Intel® NetStructure™ MPCMM0001 Chassis Management Module

Software Technical Product Specification

April 2005

Order Number: 273888-007

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

Designers must not rely on the absence or characteristics of any features or instructions marked “reserved” or “undefined.” Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them.

The Intel® NetStructure™ MPCMM0001 Chassis Management Module may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

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

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

Contents

Figures

Tables

Revision History

Date	Revision	Description
April 2005	007	Firmware version 5.2
August 2004	006	Firmware version 5.1.0.757
April 2004	005	Version 5.1 TPS Added Re-Enumeration Section Added Process Monitoring Section
January 2004	004.1	Version 4.1 TPS

Introduction 1

1.1 Overview

The Intel® NetStructure™ MPCMM0001 Chassis Management Module is a 4U, single-slot CMM intended for use with AdvancedTCA* PICMG* 3.0 platforms. This document details the software features and specifications of the CMM. For information on hardware features for the CMM refer to the *Intel® NetStructure™ MPCMM0001 Hardware Technical Product Specification*. Links to specifications and other material can be found in Appendix B, "Data Sheet Reference."

The CMM plugs into a dedicated slot in compatible systems. It provides centralized management and alarming for up to 16 node and/or fabric slots as well as for system power supplies, fans and power entry modules. The CMM may be paired with a backup for redundant use in high-availability applications.

The CMM is a special purpose single board computer (SBC) with its own CPU, memory, PCI bus, operating system, and peripherals. The CMM monitors and configures IPMI-based components in the chassis. When thresholds (such as temperature and voltage) are crossed or a failure occurs, the CMM captures these events, stores them in an event log, sends SNMP traps, and drives the Telco alarm relays and alarm LEDs. The CMM can query FRU information (such as serial number, model number, manufacture date, etc.), detect presence of components (such as fan tray, CPU board, etc.), perform health monitoring of each component, control the power-up sequencing of each device, and control power to each slot via IPMI.

Assumptions: This document assumes some basic Linux* knowledge and the ability to use Linux text editors such as vi.

1.2 Terms Used in this Document

Table 1. Glossary (Sheet 1 of 2)

Acronym	Description
BIST	Built-In Self Test
CDM	Chassis Data Module
CLI	Command Line Interface
CMM	Chassis Management Module
DHCP	Dynamic Host Configuration Protocol
FFS	Flash File System
FIS	Flash Image System
FPGA	Field-Programmable Gate Arrays
FRU	Field Replaceable Unit
HS	Hot Swap
IPMI	Intelligent Platform Management
IPMB	Intelligent Platform Management Bus

Table 1. Glossary (Sheet 2 of 2)

Acronym	Description
IPMI	Intelligent Platform Management Interface
LED	Light Emitting Diode
MIB	Management Information Base
MIB II	RFC1213 - A standard Management Information Base for Network Management
PEM	Power Entry Module
PICMG	PCI Industrial Computer Manufacturers' Group
RMCP	Remote Management Control Protocol
RPC	Remote Procedural Calls
SBC	Single Board Computer
SDR	Sensor Data Record
SEL	System Event Log
ShMC	Shelf Management Controller
SNMP	Simple Network Management Protocol
SSH	Secure Socket Shell
TFTP	Trivial File Transfer Protocol
UDP	User Datagram Protocol
WDT	Watchdog Timer

Software Specifications 2

2.1 Red Hat* Embedded Debug and Bootstrap (Redboot)

Upon initial power on, the CMM enters into the Redboot firmware to bootstrap the embedded environment. Upon execution, Redboot acts as a TFTP server and checks for a TFTP connection to a client. If a TFTP connection exists, Redboot will accept a firmware update that is pushed down from the client, check the firmware update for data integrity, and then write the update to the flash.

Note: Firmware updates using the Redboot TFTP method are supported for backwards compatibility. However, updating from within the OS using the CLI is the preferred method of updating CMM firmware. For information on the firmware update process refer to Section 23, “Updating CMM Software” on page 204.

Under normal circumstances, Redboot runs through the standard diagnostics, memory setup, decompresses the OS kernel, and boots into that kernel.

2.2 Operating System

The CMM runs a customized version of embedded BlueCat* Linux* 4.0 on an Intel® 80321 processor with Intel® XScale® technology. Development support for BlueCat Linux is available on the web at http://www.lynuxworks.com.

2.3 Command Line Interface (CLI)

The Command Line Interface (CLI) connects to and communicates with the intelligent management devices of the chassis, boards, and the CMM itself. The CLI is an IPMI-based library of commands that can be accessed directly or through a higher-level management application. Administrators can access the CLI through Telnet, SSH, or the CMM’s serial port. Using the CLI, users can access information about the current state of the system including current sensor values, threshold settings, recent events, and overall chassis health, access and modify shelf and CMM configurations, set fan speeds, perform actions on a FRU, etc. The CLI is covered in Section 8, “The Command Line Interface (CLI)” on page 71.

2.4 SNMP/UDP

The chassis management module supports both queries and traps on SNMP (Simple Network Management Protocol) v1 or v3. The SNMP version can be configured through the CLI interface. The default is for SNMP v1. A MIB for the entire platform is included with the CMM. The CMM can send out SNMP traps to up to five trap receivers.

Along with SNMP traps, the CMM sends UDP (User Datagram Protocol) alerts to port 10000. The content of these UDP alerts is the same as the SNMP traps. SNMP is covered in Section 17, “SNMP” on page 140.

2.5 Remote Procedural Call (RPC) Interface

In addition to the console command-line interface, the CMM can be administered by custom remote applications via remote procedure calls (RPC). RPC is covered in Section 19, “Remote Procedure Calls (RPC)” on page 174.

2.6 RMCP

RMCP (Remote Management Control Protocol) is a protocol that defines a method to send IPMI packets over LAN. The RMCP server on the CMM can decode RMCP packages and forward the IPMI messages to the appropriate channels including: SBC blades, PEMs, and FanTrays or local destination within the CMM. When there is a responding IPMI message coming from SBC blades, PEMs, or FanTrays destined to RMCP client, the RMCP server will format this IPMI message into a RMCP message and send it to through the designated LAN interface back to originator. RMCP is covered in Section 20, “RMCP” on page 190.

2.7 Ethernet Interfaces

The CMM contains two Ethernet ports. The software can configure each of these ports to either the front panel, to the backplane, or to the rear transition module (RTM). Information on configuring the Ethernet interfaces is covered in Section 8.3.1, “Setting IP Address Properties” on page 72.

2.8 Sensor Event Logs (SEL)

The AdvancedTCA CMM implements system event logs according to Section 3.5 of the PICMG 3.0 Specification. The SEL contained on the CMM is fully IPMI compliant.

2.8.1 CMM SEL Architecture

The MPCMM0001 uses a single flat SEL file stored locally in the /etc/cmm directory. The SEL maintains a list of all the sensor events in the shelf. Each of the managed devices may keep its own SEL records in local SELs, but the master copy for the shelf is maintained by the CMM.

The SEL is limited to 65536 bytes. In order to keep the SEL from getting full, which can cause loss of error logging, the SEL is checked every 15 minutes by the CMM, and if the size of the cmm_sel is greater than 40000 bytes, the SEL is archived in gzip format and saved in /home/log/SEL. The names of the saved logs will be cmm_sel.0.gz, cmm_sel.1.gz, and so on, to a maximum of 16 logs where they are then rolled over.

Note: Archived files should NEVER be decompressed on the CMM as the resulting prolonged flash file writing could disrupt normal CMM operation and behavior. Using FTP, transfer the files to a different system before decompressing the archive using utilities such as gzip.

2.8.2 Retrieving a SEL

To retrieve a SEL from the CMM, issues the following command:

```
cmmget [-l location] -d sel
```

Where location is one of {cmm, blade[1-14], fantray1, PEM[1-2]}. Even though the CMM uses a single flat SEL for system events, the 'cmmget' command will filter the SEL and only return events associated with the provided location. Also, some individual FRUs may keep their own local SELs (i.e., blades).

2.8.3 Clearing the SEL

The following command will clear the SEL on both the active and the standby:

```
cmmset -d clearsel -v clear
```

Note: Since the CMM uses a single flat SEL for system events, this command clears the entire shelf SEL, not just a filtered subset.

2.8.4 Retrieving the Raw SEL

To retrieve the SEL in its raw format from a location, issue the following command:

```
cmmget -l [location] -d rawsel
```

2.9 Blade OverTemp Shutdown Script

The CMM software includes predefined script settings specifically for the MPCBL0001 board, which will automatically shut down a board when the "baseboard temp" sensor on that board crosses the upper critical threshold. This is done to prevent a runaway thermal event on the board from occurring. If this functionality is needed when using boards other than the MPCBL0001, the user will need to associate the name of the thermal sensor and the threshold with the board shutdown script:

```
cmmset -l bladeN -d majoraction -t [temp sensor name] -v
overtempbladepoweroff [Blade Number]
```

Please refer to Section 18, "CMM Scripting" on page 164 for more information on assocating a script to an event.

When using the CMM with boards other than the MPCBL0001, as long as there is no sensor name titled "baseboard temp" associated with the particular board being used, then there is no issue leaving these settings intact. If needed, to deactivate these settings for each physical slot, use the command:

```
cmmset -l bladeN -d majoraction -t “baseboard temp” -v none
```

where bladeN is the blade, corresponding to the physical slot number, on which to remove the automatic shutdown setting (blade[1-16]). Please refer to Section 18, "CMM Scripting" on page 164 for more information on removing script actions.

Redundancy, Synchronization, and Failover 3

3.1 Overview

The CMM supports redundant operation with automatic failover in a chassis using redundant CMM slots. In systems where two CMMs are present, one acts as the active shelf manager and the other as standby. Both CMMs monitor each other, and either can trigger a failover if necessary.

Data from the active CMM is synchronized to the standby CMM whenever any changes occur. Data on the standby CMM is overwritten. A full synchronization between active and standby CMMs occurs on initial power up, or any insertion of a new CMM.

The active CMM is responsible for shelf FRU information management when CMMs are in redundant mode.

3.2 Synchronization

To ensure critical files on the standby CMM match the data on the active CMM, the active CMM synchronizes its data with the standby CMM, overwriting any existing data on the standby CMM.

An exception to this is the password reset procedure, detailed in Section 9, “Resetting the Password” on page 99. When the password reset switch is activated on the standby CMM, the password will be synchronized to the active CMM.

The CMMs will initially fully synchronize data from the active to the standby CMM just after booting. An insertion of a new CMM will also cause a full synchronization from the active to the newly inserted standby. Date and time are synched every hour. Partial synchronization will also occur any time files are modified or touched via the Linux* “touch” command with the exception of all *.sif and *.bin files in the /etc/cmm directory.

The *.sif (ALL SIF files), and *.bin (SDR Files) files under /etc/cmm are synchronized only once (when the CMMs establish communication). A 'touch' on those files at any later time will not perform a sync operation. Also, any updates to these files always happen as part of the software updates and not in isolation.

Note: During synchronization, the health event LEDs on the standby CMM may blink on and off as the health events that were logged in the SEL are synchronized.

Below is a list of items that are synchronized between CMMs. During a full synchronization, all of these files and data are synchronized. A change to any of these files results in that file being synched. The active CMM overwrites these files on the standby CMM.

There are two "levels" of files that get synchronized. In order to normally manage the chassis, the priority 1 files must be synchronized after power up or installation of a brand new CMM into the chassis. It is absolutely necessary that a standby CMM has the priority one files synched before a successful failover can occur. When a brand new CMM boots the first time as a standby, if a CMM failover is forced before all priority 1 data items are synchronized to the standby CMM, the standby CMM can still become the active CMM but may not be able to properly manage the FRUs in the chassis.

Table 2. CMM Synchronization (Sheet 1 of 2)

File(s) or Data	Description	Path	Priority
date and time	Date and time	IPMB	1
IP Address Settings	CMM eth1, eth1:1, and eth0 IP address settings to allow CMMs to discover the other's IP information.	IPMB	1
/etc/cmm.cfg	CMM's main configuration file	Ethernet	1
/etc/cmm/cmm_sel	System SEL	Ethernet	1
/etc/cmm/sensors.ini	Sensor Set Values	Ethernet	1
Ekey Controller Structures	Ekey Controller Structures	Ethernet	1
Bused EKey Token info	Bused EKey Token info	Ethernet	1
IPMB User States	IPMB User States	Ethernet	1
Fan States	Fan States	Ethernet	1
Cooling State	Cooling State Information	Ethernet	1
User LED States	User LED States	Ethernet	1
SDR structures and SIPI Controller Info	SDR structures and SIPI Controller Info	Ethernet	1
PHM FRU state, Power Usage and Power Info	PHM FRU state, Power Usage and Power Info	Ethernet	1
FIM FRU Cache (Local and Temp)	FIM FRU Cache (Local and Temp)	Ethernet	1
SEL Time	SEL Time	IPMB	1
SEL Events	Individual SEL Events	IPMB	1
/etc/cmm/fantray.cfg	Fantray settings needed by cooling manager	Ethernet	1
/etc/cmm.ini	Provides configuration values like the bus mapping	Ethernet	2
/etc/passwd	Password file	Ethernet	2
/etc/shadow	Password file	Ethernet	2
/etc/cmdPrivillege.ini	Provides privilege related configuration values for RMCP	Ethernet	2
/etc/cmm/*.bin	All SDR Files	Ethernet	2
/etc/cmm/*.sif	All SIF Files	Ethernet	2
/etc/var/snmpd.conf	SNMP configuration files	Ethernet	2
/etc/snmpd.conf	SNMP configuration files	Ethernet	2
/home/scripts	Entire user scripts area	Ethernet	2
Prompt file	Prompt file	Ethernet	2
/etc/actionscripts.cfg	Event action settings	Ethernet	2

Table 2. CMM Synchronization (Sheet 2 of 2)

File(s) or Data	Description	Path	Priority
Issues files	Issues files	Ethernet	2
/usr/local/cmm/temp/pmssync.ini	Recovery Action and escalation action for all the monitored processes except monitor process	Ethernet	2
/usr/local/cmm/temp/pmsshadowsync.ini	Recovery action and escalation action for monitor process	Ethernet	2

Note: The /.rhosts file is used for synchronization and should NEVER be modified.

3.3 Heterogeneous Synchronization

Beginning in version 5.2 firmware, the CMM can synchronize data between differing CMM versions. The firmware delineates synchronization from firmware versioning, thus allowing seamless synchronization between all CMM versions. A form of internal data versioning maintained by the CMM helps achieve this.

Note: SDR/SIF and user scripts differ slightly in synchronization architecture as described below.

3.3.1 SDR/SIF Synchronization

Sensor Data Records (SDRs) and Sensor Information Files (SIFs) will be synchronized only between CMMs having the same version for this data item (even if the CMM firmware versions differ).

