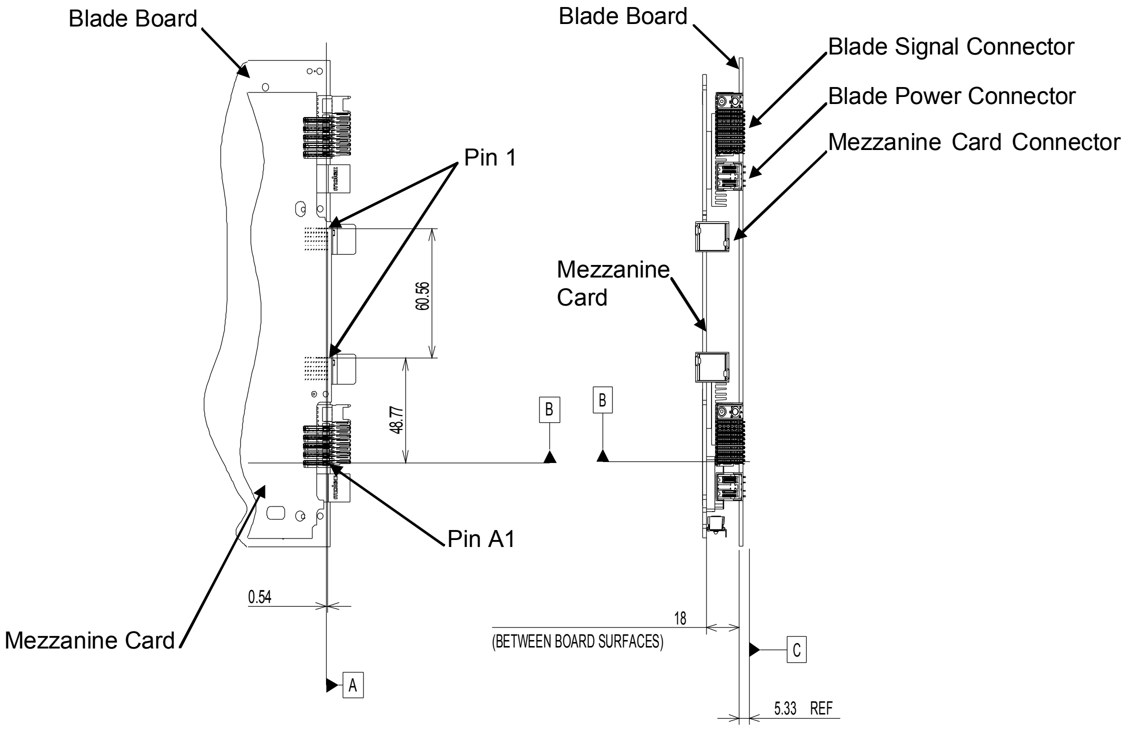


Figure 4.3 - Mezzanine Connector Location

Areas of the blade board and components that may be accessible to a user when a blade is installed in a BladeServer chassis should not have unprotected 240 VA circuits. The 240 VA keep-out area shown in Figure 4.4, which is depicted by a cross hatched area, is required to prevent access to the blade from the rear of the BladeServer chassis when a blade is installed. Other areas, not shown in the diagram, may be required to be protected if a blade component, such as a front bezel or cover, can be removed without tools and then installed in a BladeServer chassis and powered. Consult with your safety agency representative for detailed requirements.

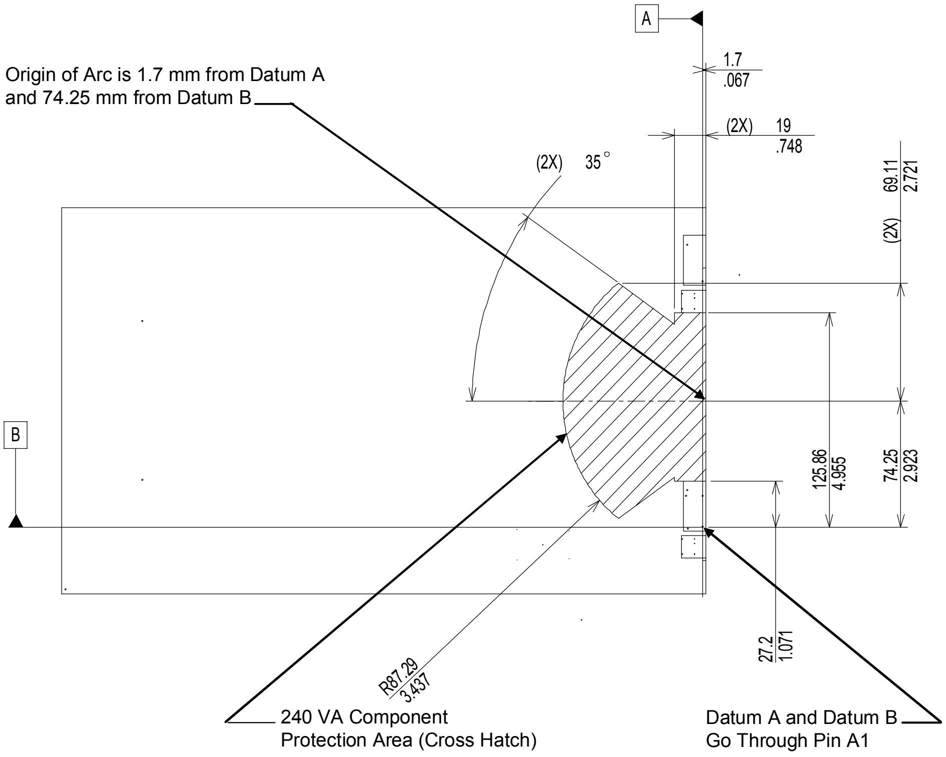


Figure 4.4 – Processor Blade Board 240 VA Keep Out Area

4.3 Processor Blade Enclosure Mechanical Drawings

Below are drawings that provide the required dimensions for a Processor Blade enclosure for both a single and a double wide Blade. Blade widths greater than double wide are acceptable but are beyond the scope of this specification. Datum A and B are the same as those for the blade board shown above. Datum C is defined as the outside of the Blade enclosure near the Blade board. The blade enclosure shown is for reference and may show design details not required for a given blade design. The drawings define the Processor Blade enclosure envelop required to plug into a BladeServer chassis and does not give full details of how such a blade chassis may be constructed. All dimensions are in millimeters (mm). A general tolerance of +/- 0.25mm should apply to all dimensions shown unless otherwise specified. All dimensioned features are critical and must be included in the blade enclosure design unless otherwise noted as an “Optional Feature”. The drawings are not to scale.

The maximum weight of a single wide Processor Blade is not to exceed 5,444 grams (12 lbs.).

4.3.1 Single Wide Processor Blade Enclosure

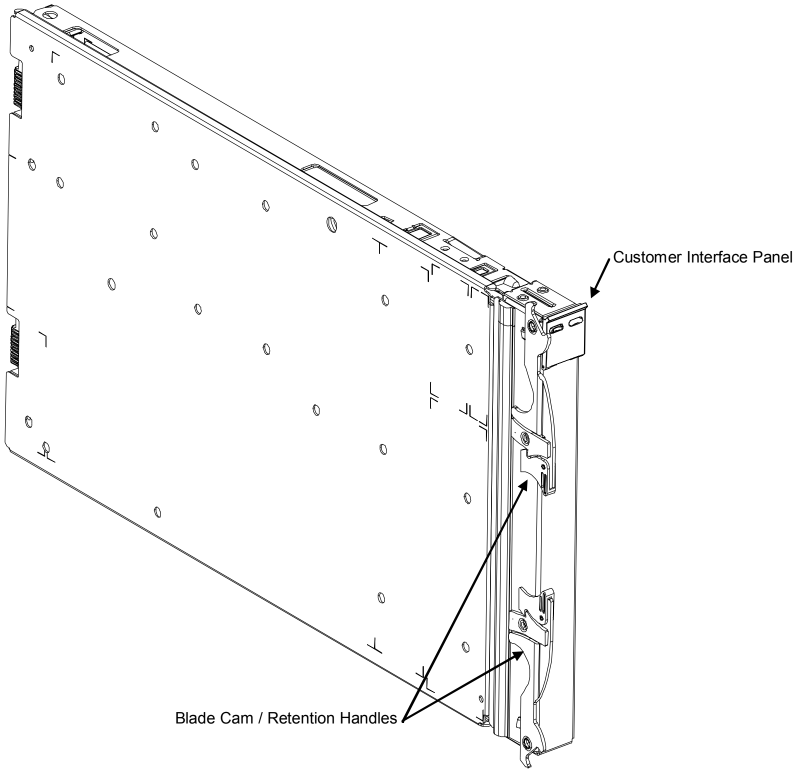


Figure 4.5- Single Wide Processor Blade – Front Isometric View

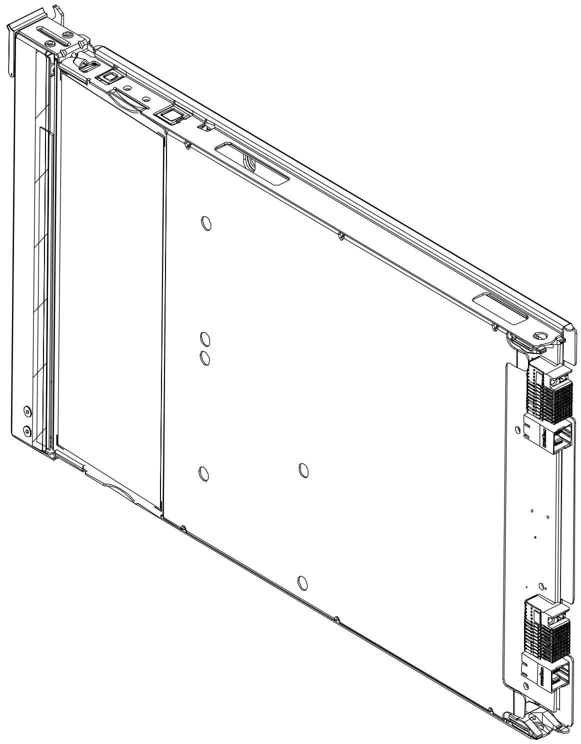


Figure 4.6 - Single Wide Processor Blade – Rear Isometric View

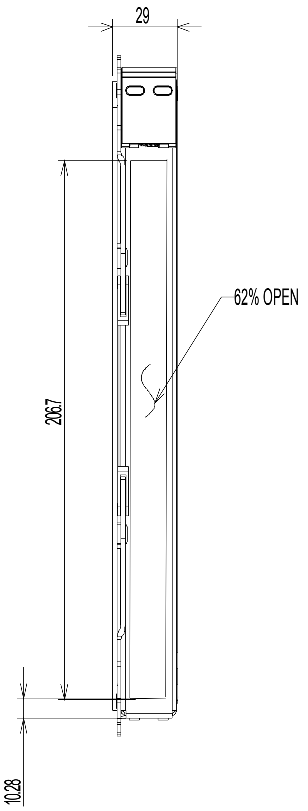
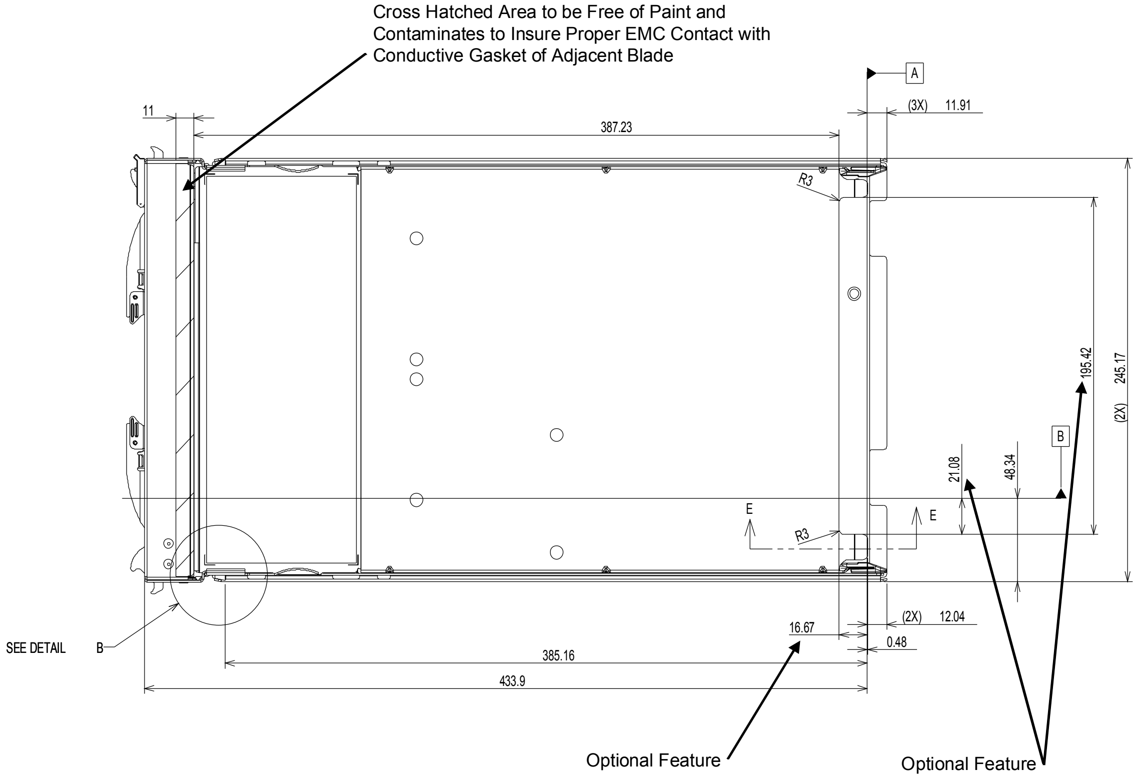


Figure 4.7 - Single Wide Processor Blade – Front View

Figure 4.7 defines the typical perforated air inlet area of 62 percent for a blade. The open area may vary depending on the specific design requirements.


Cross Hatched Area to be Free of Paint and Contaminates to Insure Proper EMC Contact with Conductive Gasket of Adjacent Blade

Optional Feature

Optional Feature

Figure 4.8 - Single Wide Processor Blade – Right Side View (Connectors Are Not Shown)

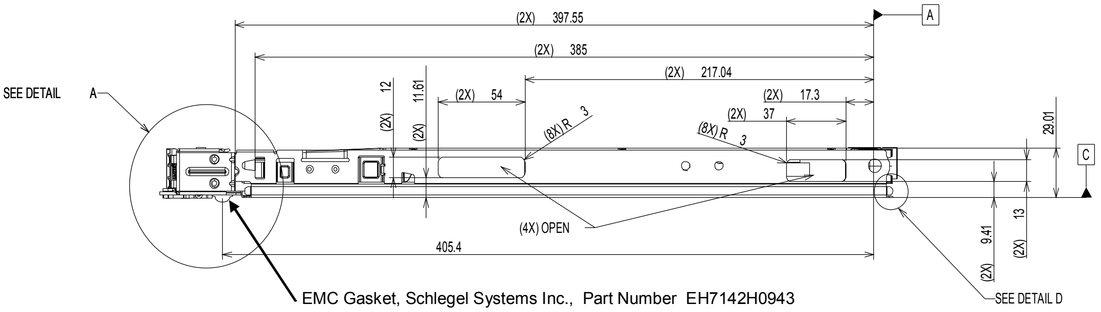

EMC Gasket, Schlegel Systems Inc., Part Number EH7142H0943

Figure 4.9 - Single Wide Processor Blade – Bottom View

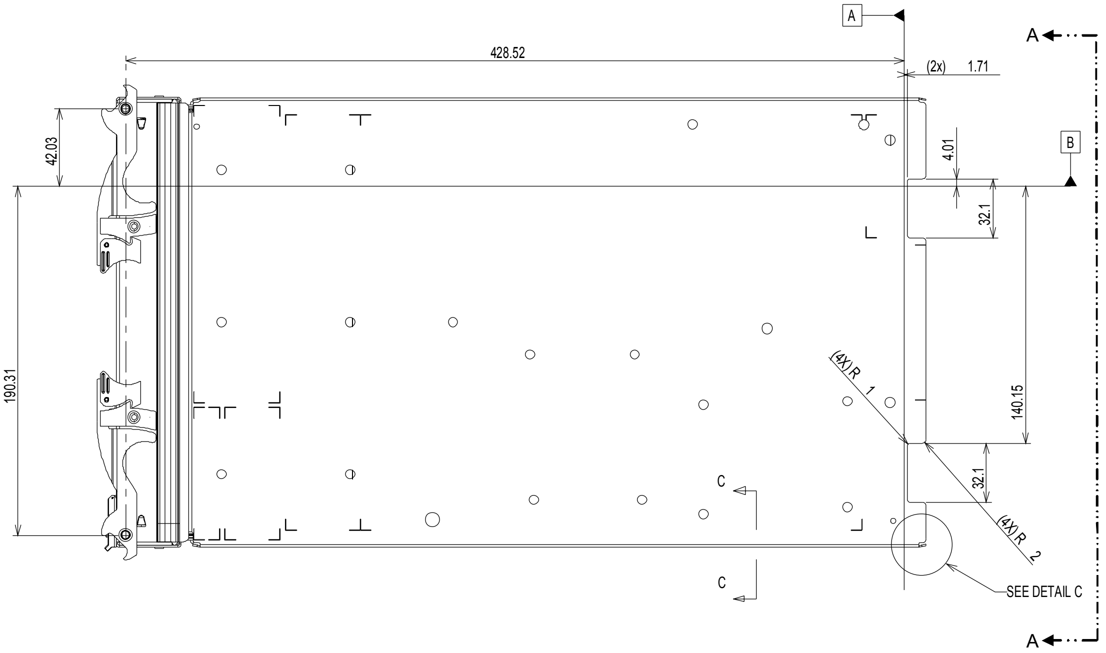


Figure 4.10 - Single Wide Processor Blade – Left Side View

Detail B in Figure 4.11 defines the height and style of a domed half shear which is must be included on the top and bottom of the blade. The purpose of this feature is to provide a stop for the blade relative to the BladeServer chassis during shock and vibration. Note that this feature must be constructed using a domed half shear versus a bridge lance in order to maximize strength.

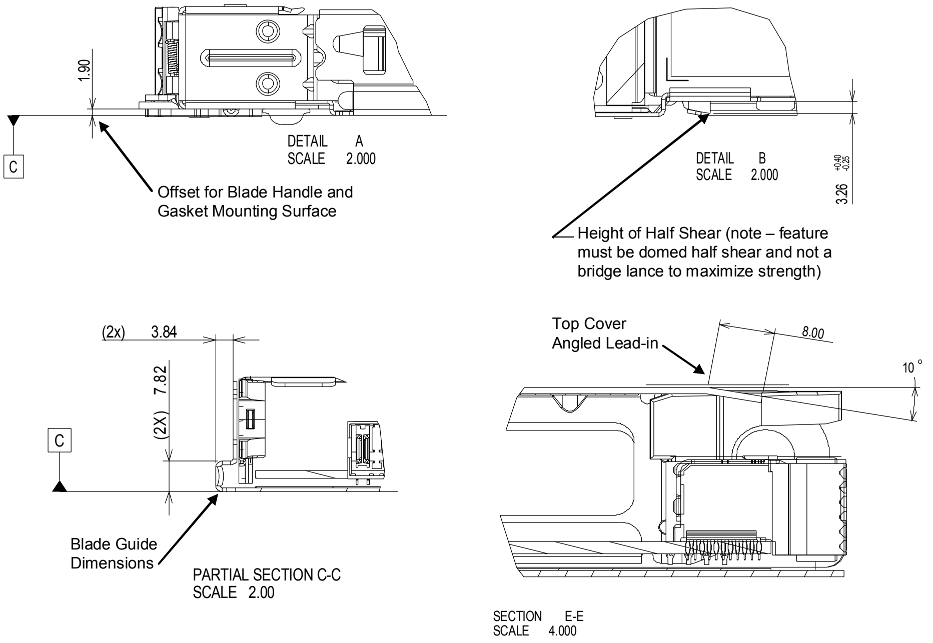


Figure 4.11 - Processor Blade Section and Detail Views

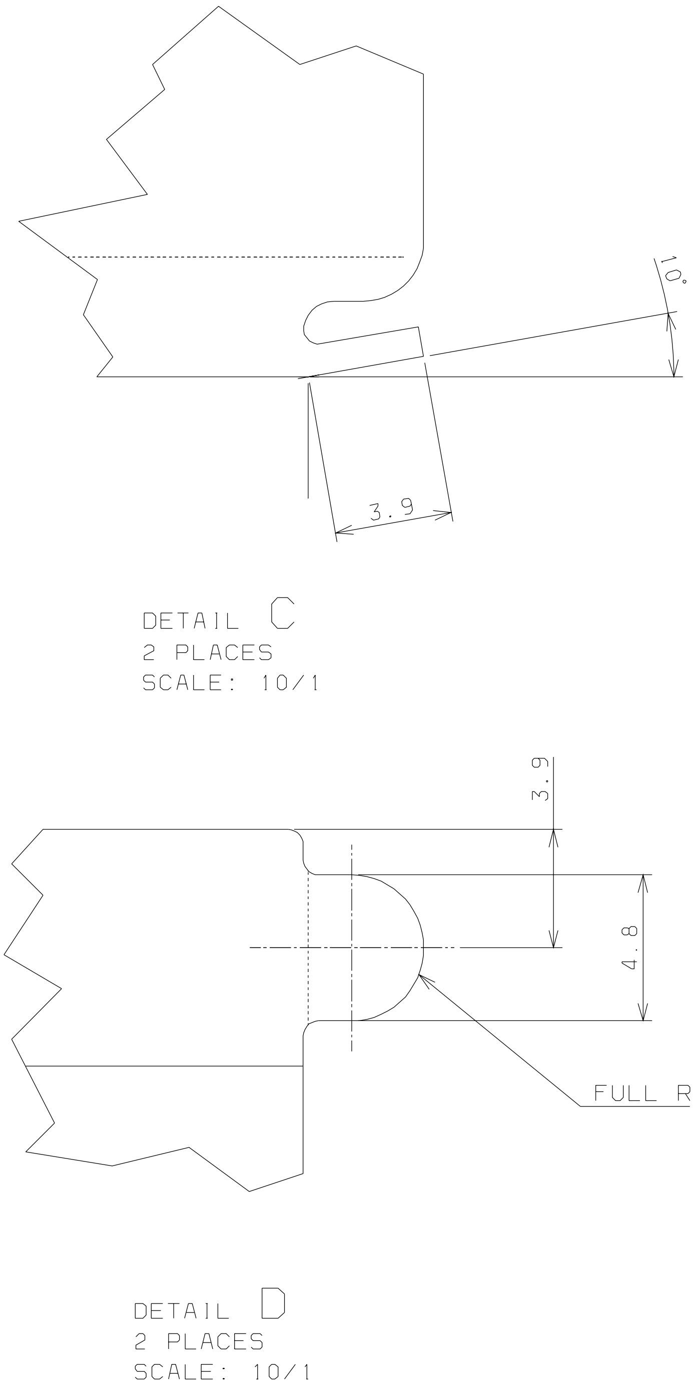
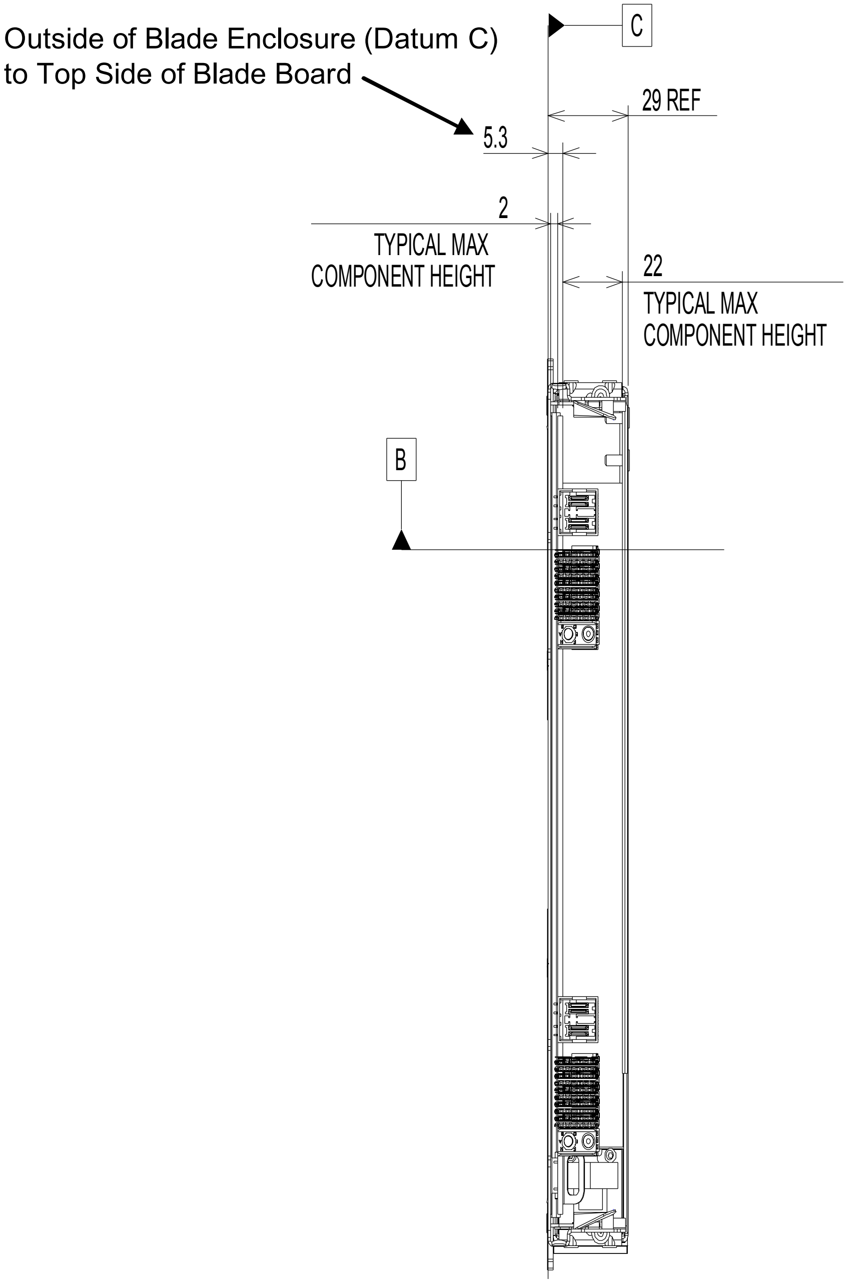


Figure 4.12 - Processor Blade Detail Views

The location of the top side of the blade board to the outside of the blade chassis (Datum C) is shown in Figure 4.13. In addition, typical maximum components heights are provided. The component heights will vary depending on blade board thickness and blade construction.



View A-A

Figure 4.13 – Processor Blade Board Location Relative to Datum C

4.3.2 Double Wide Processor Blade Enclosure

A double wide blade is shown in Figure 4.14. The blade is 59 mm wide which is 30 mm wider than a single wide blade. The figure shows the front bezel as 2 halves, however, a single, double wide blade bezel is acceptable.

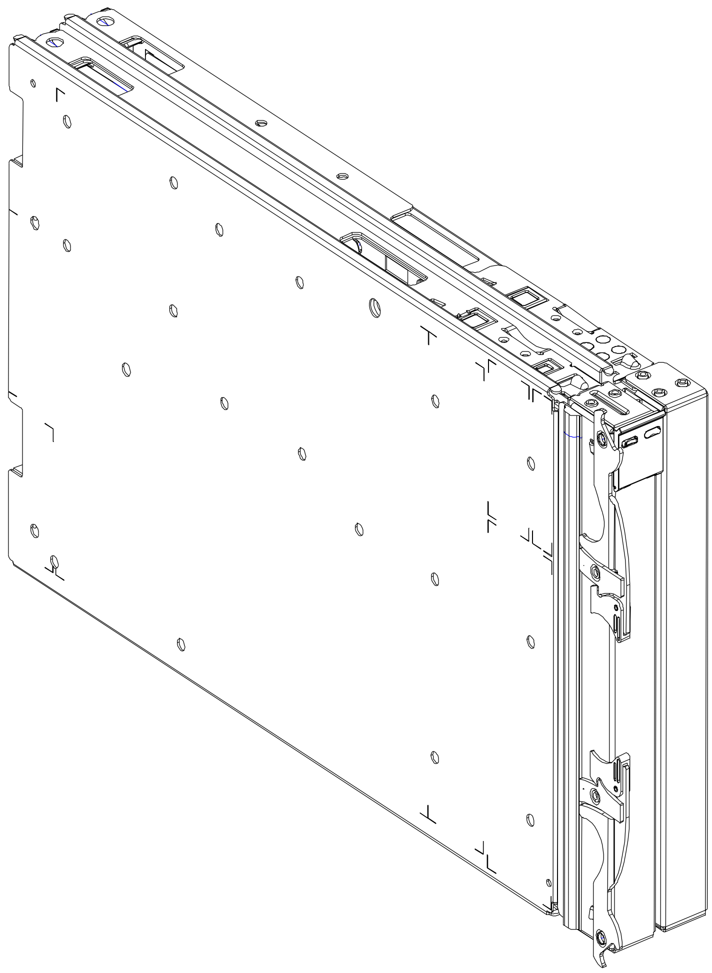


Figure 4.14 - Double Wide Processor Blade – Front Isometric View

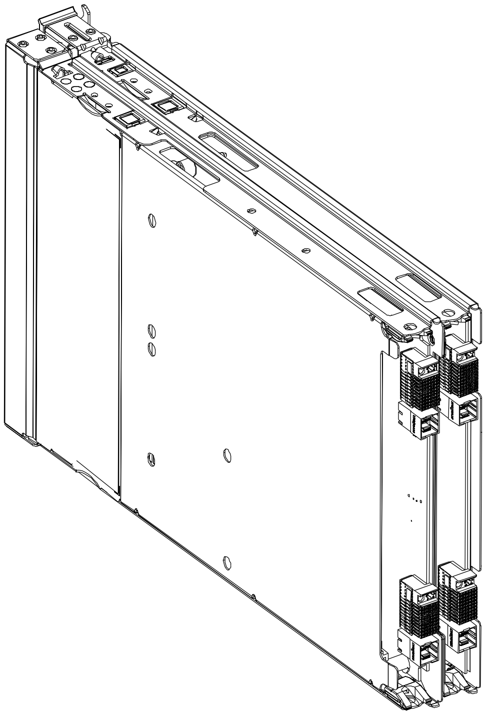


Figure 4.15 - Double Wide Processor Blade – Rear Isometric View

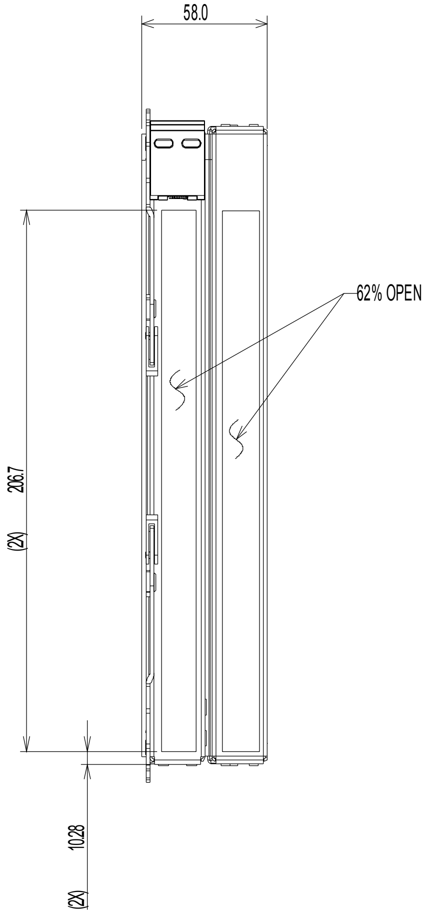


Figure 4.16 - Double Wide Processor Blade – Front View

Figure 4.16 defines the typical perforated air inlet area of 62 percent for a blade server. The open area may vary depending on the specific design requirements.

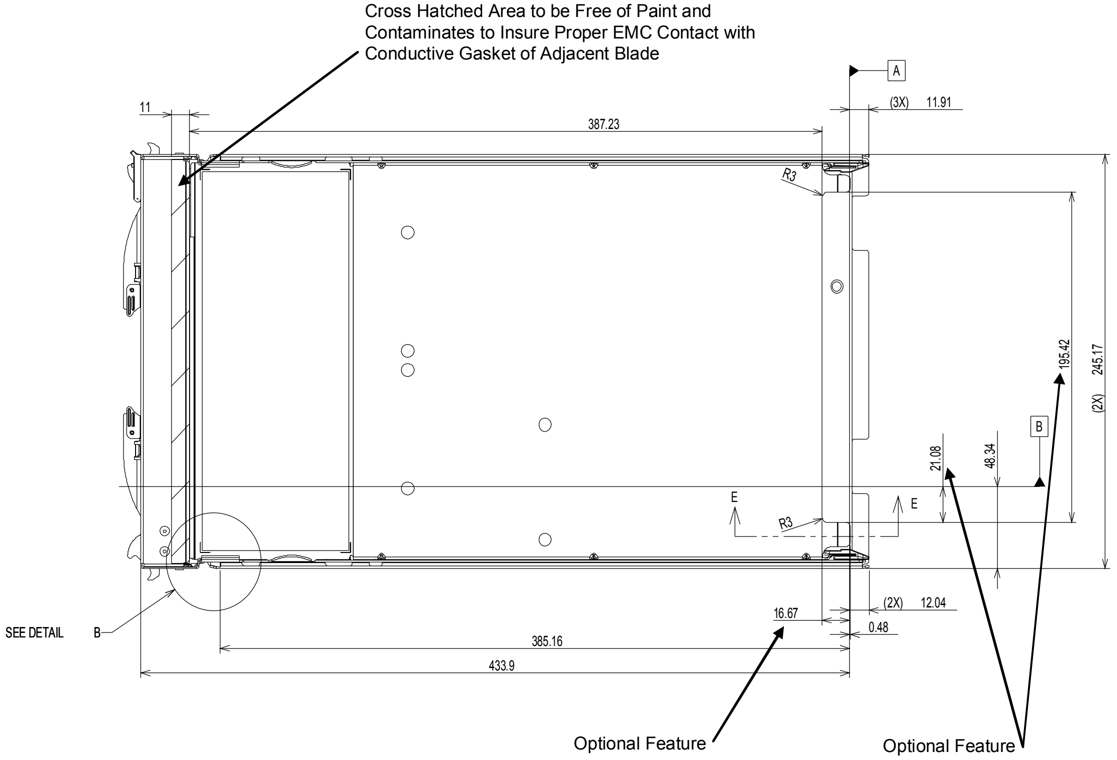


Figure 4.17 - Double Wide Processor Blade – Right Side View

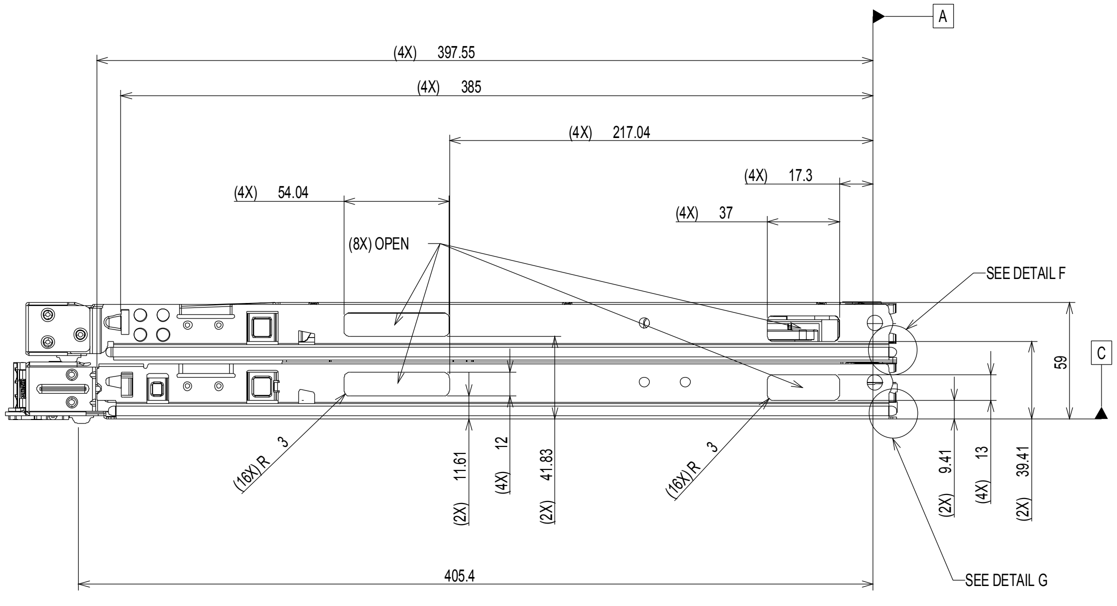


Figure 4.18 - Double Wide Processor Blade – Bottom View

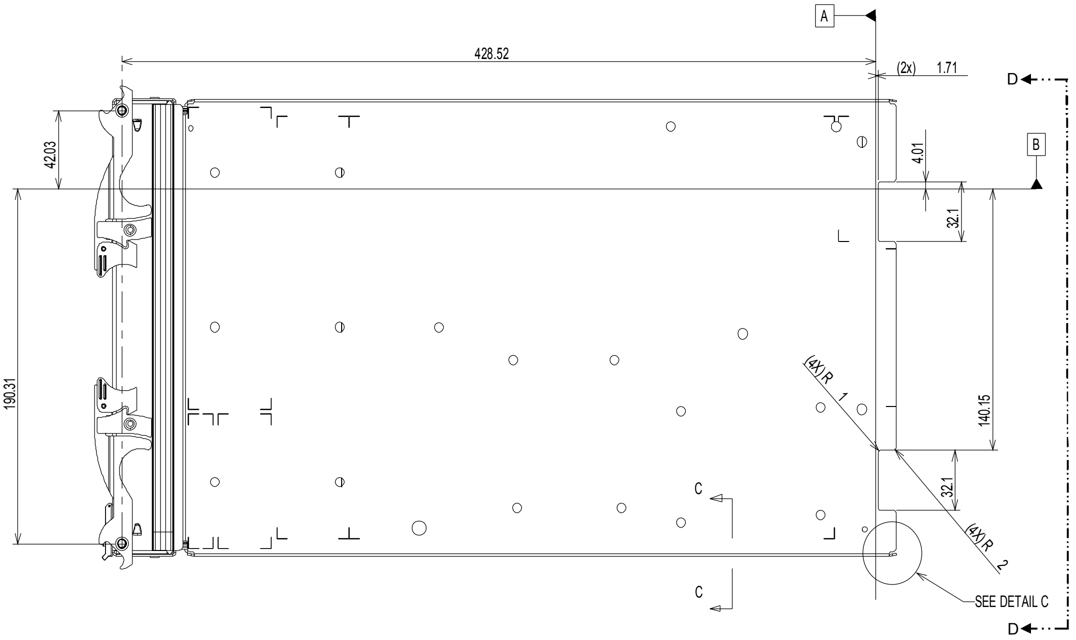
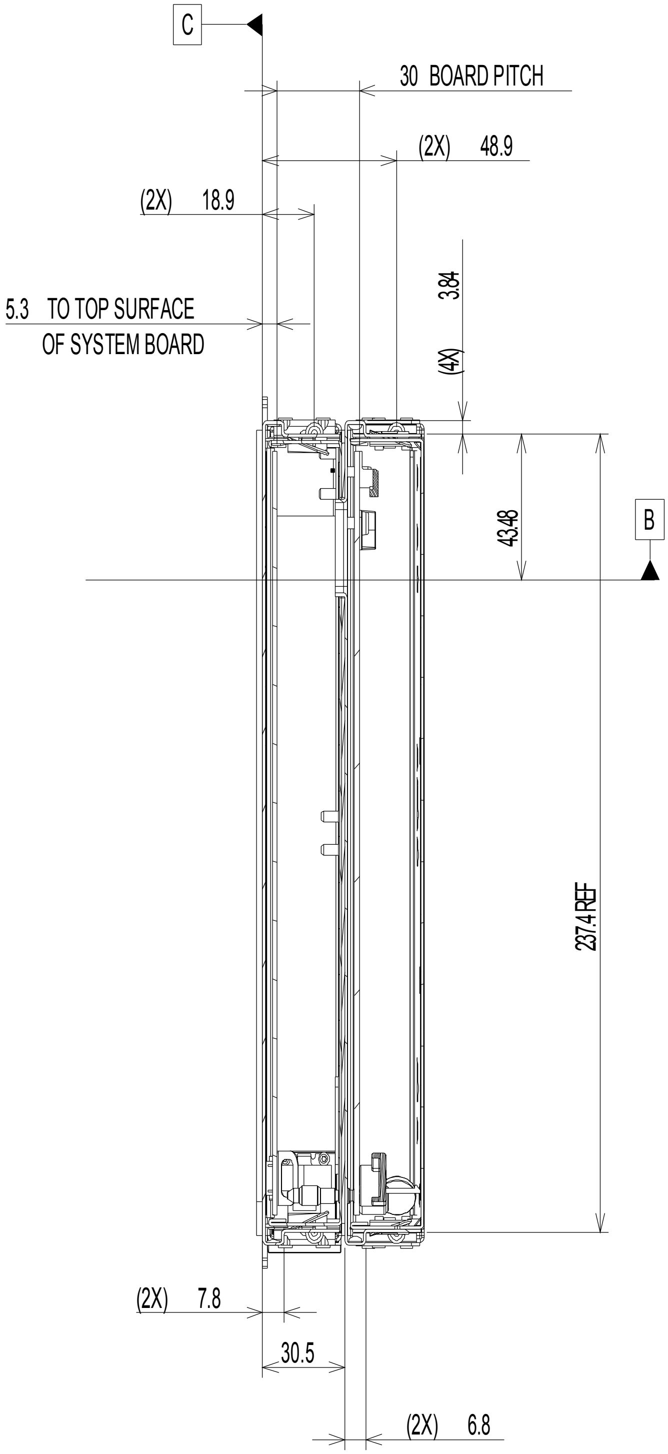


Figure 4.19 - Double Wide Processor Blade – Left Side View



View D-D

Figure 4.20 - Processor Blade Board Location Relative to Datum C and Expansion Board Location

Detail B in Figure 4.21 defines the height and style of a domed half shear which is must be included on the top and bottom of the blade. The purpose of this feature is to provide a stop for the blade relative to the BladeServer chassis during shock and vibration. Note that this feature must be constructed using a domed half shear versus a bridge lance in order to maximize strength

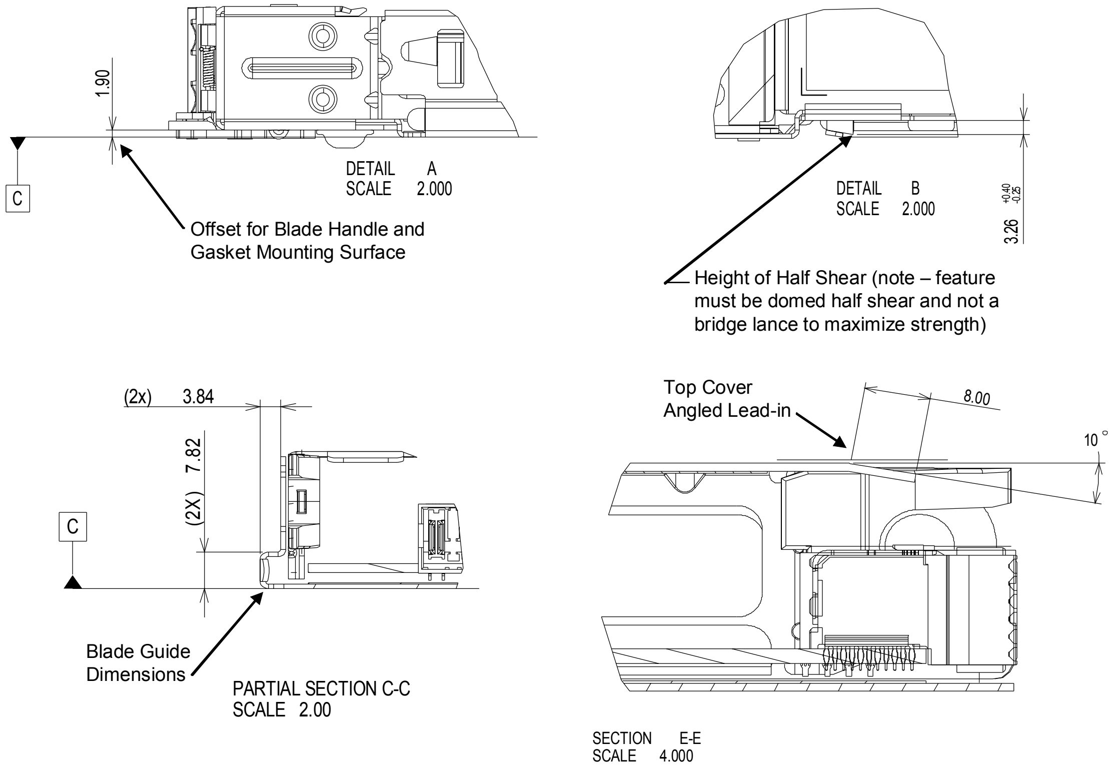


Figure 4.21 - Processor Blade Section and Detail Views

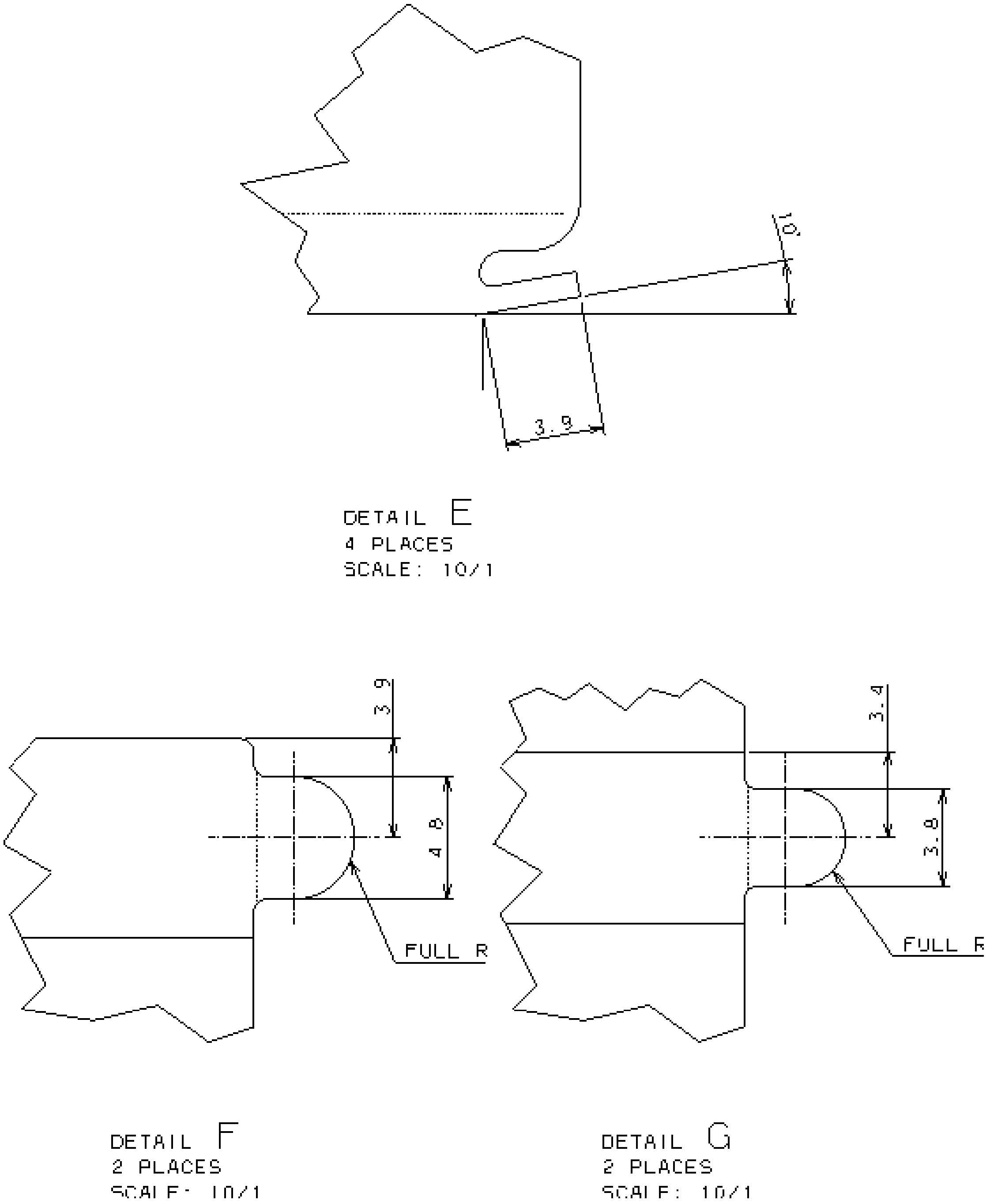


Figure 4.22 - Double Wide Processor Blade - Details

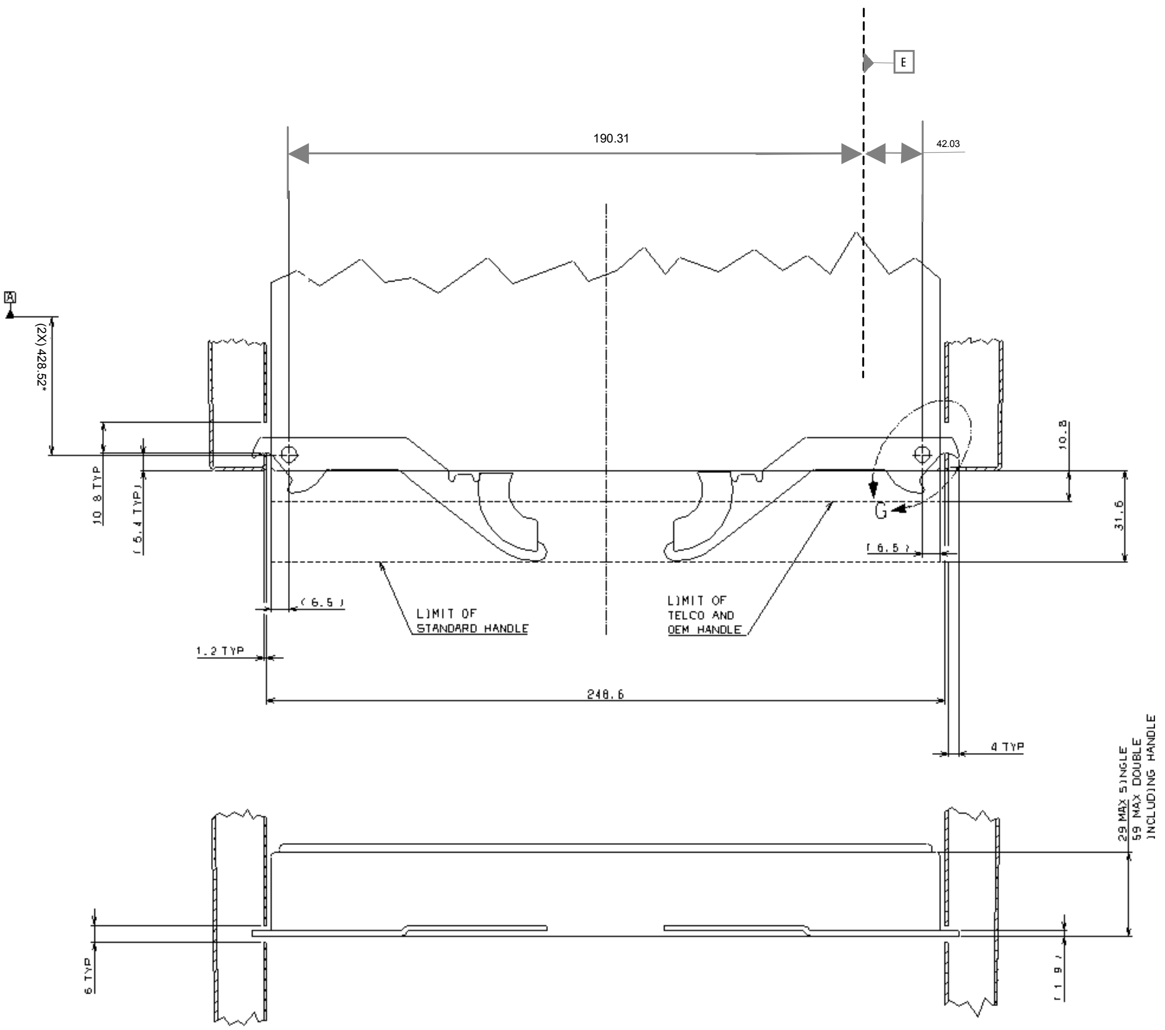


Figure 4.23 - Processor Blade Cam / Retention Handle Details

Figure 4.23 and Figure 4.24 define the Blade cam / retention handle critical dimensions and the interface of the cams to the BladeServer chassis. Depending on the application of the blade, there are limits to the dimension of which the blade handle can protrude from the front of the blade.

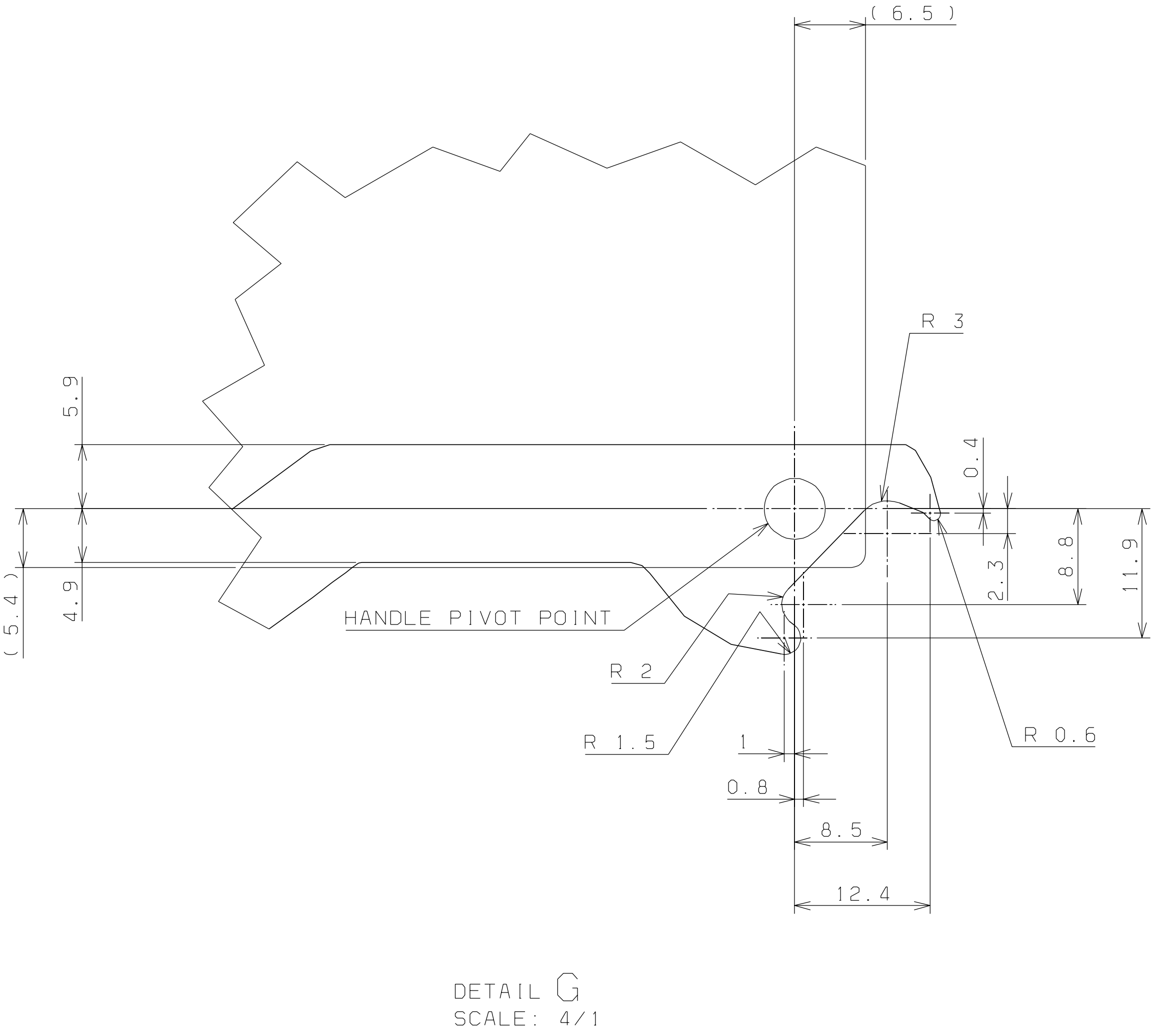


Figure 4.24 - Processor Blade Cam Handle Details

4.3.3 Reserving Label Space for End User

It is important that an adequate area is available to the end user for applying labels (e.g., asset tag). Therefore there shall be a customer usable label area that can accommodate a minimum label size of 7mm x 40mm that is visible without removing a blade from a system and will not inadvertently block air-flow. The current lower handle on the blade bezel, as described in the mechanical design kit, is designed to provide this area (see Figure 4.24).

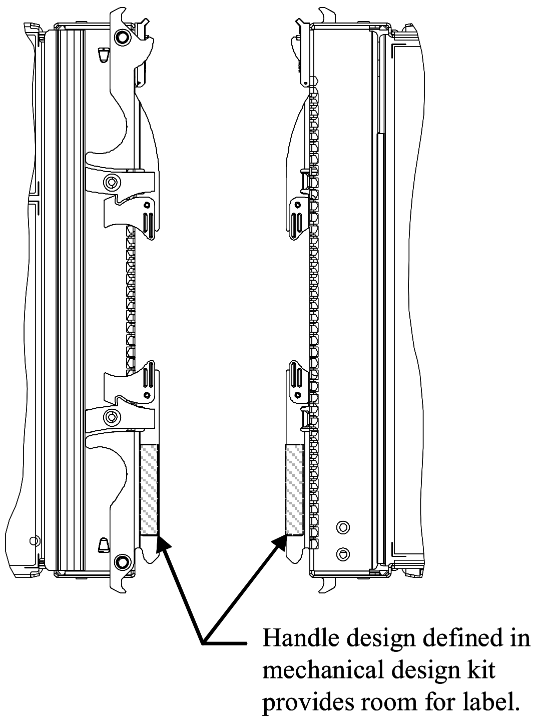

Handle design defined in mechanical design kit provides room for label.

Figure 4.24 – Example of Label Space on Lower Handle of a Blade

5 IPMI Management Interface

5.1 Introduction

The purpose of this section is to define the BladeServer IPMI OEM Architecture that must be adhered to by Processor Blades. The intended audience is Processor Blade IPMI Firmware developers working on IPMI compliant Processor Blades to be used in the BladeServer chassis.

The OEM Architecture requirements defined by this document do not replace any of those requirements established by the IPMI standards, but instead build upon and augment the IPMI standards with additional OEM requirements for proper system wide BladeServer operation.

The BladeServer IPMI OEM Architecture defined by this document is targeted for Processor Blades that are IPMI Version 1.5 compliant.

5.1.1 Definitions

AMM	Chassis management module component. AMM is the Advanced Management Module and is the 2nd generation of the Management Module for BladeServer.
BladeServer Chassis	A chassis that hosts Processor Blades (hardware and firmware) and provides shared network and storage switches, shared power supply, shared cooling system, and various interconnections through a common midplane. There are various chassis Types that include BladeServer, BladeServer High Speed (BladeServerH), BladeServerT (BladeServer Telco) and BladeServer HT (BladeServer High Speed Telco).
BMC	Baseboard Management Controller is the local management controller on a Processor Blade.
CEL	Chassis Event Log. The Event Log maintained on the MM.
DEM	Directed Event Message. An unsolicited message containing SEL data sent from a Processor Blade to the MM for the purpose of logging and alerting.
DES	Directed Event String. An unsolicited message containing a formatted ASCII string for the purposes of logging.
DIMM	Dual In-line Memory Module.
FRU	Field Replaceable Unit. Data, such as part number, serial number, etc. pertaining to a particular replaceable part, such as replaceable memory (DIMM), fans, power supplies, etc.
IPMI	Intelligent Platform Management Interface.
Management Module (MM)	Chassis management module component.
OEM Command	Additional IPMI command extensions beyond those defined by

	the IPMI Standards Specification.
Processor Blade	A complete compute node with some combination of processor(s), memory, network interfaces, with supporting hardware and firmware. The Processor Blade may occupy one or more blade slots in the BladeServer chassis.
SDR	Sensor Data Record (see IPMI Specification).
SEL	System Event Log (see IPMI Specification).
SMS	System Management Software (see IPMI Specification).
SOL	Serial over LAN. A terminal serial communication substitute performed on network media normally used by systems that lack serial connectors.
Solicited	This term is used in relation to the MM. Solicited means messages sent by the MM to the Processor Blade and a response is expected to be returned.
SPD	Serial Presence Detect. 256 bytes stored in EEPROM on a Dual In-line Memory Module (DIMM), containing FRU data, timing characteristics, etc.
Switch Module (SM)	Switch Module, providing switching services for blade communication media (Ethernet, Fiber-channel, etc.).
Unsolicited	This term is used in relation to the MM. Unsolicited means messages originated by the Processor Blade and not in response to a message sent by the MM.

5.1.2 AMM/MM Support for Processor Blade’s

The information provided in the following sections describes the architectural definition of the Processor Blade and AMM (Advanced Management Module) interface. The version of this specification titled ‘Base Specification for Processor Blade Subsystems, version 2.07’ described the MM architectural interface. The AMM provides enhanced functionally that the MM will not provide. Although the AMM provides for backwards compatibility and support of Processor Blades that implemented the previous versions of this specification there are new requirements stated in this specification that must be implemented to ensure correct BladeServer Chassis operations when an AMM is installed in the chassis. The basis of this version of the specification assumes that the AMM firmware is at ‘Version 1.32D, Build ID: BPET32D’ or later. The term MM is used generically in this version of the document but implies an AMM. As noted previously the AMM is the second generation of the MM and requires additional functions from the Processor Blade than the first generation MM.

5.1.3 Related Documents

- BladeServer - Base Specification For I/O Expansion Cards
- BladeServer - Base Specification For Switch Module Subsystems
- BladeServer - Base Specification For VPD

- BladeServer - Base Specification For External Systems Management Integration
- Intelligent Platform Management Interface Specification, Version 1.5, Feb 20, 2002

5.2 BladeServer IPMI OEM Architecture Description

The term '*BladeServer IPMI OEM Architecture*' is used to describe the additional requirements for blade server systems from multiple vendors to operate in a uniform and pre-deterministic manner in the BladeServer chassis. This is in addition to those requirements defined by the IPMI standard.

Chassis resources shared amongst multiple systems from multiple vendors must be managed by the MM in a coordinated manner for all the Processor Blades. The *BladeServer IPMI OEM Architecture* is IPMI compliant and therefore used to build upon and augment, not replace, IPMI methods, while at the same time providing BladeServer vendors a standard command set

5.2.1 Architecture Hierarchy

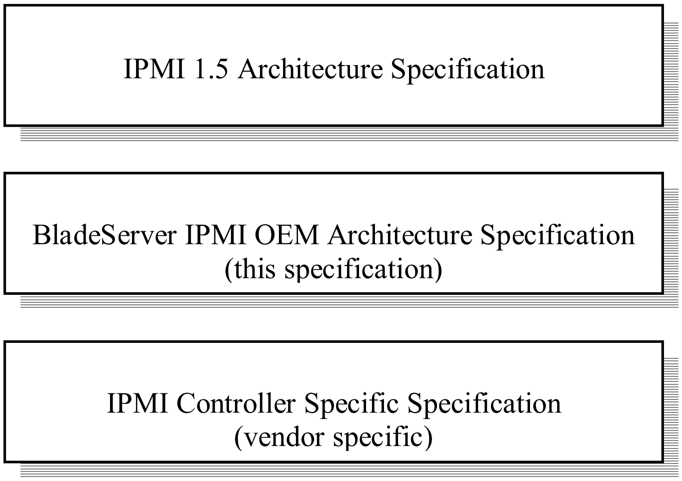

IPMI 1.5 Architecture Specification
BladeServer IPMI OEM Architecture Specification
(this specification)
IPMI Controller Specific Specification
(vendor specific)

Figure 5-1 IPMI Architecture Hierarchy

This section is targeted to Processor Blade BMC Firmware developers. BMC Firmware developers must ensure that their designs comply with a hierarchy of architectures, with IPMI Architecture specifications (IPMI Version 1.5 for this document) as the highest hierarchical layer, and *BladeServer IPMI OEM* Architecture being the next layer of hierarchy. The Processor Blade controller specific architecture layer must comply with higher levels of the architecture hierarchy.

Processor Blades are free to extend their IPMI command stack and capabilities, but must ensure all controller specific OEM commands do not violate any other higher level hierarchy architecture.

An example of this hierarchical compliance is that a Processor Blade would not be allowed to add or execute a controller-specific OEM command that sets general power permission (not including local power control, i.e. Wake on LAN, local power control, etc. once power permission is granted). This document defines that general power permission must only be granted by the MM to the Processor Blade and provides BladeServer IPMI OEM command extensions to do so. Because of this architecture definition by the BladeServer IPMI OEM Architecture, Processor Blades must not extend their own controller specific OEM commands (OEM commands defined by Processor Blade products) to perform or in any way affect shared resources such as Power Permissions.

5.2.2 BladeServer IPMI OEM Commands

BladeServer IPMI OEM Architecture defines additional commands and behavior for the BladeServer to provide for chassis resource sharing, power management and system management.

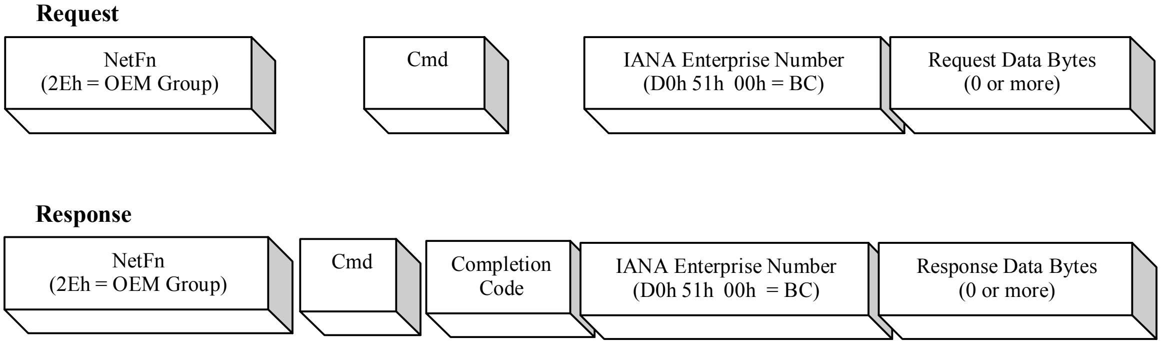


Figure 5-2 OEM Group Command

Figure 5.2 represents the IPMI Request and Response messages with only the OEM specific fields of interest shown. Addresses, sequence numbers, LUN, checksums, etc. have been intentionally left out to avoid any interface specific reference. Only those fields of importance to clarify the BladeServer IPMI OEM command definitions are shown and discussed.

The BladeServer IPMI OEM Command extensions are defined by this document and are grouped into the OEM Group Network Function 2Eh. The first three bytes of the request data and the first three bytes after the completion code in the response data includes the IANA assigned Enterprise Number for 20944d (least significant byte first). This is in compliance to the OEM Group defined by the IPMI Version 1.5 Specification. The IPMI OEM Group command set IANA assigned *'Modular Blade Server'* value 20944d (0051D0h, least significant byte first). is the only IPMI OEM Group command set supported by the BladeServer IPMI OEM Architecture.

The BladeServer IPMI OEM commands may be safely added to any IPMI command table without any danger of redefining controller specific OEM command extensions previously supported.

The BladeServer IPMI OEM command set will only be supported on the RS485 private interface between the MM and Processor Blades. These commands must never be accepted on any other interface, whether session-based, session-less, or the internal system interface. The only exception is for the commands in section 5.19, *Get Pre-Set Watts, Set Component Watts Inventory, CPU Throttle Table* and possible ' BladeServer Examples of IPMI Controller Specific OEM Commands' in section 5.19.1.2 which can be sent to the BMC by host firmware (not from the OS executing on the Processor Blade).

5.3 System Management

System Management in this section of this document refers to the ability of the Management Module (MM) to monitor and control a BMC, for the purpose of monitoring the system health of a blade as well as control of a blade within the BladeServer chassis. These functions include but are not limited to blade configuration, monitoring system information, such as temperatures, voltages, fans, bus errors, physical security, logging system events, alerting users, auto recovery from failures, and providing inventory information to the MM.

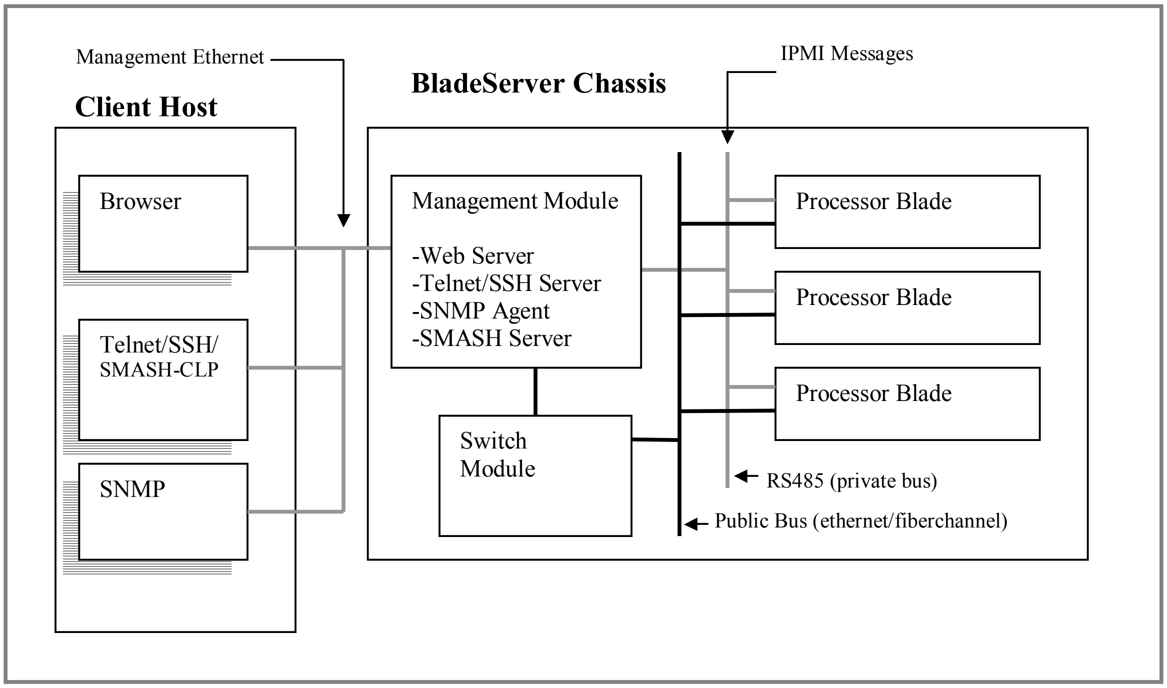


Figure 5-3 System Management Interfaces

Figure 5.3 shows a BladeServer chassis with a Management Module (MM) Switch Module (SM) and several Processor Blades. Connectivity from Processor Blades and the MM is provided by a private bus known as the RS485 bus. This bus and bus protocol must be adhered to by all Processor Blades.

The MM to Processor Blade interface exchanges IPMI messages between the MM and Processor Blade's BMC (encapsulated in the proprietary RS485 packets) for blade management.

IPMI based Processor Blades have the potential capability to be managed by multiple interfaces, such as using IPMI commands by a Host SMS IPMI application running on the blade (in-band), a side-band interface such as the MM over the RS-485 bus using IPMI commands, and via an external chassis connection such as Ethernet ports using IPMI commands (out-of-band).

For BladeServer compliant Processor Blades, all blade management and configuration will be performed via the side-band MM RS485 interface to a BMC. The BMC must block management and configuration commands that might alter shared chassis resources on the BMC IPMI channels and the Host Interface to the BMC. In addition, it is the responsibility of the Processor Blade firmware implementation that supports multiple applications to prevent the MM interface from being non-responsive for any reason, such as excessive commands issued to the BMC from an in-band Host application

The Processor Blade BMC should be implemented in such a way that the MM does not need to consider the existence of other applications that may be communicating with the Processor Blade.

Out-of-band access utilizing the Processor Blade's IPMI LAN channel(s) to allow IPMI management of the Processor Blade must be prohibited.

Processor Blades can be managed from external applications by connecting to the MM's session based user interfaces via the MM's external Ethernet. The MM therefore acts as an aggregation point for management of all the chassis components including the processor blades. Multiple interfaces into the MM are available, including CLI, SMASH CLP, SNMP, Web, etc. The advantage of this single access point interface is that a application can manage all the Processor Blades and the chassis concurrently, with no additional connection requirements.

5.4 IPMI Command Policy

The BladeServer chassis consists of chassis resources (i.e. power supplies, fans, and shared peripherals such as a KVM, USB ports and DVD/CD-Rom device) which are shared between multiple Processor Blades. The intent of this section is to identify the need for Processor Blades to reject or block IPMI commands which will affect shared resources within the chassis.

IPMI commands targeted to a single blade are required to only affect that blade and from a chassis management perspective, must not affect another blade or reduce the shared chassis resources (see section 5.2.1) of another chassis component either directly or indirectly. Therefore, an IPMI Command Policy is required to prevent in-band and out-of-band IPMI commands from having an undesired impact on the chassis.

IPMI out-of-band access via the Processor Blade LAN channel is not to be made available for external users for management purposes, applications running on the host are free to send any IPMI command supported by the blade as long as the restrictions discussed in section 5 are adhered to.

For Processor Blades that are in compliance with this architecture document and wish to reduce the security exposure of their internal interface (i.e. any application on the Host that issues IPMI commands), a product specific implementation to block certain IPMI commands on its internal interface may be supported. This is strictly a product based solution and is not defined by this document.

5.5 Processor Blade Chassis Initialization

Processor Blade Chassis Initialization refers to the process between the Processor Blade and MM that is executed upon Processor Blade Discovery. This process is executed upon any of the following events:

- On power up of the BladeServer chassis. MM will perform Processor Blade Chassis Initialization to all blades[1].

- The Processor Blade is hot-plugged into the chassis. Blade is discovered by hardware presence detected by MM[1].

- The Processor Blade undergoes a BMC reset, such as the BMC exiting firmware mode and going operational. New Blade configuration is discovered by IPMI messages (*Blade State Change)* sent from Processor Blade to MM[1].

- The first MM being plugged into an operational BladeServer chassis.

- The MM going through a reset sequence in an operational BladeServer chassis.

The Processor Blade is detected in the BladeServer by the MM with the use of the presence detect pin on the BladeServer chassis mid-plane. The Processor Blade BMC will set its power permissions to off and ignore any other input for power requests by any SMS until its power permissions are changed by the MM.

Figure 5.4 presents the sequence of events which occur upon Processor Blade discovery. Message flows are between the MM and the Processor Blades using the internal RS485 private bus.

Note (1) The BMC implementation should only require several seconds (5-10 sec) to perform its internal initialization in order to respond to the message flows described below.

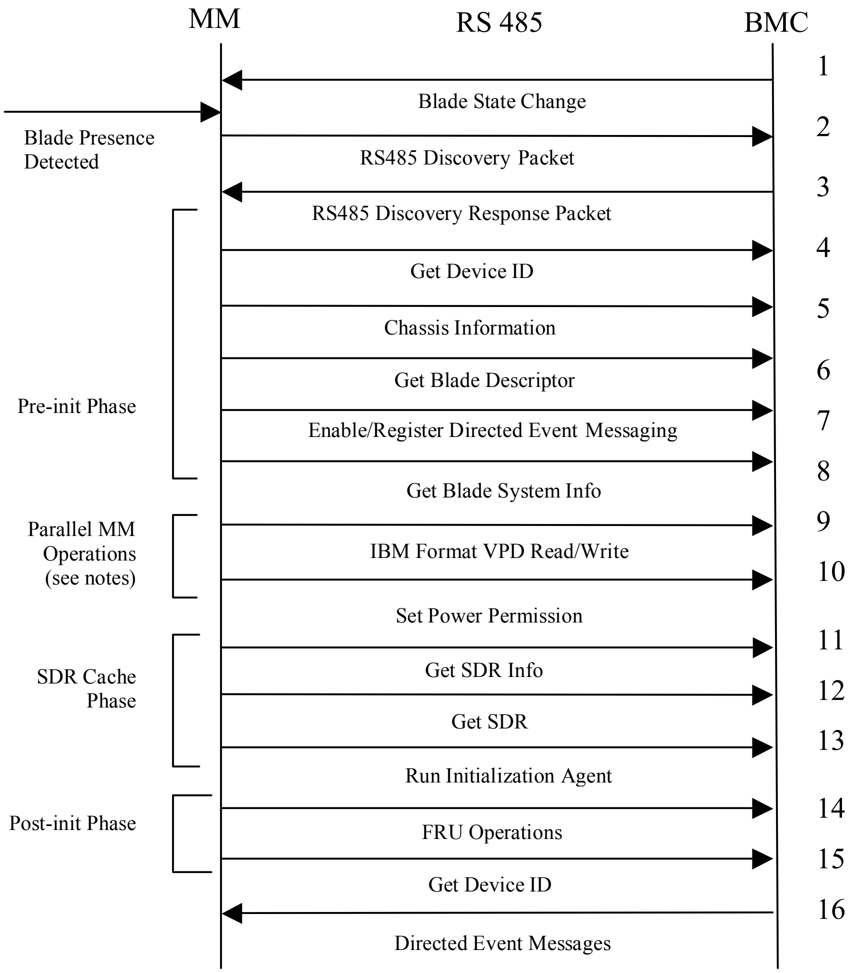


Figure 5-4 Processor Blade Chassis Initialization and Insertion

Note that Figure 5.4 does not generally show the IPMI responses from the Processor Blade or MM. Except where a response needs to be described, the response flows are not shown. A failure or timeout response to an IPMI command issued by the MM must result in a retry by the MM. A rejected response may result in a failure to give power permission to the blade.

The initialization flows are broken into several phases. The pre-init phase is performed to obtain the minimum amount of information from the Processor Blade so that power permission may be granted as soon as possible, allowing the Processor Blades to power up quickly. Note the following: (1) any events sent from the BMC to the MM are ignored and discarded until the post-init phase is completed, (2) The inventory collection (VPD reads and writes) and the details of granting power permission flows are not shown (for further information see section 5.7.5) These are performed after the pre-init phase and may occur in parallel with the SDR caching phase.

Although the flow shown is the current order in which commands are sent to Processor Blades from the MM, the MM is free to change the order and issue any of the listed IPMI commands at any time in any sequence.

The inventory collection (VPD reads and writes) and power permission flows are not shown.

The following descriptions are to be used with the flows presented in Figure 5.4. Note that any failure before the power permission step will prevent power permission from ever being granted.

1. Before initialization, the IPMI blade may send messages, such as "Blade State Changes" (BSC). For IPMI blades that are BMC reset or flashed, the BSC message is used to trigger the MM to initiate the "RS485 Discovery Packet", which is done when the blade is already physically present in the its slot and has been previously "learned" by the MM. The BSC informs the MM that the Processor Blade's BMC has performed a reset. For hot-plugged blades, the hardware "Blade Presence Detected" is used to initiate the "RS485 Discovery Packet" and any other traffic from the IPMI blade, such as a "Blade State Change", is ignored. Processor Blades must recognize the pre-"Processor Blade Chassis Initialization" state and avoid sending the *Blade State Change* message, which only serves to cause unnecessary arbitration of the RS485 bus.

2. Once the IPMI blade is detected (either by hardware presence detect or Blade State Change), an RS485 Discovery Packet is sent to the IPMI blade directly. The *RS485 Discovery* packet is not broadcast from the MM, but is addressed to the specific blade.

3. The IPMI blade returns an RS485 discovery response packet. This will inform the MM that the responding blade uses IPMI for its server management and communications protocol.

4. The MM will then sends the mandatory IPMI "Get Device ID" message. The response to this message identifies the unique manufacturer and product IDs for the Processor Blade. This message also identifies to the MM the operational state of the IPMI blade's BMC. If the BMC is not in an operational state the blade will be noted as in a **"Firmware" state** and further initialization will be greatly simplified, using only the *"Get Blade Descriptor"* message (explained in next step 6) before completing blade initialization. The blade is considered in "non-operational" state and only accepts a subset of IPMI commands (see Table 5.1) including Firmware flashing commands. If the BMC is operational, the blade is considered operational and will continue to follow this initialization sequence.

5. The MM will then send the "Chassis Information" message that provides the BMC with basic information on type of chassis the Processor Blade is installed in. The BMC will send the message response.

6. The MM then sends the OEM command "Get Blade Descriptor" to obtain required data from the targeted Processor Blade. This command must be responded to regardless of the operational state of the Processor Blade's BMC

7. The OEM IPMI "Set Directed Event Message Enable" will then be sent by the MM. This informs the Processor Blade to start sending all SEL events to the MM. This also allows the MM to disable the DEM.

8. The MM will then send the OEM "Get Blade System Information" message to obtain further information from the BMC.

9. The MM will now perform several OEM commands known as “IBM Format VPD Read” and “IBM Format VPD Write” commands. These commands will write/read further required MM data to/from the blade, including but not limited to blade configured system/blade name and chassis information such as type with associated UUID.

10. After the pre-init phase is completed, the MM will send the *Set Power Permission* message if the pre-init phase has recognized the Processor Blade and its installed hardware options (I/O expansion cards, blade extensions, etc). This is done in parallel with the SDR caching phase. Once power permission is granted, the blade is now capable of operating with Power Domain 2 which contains the majority of the Processor Blade hardware such as the CPUs, Memory and I.O. Operations such as powering on the blade and booting to the Operating System are now permitted.

11. After the pre-init phase is completed and in parallel with step 11, the MM begins its SDR caching phase, where all the IPMI blade’s SDRs are read by the MM. The first mandatory IPMI command used is the “Get SDR Info” command.

12. The MM then sends the mandatory IPMI “Get SDR” command to gather the current SDR Repository contents to the MM. This process will continue until the entire blade SDR’s are cached on the MM. The process follows that described by the IPMI standard (i.e. SDR request size negotiation, reservation, etc)

13. Once the SDR’s are cached, the MM goes into the “Post-init phase”. It’s important to note that unsolicited events from the IPMI blade are ignored until post-init phase is entered. To ensure all hardware sensor alerts were captured, the MM will now send the IPMI defined optional command “Run Initialization Agent” to have the blade’s SDR Repository rebuilt. This is done so that any missed hardware sensor alerts are regenerated and processed by the MM. It is mandatory that the Processor Blade support this IPMI optional command.

14. The MM will obtain FRU data from the blade. The commands used will be IPMI standard FRU Read/Write commands and/or IBM OEM VPD Read/Write Commands. Please see 5.11 FRU Data on page 108 for details.

15. While the SDR Repository is rebuilt on the Processor Blade, the MM will monitor the state of the IPMI blade by sending “Get Device ID” and checking the operation state of the BMC, waiting for the BMC to become operational. Once the SDR Repository is rebuilt and the BMC becomes operational, the initialization process will complete and the blade will be available for system management operations.