3.3.2 User Scripts Synchronization and Configuration

By default, user scripts are synchronized only between CMM’s with same firmware versions. User can control the user scripts synchronization irrespective of CMM version differences by modifying the value of a configuration flag - "SyncUserScripts" (in the CMM configuration file, cmm.cfg under /etc). The configuration flag can be modified using the cmmget/cmmset commands. This flag can be read/set through any of the CMM interfaces (i.e., CLI, SNMP and RPC).

Only when CMM firmware versions differ will the value of this flag determines if user scripts should be synchronized or not. Between same firmware versions, the user scripts directory will continue to be synchronized and this flag ignored.

3.3.2.1 Setting User Scripts Sync Configuration Flag

To set the value of the Scripts Synchronization configuration flag, the following CMM command is used:

```
cmmget -l cmm -d syncuserscripts -v [equal/upgrade/downgrade/always]
```

Where:

equal: Synchronizes user scripts only when the CMM versions are same. This is the ***default value***.

upgrade: Synchronizes user scripts only when the other CMM has a newer firmware version.

downgrade: Synchronizes user scripts only when the other CMM has an older firmware version.

always: Synchronizes user scripts irrespective of version differences.

3.3.2.2 Retrieving User Scripts Sync Configuration Flag

To retrieve the value of the Scripts Synchronization configuration flag, the following CMM command is used:

```
cmmget -l cmm -d syncuserscripts
```

The value returned will be one of: Equal, Upgrade, Downgrade, Always, or Error on failure.

3.3.3 Synchronization Requirements

For synchronization to occur:

- The CMMs must be able to communicate with each other over their dedicated IPMB. The CMMs use a heartbeat via their dedicated IPMB to determine if they can communicate with each other over IPMB.
- An Ethernet connection must exist between the two CMMs. The CMMs must be able to ping each other via Ethernet for synchronization to be successful. This can be a connection through the Ethernet switches in the chassis, which requires both switches to be present in the chassis; a connection can occur through an external Ethernet switch connected to the front ports of the CMM pair, or alternatively, the connection can be a crossover cable connecting the two front ports of the CMM pair. If synchronization fails on eth1, then it will be attempted on eth0. If the CMMs cannot successfully ping each other via eth0 or eth1, then synchronization between the CMMs cannot occur.

A failure of any priority 1 synchronization will result in a health event being logged in the CMM SEL and will inhibit a failover from occurring.

3.4 Initial Data Synchronization

It is absolutely necessary that a standby CMM has the priority one files synched before a successful failover can occur. A standby CMM can still become active if all priority one synchronization has not been completed, but it may not be able to properly manage all the FRU’s in the chassis.

The CMM implements the “Datasync Status” sensor to determine the state of synchronization and if synchronization has completed. successfully.

3.4.1 Initial Data Sync Failure

If CMM encounters any failure during data synchronization it marks the data synchronization failure and logs a SEL event and sends an SNMP trap. Duplicate failures are not reported multiple times. As soon as CMM is out of failure condition it will reset data synchronization failure state.

The CMM will continue trying to synchronize as long as there are two CMMs present in the chassis and they are able to communicate via their cross-connected IPMB.

3.5 Datasync Status Sensor

A sensor named “Datasync Status” exists in order to make the Datasync state information available to the user. This sensor tracks the status of the Datasync module and will make its status available through the various CMM interfaces. This sensor is used to query the data synchronization states, and log SEL events for initial synchronization complete event. It is a discrete OEM sensor with status bits representing the state of different parts of the Datasync module.

Note: The Datasync Status sensor can only be queried through the active CMM.

3.5.1 Sensor bitmap

When the Datasync starts the first time through in a dual CMM system and whenever the CMM changes between Active and Standby, the status bits are all cleared to 0x0000.

- Bit 0 (Running) is set when the datasync module is active.
- Bit 1 (P1Done) is set when the priority 1 data syncs are done, and cleared when priority 1 data needs to be synced.
- Bit 2 (P2Done) is set when the priority 2 data syncs are done, and cleared when a priority 2 data needs to be synced.
- Bit 3 (InitSyncDone) is set when both priority 1 and priority 2 data syncs are done, and stays set (latches) until the CMM changes between Active and Standby, or looses contact with the partner CMM.
- Bit 4 (SyncError) is set if an error was detected, and cleared when no data items have errors.

3.5.2 Event IDs

The “Datasync Status” sensor will use event ids 0x420 to 0x42f. The following new event ids are used to log various events for these requirements. These event ID’s can be used to associated scripts with the respective events.

Event	Event ID
Initial Data Synchronization complete	0x420 (1056)

3.5.3 Querying the Datasync Status

The status of the data synch sensor can be queried using the following CLI command:

```
cmmget –l cmm –t “Datasync Status” –d current
```

Output of the command is as follows:

Initial State:

```
The current value is 0x0001

Initial Data Synchronization is not complete.

   There is Priority 1 data to sync.
   There is Priority 2 data to sync.

No Data Synchronization problems known.
```

Initial Data Synch Incomplete, Pri 1 Data Synced, Pri 2 Data Not Synched

```
The current value is 0x0003

Initial Data Synchronization is not complete.

   Priority 1 data is synced.

   There is Priority 2 data to sync.

No Data Synchronization problems known.
```

Initial Data Sync is complete, Priority 1 and Priority 2 are also synced

```
The current value is 0x000f

Initial Data Synchronization is complete.

   Priority 1 data is synced.

   Priority 2 data is synced.

No Data Synchronization problems known.
```

Initial Data Sync failure

```
The current value is 0x0013

Initial Data Synchronization is not complete.

   Priority 1 data is synced.

   There is Priority 2 data to sync.

Data Synchronization has encountered a problem in synchronizing data.
```

Initial Data Sync is complete and Priority 1 data is changed

```
The current value is 0x000d

Initial Data Synchronization is complete.

   There is Priority 1 data to sync.

   Priority 2 data is synced.

No Data Synchronization problems known
```

Data Sync failure of Priority 1 Data occurs after Initial Data Sync and there is a Data Sync Problem

```
The current value is 0x001d

Initial Data Synchronization is complete.

   There is Priority 1 data to sync
   Priority 2 data is synced.
Data Synchronization has encountered a problem in synchronizing data.
```

Data Sync becomes normal after Data Sync failure

```
The current value is 0x000f
Initial Data Synchronization is complete.
   Priority 1 data is synced.
   Priority 2 data is synced.
No Data Synchronization problems known
```

Single CMM

```
The current value is 0x0000
Datasync disabled - there is no partner CMM present.
```

3.5.4 SEL Event

The Datasync Status sensor generates the following two SEL events:

- When the active CMM is or becomes the only CMM, or the active CMM loses communication with the standby CMM, the following event will be logged:

```
[Day] [Month] [Date] [Time] [Year]
     CMM[n]: CMM Datasync Status    Initial Data Synchronization is
complete. Deasserted
```

- The following event will be logged in the SEL when initial data synchronization is complete:

```
[Day] [Month] [Date] [Time] [Year]
     CMM[n]: CMM Datasync Status    Initial Data Synchronization is
complete. Asserted
```

Where

n: The number of the CMM generating the event.

3.5.5 SNMP Trap

The Datasync Status sensor generates following two SNMP traps:

- When the active CMM is or becomes the only CMM, or the active CMM loses communication with the standby CMM, the following SNMP trap will be generated.

```
[Month] [Date] [Time] [hostname] snmptrapd[xxxxx]: [IP Address]:
Enterprise Specific Trap (25) Uptime: [Time], SNMPv2-
SMI::enterprises.343.2.14.1.5 = STRING: "Time : [Day] [Month] [Date]
[Time] [Year], Location : [location] , Chassis Serial # : [xxxxxxxx],
Board : CMM[x] , Sensor : Datasync Status , Event : Initial Data
Synchronization  complete: Deasserted "
```

- When initial data synchronization is complete, the following SNMP trap is generated:

```
[Month] [Date] [Time] [hostname] snmptrapd[xxxxx]: [IP Address]:
Enterprise Specific Trap (25) Uptime: [Time], SNMPv2-
SMI::enterprises.343.2.14.1.5 = STRING: "Time : [Day] [Month] [Date]
[Time] [Year], Location : [location] , Chassis Serial # : [xxxxxxxx],
Board : CMM[x] , Sensor : CMM[x]:Datasync Status , Event : Initial Data
Synchronization is complete. Asserted "
```

3.5.6 System Health

The “Datasync Status” sensor will not contribute to the system health. However sync failures are captured by the “File Sync Failure” sensor and it contributes to the system health

3.6 CMM Failover

Once information is synchronized between the redundant CMMs, the active CMM will constantly monitor its own health as well as the health of the standby CMM. In the event of one of the scenarios listed in the sections that follow, the active CMM will automatically failover to the standby CMM so that no management functionality is lost at any time.

3.6.1 Scenarios That Prevent Failover

The following are reasons a failover can NOT occur:

- The active CMM can NOT communicate with the standby CMM via their IPMB bus.
- Not all priority 1 data has been completely synchronized between the CMMs.

To determine the active CMM at anytime, use the CLI command:

```
cmmget -l cmm –d redundancy
```

This command will output a list stating if both CMMs are present, which one is the active CMM, and which CMM you are logged in to. CMM1 is the CMM on the left when looking from the front of the chassis, and CMM2 is on the right.

3.6.2 Scenarios That Failover to a Healthier Standby CMM

The scenarios listed below can only cause a failover if the standby CMM is in a healthier state than the active CMM. The health of the CMM is determined by computing a CMM health score, which is equal to the sum of the weights of the following active conditions. A CMM health score is determined for each CMM whenever any of these conditions occur on the active CMM. The CMM health score is composed of the sum of the weights of any of the three conditions listed below. Each condition has a default weight of 1 assigned to it, causing all conditions to have equal importance in causing failover.

To determine if a failover is necessary when one of these conditions occurs, the active CMM computes its CMM health score, and requests the health score of the standby CMM. If the score of the standby CMM is LESS than the score of the active CMM, a failover will occur. If a failover does not occur, the CMM SEL will contain an entry indicating the reason failover did not occur.

1. **SNMPTrapAddress1 ping failure**:

The active CMM will failover to the standby CMM if the active CMM cannot ping its first SNMP trap address (SNMPTrapAddress1) over any of the available Ethernet ports, but the standby CMM can. The trap address is set using the command:

```
cmmset –l cmm –d snmptrapaddress1 –v [ip address]
```

Only a ping failure of the first SNMP trap address (SNMPTrapAddress1) can cause a failover. SNMPtrapaddress2 through SNMPtrapaddress5 do not perform this ping test.

Note: The frequency of the ping to the first trap address can vary from one second to approximately 20 seconds.

2. **Critical events on the active CMM:**

   The active CMM has critical events for any of the CMM sensors (not critical chassis or blade events) and the standby CMM does not. If both CMMs have critical CMM events, then the number of major and minor CMM events is examined to decide if a failover should occur. The number of major events is compared, and if they are equal, the number of minor events is used.

3.6.3 Manual Failover

The following command can be issued to the active CMM to manually cause a failover to the standby CMM:

```
cmmset -l cmm -d failover -v [1/any]
```

Where:

1: Will failover only to a CMM with the same or newer version of firmware.

any: Will failover to any version of firmware.

A manual failover can only be initiated on the active CMM. A failover will only occur if the standby CMM is at least as healthy as the active CMM. Once the command executes, the former standby CMM immediately becomes the active CMM.

If the failover could not occur, the CLI will indicate the reason why the failover could not occur, and a SEL event will be recorded.

In addition, opening the ejector latch on the active CMM will initiate a failover, but only if the standby is at least as healthy as the active.

3.6.4 Scenarios That Force a Failover

The following scenarios cause a failover as long as the standby CMM is operational, even when it is less healthy than the active:

- The active CMM is pulled out of the chassis.
- The active CMM’s healthy signal is de-asserted.
- A “reboot” command issued to the active CMM.
- The front panel alarm quiet switch button on the active CMM is pushed for more than five seconds. If the button continues to be pressed for more than 10 seconds, the CMM does not reset.

3.7 CMM Ready Event

The CMM Ready Event is a notification mechanism that informs the user when all CMM modules are fully up and running. The CMM is ready to process any request after receiving this event.

The CMM uses the "CMM Status" sensor when generating the CMM Not Ready event. Please refer to Table 46, “CMM Status Event Strings (CMM Status)” on page 118 for CMM status event strings.

Table 3. CMM Status Event Strings (CMM Status)

Event String	Event Code	Event Severity
“CMM is not ready.”	1024	Minor
“CMM is ready.”	1025	OK
“CMM is Active”	1026	OK
“CMM is Standby”	1027	OK
“CMM ready timed out”	1028	Minor

A CMM Not Ready Assertion SEL event is generated on a CMM when it transitions from standby mode to active mode during a failover or on the active CMM on power up. The event is only generated on the newly active CMM. The “CMM is Ready” event is generated after all CMM modules (board wrapper processes) are up and running and the SNMP daemon is active.

Built-In Self Test (BIST) 4

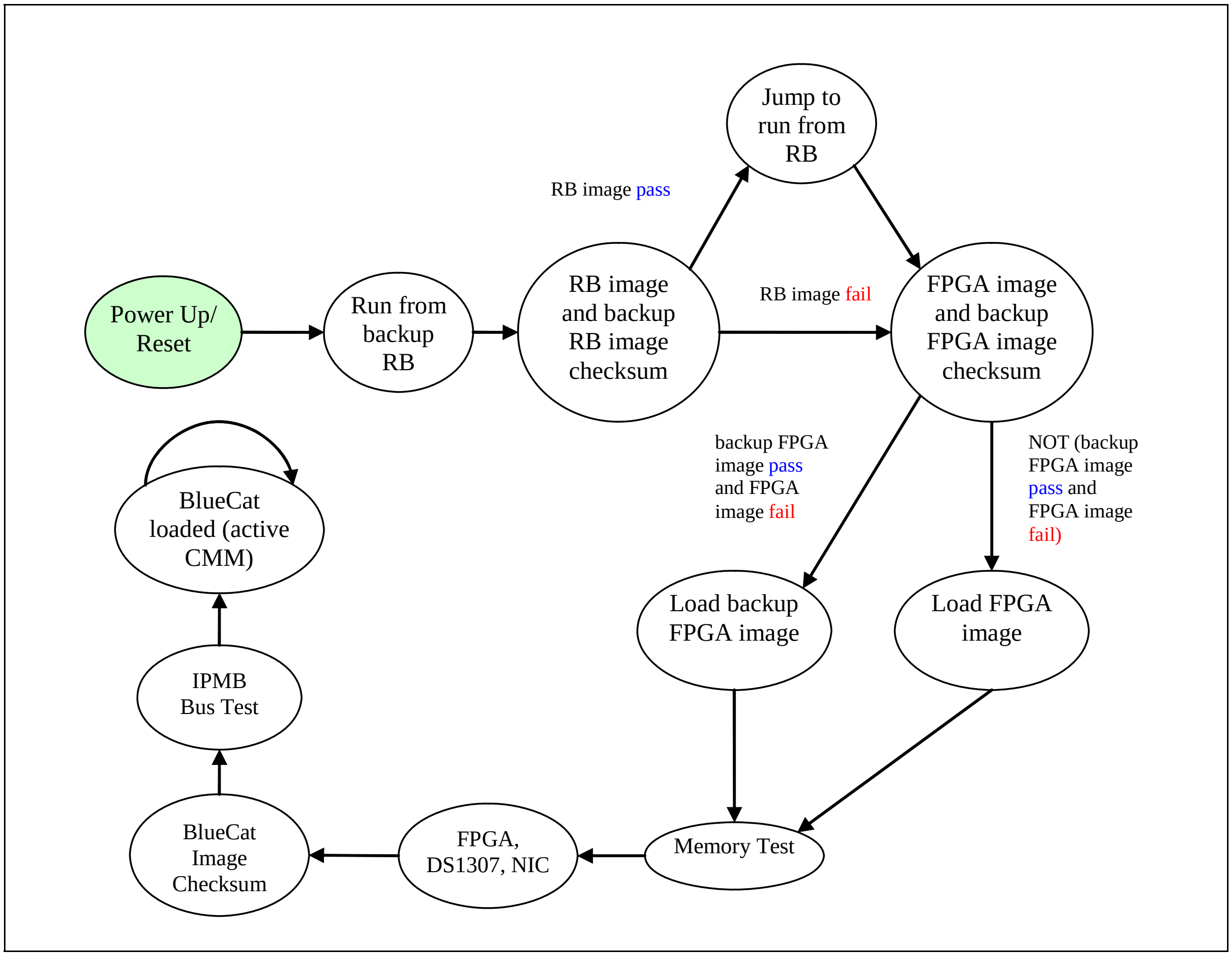
The CMM provides for a Built-In Self Test (BIST). The test is run automatically after power up. This test detects flash corruption as well as other critical hardware failures.

Results of the BIST are displayed on the console through the serial port during boot time. Results of BIST are also available through the CLI if the OS successfully boots. If the BIST detects a fatal error, the CMM is not allowed to function as an active CMM.

4.1 BIST Test Flow

The following state diagram shows the order of the tests RedBoot runs following a power-up or front-panel reset. On every state before reaching active CMM, if there is an error, RedBoot will log the error event into the EEPROM, route the error message to the serial port, and continue booting. If the execution hangs before the OS loads due to the nature of the error, the CMM hangs. If the OS successfully boots, it alerts users to any errors that occurred during boot.

Figure 1. BIST Flow Chart


Jump to run from RB
RB image pass
Power Up/ Reset
Run from backup RB
RB image and backup RB image checksum
RB image fail
FPGA image and backup FPGA image checksum
backup FPGA image pass and FPGA image fail
NOT (backup FPGA image pass and FPGA image fail)
BlueCat loaded (active CMM)
Load backup FPGA image
Load FPGA image
IPMB Bus Test
BlueCat Image Checksum
FPGA, DS1307, NIC
Memory Test

The BIST has been broken down into stages consisting of groups of tests that run at certain times throughout the boot process. The following table shows the different BIST stages and the tests associated with each stage:

Table 4. BIST Implementation

Boot-BIST	Early-BIST	Mid-BIST	Late-BIST
RedBoot image checksum	Strobe WDT to extend timeout period	Extended memory test	BlueCat image checksum
FPGA image checksum		FPGA version check	IPMB bus test
Base memory test		DS1307 RTC test	
		Local PCI bus/NIC presence test	

4.2 Boot-BIST

The codes in Boot-BIST are executed at the very early stage of the RedBoot bootstrap, which is just before the FPGA programming and memory module initialization. Boot-BIST performs checksum checking over the RedBoot image and the FPGA image. A checksum error will be detected if there is a mismatch between the calculated checksum and the stored checksum in FIS directory.

Boot-BIST also performs a Base Memory Test for the first 1 MByte of memory. Whenever there is an error, BIST will inform the user by prompting a warning message through the console terminal and log the event to event-log area.

4.3 Early-BIST

The early BIST stage extends the reset timeout period on the watchdog timer (MAX6374) by strobing GPIO7 on FPGA1. This prevents any possible hardware reset during the BIST process. The watchdog timer is enabled after the ADM1026 GPIO initialization and disabled once it reaches the RedBoot console. The OS enables the watchdog timer again and starts the strobing thread at the kernel level.

4.4 Mid-BIST

This stage of BIST performs the Extended Memory Test to scan and diagnose the possible bit errors in the memory. It starts scanning from 1 MByte to the 128 MByte. It does not test the memory below 1 MByte because a portion of RedBoot has already loaded and resided on it.

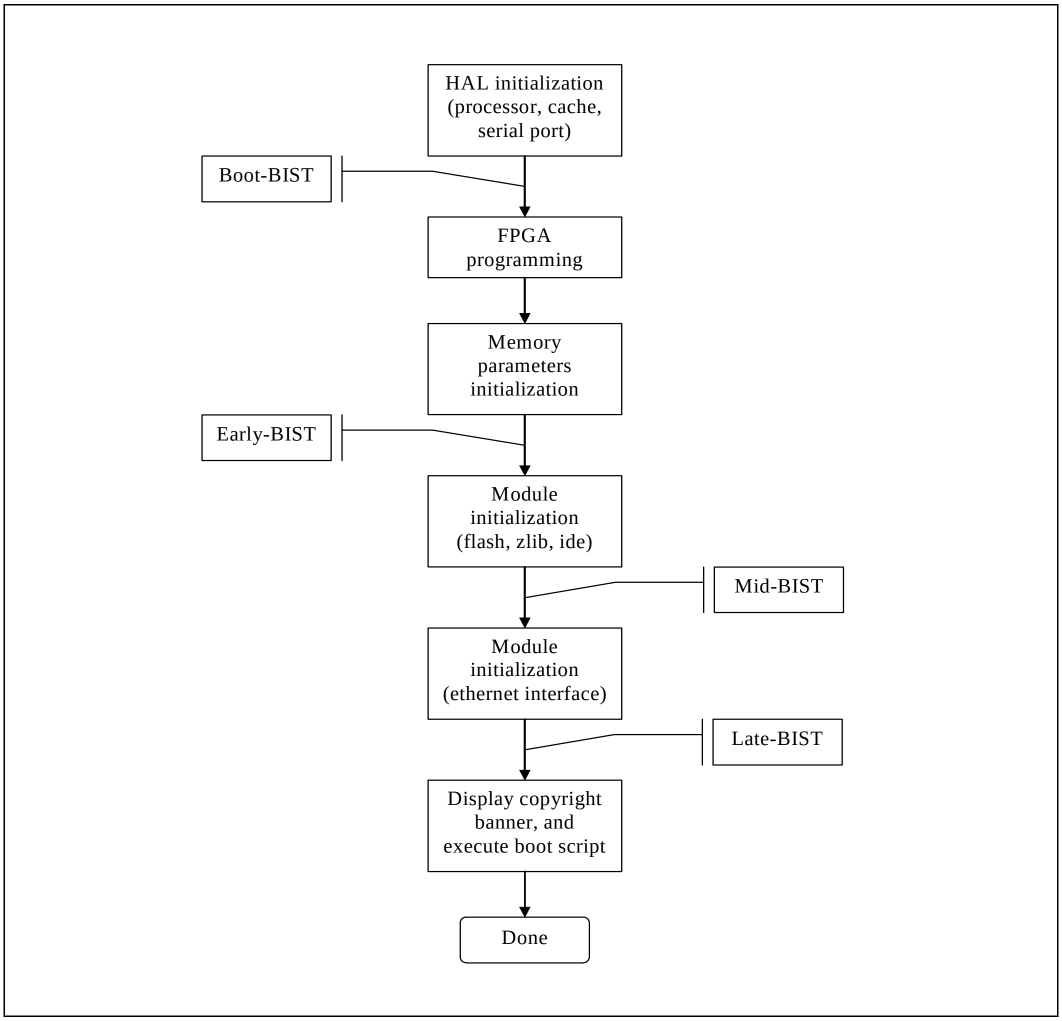
The memory test includes the walking ones test 32-bit address test, and 32-bit inverse address test. Furthermore, voltage and temperature ratings will be verified to lie within the hardware tolerable ranges. The FPGA firmware version is checked and will alert if an older version of an FPGA image has been detected. Also, system date and time is read from the real-time clock and displayed through the console terminal. NIC presence is also checked here, though the NIC self-test happens later when the driver is loaded.

4.5 Late-BIST

Late-BIST disables the watchdog timer once RedBoot is fully loaded. It then verifies the checksum of the OS image with a stored checksum at the top of flash memory, before proceeding with the boot script execution.

The following diagram shows the times during the boot cycle the when various stages of BIST are performed.

Figure 2. Timing of BIST Stages


HAL initialization (processor, cache, serial port)
Boot-BIST
FPGA programming
Memory parameters initialization
Early-BIST
Module initialization (flash, zlib, ide)
Mid-BIST
Module initialization (ethernet interface)
Late-BIST
Display copyright banner, and execute boot script
Done

4.6 QuickBoot Feature

This feature will skip all the diagnostics tests in the mid-BIST and late-BIST, once it has been enabled. However, Flash Test and Base Memory Test in the boot-BIST will still execute, even with this feature enabled. The default setting is QuickBoot enabled.

When QuickBoot feature has been disabled, user has the choice to optionally enable or disable the Extended Memory Test (in mid-BIST) and the OS Image Checksum Test (in late-BIST) individually.

4.6.1 Configuring QuickBoot

```
RedBoot> fconfig
...
Enable QuickBoot during BIST: false
```

```
Execute extended memory test: true
OS image checksum at boot: true
...
Update RedBoot non-volatile configuration - are you sure (y/n)? y
```

The default 'Enable QuickBoot during BIST' is true. When 'Enable QuickBoot during BIST' set to false, there will be two additional options displayed in the configuration menu. They are 'Execute extended memory test' and 'OS image checksum at boot' options. User can selectively enable one or both tests during the QuickBoot disabled mode. Both options will not be shown in the configuration menu if the QuickBoot is enabled. These options will go into effect during the next boot.

4.7 Event Log Area and Event Management

Errors detected by the BIST are stored in an event log. The event-log area is designed to have up to 269 entries. Each entry is 14 bytes. The event-log area is located in EEPROM on the CMM. The BIST can place entries into the event log until it becomes full. Once full, any new entries will be lost. The BIST event log is cleared by the OS once the OS logs any BIST errors into the SEL.

At OS start-up, the CMM reads the contents of BIST results in the reserved event log area and stores the errors as entries in the CMM SEL. This allows the CMM application to take the appropriate action based upon the SEL events as a result of RedBoot BIST tests. If there is not enough space to log the events in the CMM SEL, no results are logged to the CMM SEL.

The BIST event log is erased only after the event log is stored into the CMM SEL. Event strings for BIST events are listed in Section 11, “Health Events” on page 104.

4.8 OS Flash Corruption Detection and Recovery Design

The OS is responsible for the flash content integrity at runtime. Flash monitoring under the OS environment can be divided into two parts: Monitoring static images and monitoring dynamic images.

Static images refer to the RedBoot image, FPGA image and BlueCat image in flash. These images should not change throughout the lifetime of the CMM unless they are purposely updated or corrupted. The checksum for these files is written into flash when the images are uploaded.

Dynamic image refers to the OS Flash File System (JFFS2). This image dynamically changes throughout the runtime of the OS.

4.8.1 Monitoring the Static Images

A static test is run every 24 hours during CMM operation. The static test reads each static image (RedBoot, FPGA, BlueCat), calculates the image checksum, and compares with the checksum in the RedBoot configuration area (FIS). If the checksum test fails, the error is logged to the CMM SEL.

4.8.2 Monitoring the Dynamic Images

For monitoring the dynamic images, the CMM leverages the corruption detection ability from the JFFS(2) flash file system. At OS start-up, the CMM executes an initialization script to mount the JFFS(2) flash partitions (/etc and /home). If a flash corruption is detected, an event is logged to the CMM SEL.

During normal OS operation, flash corruption during file access can also be detected by the JFFS(2) and/or the flash driver. If a flash corruption is detected, an event is logged to the CMM SEL.

4.8.3 CMM Failover

If during normal OS operation a critical error occurs on the active CMM, such as a flash corruption, the standby CMM is checked to see if it is in a healthier state. If the standby CMM is in a healthier state, then a failover will occur. See Section 3, “Redundancy, Synchronization, and Failover” on page 21.

4.9 BIST Test Descriptions

4.9.1 Flash Checksum Test

This test is targeted to verify the RedBoot image and FPGA image are not corrupted. This test calculates the CRC32 checksum from the RedBoot image, then compares with the image checksum stored in the FIS directory. If one mismatches another, BIST switches to the backup image. If checksum mismatch was found from the FPGA image, BIST loads the backup image to program the FPGA device.

4.9.2 Base Memory Test

This test writes the data pattern of 55AA55AA into every 4 bytes of the memory below 1 MByte. Its objective is to verify the wire connectivity of address and data pins between the memory module and the processor. The test first writes the data pattern into the complete first 1 MByte, then verifies the written data pattern by reading them from the memory module. If the data pattern mismatches, the test logs the error event into the event-log area and routes the error message to the serial port.

4.9.3 Extended Memory Tests

Walking Ones Test

This test is targeted to verify the data bus wiring by testing the bus one bit at a time. The data bus passes the test if each data bit can be set to 0 and 1 independently of the other data bits.

32-Bit Address Test

This test is targeted to verify the address bus wiring. The smallest set of addresses that will cover all possible combinations is the set of “power-of-two” addresses. These addresses are analogous to the set of data values used in the walking ones test. The corresponding memory locations are 0001h, 0002h, 0004h, 0008h, 0010h, 0020h, and so on. In addition, address 0000h must also be tested. To confirm that no two memory locations overlap, initial data value is first written at each power-of-two offset within the device. Then a new value is written–an inverted copy of the initial value to the first test offset. It is then verified that the initial data value is still stored at every other power-of-two offset. If a location is found, other than the one just written, that contains the new data value, there is a problem with the current address bit. If no overlapping is found, the procedure is repeated for each of the remaining offsets.

32-Bit Inverse Address Test

This test behaves similarly to the memory test described above, except the addresses are tested in the inverse direction. This test helps to identify a broader scope of possible addressing errors inherent in the memory modules.

4.9.4 FPGA Version Check

This test is targeted to verify the correct FPGA image programmed into both FPGA chips. It displays the FPGA version on both FPGAs. Both versions should be the same. If the programmed version is older than expected, an event is logged to the SEL.

4.9.5 DS1307 RTC (Real-Time Clock) Test

This test is targeted to verify the functionality of DS1307 RTC chip. This test displays the date/time settings from the RTC and validates the readings. If any readings are found to be non-BCD format, an event is logged to the SEL. This test also captures current time, sleeps a while, and compares the previously captured time and new time. If they differ, it means the RTC is working. If not, an event is logged to the SEL.

4.9.6 NIC Presence/Local PCI Bus Test

This test generates the PCI bus transaction by scanning the PCI buses available on the board. This test detects the two Ethernet devices and verifies each device has the valid Vendor ID and Device ID in the PCI configuration space. NIC internal self-test is not performed here, as the self-test is executed when loading the Ethernet driver.