16. Any “Directed Event Messages” sent by the BMC will now be processed by the MM and any required alerts generated.

Note: VPD can be written at any time by the MM before or after PD2 is turned on. For example, the History Log in VPD block 2, the Dynamic Block Systems Management Area can be written by the MM before power permission has been granted to inform the blade of the chassis type in the Inform field.

For formats of the OEM commands used in the Processor Blade Chassis Initialization, please refer to 5.19.3.

5.5.1 Initialization of Inoperable Processor Blade (Boot Firmware)

Processor Blades may encounter a problem where the BMC firmware is inoperable. This may be due to Processor Blade firmware corruption, possibly encountered during a failed firmware flash update. This BMC firmware is normally referred to as the “operational” firmware.

The Processor Blade Chassis Initialization is partially executed for a Processor Blade that is in this inoperable state. This message exchange is performed between the MM and the Processor Blade “boot” firmware, which should be available before the operational firmware is loaded.

“Boot” firmware is a general term to describe the initial firmware running on the BMC which supports the initialization of the system and loading of the BMC operational firmware. Other in-band, private firmware flash update mechanisms for the BMC may exist and are outside of the scope of this specification.

The boot firmware is normally protected by some means to prevent corruption due to flashing operations, which prevents the Processor Blade becoming completely inoperable. However, most Processor Blades will provide some technique for boot firmware updates as well, though the techniques are more substantially involved than the operational boot firmware update and are outside of the scope of this document.

Due to the possibility of operational firmware corruption, the boot firmware must be able to communicate with the BladeServer chassis’s MM. This requires the boot firmware to support the following messages in Table 5.1, which is the minimum set of messages required to restore operational firmware.

Table 5-1 Boot Firmware IPMI Command Set

1. RS485 Discovery Packet
2. RS485 Discovery Response Packet
3. Get Device ID
4. Get Blade Descriptor
5. Get Self Test Results
6. Blade State Change
7. Flash Start
8. Flash Data
9. Flash Fill (optional)
10. Flash End

The purpose of having the boot firmware supporting the reduced message set is:

1. To enable the MM to recognize the Processor Blade and obtain minimal information required to allow selection of the proper Operational flash image to load (via remote flash from MM).
2. To inform users of the inoperable state of the blade if the situation arises. The MM and other Processor Blade utilities may then be used to attempt to flash the Processor Blade with a new firmware image.

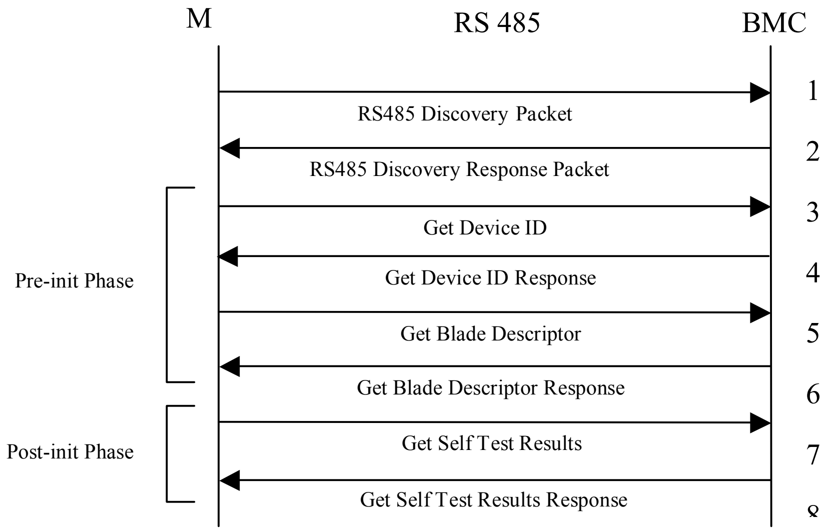


Figure 5-5 Inoperable Processor Blade Chassis Initialization

1. MM sends *RS485 Discovery Packet* to inoperable Processor Blade.

2. The inoperable Processor Blade responds with the RS485 Discovery Response Packet.

3. The MM sends the *Get Device ID* command to the inoperable Processor Blade.

4. The inoperable Processor Blade responds with the *Get Device ID Response* message with the Device Available' bit indicating the BMC is in the device firmware mode.

5. The MM via step 4 has determined that the Processor Blade is in the inoperable state (running out of Boot FW and cannot load/execute Operational FW) and will send the OEM *Get Blade Descriptor* to get the minimum information required for Processor Blade identification, which includes the present build name of the operational firmware last installed on the target Processor Blade.

6. The Processor Blade will respond with the desired information in the *Get Blade Descriptor Response* message. The MM will now conclude the Processor Blade Chassis Initialization and flag the Processor Blade as not available.

7. MM sends the mandatory IPMI command *Get Self Test Results* to obtain further information as to why Processor Blade is in a fault or inoperable state so that information may be logged in the MM Chassis Event Log and perform any required alerting.

8. The Processor Blade returns the *Get Self Test Results Response*, as described in IPMI Architecture Specifications.

For Processor Blades that have been discovered in the non-available or inoperable state (i.e. Operational FW can not be executed) will never receive power permission. Users will be flagged to the Processor Blade condition so that action may be taken to bring the Processor Blade to an operational state.

5.6 MM/Blade State Synchronization

The MM obtains information from each Processor Blade for overall chassis management. This information stored in the MM must remain in synch with the current state of the Processor Blade. A BladeServer IPMI OEM command known as the *Blade State Change* must be used by the Processor Blade to keep state information from the Processor Blade to the MM in synch. Blade state data is defined as:

1. The Processor Blade operational state (see Blade State Change message format in BladeServer IPMI OEM Formats section).
2. The data tracked by the MM for that particular Processor Blade.

For Processor Blade operational changes (BMC operational, power cycle, etc.), an unsolicited IPMI *Blade State Change* is sent to the MM to inform it of the operational state change of the Processor Blade. This same technique is used to inform the MM if blade information obtained by the MM has been modified (refer to Blade State Message format for defined areas of information).

When the blade alters the state of a facility such as VPD the BMC is required to issue a Blade State Change message to the MM informing the MM that VPD has been altered. The implementation should attempt to minimize the corresponding number of BSC messages on the RS-485 interface. The Blade State Change message also informs the MM that the VPD write operation is complete. For example, when a host application writes an N-byte field(s) to VPD the actual write to VPD may be broken into multiple transactions. The BMC may need the application to inform it when the entire transaction is complete in order to issue a single Blade State Change message to the MM.

The MM obtains this data from the Processor Blade using IPMI OEM commands which have been previously described in section 5.5 *Processor Blade Chassis Initialization.*

When any of the data obtained from the blade is modified from an entity other than the MM via in-band internal system interface applications, BMC, hardware or BIOS (such as a hardware or firmware failure) then the BMC must send an IPMI *Blade State Change* message to the MM. The *Blade State Change* message does not contain the changed data itself, but only informs the MM that something has changed. The MM will then perform the necessary operations to collect the newly modified data from the Processor Blade.

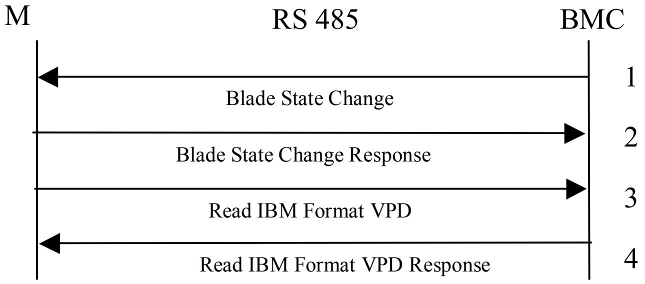


Figure 5-6 Blade State Change Example

Note in the example above that a VPD change to Block 0, Fixed Block Manufacturing Data was reported. This should only be done when the data actually change in the VPD. If the VPD was written to and the data was identical as that which was there previously, this would not be a data change and no *Blade State Change* message would be sent.

The set of data that must be kept synchronized by the Processor Blade is defined in the *Blade State Change* Message format, which can be found in section 5.19.3 OEM Message Formats.

5.7 Power Management

Processor Blades obtain their power from chassis power supplies. These power supplies are a shared resource within the chassis which is managed by the MM. Processor Blades must conform to the power requirements listed in this section.

This document uses the terminology of "Power Domain" to refer to power boundaries within the Processor Blade that are controlled by the MM. Power Domain 1 (PD1) is normally referred to by IPMI standards as "standby" power and in this specification as Auxiliary Power. For example, when the blade is in PD1 the BMC must be active for systems management purposes. In addition perhaps other peripherals, such as networking ports may also be active to allow for handling 'Wake On LAN' packets. Power Domain 2 (PD2) refers to the second power domain used by the Processor Blade when the blade is in a fully powered on functional state (i.e. CPUs, Memory, I/O etc).

5.7.1 Chassis Power Permissions

BladeServer Processor Blades must not allow themselves to be powered up, via local power control (button on front panel), external commands or any other means, until the MM grants 'power permission'. The Processor Blade will receive a power permission value from the MM soon after the blade is plugged into the chassis or AC power to the chassis is applied. The blade must follow the power permissions setting regardless of any other state or policy on the blade. The Processor Blade must only allow the MM to affect the power permissions. The blade will not save the Power Permissions value in non-volatile storage across PD1 power cycling. This behavior allows blades to be plugged into another chassis with different power permissions.

As mentioned above, power is a shared chassis resource and power permissions is the mechanism by which the MM controls that resource. Power Permissions are determined during the Processor Blade Chassis Initialization, as defined in section 5.5, or when a blade is installed into the chassis The MM will obtain information during the pre-init phase and determine if the power permissions is to be granted. The MM will send an IPMI OEM *Set Power Permission* message to the Processor Blade informing it of the current power permission state. Once permission is granted, the Processor Blade is enabled to turn on Power Domain 2. If the appropriate local power control permissions are granted by the MM (permission bits set to "enabled"), local power commands on the blade, such as from a front panel power button on the Processor Blade or WOL can be accepted. It is critical that the Processor Blade not allow power on capability unless permitted by the MM. Only the MM is allowed to change power permissions using the IPMI OEM *Set Power Permission* message. The command must be blocked or otherwise not take effect if issued by any other SMS.

In-band signaling via IPMI commands to turn-off power is allowed only when Local Power Control is enabled.

There are multiple issues to take into account when the Processor Blade makes a determination whether to grant the execution of any power request, which are:

1. Power Permission states (Power-On Permission, Wake-On-LAN, Local Power Control) which are granted only by MM.

2. Processor Blade reaction to magic packets for Wake-on-LAN and other local power controls, such as front panel power buttons.

For example, a Processor Blade power on command has been received on the internal RS485 interface (in-band). The following process must be complied with (local power control will be covered in the next section).

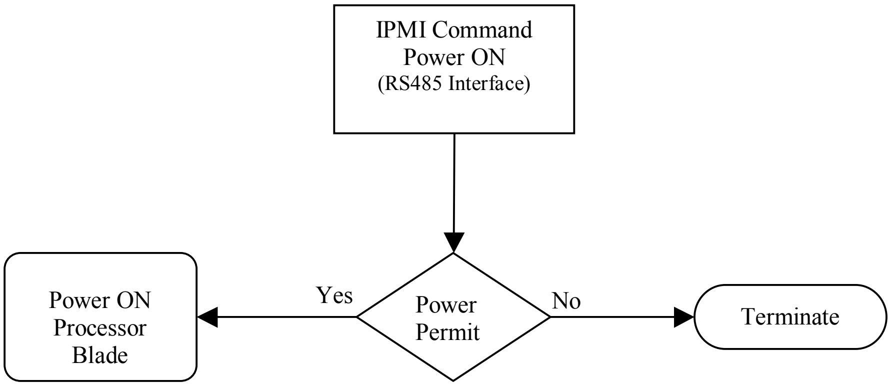


Figure 5-7 Power Permission Process

Figure 5.7 shows the blade receiving IPMI standard commands to power up the system (Power Domain 2). Shown is the command being received on the side-band (RS485). If power permission has not been granted, all power commands, regardless of interface, are ignored. If power permission has been granted, the PD2 state change request may be honored.

Please refer to section 5.19.3 OEM Message Formats for the *Get Power Permission* and *Set Power Permission* IPMI formats.

5.7.2 Local Power Control

Local power control granted by the MM references the Processor Blade's own power control, enabling or disabling local power mechanisms such as the front panel power button and Wake-On-LAN. Providing such control is usually to allow users to disable local power control so that the Administrator can determine if personnel with physical access to the chassis have the authority to control local power to a Processor Blade.

Local power control must be granted by the MM before any other methods are used to set local power settings. The Processor Blade may allow any entity the ability to disable or enable local power control if the power permission does not block the local power control command.

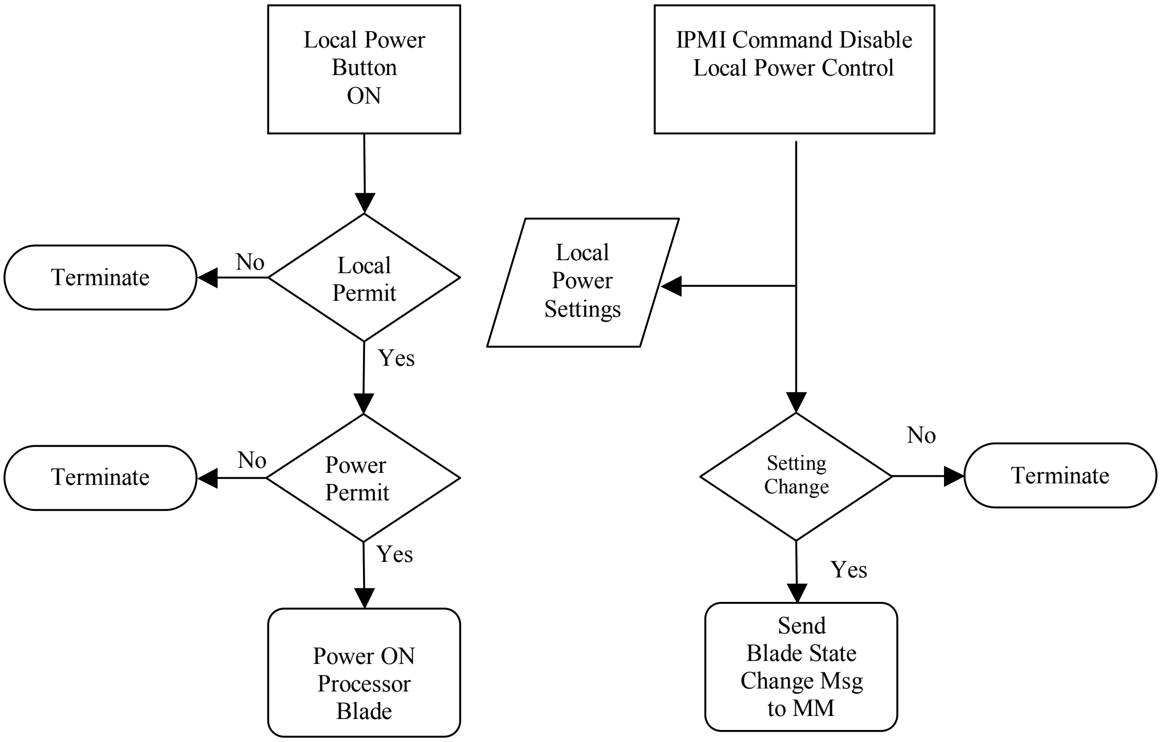


Figure 5-8 Local Power Control Process

Figure 5.8 presents two logical flows. The first flow on the left demonstrates the operation of a front panel power on request. Both the Local Permit and Power Permit states are validated before taking action on the request. The second flow on the right demonstrates an IPMI command being received to disable some local power state, such as disabling or enabling Wake on LAN (WOL). The current state is checked and if the state changes, the blade is required to send a Blade State Change (BSC) message to the MM to inform it of the local power state change.

Power Permissions are required to allow the Local Power Button to power on PD2 of a Processor Blade. When the local power button is depressed and a previous *'Set Power Permissions'* command has been issued with Local Power Control(0:0)=enable and Power On Permissions(1:1)= Allow then PD2 can be powered on.

Even with local power control enabled, one must still follow general power permissions controlled by the MM. If a control like the power button is depressed with local power control granted and system power is desired, it would still be blocked if general power permission was not granted by the MM.

Other power control capabilities, such as Wake on LAN, behave in the same manner as the local power control.

5.7.3 Power and Power Permission Defaults

When a processor blade is initially inserted in the chassis or AC power is applied to the chassis the following are the default values until changed by the MM.

- Power State is off
- Local Power Enable is set to disabled
- Power-on Permission is set to disabled
- WOL is disabled

5.7.4 Power Control Initiators and Behaviors

Table 5-2 - Power Control Initiators and Behaviors

Initiator	External Signal Name or Internal Subsystem	Local Power Control Permission	Power Permission	Processor Blade Behavior
Blade Front Panel Power Button pressed	FP Power button	Must be enabled	Must be enabled	BMC turns power ON or OFF
Message initiated by MM to change power state	BMC handles message	May or may not be enabled	Must be enabled	BMC turns power ON/OFF or power cycle
Message source other than MM to turn on PD2	BMC handles message	Must be enabled	Must be enabled	BMC turns power ON. BMC must prevent a power on state change other than from MM
Message source other than MM to turn off PD2	BMC handles message	Must be enabled	Must be enabled	BMC turns power off
Chipset	sleep signal	Must be enabled	Must be enabled	Turns power ON or OFF
12v PS Redundancy is Lost	12v PS is non-redundant	May or may not be enabled	Must be enabled	Blade must throttle to pre-set level if PD2 is on

Return of 12v PS Redundancy after throttling has occured	12v PS is Redundant	Must be enabled	Must be enabled	Blade stays at pre-set throttle level until MM issues command to un-throttle or change the throttle value
Processor Blade temperature reaches upper critical threshold	For example: (a)Thermal sensor (b)CPU_x Thermal Threshold	May or may not be enabled	Must be enabled	BMC turns power OFF
Voltage Sensors	For Example: Processor voltage regulator or overcurrent detecetion	May or may not be enabled	Must be enabled	BMC turns power OFF
CPU IERR	All Processor(s) have asserted IERR	May or may not be enabled	Must be enabled	BMC turns power OFF
Notes: (1)The Processor Blade will always default to a power-off state immediately when standby power returns after AC power is restored to the chassis. The Processor Blade must not perform power state retention.				

5.7.5 Power Restore Policy

The BladeServer policy for the case of power restoration after chassis power AC loss is that all the Processor Blades will be returned to their previous power state. The IPMI standard command *"Get Chassis Status"* is used by the MM to obtain the Processor Blade current power state. This command includes a field for the setting of the Processor Blade's "power restore policy" such that it is possible to set policy to handle AC power interruptions. This setting is ignored by the MM due to the enforcement of the general BladeServer chassis policy for power restoration.

5.7.6 Power Throttle & Cooling Management

Power throttling is optional for blades that consume 250w or less of power in BladeServer and 350W or less of power in BladeServerH. Power throttling is mandatory for blades that consume greater than these limits. For blades that implement power throttling the following is required.

The intent of this section is to describe the Power/Thermal requirements that a ServerBlade must implement. Power and thermal loads in Blade Center are managed as a shared resource in the chassis by the MM. ServerBlades must interact with the MM as describe below.

Since processor(s) on the ServerBlade are by far the biggest contributor of the power/thermal load in a ServerBlade the following text describes power/thermal throttling with a focus on throttling the processor(s) in order to reduce the power/thermal load placed on the chassis by the blade. Implementations are free to throttle processor(s) and/or other components using various temperature sensing mechanisms such as a thermal diode connected to the processor to reduce the thermal load the blade has upon the chassis. In addition, the BMC is one of many mechanisms by which control of processor(s) may be initiated.

Due to various reasons, the MM may request the ServerBlade to enter into a mode where the BMC may have to initiate throttling for thermal reasons. The IPMI OEM command 'Set Thermal Throttle Enable' is issued by the MM to the BMC to enable or disable this mode on the ServerBlade.

When this mode is enabled on the ServerBlade throttling of the processor(s) must be performed when the thermal diode of the processor(s) exceeds the processor Tcontrol (Tc) threshold value. When the value of Tc is exceeded the BMC must initiate throttling in an attempt to prevent the processor(s) from exceeding Tc. Since throttling is affecting the processor performance a stepped approach where by the amount of throttling will be increased if Tc continues to be exceeded is recommended. Once the measured value of the thermal diode value falls below the Tc threshold, the BMC will un-throttle the processor(s). An appropriate amount of hysteresis is required to insure that the processor(s) will not repeatedly throttle and un-throttle.

When this mode is disabled on the ServerBlade no BMC invoked throttling for thermal reasons is required. The BMC is required to notify the MM when the value of Tc has been exceeded. The IPMI OEM message 'Blade State Change' is issued by the BMC to indicate that the BMC has determined that the processors(s) have exceeded Tc. See Section 5.19 BSC message 'Action Taken' byte for the appropriate values. The MM may increase the blower(s) speed when this occurs. Note: the actual threshold value for Tc may vary by design of the ServerBlade.

Additionally, independent of whether the BMC has been enabled or disabled via the 'Set Thermal Throttle Enable' IPMI OEM command the BMC is required to notify the MM when the processors(s) are throttling and un-throttling. The IPMI OEM message 'Blade State Change' is issued by the BMC to indicate that the BMC has taken a action to throttled or un-throttled the processor(s), see Section 5.19 BSC message 'Action Taken' byte for the appropriate values.

When a Power Module failure is detected by the blade then the blade must reduce its power consumption to the value set in the OEM IPMI Set Pre-Set Watts command (see Table 5.14) within 1 second of detecting the failure. The blade must detect loss of redundant input power when either of the two conditions below occur:

1. EPOW_N – an early warning indication given prior to the loss of a Power Module when AC power is removed. This signal may go active during power line disturbances (PLD) allowed per the "no interruption in function region" as defined in the ITI (CBEMA) Curve (http://www.itic.org). The BladeServer chassis will also indicate an EPOW_N to the blade if +12Vdc goes out of regulation tolerance for any condition (power supply or system fault).
2. A measured difference between either of the input voltage sources to the ORed voltage value that is greater than or equal to -0.25 volts.

In order for the MM to handle power management of the chassis, the ServerBlade must implement the following behaviors:

1. The ServerBlade hardware must be enabled to detect a drop-off in the voltage at hardware speeds (see Section 2.11) on the redundant 12V supplies to the ServerBlade. This could be for example due to a loss of AC power. When this condition is detected by the BMC, it may be required to throttle the processor(s) to a value that is has been pre-set by the MM. The command used to pre-set the amount of throttle for this conditions is the OEM command *Set Pre-Set Watts.*
2. The BMC must inventory components (processors, DIMMs etc) on the ServerBlade in order to report power consumption information when requested by the MM. See *Get Watt Usage* OEM Command for additional details. The BMC must be able to report to the MM when the processors(s) have throttled or un-throttled. See OEM command *Blade State Change* with appropriate 'Action Taken' field for additional details.
3. When the ServerBlade is going through a power-on sequence and booting, the BMC must report to the MM that it now has a more accurate power consumption value for the

installed components. This is accomplished using the OEM command *Blade State Change* with the appropriate 'Action Taken' field indicating that the wattage value has been updated. This could be for reasons that a ServerBlade does not have access to all components prior to the blade being powered on. The MM will subsequently issue the OEM command *Get Watt Usage* to determine the updated watt usage(post power-on inventory) for the ServerBlade. The MM will then issue OEM command *Set Pre-Set Watts* to indicate to the BMC that the ServerBlade has been granted or denied permission to continue the boot process. In addition an indication of the power value for the pre-set amount of throttling will be sent.

4. The MM may issue to the BMC at any time a new throttle value. The BMC will need to handle this and immediately throttle/un-throttle to this value. In addition the BMC must set the value passed in the 'pre-set watts' command incase power redundancy is lost. The command used to set this value is the OEM command *Set Pre-Set Watts.*
5. If the BMC returns an IPMI command 'Completion Code' of C0h – 'BMC is Busy' for the *Get Watts Usage* command; the MM will resend the *Get Watts Usage* command after a small delay. The operation will continue until the BMC returns a Completion Code other than C0h – 'BMC is Busy' or the MM reaches a predefined number of retries, which ever occurs first. If a completion code other than 00h – 'Command Completed Normally', C0h – 'BMC is Busy' is received or the MM reaches its command retry limit, the MM will treat this as an error case and power permission will be denied.

When the ServerBlade is inserted into the chassis or the chassis is A/C cycled, the BMC must not allow the ServerBlade to be powered up, via local power control (button on front panel), external commands or any other means, until the MM grants power permission. The BMC should also perform an inventory of all the components that reside on the ServerBlade and determine the power usage of these components. The components include but are not limited to Processor(s), Memory DIMM(s), Hard Drives and I/O Expansion Cards. The BMC must sum the power values of the components and provide this value to the MM. This value is referred to as 'pre-boot inventory power allocation value' (prior to the ServerBlade being granted permission to power on) and will allow the MM to determine, due to power limitations, if power permission will be granted to the ServerBlade. If the BMC cannot determine the power consumption of these components then the BMC may use the maximum watts for the Serverblade that can be found in the physical characteristics field of the VPD. This allows for a ServerBlade Design in which completely accurate value cannot be determined at the pre-boot time prior to the ServerBlade being powered on.

Once power permission is granted to the BMC and the ServerBlade is subsequently powered on but before the boot process is started the BMC will send a message to the MM indicating that a updated 'maximum watts' is available. This implies that the BMC has a 'post power-on inventory value' available for the MM which may be a more accurate power consumption value due to information gathered when PD2 is active. The ServerBlade must hold off completely finishing the boot process until the MM has indicated it has granted permission to do so. Once this permission is granted then the ServerBlade can complete the boot process and consume the 'post power-on inventory' value of power.

A summary of the message exchange and flow when a ServerBlade is inserted into the chassis is shown in Figure 5.9.

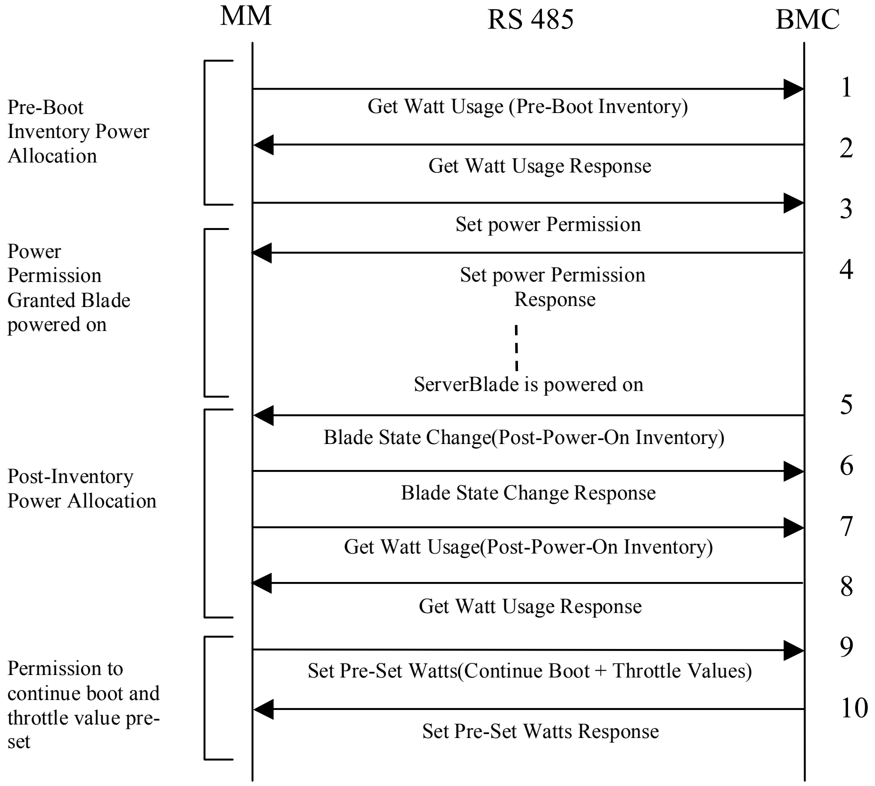

MM
RS 485
BMC
Pre-Boot Inventory Power Allocation
Power Permission Granted Blade powered on
Post-Inventory Power Allocation
Permission to continue boot and throttle value pre-set
1
Get Watt Usage (Pre-Boot Inventory)
2
Get Watt Usage Response
3
Set power Permission
4
Set power Permission Response
ServerBlade is powered on
5
Blade State Change(Post-Power-On Inventory)
6
Blade State Change Response
7
Get Watt Usage(Post-Power-On Inventory)
8
Get Watt Usage Response
9
Set Pre-Set Watts(Continue Boot + Throttle Values)
10
Set Pre-Set Watts Response

Figure 5-9 Power Management Blade Initialization

1. The MM sends the *Get Watts Usage* command to determine the pre-boot power inventory information from the BMC.

2. The BMC will respond with the *Get Watts Usage Response* message, which includes information on the pre-boot power required by the ServerBlade.

3. The MM determines if the blade can power on and sends the *Set Power Permission* that grants the BMC power permission.

4. The BMC responds to the the *Set Power Permission*

5. Subsequently the ServerBlade is powered on by the user. The BMC turns on PD2 and begins a power on sequence and starts to boot(run POST). During POST the BIOS and BMC will exchange information in order to allow the BMC to report (even if the pre/post values do not change) the post-power-on power consumption values (see section 5.7.5 for additional information). A *Blade State Change* will be issued to notify the MM that the BMC has the updated the current power consumption values.

6. The MM sends a response to the *Blade State Change.*

7. The MM sends the *Get Watts Usage* command to determine the post power-on inventory information from the BMC.

8. The BMC will respond with the *Get Watts Usage Response* message, which includes information on the post power-on inventory information.

9. The MM sends a *Set Pre-set Watts message to the BMC* to indicate to the BMC to continue the boot process and what throttle value needs to be used in case of loss of redundant power to the ServerBlade.

10. *Set Pre-set Watts message* response.

5.7.7 Over-Temperature Management

The intent of this section of the specification is to specify consistent Over-Temperature behavior between Processor Blades and the Management Module (MM) when an over-temperature condition occurs. All blades shall report Over-Temperature conditions and related behaviors as defined below. All temperature sensing logic must be in PD1 and controlled by the BMC.

The Processor Blade shall protect itself from damage by removing power to PD2 when the *Shutdown* over-temperature condition occurs. Associated with this behavior comes the need for the Processor Blade to inform the MM of the state of PD2 by issuing the OEM IPMI Blade State Change Message:Power Down*,* which will indicate the current state of PD2 as powered off.

Two temperature thresholds from any number of sensors are required in all Processor Blades representing *Warning* and *Shutdown* conditions. When any sensor reaches its threshold value the Processor Blade shall issue the IPMI OEM Directed Event Message (DEM) which contains the SEL associated to the threshold event. A Processor Blade vendor may have any number of sensors located near critical components throughout the Processor Blade. The Processor Blade shall analyze each sensor and either reports the Warning temperature or the Critical Shutdown temperature.

The presence and/or failure state of the MM cannot be guaranteed. Therefore, the Processor Blade shall protect itself from running outside of its operating range by turning off PD2 when the Shutdown temperature is reached.

5.7.7.1 Over-Temperature Warning Sensor & Reporting Logic

The first lower temperature threshold (*Warning* temperature) is set to a trip point that is within the operating conditions of the Processor Blade as dictated by the chassis thermal specification (see section 3) and is considered to be the lower threshold of the maximum operating range by the Processor Blade vendor. The trip-point value of the sensor is selected by the Processor Blade vendor.

The Processor Blade must issue a threshold event (via SEL contained in the DEM) to the MM to report the *Warning* temperature status when the Processor Blade temperature meets or exceeds the first thermal threshold. To properly signal the MM of the *Warning* temperature status the SEL must contain the Sensor Type = 01h, temperature and the Generic Event/Reading Type Code = 01h - Warning Temperature, Lower Non-Critical – Going High. When the MM detects the *Warning* temperature status the MM may immediately ramp up the speed of the fans in an attempt to reduce the temperature of the Processor Blade.

Subsequently the Processor Blade must issue a threshold event to the MM to report the *Warning* temperature status when the Processor Blade temperature falls below the first thermal threshold. To properly signal the MM of the *Warning* temperature status the event's SEL must contain the Sensor Type = 01h, temperature and the Generic Event/Reading Type Code = 00h - Warning Temperature, Lower Non-Critical – Going Low. When the MM detects the *Warning* temperature status the MM may ramp down the speed of the fans.

Note that hysteresis must be provided between the set and reset temperature points such that DEM messages are not issued when the temperature of the Processor Blade is close to the *Warning* threshold.

5.7.7.2 Over-Temperature Shutdown Sensor & Reporting Logic

The *Shutdown* temperature threshold sensor is set to a trip point greater than the *Warning* threshold. This protects the Processor Blade from running outside of its maximum reliable operating temperature and from possibly destroying components within the Processor Blade. The Processor Blade shall protect itself from damage by turning off power domain 2 (PD2) when the *Shutdown* temperature condition is detected. The trip-point value of the sensor is selected by the Processor Blade vendor with respect to the chassis thermal specification. Shutting down PD2 will immediately start the cooling process on the Processor Blade.

When the Processor Blade temperature meets or exceeds the Shutdown threshold then the Processor Blade issues a threshold event (SEL contained in a DEM) to the MM to report the *Shutdown* temperature status. To properly signal the MM of the *Shutdown* temperature status the SEL must contain the Sensor Type = 01h, temperature and the Generic Event/Reading Type Code = 0Bh - Critical Temperature, Upper Non-Recoverable – Going High. When the MM detects the *Shutdown* temperature status the MM may immediately ramp up the speed of the fans in an attempt to reduce the temperature of the Processor Blade. In addition the Processor Blade will inform the MM of the state of PD2 by issuing the OEM IPMI Blade State Change Message: Power Down*,* which will indicate the current state of PD2 as powered off.

The Processor Blade issues a threshold event to the MM to report the *Shutdown* temperature status when the Processor Blade temperature falls below the *Shutdown* threshold. To properly signal the MM of the *Shutdown* temperature status the SEL must contain the Sensor Type = 01h, temperature and the Generic Event/Reading Type Code = 0Ah - Critical Temperature, Upper Non-Recoverable – Going Low.

If the administrator uses the MM or another mechanism to turn on PD2 after the Processor Blade is shutdown for an Over-Temperature condition, then the Processor Blade may require some amount of time to acquire the current temperature. Between the time that the Processor Blade powers on and the temperature of the Processor Blade is determined, the Processor Blade shall run in a "low heat generation" mode. This is to prevent the Processor Blade from reaching the *Shutdown* threshold and running above its shutdown threshold. In other words, power cycling the Processor Blade is not a guaranteed method of removing the Shutdown condition and the Processor Blade is responsible to ensure that the module does not run above its maximum reliable operating temperature.

In the event that the administrator uses the MM or another mechanism to attempt a power on of the blade while any over temperature conditions exist then the Processor Blade shall not turn power back on to the Processor Blade and will continue to report PD2 being off.

5.8 Logging and Alerting

An important operation for managed systems is the process of logging system events and errors for problem identification, predictive failure analysis, resource usage, and failure history. Also important is the ability to alert system administrators of these events so that action may be taken after failures occur or, in the case of PFA's (Predictive Failure Analysis), action can be taken prior to the loss of a resource.

The BladeServer system consists of multiple Processor Blades, where each Processor Blade and MM may have its own logging and alerting mechanism. The MM serves as a central collection for Event Log events, a single location in which a system administrator may observe events impacting the chassis as well as all the Processor Blades located in that chassis. The MM logging and an alerting mechanism to external management applications provides ease of use by providing a single administrative location which allows correlation of events that affect any number of components in BladeServer (blades, MM, power supplies etc).

5.8.1 Event Logs

The BladeServer logging is performed on each Processor Blade and a centralized log is located on the MM. This centralized log on the MM is called a Chassis Event Log (CEL) and contains chassis events and events relating to each Processor Blade.

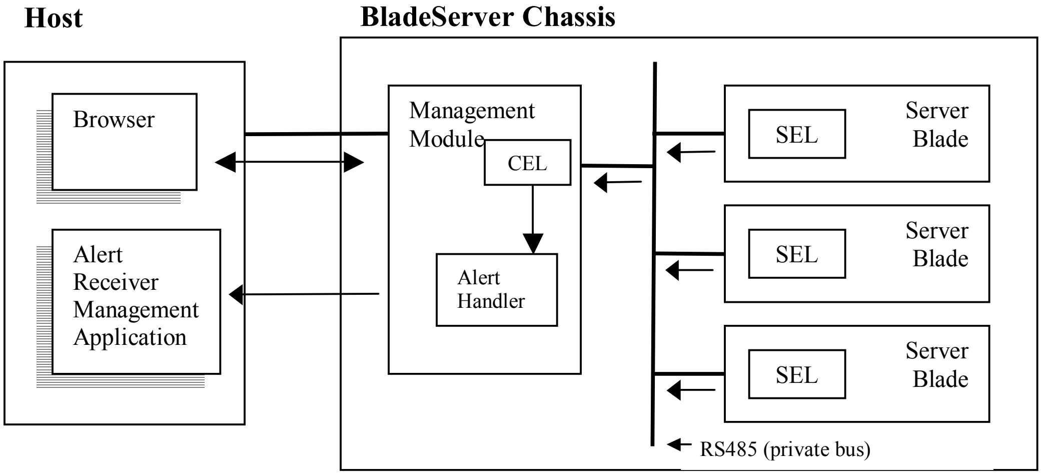


Figure 5-10 Event Logs

Figure 5.10 shows the existence of event logs on the MM and all the Processor Blades. Each BMC collects events from its Processor Blade and places those events into a System Event Log (SEL). The event is also sent by the BMC to the MM by way of the IPMI OEM command *Directed Event Message* (DEM). The DEM includes the SEL event, in same format as that received by the Processor Blades BMC. The MM will parse the received SEL to make an entry into the Chassis Event Log (CEL). All unrecognized SEL events received will be logged into the MM Chassis Event Log with an “unknown event” description. All Chassis Event Log entries created due to events received from IMI Processor Blades, regardless if SEL data is recognized or not, will include the received SEL data as part of the MM Chassis Event Log string entry. In addition Figure 5.10 depicts the MM not only having a CEL but also providing the ability to send Processor Blade events that are added to the MM’s CEL as alerts to an external management application. Alerting the end user about Processor Blade events is only possible via the MM. The MM has a configurable alerting mechanism that uses SNMP Traps, email, etc. to send to the appropriate external alert receiver management application.

The MM is not the owner or manager of the System Event Log located on each Processor Blade. It is assumed each Processor Blade has its own system management software to manage the log. If not, the SEL log may fill up and take no further entries. However, even if this situation does occur, the Processor Blade is still required to send DEM’s containing any further SEL events received, regardless if they can be placed in the Processor Blade’s System Event Log or not.

The CEL in the MM is a limited resource of finite size. Care must be taken that the MM log buffer is not overrun.

5.8.1.1 Directed Event Strings

As mentioned previously the *Directed Event Message* (DEM) purpose is to send blade events to the MM for logging the SEL event data into the CEL and sending alerts to external applications. However in some unique cases there may not be an IPMI standard SEL event that is appropriate to signal a particular condition to the MM. In lieu of sending a DEM the OEM IPMI *Directed Event String (DES)* should be used. The DES may be used for support of additional functions such as BIOS detected errors, BIOS POST progress, and Prevent Failure Analysis (PFA) errors.

The DES is used by the BMC to send a host generated (BIOS or Diagnostic) ASCII string to the MM for centralized logging into the CEL. The command contains support to handle sequenced commands when the string to be sent exceeds a single 80 character ASCII message.

Although DES support is provided by the MM to log the event to the MM’s CEL the ability to send event data as an alert to an external management application is not provided when using a DES. Therefore using standard SEL events is the more robust, provides better functionality and is the preferred approach.

The first byte of the data portion of the OEM IPMI DES command is known as a sub-command byte and indicates how the remaining data portion is to be handled. Contained in the sub-command are bits indicating Op-code, which informs the MM how to parse and handle the data portion of the DES command.

Op-Code	Function
0x90	Chassis Event Log String
0x80	PFA Information
0x81	BIOS Progress Alert

Please refer to section 5.19.3 OEM Message Formats for the format data of the DES command.

5.8.1.1.1 Log Strings

A DES may be sent to enter a log entry into the MM Chassis Event Log (CEL). The command data format is as follows:

```
byte[0] = sub-command byte = (Op-code | Prefix | Severity)
byte[1] = string char 1 , byte[2] = string char 2 , ....
```

Op-code:

90h =	Chassis Event Log String

Prefix:

00h =	POSTBIOS
04h =	SMI Handler
08h =	DIAGS
0Ch =	Other

Severity:

00h =	Informational
01h =	Warning
02h =	Error

The prefix and severity control how the string will be displayed when viewing the MM CEL.

5.8.1.1.2 Predictive Failure Analysis (PFA) Data

PFA data may be sent to the MM using the DES command. This data is then used by the MM for logging and alerting action.

```
byte[0] = sub-command byte = Op-code , byte[1] = PFA Data Byte
```

Op-code:

80h =	PFA Information

PFA Data Byte:

00h =	Excessive Single Bit Error Correction Code (ECC) Errors (Threshold should be set by BIOS setup option)
01h =	Multiple Bit ECC Error
02h =	Excessive (>96) Recoverable Memory Channel Architecture (MCA) Occurred
03h =	Unrecoverable MCA Occurred
04h =	PCI Parity Error (PERR) occurred
05h =	PCI System Error (SERR) occurred

5.8.1.1.3 BIOS State Progress

The BIOS supplies the MM with a running status of boot progress. This information is used by the MM to maintain boot state of the blade and maintain boot statistics.

```
byte[0] = sub-command byte = Op-code , byte[1] = Boot State Alert
```

<u>Op-code</u>:

81h = BIOS Progress Alert

<u>Boot State Alert</u>:

01h =	"System in POST"
02h =	"System stopped in POST (Error Detected)"
03h =	"Booted Flash or System partition"
04h =	"Booting OS or in unsupported OS"
05h =	"OS booted"
06h =	Lock MM from reading VPD
07h =	Reserved

5.8.1.2 Logging During Processor Blade Chassis Initialization

The MM will not process any DEM message until a *Set Directed Event Message* command (enable) is sent to the Processor Blade during chassis initialization. In addition to insure that any events detected by the BMC prior to the MM sending the *Set Directed Event Message* command, the MM will inform the BMC to regenerate DEM's for any current out-of-spec conditions by sending the IPMI Run Initialization Agent command. IPMI defines the *Run Initialization Agent* command as optional, but it is mandatory for all BladeServer Processor Blades.