4.9.7 OS Image Checksum Test

This test is targeted to verify the OS image stored in the flash is not corrupted. This test calculates the CRC32 checksum from the OS image, and then compares it with the image checksum stored in the FIS directory. If one mismatches another, BIST will log an error event to the SEL.

4.9.8 CRC32 Checksum

CRC32 is the 32-bit version of Cyclic Redundant Check technique, which is designed to ensure the bits validity and integrity within the data. It first generates the diffusion table, which consists of 256 entries of double-word; each entry is known as a unique diffusion code. The checksum calculation is started by fetching the first byte in data buffer, exclusive-OR with the temporary checksum value. The resulting value is AND-ed with 0xFF to restrict an index from 0 to 255 (decimal). That index is used to fetch a new diffusion code from the table. Next, the newly fetched diffusion code is exclusive-OR with the most significant 24 bits of the temporary checksum value (effectively 8 bits left-shifting the checksum value). The resulting value is the new temporary checksum value. The calculation process is repeated until the last byte in the data buffer. The final temporary checksum value becomes the final checksum value.

4.9.9 IPMB Bus Busy/Not Ready Test

- The objective of the test is to identify any potential FPGA lockup before loading the BlueCat. When the FPGA is detected to be locked up, an event indicating which bus actually failed is logged into the Event log.

Re-enumeration 5

5.1 Overview

The Chassis Management Module has the ability to re-enumerate devices in the chassis in the event that the chassis loses and then regains CMM management. This allows the CMM to query information on all devices in the chassis on startup if there are no active CMMs in that chassis already containing that information from which it can receive via a regular synchronization. This is achieved without having to restart the individual blades already present in the chassis.

Re-enumeration provides a way to recover from situations such as double failures where both the CMMs have failed or been accidentally removed from the chassis. For the CMM to identify the contents of the chassis, it first determines if it should do this function. The Standby CMM does not re-enumerate its information and relies on the information synchronized from the Active CMM in case a failover occurs. After the startup, the Active CMM determines what Entities are present. Then for each of these Entities, the CMM queries it to get state and other information to be able to properly manage the Entity as well as the entire chassis. The CMM stays in M2 state until re-enumeration is complete.

The CMM re-enumeration process obtains the following information for each FRU in the chassis:

- Presence
- M-State
- Power Usage
- Sensor Data Records
- Health Events
- Board EKey Usage
- Bused EKey Usage

5.2 Re-enumeration on Failover

In case of forced failover, the newly Active CMM will do re-enumeration if following conditions are satisfied:

- Re-enumeration has not completed on the Active CMM.
- Active CMM has not yet synchronized the re-enumerated data over to the Standby CMM.

In case the newly Active CMM has to do re-enumeration, it will switch to M2 state before starting re-enumeration. The Blue LED uses long blinks to provide visual indication of the state of the CMM. It is recommended that the Entities in the chassis be not activated or deactivated while re-enumeration is in progress.

5.3 Re-enumeration of M5 FRU

If, during re-enumeration, the CMM discovers that a FRU is requesting for deactivation (State M5), it denies the request and informs the FRU to go back to Active (M4) state if there is no frucontrol script present (refer to Section 18.5, “FRU Control Script” on page 169). Otherwise, the CMM executes the frucontrol script and lets it handle the deactivation of the FRU.

5.4 Resolution of EKeys

During re-enumeration, the CMM determines the status of EKeys of the Boards present in the chassis. If there are interfaces which can be enabled with respect to other end-point, the CMM completes the EKeying process as per Section 14.1. If there are EKeys enabled to a slot but CMM was unable to discover a Board in that slot, it assumes that the Board in that slot is in M7 (Communication Lost) state.

5.5 Events Regeneration

The Re-enumeration agent sends out the "Set Event Receiver" command to all the Entities in the chassis. On receiving the command, the Entities re-arm event generation for all their internal sensors. This will cause them to transmit the event messages that they have based on the current event conditions. These events will be logged in the SEL.

Note: The regeneration of events may cause events to be logged into the SEL twice. This could result in configured eventaction scripts running twice.

During the process of identifying the chassis content, once the CMM determines that the Entity is a fantray, it automatically sets the fan speeds to the critical level. The speeds are not brought back to normal level until it has determined that there are no thermal events in the chassis.

Process Monitoring and Integrity 6

6.1 Overview

The Chassis Management Module monitors the general health of processes running on the CMM and can take recovery actions upon detection of failed processes. This is handled by the Process Monitoring Service (PMS).

Upon detecting unhealthy processes, the PMS will take a configurable recovery action. Examples of recovery actions include restarting the process, failing over to the standby CMM, etc.

The PMS itself is also monitored to ensure that it is operating correctly. The PMS is monitored in both a single CMM configuration and a redundant CMM configuration. When faults are detected in the PMS, corrective actions are taken.

The PMS also provides dynamic configuration and status information through the CLI, RPC, and SNMP interfaces. For example, users can administratively lock/disable monitoring of a process while the PMS is running to suit their particular needs. The PMS also provides static configuration to allow customers the ability to tune the static system parameters for the given platform. Examples of these parameters may include monitoring interval, retries, and ramp-up times.

6.1.1 Process Existence Monitoring

Process existence monitoring utilizes the operating system's process table to determine the existence of the process. When the CMM software is started, the PMS initializes and determines the set of processes to monitor for process existence. The PMS periodically queries the operating system for the existence of that set of processes. When a monitored process is found not to exist, the PMS will generate a SEL entry and take a recovery action.

Process existence monitoring can be utilized on all permanent processes (processes which exist for the life of the CMM software as a whole). It is particularly useful when monitoring processes that were not specifically developed for running on the CMM. Applications that are provided by the operating system vendor are examples of these types of processes. For the Linux* operating system, processes like syslogd and crond would be good examples.

6.1.2 Thread Watchdog Monitoring

Thread watchdog monitoring requires that the process being monitored notifies the PMS of its continued operation. Notifying the PMS will allow the PMS to monitor the process for existence and conditions where a process locks-up. Each thread requiring monitoring within a process using the thread watchdog will register with the PMS. The PMS will loop through its list of registered threads and determine if the set of registered threads are operating. When any thread is determined to be unresponsive (i.e., not notifying the PMS of its continued operation), the PMS will generate a SEL entry and take a recovery action.

Thread watchdog monitoring can be used on all processes that are instrumented with the PMS thread watchdog API. It provides more functionality then process existence monitoring and can be used in conjunction with process integrity monitoring to provide a comprehensive solution. Thread watchdog monitoring is relatively lightweight and can be done every second, although, the process being monitored may dictate a (much) lower frequency depending on how often it is capable of feeding the watchdog.

6.1.3 Process Integrity Monitoring

The Process Integrity Executable (PIE) will be responsible for determining the health of process or processes. When a PIE finds an unhealthy process, it will notify the PMS of the errant process so that the PMS can take the appropriate action. An example of a PIE would be one that monitored the Simple Network Management Protocol (SNMP) process. The PIE could utilize SNMP get operations to query the SNMP process. If the SNMP process cannot respond to the queries with the appropriate information, the process would be considered unhealthy and the PIE would notify the PMS.

Process integrity monitoring may be used in conjunction with existence monitoring to provide a comprehensive solution.

6.2 Processes Monitored

Below is a list of processes that are monitored for Process Existence on the CMM by the Process Monitoring Service.

Table 5. Processes Monitored

Process Monitored	Process Command Line / Process Name	Target Name	Monitoring Level
CMM Wrapper Process	./WrapperProcess 23	PmsProc 23	Existence and Integrity
CMM Wrapper Process	./WrapperProcess 255	PmsProc50	Existence and Integrity
SNMP Daemon	/usr/sbin/snmpd -c /etc/snmpd.conf	PmsProc51	Existence and Integrity
CLI Server	./cli_svr	PmsProc52	Existence
Cron Daemon	/bin/crond	PmsProc100	Existence
Inet Daemon	xinetd -stayalive -reuse	PmsProc101	Existence
Syslog Daemon	/sbin/syslogd	PmsProc102	Existence
CMM Command Handler	./cmd_hand	PmsProc53	Existence
CMM Blade Process Manager	./BPM	PmsProc54	Existence and Integrity
CMM Wrapper Process [0-39]	./WrapperProcess[#] (0-39)	PmsProc[#] (60-99)	Integrity
Pms Monitor	./PmsMonitor	PmsProc3	Existence and TWL
Pms Shadow	./PmsMonitor shadow	PmsProc2	Existence and TWL

6.3 Process Monitoring Targets

The following targets are provided for the Process Monitoring Service under the cmm location:

- PmsGlobal
  Target for PMS global data
- PmsProc[#]
  Target for each process monitored
- PmsPie[#]
  Target for each PMS PIE

Use the following CLI command to view the targets for the processes being monitored.

```
cmmget -l cmm -d listtargets
```

The particular processes being monitored will be listed (e.g., PmsProc23, PmsProc100). To view the name of the process being monitored use the following example command:

```
cmmget -l cmm -t PmsProc34 -d ProcessName
```

Table 5, "Processes Monitored" contains the list of processes monitored and the command lines and the target names. The ProcessName dataitem will return the Process Command Line.

6.4 Process Monitoring Dataitems

The following dataitems are used to retrieve information on and configure the Process Monitoring Service (used with PmsGlobal or PmsProc[#] targets on the cmm location).

- AdminState
- RecoveryAction
- EscalationAction
- ProcessName
- OpState

More information on the usage and descriptions of these dataitems can be found in Section 8, "The Command Line Interface (CLI)" on page 71.

6.4.1 Examples

The following example will set the global PMS AdminState to locked:

```
cmmset -l cmm -t PmsGlobal -d AdminState -v 2
```

The following example will get the recovery action assigned to a monitored process:

```
cmmget -l cmm -t PmsProc34 -d RecoveryAction
```

The following example will get the admin state to a PIE:

cmmget -l cmm -t PmsPie176 -d AdminState

6.5 SNMP MIB Commands

SNMP commands are implemented in the CMM mib for Process Monitoring. The list of new commands can be found in the CMMs MIB file or in Section 17, “SNMP” on page 140.

6.6 Process Monitoring CMM Events

The “Process Monitoring Service” sensor types are used to assert and de-assert process status information such as process presence not detected, process recovery failure, or recovery action taken. See Section 11.4, “List of Possible Health Event Strings” on page 108 for event strings, codes, and severities for Process Monitoring.

Event severities are configurable by the user and are unique to the process being monitored.

The processes that are monitored and their default severities are listed below. Severities are configured (while PMS is not running) by changing the ProcessSeverity field in the configuration file (pms.ini). Values for severity: 1 = minor, 2 = major, 3 = critical.

- **./WrapperProcess 23**
  ProcessSeverity = 2
- **./WrapperProcess 255**
  ProcessSeverity = 2
- **/usr/sbin/snmpd -c /etc/snmpd.conf**
  ProcessSeverity = 2
- **./cli_svr**
  ProcessSeverity = 2
- **/bin/crond**
  ProcessSeverity = 2
- **xinetd -stayalive -reuse**
  ProcessSeverity = 2
- **/sbin/syslogd**
  ProcessSeverity = 1
- **./PmsMonitor**
  ProcessSeverity=2
- **./PmsMonitor shadow**
  ProcessSeverity=2
- **./WrapperProcess0 through ./WrapperProcess39**
  ProcessSeverity=2
- **./cmd_hand**
  ProcessSeverity=3
- **./BPM**

```
ProcessSeverity=3
```

Note: The recovery action and escalation action should not be set to "no action" for the xinetd process. This process is involved in data synchronization between the CMMs.

Note: When a user tries to change the recovery action for cmd_hand or BPM to values other than allowed via the CLI API, the error string displayed is:

```
"Recovery action not allowed for this target."
```

6.7 Failure Scenarios and Eventing

This section describes the process fault scenarios that are detected and handled by the PMS. It also describes the eventing that is associated with the detection and recovery mechanisms. Each scenario contains a brief textual description and a table that further describes the scenario.

In the table, the Description column outlines the current action. The Event Type String defines the text for the event that is written to the SEL. The text in this field describes the portion of the event containing event-specific string (the remainder of the event text is standard for all events). However, for PMS the target name (sensor name) will be PmsProc<#> instead of the name of the sensor (where # is the unique identifier of the given process).

The UID indicates the unique identifier for the process causing the event. An ID of 1 indicates the monitoring service itself (global) and an ID of # indicates an application process.

The Assert column indicates if the event is asserted or de-asserted. For items that are just written to the SEL for informational purposes, the assertion state is not applicable. However, it is required by the interface and therefore it will be set to de-assert.

The Severity column will define the severity of the event. A severity of Configure indicates that the severity is configurable. The configurable severities are available in the Configuration Database. The remaining columns (SNMP traps, health events, LEDs, and telecommunication alarms) define what indicator will be triggered by the event.

6.7.1 No Action Recovery

In this scenario PMS detects a process fault. The PMS is configured to take no action and therefore disables monitoring of the process.

Table 6. No Action Recovery

Description	Event String	UID	Assert	Severity
PMS detects a faulty process. The mechanism (existence, thread watchdog, or integrity) used to detect the fault will determine which of the event type strings will be used.	Process existence fault; attempting recovery or Thread watchdog fault; attempting recovery or Process integrity fault; attempting recovery	#	Assert	Configure
The recovery action specified is "no action".	Take no action specified for recovery	#	N/A	Configure
No attempt will be made to recover the process. The PMS will stop monitoring the process. See Section 6.7.11, "Process Administrative Action" on page 53, for information about how to re-enable monitoring and de-assert the event.	Process existence fault; monitoring disabled or Thread watchdog fault; monitoring disabled or Process integrity fault; monitoring disabled	#	Assert	Configure

6.7.2 Successful Restart Recovery

In this scenario PMS detects a process fault. The configured recovery action is: restart the process. The PMS is able to successfully recover the process by restarting it.

Table 7. Successful Restart Recovery

Description	Event String	UID	Assert	Severity
PMS detects a faulty process. The mechanism (existence, thread watchdog, or integrity) used to detect the fault will determine which of the event type strings will be used.	Process existence fault; attempting recovery or Thread watchdog fault; attempting recovery or Process integrity fault; attempting recovery	#	Assert	Configure
The recovery action specified is "process restart".	Attempting process restart recovery action	#	N/A	Configure
PMS was successfully able to restart the process	Recovery successful	#	De-assert	OK

6.7.3 Successful Failover/Restart Recovery

In this scenario PMS detects a process fault. The configured recovery action is: failover to the standby CMM and then restart the failed process. The PMS is able to successfully recover the process by restarting it.