Note: Any DEM that is sent before the Processor Blade Chassis Initialization sends the *Set Directed Event Message* will be ignored by the MM.

5.9 Session Based Interfaces

Session Based Interfaces are those defined by the IPMI standards as being managed through sessions. These include LAN interfaces (i.e. Ethernet), serial (i.e. modem), etc. Session based interfaces allow out-of-band access to the BMC.

Due to concerns such as security, all systems must initialize with all session based interfaces disabled. The only IPMI LAN session based interface used by Processor Blades must be for the purpose of SOL and therefore the BMC shall only accept session activation from the MM.

5.10 KVM and Media Tray Resource Sharing

The BladeServer provides shared resource for the Keyboard, Video, and Mouse (KVM), as well as a Media Tray (with integrated USB devices) which allows Processor Blades to access a chassis Compact Disc (CD) drive or Digital Video Disk (DVD) drive and a Floppy Disk Drive (FDD). This section only discusses the IPMI methods (via IPMI messages) for the Processor Blade to negotiate with the MM for KVM and Media Tray resources.

Since this resource is shared among many Processor Blade systems, a coordinated method must be defined so that Processor Blades may request and be assigned these resources.

A set of IPMI OEM commands (Request Access to KVM-MT, Set Blade KVM-MT Selection and Get Blade KVM-MT Selection) have been defined for Processor Blades to report support for KVM and MT sharing, request ownership of the KVM or MT, and to release the shared KVM or MT resource. This command set is based on a request and grant mechanism. Processor Blades must never use KVM-MT resources until the MM grants those resources to the Processor Blades. The blade must wait for KVM-MT permission from the MM before allowing such resources to be used by the Processor Blade.

The MM may be blocking the switching of KVM-MT due to user's locking KVM and MT resource switching and the Processor Blade may never receive KVM-MT ownership permission until such locking is removed.

5.10.1 Obtaining KVM-MT Resources

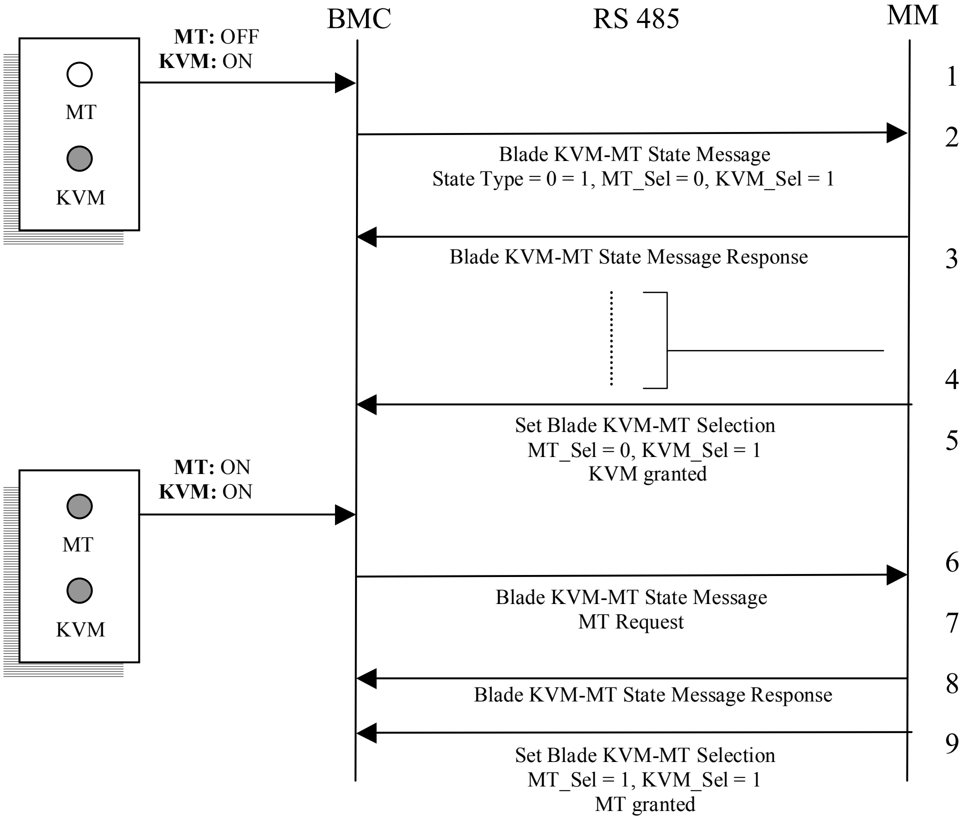

BMC
RS 485
MM
MT: OFF
KVM: ON
MT
KVM
1
2
Blade KVM-MT State Message
State Type = 0 = 1, MT_Sel = 0, KVM_Sel = 1
Blade KVM-MT State Message Response
3
4
Set Blade KVM-MT Selection
MT_Sel = 0, KVM_Sel = 1
KVM granted
5
MT: ON
KVM: ON
MT
KVM
6
Blade KVM-MT State Message
MT Request
7
Blade KVM-MT State Message Response
8
9
Set Blade KVM-MT Selection
MT_Sel = 1, KVM_Sel = 1
MT granted

Figure 5-11 Obtaining KVM-MT Resources

Figure 5.12 shows the message flow process for requesting and getting the KVM-MT resources. Resources are not available until the MM informs the Processor Blade that the KVM-MT resources have been assigned to the Processor Blade.

1. The Keyboard, Mouse, and Video button is depressed on the Processor Blade front panel. The MT button is not depressed.

2. The Processor Blade sends the IPMI OEM *Blade KVM-MT State Message* with the State Type bit set to 0b for request action and the MT – KVM selection bits set to MT_Sel = 0b, KVM_Sel = 1b.

3. The MM responds that the message was received and request is known.

4. There may be a delay when requesting KVM-MT resources. A current owning Processor Blade may not release the KVM-MT resources immediately. Switching KVM-MT owners may be delayed up to eight seconds, at which time the requestor may receive a *Set Blade KVM-MT Selection* and be given ownership. In addition, the KVM-MT selection may also be locked to a particular Processor Blade due to system administrator settings. If this is the case, the requesting Processor Blade will not receive the *Set Blade KVM-MT Selection The Processor Blade* must never consider itself the owner of the

KVM-MT without the proper indication from the MM.

5. After the MM has taken the appropriate actions to inform the blade to switch the shared KVM hardware to the requesting Processor Blade and in the case when the KVM is not locked, the MM sends an IPMI OEM *Set Blade KVM-MT Selection* to the Processor Blade, with the MT – KVM selection bits set to MT_Sel = 0b, KVM_Sel = 1b. The Processor Blade is now free to use the KVM resources.

6. The Media Tray button is depressed on the Processor Blade front panel. The KVM button is not depressed.

7. The Processor Blade sends the IPMI OEM *Blade KVM-MT State Message* with the State Type bit set to 0b and the MT – KVM selection bits set to MT_Sel = 1b, KVM_Sel 1b. Note that the both selection bits are set to 1b. This informs the MM that both are still required, even though the Media Tray button only was depressed. Each time this message is sent, the selection bits must reflect the current state of the KVM-MT selection.

8. The MM responds that the message was received and request is known.

9. After the MM has taken the appropriate actions to inform the blade to switch the hardware to the requesting Processor Blade, the MM sends an IPMI OEM *Set Blade KVM-MT Selection* to the Processor Blade, with the MT – KVM selection bits set to MT_Sel = 1, KVM_Sel = 1b. The Processor Blade is now free to use both the KVM and MT resources.

Note that this example covers one resource at a time being requested. The Processor Blade may request resources at the same time if both are selected at the same time. The flow would be the same as above, selecting both KVM and MT with one request.

5.10.2 Releasing KVM-MT Resources

A Processor Blade which owns the KVM-MT resources will continue to own them until the resources are requested elsewhere. The Processor Blade does not release the resources via its front panel KVM-MT buttons which operate only as a requesting mechanism. When the KVM-MT buttons on the Processor Blade front panel are depressed and the KVM-MT resources are already owned by that same blade, then the action will be ignored by the blade and resources remained assigned.

KVM-MT resources are released by the Processor Blade due to other user requests for the KVM-MT shared resource. If this occurs, the MM will inform the owning Processor Blade that KVM-MT resources are being deselected. The Processor Blade will have up to eight seconds to release control and send an IPMI OEM *Blade KVM-MT State Message* with the newly compliant KVM-MT state and appropriate request bit set to ‘0’. If this message is not sent within the eight seconds, MM will provide permission to the new KVM-MT requestor regardless if previous owner sent the *Blade KVM-MT State Message* or not. This release method is used so that the current owning Processor Blade may release the resource gracefully by completing current operations. With this in mind, the owning Processor Blade should send the *Blade KVM-MT State Message* with the selection bits indicating the release of resources as soon as possible.

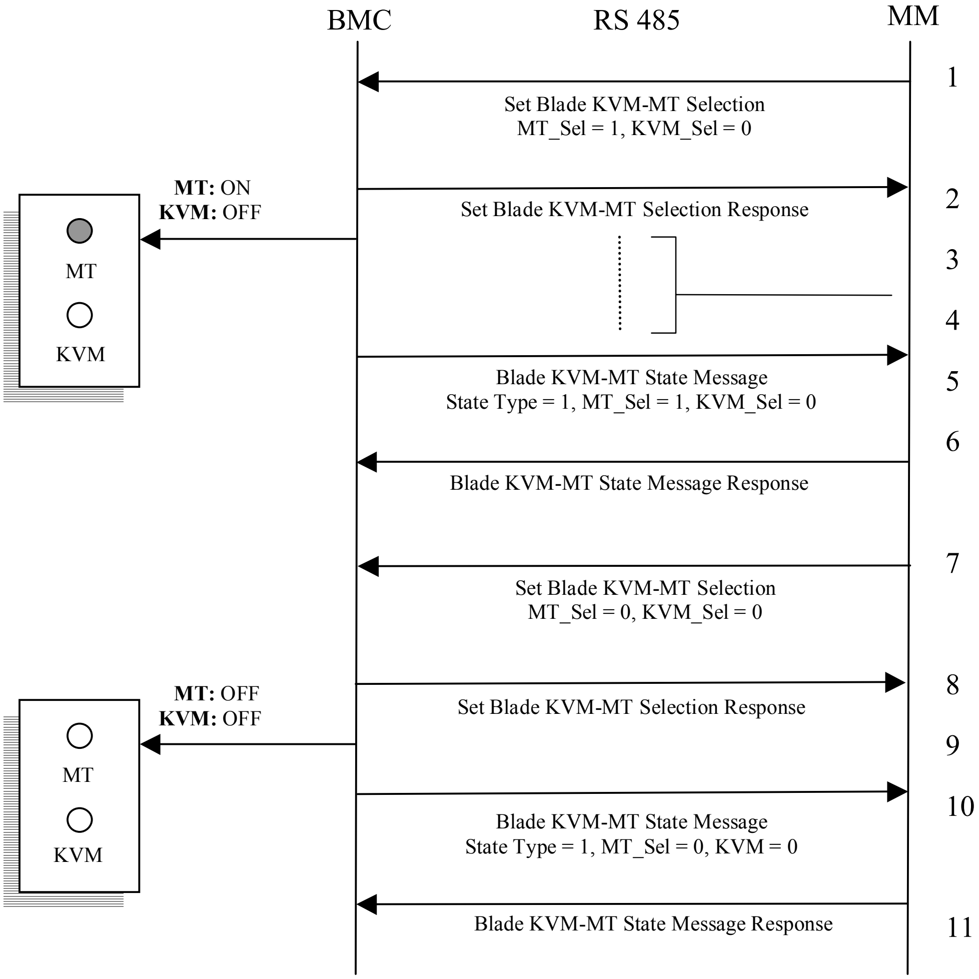

BMC
RS 485
MM
1
Set Blade KVM-MT Selection
MT_Sel = 1, KVM_Sel = 0
MT: ON
KVM: OFF
2
Set Blade KVM-MT Selection Response
MT
KVM
3
4
5
Blade KVM-MT State Message
State Type = 1, MT_Sel = 1, KVM_Sel = 0
6
Blade KVM-MT State Message Response
7
Set Blade KVM-MT Selection
MT_Sel = 0, KVM_Sel = 0
MT: OFF
KVM: OFF
8
Set Blade KVM-MT Selection Response
MT
KVM
9
10
Blade KVM-MT State Message
State Type = 1, MT_Sel = 0, KVM = 0
Blade KVM-MT State Message Response
11

Figure 5-12 Releasing KVM-MT Resources

1. The MM sends the *Set Blade KVM-MT Selection* to the Processor Blade which owns both the KVM and MT to inform it to unselect the KVM.

2. The KVM owning Processor Blade sends the response for the received *Set Blade KVM-MT Selection* message.

3. The Processor Blade turns off its front panel LED for KVM selection. The Processor Blade may wish to delay the operation for a small fraction of time to allow any current operations to finish. It may release the resource immediately as well, dependant on Processor Blade firmware implementation.

4. The MM will allow up to eight seconds for the Processor Blade to send it notification that the resource has been released before reassigning the KVM to the new owner. If the owning Processor Blade fails to notify the MM that it has released the KVM, the KVM will be granted to the new owner after this period of delay. Processor Blades should release the desired resource as soon as possible and send the release notification (*Blade KVM-MT State Message*) as soon as possible.

5. The Processor Blade sends the *Blade KVM-MT State Message* with the State Type bit set to 1b for a state change notification and the MT – KVM bits set to MT_Sel – 0b, KVM_Sel = 1b.

6. The MM responds to the *Blade KVM-MT Message* and immediately sends a *Set Blade KVM-MT Selection* (not shown) to the new owner. The MM will also take the appropriate hardware actions to switch the resources from the releasing Processor Blade.

7. The MM sends the *Set Blade KVM-MT Selection* to the Processor Blade which owns the MT only to inform it to unselect it.

8. The MT owning Processor Blade sends the response for the received *Set Blade KVM-MT Selection* message.

9. The Processor Blade turns off its front panel LED for MT selection. The Processor Blade may wish to delay the operation for a small fraction of time to allow any current operations to finish. It may release the resource immediately as well, dependant on Processor Blade firmware implementation.

10. Unselecting the MT as soon as possible, the Processor Blade sends the *Blade KVM-MT State Message* with its new KVM-MT state by setting the State Type bit to 1b for state change notification and the KVM – MT bits to MT_Sel = 0b, KVM_Sel = 0b.

11. The MM responds to the *Blade KVM-MT Message* and immediately sends a *Set Blade KVM-MT Selection* (not shown) to the new owner. The MM will also take the appropriate hardware actions to switch the resources from the releasing Processor Blade.

The previous example was releasing one resource at a time. The MM can also instruct the Processor Blade to release both the KVM and MT resources at the same time, where the KVM – MT bits of the *Set Blade KVM-MT Selection* bits are set to MT_Sel = 0b, KVM_Sel = 0b.

5.11 FRU Data

Processor Blades store Field Replaceable Unit (FRU) non-volatile data for its components, including the Processor Blade itself. This data, including Asset Tags, part numbers, serial numbers, etc. is stored on the Processor Blade in non-volatile storage, usually an EEPROM.

Processor Blades advertise the existence of a FRU by the use of *FRU Locator Records* (SDR type 11h). This record identifies the FRU option, the access method for obtaining the FRU data, and the format of the vital product data once obtained.

The requirements for FRU devices are:

1. A FRU Locator Record must be provided for all FRU devices.
2. All FRU devices on a blade must be represented as a Logical FRU Device for access purposes.
3. Must provide FRU data in format supported by the MM firmware. Current formats supported by the MM firmware are:

- IPMI FRU Format
- SPD (Serial Presence Detect - is the EEPROM on a memory module that contains technical and manufacturing information about the memory module)
- IBM Format VPD

4. All FRU devices have associated IPMI presence sensors for FRU detection purposes.

5.11.1 IBM VPD Requirements

The IPMI specification strongly recommends that FRU manufacturers use IPMI FRU formats. This specification requires IBM Vital Product Data formats in support of legacy hardware, applications, and IBM field support. This requirement may be met by supporting IBM VPD formats directly via FRU data format and the use of IPMI standard *FRU Read/Write* commands. Alternatively, BMC's may advertise FRU data in another standard format, such as IPMI FRU Data format, and provide the required IBM VPD information via the OEM *Read/Write IBM Format VPD* commands. For those FRU devices where IBM VPD is required, the MM will use the OEM *Read/Write IBM Format VPD* if the advertised FRU data format in the FRU Locator Records is not identified as IBM VPD format.

FRU Devices that have an IBM VPD requirement are the following:

- Main Planar Board – the main system board.
- I/O Expansion Card(s) - For example an I/O expansion card that supports expansion of the network interfaces.
- Blade Expansion Module (BEM) – An expansion blade option that connects to the main planar board. This module extends the functionality of the main planar board. There are several types of these modules:
    - Supports extending the Processor Blade I/O support; for example additional local hard drives, additional I/O Expansion Cards or PCI expansion support.
    - Supports adding additional capabilities of the Main Planar Board that adds additional processors, memory and/or I/O adapters.
    - If the Main Planar Board supports various BEM's, it may be possible that the Main Planar Board and BEM's can be stacked together to makeup various multi-wide blade combinations.

The list above may be expanded in future versions of this document, though such expansion will not impact legacy support.

Notes:
(1)The MM requires that if VPD information is modified by some other mechanism not originated by the MM, a *Blade State Change* message must be sent by the BMC to the MM. Modified means the data actually changed in the VPD, not that the VPD was written to. If the information was changed in the VPD, the MM must be notified so that VPD data saved on the MM may need updated.

(2) MM follows IPMI requirements to read vital product data in small data payload increments, 24 bytes or less. However, to improve performance issues, the MM supports using larger command packets for these read/write operations. However, the BMC must advertise that it supports larger command packets before this option is used by the MM. Please seeTable 5-17 Blade Capability Definitions' for the "IPMI Command Larger Payload Support" bit definition.

Refer to section 5.19.3 for format information on the IBM *VPD Format Read* and IBM *VPD Format Write* commands. In addition more information on the VPD structure can be found in the '*BladeServer - Base Specification For VPD*'.

5.11.2 FRU Locator Record Support of VPD Formats

Processor Blades have the option to support standard formats for FRU data and provide any required IBM VPD information via support of the IPMI OEM *Read/Write IBM Format VPD* commands. This section deals with those blades that may alternatively wish to advertise IBM VPD formats for their FRU data. In this case, the MM will use standard IPMI *FRU Read/Write* commands to obtain the IBM VPD information.

IPMI does not define a Device Type for IBM VPD formats. Table 5-3 IPMI FRU Locator Record represents an OEM extension to allow for IBM VPD format advertisement.

Table 5-3 IPMI FRU Locator Record

Byte	Field Name	Size	Description
	RECORD HEADER		
1:2	Record ID	2	As per IPMI Specification.
3	SDR Version	1	
4	Record Type	1	
5	Record Length	1	
	RECORD KEY BYTES		
6	Device Access Address	1	As per IPMI Specification.
7	FRU Device ID / Device Slave Address	1	[7:0] Logical FRU Device ID. Logical FRU device support required.
8	Logical-Physical / Access LUN / Bus ID	1	[7] - logical/physical FRU device = 1b Device is a logical FRU Device. [6:5] Reserved = 00b [4:3] LUN [2:0] = 000b (Device is logical FRU Device).
9	Channel Number	1	As per IPMI Specification.
	RECORD BODY BYTES		
10	Reserved	1	
11	Device Type	1	= 10h Logical FRU Inventory Device
12	Device Type Modifier	1	= 00h IPMI FRU Inventory Format = FFh Unspecified Format (see OEM field)
13	FRU Entity ID	1	As per IPMI Specification.
14	FRU Entity Instance	1	As per IPMI Specification.
15	OEM	1	[7] 1b = VPD format [6:0] Reserved = 0000000b

Byte	Field Name	Size	Description
16	Device ID String Type/Length		As per IPMI Specification.
17:+N	Device String		

5.11.2.1 Main System Board VPD

If the BMC uses the FRU Locator records format as the method of advertising IBM VPD formats by a blade product, then the main system board requires special care. For IPMI, a *Management Controller Locator* Record is provided by the blade as per the IPMI standards. This record indicates the capabilities of the BMC. The BMC can indicate that it acts as an inventory device and uses the Logical Device ID of 0h for accessing inventory data. The format of this data is always IPMI FRU data.

To allow the main system board to support IBM VPD formats via standard IPMI *FRU Read/Write* commands, then an additional *FRU Locator Record* is required. The record should identify the access method as a logical fru device supporting VPD format as described in 5.11.2. The IPMI Entity ID 07h "Main System Board" will be used for this record.

5.11.3 Entity Association Records for Physical Containment

Topology is further identified by the BMC's use of Entity Association Records. These SDR's are used for logical containment, but will be used by the MM for physical containment as well. This provides the BMC a means to present topology and inventory data in a clearer and more organized manner. In other words, Entity Association Records can show FRU containment on another FRU on the Processor Blade, where if the Entity Association Record is not available, all FRU's would be shown as contained on the blade main system board (no containment data).

Table 5-4 Entity Association Record - SDR Type 08h

Byte	Field Name	Size	Description
	RECORD HEADER		
1:2	Record ID	2	As per IPMI Specification.
3	SDR Version	1	
4	Record Type	1	
5	Record Length	1	
	RECORD KEY BYTES		
6	Container Entity ID	1	Entity ID of container entity.
7	Container Entity Instance	1	Instance ID of container entity. [7] - 0b = container entity is logical group container. 1b = container entity is physical container.
8	flags	1	As per IPMI Specification.
9	Contained Entity 1 / Range 1 entity	1	
10	Contained Entity 1 Instance	1	

Byte	Field Name	Size	Description
	RECORD BODY BYTES		
Body is as per IPMI Specification.			

Note: the additional definition of the Container Entity Instance field of byte 7 of the Entity Association Record SDR that is shown in ***Error! Reference source not found.***. This method is used to convey to the AMM firmware that the container entity is a physical container, and allows fuller topology support.

5.11.3.1 Device-relative Entity Association Records

The Device-relative Entity Association Record – SDR Type 09h is similarly to be used. The Container Entity Instance field sets the MSb [7] location to a 0b to indicate the standard logical group container type and 1b to indicate a physical container type.

5.12 Processor Blade Front Panel Push Buttons

The processor blade contains the following front panel push buttons:

- Power control Button
- Keyboard/Video/Mouse Select Button
- MEDIA/USB Select Button
- NMI Reset Button

The BMC must monitor the front panel push buttons. The BMC firmware is required to de-bounc the push buttons. The de-bounce time is 50msec, therefore the signal must be in a constant low state for 50msec before it is treated as asserted.

5.12.1 Power Button

This push button controls the power on or off of the processor blade. *Refer to the section on Power Management for details.*

5.12.2 KVM Select Button

This push button is utilized to request that the KVM be allocated to the processor blade; refer to the section on KVM and Media Tray Resource Sharing for details on BMC to MM protocol.

5.12.3 Media/USB Select Button

This push button is utilized to request that the Media/USB be allocated to the processor blade; refer to the section on KVM and Media Tray Resource Sharing for details on BMC to MM protocol.

5.12.4 NMI Reset Button

This hidden reset button is used to initiate a system NMI for the processor blade.

5.13 LEDs and Usage

The LEDs listed in this section are required for Blade operation. The LEDs include front panel LEDs as referenced in the section 5.13.1. LEDs that are internal to the blade and not physically visible to a system administrator are described in section 5.13.2. The MM requires the Processor Blade to support the capability of the MM to get the current state of an LED located on the Processor Blade and in some cases (detailed below) to change the state of an LED. This is a very important feature where multiple Processor Blades exists in a single chassis.

While most LEDs are controlled directly by the blade BMC there are commands to provide external support for reading and controlling the state of the LED's. One reason is for illuminating LEDs may be for example a DIMM failure, when system firmware (BIOS) detects a failure that cannot be detected by the BMC. The other reason for external control is for testing the LEDs in a diagnostic mode. There are also methods for determining which LEDs are supported on the given Processor Blade and their type and usage.

The MM employs a method of learning all the Processor Blade supported LED's and their associated properties by the use of OEM SDRs which use the "Modular Blade Server" IANA to scope to BladeServer Architecture. In this way, a single method is defined for all BladeServer products to use and allow the MM firmware to display LED properties, states and control. These SDR formats may be found in section 5.25.1 ***OEM Type 01h – LED*** Properties. When the MM needs to determine or control the state of the LED's the OEM IPMI command *LED Set/Get* in Table 5.14 is utilized.

These LEDs are to be controlled directly by the BMC, host firmware and/or the MM and are not to be set by a Host IPMI SMS. The Host SMS is allowed to read LED states.

5.13.1 Processor Blade Front Panel LEDs and Behavior

The following are the Processor Blade Front Panel LEDs.

- Power LED
- Location LED
- Information LED
- Fault LED
- Keyboard, Video, Mouse Selected LED
- CD, FDD and USB Selected LED
- Activity LED

The KVM, CD, FDD, Location, Blade Fault and Information LEDs should illuminate when the blade is powered off (in standby).

LED Description	**LED State**	**Condition**
Power LED shows state of blade DC power and is controlled by the BMC	Off	Standby power(PD1) is not available to the blade
	On	The blade PD2 is on

	Slow Blink (~1 Hz)	Standby power available (PD1) and the MM has granted power-on permission, but the blade has not been powered on (PD2 is off).
	Fast Blink (~4 Hz)	Standby power available (PD1) and power-on permission has not been granted. This LED must blink at this rate before changing to the 'Slow Blink' rate while MM is establishing the power permissions for the blade.
Location LED is used as a visible indicator of the blade location and is controlled by the BMC	Off	MM has requested BMC to turn LED off
	ON	MM has requested BMC to turn LED on
	Blink (~1/2 Hz)	MM has requested BMC to blink LED
	Additional LED state information	When PD1 is activated at the blade, the LED state should be off. BMC must not preserve the state of this LED if PD1 power is removed for any reason. The state of the LED will be preserved across blade power-on/off or reset.
Fault LED is used as a visible indicator of a blade error condition	Off	BMC has not detected any blade fault conditions, or a previous fault condition has been corrected
	On	BMC has detected that one of the following blade fault conditions exist: • Processor disabled • DIMM fault • One of the failure LED(s) are on • Temperature or voltage critical threshold exceeded • Power fault • Processor mismatch or incorrectly installed The BMC must always inform the MM when a fault is detected via a DEM.
Information LED is used as a visible indicator that a blade may have a condition detected by the MM that may cause it to run in a degraded mode	Off	MM has not detected a blade problem or MM directed blade to turn off this LED
	On	MM has detected that the blade may have a problem and directed the blade to turn on the LED
KVM Select LED is used as a visible indicator that a blade has ownership of the KVM	Off	KVM is not selected to the blade
	On	KVM is selected to the blade
	Blink (~1/2 Hz)	Blade has requested ownership of the KVM but ownership has not yet been granted
Media/USB Select LED is used as a visible indicator that a blade has ownership of the media tray	Off	Media/USB is not selected to the blade
	On	Media/USB is selected to the blade
	Blink (~1/2 Hz)	Blade has requested ownership of the Media/USB but ownership has not yet been granted
Activity LED is used as a visible indicator that the blade has hard disk or network activity (BMC does not control this LED)	Off	No hard disk or network activity
	Blink	Hard disk is being accessed or one of the network ports on the blade is being used
Notes: (1) Blink modes have a nominal 50/50 duty cycle.		

Table 5-5 - Blade LED Table

5.13.2 Board failure LEDs on Processor Blade

The following table shows the Processor Blade planar LEDs and behaviors required. As mentioned previously the planar LEDs and the SDR formats may be found in 5.25.1 ***OEM Type 01h – LED*** Properties. When the MM needs to determine the state of the LEDs the OEM IPMI command *LED Get* in Table 5-27 OEM Formats is utilized..

Table 5-6 - Board Failure LEDs and Behavior

LED	Failure Condition	Front Panel Fault LED Set	LED Set	Blade power cycle(off/on) will clear LED	Blade reset will clear LED	Signal deasserted or threshold recovered	Recovery Actions by Blade
DIMM Fault LEDs (1 per DIMM)	**DIMM failure** (detected by BIOS) or **No usable DIMMs present**[1]	Yes	Yes	Yes	Yes	n/a	None
Processor Board Planar Fault LED	**Processor board VR fault** – fault signal assertion	Yes	Yes	Yes	Yes	Yes	Power down blade
Board Temp Fault LED	**Temperature fault** –temp sensor has upper critical threshold crossing	Yes	Yes	Yes	Yes	Yes	If any processor internal temp sensor or BSE temp sensor reaches upper critical threshold, the BMC powers down the blade.
	Processor ThermTrip – signal assertion	Yes	Yes	Yes	Yes	Yes	Powers down blade
	Temperature Warning Threshold exceeded	Yes	Yes	Yes	Yes	Yes	None
Processor Fault LEDs (1 per processor)	**Processor IERR –** signal assertion	Yes	Yes	No	No	No	Resets blade. If all installed processors have had valid IERRs, blade is powered down.

	Processor Error, Config Error, Proc Disabled	Yes	Yes	No	No	n/a	These errors may result in a system reset.
Processor Mismatch LED	**Processor mismatch** -signal assertion	Yes	Yes	n/a – HW prevents blade from powering on	n/a – HW prevents blade from powering on	No	
	Processor mismatch – detected by BIOS	Yes	Yes	Yes[2]	Yes[2]	n/a	
BMC Fault LED	**BMC Self-test Failure** - BMC first initializes after being reset or having standby power applied.	Yes	Yes	No	No	n/a	If BMC detects corrupt operational code, it enters FW Transfer Mode.
NMI Fault LED	**NMI Fault** – signal assertion (not by BMC)[3]	Yes	Yes	Yes	No	No	None
Hard Drive Fault LEDs (1 per drive)	The BMC does not detect faults	n/a	n/a	n/a	n/a	n/a	n/a

Note 1: Fault state is asserted for all DIMM slots. In this case, fault state may be asserted for slots that have no DIMMs installed.
Note 2: When BIOS detects a fatal processor mismatch condition, it will power down the system within 5-10 seconds. If the system is reset or power cycled before this time is up, then the errors will be cleared.
Note 3: Fault LEDs are not lit due to an NMI asserted by the BMC, such as due to a Front Panel NMI Reset Button.

5.14 Remote Flashing

The MM supports the capability of flashing the operational BMC firmware (not kernel/boot firmware) on a Processor Blade. In order for the MM to provide support for flashing of the BMC firmware on the Processor Blade the following must be provided:

- A “Blade Firmware Update File Format” (BFUFF) image file must be provided for the blade product. This file described below will be uploaded to the MM and then utilized to perform a firmware update of the BMC. The “Blade Firmware Update File” (BFUF) is made up of two components, a XML header and the BMC binary flash image.

- The BMC firmware’s binary image will be sent incrementally to the BMC using the defined set of BladeServer IPMI OEM firmware update commands (see flash command sequence in section 5.14.3). This support is mandatory to allow blade flashing from the MM when the blade is in an inoperable state and cannot obtain power permission or for other reasons cannot be powered on to perform a local in-band update.

5.14.1 Blade Firmware Update File Format

The BFUFF is a file structure that contains the BMC firmware distributed by the blade product vendor to allow for the MM to handle BMC firmware updates generically for various blade vendors in a generic way. The MM will provide a method to upload the file to the MM and then perform a firmware update command sequence to the BMC (described in section 5.14.3). This image is made up of a XML header that provides meta-data used to identify the file contents and the binary operational firmware data that will be sent to the BMC. The binary data sent to the BMC will be flashed into the BMC's non-volatile storage using the IPMI OEM flash command sequence. The format for the BFUFF is as described below.

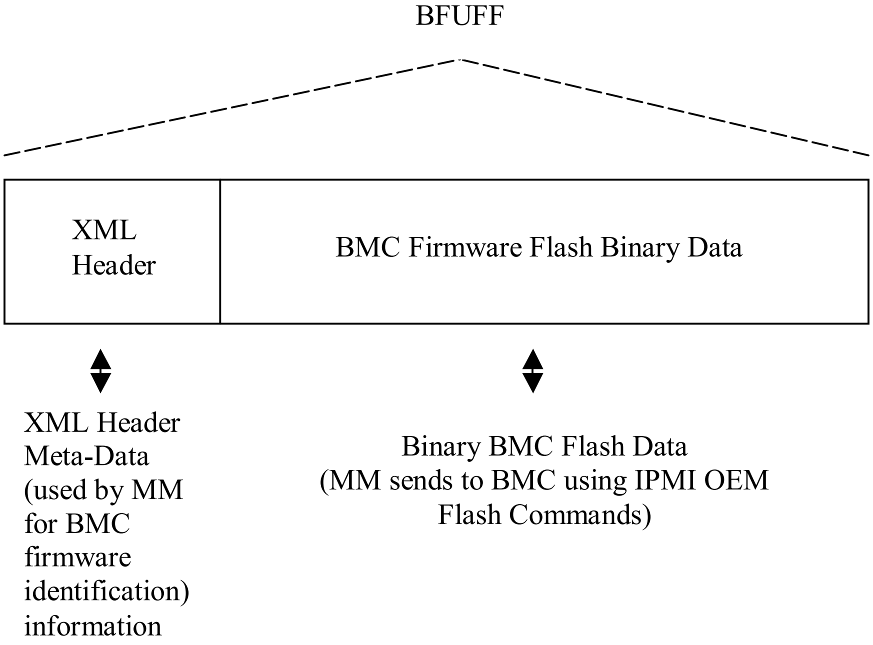

BFUFF
XML Header
BMC Firmware Flash Binary Data
XML Header Meta-Data (used by MM for BMC firmware identification) information
Binary BMC Flash Data (MM sends to BMC using IPMI OEM Flash Commands)

Figure 5-13 Blade Firmware Update File Format

5.14.2 XML Header Description

The meta-data contained in the firmware update file is specified using XML. The complete XML schema is given below. Note that this schema has an attribute "version" of XML data type "formatversion_type" which restricts it to currently being version 1.0.

5.14.2.1 BladeServer BMC Firmware Update XML Schema Definition (“bmc_upd.xsd”)

```
<?xml version="1.0"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <xsd:annotation>
    <xsd:documentation>Blade Firmware Update File Format</xsd:documentation>
  </xsd:annotation>
  <xsd:element name="BladeFWUpdFmt" type="blade_fw_upd_fmt_type"/>
    <xsd:complexType name="blade_fw_upd_fmt_type">
      <xsd:attribute name="version" type="formatversion_type" use="required"/>
        <xsd:sequence>
          <xsd:element name="image" type="image_type"/>
        </xsd:sequence>
    </xsd:complexType>
    <xsd:complexType name="image_type">
      <xsd:sequence>
        <xsd:element name="size" type="xsd:unsignedInt">
          <xsd:annotation>
            <xsd:documentation>Size (in bytes) of the binary image in this update</xsd:documentation>
          </xsd:annotation>
        </xsd:element>
        <xsd:element name="copyright" type="copyright_type">
          <xsd:annotation>
            <xsd:documentation>Copyright text for the firmware update</xsd:documentation>
          </xsd:annotation>
        </xsd:element>
        <xsd:element name="manuf_id" type="xsd:unsignedInt">
          <xsd:annotation>
            <xsd:documentation>Manufacturer id assigned by IANA (same value in Get Device ID cmd) </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
        <xsd:element name="product_id" type="xsd:unsignedLong">
          <xsd:annotation>
            <xsd:documentation>Product id assigned by manufacturer (same value in Get Device ID cmd)</xsd:documentation>
          </xsd:annotation>
        </xsd:element>
        <xsd:element name="device_type" type="devicetype_type">
          <xsd:annotation>
            <xsd:documentation>Type of BladeServer - used to determine flashing mechanism</xsd:documentation>
          </xsd:annotation>
        </xsd:element>
        <xsd:element name="release_date" type="xsd:date">
          <xsd:annotation>
            <xsd:documentation> User-readable release date of this firmware update</xsd:documentation>
          </xsd:annotation>
        </xsd:element>
        <xsd:element name="build_id" type="buildid_type">
          <xsd:annotation>
            <xsd:documentation> User-readable Build identifier assigned by manufacturer</xsd:documentation>
          </xsd:annotation>
        </xsd:element>
        <xsd:element name="file_name" type="filename_type">
          <xsd:annotation>
            <xsd:documentation>Binary image file name in this firmware update package.</xsd:documentation>
          </xsd:annotation>
        </xsd:element>
        <xsd:element name="image_name" type="imagename_type">
          <xsd:annotation>
            <xsd:documentation>User-readable name of the firmware image</xsd:documentation>
          </xsd:annotation>
        </xsd:element>
        <xsd:element name="image_version" type="imageversion_type">
          <xsd:annotation>
            <xsd:documentation>Version of this firmware image</xsd:documentation>
          </xsd:annotation>
        </xsd:element>
      </xsd:sequence>
    </xsd:complexType>
    <xsd:simpleType name="copyright_type">
      <xsd:restriction base="xsd:string">
        <xsd:maxLength value="255"/><xsd:whiteSpace value="preserve"/>
      </xsd:restriction>
    </xsd:simpleType>
    <xsd:simpleType name="devicetype_type">
      <xsd:restriction base="xsd:unsignedShort">
        <xsd:enumeration value="0x0001">
          <xsd:annotation>
            <xsd:documentation>This image is for a blade server management controller</xsd:documentation>
          </xsd:annotation>
        </xsd:enumeration>
      </xsd:restriction>
    </xsd:simpleType>
    <xsd:simpleType name="buildid_type">
     <xsd:restriction base="xsd:string">
        <xsd:maxLength value="15"/>
        <xsd:whiteSpace value="preserve"/>
     </xsd:restriction>
    </xsd:simpleType>
    <xsd:simpleType name="filename_type">
      <xsd:restriction base="xsd:string">
        <xsd:maxLength value="255"/>
        <xsd:whiteSpace value="preserve"/>
      </xsd:restriction>
    </xsd:simpleType>
    <xsd:simpleType name="imagename_type">
      <xsd:restriction base="xsd:string">
        <xsd:maxLength value="511"/>
        <xsd:whiteSpace value="preserve"/>
      </xsd:restriction>
    </xsd:simpleType>
    <xsd:simpleType name="imageversion_type">
     <xsd:restriction base="xsd:string">
        <xsd:maxLength value="15"/>
        <xsd:whiteSpace value="preserve"/>
    </xsd:restriction>
    </xsd:simpleType>
      <xsd:simpleType name="formatversion_type">
        <xsd:restriction base="xsd:string">
          <xsd:pattern value="[1][.][0]"/>
        </xsd:restriction>
      </xsd:simpleType>
  </xsd:schema>
```

5.14.2.2 BladeServer BMC Firmware Update XML Header Example

Below is an example XML header that would be embedded in the Blade Firmware Update File. This is based on the XML schema defined previously in the ‘bmc_upd.xsd’ in section 5.14.2.1.

```
<?xml version="1.0" ?>
<IBMBladeCenterFWUpdFmt version="1.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="file:///c:/bmc_upd.xsd">
  <image>
    <size>445854</size>
    <copyright>Copyright ABC</copyright>
    <manuf_id>2</manuf_id>
    <product_id>34</product_id>
    <device_type>1</device_type>
    <release_date>2007-01-27</release_date>
    <build_id>BWBT24A</build_id>
    <file_name>bmcabc.bin</file_name>
    <image_name>ABC Blade BMC Firmware</image_name>
    <image_version>1.3.5.7</image_version>
  </image>
</IBMBladeCenterFWUpdFmt>
```

5.14.3 MM to Processor Blade Firmware Update Sequence

To standardize to a single firmware update method for blades in the BladeServer chassis, an *IPMI OEM* flash command set has been defined. This support is mandatory to allow Processor Blade flashing from the MM when the Processor Blade is in either a powered on/off state or in a state whereby the BMC cannot obtain power permission.

The benefits of the MM remote flash support are:

- Flash operation available in a blade power down state.

- Security. More secure option due to initiation of operation is performed by way of the external MM port, which is on a management LAN that may be user configured as a private network with security in place to protect from public LANs.

- Simplicity. Since a common method is used for all blades, regardless of vendor, the user does not have to learn the tools that are specific to that vendor product.

- More User Options. The system administrator may use BladeServer Chassis management utilities to flash multiple vendor blades, providing more options on how the operation will be performed in a common fashion.

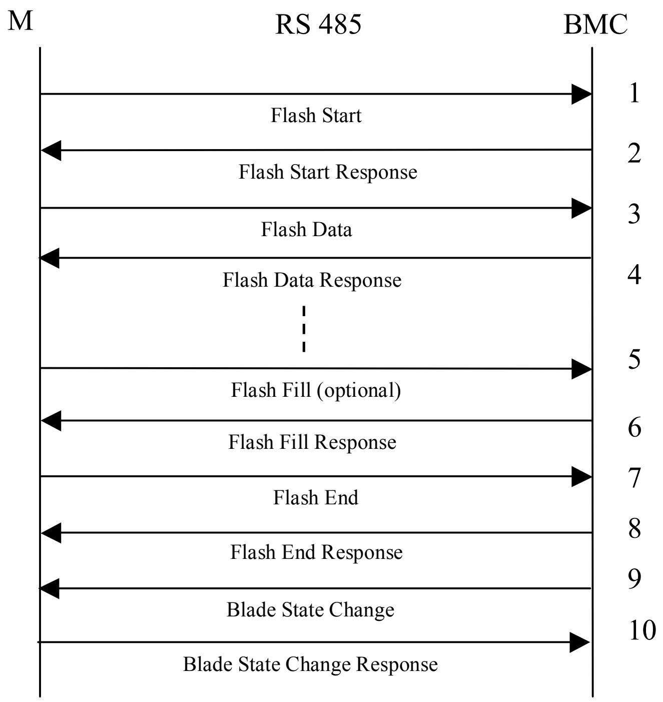


Figure 5-14 MM Remote Flash Operation

1. The MM sends the *Flash Start* command.

2. The Processor Blade should respond with the *Flash Start Response* message, which also includes the maximum size in bytes the Processor Blade can accept in a single *Flash Data* command.

3. The MM then proceeds to send the flash firmware with the *Flash Data* command. This process is repeated until the firmware has been sent to the Processor Blade.

4. The Processor Blade will save the firmware data received and confirm to the MM that the data has been received successfully by sending the *Flash Data Response* message. This process is repeated for each *Flash Data* message received.

5. During the firmware transfer, the MM may encounter a large firmware portion of data that contains a repeatable pattern. To facilitate firmware flashing efficiency by reducing the number of *Flash Data* commands, the MM will perform the optional *Flash Fill* command. If the blade responds with an IPMI Completion Code of “Command Not Supported”, the operation will cease and continue with Flash Data commands. If successful, this *Flash Fill* command will be used to greatly reduce the time required for the flash operation.

6. The Processor Blade responds with the *Flash Fill Response* if supported, else the completion code of “Command Not Supported” is returned.

7. After successfully completing the firmware transfer, the MM will indicate the end of flashing operations by sending a *Flash End* command.

8. The Processor Blade sends the *Flash End Response* message to the MM. The Processor Blade should now activate the operational level firmware.

9. The BMC in the Processor Blade should now activate the operational BMC level firmware that was just loaded. This occurs when the Processor Blade BMC exits firmware mode and enters operational state. When the BMC becomes available, the *Blade State Change* message is sent by the Processor Blade to notify the MM that the Processor Blade has undergone a blade state change.

10. The MM sends the *Blade State Change Response* and notes the availability of the Processor Blade BMC. The MM will initiate the Processor Blade Chassis Initialization (see section 5.5 Processor Blade Chassis Initialization). If the *Blade State Change Response* message is not sent or a completion code other than success is returned, then the MM may not grant power permission or may restart the flash process.

Notes:

(1) Any error or non-zero completion code detected by the MM may result in the MM not granting power permission or restarting the flash process.

(2) The BMC remote flash process described above details the sequence of commands that needs to occur in performing a successful update of the BMC firmware. During this sequence of commands, there could be external reasons that the flash process fails. For example a RS485 bus hang may occur, a MM is reset/removed or the ServerBlade is removed. The BMC should attempt to minimize the impact of failures during this flash process that would leave the operational firmware in a corrupted state.
The following should be part of the BMC flashing design:

- Staging the flash image data into a buffer and insuring the complete image is received from the MM before updating the operational firmware is recommended but optional.
- Provide a timeout at the start of the flash process. This timeout would be after the 'Flash Start Command' has been processed by the BMC and prior to the first 'Flash Data Command' has been processed by the BMC. If this timeout is exceeded, then the BMC should abort the flash process and revert back to its operational firmware.

5.15 Processor Blade LAN Configuration

The MM uses IPMI LAN Parameters Configuration commands to allow system administrators access and management of Processor Blade LAN configurations. The Processor Blade LAN channel is used in support of the MM Serial-Over-LAN (SOL) Proxy service.

The MM will use the IPMI LAN Parameter commands to setup the Processor Blade for SOL support if such support has been advertised by the Processor Blade to the MM (learned via the capability bits in the *BladeServer IPMI OEM Get Blade Descriptor* OEM command) and the system administrator has enabled MM SOL Proxy support for that particular blade. IPMI Sessions must be allowed for SOL.

The *BladeServer IPMI* OEM command *Restore LAN Configuration* is to be supported to allow the MM to reset the Processor Blade LAN configuration to its default values. This command must only be supported on the interface between the Processor Blade and MM. This command is not valid on any other interface/channel.

LAN defaults are implementation specific, defined by the target blade environment.

5.16 MM SOL Proxy Support

Processor Blades and some BladeServer chassis have no external serial port connectors. To provide serial port function for each Processor Blade in BladeServer the MM provides SOL proxy services from its external LAN port to those Processor Blades with SOL support. This service can be disabled or enabled using the various MM User Interfaces.

SOL for Processor Blades is an optional feature and is discovered by the MM during the Processor Blade Chassis Initialization process using the *Get Blade Descriptor* command. The MM may be configured to act the role of an SOL proxy service and if so, will establish an IPMI SOL Session to the Processor Blade. The Processor Blade's BMC LAN Parameters will be configured by the MM to the appropriate BMC VLAN, BMC IP address and BMC Gateway IP address to allow the IPMI SOL Session to take place.

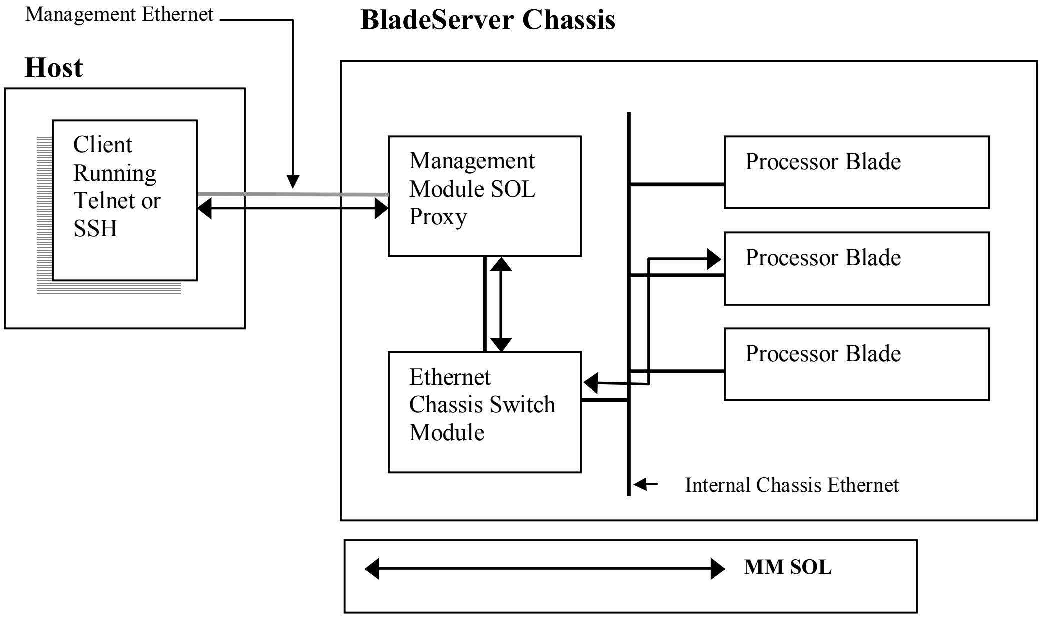

Management Ethernet
BladeServer Chassis
Host
Client Running Telnet or SSH
Management Module SOL Proxy
Processor Blade
Processor Blade
Processor Blade
Ethernet Chassis Switch Module
Internal Chassis Ethernet
MM SOL

Figure 5-15 SOL Support

Figure 5.15 represents a client establishing either a telnet and/or SSH session to the MM over the MM external Ethernet port. The MM SOL proxy support will then allow external users to interact via the telnet and/or SSH client with the Processor Blades that have an internal active SOL session established. SOL is an optional function provided by BladeServer. References to how the MM exposes SOL function is shown for informational purposes only.

5.16.1 Serial Over LAN Operation

SOL allows for bi-directional transport of the processor blade's serial port data encapsulated in Remote Management Control Protocol RMCP packets over an Ethernet LAN interface. The packet payload will use IPMI as the transport for the bi-directional host serial port data stream.

As shown below the BMC will have two sideband connections, one to the processor blade's serial port and the other to the processor blade's NIC's. The processor blade serial port is connected to the UART using the standard RS-232 interface. The BMC is connected to the Ethernet NIC using an interface such as the SMBus that will allow transfer of the serial port data between the BMC and the NIC. This allows for the BMC to process the data packets to and from the LAN interface. The LAN interface connects to the internal management infrastructure using an Ethernet Switch Module in either SM Bay 1 or 2 and allows the MM to act as a proxy for the SOL data.

Note: For simplicity, Figure 5.16 does not show a redundant NIC, but one should be assumed and therefore the BMC would have two SMBus connections, one to each NIC.
UART

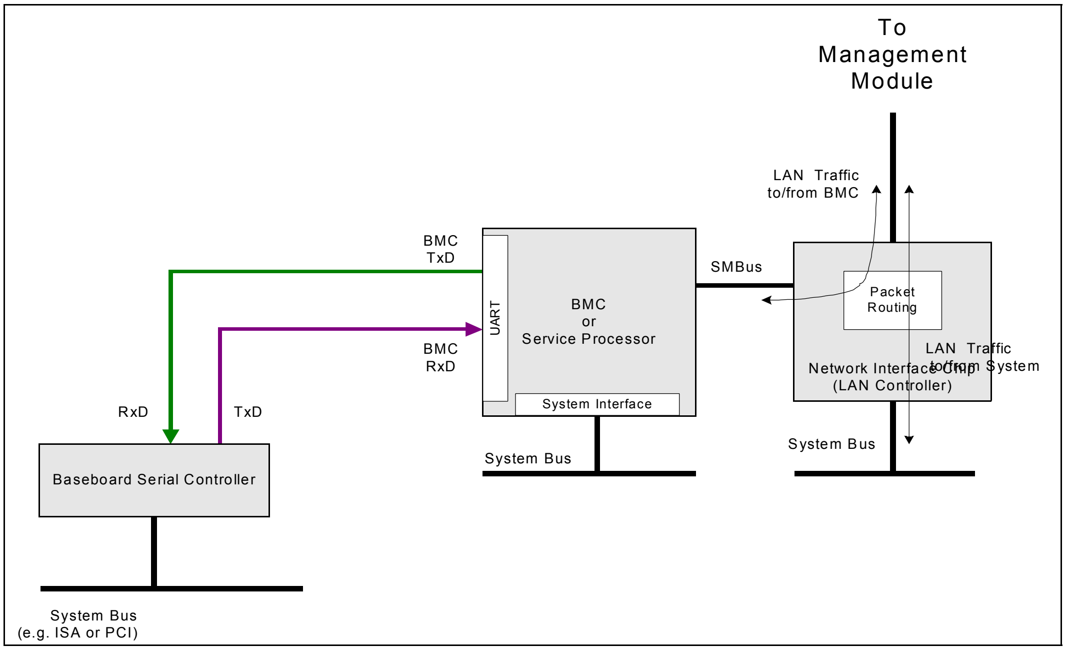


Figure 5-16 SOL Diagram

The BMC must only allow the MM to establish an IPMI-over-LAN session with the BMC. Once the session is established, the MM can request that SOL be activated. Once the session is established and SOL is activated any outgoing characters from the Processor Blade's serial controller will be packetized by the BMC and sent to MM over the Processor Blade's Ethernet NIC. Conversely, in-bound LAN packets from the MM carrying characters for the Processor Blade's serial controller have their character data extracted by the BMC and delivered to the host serial controller. SOL therefore provides LAN access to interact in the pre-OS (ex. BIOS) console data or a text based OS console interaction (e.g., DOS, Linux, Microsoft Emergency Management Services).

5.16.2 Management VLAN

Figure 5.17 depicts the BladeServer internal LAN infrastructure between the MM, SM and processor blades. A private management VLAN between the MM and each of the blades is shown. For simplification, the figure does not show the redundant components (redundant switch, and redundant processor blade LAN connections), but they are symmetrical.

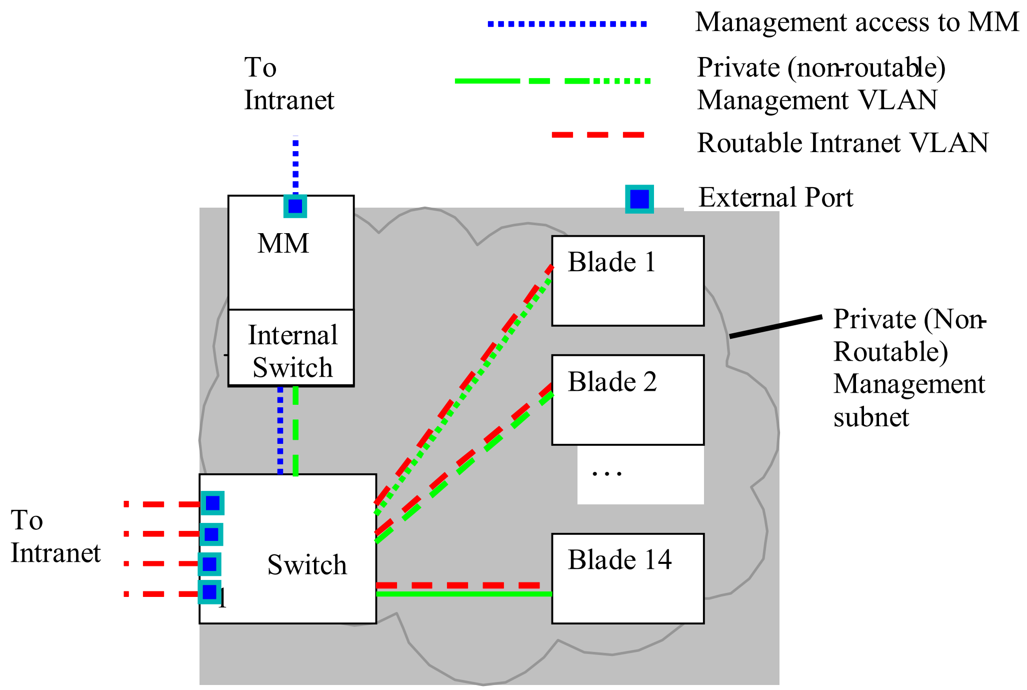


Figure 5-17

The MM communicates over Ethernet using switch modules and the BMC on the blades using a private VLAN. Serial over LAN traffic flows over this VLAN from the blade to the MM. The internal processor blade port and the management port of the switch are pre-configured to support tagging of the packets for the management VLAN. When the blades are inserted into the chassis, the MM sends the VLAN tag and IP configuration information to the BMC on the blades for the purpose of SOL. This is done by sending a message via the RS485 bus. The BMC must use information in this command to configure the network controller on the blade with the VLAN tags. Once this is done, the private VLAN between the MM, SM and the blades is completely set up.

When the MM needs to start up a SOL session with a blade BMC, it also sends an IPMI message over the RS485 bus to the BMC with a username/password combination. This username/password information is later used in establishing the SOL session using IPMI/RMCP protocol, thus removing the burden of pre-configuration of username/passwords on individual blades.

The blade may have to be powered on at least once to be able to get the MAC address from the blade NIC, so that it can send the gratuitous ARP to allow the MM to start communicating with it with RMCP packets.

5.16.3 Processor Blade BMC IP Configuration and VLAN Setup

As mentioned previously, the MM will send an IPMI command over the RS485 bus to the Processor Blade BMC to configure the IP address, Subnet Mask , Default Gateway Address and VLAN tag to be used for the SOL traffic. The IPMI command used is '*Set LAN Configuration'*. The '*Set LAN Configuration'* command, in addition the following OEM LAN configuration parameters '208' and '209' have been added to this command to support setting the VLAN tag id and VLAN control. From the data in the IPMI command the BMC will configure the BMC and NIC. This will allow packets with the matching VLAN tag in the MAC header to be passed on to the BMC. Outgoing packets from the BMC to the MM must contain the VLAN tag as configured by the MM.

OEM LAN Configuration Parameters		
Code	Parameter	Description
208 (OEM)	VLAN ID	data 1 – VLAN ID (msb) data 2 – VLAN ID (lsb) This specifies the VLAN ID portion of the VLAN tag for the channel. The *VLAN tag enable* parameter must be set for 'enable' in order for tagging to be applied.
209 (OEM)	VLAN tagging control	data 1 – VLAN tag enable 7:1 reserved 0:0 0 = disable VLAN tagging 1 = enable VLAN tagging When VLAN tagging is enabled, only Ethernet packets that have the correct VLAN tag, as specified by the *VLAN ID* configuration parameter, will be accepted by the BMC. Other packets will be silently rejected. All outgoing packets transmitted by the BMC will have the VLAN tag field inserted.

The BMC's LAN channels have default configuration values. As mentioned previously, these defaults can be changed through standard IPMI messaging and LAN configuration commands, the MM can force the defaults to be reinstated through the IPMI OEM command *Restore Blade LAN Defaults*.

The following table defines the 2 LANs supported for SOL session establishment and the connection to the respective Switch Modules.

Channel ID	Processor Blades Lan Interface	Chassis Swith Module Interface
6	LAN 2(Secondary)	Legacy Switch Module Bay 2
7	LAN 1(Primary)	Legacy Switch Module Bay 1

5.16.3.1 SOL NIC Teaming and Failover

The processor blade must have 2 LAN ports to support NIC teaming for OS based Ethernet traffic. Therefore the BMC must also support NIC teaming and fail-over. As is normally done with IPMI (non-SOL) traffic, the BMC responds to any incoming commands over the same channel that it received the command. For SOL traffic, the BMC transmits SOL packets over the same channel as the last received packet in the session. Therefore, when NIC teaming and failover are supported, the MM will configure both LAN channels of the BMC with the appropriate IP configuration and VLAN to insure the BMC will support a failover correctly.

5.16.4 Blade BMC Serial Redirection

Once the IP configuration and VLAN tag have been set up, the MM establishes an IPMI-over-LAN session with the BMC. Once the session is established, the MM can request that SOL be activated. From this point, any outgoing characters from the host serial port are packetized by the BMC and sent to the MM over the secure private VLAN. Conversely, in-bound VLAN packets carrying characters for the system serial controller have their character data extracted by the BMC and delivered to the host serial port.

The SOL character data is carried as SOL messages that are carried in UDP datagrams. The packet format is built on top of IPMI-over-LAN RMCP protocol with extensions to support SOL messages as a new message type. This is further detailed in the following sections in this document.

Flow control between the blade and the MM is handled by use of acknowledges (ACKs) and negative-acknowledges (NACKs) that indicate whether the BMC is ready to accept more data.

Note: SOL data to/from a processor blade will be forwarded by the primary processor blade Ethernet NIC during normal operation and will fail over to the secondary NIC if required due to failure or removal of the switch module associated with the primary NIC or failure of the primary NIC.

5.16.5 Serial-over-LAN protocol

5.16.5.1 Overall Packet Format

The packet format for the SOL packets is RMCP with IPMI header as defined below. The payload and the protocol are further defined in more detail in the following sections.

Contents	**Type**	**Offset**	**Value**	
Destination Address	6 bytes	00h		MAC
Source Address	6 bytes	06h		
VLAN Tag	4 Bytes	0Ch		
Frame Type	2 bytes	10h	0800h	
Version and Header Length	1 byte	12h		IP Header
Service Type	1 byte	13h		
Total Length	2 bytes	14h		
Identification	2 bytes	16h		

Flags & Fragment Offset	2 bytes	18h		
Time to Live	1 byte	1Ah		
Protocol	1 byte	1Bh	11h	
Header Checksum	2 bytes	1Ch		
Source IP Address	4 bytes	1Eh		
Destination IP Address	4 bytes	22h		
Source Port	2 bytes	26h		UDP Header
Destination Port	2 bytes	28h	026Fh	
UDP Length	2 bytes	2Ah		
UDP Checksum	2 bytes	2Ch		

Version	1 byte	2Eh	06h (ASF)	RMCP Header
Reserved	1 byte	2Fh		
Sequence Number	1 byte	30h		
Class of Message	1 byte	31h		
Authentication Type	1 byte	32h		IPMI RMCP Data
Session Sequence #	4 bytes	33h	0 – SoL Data 1 – Control Data	
Session ID	4 bytes	37h		
Message Authentication Code (This field is not present if the Authentication Type is set to 'none')	16 bytes	3Bh		
IPMI/SOL Message Length	1 byte	3Bh or 40h		
IPMI/SOL Message	Variable	3Ch or 41h	SoL Data, See SOL Message Format	
CRC	4 bytes			

5.16.5.2 Session Sequence Number Handling

The in-bound and out-bound session sequence numbers in the Session Header (see the IPMI 1.5 specification for more information on the Session Header) are shared across SOL and IPMI packets. SOL packets include their own message sequence numbers that are used for tracking missing and retried SOL messages.

5.16.5.3 SOL Packet Acknowledge and Retries

A packet acknowledge is of one of two types:

- An ACK, indicating that the packet has been received and all its data has been accepted
- A NACK, indicating that the packet was received but some or all of the data could not be accepted

To improve efficiency, the packet acknowledgment information for the last received packet can be carried in a send packet that also carries new SOL character data. Conversely, a packet can be an ACK-only packet that carries ACK or NACK information, but no data. However, packets are only acknowledged if they contain character data. Therefore, ACK-only packets themselves are not acknowledged. (Note: The MM assumes that the Processor Blade does not support the MM sending a NACK to the BMC unless this capability bit is set in the *Get Blade Descriptor* in the IPMI OEM command (see Table 5-24 for more information).

Except for ACK-only packets, the MM acknowledges each SOL packet that it receives. If the BMC does not receive an ACK packet within a timeout interval, the BMC will resend (retry) the packet. The number of retries and the amount of time between retries are configurable through the SOL Configuration Parameters.

It is possible that additional characters could be received from the Processor Blade serial controller while the BMC was waiting for a retry timeout. In order to improve throughput, the BMC is allowed to append additional characters to a packet when it resends it. To support this, the MM must accept retried packets (packets with the same packet sequence number) and check to see if the packet contains additional data. If the packet does contain addition data, the MM should accept the data and acknowledge the packet using the packet sequence number and character offset value that it had received.

5.16.5.4 SOL Message Format

Table 5.5 specifies the fields that make up the SOL Message field in an SOL Packet.

Table 5-7 - SOL Message Format

Field	Size	Description
Packet Sequence Number	1	Sequence Number for this packet. Sequence numbers must be non-zero. Multiple outstanding sequence numbers are not supported in this version of the specification. Retried packets use the same sequence number as the first packet. [7:4] Reserved

Field	Size	Description
		[3:0] Packet sequence number. 0h = ACK-Only packet.
Packet ACK /NACK Sequence Number	1	Sequence Number for packet being ACK'd or NACK'd. [7:4] Reserved. Write as 0h. (Future spec may use this to specify a range of packets being acknowledged) [3:0]: Packet sequence number being ACK'd/NACK'd. 0h: Informational packet. No request packet being ACK'd or NACK'd.
Character Offset	1	Character Offset 1-based. This field indicates the number of characters accepted from the packet, if any. 00h = Packet received but no data bytes accepted. **For BMC-to-MM:** In order to improve throughput, the BMC is allowed to append additional characters to a packet when it resends it. To support this, the MM must accept retried packets (packets with the same packet sequence number) and check to see if the packet contains additional data. If the packet does contain addition data, the MM should accept the data and acknowledge the packet using the packet sequence number and character offset value that it had received. *The MM must either accept all packet data sent to it or NACK the entire packet. It is not allowed to accept partial packet data.* **For MM-to-BMC:** The BMC is allowed to accept partial packet data by NACK'ing the packet and returning a character offset that is less than the data length sent by the MM. The MM would then send the next packet starting with the first byte of data that the BMC rejected.
Seed Count	1	[7:4] Reserved. [3:0] Seed Count Indicates which packets are given new 'seed' after the random number for encryption has been changed by the MM. The BMC increments this value each time the seed is updated using the *Activate SOL* command. This field is provided because it is possible that an SOL packet could already be committed to be transmitted even though the *Activate SOL* command was already responded to. Refer to the description of the *Activate SOL* command for more info.

Field	Size	Description	
Operation / Status	1	Packet to MM:	Packet to BMC:
		Operations are executed *before* character data is transferred. [7] 1b: Packet character data was encrypted by the BMC. 0b: Packet character data was not encrypted by the BMC. [6] 1b: Packet is being NACK'd. The BMC is unable to accept all character data from packet. Note: Operation field is still accepted even if packet is NACK'd. 0b: ACK. BMC ready to accept next packet of character data. [5] A NACK packet with this status will be automatically sent one time after this bit changes state. A NACK packet with "Character transfer is unavailable" status will also be sent for each character transfer request from the MM when the system is in a powered-down or sleep state. 1b: Character transfer is unavailable because system is in a powered-down or sleep state.0b: SOL character transfer is available. [4] A NACK packet with this status will be automatically sent one sent once, just before the BMC deactivates SOL because of a front panel power-button or a reset. 1b: SOL is deactivated/deactivating. [MM can use this to tell if SOL was deactivated by some other party, or by local pushbutton reset or power on/off]. 0b: SOL is active. [3:0] Reserved	Operations are executed *before* character data is transferred. [7] 1b: Packet was encrypted by MM. The BMC must decrypt before delivering character data to host serial port. 0b: The BMC does not decrypt packet. [6] 1b: Packet is being NACK'd by the MM. 0b: Packet is being ACK'd by the MM. [5] Assert RI (may not be supported on all implementations) - Goal is to allow this to be used for generating a WOR. [4] 1b: Generate BREAK (300 ms, nominal) [3] 1b: Deassert CTS to the baseboard serial controller. This is the default state when SOL is deactivated. 0b: Let BMC control CTS. [2] 1b: Deassert DCD/DSR to baseboard serial controller 0b: Assert DCD/DSR to baseboard serial controller. [1] 1b: Drop Inbound Character Data (drop (flush) data from MM to BMC [Not including the data carried in this packet, if any] [0] 1b: Flush Outbound Character Data (flush data from BMC to MM)
Character Data	Varies	Data length, in bytes, is equal to the IPMI Message/SOL Message Length field in the Session Header, minus the bytes for the Packet Sequence Number, through and including the Operation Field.	

5.16.5.5 SOL Activation

SOL is activated using the *Activate SOL* command after the session is established. This command also allows certain settings such as encryption to be enabled/disabled. SOL remains activated across system resets and PB2 power cycles as long as the corresponding IPMI LAN session remains activated. SOL is supported only between a single LAN session per blade, and a single, fixed, serial port on the host through the BMC.

5.16.5.6 SOL Deactivation

SOL can be deactivated using the *Activate SOL* command. It will also be automatically deactivated when the LAN session is closed, either by the *Close Session* command or by a session inactivity timeout. Valid SOL packets from the MM count as session activity for resetting the session inactivity timeout. A MM can elect either to periodically send an IPMI command over the session or an empty (no character data) SOL packet to keep the session active.
The BMC power-on default state for SOL is deactivated. SOL is also deactivated and unavailable to a blade during firmware update.

5.16.5.7 Outbound Serial Character Buffering

The BMC buffers the characters from the host serial port and then formats them into an SOL-stream packet for transmission on the LAN. There are two criteria that the BMC will use to decide when to send out the character data in an SOL packet:

- Number of characters received. This tunable value is set via the Character Send Threshold in the SOL configuration parameters.
- Time since last SOL packet was transmitted. This tunable value is set by the Character Accumulate Interval parameter in the SOL configuration parameters.

5.16.5.8 Serial Handshake

SOL requires the host serial port to use hardware handshake and, for SOL and the host serial port, to be configured for the same fixed bit rate. A MM application can force the BMC to deassert RTS and DCD/DSR to stop the host serial port from transmitting by using the Control field in the SOL packet. The BMC asserts RTS and DCD/DSR to the host serial port and begins to accept characters as soon as SOL is activated.

5.16.5.9 Encryption (Not supported by MM)

SOL supports the encryption of character data but the MM does not support it.

5.16.5.10 SOL Configuration Data

Table 5.6 lists the SOL configuration parameters. Parameters are read-write and non-volatile unless otherwise specified.

Table 5-8 - SOL Configuration Data

Parameter	#	Parameter Data (non-volatile unless otherwise noted) 1
Set In Progress (volatile)	0	Data 1: This parameter is used to indicate when any of the following parameters are being updated, and when the updates are completed. The bit is primarily provided to alert software than some other software or utility is in the process of making changes to the data. [7:2]: Reserved [1:0]: 00b: Set complete. If a system reset or transition to powered down state occurs while 'set in progress' is active, the BMC will go to the 'set complete' state. If rollback is implemented, going directly to 'set complete' without first doing a 'commit write' will cause any pending write data to be discarded. Point? 01b: Set in progress. This flag indicates that a utility or other software is presently doing writes to parameter data. It is a notification flag only, it is not a resource lock. The BMC does not provide any interlock mechanism that would prevent other software from writing parameter data while the set in progress bit is set. 10b, 11b: Reserved
SOL Enable	1	Byte 1: [7:1]: Reserved [0]: SOL Enable. Note, this controls whether the Activate SOL command will be accepted. Whether an SOL stream can be established is also dependent on the Access Mode and Authentication settings for the corresponding LAN channel. The enabled/disabled state and access mode settings for the serial/modem channel have no effect on SOL.. 1b = enable SOL 0b = disable SOL
SOL Authentication	2	Byte 1: SOL Authentication Enable [7]: 1b: Transmit and Accept SOL Packets with Authentication Type = none Note: Authenticated packets are also accepted. 0b: SOL Packets must be authenticated according to SOL Privilege Level, below. The packet will be authenticated as if it were an IPMI Message with that required privilege level. [6:4]: Reserved [3:0]: SOL Privilege Level. Sets the minimum operating privilege level that is required to be able to activate SOL using the Activate SOL command. 0h: Reserved 1h: reserved 2h: USER level 3h: OPERATOR level 4h: ADMINISTRATOR level 5h: OEM Proprietary level all other: Reserved

Parameter	#	Parameter Data (non-volatile unless otherwise noted) 1
Character Accumulate Interval & Character Send Threshold	3	Byte 1: Character Accumulate Interval in 5 ms increments. 1-based. This sets the typical amount of time that the BMC will wait before transmitting a partial SOL character data packet. (Where a partial packet is defined as a packet that has fewer characters to transmit than the number of characters specified by the Send Threshold - see next field in this parameter). A packet will not be sent 00h = reserved Byte 2: Character Send Threshold. 1-based. The BMC will automatically send an SOL character data packet containing this number of characters as soon as this number of characters (or greater) has been accepted from the host serial port into the BMC. This provides a mechanism to tune the buffer to reduce latency to when the first characters are received after an idle interval. In the degenerate case, setting this value to a ‘1’ would cause the BMC to send a packet as soon as the first character was received. This can be useful if the character accumulate interval is large. If the BMC is waiting for an acknowledge from the previous packet, it will ignore this threshold and continue to collect data until it has a full packet’s worth.
SOL Retry	4	Byte 1: Retry Count [7:3]: Reserved [2:0]: Retry count. 1-based. 0 = no retries after packet is transmitted. Packet will be dropped if no ACK/NACK received by time retries expire. Byte 2: Retry Interval. 1-based. Retry Interval in 10 ms increments. Sets the time that the BMC will wait before the first retry and the time between retries when sending SOL packets to the MM. 00h: Retries sent back-to-back
SOL Serial Settings (non-volatile)	5	Serial communication with the BMC when SOL is activated always occurs using 8 bits/character, no parity, 1 stop bit, and RTS/CTS (hardware) flow control. Data 1 [7:4]: Reserved [3:0]: Bit Rate. 1-5h = reserved. Support for bit rates other than 19.2 kbps is optional. The BMC must return an error completion if a requested bit rate is not supported. It is recommended that the parameter out-of-range (C9h) code be used for this situation. 0h: Use setting presently configured for IPMI over serial channel used by BIOS serial redirection. The setting will be used even if the access mode for the serial channel is set to ‘disabled’. 6h: 9600 bps 7h: 19.2 kbps 8h: 38.4 kbps 9h: 57.6 kbps Ah: 115.2 kbps
SOL Serial Settings (volatile)	6	Set volatile version of SOL Serial Settings. Data follows that for parameter #5.

5.16.5.11 SOL Commands

Table 5-9 - SOL Commands

Command	Request	Response
[01h] Activate SOL (Required privilege level determined by setting in SOL Configuration parameters)	Byte 1 [7]: Reserved [6:4]: Encryption Type 000b: None [3:2]: Reserved [1]: Reserved [0]: SOL Activation 1b: Activate SOL (no-op if SOL already activated) 0b: Deactivate SOL (no-op if SOL already deactivated) The BMC asserts RTS and DCD/DSR to the host serial port and starts accepting characters as soon as SOL is activated. Byte 2:5 Encryption Seed (eSeed) This is a 32-bit random number that the MM selects whenever the Activate SOL command is sent. The number can be used in the encryption algorithm	Byte 1 Completion Code Generic plus the following command-specific completion codes: 80h: SOL already active on another session (required). This will be returned any time an attempt is made to execute the Active SOL command and SOL is already activated for another session. 81h: SOL disabled Byte 2 [7]: SOL Authentication Setting 1b: SOL packets accepted with Authentication Type = none 0b: SOL packets must be authenticated per SOL Privilege (See Table 5.6 – SOL Configuration Data) [6:3]: Reserved [2:0]: SOL Encryption Type 000b: None Byte 3: SOL Seed Count This value is incremented each time a new Encryption Seed value is received in the SOL Activate request. The count is set back to 0h whenever SOL is deactivated. [7:4]: Reserved. [3:0]: Seed count Byte 4: Inbound Payload Size Maximum number of character data bytes BMC can accept in SOL packet from MM. 1-based. (1 = 1 byte) Byte 5: Outbound Payload Size Maximum number of character data bytes BMC may deliver in SOL packet to MM. 1-based (1 = 1 byte)

Command	Request	Response
[02h] SOL Activating (User Level required)	Byte 1: Session state [7:4]: Reserved [3:0]: Session state 0h: SOL Activating Byte 2: SOL version 10h: Version 1.0	Byte 1 Completion Code. (Console does not need to respond to this command. BMC will enter SOL immediately after sending the SOL Activating request.)
[03h] Set SOL Configuration (Admin Level required)	Byte 1: Reserved Byte 2: Parameter selector Byte 3:N Configuration parameter data	Byte 1: Completion Code. Generic, plus the following command-specific completion codes: 80h: Parameter not supported 81h: Attempt to set the set in progress value (in parameter #0) when not in the set complete state. (This completion code provides a way to recognize that another party has already claimed the parameters) 82h: Attempt to write read-only parameter 83h: Attempt to read write-only parameter
[04h] Get SOL Configuration (User Level required)	Byte 1 [7] 0b: Get parameter 1b: Get parameter revision only [6:0]: Channel Numbers Byte 2: Parameter selector Byte 3: Set Selector. Selects a particular set or block data under the given parameter selector. 00h if parameter does not use a set selector. Byte 4: Block Selector (00h if parameter does not require a block number)	Byte 1: Completion Code. Generic plus the following command-specific completion codes: 80h: Parameter not supported. Byte 2: [7:0] Parameter revision. Format: MSN = present revision. LSN = oldest revision parameter is backward compatible with. 11h for parameters in this specification Byte 3:N Parameter data.
[5h] Get SOL Status (User Level required)	–	Byte 1: Completion Code Byte 2: [7:1]: reserved [0]: 1b: SOL is activated 0b: SOL is deactivated Byte 3:6 Session ID Session ID for session that SOL is activated for. 00_00_00_00h if SOL is not activated

5.17 RS485 Interface

The RS485 Interface is the BladeServer private proprietary bus between the MM and the Processor Blades. It is a physically secure and internal bus that has no external chassis access. IPMI messages are sent on this bus encapsulated in RS485 packets.

5.17.1 RS485 Packet Formats

This section provides the data formats for the packets on the shared RS485 bus.

5.17.1.1 RS485 Discovery Packet

The RS485 Discovery Packet is sent by the MM once a Processor Blade is detected. Processor Blades are detected by a blade presence HW signal when a Processor Blade is hot-plugged into the chassis or power is applied to the chassis for the first time. It is also detected when the Processor Blade has been previously discovered and the Processor Blade performs a BMC reset or enters into an operational mode and sends an unsolicited RS485 packet with the encapsulated IPMI OEM *Blade State Change* message. *Blade State Change* messages are ignored if the Processor Blade has not been previously discovered via blade presence and performed a Processor Blade Chassis Initialization.

The Processor Blade will only listen to messages targeted to receive this message with the destination address set to its current slot number. For an N-wide blade, the slot number is the slot that contains the primary planar[3]. Once received for the proper slot address, the *RS485 Discover Response* packet should be sent as a reply.

Table 5-10 RS485 Discovery Format

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
sy	Go	da	sa	lh	ll	pt	sb	cr	pd	op	cm	tm	ch	cl	ed

sy	sync bytes. There may be many of these 0xFD always
go	"go code" 0xFC always
da	destination address blade slot number
sa	source address 0xE0 for MM originator
lh, ll	high/low bytes of byte-length minimum 0, maximum 1028.
pt	packet type 0xF3
sb	solicited byte 0x50

[3] Refer to **section 2.4 RS485 bus Interface** and **Table 2-8 - Processor Blade Slot Address Assignments** to determine which RS485 address this blade should respond to.

cr	correlator byte response packet must contain this correlator byte
pd	pad byte 0x00
op	Op byte 0x02
cm	command byte 0xF2 = discovery
tm	termination byte 0x0E = discovery termination byte
ch, cl	high/low bytes of checksum equals (zero minus (16-bit sum of destination byte to end byte with checksum bytes set to zero))
ed	end byte 0xFF always

5.17.1.2 RS485 Discovery Response Packet

The Processor Blade will send this *RS485 Discovery Response* packet upon receiving the *RS485 Discovery* packet.

Table 5-11 RS485 Discovery Response Format

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
sy	go	da	sa	lh	ll	pt	sb	cr	pd	op	cm	rs	ch1	ch2	ch3	ch4	tm	ch	cl	ed

sy	sync byte. There may be many of these 0xFD always
go	"go code" 0xFC always
da	destination address 0xE0 for MM
sa	source address blade slot number
lh, ll	high/low bytes of byte-length minimum 0, maximum 1028.
pt	packet type 0xF3
sb	solicited byte 0x50
cr	correlator byte required to be same correlator received in RS485 Discovery packet
pd	pad byte 0x00
op	Op byte 0x07
cm	command byte 0xF2 = Discovery
rs	reserved 0x00
ch1	char byte 1 0x49, 'I'
ch2	char byte 2 0x50, 'P'
ch3	char byte 3 0x4D, 'M'
ch4	char byte 4 0x49, 'I'
tm	termination byte 0xDF = discovery response termination byte
ch, cl	high/low bytes of checksum equals (zero minus (16-bit sum of destination byte to end byte with checksum bytes set to zero))
ed	end byte 0xFF always

5.17.1.3 RS485 Encapsulated IPMI Commands

This packet is used to send and receive encapsulated IPMI commands once the *RS485 Discovery Response* packet has been returned to the MM. Unsolicited messages, such as the *Blade State Change* may be sent before the *RS485 Discovery Packet Response* is sent. However, if the Processor Blade is not known (has not completed Processor Blade Chassis Initialization), these unsolicited messages may be ignored, though IPMI responses will be sent to prevent retransmissions.

Processor Blade BMC firmware implementers should note that the initial power on of the chassis could create congested traffic conditions on the RS485 bus if all blades in a fully populated chassis start sending unsolicited traffic to the MM when their standby power is first applied. This greatly increases the time it takes to fully perform Processor Blade Chassis Initialization to each Processor Blade.

Note: When power is first applied to the BMC the BMC shall not issue IPMI commands to the MM until after RS485 discovery and Processor Blade Chassis Initialization is complete. However, if Processor Blade Chassis Initialization has run then unsolicited traffic is allowed which permits the BMC to signal the MM that it has underwent a Blade State Change

Table 5-12 RS485 Encapsulated IPMI Command Format

	1	2	3	4	5	6	7	8	9	10 …		… n-2	n-1	n	n+1
sy	go	da	sa	Lh	Ll	pt	sb	cr	pd		pl (IPMI Command)		ch	cl	ed

sy	sync byte. There may be many of these 0xFD always
go	“go code” 0xFC always
da	destination address 0xE0 if destination is MM blade slot number if destination is a BMC blade
sa	source address 0xE0 if sent by MM blade slot number if originated by BMC blade
lh, ll	high/low bytes of byte-length minimum 0, maximum 1028.
pt	packet type 0xF5 = IPMI Command
sb	solicited byte 0x50 solicited or unsolictited messages from the MM to the BMC 0x50 solicited messages from the BMC to the MM 0x51 unsolicited messages from the BMC to the MM
cr	correlator byte If originator of command, user sets. If receiver, must be same correlator in response as that received in request.
pd	pad byte 0x00
pl	payload IPMI Command. Refer to section 5
ch, cl	high/low bytes of checksum equals (zero minus (16-bit sum of destination byte to end byte with checksum bytes set to zero))
ed	end byte 0xFF always

5.17.2 IPMI Command Format

The RS485 payload is the IPMI Message LAN format as presented in IPMI v1.5, section 12.4. The MM address is defined as 0x41.

Table 5-13 IPMI Request Format

rsAddr	Response addr (SA or sw ID)
NetFn/RsLUN	Response LUN
Checksum	Header checksum
rqAddr	Request addr (SA or sw ID)
rqSeq/rqLUN	Request sequence and LUN
CMD	Command
Data:N	Request data.
Checksum	IPMI command checksum

Table 5-14 IPMI Response Format

rqAddr	Request addr (SA or sw ID)
NetFn/RqLUN	BMC LUN = 00b (always) Request LUN
Checksum	Header checksum
rsAddr	Response addr (SA or sw ID)
rqSeq/rsLUN	Request sequence and Response LUN
CMD	Command
CC	Completion Code
Data:N	Response data.
Checksum	IPMI command checksum

The RS-485 interface from the blade to the MM is a shared chassis resource. With the exception of Blade State Change commands, the BMC must not allow SMS applications running on the Host to issue commands which directly make use of the RS-485 interface.