Table 8. Successful Failover/Restart Recovery

Description	Event String	UID	Assert	Severity
PMS detects a faulty process. The mechanism (existence, thread watchdog, or integrity) used to detect the fault will determine which of the event type strings will be used.	Process existence fault; attempting recovery or Thread watchdog fault; attempting recovery or Process integrity fault; attempting recovery	#	Assert	Configure
The recovery action specified is "failover and restart".	Attempting process failover & restart recovery action	#	N/A	Configure
PMS executes a failover. Note this step is skipped when running on the standby CMM.	The existing code generates the events for failover. They are separate from process monitoring events and are not described here.	-	N/A	N/A
PMS was successfully able to restart the process Note PMS will execute this step even if the failover is unsuccessful (standby not available, unhealthy, etc.).	Recovery successful	#	De-assert	OK

6.7.4 Successful Failover/Reboot Recovery

In this scenario, PMS detects a process fault. The configured recovery action is: failover to the standby CMM and upon successfully executing the failover, reboot the now standby CMM. The recovery actions are successful.

Table 9. Successful Failover/Reboot Recovery

Description	Event String	UID	Assert	Severity
PMS detects a faulty process. The mechanism (existence, thread watchdog, or integrity) used to detect the fault will determine which of the event type strings will be used.	Process existence fault; attempting recovery or Thread watchdog fault; attempting recovery or Process integrity fault; attempting recovery	#	Assert	Configure
The recovery action specified is "failover & reboot"	Attempting failover & reboot recovery action	#	N/A	Configure
PMS executes a failover. Note this step is skipped when running on the standby CMM.	The existing code generates the events for failover. They are separate from process monitoring events and are not described here.	-	N/A	N/A
PMS is running on the standby CMM (failover was successful or already running on the standby), PMS recovers the CMM by rebooting. Upon initialization of PMS after the reboot. The monitor will de-assert the event.	Monitoring initialized	#	De-assert	OK

6.7.5 Failed Failover/Reboot Recovery, Non-Critical

In this scenario, PMS is running on the active CMM and detects a monitored process fault. The severity of the process is configured to a value that is not critical. The configured recovery action is: failover to the standby CMM and upon successfully executing the failover, reboot the now standby CMM. The failover recovery action is unsuccessful (standby is not available, etc.). The process being monitored is not of a critical severity and therefore the reboot of the CMM will not be performed.

Table 10. Failed Failover/Reboot Recovery, Non-Critical

Description	Event String	UID	Assert	Severity
PMS detects a faulty process. The mechanism (existence, thread watchdog, or integrity) used to detect the fault will determine which of the event type strings will be used.	Process existence fault; attempting recovery or Thread watchdog fault; attempting recovery or Process integrity fault; attempting recovery	#	Assert	Configure
The recovery action specified is "failover & reboot"	Attempting failover & reboot recovery action	#	N/A	Configure
PMS executes a failover	The existing code generates the events for failover. They are separate from process monitoring events and are not described here.	-	N/A	N/A
PMS detects that it is still running on the active CMM. The process is not critical and therefore the reboot operation will not be performed.	Failover & reboot recovery failure	#	N/A	Configure
No attempt will be made to recover the process. The PMS will stop monitoring the process. See Section 6.7.11, "Process Administrative Action" on page 53, for information about how to re-enable monitoring and de-assert the event.	Process existence fault; monitoring disabled or Thread watchdog fault; monitoring disabled or Process integrity fault; monitoring disabled	#	Assert	Configure

6.7.6 Failed Failover/Reboot Recovery, Critical

In this scenario, PMS is running on the active CMM and detects a monitored process fault. The severity of the process is configured to be critical. The configured recovery action is: failover to the standby CMM and upon successfully executing the failover, reboot the now standby CMM. The failover recovery action is unsuccessful (standby is not available, etc.). The process being monitored is of a critical severity and therefore the reboot of the CMM will be performed.

Table 11. Failed Failover/Reboot Recovery, Critical

Description	Event String	UID	Assert	Severity
PMS detects a faulty process. The mechanism (existence, thread watchdog, or integrity) used to detect the fault will determine which of the event type strings will be used.	Process existence fault; attempting recovery or Thread watchdog fault; attempting recovery or Process integrity fault; attempting recovery	#	Assert	Configure
The recovery action specified is "failover & reboot"	Attempting failover & reboot recovery action	#	N/A	Configure
PMS executes a failover.	The existing code generates the events for failover. They are separate from process monitoring events and are not described here.	-	N/A	N/A
PMS detects that it is still running on the active CMM. The process is critical and therefore the reboot operation is performed. Upon initialization of PMS after the reboot. The monitor will de-assert the event.	Monitoring initialized	#	De-assert	OK

6.7.7 Excessive Restarts, Escalate No Action

In this scenario PMS detects a process fault. The configured recovery action is: restart the process. However, the PMS also detects that the process has exceeded the threshold for excessive process restarts. Therefore, the PMS will execute the escalation action. The escalation action is configured for no action.

Table 12. Existence Fault, Excessive Restarts, Escalate No Action (Sheet 1 of 2)

Description	Event String	UID	Assert	Severity
PMS detects a faulty process. The mechanism (existence, thread watchdog, or integrity) used to detect the fault will determine which of the event type strings will be used.	Process existence fault; attempting recovery or Thread watchdog fault; attempting recovery or Process integrity fault; attempting recovery	#	Assert	Configure
The recovery action specified is "process restart"	Attempting process restart recovery action	#	N/A	Configure

Table 12. Existence Fault, Excessive Restarts, Escalate No Action (Sheet 2 of 2)

Description	Event String	UID	Assert	Severity
PMS detects that the process has been restarted excessively.	Recovery failure due to excessive restarts	#	N/A	Configure
PMS attempts to execute the escalated recovery action. Since the recovery action is "no action", PMS disables monitoring of the process.	Take no action specified for escalated recovery	#	N/A	Configure
No attempt will be made to recover the process. The PMS will stop monitoring the process. See Section 6.7.11, “Process Administrative Action” on page 53, for information about how to re-enable monitoring and de-assert the event.	Process existence fault; monitoring disabled or Thread watchdog fault; monitoring disabled or Process integrity fault; monitoring disabled	#	Assert	Configure

6.7.8 Excessive Restarts, Successful Escalate Failover/Reboot

In this scenario PMS detects a process fault. The configured recovery action is: restart the process. However, the PMS also detects that the process has exceeded the threshold for excessive process restarts. Therefore, the PMS will execute the escalation action. The configured escalation recovery action is: failover to the standby CMM and upon successfully executing the failover, reboot the now standby CMM. The escalated recovery action is successful.

Table 13. Excessive Restarts, Successful Escalate Failover/Reboot

Description	Event String	UID	Assert	Severity
PMS detects a faulty process. The mechanism (existence, thread watchdog, or integrity) used to detect the fault will determine which of the event type strings will be used.	Process existence fault; attempting recovery or Thread watchdog fault; attempting recovery or Process integrity fault; attempting recovery	#	Assert	Configure
The recovery action specified is "restart process"	Attempting process restart recovery action	#	N/A	Configure
PMS detects that the process has been restarted excessively.	Recovery failure due to excessive restarts	#	N/A	Configure
The escalated recovery action specified is "failover and reboot"	Attempting failover & reboot escalated recovery action	#	N/A	Configure
PMS executes a failover. Note this step is skipped when running on the standby CMM.	The existing code generates the events for failover. They are separate from process monitoring events and are not described here.	-	N/A	N/A
PMS is running on the standby CMM (failover was successful or already running on the standby), PMS recovers the CMM by rebooting. Upon initialization of PMS after the reboot. The monitor will de-assert the event.	Monitoring initialized	#	De-assert	OK

6.7.9 Excessive Restarts, Failed Escalate Failover/Reboot, Non-Critical

In this scenario PMS detects a process fault. The severity of the process is configured to a value that is not critical. The configured recovery action is: restart the process. However, the PMS also detects that the process has exceeded the threshold for excessive process restarts. Therefore, the PMS will execute the escalation action. The configured escalation recovery action is: failover to the standby CMM and upon successfully executing the failover, reboot the now standby CMM. The failover recovery action is unsuccessful (standby is not available, etc.). The process being monitored is not of a critical severity and therefore the reboot of the CMM will not be performed.

Table 14. Excessive Restarts, Failed Escalate Failover/Reboot, Non-Critical

Description	Event String	UID	Assert	Severity
PMS detects a faulty process. The mechanism (existence, thread watchdog, or integrity) used to detect the fault will determine which of the event type strings will be used.	Process existence fault; attempting recovery or Thread watchdog fault; attempting recovery or Process integrity fault; attempting recovery	#	Assert	Configure
The recovery action specified is "restart process"	Attempting process restart recovery action	#	N/A	Configure
PMS detects that the process has been restarted excessively.	Recovery failure due to excessive restarts	#	N/A	Configure
The escalated recovery action specified is "failover and reboot"	Attempting failover & reboot escalated recovery action	#	N/A	Configure
PMS executes a failover.	The existing code generates the events for failover. They are separate from process monitoring events and are not described here.	-	N/A	N/A
PMS detects that it is still running on the active CMM. The process is not critical and therefore the reboot operation will not be performed.	Failover & reboot escalated recovery failure	#	N/A	Configure
No attempt will be made to recover the process. The PMS will stop monitoring the process. See Section 6.7.11, "Process Administrative Action" on page 53, for information about how to re-enable monitoring and de-assert the event.	Process existence fault; monitoring disabled or Thread watchdog fault; monitoring disabled or Process integrity fault; monitoring disabled	#	Assert	Configure

6.7.10 Excessive Restarts, Failed Escalate Failover/Reboot, Critical

In this scenario, PMS detects a process fault. The severity of the process is configured as critical. The configured recovery action is: restart the process. However, the PMS also detects that the process has exceeded the threshold for excessive process restarts. Therefore, the PMS will execute the escalation recovery action. The configured escalation recovery action is: failover to the standby CMM and upon successfully executing the failover, reboot the now standby CMM. The failover recovery action is unsuccessful (standby is not available, etc.). The process being monitored is of critical severity and therefore the reboot of the CMM will still be executed even though the CMM is still active.

Table 15. Excessive Restarts, Failed Escalate Failover/Reboot, Critical

Description	Event String	UID	Assert	Severity
PMS detects a faulty process. The mechanism (existence, thread watchdog, or integrity) used to detect the fault will determine which of the event type strings will be used.	Process existence fault; attempting recovery or Thread watchdog fault; attempting recovery or Process integrity fault; attempting recovery	#	Assert	Configure
The recovery action specified is "restart process"	Attempting process restart recovery action	#	N/A	Configure
PMS detects that the process has been restarted excessively.	Recovery failure due to excessive restarts	#	N/A	Configure
The escalated recovery action specified is "failover and reboot"	Attempting failover & reboot escalated recovery action	#	N/A	Configure
PMS executes a failover.	The existing code generates the events for failover. They are separate from process monitoring events and are not described here.	-	N/A	N/A
PMS detects that it is still running on the active CMM. The process is critical and therefore the reboot operation is performed. Upon initialization of PMS after the reboot. The monitor will de-assert the event.	Monitoring initialized	#	De-assert	OK

6.7.11 Process Administrative Action

In this scenario, PMS has detected a fault in a process, but has not been able to recover the process (recovery is configured for no action, etc.). This causes PMS to operationally disable monitoring of the process. To re-enable monitoring of the process, an operator must administratively lock the process, take the necessary actions to fix the process, and administratively unlock the process.

Table 16. Administrative Action

Description	Event String	UID	Assert	Severity
Operator administratively locks monitoring of the process	None	-	N/A	N/A
Operator takes actions to fix the problem	N/A	-	N/A	N/A
Operator administratively unlocks monitoring of the process causing monitoring to restart	Monitoring initialized	#	De-assert	OK

6.7.12 Excessive Failover/Reboots, Administrative Action

Prior to executing any failover/reboot the PMS will determine if the failover/reboot threshold has been exceeded. If it has, the PMS will be operationally disabled. When PMS is disabled, all process monitoring is halted. To re-enable the PMS, the operator must lock the global administrative state. The operator can then fix the problem and administratively unlock the global administrative state.

The following events are generated against the PMS Monitor (unique ID 1). The events for the process or processes that caused this condition to occur will also be present, but are not described in this table. They are defined in the scenarios provided above.

Table 17. Excessive Failover/Reboots, Administrative Action

Description	Event String	UID	Assert	Severity
PMS detects excessive failover/ reboots	Excessive reboots/failovers; all process monitoring disabled	1	Assert	Major
Operator locks the global administrative state	None	-	N/A	N/A
Operator takes actions to fix the problem	N/A	-	N/A	N/A
Operator unlocks the global administrative state causing monitoring to be resumed	Monitoring initialized	1#[a]	De-assert	OK

[a] The "Monitoring initialized" will be generated for the monitor (unique 1) as well as the individual processes that are administratively unlocked.

6.8 Process Integrity Executable (PIE)

The Process Integrity Executable (PIE) for the Chassis Management Module's (CMM) Blade Proxy Manager (BPM) and Wrapper Processes is responsible for determining the health of the Wrapper Processes. Monitoring the integrity means not only monitoring the fact that the process is running but that it is functioning properly.

The PIE will monitor the BPM, CMM Wrapper Process (Wrapper Process number 255) and Chassis Wrapper Processes (23). It will also monitor the Wrapper Processes for intelligent (have a management controller) blades, power supplies, and fans. Wrapper Processes for non-intelligent devices will not be monitored.

PIE will monitor the BPM and Wrapper Processes. The Wrapper Processes have two categories for integrity monitoring. The first category contains the static processes. Static processes are processes that are always present while the CMM software is running. The CMM (255) and chassis (23) Wrapper Processes are the static processes. The second category contains all the dynamic Wrapper Processes. Dynamic processes are ones that come and go as the configuration of the chassis changes (such as a blade insertion or removal). The fan, power supply, and blade Wrapper Processes belong to the dynamic category.

6.9 Configuring pms.ini

The pms.ini file is the Process Monitoring Service (PMS) and Process Integrity Exectuable (PIE) configuration file. It contains all of the non-volatile configuration data for the service. This file can be found in the /etc/cmm directory on the CMM. It is an ASCII based text file that can be edited with vi or any other text editor.

Note: Any changes made to the pms.ini file will be overwritten during a firmware update. Care should be made to preserve the file or any changes before a firmware update is done so that the file and changes can be restored following the update.

The dynamic data fields (except the AdminStates) in this file will be replicated to the standby CMM via the CMM Data Synchronization Service. If invalid data is provided for a particular field (i.e. out of range), the default value, if one exists, will be used.

If invalid data is provided for a particular field (i.e. out of range), the default value, if one exists, will be used. If a default value is not possible, that entire section not be used. For example, PmsProcess012 will be ignored if no value is given for its CommandLine.

Database changes are classified in two categories: dynamic and static. Dynamic changes are initiated by an interface (RPC, CLI, or SNMP). The change will take effect in the PMS and the data in this file will be updated. Dynamic changes can be made while the PMS is running.

Static changes are made directly to this file and must be done while the PmsMonitor is not running.

6.9.1 Global Data

This data applies to the PMS as a whole (not specific to a process). There must be one and only one set of this data.

6.9.1.1 PMS Administrative State

The PMS administrative state determines if monitoring of all processes will be allowed.

Values: 1 - unlocked (enabled), 2 - locked. Default: 1. (dynamic)

```
AdminState = 1
```

6.9.1.2 PMS Excessive Reboot/Failover Count

The maximum number of reboots or failover attempts allowed (over the interval specified in the field below).

Values: 2 - 255. Default: 3.

```
ExcessiveRebootOrFailoverCount = 3
```

6.9.1.3 PMS Excessive Reboot/Failover Interval

The interval, in seconds, over which the maximum number of reboots/failovers will be measured.

Values: 1 - 65535. Default: 900.

```
ExcessiveRebootOrFailoverInterval = 21600
```

6.9.2 Process Specific Data

This data applies to a specific process running on the CMM. There will be one **set** of this data for each process.

The following information describes each of the fields in the process specific section.

6.9.2.1 Process Section Name

The section name MUST follow the pattern "PmsProcessXXX" where XXX is a number from 010 to 175 inclusive. PmsProcess section names must be unique but are NOT significant in any other way. Specifically, they are NOT required to match the UniqueID field for the section.

```
[PmsProcess151]
```

6.9.2.2 Unique ID

This is a unique identifier for the program and its arguments. It is essentially the short version of the "Process Name and Arguments" field above.

Values:

0 = Reserved
1 = PMS Monitor (global)
2 = PMS Shadow
3 = PMS Monitor (for shadow monitoring)
4-9 = Reserved
10-150 = CMM processes
151-175 = User processes
176-200 = PIEs
201-255 = Reserved
Default: None.

```
UniqueID = 151
```

6.9.2.3 Chassis Applicability

This is a list of chassis types for which this particular Uid is valid. The list is comma delimited. Spaces are ignored. If this key is not present, then the Uid is valid on all chassis.

Values: MPCHC0001, ZT5085, ZT5088, ZT5089, ZT5090, ZT5091.

```
ChassisApplicability = MPCHC0001, ZT5085, ZT5088, ZT5089, ZT5090, ZT5091
```

6.9.2.4 Process Name and Arguments

This string contains the program name including its path and its associated command line arguments. This field will be used to monitor a program and therefore must be an exact match to how the program is represented in the OS. The program name and command line arguments are space separated with the program name being the first entry in the string. If an individual argument contains spaces, the argument must be encapsulated in quotation marks. The program name and arguments will uniquely identify the entry. This means if the same program is started multiple times with different arguments, each of them will require a separate entry.

Values: N/A. Default: None.

```
CommandLine = MyProcess -x -y
```

6.9.2.5 Start Program Name and Arguments

This is the program name and arguments used to start the program. This differs from the monitoring program name and arguments because some programs are started via scripts. For example many Linux system programs are started via startup scripts located in the "init.d" directory.

Values: N/A. Default: None.

```
StartCommandLine = MyProcess -x -y
```

6.9.2.6 Administrative State

The process administrative state determines if the process will be monitored.

Values: 1 - unlocked (enabled), 2 - locked. Default: 1. (dynamic)

```
AdminState = 1
```

6.9.2.7 Process Existence Interval

This is the interval in seconds in which to verify that a process exists. A value of 0 disables Existence Monitoring.

Values: 0 - 65535. Default: 2.

```
ProcessExistenceInterval = 2
```

6.9.2.8 Thread Watchdog Retries

This is the number of retries (number of thread watchdog intervals) to wait for notification from a thread. Recovery takes place on retries+1 missed thread watchdog intervals.

Values: 0-10, default: 3.

```
ThreadWatchdogRetries = 3
```

6.9.2.9 Process Ramp-up Time

The amount of time in seconds necessary for the process to initialize and be functional.

Values: 0-255. Default: 60.

```
ProcessRampUpTime = 3
```

6.9.2.10 Process Severity

An indicator for the importance of a given process. This severity will determine at what level SEL entries are generated and when reboots should occur on an active CMM.

Values: 1 = minor, 2 = major, 3 = critical. Default: 1.

```
ProcessSeverity = 1
```

6.9.2.11 Recovery Action

This is the recovery action to take upon detection of a failed process.

Values: 1 = no Action, 2 = process restart, 3 = failover and process restart, 4 = failover and reboot. Default: 1. (dynamic)

```
RecoveryAction = 1
```

6.9.2.12 Process Restart Escalation Action

This determines the action to take if the RecoveryAction includes "process restart" and it fails.

Values: 1= no action, 2 = failover and reboot. Default: 1. (dynamic)

```
ProcessRestartEscalationAction = 1
```

6.9.2.13 Process Restart Escalation Number

This is the number of process restarts that are allowed (within the interval specified below) before escalation starts.

Values: 1 - 255. Default: 5.

```
ProcessRestartEscalationNumber = 5
```

6.9.2.14 Process Restart Escalation Interval

This is the interval in seconds at which the number of restarts will be limited (see above).

Values: 1 - 65535. Default: 900.

```
ProcessRestartEscalationInterval = 900
```

6.9.3 Process Definition Section of pms.ini

The following sections describe and give examples of each of the process types that are defined in the pms.ini file.

6.9.3.1 Shadow Process

Shadow process must exist to monitor "Monitor Process". Therefore this process should never have a recovery action of "no action".

```
[PmsProcess002]

UniqueID = 2

CommandLine = ./PmsMonitor shadow

StartCommandLine = ./PmsMonitor shadow

AdminState = 1

ProcessExistenceInterval = 2

ThreadWatchdogRetries = 5

ProcessRampUpTime = 5

ProcessSeverity = 2

RecoveryAction = 2

ProcessRestartEscalationAction = 1

ProcessRestartEscalationNumber = 10

ProcessRestartEscalationInterval = 4800
```

6.9.3.2 Monitor Process

This process must exist to monitor all other processes. Therefore this process should never have a recovery action of "no action".

```
[PmsProcess003]

UniqueID = 3

CommandLine = ./PmsMonitor

StartCommandLine = ./PmsMonitor

AdminState = 1

ProcessExistenceInterval = 2

ThreadWatchdogRetries = 5

ProcessRampUpTime = 5

ProcessSeverity = 2

RecoveryAction = 2

ProcessRestartEscalationAction = 1

ProcessRestartEscalationNumber = 10

ProcessRestartEscalationInterval = 4800
```

6.9.3.3 Chassis Wrapper Process

```
[PmsProcess023]

UniqueID = 23

CommandLine = ./WrapperProcess 23

StartCommandLine = ./WrapperProcess 23

AdminState = 1

ProcessExistenceInterval = 2

ProcessRampUpTime = 10

ProcessSeverity = 2

RecoveryAction = 2

ProcessRestartEscalationAction = 2

ProcessRestartEscalationNumber = 4

ProcessRestartEscalationInterval = 5400
```

6.9.3.4 CMM Wrapper Process

This process must exist to execute interface commands (CLI, SNMP, etc.) for the CMM. Therefore this process should never have a recovery action of "no action".

```
[PmsProcess050]

UniqueID = 50

CommandLine = ./WrapperProcess 255

StartCommandLine = ./WrapperProcess 255

AdminState = 1

ProcessExistenceInterval = 2

ProcessRampUpTime = 10

ProcessSeverity = 2

RecoveryAction = 2

ProcessRestartEscalationAction = 2

ProcessRestartEscalationNumber = 15

ProcessRestartEscalationInterval = 60
```

6.9.3.5 SNMP

```
[PmsProcess051]

UniqueID = 51

CommandLine = /usr/sbin/snmpd -c /etc/snmpd.conf

StartCommandLine = /usr/sbin/snmpd -c /etc/snmpd.conf

AdminState = 1

ProcessExistenceInterval = 2

ProcessRampUpTime = 30

ProcessSeverity = 2

RecoveryAction = 2

ProcessRestartEscalationAction = 2

ProcessRestartEscalationNumber = 4

ProcessRestartEscalationInterval = 5400
```

6.9.3.6 CLI Server

```
[PmsProcess052]

UniqueID = 52

CommandLine = ./cli_svr

StartCommandLine = ./cli_svr

AdminState = 1

ProcessExistenceInterval = 2

ProcessRampUpTime = 10

ProcessSeverity = 2

RecoveryAction = 2

ProcessRestartEscalationAction = 2

ProcessRestartEscalationNumber = 5

ProcessRestartEscalationInterval = 300
```

6.9.3.7 Command Handler

Note: PmsProc053 represents a crucial process cmd_hand (command handler) of the CMM software stack. This process cannot be restarted properly if it terminates unexpectedly. Hence, none of the recovery actions that attempt to restart a process i.e., 2 (Restart), 3 (Failover & Restart) are allowed as valid recovery actions for cmd_hand. The default recovery action for cmd_hand process is 4 (failover and reboot) and that cannot be changed to anything else. A recovery action of 1 (No Action) is also not allowed because of the severity of the process.

In the event that cmd_hand process terminates unexpectedly, and the default recovery action kicks in, there is 2-3 minute delay before the CMM actually reboots. This is normal and expected because PMS makes multiple tries to failover, and times out because cmd_hand does not respond.

```
[PmsProcess053]

UniqueID = 53

CommandLine = ./cmd_hand

StartCommandLine = ./cmd_hand

AdminState = 1

ProcessExistenceInterval = 2

ProcessRampUpTime = 10

ProcessSeverity = 3

RecoveryAction = 4

ProcessRestartEscalationAction = 2

ProcessRestartEscalationNumber = 5

ProcessRestartEscalationInterval = 300
```

6.9.3.8 BPM

Note: PmsProc054 represents a crucial process of the CMM software stack. This process cannot be restarted properly if it terminates unexpectedly. Hence, none of the recovery actions that attempt to restart a process i.e., 2 (Restart) or 3 (Failover & Restart) are allowed as valid recovery actions for BPM. The default recovery action for BPM process is 4 (failover and reboot) which can only be changed to 1 (No Action).

```
[PmsProcess054]

UniqueID = 54

CommandLine = ./BPM

StartCommandLine = ./BPM

AdminState = 1

ProcessExistenceInterval = 2

ProcessRampUpTime = 10

ProcessSeverity = 3

RecoveryAction = 4
```

```
ProcessRestartEscalationAction = 2
ProcessRestartEscalationNumber = 5
ProcessRestartEscalationInterval = 300
```

6.9.3.9 Dynamic Wrapper Process 0-39

```
[PmsProcessXYZ]
UniqueID = XYZ
ChassisApplicability = MPCHC0001, ZT5091, MPCHC5091
CommandLine = /usr/local/cmm/bin/WrapperProcess N
StartCommandLine = /usr/local/cmm/bin/WrapperProcess N
AdminState = 1
ProcessExistenceInterval = 0
ProcessRampUpTime = 10
ProcessSeverity = 2
RecoveryAction = 2
ProcessRestartEscalationAction = 2
ProcessRestartEscalationNumber = 4
ProcessRestartEscalationInterval = 5400
```

Where:

XYZ: A unique number given to the process in the range defined in Section 6.9.2.2, “Unique ID” on page 56 above (10-150).

N: The number of the wrapper process you are defining. For example, if defining wrapper process 21, then N would be 21.

6.9.3.10 Inet Daemon

```
[PmsProcess101]
UniqueID = 101
CommandLine = xinetd -stayalive -reuse
StartCommandLine = xinetd -stayalive -reuse
AdminState = 1
ProcessExistenceInterval = 2
ProcessRampUpTime = 5
```

```
ProcessSeverity = 2
RecoveryAction = 2
ProcessRestartEscalationAction = 2
ProcessRestartEscalationNumber = 5
ProcessRestartEscalationInterval = 300
```

6.9.3.11 Syslog Daemon

```
[PmsProcess102]
UniqueID = 102
CommandLine = /sbin/syslogd
StartCommandLine = /sbin/syslogd
AdminState = 1
ProcessExistenceInterval = 2
ProcessRampUpTime = 5
ProcessSeverity = 1
RecoveryAction = 2
ProcessRestartEscalationAction = 2
ProcessRestartEscalationNumber = 5
ProcessRestartEscalationInterval = 300
```

6.10 Process Integrity Executable (PIE) Specific Data Config

This data applies to each Process Integrity Executable (PIE). One PIE may monitor multiple CMM processes or only one CMM process. There will be one set of this data for each PIE.

The following information describes each of the fields in the PIE specific section. Lines with a '*' prefix, indicate the actual fields (the prefix is not part of the field name).

6.10.1 PIE Section Name

The section name MUST follow the pattern "PmsPieXXX" where XXX is a number from 176 to 200 inclusive. PmsPie section names must be unique but are NOT significant in any other way. Specifically, they are NOT required to match the UniqueID field for the section.

```
[PmsPie176]
```

6.10.2 Process Integrity Executable

The name, including its path and command line arguments, of the PIE to be executed periodically. This is used to start the program and may, in the future, be used to monitor the program and therefore must be an exact match to how the program is represented in the OS. The program name and command line arguments will all be space separated with the program name being the first entry in the string. If an individual argument contains spaces, the argument must be encapsulated in quotation marks. The program name and arguments will uniquely identify the entry. This means if the same program is started multiple times with different arguments, each of them will require a separate entry. Each PIE will likely have PIE specific options that can be specified through the command line. This options must be included in the arguments to the "ProcessIntegrityExecutable" command.

```
ProcessIntegrityExecutable = ./PmsPieSnmp
```

6.10.3 Unique ID

This is a unique identifier for the executable and its arguments. It is essentially the short version of the "Process Integrity Executable" field above. It is used for logging and CSL access.

Values:

0 = Reserved

1 = PMS Monitor (global)

2 = PMS Shadow

3 = PMS Monitor (for shadow monitoring)

4-9 = Reserved

10-150 = CMM processes

151-175 = User processes

176-200 = PIEs

201-255 = Reserved

Default: None.

```
UniqueID = 176
```

6.10.4 Administrative State

The PIE administrative state determines if the PIE will be restarted at the next interval.

Values: 1 - unlocked (enabled), 2 - locked. Default: 1. (dynamic)

```
AdminState = 1
```

6.10.5 Process Integrity Interval

This is the interval in seconds between executions of the PIE.

Values: 0 - 65535, where 0 indicates that the PIE only gets executed once.

Default: 3600.

```
ProcessIntegrityInterval = 3600
```

6.10.6 Chassis Applicability

This is a list of chassis types on which this particular Pie should be run. The list is comma delimited. Spaces are ignored. If this key is not present, then the Pie will run on all chassis.

Values: MPCHC0001, ZT5085, MPCHC5085, ZT5088, MPCHC5088, ZT5089, MPCHC5089, ZT5090, MPCHC5090, ZT5091, MPCHC5091.

```
ChassisApplicability = MPCHC0001, ZT5085, MPCHC5085, ZT5088, MPCHC5088,
ZT5089, MPCHC5089, ZT5090, MPCHC5090, ZT5091, MPCHC5091
```

6.10.7 PmsPieSnmp Command Line

The command line usage of PmsPieSnmp is:

PmsPieSnmp [-f SuccessiveFailureNumber]

where:

-f : This is the number of allowed successive integrity failures before the PMS performs recovery on the faulting process. PMS performs recovery on "this number + 1".