5.17.3 RS485 Bus Operation

Communications between the MM and the Processor Blade BMC occurs over a redundant shared RS485 bus that interconnects the 2 MM slots and the blade slot. Only one of the redundant busses is active at any given time. The RS485 bus is driven at a baud rate of 57600 and the UART format is 1 start bit, 8 data bits, 1 stop bit, and no parity.

The bus operates in a multi-master mode where any station can send a packet any time the bus is not already in use. It is symmetrical across all stations – meaning MMs and Processor Blade all use the same algorithms and timings to arbitrate, send, and receive packets. The protocol is not reliable in the sense that packet delivery is not guaranteed: some packets may be dropped and some may be duplicated due to retries. If reliability is required, it is the responsibility of the MM to institute it through sequence numbers or other mechanisms embedded in the packet data payload.

The generic term ‘station’ will be used in the description to refer to either the BMC or MM endpoint of RS485 communication.

5.17.4 RS485 Address Determination

Each Processor Blade station has a unique fixed 8-bit address on the RS485 bus associated with the slot in which it resides. The address range is from (x01-x0E) and the address is based on the 4 slot address pins routed to the Processor Blade connector. If a Processor Blade spans more than one slot then the slot ID assigned to the blade unit is the lowest (Non-Telco chassis) / highest (Telco chassis) number slot ID that the entire blade unit occupies. The active MM has a fixed address of 0xE0. Processor Blades must only accept packets from the MM address of 0xE0 and discard packets from other Processor Blade addresses even if addressed to them. In addition the Processor Blade must only address the MM using the address 0xE0.

5.17.5 Redundant RS485 Bus Selection and Management Module Redundancy

The BMC connects to 2 RS485 interfaces. This allows them to communicate with whichever of the two redundant MM’s for purposes of redundancy. The BMC is responsible for activating the correct interface based on the state of the “Bus Select A” and “Bus Select B” signals. The BMC must ignore RS485 bus activity or in other words de-select itself from the RS485 bus when the Bus Select signals are 00b or 11b, since this indicates either a MM fault, absence of a MM (or any MM), or MM initialization is in progress. The blade station is required to maintain status of these signals in case the MM is inactive or disabled. This is critical because in the event of a MM failure or removal it should not affect operation of the Processor Blade

Bus Select B	Bus Select A	Bus State
0	0	De-select Bus (Reserved)
0	1	Bus A Active
1	0	Bus B Active
1	1	De-select Bus

5.17.6 RS485 Bus De-gating and BMC Reset

In a situation that the MM determines that the Processor Blade may be causing an overall RS485 bus problem, the Processor Blade hardware must implement support to allow the MM to de-gate the Processor Blade from the RS485 bus. The MM will set a hardware signal on the Processor Blade midplane connector that will immediately de-gate the blade from the bus.

In addition, for recovery reasons the Processor Blade may be signaled to not only de-gate from the RS485 interface but additionally perform a BMC reset. The MM will set a hardware signals on the Processor Blade midplane connector that will immediately de-gate the blade from the RS485 bus and also reset the BMC.

A summary of the use of the signals for de-gating and BMC reset on the Processor Blade connection to the midplane is as follows (Note: the signals are active low):

BMC_RS_485_RESET[A/B}_N	RS_485_DEGATE[A/B]_N	Processor Blade Action
0	0	De-gate RS485 bus and Reset BMC
0	1	No Action performed
1	0	De-gate RS485 bus
1	1	No Action performed
Note: Signal descriptions with [A/B]indicate these signals are redundant and the signals in use are determined by the state of the ‘Bus Select A/B’.		

5.17.7 Packet Control Bytes

A sending station on the bus uses five different control bytes to communicate the packet state to the receiving stations. In order to insure that these bytes cannot be encountered in a packet, a sixth control byte is used as an escape. When a packet byte that matches one of the control bytes is encountered, the escape byte is sent first followed by the inverse of the packet byte. This insures none of the other control bytes will ever be received from the RS485 bus while a valid packet is being received. The control bytes are defined as follows:

Control Byte	Value	Description
Unused	0xfa	
ESC	0xfb	Invert next packet byte before storing and calculating checksum
GO	0xfc	First byte of a normal packet
SYNC	0xfd	A station is arbitrating to use the bus
NO_GO	0xfe	Abort the receive that is in progress
END	0xff	Last byte of a packet

5.17.8 Packet Format

Each packet on the RS485 bus consists of a basic RS485 packet with an IPMI request or response embedded within it, as identified by the Type Byte (set to F5h) in the RS-485 packet header. See section 5.17.1.3 ‘RS485 Encapsulated IPMI Commands’ for the packet formats.

5.17.8.1 Format of Embedded IPMI Message

The embedded IPMI message follows the IPMI LAN message format as described in the *IPMI v1.5 Specification.* The processor blade stations and the MM use the following IPMI requester/responder addresses in these messages:

- The IPMI address for all processor blades is 0x20
- The IPMI address for the MM is 0x41. This is derived using the standard SW ID of 0x20 for System Management SW, given in the *System Software ID* table of the *IPMI v1.5 Specification.*

5.17.9 Bus Operation

The physical RS485 bus is a shared bus and allows any station on the bus to transmit at any time. In order to communicate meaningfully, however, only one station can do so at a time. To ensure this, before sending a station must first arbitrate for the exclusive right to transmit on it. The one that wins arbitration sends its packet while all other stations receive it. Once the send is complete, the bus again becomes free and arbitration can occur for the right to send the next packet.

When two or more stations transmit simultaneously on the bus it is called a bus collision. Collisions are expected during arbitration (though the protocol attempts to minimize them) but should not occur during packet transmission. To detect collisions, whenever a station transmits it must also keep its receiver active and verify that what it tried to send was actually seen correctly on the bus. This is true both during arbitration (when the station is sending 8 SYNC bytes) and during packet transmission. Each station must also monitor its UART for framing errors, another indication of bus collisions. There is a very small chance of an undetected collision where two stations try to send the same byte at approximately the same time. During arbitration when this is most apt to occur, each station waits a random delay between sending so this condition does not persist.

The physical bus can be in any of three physical states: idle, arbitrating, and packet transmission. Each station goes through a more complicated set of internal states based on both the physical bus state and its role (sending or receiving). Transitions between these internal station states are driven by the reception of control bytes, data bytes, or errors due to bus collision. Timeouts and local events (packet ready to send, packet done being sent) also may cause transitions. The figure below shows the station states and the transitions between them. When the bus is idle, all stations are in the IDLE state. During arbitration, a station will either be in the IDLE state if they don't have anything to send, the SEND_SYNC state (actively arbitrating), or else the BACKOFF state (waiting to try arbitrating again). During packet transmission the station is in either the RECEIVE or SEND_PKT state, depending on if it won arbitration or not.

The MM should have priority getting the bus over blades sending unsolicited requests. For example if multiple blades try to send alerts to the MM at once, the MM may not be able to get the bus often enough to make much progress handling the first alert before the other blades timeout and resend. To ensure the MM has priority, when the bus becomes free after a packet is sent stations who have solicited packets to send (MM requests & responses, blade responses) immediately enter the SEND_SYNC state and begin arbitrating. Stations wishing to send unsolicited packets (blade requests) must first enter the BACKOFF state and wait before arbitrating. This gives the solicited senders first chance to win the bus.

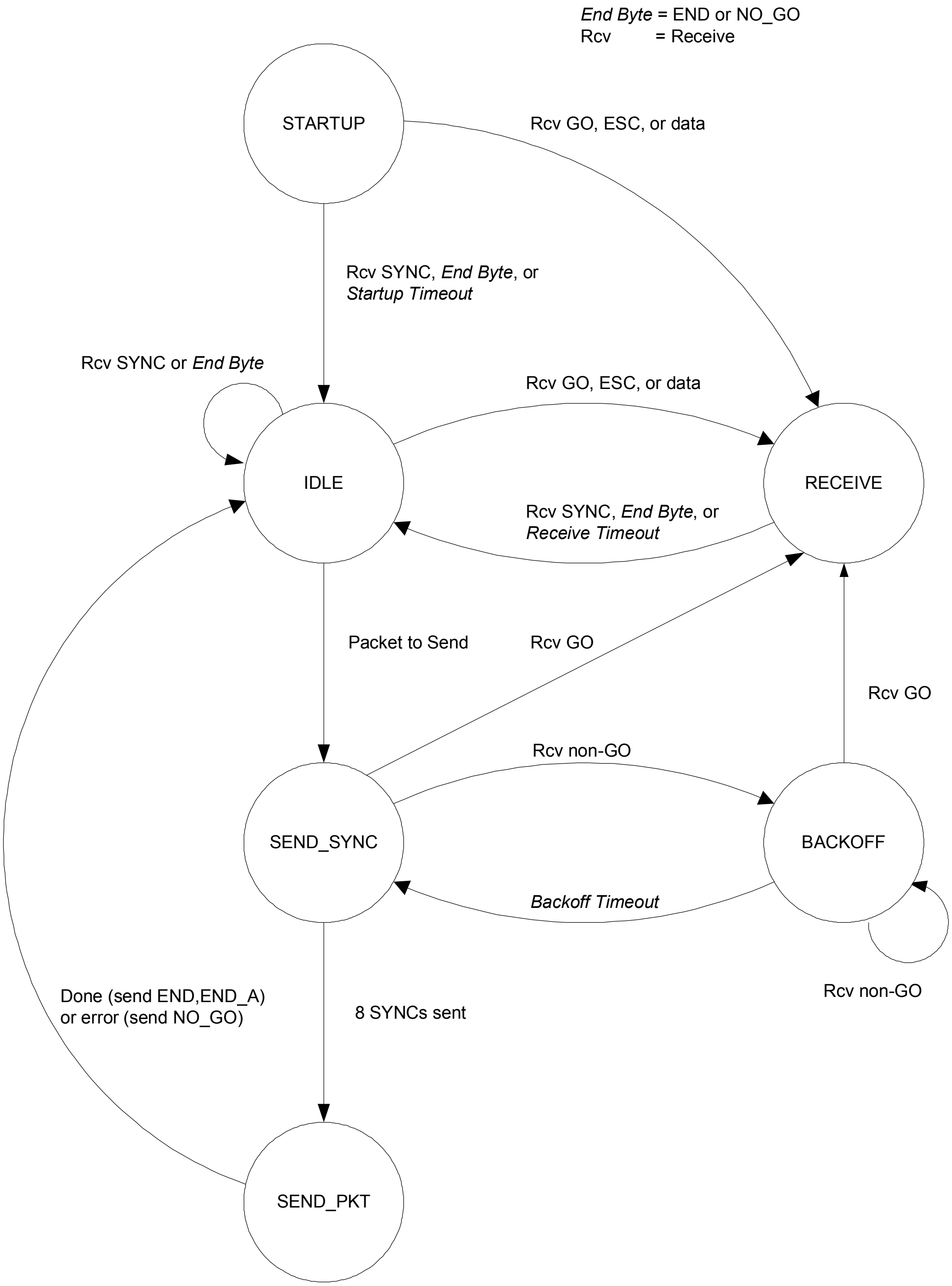


Figure 5-18 – Internal station states and transitions

5.17.9.1 STARTUP State

Whenever the station enables its bus driver, it first enters the STARTUP state. The station does not know the current state of the bus so it must monitor it for a time to ensure it is idle before trying to send. If the station receives a SYNC byte or one of the packet end markers (END or NO_GO) then it knows packet transmission has just completed and it goes to the IDLE state. If it receives a GO, ESC, or data byte it knows packet transmission is occurring and goes to the RECEIVE state to receive the rest of the packet. Otherwise, it times out (*Startup Timeout*, 500mS) and goes to the IDLE state.

5.17.9.2 IDLE State

The station has nothing to send and is not receiving. Other stations may be arbitrating to send.

5.17.9.3 RECEIVE State

The station is receiving a packet sent by another station. It enters the RECEIVE state whenever it sees a GO byte and leaves it when it either receives an end-of-packet indicator or else the *Receive Timeout* (2 seconds) is exceeded. End-of-packet indicators include the END byte (successful termination) and the NO_GO or SYNC bytes (error termination). Receiving a SYNC byte indicates another station thinks the bus is idle and has begun arbitrating

The packet is thrown away unless it was correctly framed with GO & END bytes, the checksum verifies, the length in the packet matches the actual length, there were no receive errors during the packet, and the packet is addressed to the local station.

5.17.9.4 SEND_SYNC State

To win arbitration, the station must successfully send 8 SYNC bytes without a collision. Since all stations are free to transmit during bus arbitration, it is important that each station get off the bus as quickly as possible to decrease collisions. When sending a SYNC byte, the station must enable its transmitter, send the byte, and disable its transmitter again as quickly as possible. The maximum time this is allowed to take is defined as the *SYNC Transmit Time* (500uS).

There is a random delay (the *SYNC Delay Time*) before each SYNC byte is sent to decrease the chance of collisions and ensure that if two stations simultaneously send the first SYNC (an undetectable collision), they will be unlikely to send the remaining SYNCs simultaneously. The *SYNC Delay Time* is between 1 and 8 *SYNC Transmit Time* intervals (500uS – 4mS).

5.17.9.5 BACKOFF State

If a collision is detected after sending a SYNC byte, the station enters the BACKOFF state and waits a random time (the *Backoff Timeout)* before trying to arbitrate again. This reduces the chance of consecutive collisions by allowing other stations onto the bus. During this timeout, if it sees a GO byte, it knows another station has won arbitration and has begun sending a packet so the local station goes to the RECEIVE state. Once the packet is received and the bus is idle, it can re-arbitrate by entering the SEND_SYNC state. If it receives anything else (or sees an error), it assumes another collision has taken place, computes another random *Backoff Timeout*, and waits before trying to arbitrate again.

The *Backoff Timeout* is a random delay between 3mS and 18.5mS. It is recommended that it be calculated as the minimum value(of *Backoff Timeout)* plus a random number (0-31) of *SYNC Transmit Time* intervals of 500uS each.

5.17.9.6 SEND_PKT State

Once the station has won arbitration it enters the SEND_PKT state and sends the GO byte, the packet contents, and then finishes with an END byte. If any of the data in the packet matches the value of a control byte, the station first sends the ESC byte and then the inverted data byte. If it sees errors during the send process (as indicated by a mismatch between what it sends and what it receives as actually on the bus), it aborts packet transmission and sends a NO_GO byte telling other stations the packet should be dropped and that the bus is now free. The station must attempt to send the packet a minimum of ten times (*Send Attempts*) before aborting the send.

To help prevent UART overruns on the receiving stations, the sending station must ensure a minimum of at least *Transmit Delay Time* between each byte sent. The maximum time between bytes is constrained by getting the entire packet sent within the *Receive Timeout* time.

During the entire SEND_PKT state the station can be actively driving the bus by having its transmitter enabled. Once the last byte has been sent, the sender must disable its transmitter before the shortest *SYNC Delay Time* occurs so it doesn't interfere with arbitration of the next packet.

5.17.10 Bus Error Handling

Bus errors are detected as either UART framing errors(i.e a bus collision as described above), UART overruns (indicating the local station dropped bytes and may be out of sync with the bus state), or else during transmission when the echo received doesn't match what was sent.

Bus errors are handled differently depending on what state the station is in. In the IDLE or STARTUP state they are ignored. During the SEND_SYNC and BACKOFF states they are assumed to be due to another station arbitrating so the local station backs off and re-starts it *Backoff Timeout* timer. During the RECEIVE state it indicates an error receiving the packet so the packet is thrown away once it is received. Finally, if a bus error occurs during the SEND_PKT state it indicates there is confusion about the bus state so the local station aborts the packet by sending a NO_GO byte and then re-arbitrating to resend the packet. This includes errors that occur during the transmission of the final END byte.

5.17.11 Command Response Timeout and Retries

When a station sends a request packet to another station it waits for the corresponding response packet. If it does not receive the response within a certain timeout (*Response Timeout*), it resends the request. The sending station must send the request a minimum of three times (*Command Attempts*) before aborting. The *Response Timeout must* be at least 100mS. Note, that these retries are in addition to the bus error retries, so the maximum total number of attempts is *Command Attempts* multiplied by *Send Attempts*, though this is extremely unlikely.

5.17.12 Timing and Retry Values

Timer/Retry Counter	Value	Type	Description
Startup Timeout	500 msec	Min	When starting up, time to watch the bus before deciding it is idle
Receive Timeout	2sec	Min	How long to wait before giving up on receiving a packet and assuming the bus is really idle
SYNC Transmit Time	500 usec	Max	How long a station can have its transmitter enabled while sending a SYNC byte
SYNC Delay Time	500 usec to 4 msec	Random Interval	How long to delay (and monitor the bus) before sending out a SYNC byte during arbitration. Must be randomly chosen, recommended as 1 to 8 *SYNC Transmit Times*
Backoff Timeout	3 msec to 18.5 msec	Random Interval	How long after a failed arbitration attempt to backoff before attempting another one. Must be randomly chosen, recommended as 3mS plus 0-31 *SYNC Transmit Times*
Transmit Delay Time	50 usec	Min	Time between bytes during packet transmission
Response Timeout	100 msec	Min	Time to wait after successfully sending an IPMI request to receive the response before retrying
Command Retry Attempts	3	Min	Number of times to send a command request
Packet Send Retry Attempts	10/15	Min/Max	Number of times to try sending a packet

5.18 BMC Communication Interface to BIOS

The following section describes an example implementation between a BMC in a blade and firmware running on the host processor of the blade.

Host firmware such as BIOS communicates with the BMC for various functions utilizing the KCS (Keyboard Control Style) interface[2]. The BIOS will initialize the South-Bridge of the processor blade to enable the LPC decode registers so that the KCS I/O address space is available for BMC communication. The BIOS will insure this initialization is done as early as possible in POST to provide the ability for BIOS to send and receive information to and from the BMC. The BIOS must follow the KCS request/response interface format as specified in the IPMI specification for BMCs that implement an IPMI interface to the host. For additional information on the KCS interface see the IPMI Specification. Figure 5-19 is a summary of the typical hardware interfaces between the Processor BIOS, BMC and the MM.

The BMC has the ability to interrupt the BIOS using a Systems Management Interrupt (SMI). SMI can be used for example to handle PFA's detection on hardware components. See section 5.18.4 for additional information.

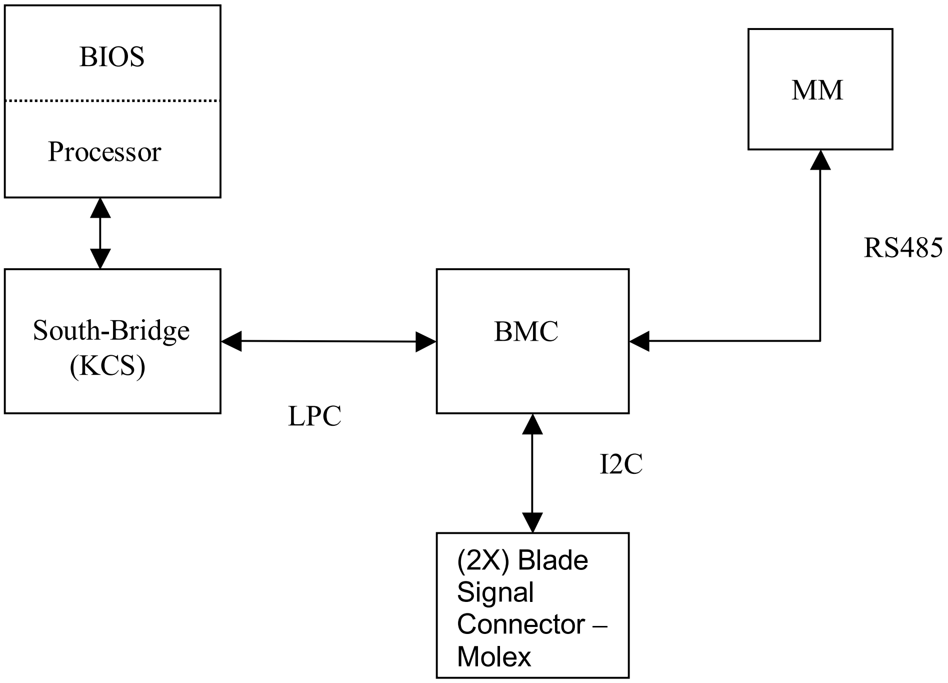


Figure 5-20 - BIOS Interface to BMC

5.18.1 BladeServer System Management Function Requirements

In order for a ServerBlade to function correctly in the BladeServer chassis certain basic functions must be provided by the BIOS. These functions are summarized below and discussed further in the following sections:

- Must provide a POST and initialize the hardware of the processor blade.
- Must provide an interface to exchange information with the BMC.
- Must provide some level of watchdogs (or timers) for error detection.
- If failures or errors are detected by the BIOS, BIOS must send error log information to the BMC which can forward this information to the MM in order for a system administrator to determine why the processor blade has an error.
- Must provide the MM with POST progress information
- Must provide power and thermal information on the processor blade that cannot be obtained directly by the BMC.
- Must be able to read and write various fields of the VPD for processor blade functions and systems management, see section 5.18.3.
- Must provide for and handle a BMC initiated interrupt mechanism (or an equivalent method to interrupt the BMC). For example on a IA-32 compute blade, a Systems Management Interrupt (SMI) may be asserted by the BMC for this purpose.

Prior to a Processor Blade being powered on the MM and BMC will have preformed the following:

- The MM would have granted power permission as described in the section 5.5 Processor Blade Chassis Initialization.
- The MM will have also determined from the BMC the ‘pre-boot inventory’ amount of required power that the blade requires to power on and execute POST. For additional information see section 5.7.5.

IPMI provides for the ability for the BMC to support various watchdog timers that can be utilized for a number of system timeout functions by both systems management software and BIOS. The BIOS and BMC is required to provide some minimum watchdog support and some optional watchdog support.

- An *Automatic BIOS Recovery (ABR) Watchdog* is required. The BMC should start an *ABR Watchdog* prior to POST being initiated as the processor blade enters PD2. For recovery reasons, a typical blade BIOS flash structure should partition the flash to contain two banks, with each bank containing a BIOS image, one primary and one backup. In addition the BMC requires the ability for a hardware mechanism to select which image is used for POST. An ABR watchdog mechanism allows the BMC to recover a processor blade from a corrupted BIOS image. The BIOS may become corrupted for example when a flash update of the BIOS is in progress and a power failure occurs. Utilizing the ABR watchdog, the BMC can determine if POST has failed to complete due to a corrupted image and when the watchdog triggers the BMC can toggle which bank of the flash to use on the next reset of the processor blade.

  During POST the BIOS may have reasons to disable the ABR for a period of time, for example during a memory test. Disabling during the time that memory is being validated will insure that an ABR watchdog will not trigger. This would have the effect of causing the BMC to switch to the backup image when a failure in POST has occurred due to a memory problem (for example no memory is installed).

- Optional watchdog support may be provided by the BIOS. As mentioned previously this may be configurable via a setup utility to allow the administrator to enable and disable the watchdogs. Some typical examples of watchdogs provided may be a ‘BIOS/POST watchdog’ and ‘OS Loader Watchdog’.

During POST the BIOS may need to generate status (progress) indications or events for error conditions in order for the system administrator to utilize the MM CEL to determine the processor blade state as POST is executed. The two mechanisms the BIOS may use for this are as follows:

- IPMI OEM *Directed Event String* command utilizing the functions of *‘Chassis Event Log String’* and *‘BIOS Progress Alert’*. When BIOS uses this command the information will be saved to the MM’s CEL. For additional information on use of this command see the section 5.8.1.
- IPMI *‘Platform Event Message’* command using a sensor type of ‘System *Firmware Progress’*. When BIOS uses this command the information will be saved to both the processor blade’s SEL and the MM’s CEL.

During POST the BIOS may need to turn on various processor blade board LEDs. For example if BIOS detected an error in a memory DIMM the *‘DIMM Fault LED’* would be required to be set. To set the board LEDs the BIOS will use the IPMI OEM command ‘LED Set’. For additional information on what conditions BIOS will need to set the LEDs see section 5.13.1 Board Failure LEDs on Processor Blade.

5.18.2 BIOS and POST Initialization Sequence

The process of POST and Processor Blade Initialization refers to the interaction between the BIOS and BMC. This process is executed upon any of the following events:

- On power up of the processor blade (when transitioning to PD2)
- When a processor blade reset occurs (for example a reset initiated by the MM)

Figure 5.4 represents a possible sequence of interactions which occurs upon POST being run on the Processor Blade. Message flows are between the BIOS and the BMC over the KCS interface.

Note: Figure 5.4 does not generally show the IPMI responses from the BMC. Except where a response needs to be described, the response flows are not shown. A failure or timeout response to an IPMI command issued by the BIOS must result in at least one retry by the BIOS. A rejected response may result in a failure to allow a blade to complete POST and load the OS.

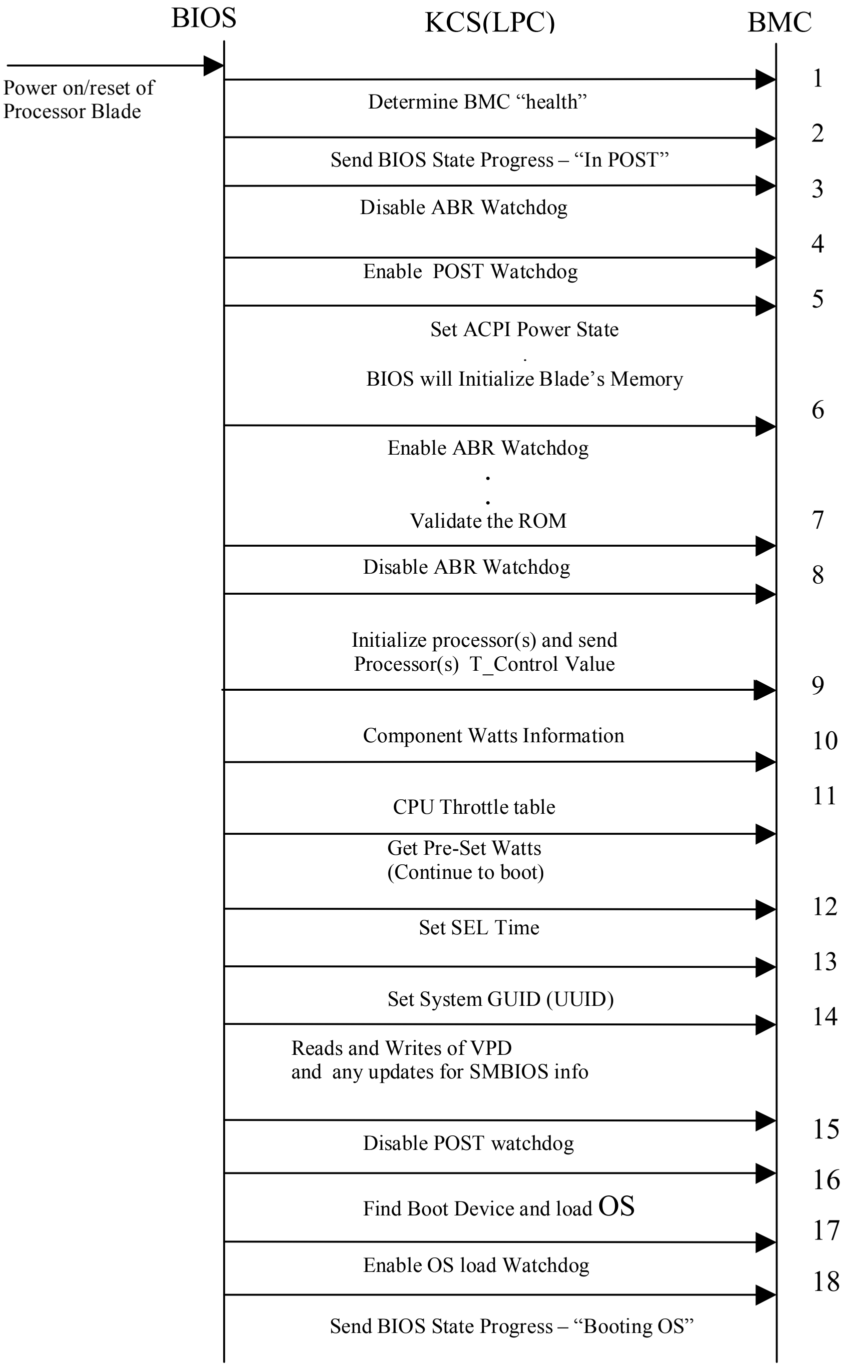

BIOS
KCS(LPC)
BMC
Power on/reset of Processor Blade
1
Determine BMC "health"
2
Send BIOS State Progress – "In POST"
3
Disable ABR Watchdog
4
Enable POST Watchdog
5
Set ACPI Power State
BIOS will Initialize Blade's Memory
6
Enable ABR Watchdog
Validate the ROM
7
Disable ABR Watchdog
8
Initialize processor(s) and send Processor(s) T_Control Value
9
Component Watts Information
10
11
CPU Throttle table
Get Pre-Set Watts (Continue to boot)
12
Set SEL Time
13
Set System GUID (UUID)
14
Reads and Writes of VPD and any updates for SMBIOS info
15
Disable POST watchdog
16
Find Boot Device and load OS
17
Enable OS load Watchdog
18
Send BIOS State Progress – "Booting OS"

Figure 5-21 - Example: Processor Blade POST and BMC Interaction

The steps of Figure 5.21 are described below.

1. The Processor Blade is powered on. The BIOS should determine the 'health' of the BMC by issuing the IPMI command *'Get Self Test Results'*. If the BMC fails to respond or

indicates that it is in a failure state the BIOS should abort POST and attempt to notify the administrator.

2. BIOS will send the BMC an IPMI OEM *Directed Event String* command that indicates that POST has been initiated. The DES command will contain the '*BIOS State Progress'* subcommand with the '*Boot State'* set to '*System in POST'*. This command will be forwarded by the BMC to the MM. See section 5.8.1 for additional information.
3. BIOS may disable the ABR watchdog in preparation for memory initialization, using the OEM IPMI command '*Configure Boot Block Recovery'*.
4. BIOS if required will issue the IPMI command *'Set Watchdog Timer'* to enable the timer for *'BIOS/POST'* watchdog.
5. BIOS will issue the IPMI command *'Set ACPI Power State'* with *'Power State'* set to *'Working'* to allow systems management software to determine the current ACPI power state of the processor blade. In addition BIOS initializes the memory on the Processor Blade.
6. BIOS may re-enable ABR watchdog using the OEM IPMI command 'Configure *Boot Block Recovery'*. In addition BIOS determines if the BIOS flash image is valid. BIOS could determine this for example by calculating a checksum for the entire image. If the flash image is not valid then BIOS will send a 'Platform Even Message' with the event information 'Firmware (BIOS) ROM corruption detected' and hang in POST. A POST hang will cause the BMC to detect an ABR watchdog timeout and handle this watchdog as described previously.
7. BIOS may disable the ABR watchdog using the OEM IPMI command 'Configure *Boot Block Recovery'*.
8. BIOS will continue initialization of the hardware and determine the processor(s) configuration. For each processor the BIOS will determine the Tcontrol (Tc) value and send this information to the BMC using the IPMI OEM command *'Processor T_Control Information'*. For additional information on the use of Tc see the section 5.7.5.
9. BIOS will take an inventory of all the installed processor(s) and memory DIMM(s) and then send to the BMC power information for processor(s) and memory DIMM(s) using the OEM IPMI command *'Watts Inventory Information'*. For additional information see the section 5.7.5.
10. BIOS will send throttle information to the BMC for each processor using the OEM IPMI command *'CPU Throttle Table Information'*. For additional information see the section 5.7.5.
11. The above messages to the BMC (step 8-10) have provided information to allow for the BMC and MM to make a determination if the processor blade is allowed to continue POST and eventually boot into the OS. Therefore the BIOS must now send the OEM IPMI command 'Get Pre-Set Watts'. The BIOS will look at the response to this command and determine if permission has either been granted or not granted to continue booting the blade. If permission is not granted, the BIOS must halt the boot process and the BMC will subsequently power-off the Processor Blade.
12. BIOS is required to send the IPMI Command *'Set SEL Time'*.
13. BIOS is required to send a 'Globally Unique ID (GUID)' to the BMC using the OEM IPMI Command *'Set System GUID'*. The GUID data in the command is made up of the'Universal Unique ID' (UUID) that is a field within the VPD of the Processor Blade. For more information on VPD see the 'Base Specification for VPD'.
14. BIOS is required to use VPD information for the following:
    a. Write the Processor Blades BIOS firmware revision information in the 'Code Level Version' fields of VPD. The OEM IPMI command 'Write Legacy Format VPD' is used for this purpose. For more information on VPD see the 'Base Specification for VPD'.

b. Utilize information about the KCS interface as well as information read from the VPD to set the appropriate fields of various *System Management BIOS (SMBIOS)* records. Various systems management software running on the processor blade may require information about the IPMI system interfaces, interrupts and chassis information. This information is typically stored in *System Management BIOS (*SMBIOS) records. These records include '*IPMI Device Information Record', 'Base Board Information Record' and 'System Enclosure or Chassis Record'.* For additional information see both the IPMI and SMBIOS specification. (IPMI - Intelligent Platform Management Interface Specification v1.5, System Management BIOS Reference Specification).

15. The POST phase of BIOS is complete therefore if a 'BIOS/POST watchdog was started, BIOS will now issue the IPMI command '*Set Watchdog Timer'* to disable the timer for *'BIOS/POST'* watchdog.
16. The BIOS will now determine the boot sequence based on the configured boot device list. The boot sequence is the list of devices and order in which to attempt to boot the OS from. They typically include devices such as CD, floppy, hard drive, Network, USB Flash Drive. The BIOS as well as the MM can configure the boot device list. The BIOS will therefore read the VPD field *'Boot Path'* using the OEM IPMI command *'Read Legacy Format VPD'* to determine the possible devices to load the OS from. For more information on VPD see the 'Base Specification for VPD'. The BIOS will determine if a valid boot device exists and then load the OS.
17. BIOS if required will issue the IPMI command *'Set Watchdog Timer'* to enable the timer for *'OS Load'* watchdog. Typically an OS application or driver will need to disable this watchdog using the same IPMI command once the OS load has completed.
18. BIOS will send the BMC an IPMI OEM *Directed Event String* command that indicates that POST has been initiated. The DES command will contain the '*BIOS State Progress'* subcommand with the '*Boot State'* set to '*Booting OS'*. This command will be forwarded by the BMC to the MM. See section 5.8.1.1 for additional information.

5.18.3 BIOS use of VPD

The BIOS on the processor blade is required to utilize the VPD that is contained in the EEPROM of the blade for information exchange between the BMC and the MM. The specific fields used are provided in the table below and are noted in section 5.18.1. The OEM IPMI command *'Read/Write Legacy Format VPD'* is used for this purpose. For more information on VPD see the 'Base Specification for VPD'.

VPD Block	Field name	Usage
Fixed Block Manufacturing Data	Machine Type Model	(1) If BIOS allows this field to be updated from an update utility then the VPD data also needs to be written. A update to this VPD block will cause the BMC to initiate a OEM IPMI command *'Blade State Change'* indicating '*Main Planar VPD Modified*' to the MM.

VPD Block	Field name	Usage
	Machine Serial Number	Treated the same as (1)
	Asset ID	Treated the same as (1)
	UUID	Treated the same as (1)
	MAC Addresses(s)	BIOS will update this field with the processor blade's on board NIC(s) MAC address(s) (i.e. on planar board or soldered down NIC(s))
Dynamic Block Controller Area	Code Level 1 Version BIOS	(2) When the BIOS firmware is updated, then the VPD information needs to be written
	Code Level 2 version DIAGS	Same as (2), when processor blade contains a ROM based DIAGS.
Dynamic Block Systems Management Area	Boot path	Used to determine the boot sequence for the blade. In addition, must be updated with boot device information from either the BIOS or MM.
	History pointer and History Log	Will be updated by the MM and used by BIOS to read chassis level information. For example can be used to obtain the current SeverBlade chassis slot and chassis UUID.

5.18.4 BIOS and Predictive Failure Analysis Handling

Processor Blades may report certain Predictive Failure Analysis (PFA) events to the MM using the OEM command *Directed Event String* (DES).

If the Processor Blade is in the power-on state. the BIOS will periodically collect PFA information about certain error conditions. For example on a IA-32 compute blade, an SMI signal may be asserted by the BMC (infrequently (for example once an hour) for system performance reasons) to support PFA detection by the system BIOS. The SMI will request the BIOS to check for any errors that require a PFA. If any errors are found then BIOS will log an error to the MM using the IPMI OEM command *Directed Event String* (DES). In addition the boards LEDs are set to indicate what component(s) may be failing.

5.19 Processor Blade IPMI SDR Requirements

Blades are required to support IPMI Version 1.5 and support the OEM commands with associated flows as defined by this IPMI OEM document. The intent of this section is to further clarify the requirements placed on the Processor Blade in the usage of the various standard SDRs provided by implementation of the IPMI specification.

5.19.1 IPMI Standard Event Processing

The MM firmware will process BMC SEL events sent using a DEM in an IPMI standard method. SEL events and sensor SDRs are parsed according to IPMI standard rules. The data and severity of the processed events shown on the MM user interfaces and the alerts sent to an external management application will depend on how the BMC on the Processor Blade product implements events. Therefore the BMC has a large amount of control on how events with associated information are presented to customers. The MM will include SDR ID strings to assist identifying events to product descriptions. The two methods of event support on the MM include ‘Sensor Specific’ and ‘Generic Discrete’.

The MM IPMI Standard Event Handling firmware has strings associated with all applicable IPMI standard Entity IDs, Sensor Types, Sensor Specific Offsets, and Sensor Specific Event Data 1 and 2 values. The MM CEL string associated with an IPMI standard event will be constructed from the event’s characteristics. The MM will also include the SDR string in the CEL entry.

Processor Blades should follow the guidelines listed below to work with the MM IPMI Standard Event Handling firmware.

1. Whenever possible, the BMC should use the appropriate sensor specific Event/Reading Type Codes. This will give the customer more detailed information than a Generic Discrete Event.
2. OEM Sensor Types for Sensor Specific Event/Reading Type Code or OEM Event/Reading Type Codes are not supported.
3. If the MM firmware receives a event that indicates a OEM sensor type then the MM firmware will indicate a ‘OEM Sensor’ in the CEL entry.
4. All events with MM severity level of Warning or Critical (defined either by the MM or the BMC) will create alarms and will be tracked by the MM user interface. These alarms will stay active on the MM user interfaces until the event is de-asserted by the BMC. If the event is never de-asserted by the BMC, the alarm status for a given Processor Blade will remain warning or critical. Please see the next section for more information on event severity.
5. The BMC should not report a ‘recover’ event as and ‘unrecoverable’ event. If the event can be recovered, the BMC should use a recoverable event description.
6. For an IPMI discrete event that causes the MM to create an event with severity level of Warning or Critical, the MM must receive a de-assertion of the same offset in order to clear the event. The MM will not clear an active event based on the assertion of a new offset that changes the current state to a less severe state. For example:
    a. The MM will create a Critical event when it receives an assertion for Event/Reading Type Code 0Bh with a offset of 05h which is defined as “Redundancy States - Non-redundant: Insufficient resources to maintain normal operation”. The MM will only clear this event if it receives a de-assertion for the same offset. For this sensor, the IPMI standard defines offset 07h as “Redundancy Degraded from Non-redundant”. If the MM receives an assertion for offset 07h, the MM will create another event with severity level of Warning. From the IPMI standard definition of the offsets, the first event is cleared by the second event. But the MM will have both events active.

.

5.19.1.1 Event Severity

The MM provides for support for three severity levels, defined as 'Critical', 'Warning' and 'Informational'.

The MM has assigned a default severity level for every standard IPMI event based on the event description. For Generic Discrete Events, the severity is not implicit by the offset descriptions and therefore has been defaulted to severity level of "Informational". For example, Event/Reading Type Code 03h, digital discrete, with offset 00h – "State De-asserted" or Offset 01h – "State Asserted" the default severity level is "Informational".

If desired, the BMC may increase the severity of any discrete event by providing the severity as part of Event Data 2 field as defined by the IPMI standard. The translation of IPMI severity state to MM severity state is listed below.

Table 5-15

IPMI Severity Event States Offsets (Read Type 0x07)	MM Event Severity
0x00 - Transition to OK	Informational
0x01 - Transition to non-critical from OK	Warning
0x02 - Transition to critical from less severe	Critical
0x03 - Transition to non-recoverable from less severe	Critical
0x04 - Transition to non-critical from more severe	Warning
0x05 - Transition to critical from non-recoverable	Critical
0x06 - Transition to non-recoverable	Critical
0x07 - Monitor	Informational
0x08 - Informational	Informational

5.19.1.2 Event Strings

The Event strings that are added to the MM's CEL and sent to an external application are built in an IPMI standard manner. This includes the strings provided in the SDR's supported by the BMC on the Processor Blade. To assist in providing as much information to the customer as possible, especially in identifying the exact condition and the associated entity, the MM takes advantage of the Sensor SDR's, FRU Locator Record SDR's, Entity Association Record and Device Relative Entity Association Record SDR's (see section 5.11 FRU Data for additional information).

The lack of FRU Locator Records and Entity Association Records will not prevent the MM from processing standard IPMI events. However, the absence of these SDR records translates to less information to the customer and the Processor Blade would appear as deficient in functionality. Therefore support for FRU Locator Records and Entity Association Records is a requirement.

5.19.1.3 Sensor SDRs.

Sensor SDRs are mandatory requirements for standard event processing. The SDR provides for a *Sensor 'ID' String* and is used to assist the customer in identifying the specific details of an event.

5.19.1.4 FRU Locator Record SDR

The MM will attempt to find the FRU the sensor is associated to. This is done by searching the FRU Locator Records (using Entity and Instance of sensor's SDR). If found the Sensor ID string may be taken advantage of. This record is then used to obtain containment information (see section 5.11 FRU Data for additional information).