Values: 1 - 100. Default = 3

For Example

```
PmsPieSnmp

PmsPieSnmp -f2

PmsPieSnmp -f 2
```

6.10.8 SNMP PIE Section of pms.ini

```
[PmsPie176]

ProcessIntegrityExecutable = ./PmsPieSnmp -f2

UniqueID = 176

AdminState = 1

ProcessIntegrityInterval = 300
```

6.11 WP/BPM PIE

The command line usage of PmsPieWp is:

PmsPieWp [-s] [-d[NumberOfDynamicWrappersPerRun]] [-f SuccessiveFailureNumber]

where:

-s: check static wrappers (optional)

-d: check dynamic wrappers and bpm threads (optional)

Number of dynamic wrappers and bpm threads to check on each run (optional)

Values: 0 - 100. Default : 0 = process all dynamic wrappers and bpm threads on each execution.

-f: Successive Failure Number - This is the number of allowed successive integrity failures before the PMS performs recovery on the faulting process. PMS performs recovery on "this number + 1".

Values: 1 - 100. Default = 3

Example:

```
PmsPieWp -s -f2 - check static wrappers

PmsPieWp -d0 -f2 - check all dynamic wrappers and all BPM threads

PmsPieWp -s -d0 -f2 - Check static and all dynamic wrappers all BPM
threads

PmsPieWp -s -d10 -f2 - Check static and 10 dynamic wrappers and BPM
threads

PmsPieWp -s -d10 -f 2 - Check static and 10 dynamic wrappers and BPM
threads
```

6.11.1 WP/BPM Section of pms.ini

```
[PmsPie180]

ProcessIntegrityExecutable = ./PmsPieWp -s -d0 -f2

UniqueID = 180

AdminState = 1

ProcessIntegrityInterval = 300
```

Power and Hot Swap Management 7

The CMM is responsible for the management of FRU hot-swap activities. The CMM listens to FRU hot-swap SEL messages from IPMI devices and distributes power to each FRU after negotiating with the respective IPMI device fronting the FRU. The CMM also manages the shelf-wide power budget. The CMM also polls IPMI devices to get the status of each FRU fronted by the IPMI device. The CMM uses shelf FRU information to guarantee power-up sequence delays between boards.

Once the CMM receives the shelf FRU information on power budget and power sequence delays, it is ready to service FRU hot-swap requests from respective IPMI devices.

7.1 Hot Swap States

The CMM defines the hot swap status of a FRU as being in one of eight states. CMM documentation often refers to only the letter/number designation of that state (M0 - M7). Here is a list of what each of those states means:

- "State M0 - Not Installed"
- "State M1 - Inactive"
- "State M2 - Activation Request"
- "State M3 - Activation In Progress"
- "State M4 - Active"
- "State M5 - Deactivation Request"
- "State M6 - Deactivation In Progress"
- "State M7 - Communication Lost"

7.2 FRU Insertion

When the CMM receives a request that a FRU is ready to activate, it will compute the FRU’s power, get the power levels, and check the available power budget.

The Set_Power_Level command will be sent only when the necessary power budget, from each of the redundant power feeds, is available to satisfy FRU's desired power level. If a FRU can't be activated at the time of the request, it should remain in the M3 state and shall be powered up when the necessary power budget becomes available. If the FRU decides to operate at a lower power level and notifies the Shelf Manager and the new power level is within the current Shelf Power envelope, the CMM shall send the Set_Power_Level (new desired level) command to the FRU.

7.3 Graceful FRU Extraction

When the CMM receives a FRU Hot swap request for extraction, the CMM will send the deactivate state command, and the FRU will transition to M6 state and begin its shut-down procedures. Once the FRU has shut down, it transitions to M1 state, and the CMM then reclaims the FRU’s power and adjusts the power budget for the newly available power.

7.4 Surprise FRU Extraction/IPMI Failure

The CMM detects a surprise FRU extraction or a failure of the IPMI device fronting the FRU if a device previously in one of the M2-7 states reports a transition to the M2 state. If this scenario is detected, the CMM assumes one of three things has happened:

- Surprise extraction and reinsertion of the same (or another) FRU.
- IPMI Device fronting the FRU failed, FRU was extracted, then the same (or another) FRU is reinserted.
- Watchdog Timer (WDT) on the IPMI device restarted the IPMI Device firmware.

Once this occurs, the CMM shall reclaim all the resources allocated to that FRU. The CMM will log a SEL message describing the situation, i.e. IPMI device failure or surprise extraction. From this point the CMM shall follow the sequence of actions described in Section 7.2, “FRU Insertion”.

7.5 Forced Power State Changes

An external authorized entity (e.g., a management interface like RMCP) can request FRU power state changes like Power OFF, RESET etc. The CMM is responsible for handling these requests.

7.6 Power Management on the Standby CMM

The standby CMM does not participate in any power management activities in the standby mode. The CMM is in a hot standby state on a standby CMM. The standby CMM starts performing power management activities as soon as it becomes the active CMM.

7.7 Power Feed Targets

The CLI allows certain get and set actions to be taken on power feeds for a location. They include the following dataitems; maxexternalavailablecurrent, maxinternalcurrent, and minexpectedoperatingvoltage. These dataitems are described in Section 8, “The Command Line Interface (CLI)” on page 71.

To find the number of feed targets, use the command:

```
cmmget -l cmm -d feedcount
```

This returns an integer, indicating the number of power feeds.

As an example, the MPCHC0001 chassis with four power feeds coming from the PEMs will return the number 4, meaning there are four feed targets (feed1, feed2, feed3, and feed4). They correlate to the physical feeds on the MPCHC0001as follows:

feed1 = FeedA1
feed2 = FeedB2
feed3 = FeedA2
feed4 = FeedB1

Refer to the chassis documentation for more information on power feeds.

7.8 Pinging IPMI Controllers

The following lists the values of time to delay and number of pings that the CMM uses to determine the state of a FRU.

Table 18. Time to Delay and Number of Attempts

Variable	Description	Value
DelayBetweenPingLoops	The number of microseconds to delay between each ping loop. This is essentially the amount of time from the ping of the last IPMI Controller in the list to the ping of the first controller in the list.	10000000 (10 seconds)
DelayBetweenIPMControllerPings	The number of microseconds of delay between the ping on one controller that is in the list and the ping of the next one on the list. This delay does not apply after the last controller in the list.	0
NumberFailedAttemptsBeforeAlert	How many failed attempts to contact the IPMI Controller must occur prior to raising an event that communication has been lost.	3

The Command Line Interface (CLI) 8

8.1 CLI Overview

The Command Line Interface (CLI) connects to and communicates with the intelligent management devices of the chassis, boards, and the CMM itself. The CLI is an IPMI-based library of commands that can be accessed directly or through a higher-level management application. Administrators can access the CLI through a Telnet session, SSH, or through the CMM's front panel serial port. The CLI functions are also available through SNMP get/set commands and an RPC interface. Using the CLI, users can access information about the current state of the system including current sensor values, threshold settings, recent events, and overall chassis health.

Note: The CLI uses the term “blade” when referring to boards.

8.2 Connecting to the CLI

The CMM provides three connections on its front panel.

- Two Ethernet connections via an RJ-45 connector
- An RS-232 serial port interface also via an RJ-45 connector

These same ports are also available on the rear transition module.

Any of these interfaces can be used to log into the CMM as well as the Ethernet interface provided through the backplane of a chassis. Use Telnet to log into the CMM over an Ethernet connection, or use a terminal application or serial console over the RS-232 interface. See the *Intel® NetStructure™ MPCMM0001 Hardware Technical Product Specification* for electrical pinouts of the above interfaces.

If logging in for the first time to set up or obtain the CMMs IP addresses, use the serial port console interface to perform configuration.

8.2.1 Connecting through a Serial Port Console

Connect an RS-232 serial cable with an RJ-45 connector to the serial console port on the front of the CMM. Set your terminal application settings as follows:

- Baud – 115200
- Data Bits – 8
- Parity – None
- Stop Bits – 1
- Flow Control – Xon/Xoff or none

Connect using your terminal application.

8.3 Initial Setup— Logging in for the First Time

Logging in for the first time must be done through the serial port console to properly configure the Ethernet settings and IP addresses for the network.

The username for the CMM is ***root***. The default password is ***cmmrootpass***.

At the login prompt, enter the username: ***root***

When prompted for the password, enter: ***cmmrootpass***

The root password can be changed using the ***passwd*** command. For information on resetting the CMM password back to default, refer to Section 9, “Resetting the Password” on page 99.

8.3.1 Setting IP Address Properties

Note: Changing any of the IP address settings and restarting the network could result in a failover occurring based on the rules governing redundancy specified in Section 3, “Redundancy, Synchronization, and Failover” on page 21.

By default, the CMM assigns IP addresses statically:

- eth0, labeled “Ethernet A” on the front panel, is configured with the static IP address 10.90.90.91
- eth1, labeled “Ethernet B” on the front panel, is configured with a static IP address of 192.168.100.92
- eth1:1, an alias of eth1 is used to always point to and be active on the active CMM, is configured with a static IP address of 192.168.100.93

On initial power-up of a chassis with two CMMs, both CMMs will have the same IP addresses assigned by default. When the chassis is powered up, the standby CMM automatically decrements its IP address by one less than the active CMM if it detects a conflict.

Example:

1. A dual CMM Chassis is powered up.
2. Active CMM assigns IP address of 192.168.100.92 to eth1 on the active CMM.
3. Standby CMM assigns IP address of 192.168.100.91 to eth1 on the standby CMM.

At this point the static IP addresses must be changed to appropriate values for their network configuration, and ensure that the two CMMs do not contain duplicate IP addresses on eth0 and eth1 to avoid address conflicts on the network.

eth0 and eth1 can also be set using DHCP. eth1:1 will always remain static.When setting both eth0 and eth1 to DHCP, use the /etc/pump.conf to determine which interface should own the default gateway. The default is for eth0 to own the default gateway. To configure eth1 to own the default gateway, and thereby eth1:1, uncomment the two lines under the eth0 section of /etc/pump.conf and comment the two lines under the eth1 section of that file. Save the file and run the /etc/rc.d/ network reload script.

Note: It is recommended that both CMMs use static IP addresses for all interfaces. DHCP addresses may be unexpectedly lost or changed in some network configurations.

Note: eth0 should always be set to a different subnet than eth1/eth1:1. Failure to set eth0 to a different subnet than eth1 will cause network errors on the CMM and redundancy will be lost.

8.3.1.1 Setting Static IP Information for eth0

1. Open the /etc/ifcfg-eth0 file using the vi editor. By default, the file contains three variables.

   Example:

   — BOOTPROTO=“static”

   — DEVICE=“eth0”

   — STATICIP=“10.90.90.91”

2. Ensure the BOOTPROTO value is set to static.

Note: Linux is case sensitive, so ensure that the BOOTPROTO variable is entered in lower case letters in the step above.

3. Set the STATICIP variable to the IP address you want to assign to that interface.
4. To set the netmask for eth0, add the NETMASK0 variable and set it to the appropriate netmask for your network.
5. To set the gateway for eth0, add the GATEWAY0 variable and set it to the appropriate value for the gateway on your network.
6. To activate the changes, at the user prompt (from the root “/” directory), type:

```
/etc/rc.d/network reload
```

8.3.1.2 Setting Static IP Information for eth1 and eth1:1

Note: eth1:1 is static IP address only. It does not support DHCP.

1. Open the /etc/ifcfg-eth1 file using the vi editor. By default the file contains five variables.

   Example:

   — BOOTPROTO=”static”

   — DEVICE=”eth1”

   — SETIP=”both”

   — STATICIP1=”192.168.100.91”

   — STATICIP2=”192.168.100.93”

2. Set the STATICIP1 variable to the IP address you want to assign to eth1.
3. Set the STATICIP2 variable to the IP address you want to assign to the active CMM on the network. This value should ONLY be set on the active CMM, as it will be synchronized to and overwritten on the standby CMM.
4. Set the SETIP variable to assign IP addresses eth1 and eth1:1 based on the following table:

Table 19. SETIP Interface Assignments when BOOTPROTO=”static”

Interface	SETIP=1	SETIP=2	SETIP=Both	Other
eth1	STATICIP1	STATICIP2	STATICIP1	Previous Value
eth1:1	disabled	disabled	STATICIP2	disabled

5. Add the NETMASK1 variable and set it to the appropriate netmask for STATICIP1 for your network.
6. Add the NETMASK2 variable and set it to the appropriate netmask for STATICIP2 for your network. The NETMASK2 variable needs to be correct to allow for true redundant operation.
7. Add the GATEWAY1 variable and set it to the appropriate value for the gateway for STATICIP1.
8. Add the GATEWAY2 variable and set it to the appropriate value for the gateway for STATICIP2
9. To activate the changes, at the user prompt (from the root “/” directory), type:

```
/etc/rc.d/network reload
```

Note: The eth1:1 address should only be changed on the active CMM. The new address will be synchronized to the standby CMM automatically when the /etc/rc.d/network reload command is executed. Also, the eth1:1 should be changed with the procedure above and NOT by using the ifconfig command manually. This method will cause the eth1:1 information to not be synchronized to the standby.

8.3.1.3 Setting eth0 to DHCP

1. Using the vi editor, change the BOOTPROTO variable in the /etc/ifcfg-eth0 file to **dhcp**.

Note: Linux is case sensitive, so ensure that the BOOTPROTO value is entered in lower case letters in the step above.

2. To activate the changes, the user can reboot the CMM, or at the user prompt (from the root “/” directory) on the active CMM, type:

```
/etc/rc.d/network reload
```

Note: A DHCP server must be present on the network for the CMM to get a valid IP address. The network reload command will refresh the IP addresses on both network interfaces.

8.3.1.4 Setting eth1 to DHCP

1. Using the vi editor, change the BOOTPROTO variable in the /etc/ifcfg-eth1 file to **dhcp**.
2. eth1:1 will still use a static IP address in this configuration. Set the STATICIP2 variable to the IP address you want to assign to the active CMM on the network. This value should ONLY be set on the active CMM, as it will be synchronized to and overwritten on the standby CMM.
3. Add the NETMASK1 variable and set it to the appropriate netmask for STATICIP1 for your network.

4. Add the NETMASK2 variable and set it to the appropriate netmask for STATICIP2 for your network. The NETMASK2 variable needs to be correct to allow for true redundant operation.
5. Add the GATEWAY1 variable and set it to the appropriate value for the gateway for STATICIP1.
6. Add the GATEWAY2 variable and set it to the appropriate value for the gateway for STATICIP2
7. Set the SETIP variable to assign IP addresses eth1 and eth1:1 based on the following table:

Table 20. SETIP Interface Assignments when BOOTPROTO=”dhcp”

Interface	SETIP=1	SETIP=2	SETIP=Both	Other
eth1	dynamic	STATICIP2	dynamic	dynamic
eth1:1	disabled	disabled	STATICIP2	disabled

8. To activate the changes, at the user prompt (from the root “/” directory), type:

```
/etc/rc.d/network reload
```

8.3.2 Setting a Hostname

The hostname of the CMM is a logical name that is used to identify a particular CMM. This name is shown at login time just to the left of the login prompt on the serial port interface when configured (i.e., “MYHOST login:”) and advertised to any DNS servers on a network. If there is no entry in /etc/HOSTNAME, the login prompt will not have anything next to it. By default, the hostname is set to the product name (i.e. MPCMM0001).