5.19.1.5 Entity Association Record and Device Relative Entity Association Record SDR

The Entity Association Record is used to provide logical containment information. The BMC's IPMI OEM support extends this definition to physical containment as well. Entity Association Records will be parsed to build containment information into customer event strings, so that FRU faults or information may be clearly identified as to location (see section 5.11 FRU Data for additional information).

For improved topology support, the BMC's OEM support defines further options for the Entity Association Records and Device-relative Entity Association Records so that physical containers may be supported as well.

5.20 Alarms

All events that are defined by MM as 'Warning' or 'Critical' severity will cause an *'alarm'*. An alarm is an indication the MM provides to the customer of the overall BladeServer chassis health. An alarm will remain active until the BMC sends an event to de-assert the alarm. Upon recovery of an event the blades are required to send de-assert events to remove the alarm indication.

5.21 Using Standard Events versus DES

As previously mentioned, standard events must be used where ever possible. Processor Blade implementations should understand that the use of DES (Directed Event String) as described in section 5.8.1.1 on Directed Event Strings to write entries into the MM event log does not provide any support such as alarms and alerts to external applications. Blade implementations should send IPMI standard events to obtain full support of logging to the CEL, alarms and alert indications.

It is also important to note that when a blade implementation must deviate from standard IPMI methods for events and use a DES it must not attempt to use both a DES and a standard event for the same condition. If this happens there will be redundant and possibly inconsistent and confusing information in the CEL. Therefore blade implementation must ensure redundancy is avoided.

5.22 MM and Blade Capability Discover

The MM firmware provides support for numerous functions that work with the blade's firmware participation. This function(s) supported need to be advertised by both the blade and the MM firmware. When the blade learns of MM firmware capabilities, this is termed as "Capability

Advertising". When the MM firmware obtains the blade capability, this is termed as "Capability Learning". The capability terminology is in reference to the MM firmware, advertising when reporting to the blade, learning when obtaining from the blade.

5.22.1 MM Capability Advertising

The MM firmware and chassis capabilities are advertised to the blade with the OEM *Get Chassis Info* command. Each capability advertised to the BMC is defined to a bit definition in a byte vector.

Table 5-16 MM Capability Definitions

Byte	Description
15-22	**Reserved for Future Use**
**Note: There are no capabilities defined, but the BMC should reserve these fields for future enhancements that the BMC would learn from the MM on the command exchange.	

5.22.2 Blade Capability Learning

The MM firmware must learn which operations are supported on a per blade basis. Blade products must advertise their particular capabilities to enable or disable MM operations to that particular blade.

There are two methods that the blade may advertise its capabilities to the MM firmware, one by an IPMI OEM command response from the BMC and the other by an OEM SDR. Resolving which method to use depends on when the MM firmware must learn the capability, since the method employed is dictated by at what point in time during blade initialization the MM requires this information.

5.22.2.1 Capability Learning via OEM Command

The first method is to learn capability via the IPMI OEM command *Get Blade Descriptor* which contains a set of pre-defined capability bits. This method allows the MM firmware to learn capabilities early in the initialization process without having to wait to read the BMC's entire SDR Repository.

The format of the *Get Blade Descriptor* is defined in section 5.27. The table below defines the bit definitions and descriptions of the *Get Blade Descriptor* command:

Table 5-17 Blade Capability Definitions

Byte	Bit	Description
15	0	**SOL** Blade can support IPMI Serial Over LAN sessions with the MM firmware. User is provided SOL support through the MM external port. SOL configuration options will be exposed to MM firmware users for blades reporting this capability.
	1	**Reserved – 0b**

<table>
<tr><th>Byte</th><th>Bit</th><th>Description</th></tr>
<tr><td></td><td>2</td><td><b>KVM</b>
Blade has support for local chassis keyboard, video, and mouse shared resource. The MM firmware uses this reported capability to setup its policy tables for KVM control.</td></tr>
<tr><td></td><td>3</td><td><b>MT</b>
Blader has support for chassis Media Tray (CD/DVD/FD drives) shared resource. The Blade can also attach to Media Tray USB, but not necessarily boot from it. The MM firmware uses this reported capability to setup its policy tables for MT control.</td></tr>
<tr><td></td><td>4</td><td><b>WOL</b>
Blade has support for Wake on LAN. WOL is used for blade local power control from other location on LAN. This reported capability instructs the MM firmware to expose WOL as a user configurable option.</td></tr>
<tr><td></td><td>5</td><td><b>Reserved – 0b</b></td></tr>
<tr><td></td><td>6</td><td><b>Remote Media</b>
The blade supports using USB to attach remote devices that are provided by the MM. This is referred to as “legacy remote media”. It does not infer concurrency.</td></tr>
<tr><td></td><td>7</td><td><b>Video Support</b>
Blade supports on-blade video for MM control. For the blade to support remote control it must also support the KVM capability.</td></tr>
<tr><td>16</td><td>0</td><td><b>Video/USB over IP</b>
The Blade supports using video and USB devices over Internet Protocol (IP) via concurrent KVM</td></tr>
<tr><td></td><td>1</td><td><b>Bootpath Unsupported</b>
Bootpath configuration is not supported by this blade.</td></tr>
<tr><td></td><td>2</td><td><b>External Port LAN Configuration</b>
Blade supports the MM OEM command <i>Set External Port LAN Configuration</i>. This command requests the blade to change the LAN settings on one of its external Ethernet ports as seen by the main processor (not BMC) on the blade.</td></tr>
<tr><td></td><td>3-5</td><td><b>Power Management</b>
<table>
<tr><th>Bit 3</th><th>Bit 4</th><th>Bit 5</th><th>Definition</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>BMC firmware supports static fixed power management. Note: default for all IPMI blades that have only fixed power management firmware support.</td></tr>
<tr><td colspan="4">All other combinations are reserved.</td></tr>
</table></td></tr>
<tr><td></td><td>6</td><td><b>CPU Analog Temperature Readings Not Supported</b>
When this bit is '0', the blade supports CPU analog temperature sensors. When this bit is '1', the blade does not support CPU analog temperature sensors.</td></tr>
<tr><td></td><td>7</td><td><b>Internal Bus Flash Not Supported</b>
When this bit is ‘1b’ it indicates the blade does NOT support internal bus (RS485) flash from the MM. If this bit is ‘0b’ it indicates the Blade does support flash via the internal bus (RS485).</td></tr>
<tr><td>17</td><td>0</td><td><b>Reserved – 0b</b></td></tr>
<tr><td></td><td>1</td><td><b>USB Disk Boot Option</b>
Blade supports the on-blade USB disk boot option. This informs the MM firmware to make the option available for bootpath configuration.</td></tr>
<tr><td></td><td>2</td><td><b>Reserved – 0b</b></td></tr>
</table>

Byte	Bit	Description
	3	**Enhanced Power Throttling Supported** This bit will be set to '1' when the BMC supports 'Enhanced Power Throttling" as follows; The base power throttling behavior continues to be that power failure detection is based on blade detecting the assertion of an EPOW_N signal on one of the blades VHDM connectors OR detecting a loss of 12V input power on one or the blades power connectors. The enhanced behavior is that the blade must remain throttled to the MM pre-set power value (send to BMC using the 'Set Pre-Set Watts' command) if either there is a continued assertion of an EPOW_N signal or loss of 12V A/B power. Note: if the BMC set's this bit to '0' then in there may be some cases that the MM will assume the blade does not support throttling for assertion of an EPOW_N signal or loss of 12V A/B power.
	4-6	**Reserved – 000b**
	7	**Reserved – 0b**
18	0	**SOL NACK Support** This bit is set to '1' if the BMC supports receiving NACK's from the MM during SOL session. This bit can be used for flow control between the AMM and the BMC. While the NACK support is defined in the SOL specification, this bit allows the MM to have prior knowledge of this capability. This bit, if not set, identifies those blades that have not implemented the support for an SOL NACK, which will ensure the MM will not send NACK's.
	1-7	**Reserved – 0000000b**
19-22	N/A	**Reserved – 00h**

5.22.2.2 Capability Learning via OEM SDR

The second method for learning blade capabilities is by OEM SDR definition (see section 5.25.2) which advertises capabilities to the MM which are learned after the blade's SDR repository has been read from the BMC. The capabilities learned by this method are learned towards the end of blade's initialization sequence, but before the blade is fully initialized and becomes available for MM user management purposes.

5.23 IPMI Entity ID Usage Conventions

This section describes the MM and blade usage of IPMI defined *Entity ID's.*

5.23.1 BladeServer IPMI Entity ID Usage

The following is a list of the blade's Entity ID's and types known to the MM. Other Entity Types for sensors may be used, but will be handled in a generic manner.

IPMI Entity ID Code	IPMI Entity	Description
03h	Processor	CPU's
04h	Disk	Hard Disk Drives
07h	System Board / Main System Board	• Main board for Processor Blades • Occupies a chassis blade slot • Contains the BMC that

		communicates with the MM
0Bh	Add-in Card	• Used for cKVM • Does not increase width of blade
10h	System Internal Expansion Board (may contain I/O expansion connectors)	• Blade Expansion Module (BEM) - PEU (PCI Expansion Unit) - BSE (Blade Storage Expansion) - CPU or Memory expansion - Increases the width of the blade and therefore occupies a slot, does not contain a BMC
14h	Power Converter Module / DC-to-DC converter	• VRMs and VRDs
2Ch	I/O Module	• I/O Expansion Card (multiple instances are supported) • Connects blade to Chassis Switch Modules • Does not increase width of blade

Table 5-18 Entity IDs

5.24 Reserved Sensor Types

Some MM operations are facilitated by IPMI events, usually by standard events. However, some operations require OEM events from the blades, and all blades must be consistent to take advantage of MM event operation support. The Sensor Type has an OEM range assigned by the IPMI standard. Due to the lack of the IPMI standard providing a scope mechanism for Sensor Types, the BladeServer IPMI OEM Architecture is reserving the last sixteen OEM values (*F0h-FFh)* for Processor Blade chassis operations use.

5.25 OEM SDR Formats

This section details the formats for the OEM SDRs that have been defined with BladeServer scope. OEM SDRs have BladeServer scope by using the IANA value defined by the OEM SDR format in Table 5-19 OEM Record - SDR Type C0h.

Table 5-19 OEM Record - SDR Type C0h

Byte	Field Name	Size	Description
RECORD HEADER			
1:2	Record ID	2	As per IPMI Specification.

Byte	Field Name	Size	Description
3	SDR Version	1	
4	Record Type	1	
5	Record Length	1	
RECORD KEY BYTES			
6:8	Manufacturer ID	3	Manufacturer ID code. LS byte first. Most significant 4 bits = reserved (0000b). 000000h = unspecified, 0FFFFFh = reserved. This value is binary encoded 0051D0h (20944d) = '*Modular Blade Server'*
9:9+N	OEM Data	N	OEM Data, N bytes.

Once the OEM SDR is recognized as BladeServer IPMI Architecture compliant, the MM will parse the "OEM Data" segment according to the rules defined by this document. The OEM Type Code, as shown in Table 5-20, is parsed first to establish the appropriate data format.

Table 5-20 OEM SDR Data

SDR Byte	Field Name	Size	Description
OEM Data			
9	OEM Type	1	OEM Type Code, defines format parsing rules.
10:10+N	Data	N	Data formatted according to the rules defined by OEM Type Code(s) defined in the following sections.

Table 5-21 OEM SDR Types

Type	Name	Description
0x01	LED Properties	Contains information on LED properties, such as color, identification, etc.
0x02	Capability Report	Reports blade capabilities that do not have to be accessed before the MM caches the blade SDRs.
0xED	Reserved	Special Legacy support, do not assign.

This rest of this section provides the OEM Type Codes defined and their corresponding formats.

5.25.1 OEM Type 01h – LED Properties

An OEM Type of 01h designates the OEM SDR as containing LED information and properties.

Table 5-22 LED Properties Format

SDR type 0xC0 – OEM Record			
Byte	Field	Size	Description
9	OEM Type	1	01h = LED Properties.

10	LED Owner ID	1	[7:1] - 7-bit I^2C Slave Address, or 7-bit system software ID. [0] - 0b = ID is IPMB Slave Address, 1b = system software ID
11	LED Owner LUN	1	[7:4] - Channel Number The Channel Number can be used to specify access to LED owning controllers that are connected to the BMC via channels other than the primary IPMB. [3:2] - reserved [1:0] - LED Owner LUN.
12	Entity ID	1	Indicates the physical entity that the LED is associated to.
13	Entity Instance	1	[7] - 0b = treat entity as a physical entity per Entity ID table. 1b = treat entity as a logical container entity.
14:15	LED Identifier	2	This is a unique identifier for this LED (MSB first).
16	Color Set	1	Bit mask describing the color(s) available to this LED: 0x01: Red 0x02: Orange 0x04: Yellow 0x08: Green 0x10: Blue
17	Location	1	Bit mask describing the location(s) of this LED: 0x01: Front Panel 0x02: Diagnostic Card 0x04: Planar 0x08: FRU 0x80: Rear Panel
18	Description Type/Length Code		7:6 – 11b = 8 bit ASCII (All others reserved) 5 – Reserved 4.0 – length of the following data in characters. 00000b indicates none following. 11111b = reserved.
19:+N	Description	N	ASCII description of this LED.

LED IDs are arbitrarily assigned by the product. Most LED's are learned from SDR with properties and descriptive strings. However, to support legacy architected LED's, some LED IDs are reserved. The reserved LED Ids cover the standard front panel LEDS that all blades must support.

Table 5-23 Reserved LED IDs

LED ID	Description
0000h	Front Panel Fault
0001h	Front Panel Identification
0003h	Front Panel Information
0023h	Front Panel KVM select
0024h	Front Panel Media select

5.25.2 OEM Type 02h – Capability Report

This OEM SDR reports blade capabilities for which the MM firmware can support.

Table 5-24 Capabilities Report Format

SDR type 0xC0 – OEM Record			
Byte	**Field Name**	**Size**	**Description**
9	OEM type	1	02h – Identifies this OEM record as an capabilities record
10	Number of Capabilities	1	Defines the number of capabilities (TLV entries) described in this SDR.
11	Capability Type	2	Type of capability supported by the Blade
13	Capability Length	1	Length of the data. This is optional and may be set to 0 if no data is required for the capability.
14	Capability Value	N	Capability data if appropriate (see type definitions). Length is defined in the previous byte. Value is limited to 16 bytes maximum.
N+1:M	Repeat bytes 11 thru N for the next capability.		

The capability report format includes a capability type which is defined in Table 5-25.

Table 5-25 Capability Type Definition

Capability Type	Value	Description
0000h	N/A	Reserved
0001h	None	BMC Dump Supported
0002h – FFFFh	None	Reserved

5.26 Blade Chassis and Processor Blade Compatibility

Various situations may exist that prevent a Processor Blade product from functioning correctly in the different BladeServer chassis types. This is known as blade and chassis compatibility detection and blade product developers should keep these issues in mind during product design.

A few examples of compatibility issues are:

1) A specific configuration of processors, memory devices (DIMM), hard-drives, and other Processor Blade components may have some configurations that will work correctly in one chassis model, such as the BladeServer High Speed Chassis, but may have serious operational issues in a BladeServer chassis model. For example a blade may have serious thermal problems due to the maximum processor power and downstream DIMMs installed in a particular BladeServer chassis type, such as the BladeServer chassis, which may not have the

appropriate air flow and cooling capabilities for the Processor Blade.

In the example discussed above the Processor Blade product may wish to support a new higher power (wattage) processor in all types of BladeServer chassis. But after analyzing the thermal characteristics of the blade it was determined that in some BladeServer chassis models, this could cause serious thermal problems. For this example, there may be multiple approaches by the Processor Blade product to solve this problem.

a) Set the p-state (Performance-State) of the processors so that the thermal load would be reduced on the overall Processor Blade if the chassis type discovered is recognized as inappropriate for full p-state. The BMC or host firmware would ensure the blade runs at a reduced Voltage and Frequency (V/F).

b) Prevent power to be applied to the Processor Blade product due to the BMC determining that the current chassis type the blade is installed in may cause potential hard-drive thermal issues or other thermal related problems. The issue would be reported via events and alarms to explain the power denial due to chassis incompatibility.

2) A Processor Blade may have some other limitations that prevents it from functioning correctly due to:

a) BMC detects that the BladeServer chassis model is incompatible.

b) Processor Blade requires a specific MM model (for Example AMM versus MM).

The BMC responsibilities when a chassis compatibility issue is detected:

1) BMC will be responsible for powering off the Processor Blade after detecting a chassis type mismatch (chassis compatibility issue).

2) BMC will be responsible to insure the Processor Blade cannot be powered on via any power-on command to the BMC, regardless of origin, such as front panel request, IPMI commands, WOL, etc.

3) There is no specific MM defined event to be sent by the BMC to report a chassis compatibility issue, though the Processor Blade product is still free to send an event to report the exact nature as to the reason for the chassis compatibility issue.

4) The OEM completion codes for the OEM command ‘Get Watt Usage’ is used to inform the MM of the chassis compatibility issue. Please refer to *Get Watt Usage* command definition for the defined completion code values.

The following section describes various ‘Chassis Compatibility’ scenarios that may occur. Blade products must comply with the operations defined:

5.26.1 Compatibility on Processor Blade Insertion/Chassis Power On

This scenario discusses the operation where a Processor Blade is inserted into an already powered on chassis or AC power is applied to the chassis and the Processor Blade is discovered during chassis initialization.

1) The MM sends the 'Chassis Information Command' command to the BMC as per the previously defined blade initialization flows. The command contains information identifying chassis and MM types, including MM firmware version information.

2) During the Pre-Boot Inventory Power sequence, a *Get Watt Usage* command will be sent to the BMC.

3) Further operations are dependant upon the Processor Blade architecture that is implemented:

    a) The Processor Blade architecture may still require further information exchange between the BMC and host firmware (BIOS) to discover additional Processor Blade information.

        (1) BMC returns information in the *Get Watt Usage* command response with the power requirements prior to a POST running. The MM uses this information to make a decision on the current power budget. If the processor Blade can be contained in the power budget, the MM will grant power permission to the blade (standard behavior).

        (2) With power permission granted, the administrator may now power on the Processor Blade. During a chassis AC cycle, the MM may also automatically powers on the Processor Blade due to Processor Blade previously being powered on.

        (3) BMC exchanges information with BIOS during POST and determines the installed processors have excessive power and thermal requirements for the current chassis.

        (4) The BMC sends a Blade State Change (BSC) to the MM for 'Maximum Watt Updated'.

        (5) The MM sends a 'Get Watt Usage' command to the BMC. The BMC responds with a Completion Code of 02h.

        (6) The MM will recognize the "Blade Chassis Mismatch" completion code and sends a 'Set Pre-Set Watts' command with the indication of 'permission denied to continuing booting'.

        (7) BMC must indicate to BIOS to not continue booting and power off the blade.

        (8) The BMC will send the MM a *Blade State Change* message with a 'power down' state notification.

        (9) The MM sends a *Set Power Permission* command to the BMC with the 'power permission – denied' indication. Any subsequent power-on requests received by the BMC from any source must not be honored.

        (10) The MM will create CEL entry and alarm on the fact there is a chassis type and blade incompatibility.

(11) The BMC must continue to send the *'Blade Chassis Mismatch'* completion code on any subsequent '*Get Watt Usage'* command issued by the MM.

(12) The BMC should save the chassis incompatibility condition to a non-volatile storage area to optimize and handle a BMC reset case so that another POST sequence is not required. If at any point the blade loses AC power or is removed from the chassis the non-volatile storage area must be cleared.

b) The Processor Blade architecture does not require further information exchange between the BMC and BIOS to discover additional blade information.

(1) BMC immediately returns the *Get Watt Usage* command response with the completion code of 'Blade Chassis Mismatch'. Same flow will occur as described in (a) above step (6).

5.27 BladeServer OEM Commands

This section provides the formats for the IPMI OEM commands discussed elsewhere in this document.

5.27.1 BladeServer IPMI OEM Network Function

The BladeServer MM uses IPMI Commands to manage Processor Blades that support IPMI. Additional IPMI commands are defined for Processor Blades to provide the additional management requirements not covered within the IPMI standard.

Each Processor Blade usually its own unique set of OEM controller specific commands. For BladeServer management, the MM has its own set of OEM commands which shall be supported by each Processor Blade. To avoid any conflict to the product controller specific definitions, the *BladeServer IPMI OEM* requirements use an OEM group which is further qualified by the IANA OEM identifier for BladeServer - *'Modular Blade Server'.*

5.27.1.1 BladeServer IPMI OEM Commands

The required *BladeServer IPMI OEM* commands will be grouped by the following definitions:

- **Network Function: 0x2E, 0x2F – OEM Group**
- **IANA 'Modular Server' Number: 20944d (0051D0h, least significant byte first)**

The above network function is defined by IPMI standards as an OEM group and is further qualified by the IANA assigned Enterprise Number, which is 20944d for 'Modular Blade Server'.

Firmware implementers should note that these OEM commands always contain the IANA number as the first three bytes (LS-byte first) of the data portion of all OEM requests and the first three bytes of the data portion following the completion code of all responses.

The MM and Processor Blade shall parse all OEM commands received with this Network Function as a *BladeServer IPMI OEM* Command Set.

5.27.1.2 BladeServer Examples of IPMI Controller Specific OEM Commands

The BIOS may have specific requirements on the BMC to handle unique functions outside the scope of what is handled by the MM defined OEM group (network function 0x2E/0x2F). Utilizing OEM controller specific commands may then be needed. For informational purposes, examples of three such commands referenced in section 5.18.2 are provided below.

<table>
<tr><th>Command</th><th>Request, Response Data</th><th>Description</th></tr>
<tr><td>Set System GUID</td><td>**Request**:
Byte 1:16 — UUID (LSB First) Processor number[1]
Response:
Byte 1 — Completion code.</td><td>This command provides the BMC with the current UUID of the processor blade.
Note 1: Refer to the ‘Base Specification for VPD’ for additional information on UUID.</td></tr>
<tr><td>Configure ABR Watchdog</td><td>**Request**:
Byte 1 — Function Code 1
Byte 2 — Function Code 2
Byte 3 — Timeout value - MSB
Byte 4 — Timeout value - LSB
Byte 5 — Function Code 3
Response:
Byte 1 — Completion code.</td><td>This command is issued by the BIOS to request the BMC to activate or deactivate the ABR watchdog. In addition this command can be used to signal the BMC to switch to the primary boot image and then restart the processor blade and run POST.
<table>
<tr><th>Action Performed</th><th colspan="3">Function Code</th><th>Time--out</th></tr>
<tr><td></td><td>1</td><td>2</td><td>3</td><td></td></tr>
<tr><td>Disable Watchdog</td><td>00h</td><td>00h</td><td>00h</td><td>NA</td></tr>
<tr><td>Enable Watchdog</td><td>00h</td><td>01h</td><td>00h</td><td>Time</td></tr>
<tr><td>Switch to primary Image and reset blade</td><td>01h</td><td>00h</td><td>01h</td><td>NA</td></tr>
</table>
</td></tr>
<tr><td>Processor T_Control Information</td><td>**Request**:
Byte 1 — Physical Processor number[1]
Byte 2 — T_control value[2]
Response:
Byte 1 — Completion code.</td><td>This command issued by the BIOS requests the BMC to update the BMC's information about the processor blade's T_Control (temperature) value for each CPU. This information is utilized by the BMC for invoking throttling for thermal reasons. A separate command is issued for each processor.
Note 1: CPU1 = 00h, CPU2 = 01h , CPU3 = 02h, CPU3 = 04h
Note 2: Temperature value is in units of Celsius (valid range is from 00h = 0° to FFh = 255°)</td></tr>
</table>

5.27.2 BladeServer OEM Command Summary

The following table presents the list of OEM BladeServer command set. This only serves as a quick reference for command values. For details on the actual OEM message format, see section 5.19.3 OEM Message Formats.

The term “reserved” means that the field is not used and may be defined in a future version. All reserved fields must be set to all ‘0’ (zeroes). If a non-zero reserved field is received by a Processor Blade it will be ignored.

The term mandatory means that the command must be supported, regardless of blade function implemented.

The term optional means that the blade only needs to implement the command if the related function is supported on the blade. Example of this would be the KVM and/or Media Tray resource sharing request. If the blade does not require these resources, then that command is not required to request those resources. Additionally the blade would indicate this function is not supported in the blade capabilities.

Table 5-26 OEM Summary

Cmd	**Name**	**Rq[1]**	**Short Description**
00h	Get Blade Descriptor	M	MM sends to Processor Blade to obtain required OEM data.
01h	Blade State Change Message	M	Unsolicited notification sent by the Processor Blade to inform the MM that blade state has changed, including operational state and/or tracked data has changed.
02h	Set Power Permission	M	MM sends to set power permission and local power control.
03h	Get Power Permission	M	MM sends to check the current power permission settings.
04h	Set Blade KVM-MT Selection	M	MM sends to set the current state of the KVM and/or Media Tray (CD/FDD), informing blade BMC to enable or disable its local usage. This command is required even if KVM-MT resources are not required by Processor Blade.
05h	Get Blade KVM-MT Selection	M	MM sends to Processor Blade to obtain current enable/disable state for KVM and/or Media Tray (CD/FDD). This command is required even if KVM-MT resources are not required by Processor Blade.
06h	Request Access to KVM-MT	O	Processor Blade sends to MM to request use of the KVM and/or Media Tray (CD/FDD). This command is only required if KVM-MT resources are required by Processor Blade.

07h	Get Blade System Information	M	MM sends to Processor Blade to obtain additional OEM data
08h	Directed Event String	M	Processor Blade sends to MM BIOS generated ASCII string reporting BIOS progress and BIOS errors.
09h	Write IBM Format VPD	M	MM sends to Processor Blade to write data a specified VPD block. If this command is not supported, then each VPD requirement is available using standard IPMI logical FRU commands and associated SDRs are available.
0Ah	Read IBM Format VPD	M	MM sends to Processor Blade to read data from a specified VPD block. If this command is not supported, then each VPD requirement is available using standard IPMI logical FRU commands and associated SDRs are available.
0Bh	Set LED	M	MM sends to Processor Blade to change the state of an LED.
0Ch	Get LED	M	MM sends to Processor Blade to get the current state of an LED.
0Dh	Reserved		
0Eh	Restore LAN Configuration	M	MM sends to Processor Blade to reset LAN configuration to its default settings.
0Fh	Reserved		Reserved for future *BladeServer IPMI OEM* command extensions.
10h	Flash Start	M	MM sends to Processor Blade to inform it to enter flash update mode.
11h	Flash Data	M	Flash data sent from MM to Processor Blade.
12h	Flash Fill	M	An optional OEM command that can be used to increase performance of file transfers when repeating patterns are sent. This command is optional and can be responded to as an unsupported command.
13h	Flash End	M	MM sends to Processor Blade to inform it that the flashing operation has ended and the Processor Blade should now exit firmware mode and run its new image.
14h	Directed Event Message	M	Processor Blade sends SEL data to the MM using this command.
15h	Set Directed Event Message Enable	M	MM sends this command to inform the Processor Blade to start sending unsolicited DEM's.
16h	Get Directed Event Message Enable	M	MM uses this command to obtain the current setting of the Processor Blade DEM enable state.
17h-1Fh	Reserved		Reserved for future *BladeServer IPMI OEM* command extensions.
20h	Set Thermal Throttle Enable	M	MM uses this command to request BMC to enable or disable throttling the Processor Blade for thermal reasons when the Thermal Diode on the Processor(s) exceeds Tc.

21h	Get Thermal Throttle Enable	M	MM uses this command to request BMC to return the current setting of the throttle enable flag
22h	Get Watt Usage	M	MM uses this command to request the BMC to provide the value of watts used per slot that make up the Processor Blade and attached expansion units.
23h	Get Throttle Status	M	MM uses this command to request the BMC to return the current values for the reason the blade may be throttling, the current duty cycle of the processors and the current power throttle value.
24h	Set Pre-Set Watts	M	MM uses this command to request BMC to change the throttle value of the processor(s) or to set up the BMC pre-set throttle value in case of a loss of the redundant power supply to the Processor Blade
25h	Get Pre-Set Watts	M	MM and BIOS use this command to request information from the BMC on boot permission and component throttle values.
26h	Set Component Watts Inventory	M	BIOS will issue this command to the BMC to provide an inventory of power values for components on the Processor Blade
27h	Get Component Watts Inventory	M	MM will use this command to request the BMC to provide a inventory of power values for components on the Processor Blade
28h	CPU Throttle Table Information	M	BIOS will issue this command to the BMC to provide the what throttle values are for each CPU on the Processor Blade.
29-2Fh	Reserved		Reserved for future BladeServer IPMI OEM command extensions.
30h	Chassis Information	M	This command provides for exchange of basic information between the MM and BMC.
31h-4Fh	Reserved		Reserved for future BladeServer IPMI OEM command extensions.
50h	Initiate Data Collection	O	This command provides a means for the MM to initiate data (dump) collection on the Processor Blade. This command is required if capability is supported.
51h-FFh	Reserved		Reserved for future BladeServer IPMI OEM command extensions.

Note 1: Rq (Requirement) Column defines if OEM command is mandatory (M) or optional (O).

5.27.3 OEM Message Formats

Table 5-27 OEM Formats

Net Function = 0x2E			
Code	**Command**	**Request, Response Data**	**Description**
00h	Get Blade Descriptor	**Request**: Byte 1:3 — IANA Enterprise Number (LS-byte first) **Response**: Byte 1 — completion code Byte 2:4 — IANA Enterprise Number (LS-byte first) Byte 5 — reserved Byte 6 — blade width Byte 7 — PD2 power status 7:7 PD2 power on/off 0b = blade is powered off 1b = blade is powered on 6:3 reserved 2:2 Wake On LAN status 0b = disabled 1b = enabled 1:1 PD2 permission status 0b = denied 1b = allowed 0:0 local power control status 0b = denied 1b = allowed Bytes 8:14 – Build Name (ASCII)[1] Bytes 15:22 – Capabilities[2] Bytes 23 - Reserved	This command returns blade specific address, width, and power status information. Its purpose is to support a query by the MM in a Processor Blade system. **Note 1:** For Build Names less than seven bytes, this field should be left justified, padded to the right with ASCII space characters (20h). **Note 2:** Unless otherwise stated in the bit description, capability bit values are defined as 1b = supported, and 0b = not supported. Capability bit definitions may be found in Table 5-17 Blade Capability Definitions.
01h	Blade State Change Message	**Request**: Byte 1:3 — IANA Enterprise Number (LS-byte first) Byte 4 — Action taken 00h – power down[4] 01h – power up 02h – power cycle (optional) 03h – hard reset[1] 04h – BMC entered flash update mode 05h – BMC completed reset 06h – reserved 07h – Main Planar VPD modified[2] 08h – Other VPD modified[3]	This command is sent by the blade BMC to the MM when an event has caused changes to blade state. This message informs the MM of the change that has occurred and allows the MM to remain in synch with state of the blade. This includes Power Domain 2 state (power down, power on and power cycle), BMC state, FRU data updates, and SDR Repository updates. **Note 1:** The term "hard reset" means the main processor(s) has jumped to the Power-On-Reset vector.

Net Function = 0x2E			
Code	**Command**	**Request, Response Data**	**Description**
		09h – reserved 0Ah – SDR Repository Updated 0Bh – Maximum Watt Updated 0Ch – Throttle change[5] 0Dh – Un-throttled[5] 0Eh – Thermal Threshold (Tc) exceeded 0Fh – Thermal Threshold (Tc) recovery (issued only after a BSC of 0Eh has been sent) all other – reserved Byte 5 — Current power state 00h – power is off 01h – power is on all other encodes are reserved **Response**: Byte 1 — completion code Byte 2:4 — IANA Enterprise Number (LS-byte first)	**Note 2:** Only changes to the manufacturing data block of VPD require a BSC notification message from the blade BMC to the MM. **Note 3:** Manufacturing data block of VPD has been modified for compnents other than the main planar's VPD. **Note 4:** This state change action is used by the MM for power restoration of the blade **Note 5:** These blade state change messages are used to notify the MM as the processor(s) on the blade throttle and un-throttle for power management. The BMC needs to prevent flooding of the RS485 bus when these actions occur. A typical implentation should not notify the MM on every 'small' percentage change in the voltage/frequency of the processors. The implementation should use a fixed value of each throttle 'steps' to detemine when the MM would be notified. For example in it may be typical the steps are in 12.5% increments. When the blade transitioning from one step to different step the MM would be notified.
02h	Set Power Permission	**Request**: Byte 1:3 — IANA Enterprise Number (LS-byte first) Byte 4 power permissions 7:3 reserved 2:2 Wake On LAN enable 0b = disable[1] 1b = enable[2] 1:1 power on permission 0b = deny 1b = allow 0:0 local power control 0b = disable[1] 1b = enable[2]	This command is used to set the power permission state of the blade. The power permission state is stored in volatile memory by the BMC. All indicator bits refer to permission, not the current state of the blade. An example would be local power control, where if disabled, no local mechanism may power on the blade. If enabled, the power state of the blade may be changed by local mechanisms (front panel button, in-band command, etc). **Note 1:** Disable means the blade must disable all local mechanisms. **Note 2:** Enable means the blade is

Net Function = 0x2E			
Code	**Command**	**Request, Response Data**	**Description**
		Response: Byte 1 — completion code Byte 2:4 — IANA Enterprise Number (LS-byte first)	allowed to obey local blade mechanisms for power control.
03h	Get Power Permission	**Request**: Byte 1:3 — IANA Enterprise Number (LS-byte first) **Response**: Byte 1 — completion code Byte 2:4 — IANA Enterprise Number (LS-byte first) Byte 5 — power permission 7:3 reserved 2:2 Wake On LAN enable 0b = disable 1b = enable 1:1 power on permission 0b = deny 1b = allow 0:0 local power control 0b = disable 1b = enable	This command is used to get the power permission state of the blade. The power permission state is stored in volatile memory by the BMC. All indicator bits refer to permission, not the current state of the blade. An example would be local power control, where if disabled, no local mechanism may power on the blade. If enabled, the power state of the blade may be changed by local mechanisms (front panel button, in-band command, etc).
04h	Set Blade KVM-MT Selection	**Request**: Byte 1:3 — IANA Enterprise Number (LS-byte first) Byte 4 Media Tray and KVM Select 7:2 reserved 1:1 Media Tray 0b = deselect 1b = select 0:0 KVM 0b = deselect 1b = select **Response**: Byte 1 – completion code Byte 2:4 — IANA Enterprise Number (LS-byte first)	This command from the MM directs the blade BMC to select or deselect the BladeServer chassis Media Tray (CD/FDD) and the KVM (keyboard/video/mouse). Return of a normal completion code indicates that the operation was completed successfully.
05h	Get Blade KVM-MT Selection	**Request**: Byte 1:3 — IANA Enterprise Number (LS-byte first) **Response**: Byte 1 – completion code Byte 2:4 — IANA Enterprise Number (LS-byte first) Byte 5 Media Tray and KVM Select	This command returns a blade's current selection status of the BladeServer chassis Media Tray (CD/FDD) and the KVM (keyboard/video/mouse).

Net Function = 0x2E			
Code	**Command**	**Request, Response Data**	**Description**
		7:2 reserved 1:1 Media Tray 0b = deselected 1b = selected 0:0 KVM 0b = deselected 1b = selected	
06h	Blade KVM-MT State Message	**Request**: Byte 1:3 — IANA Enterprise Number (LS-byte first) Byte 4 Media Tray and KVM Select 7:7 State Type 0b = request 1b = state change 6:2 reserved 1:1 Media Tray 0b = deselect 1b = select 0:0 KVM 0b = deselect 1b = select **Response**: Byte 1 – completion code Byte 2:4 — IANA Enterprise Number (LS-byte first)	This command is sent by the blade BMC to the MM in order to request access to the BladeServerr chassis Media Tray (CD/FDD) and/or the KVM (keyboard/video/mouse). **Note**: Return of a normal completion code only indicates that the command was successfully received. It does not indicate that access has been granted. Access is granted by the MM by issuing a *Set Blade KVM-MT Selection* command to the blade BMC to allow it to use the shared resource.
07h	Get Blade System Information	**Request**: Byte 1:3 — IANA Enterprise Number (LS-byte first) **Response**: Byte 1 – completion code Byte 2:4 — IANA Enterprise Number (LS-byte first) Byte 5 – reserved Byte 6 – Maximum number of CPUs[1] Byte 7 – Maximum number of DIMMs	This command is used to obtain certain information about the blade. **Note 1:** This value represents the number of sockets that contain CPU(s) not the number of cores.
08h	Directed Event String	**Request**: Byte 1:3 — IANA Enterprise Number (LS-byte first) Byte 4 — Sequence number / flags 7:7 1b = End of String 6:0 Sequence Number[1] Byte 5:N — BIOS String.	Command is used by BMC to send MM any BIOS progress or error strings. String is placed into the MM even log. The total maximum length of a BIOS string is 160 bytes.

Net Function = 0x2E			
Code	**Command**	**Request, Response Data**	**Description**
		Response: Byte 1 — completion code Byte 2:4 — IANA Enterprise Number (LS-byte first)	**Note 1:** The sequence number is one based. The first message always has the sequence number set to a value of 1h and subsequent messages sent in order. The last message, including the first message if the entire string message is contained, must have the flag bit set to 1b.
09h	Write IBM Format VPD	**Request**: Byte 1:3 — IANA Enterprise Number (LS-byte first) Byte 4 — Offset (ls-byte) Byte 5 — Offset (ms-byte) Byte 6 – FRU Device ID[1] Byte 7:32 – Data to write (28 bytes max) **Response**: Byte 1 – Completion code Byte 2:4 — IANA Enterprise Number (LS-byte first)	This command is used to write IBM format VPD data for any component on the Processor Blade. All counts are '1' based. **Note 1:** The supported values for FRU Device ID's are listed in Table 5-28 – FRU Device IDs for IBM VPD Read/Write OEM command.
0Ah	Read IBM Format VPD	**Request**: Byte 1:3 — IANA Enterprise Number (LS-byte first) Byte 4 — Offset (ls-byte) Byte 5 — Offset (ms-byte) Byte 6 – FRU Device ID[1] Byte 7 — Byte count to read (1 to 128). **Response**: Byte 1 – Completion code Byte 2:4 — IANA Enterprise Number (LS-byte first) Byte 5:1+N – Requested data	This command is used to by the MM to read IBM format VPD data for any component on the Processor Blade. All counts are '1' based. **Note 1:** The supported values for FRU Device ID's are listed in Table 5-28 – FRU Device IDs for IBM VPD Read/Write OEM command. **Note 5:** A completion code of C0h (Node Busy) should only be used by the BMC to indicate that an installed I/O expansion card or add-in card (for example cKVM) is currently in a state that would prevent it from providing VPD information. This could be used for example when the component is in a fault state.
0Bh	LED Set	**Request**: Byte 1:3 — IANA Modular Server Numb. (LS-byte first) Byte 4:5 – LED Identifier Byte 6 – Command 00: Off 01: On	This command is used by the MM for external control of specific LEDs. Uses the LED identifier retrieved from the OEM LED Property SDRs (See 5.25.1OEM Type 01h – LED Properties).

<table>
<tr><th colspan="4">Net Function = 0x2E</th></tr>
<tr><th>Code</th><th>Command</th><th>Request, Response Data</th><th>Description</th></tr>
<tr><td></td><td></td><td>80: Blink
<b>Response</b>:
Byte 1 – completion code
Byte 2:4 — IANA Modular Server Numb.
(LS-byte first)</td><td></td></tr>
<tr><td>0Ch</td><td>LED Get</td><td><b>Request</b>:
Byte 1:3 — IANA Modular Server Numb.
(LS-byte first)
Byte 4:5 – LED Identifier
<b>Response</b>:
Byte 1 – completion code
Byte 2:4 — IANA Modular Server Numb.
(LS-byte first)
<table>
<tr><th>Byte</th><th>Field</th></tr>
<tr><td>5</td><td>Current color:
0x01: Red
0x02: Orange
0x04: Yellow
0x08: Green
0x10: Blue</td></tr>
<tr><td>6</td><td>Duty Cycle:
0x00: Off
0xFF: On
else: Blinking ratio</td></tr>
<tr><td>7</td><td>Period in 50 millisecond units.</td></tr>
<tr><td>8-9</td><td>Reason:
LED Identifier
(or)
Sensor ID</td></tr>
<tr><td>10</td><td>Reason Type:
0x01: LED Identifier
0x02: Sensor ID
0x03: Manually controlled
0x04: BIOS
0x05 - 0xFF - reserved</td></tr>
</table></td><td>This command is used by the MM to read the state of an LED. The request parameter is the ‘LED Identifier’ (bytes 4:5) is learned from the OEM LED Property SDRs (See 5.25.1OEM Type 01h – LED Properties).
‘Reason Field’ - identifies the most recent reason the LED was lit.
‘Reason Type’ - will indicate what type of information will be found in the ‘Reason field’.</td></tr>
<tr><td>0Dh</td><td>Reserved</td><td></td><td></td></tr>
<tr><td>0Eh</td><td>Restore LAN Configuration Defaults</td><td><b>Request</b>:
Byte 1:3 — IANA Enterprise Number
(LS-byte first)
Byte 4
7:4 reserved
3:0 Channel ID
<b>Response</b>:
Byte 1 – completion code
Byte 2:4 — IANA Enterprise Number
(LS-byte first)</td><td>The BMC shall restore defaults to all LAN channels.
<b>LAN Configuration Defaults</b> (Refer to the <i>Set LAN Configuration</i> and <i>Set Channel Access</i> commands in the IPMI v1.5 Specification)
• The Authentication Type Enables are set to enable straight password, MD2, MD4, and none.
• The Channel Access mode is set to always available.</td></tr>
</table>

Net Function = 0x2E			
Code	**Command**	**Request, Response Data**	**Description**
			• PEF alerting is disabled. • Per-message authentication is enabled. • User level authentication is enabled. • Gratuitous ARP's are enabled.
0Fh	Reserved		
10h	Flash Start	**Request**: Byte 1:3 — IANA Enterprise Number (LS-byte first) Byte 4:11 – Build Name (ASCII) **Response**: Byte 1 – completion code 80h – Unacceptable version Byte 2:4 — IANA Enterprise Number (LS-byte first) Byte 5 – Max packet size (ls-byte) [1] Byte 6 – Max packet size (ms-byte) [1]	This command requests the BMC to start flashing the BMC firmware. If the BMC is in a state that it cannot perform a flash operation it should return the appropriate generic completion code. **Note 1**: Processor Blade informs MM maximum number of bytes that can be supported in the Flash Data command.
11h	Flash Data	**Request**: Byte 1:3 — IANA Enterprise Number (LS-byte first) Byte 4:n – hex data **Response**: Byte 1 – completion code Byte 2:4 — IANA Enterprise Number (LS-byte first) Byte 5 – Data length (ls-byte) Byte 6 – Data length (ms-byte)	This command contains the actual hex data to flash. The first 64 bytes are the defined firmware header information, but the format of the remaining data is defined (and interpreted) by the BMC. The actual data length written is returned, and should match what was sent when a 0 completion code is returned. A non-zero completion code (or mismatched data lengths) terminates the flashing process at the MM. No more than 1024 bytes will be sent in one request.
12h	Flash Fill	**Request**: Byte 1:3 — IANA Enterprise Number (LS-byte first) Byte 4 – Fill pattern length Byte 5 – # Repeats (ls-byte) Byte 6 – # Repeats (ms-byte) Byte 7:n – fill pattern **Response**: Byte 1 – completion code Byte 2:4 — IANA Enterprise Number (LS-byte first) Byte 5 – Data length (ls-byte) Byte 6 – Data length (ms-byte)	This command is used to fill the flash of the BMC with repeating patterns. This can save significant time in the firmware update procedure. If the completion code returns "C1" (unsupported command), the MM will create the correct Flash Data request and resend. The total length of data written is returned.
13h	Flash End	**Request**:	This command signals that that all the

Net Function = 0x2E			
Code	**Command**	**Request, Response Data**	**Description**
		Byte 1:3 — IANA Enterprise Number (LS-byte first) **Response**: Byte 1 – completion code 80h = Invalid Image Byte 2:4 — IANA Enterprise Number (LS-byte first)	flash data has been sent to the BMC. The BMC should send the response, and then reset itself to activate the newly flashed code. If the BMC has the capability to store the entire flash data in a staging area prior to committing to flash, this command causes it to validate the received data and return 0x80 for an invalid image.
14h	*Directed Event Message*	***Request**:* *Byte 1:3 — IANA Enterprise Number* *(LS-byte first)* *Bytes 4:20 – Event Message Data as described in IPMI Specification.* ***Response**:* *Byte 1 — completion code* *Byte 2:4 — IANA Enterprise Number* *(LS-byte first)*	This command provides a means for the BMC to send event information to a receiver. The event information transmitted can represent various event message types. *The supported event messages types are SEL records. See* Table 5-29 SEL Directed Event Format *For Processor Blades, the only receiver supported is the MM. All events are sent over the* RS-485 Interface to the MM.
15h	Set Directed Event Message	**Request**: Byte 1:3 — IANA Enterprise Number (LS-byte first) Byte 4 – Directed event enable/disable 7:7 – enable/disable 0 = disable 1 = enable 6:0 – Directed Event Type 02h = System Event Log (SEL) All other values are reserved. **Response**: Byte 1 — Completion code. Byte 2:4 — IANA Enterprise Number (LS-byte first)	This command is sent by the MM to the BMC and provides a means to enable/disable Directed Event Message (DEM) generation by the BMC according to type. This state is not stored persistently. The BMC reset or power-on default for the BMC is to have Directed Event generation disabled for the supported Directed Event Types. For Processor Blades, the only requestor supported is the MM. All events are sent over the RS-485 Interface to the MM.
16h	Get Directed Event Message	**Request:** Byte 1:3 — IANA Enterprise Number (LS-byte first) Byte 4 – Directed Event Type 7:7 – reserved 6:0 – Directed Event Type 02h = System Event Log (SEL) All other values are reserved. **Response:**	This command is sent by the MM to the BMC and returns the current enable/disable state for Directed Event Message forarding for a specific Directed Event Type.

Net Function = 0x2E			
Code	**Command**	**Request, Response Data**	**Description**
		Byte 1 — Completion code. Byte 2:4 — IANA Enterprise Number (LS-byte first) Byte 5 – Directed event enable/disable state 7:1 – reserved 0:0 – enable/disable 0 = disabled 1 = enabled	
17h-1Fh	Reserved		
20h	Set Thermal Throttle Enable	**Request:** Byte 1:3 — IANA Enterprise Number (LS-byte first) Byte 4 – Enable Flag 7:1 – reserved 0:0 – enable/disable 0 = disabled 1 = enabled **Response**: Byte 1 — Completion code. Byte 2:4 — IANA Enterprise Number (LS-byte first)	This command is sent by the MM to the BMC instructs the BMC to enable or disable throttling the Processor Blade for thermal reasons when the Thermal Diode on the Processor(s) exceeds the Tc limit.
21h	Get Thermal Throttle Enable	**Request:** Byte 1:3 — IANA Enterprise Number (LS-byte first) **Response**: Byte 1 — Completion code. Byte 2:4 — IANA Enterprise Number (LS-byte first) Byte 4 – Enable Flag 7:1 – reserved 0:0 – enable/disable 0 = disabled 1 = enabled	This command is sent by the MM to the BMC and instructs the BMC to return the current setting of the throttle enable flag.
22h	Get Watt Usage	**Request:** Byte 1:3 — IANA Enterprise Number (LS-byte first) **Response**: Byte 1 — Completion code. 01h: Unidentifiable Hardware[4] (OEM Completion Code) 02h: Blade Chassis Mismatch[5] (OEM Completion Code) C0h-FFh: Generic completion codes[6]	This command is sent by the MM to the BMC and is a request for the BMC to provide the value in watts used per slot that make up the Processor Blade and attached expansion units. All power values are in 1 Watt increments (for example a value of x0002 would indicate 2 watts) If the blade cannot throttle then the minimum power is set equal to the

<table>
<tr><th colspan="4">Net Function = 0x2E</th></tr>
<tr><th>Code</th><th>Command</th><th>Request, Response Data</th><th>Description</th></tr>
<tr><td></td><td></td><td>Byte 2:4 — IANA Enterprise Number
(LS-byte first)
Byte 5 – Flag 0
7:5 – Reserved
4:1 – Component airflow increase[2]
0:0 = 0 pre-boot inventory
= 1 post power-on inventory
Byte 6 – Flag 1
7:0 - Reserved
Byte 7 - Blade Width[1]
Byte 8 – Slot 1 Standby Power Required
Byte 9 – Slot 1 Maximum Power Required(ms-byte)
Byte 10 – Slot 1 Maximum Power Required(ls-byte)
Byte 11 – Slot 1 Minimum Power(ms-byte) [3]
Byte 12 – Slot 1 Minimum Power(ls-byte) [3]

(repeat these 3 fields above for each based on width of blade including expansion modules)

Byte N – Slot X Standby Power Required
Byte N+1/N+2 – Slot X Maximum Power Required
Byte N+3/N+4 – Slot X Minimum Power</td><td>maximum power.

Note 1: This value is the number of slots that make up the processor blade complex that power information will be reported for, and is used to determine the number of tuples of power values.

Note 2: Setting this field will indicate to the MM to increase the minimum airflow to the blade. For example, if a blade had DIMM(s) that required more airflow than what would exist in a normal operating environment for this blade then this field should be set to the appropriate value.
<table>
<tr><th>Flag_0
Bits
4:3:2:1</th><th>Function</th></tr>
<tr><td>0000</td><td>Component(s) on blade do not require an increase in airflow to blade</td></tr>
<tr><td>0001</td><td>Component(s) on blade require a one level increase in airflow to the blade</td></tr>
<tr><td>0010</td><td>Component(s) on blade require a two level increase in airflow to the blade</td></tr>
<tr><td>0011-1111</td><td>Reserved and not supported</td></tr>
</table>
Note:3: Minimum amount of power the slot can consume. (For example this value would represent the power consumption when the blade is throttled to its lowest power utilization).

Note 4:
If the BMC detects presence of installed component(s) on the various I/O expansion connector(s) but is unable to read the VPD when this command is sent by the MM, the BMC will return a completion code of 01h and all other data in the message is considered invalid. Returning the 01h means the BMC can not determine the hardware type on at least one of the various I/O expansion connectors. This will cause the MM to deny power permission to the BMC. The MM will create an alarm and add an entry into CEL.

Note 5:</td></tr>
</table>

Net Function = 0x2E			
Code	**Command**	**Request, Response Data**	**Description**
			The BMC is sent information on the chassis type and subtype during blade initialization via *‘Chassis Information Command’*. If the BMC determines that the Processor Blade cannot be permitted to power on in a particular chassis type then the BMC must set a completion code of 02h. An example of the use of this completion code by the BMC may be that the processor blade has a thermal limitation in a particular chassis type. This will cause the MM to deny power permission to the BMC. The MM will create an alarm and add an entry into CEL. **Note 6:** If the BMC sends one of the generic completion codes the MM may retry the command for some limited number of times in expectation of a standard completion code of 00h. If a 00h completion code is never returned the MM will deny power permission to the BMC. The MM will create an alarm and add an entry into CEL.
23h	Get Throttle Status	**Request:** Byte 1:3 — IANA Enterprise Number (LS-byte first) **Response**: Byte 1 — Completion code. Byte 2:4 — IANA Enterprise Number (LS-byte first) Byte 5 – Throttle Reason Mask 7:2 – reserved 1:1 - Power 0:0 - TControl Byte 6 – Reserved Byte 7 – Blade Width[1] Byte 8 – Number of processors[2] Byte 9 – Processor Duty Cycle in percent[3] (repeates for the number of processor in Byte 8) Byte N – Throttle Power (msb-byte)[4] Byte N+1 – Throttle Power (lsb-byte)[4] (repeates for the blade width in Byte 7)	This command is sent by the MM to the BMC and instructs the BMC to return the current values for the reason the blade may be throttling, the current duty cycle of the processors and the current power throttle value. All power values are in 1 Watt increments(for example a value of x0002 would indicate 2 watts) **Note 1:** Number of slot throttle values in this message **Note 2:** This value represents the number of sockets that contain CPU(s) not the number of cores. **Note 3:** Value is a percent duty cycle. For example if the duty cycle varied from 100% to 50%. Then values would range from 64h to 32h. **Note 4:** Value of actual power being consumed for this slot.

<table>
<tr><th colspan="4">Net Function = 0x2E</th></tr>
<tr><th>Code</th><th>Command</th><th>Request, Response Data</th><th>Description</th></tr>
<tr>
<td>24h</td>
<td>Set Pre-Set Watts</td>
<td><b>Request:</b>
Byte 1:3 — IANA Modular Server Numb.
(LS-byte first)
Byte 4 – Flag_0
7:2 – Reserved
1:1 = 0 permission granted to continuing booting[2]
= 1 permission denied to continuing booting[2]
0:0 = 0 Throttle[1]
= 1 Un-throttle[1]
Byte 5 – Flag_1
7:0 - Reserved
Byte 6 - Blade Width[3]
Byte 7 – Slot 1 Throttle Value[4] (ms-byte)
Byte 8 – Slot 1 Throttle Value[4] (ls-byte)

(repeat these 2 fields above for each based on width of blade including expansion modules)

Byte N – Slot x Throttle Value[4] (ms-byte)
Byte N+1 – Slot x Throttle Value[4] (ls-byte)

<b>Response:</b>
Byte 1 — Completion code.
Byte 2:4 — IANA Modular Server Numb.
(LS-byte first)</td>
<td>This command is sent by the MM to the BMC and requests the BMC to change the throttle value of the processor(s) or to set up the BMC pre-set throttle value in case of a loss of the redundant power supply to the Blade.

All power values are in 1 Watt increments(for example a value of x0002 would indicate 2 watts)

<b>Note 1:</b>
<table>
<tr><th>Flag_ 0 Bit</th><th>No Power Failure* has been detected by the Blade</th><th>Power Failure* has been detected by Blade</th></tr>
<tr><td>0:0</td><td></td><td></td></tr>
<tr><td>0</td><td>BMC maintains current throttle setting of the processors</td><td>BMC will Invoke processor(s) to throttle or un-throttle to the value in Slot x Throttle Value field.</td></tr>
<tr><td></td><td colspan="2">BMC updates blades Pre-set Throttle Value with value in Slot x Throttle Value field.</td></tr>
<tr><td>1</td><td colspan="2">- BMC must completely un-throttle processor(s)
- BMC updates blades Pre-set Throttle Value with value in Slot x Throttle Value field.</td></tr>
</table>
*A 'Power Failure' is defined above as; assertion of an EPOW_N signal on one of the blades VHDM connectors OR a loss of 12V input power on one or the blades power connectors.

When a blade is in a power on state and has exchanged power information with the MM and no power failure condition exists the MM will either set Flag_0 - bit 0 to 0 (throttle) if presets are lower than the maximum blade power or set Flag_0 - bit 0 to 1 (un-throttle) if presets are set to maximum power.

When a power failure is detected by the blade, blade enters a power failure mode and must change the power usage level to the pre-set power value previously sent to the blade. In this failure mode the MM</td>
</tr>
</table>

<table>
<tr><th colspan="4">Net Function = 0x2E</th></tr>
<tr><th>Code</th><th>Command</th><th>Request, Response Data</th><th>Description</th></tr>
<tr><td></td><td></td><td></td><td>may subsequently send updates to the pre-set power level by setting Flag_0 - bit 0 to 0 (throttle) and blade must change to the new power level.
When the MM determines that the blades can completely un-throttle Flag_0 - bit 0 is set to 1 (un-throttle).
<b>Note 2:</b> Flag to indicate if the blade has permission to continue the boot process. If permission denied then blade must power-off.
<b>Note 3:</b> Number of slot throttle values in this message
<b>Note 4:</b> Value to be used to set the pre-set watt value and depending on state of 'Flag_0' then will be the maximum power value the slot can consume. (See table in Note 1)</td></tr>
<tr><td>25h</td><td>Get Pre-Set Watts</td><td><b>Request:</b>
Byte 1:3 — IANA Modular Server Numb.
(LS-byte first)
Byte 4 – Flag_0
7:2 – Reserved
1:1 = 0 permission granted to continuing booting[2]
= 1 permission denied to continuing booting[2]
0:0 = 0 Throttle[1]
= 1 Un-throttle[1]
Byte 5 – Flag_1
7:0 - Reserved
Byte 6 - Blade Width[3]
Byte 7 – Slot 1 Throttle Value[4] (ms-byte)
Byte 8 – Slot 1 Throttle Value[4] (ls-byte)
(repeat these 2 fields above for each based on width of blade including expansion modules)
Byte N – Slot x Throttle Value[4] (ms-byte)</td><td>This command is sent by the MM to the BMC and requests the BMC to return the current information on the throttle value of the processor(s) or the the BMC pre-set throttle value in case of a loss of the redundant power supply to the Blade.
All power values are in 1 Watt increments(for example a value of x0002 would indicate 2 watts)
<b>Note 1:</b>
<table>
<tr><th>Flag_ 0 Bit</th><th>No Power Failure* has been detected by the Blade</th><th>Power Failure* has been detected by Blade</th></tr>
<tr><td>0:0</td><td></td><td></td></tr>
<tr><td>0</td><td>BMC maintains current throttle setting of the processors</td><td>BMC will Invoke processor(s) to throttle or un-throttle to the value in Slot x Throttle Value field.</td></tr>
<tr><td></td><td colspan="2">BMC updates blades Pre-set Throttle Value with value in Slot x Throttle Value field.</td></tr>
<tr><td>1</td><td colspan="2">- BMC must completely un-throttle processor(s)
- BMC updates blades Pre-set Throttle Value with value in Slot x Throttle Value field.</td></tr>
</table></td></tr>
</table>

Net Function = 0x2E			
Code	**Command**	**Request, Response Data**	**Description**
		Byte N+1 – Slot x Throttle Value[4] (ls-byte) **Response:** Byte 1 — Completion code. Byte 2:4 — IANA Modular Server Numb. (LS-byte first)	*A ‘Power Failure’ is defined above as; assertion of an EPOW_N signal on one of the blades VHDM connectors OR a loss of 12V input power on one or the blades power connectors. When a blade is in a power on state and has exchanged power information with the MM and no power failure condition exists the MM will either set Flag_0 - bit 0 to 0 (throttle) if presets are lower than the maximum blade power or set Flag_0 - bit 0 to 1 (un-throttle) if presets are set to maximum power. When a power failure is detected by the blade, blade enters a power failure mode and must change the power usage level to the pre-set power value previously sent to the blade. In this failure mode the MM may subsequently send updates to the pre-set power level by setting Flag_0 - bit 0 to 0 (throttle) and blade must change to the new power level. When the MM determines that the blades can completely un-throttle Flag_0 - bit 0 is set to 1 (un-throttle). **Note 2:** Flag to indicate if the blade has permission to continue the boot process. If permission denied then blade must power-off. **Note 3**: Number of slot throttle values in this message **Note 4:** Value to be used to set the pre-set watt value and depending on state of ‘Flag_0’ then will be the maximum power value the slot can consume. (See table in Note 1)
26h	Set Component Watts Inventory	**Request:** Byte 1:3 — IANA Enterprise Number (LS-byte first) Byte 4 – Flag_0 7:4 – Reserved 3:0 = Component airflow increase request[1] Byte 5 – Reserved	This command is a request from firmware running on the host to the BMC and is used to update the processor blades maximum power values for CPU's and DIMM's. **Note 1:** Setting this field will indicate

<table>
<tr><th colspan="4">Net Function = 0x2E</th></tr>
<tr><th>Code</th><th>Command</th><th>Request, Response Data</th><th>Description</th></tr>
<tr><td></td><td></td><td>Byte 6 – Component Type CPU (00h)
Byte 7 – Maximum number of CPU(s) [4] that can be installed
Byte 8 – Power in watts when CPU at maximum utilization [2]
(repeat above field for each possible installed CPU)
Byte N – Component Type DIMM (01h)
Byte N+1 – Maximum number of DIMM(s) that can be installed
Byte N+2 – Power in watts for DIMM at maximum utilization [3]
(repeat above field for each possible installed DIMM)

Component Type Local Hard Disk Drive (02h) follows using format above: <i>type, max number of, power value (repeated for “max number of”)</i>

<b>Response</b>:
Byte 1 — Completion code.
Byte 2:4 — IANA Enterprise Number (LS-byte first)</td><td>to the MM to increase the minimum airflow to the blade. For example, if a blade had DIMM(s) that required more airflow than what would exist in a normal operating environment for this blade then these bits should be set to the appropriate value.

<table>
<tr><th>Flag_0 Bits 3:2:1:0</th><th>Function</th></tr>
<tr><td>0000</td><td>Component(s) on blade do not require an increase in airflow to blade</td></tr>
<tr><td>0001</td><td>Component(s) on blade require a one level increase in airflow to the blade</td></tr>
<tr><td>0010</td><td>Component(s) on blade require a two level increase in airflow to the blade</td></tr>
<tr><td>0011-1111</td><td>Reserved and not supported</td></tr>
</table>

<b>Note 2</b>: Power values in 1 watt increments. If CPU not installed this value should be set to 00h. If BIOS does not recognize the CPU then worst case power value number is used for the CPU family.

<b>Note 3:</b> Power values in 1 watt increments. If DIMM is not installed this value should be set to 00h. If BIOS determines DIMM is installed but currently unused (for example not configured or in a error state) then then worst case power value number is used.

<b>Note 4:</b> CPU information is per physical socket not per core.</td></tr>
<tr><td>27h</td><td>Get Component Watts Inventory</td><td><b>Request:</b>
Byte 1:3 — IANA Enterprise Number (LS-byte first)

<b>Response:</b>
Byte 1 — Completion code.
Byte 2:4 — IANA Enterprise Number (LS-byte first)
Byte 5 – Flag_0
7:4 – Reserved
3:0 = Component airflow increase request[1]
Byte 6 – Reserved</td><td>This command is a request for the BMC to provide the processor blades maximum power values for CPU's and DIMM's.

<b>Note 1</b>: Setting this field will indicate to the MM to increase the minimum airflow to the blade. For example, if a blade had DIMM(s) that required more airflow than what would exist in a normal operating environment for this blade then these bits should be set to the appropriate value..</td></tr>
</table>

<table>
<tr><th colspan="4">Net Function = 0x2E</th></tr>
<tr><th>Code</th><th>Command</th><th>Request, Response Data</th><th>Description</th></tr>
<tr><td></td><td></td><td>Byte 7 – Component Type CPU (00h)
Byte 8 – Maximum number of CPU(s) [4] that can be installed
Byte 9 – Power in watts when CPU at maximum utilization [2]
(repeat above field for each possible installed CPU)
Byte N – Component Type DIMM (01h)
Byte N+1 – Maximum number of DIMM(s) that can be installed
Byte N+2 – Power in watts for DIMM at maximum utilization [3]
(repeat above field for each possible installed DIMM)
Component Type Local Hard Disk Drive (02h) follows using format above: <i>type, max number of, power value (repeated for “max number of”)</i></td><td><table>
<tr><th>Flag_0 Bits
3:2:1:0</th><th>Function</th></tr>
<tr><td>0000</td><td>Component(s) on blade do not require an increase in airflow to blade</td></tr>
<tr><td>0001</td><td>Component(s) on blade require a one level increase in airflow to the blade</td></tr>
<tr><td>0010</td><td>Component(s) on blade require a two level increase in airflow to the blade</td></tr>
<tr><td>0011-1111</td><td>Reserved and not supported</td></tr>
</table>
<b>Note 2:</b> Power values in 1 watt increments. If CPU not installed this value should be set to 00h. If BIOS does not recognize the CPU then worst case power value number is used for the CPU family.
<b>Note 3:</b> Power values in 1 watt increments. If DIMM is not installed this value should be set to 00h. If BIOS determines DIMM is installed but currently unused (for example not configured or in a error state) then then worst case power value number is used.
<b>Note 4</b>: CPU information is per physical socket not per core.</td></tr>
<tr><td>28h</td><td>CPU Throttle Table Information</td><td><b>Request:</b>
Byte 1:3 — IANA Enterprise Number (LS-byte first)
Byte 4 – Cmd Revision value = 01h
Byte 5 – Reserved
Byte 6 – 01h (CPU 1 Throttle table Values in next 8 bytes) [1]
Byte 7 - Power for throttle step 8:
- Power value at lowest power consumption (maximum throttle)[2]
Byte 8 - Power for throttle step 7[2]
Byte 9 - Power for throttle step 6[2]
Byte 10 - Power for throttle step 5[2]
Byte 11 - Power for throttle step 4[2]
Byte 12 - Power for throttle step 3[2]
Byte 13 - Power for throttle step 2[2]
Byte 14 - Power for throttle step 1:
- Power value at maximum power consumption (no throttle) [2]</td><td>This command is a request from firmware running on the host to the BMC and is used to to update the BMC's information on the processor blades throttle table(s). This information is what the BMC utilizes when a situation exists to force the processor blade to throttle. A separate 28h command is issued to the BMC for each possible CPU.
<b>Note 1</b>: Number to identify which CPU the throttle table applies to. (CPU1 - 01h, CPU2 - 02h, CPU3 - 03h, CPU4 - 04h……)
<b>Note 2:</b> Power values are in watts and represent what the maximum possible CPU power utilization will be when the processor is throttled to this step.</td></tr>
</table>

Net Function = 0x2E			
Code	**Command**	**Request, Response Data**	**Description**
		Response: Byte 1 — Completion code. Byte 2:4 — IANA Enterprise Number (LS-byte first)	
29h-2Fh	reserved		
30h	Chassis Information Command	**Request:** Byte 1:3 — IANA Modular Server Numb. (LS-byte first) Byte 4: Chassis type code 61h = BC (Enterprise) 62h = BCT (Telco) Byte 5: Chassis sub-type code Chassis type code = 61h 20h[1] = BCE 00h[1] = BCE 02h = BCH 03h = BCS Chassis type code = 62h 20h[1] = BCT 00h[1] = BCT 02h = BCH-T Byte 6: Hardware Revision level of the chassis. 00h = unspecified Byte 7: MM sub-type 20h[1] = MM1 00h[1] = MM1 01h = AMM 10h = CMM2 Byte 8: Hardware Revision level of the MM. 00h = unspecified Byte 9-10[2]: MM firmware build revision and type from the build ID (see note for format). Byte 11-12[3]: The highest version of the Blade Base specification for	This command provides for exchange of information between the MM and BMC. The MM will send chassis information as well as MM information to the BMC and the BMC will respond with Blade Base Specification information. **Note 1:** This may be either 0x20 or 0x00 due to previous versions of VPD default values. **Note 2:** Only the build revision and type is included. For example, for MM build BRET79AUS, only 79A is used – byte 9 will be 79 (decimal) and byte 10 will be 0x41 (ASCII value for A). **Note 3:** If, for example, the supported Blade Base Specification version is 1.21, byte 11 will be 01h and byte 12 will be 15h. Unless otherwise stated in the bit description, capability bit values are defined as 1b = supported, and 0b = not supported. **Note 4:** Capability bit definitions may be found in Table 5-16 MM Capability Definitions.

Net Function = 0x2E			
Code	**Command**	**Request, Response Data**	**Description**
		processor blade subsystems supported by the MM (decimal). Byte 13:14: Reserved. Bytes 15:22 – Capabilities[4] **Response**: Byte 1 — Completion code. Byte 2:4 — IANA Modular Server Numb. (LS-byte first) Byte 5-6[3]: The highest version of the Blade Base specification for processor blade subsystems supported by the BMC (decimal)	
31h-4Fh	Reserved		
50h	Initiate Data Collection	**Request**: Byte 1:3 — IANA Modular Server Numb. (LS-byte first) Byte 4 — Data collection type 00h = Reserved 01h = BMC Dump 02h - 0FFh: Reserved **Response**: Byte 1 — Completion code Byte 2:4 — IANA Modular Server Numb. (LS-byte first)	This command provides a means for the MM to indicate to the BMC to initiate data (dump) collection on the ServerBlade. The dump data format and where the data is saved will be implementation dependent but should be allowed to be retrieved by an external application. When the ServerBlade supports this capability it must be reported as described in section 5.25.2 OEM Type 02h – Capability Report.
51h-FFh	Reserved		

Table 5-28 – FRU Device IDs for IBM VPD Read/Write OEM command

Device ID	Description	IPMI assigned Entity ID
0x00-0x04	Reserved	
0x05	Optional Expansion Board 1 (BSE, PEU)	0x10 – System Internal Expansion Board
0x07	CPU 1 (if CPU supported)	0x03 – Processor
0x08	CPU 2 (if CPU supported)	0x03 – Processor
0x09	CPU 3 (if CPU supported)	0x03 – Processor

0x0A	CPU 4 (if CPU supported)	0x03 – Processor
0x10	Planar VPD	0x07 – Main System Board
0x11	I/O Expansion Card 1 – I/O Expansion Card on main board with internal serdes links to chassis switch module 3 & 4	0x2C – I/O Module
0x12	I/O Expansion Card 2 - I/O Expansion card on 1st expansion board with internal serdes links to chassis switch module 1 & 2	0x2C – I/O Module
0x13	Reserved	
0x14	I/O Expansion Card 3 – I/O Expansion Card on 1st expansion board with internal serdes links to chaissis switches 3 & 4	0x2C – I/O Module
0x15	Reserved	
0x16	Optional Expansion Board 2 - BSE, PEU	0x10 – System Internal Expansion Board
0x17	I/O Expansion Card 4 – I/O Expansion Card on 2nd expansion board with internal serdes links to chassis switches 1 & 2	0x2C – I/O Module
0x18	I/O Expansion Card 5 – I/O Expansion Car on 2nd expansion board with internal serdes links to chassis switches 3 & 4	0x2C – I/O Module
0x19	Add-in Daughter Card (cKVM)	0x0B – Add-in Card
0x1A	High Speed Daughter Card	0x2C – I/O Module
0x1B-0xFF	Reserved	

Table 5-29 SEL Directed Event Format

Byte	Field	Description
1 2	SEL Record ID	For System Events, this value corresponds to the Record ID for the System Event record logged in the SEL. If the event was not logged in the SEL, this value is set to 00000h.
3	Directed Event Type	Set to 02h for SEL Directed Events.
4 5 6 7	Timestamp	Time when event was logged. LS byte first.

8 9	Generator ID	RqSA & LUN if event was generated from IPMB. Software ID if event was generated from system software. Byte 1 [7:1] - 7-bit I^2C . Slave Address, or 7-bit system software ID [0] 0b = ID is IPMB Slave Address 1b = system software ID Byte 2 [7:4] - Channel number. Channel that event message was received over. 0h if the event message was received via the system interface, primary IPMB, or internally generated by the BMC. (New for IPMI v1.5. These bits were reserved in IPMI v1.0) [3:2] - reserved. Write as 00b. [1:0] - IPMB device LUN if byte 1 holds Slave Address. 00b otherwise.
10	EvM Rev	Event Message format version (=**04h** for events in this specification)
11	Sensor Type	*For SEL Events*: Sensor Type Code for sensor that generated the event
12	Sensor #	*For SEL Events*: Number of sensor that generated the event
13	Event Dir | Event Type	*For SEL Events*: Event Dir [7] - 0b = Assertion event. 1b = Deassertion event. Event Type Type of trigger for the event, e.g. critical threshold going high, state asserted, etc. Also indicates *class* of the event. E.g. discrete, threshold, or OEM. The Event Type field is encoded using the Event/Reading Type Code. Refer to the *IPMI Specification*. [6:0] - Event Type Code
14	Event Data 1	*For SEL Events*: Per table *Event Request Message Event Data Field Contents* in the IPMI v1.5 Specification.
15	Event Data 2	*For SEL Events*: Per table *Event Request Message Event Data Field Contents* in the IPMI v1.5 Specification.
16	Event Data 3	*For SEL Events*: Per table *Event Request Message Event Data Field Contents* in the IPMI v1.5 Specification.
17		This field is reserved (set to 00h)

6 Environmental requirements

Shock and vibration test levels for blades, blade I/O expansion cards (daughter cards), and switch modules are defined in this section. Test levels for the BladeServer chassis are included for reference. It is recommended that tests at the blade, switch module, BladeServer chassis, and rack level be performed. However, in cases where this is not practical, blade and switch module testing using a rigid chassis test fixture as described below is, in most cases, acceptable.

Testing of the BladeServer chassis shall be performed at minimum and maximum chassis weight configurations. BladeServer; chassis testing shall be unpackaged, packaged, and in a rack. Testing of blades, blade I/O expansion cards, and switch modules shall be unpackaged, packaged, and in a chassis when practical (chassis unpackaged, packaged, and in a rack).

6.1 Pass/Fail Criteria

The product shall be inspected for mechanical damage after each test. Any noticeable damage is considered a failure. The product shall be operated before and after each test using an operating system and test exercise program to ensure it functions as designed.

6.2 Test Fixture

6.2.1 Operational Tests

For operational tests, the BladeServer chassis shall be clamped to the test table with rigid fixtures that support the chassis in a manner that simulates the support provided by the intended shipping package. Rack level operational vibration and shock testing is not performed.

6.2.2 Non-Operational Tests

For non-operational tests, the product shall be supported and clamped to the test table in a manner that simulates the support provided by the intended shipping package. For example, a blade should be mounted in a chassis which is clamped to the test table with rigid fixtures that support the BladeServer chassis as the shipping package does.

If a BladeServer chassis is not available, a rigid chassis test fixture for blades and switch modules may be used. This fixture shall be a rigid structure designed to simulate the support and retention mechanisms of the BladeServer. This structure should have a natural frequency of greater than 200 Hz. It shall provide the same guide channels, latch retention and connector system as a chassis. The signal and power connectors should be mounted on a board or other structure that will simulate the BladeServer midplane.

6.3 Operational Shock and Vibration

The following operational shock and vibration tests shall be performed on a test table. The tests are not performed in a rack.

6.3.1 Operational Shock

Server on, operational shock

- Vertical Input: 30.0 G for 3ms, half-sine shock pulse
- Horizontal Input: 15.0 G for 3ms, half-sine shock pulse

- Two shock inputs in each axis, one in each direction; six total

6.3.2 Operational Vibration

Server on, operational vibration: 0.27 G RMS at 5Hz to 500Hz for 30 minutes

Power Spectral Density (PSD) for operational vibration tests is provided in Table 6-1.

Frequency	G^2 / Hz (PSD Level)
5.0	$2.0x10^{-5}$
17.0	$3.0x10^{-4}$
45.0	$3.0x10^{-4}$
48.0	$3.0x10^{-4}$
62.0	$3.0x10^{-4}$
65.0	$3.0x10^{-4}$
150.0	$3.0x10^{-4}$
200.0	$8.0x10^{-5}$
500.0	$8.0x10^{-5}$

Table 6-1 Random Vibration PSD Profile for Chassis, Blade and Switch Module Operational Unpackaged Test

6.4 Non-operational Shock and Vibration

Non-operational chassis, blade, blade IO adapter card, and switch module vibration and shock test levels are defined below. Chassis level tests shall be performed with and without blades. In each case, the chassis shall be fully populated with modules. Chassis tests shall be performed using shock and vibration test tables and in a rack. Note that if blade level testing is performed in a BladeServer chassis, a maximum of 6 blades shall be tested at any one time. It is intended that blades, blade IO adapter cards, and switch modules be tested as individual units and in the chassis. However, for companies developing blades, blade IO adapter cards, and switch modules, an alternate test method is to use a rigid test fixture as described above, and test only to individual blade and switch module levels. This test method will result in a reasonable assurance that failure will not occur. However, if practical, it is recommended that chassis level testing be performed.

6.4.1 Non-operational Fragility Random Vibration

Unpackaged (fragility) random vibration tests are performed in accordance with levels in Table 6-2. These levels are for the chassis, blades, blade IO adapter cards, and switch modules. If the product fails during random vibration testing, additional sinusoidal vibration and dwell testing shall be used to determine weak areas of the product. The sinusoidal test shall consist of a sweep at 0.5G from 2Hz to 200Hz to determine the most dominant natural frequency then dwell at the natural frequency for 15 minutes. Once improvements are made to the design, the unpackaged random vibration tests shall be re-run to ensure compliance to the test levels.

Orientation	GRMS	Duration (minutes)
Bottom	1.463	15
Top	1.463	15
Right	1.463	15
Left	1.463	15
Front	1.463	15
Rear	1.463	15

Table 6-2 Non-operational Unpackaged - Chassis, Blade, Blade IO Adapter Card, and Switch Module

The random vibration test spectrum for non-operational unpackaged testing shall be in accordance with Table 6-3.

Frequency	G^2 / Hz (PSD Level)
2.0	0.0010
4.0	0.0300
8.0	0.0300
40.0	0.0100
200.0	0.0100

Table 6-3 1.463 GRMS Random Vibration Spectrum for Chassis, Blades, and Switch Module Tests

6.4.2 Non-operational Fragility Shock - Unpackaged Chassis

Table 6-4 defines the unpackaged chassis shock test levels. BladeServer does not pass an unpackaged 35 G rear drop at a change in velocity of 3.459 mm/s with 14 blades. Therefore, for blade testing, no more than 6 blades shall be tested in an unpackaged chassis at any one time.

Orientation	G's	Delta-V mm/sec (in./sec)	Wave Form
Bottom	35	3,459 (136.19)	Trapezoid
Top	35	3,459 (136.19)	Trapezoid
Right	35	3,459 (136.19)	Trapezoid
Left	35	3,459 (136.19)	Trapezoid
Front	35	3,459 (136.19)	Trapezoid
Rear	35	3,459 (136.19)	Trapezoid

Table 6-4 Non-Operational Unpackaged Fragility Chassis Shock Test Levels

6.4.3 Non-operational Fragility Shock – Unpackaged Blade and Switch Module

Unpackaged blade and Switch module shock test levels are defined in Table 6-5.

Blades and switch modules may be qualified using a rigid chassis test fixture, as described in the test fixture section of this document, in lieu of the chassis level test described in the section above. The levels in Table 6-5 shall be used to test blades and switch modules in the rigid chassis test fixture.

Orientation	G's	Delta-V mm/sec (in./sec)	Wave Form
Bottom	50	4,572 (180)	Trapezoid
Top	50	4,572 (180)	Trapezoid
Right	50	4,572 (180)	Trapezoid
Left	50	4,572 (180)	Trapezoid
Front	50	4,572 (180)	Trapezoid
Rear	50	4,572 (180)	Trapezoid

Table 6-5 Non-Operational Unpackaged Blade and Switch Module Shock Test Levels

6.4.4 Non-operational Packaged Random Vibration

Packaged random vibration tests are performed in accordance with levels in Table 6-6. These levels are for the chassis, blades, blade IO adapter cards, and switch modules.

Orientation	GRMS	Duration (minutes)
Top or Bottom	1.463	15
Right or Left	1.463	15
Front or Rear	1.463	15

Table 6-6 Non-operational Packaged - Chassis, Blade, Blade IO Adapter Card, and Switch Module

The random vibration test spectrum for non-operational unpackaged testing shall be in accordance with Table 6.3.

6.4.5 Non-operational Packaged Shock – BladeServer Chassis

Table 6-7 lists the chassis packaged drop test levels. Product acceleration response inside the package must be monitored during the test. The response must be lower than the unpackaged fragility acceleration target.

Orientation	Drop Height mm (inches)
Bottom	610 (24)
Top	610 (24)
Right	610 (24)
Left	610 (24)
Front	610 (24)
Rear	610 (24)

Table 6-7 Non-operational Packaged Chassis Drop Test Levels

6.4.6 Non-operational Shock – Packaged Blade, IO Adapter Card and Switch Module

Table 6-8 lists the drop heights for specific weight ranges to be used to test packaged blade, IO adapter cards, and switch modules. Product acceleration response inside the package must be monitored during the test. The response must be lower than the unpackaged fragility acceleration target.

Orientation	**Drop Height mm (inches)** Weight < 9.1 Kg (20 lbs)	**Drop Height mm (inches)** Weight = 9.1 Kg (20 lbs) to 18.2 Kg (40 lbs)	**Drop Height mm (inches)** Weight = 18.2 Kg (40 lbs) to 36.4 Kg (80 lbs)
Bottom	914 (36)	762 (30)	610 (24)
Top	914 (36)	762 (30)	610 (24)
Right	914 (36)	762 (30)	610 (24)
Left	914 (36)	762 (30)	610 (24)
Front	914 (36)	762 (30)	610 (24)
Rear	914 (36)	762 (30)	610 (24)
Critical Corner	914 (36)	762 (30)	610 (24)
Critical Edge	914 (36)	762 (30)	610 (24)

Table 6-8 Non-operational Packaged Blade, Blade IO Adapter Card and Switch Module Drop Test Levels

6.4.7 Non-operational Shock – Chassis, Blades, IO Adapter Cards and Switch Modules Installed in a Rack

Table 6-9 lists the non-operational rack shock test levels for a chassis with blades and switch modules installed.

Drop Height mm (in.)	**Wave Form**	**Duration**	**Target Delta V mm/sec (in./sec)**	**# of Drops**	**Drop Description**
152.4 (6)	Half Sin	< 3 ms	1,730 (68.09)	2	Bottom Face (Shipping Orientation)
50.8 (2)	Half Sin	< 3 ms	998.5 (39.31)	10	Bottom Face (Shipping Orientation)

Table 6-9 Non-operational Rack Shock Test Levels for Chassis with Blades and Switch Modules Installed

6.4.8 Non-operational Random Vibration - Rack

Random rack vibration tests shall be tested to the levels described below. If the product fails during random vibration testing, additional sinusoidal vibration and dwell testing shall be used to determine weak areas of the product. The sinusoidal test shall consist of a sweep at 0.5G from 2Hz to 200Hz to determine the most dominate natural frequency then dwell at the natural frequency for 15 minutes. Once improvements are made to the design, the unpackaged random vibration tests shall be re-run to ensure compliance to the test levels.

Orientation	**GRMS**	**Duration (minutes)**
Bottom	1.04	15

Table 6-10 Non-operational Unpackaged – Rack Testing with Chassis, Blade, Blade IO Adapter Card, and Switch Modules

The random vibration test spectrum for non-operational rack testing shall be in accordance with Table 6-11.

Frequency	G^2 / Hz (PSD Level)
2.0	0.0010
4.0	0.0300
8.0	0.0300
40.0	0.0030
55.0	0.0100
70.0	0.0100
200.0	0.0010

Table 6-11 1.04 GRMS Random Vibration Spectrum for Rack Tests

6.5 Telecom Environment

The following telecommunication listings must be supported by the blade server design. The host chassis in a telecom environment will be either the Telco 8U BladeServer or Telco 12U BladeServer chassis.

The BladeServer-T and BladeServer-HT chassis have been designed and tested for compliance to NEBS Level 3, Earthquake Zone 4 criteria as defined by Telcordia SR-3580, as well as ETSI criteria as defined by various documents released by the European Telecommunications Standards Institute.

NEBS Level 3, Earthquake Zone 4 is considered the minimal level of environmental compatibility to provide maximum assurance of network reliability within the CO environment even under harsh and catastrophic conditions. The Level 3 requirements are based on the following documents:

- Telcordia GR-63-CORE; Physical Protection, Issue 3
- Telcordia GR-78-CORE; Physical Design and Manufacture of Telecommunications Products and Equipment, Issue 1
- Telcordia GR-1089-CORE; Electromagnetic Compatibility and Electrical Safety, Issue 4

Telcordia NEBS standards documents may be purchased from the Telcordia Information Superstore at http://telecom-info.telcordia.com/site-cgi/ido/index.html.

For compliance to European Telecommunications Standards, the following ETSI document should be consulted (may be freely downloaded from http://www.etsi.org):

- **ETSI 300 386 V1.3.3 (2005-04) –** Electromagnetic compatibility and Radio spectrum Matters (ERM); Telecommunication network equipment; ElectroMagnetic Compatibility (EMC) requirements