The hostname should be configured on the each CMM. To change the hostname:

1. Using the vi editor, change the HOSTNAME variable in /etc/HOSTNAME to the desired name.
2. To activate the changes, at the user prompt (from the root “/” directory), type:

```
etc/rc.d/network reload
```

Note: Executing **network reload** also causes the network interfaces to reload their IP addresses. If DHCP is being used on a network interface, then it is possible that the IP address on that interface will change.

8.3.3 Setting the Amount of Time for Auto-Logout

For security purposes, the CMM automatically logs the user out of the current console session after 15 minutes (900 seconds). This auto-logout time can be changed by editing /etc/profile and changing the TMOUT value to the desired setting. The time-out (TMOUT) value is set in seconds (900 seconds is the default). A setting of TMOUT=0 will disable the automatic logout. This can also be set at the command line.

8.3.4 Setting the Date and Time

On the active CMM, use the **date** command in the CLI to view the current date and time for the CMM. To set the date and time on the CMM use the **setdate** command. The **setdate** command should use the following syntax:

```
setdate “mm/dd/yyyy [timezone] hh:mm:ss”
```

The date is stored on the CMM in Coordinated Universal Time (UTC). The local timezone can be included in the setdate string, and the CMM will determine the offset and automatically change the date to UTC. An example that will set the date and time to “Thu Mar 11 20:12:00 UTC 2004” is:

```
setdate “3/11/2004 PST 12:12:00”
```

The date and time are synchronized to the standby CMM when changed and then every hour.

8.3.5 Telnet into the CMM

To telnet into the CMM, point your console or telnet application to the IP address of the eth0, eth1, or eth1:1 interface on the CMM you wish to telnet to. If you wish to telnet to the active CMM, you can point the telnet application to the eth1:1 IP address. The “pointing” is accomplished using the Telnet ***open*** command. To get the IP address see Section 8.3.1, “Setting IP Address Properties” on page 72.

8.3.6 Connect Through SSH (Secure Shell)

For a more secure connection, users can connect to the CMM using SSH, or Secure Shell. SSH is a secure interface and protocol used for encrypted connections between a client and a server. Using an SSH client, open the IP address of the eth0, eth1, or eth1:1 interface on the CMM you wish to establish an SSH session with. SSH clients can be found freely available on the Internet.

8.3.7 FTP into the CMM

For security purposes, the CMM will prevent users from accessing the CMM through FTP. So before FTP’ing into the CMM, ensure the “root” entry is removed from the /etc/ftpusers file using a text editor like vi. If this entry is not removed, you will be unable to login via FTP.

Using an FTP client, FTP to the IP address of the CMM you wish to transfer files to or from and use the CLI login and password.

8.3.8 Rebooting the CMM

To reboot the CMM, type the ***reboot*** command in the CLI on the CMM that is to be rebooted. If the reboot command is issued on the active CMM in a redundant configuration, a failover to the standby CMM will occur. If the reboot command is issued on a CMM in a single CMM configuration, chassis management will be lost during the reboot process. Telnet and SSH sessions will have to be reestablished with the CMM after it is rebooted.

Note: Do not use the “init 0” or “init 6” command to reboot the CMM as problems may result.

8.4 CLI Command Line Syntax and Arguments

The command line interface on the CMM supports two types of commands: cmmget and cmmset. cmmget is used to query for information, whereas cmmset is used to write information.

There are man pages available on the CMM for these two commands. To access the man page for cmmget use the command ***man cmmget***. To access the man page for cmmset, use the command ***man cmmset***.

8.4.1 Cmmget and Cmmset Syntax

The syntax for calling the CLI from the command line is as follows:

```
cmmget [-h] [-l location] [-t target] -d dataitem
```

```
cmmset [-h] [-l location] [-t target] -d dataitem -v value
```

Where **cmmget** and **cmmset** are the CLI executables. The parameters can be in any order. The CLI is case insensitive, except for the executable name. Parameters shown in brackets are optional.

Any attribute value that contains a space must be enclosed in quotes. This happens often when specifying targets. For example, to get the current value of a sensor called *Brd Temp* on the CMM, the command would be:

```
cmmget –l cmm –t “Brd Temp” –d current
```

8.4.2 Help Parameter: -h

If the Help parameter is given, the rest of the parameters are ignored, and the help text is output to the user.

8.4.3 Location Parameter: -l

The Location parameter is the location in the system on which the user is executing the **cmmget** or **cmmset** on. If no location is given then the default location is the CMM.

Use the following cmmget command to list all valid locations in the chassis:

```
cmmget -d listlocations
```

The Location keywords are shown in the following table.

Table 21. Location (-l) Keywords

Keyword	Function
cmm	The Chassis Management Module.
bladeN	One of the CPU boards in the chassis. N refers to the chassis slot number into which the CPU board is inserted. Please refer to the Chassis documentation for slot information.
system	The entire platform.

Table 21. Location (-l) Keywords

Keyword	Function
chassis	Chassis specific information.
fantrayN	The system fantray where N is the number of the fantray. For example, fantray1 refers to the single fantray in the MPCHC0001 shelf. **NOTE:** fantray1 may also be referred to as blade15 in a 14 slot chassis or blade17 in a 16 slot chassis.
PEM1 PEM2	The system Power Entry Modules. PEM1 is in the left slot when looking from the front of chassis and PEM2 is in the right slot. **NOTE:** PEM1 may also be referred to as blade16 and PEM2 as blade17 in a 14-slot chassis; correspondingly they can be referred to as blade18 and blade19 in a 16-slot chassis.

8.4.4 Target Parameter: -t

The Target parameter is the sensor or variable that the **cmmget** or **cmmset** acts on. If `target` is not given then it is assumed that `dataitem` is an attribute of `location`. An example of this is `presence`. To obtain a list of valid targets for a device, issue the following command:

```
cmmget [-l location] -d listtargets
```

Where *location* is the device for which you want to obtain a list of targets.

The target parameter for plug-in boards and different chassis components is defined by the sensor name in the Sensor Data Record (SDR) for that device. The various boards, fantrays, and PEMs provide their own SDRs automatically.

The following table shows the values *target* can be for the CMM location.

Table 22. CMM Targets

Keyword	Description
Brd Temp	Board Temperature
CPU Temp	CPU Temperature
FilterTrayTemp[1,2]	Filter Tray Temperature Sensors
CPU Core V	CPU Core Voltage
VBAT	Battery Voltage
VTT DDR	CMM Memory voltage
+2.5V	+2.5V voltage sensor
+3.3V	+3.3V voltage sensor
+5V	+5V voltage sensor
+12V	+12V voltage sensor
CDM [1,2]	Chassis Data Modules 1 and 2
Air Filter	Air Filter
Filter Tray	Filter Tray FRU
Filter Run Time	Filter Run Time
BIST	Built-In Self Test Sensor
FRU Hot Swap	FRU Hot Swap sensor
Filter Tray HS	Filter Tray Hot Swap sensor
IPMB-0 Snsr [1-16]	IPMB 0 sensors
FRU	FRU file for CMM
all_leds	Target for configuring all user-definable LEDs on the CMM front panel
hsled	Hot swap LED on the CMM front panel
userled[1-4]	Corresponds to userled A-D on the CMM front panel
feedN	Corresponds to power feed (i.e. feed1, feed2). Use the feedcount dataitem to determine number of power feeds for component.
PmsGlobal	Target for PMS global data
PmsProcN	Target for each process monitored where N is the process number
Datasync Status	Datasync Status Sensor
CMM Status	CMM Status Sensor
PmsPieN	Process monitoring process integrity sensors
None	Same as not entering a target

8.4.5 Dataitem Parameter: -d

The dataitem is the parameter, identified by target and/or location, that the user is getting or setting. The dataitem must be given for every CLI command.

8.4.5.1 Location Dataitem lists

Table 23 through Table 29 list the valid dataitems for each location when no target is specified.

Table 23. Dataitem Keywords for All Locations

Dataitem	Description	Get/Set	CLI Get Output	Valid Set Values
listdataItems	Used to find out what data items are available on a target or location.	Get	Listing of all valid data items that can be issued for the specified location or target	N/A
health	Retrieves the health information about a particular location or target.	Get	"Location/Target has no/minor/major/critical problems"	N/A

Table 24. Dataitem Keywords for All Locations Except System

Dataitem	Description	Get/Set	CLI Get Output	Valid Set Values
listgetdataitems	Lists all available dataitems that can be retrieved with cmmget.	Get	Listing of all valid get data items that can be issued for the specified location or target	N/A
listsetdataitems	Lists all available dataitems that can be set with cmmset.	Get	Listing of all valid set data items that can be issued for the specified location or target	N/A
healthevents	Retrieves events that contribute to the health of the location or target. This is a list of events currently active on the location or target. Health events strings are documented in Section 11, "Health Events" on page 104	Get	List of currently active events. E.g. "Major Event : +12V_B Lower critical going low asserted Major Event : +12V_A Lower critical going low asserted"	N/A
listtargets	Used to find what sensors or targets are available on the location. This is the list of sensors defined by the SDR for that particular location.	Get	Listing of all the targets that are available on the location	N/A

Table 25. Dataitem Keywords for All Locations Except Chassis and System (Sheet 1 of 4)

Dataitem	Description	Get/Set	CLI Get Output	Valid Set Values
deviceid	Retrieves the device's SDR support, hardware revision, firmware / software revision, and sensor and event interface command specification revision information. Implements Get Device ID command. See IPMI 1.5 Specification Section 17.1.	Get	"GetDeviceID: < interpreted string without label> Device ID = <Device ID> SDR Support = <device provides Device SDRs> Device Revision = <Device revision> Device Available = <Device available: 0=normal operation, 1=device firmware> Firmware Revision = <Firmware revision: major.minor> IPMI Version = <IPMI version> Chassis Support = <Additional chassis device support> Bridge Support = <Additional bridge support> IPMB Event Generator Support = <Additional IPMB Event Generator support> IPMB Event Receiver Support = <Additional IPMB Event Receiver support> FRU Inventory Support = <Additional FRU inventory device support> SEL Support = <Additional SEL device support> SDR Repository Support = <Additional SDR Repository device support> Sensor Support = <Additional sensor device support> Manufacturer ID = <Manufacturer ID> Product ID = <Product ID> Aux Firmware Revision = <Auxiliary firmware revision information>"	N/A
fruactivation	Set the activation state to either activate or deactivate the FRU. The Deactivate is the same as a Graceful Shutdown.	Set	N/A	1=activate FRU 0=deactivate FRU
fruactivationpolicy	Get or Set the FRU activation policy. A Get returns whether the "Locked Bit" is set. For example, if blade 11 activation locked bit is set, and if in M1, then blade 11 cannot transition to M2 until unlocked. If blade 11 activation locked bit is not set then blade 11 can transition from M1 to M2.	Both	"<location> activation locked bit is set. If in M1, <location> cannot transition to M2 until unlocked" OR "<location> activation locked bit is not set. <location> can transition from M1 to M2"	1=set locked bit 0=clear locked bit

Table 25. Dataitem Keywords for All Locations Except Chassis and System (Sheet 2 of 4)

Dataitem	Description	Get/Set	CLI Get Output	Valid Set Values
frucontrol	Set the FRU payload to do things like Cold Reset, Warm Reset, etc. The CMM location only supports 2 (graceful reboot) and will only work on standby CMM. Using frucontrol on an active or single CMM will attempt a failover before executing the command. If failover is unsuccessful, fruncontrol will not execute and return an error.	Set	N/A	0=cold reset 1=warm reset 2=graceful reboot 3=issue diagnostic interrupt
hotswapstate	Retrieves the FRU's current M state (0-7).	Get	"<location> Hot Swap state is M[x]"	N/A
fruextractionnotify	Used to notify the Shelf Manager that a FRU has been extracted from the shelf. Example is "cmmset -l <location> -d fruextractionnotify -v 1"	Set	N/A	1=Extract FRU
ledproperties	Find out the number and type of LEDs the FRU supports and which LED it can control. Implements the Get FRU LED Properties command. See PICMG 3.0 Section 3.2.5.6.	Get	Information pertaining to number and control of the LEDs "<location> has control of <main_leds> <location> supports <number_user_leds> user leds" Where <location> is the -l parameter (can be a sub FRU) <main_leds> is Comma-separated list of <led> items <led> is hsled, led1, led2, led3 <number_user_leds> is the decimal number of user LEDs supported by FRU	N/A
picmgproperties	Query the maximum FRU Device ID supported by the IPMI controller. Implements Get PICMG Properties command. See PICMG 3.0 Table 3-9.	Get	"PICMG Properties: < interpreted string without label> PICMG Properties ID = <PICMG ID> PICMG Extension Version = <PICMG extension version=major.minor> Max FRU Device ID = <Max FRU device ID> FRU Device ID = <FRU device ID for IPMI controller>"	N/A
powerlevels	Returns the power levels available for a FRU and the number of watts drawn by each.	Get	"ATCA FRU Power Levels: Power Level 1 = A watts ... Power Level n = B watts"	N/A

Table 25. Dataitem Keywords for All Locations Except Chassis and System (Sheet 3 of 4)

Dataitem	Description	Get/Set	CLI Get Output	Valid Set Values
powerstate	Retrieves and sets power state of a blade. Get: This is used to find out the powered on/off or offline state of a blade Set: To reboot, shutdown and turn on a blade If “reset” is used on CMM location, the software will check for redundancy and a reset will only occur if a redundant partner is identified. “PowerOff” is not supported on the CMM location.	Both	"<location>: <currentState> (Mx)" where Mx is the ATCA-defined M state of the location	"Reset" "PowerOff" "PowerOn"
presence	Used to find out if a particular location is occupied or present in the chassis. This can be blades or intelligent entities like fan trays or PEMs.	Get	"<location> is <presenceState>." Where <presenceState> is "present" - if the location is present "not present" - if the location is not present or the CMM can not communicate with it.	N/A
presentpowerlevel	Get/Set the current power level of a FRU.	Get	"The FRU Power Level is <PwrLevel> Consuming <WattageValue> Watts" where <PwrLevel> is the current power level of the FRU in the range 1-20 <WattageValue> is the current power draw in watts	N/A
sel	Returns the System Event Log of the specified location.	Get	Listing of the interpreted SEL log of the location. The listing is of the format: “<Entry1>\n\n<Entry2>…” where <EntryM> is of the format: <Timestamp in Linux date format>\n\t<SensorName>\t<EventDescription>	N/A
grantedboardekeys	Get the EKeys that have been granted to the Board.	Get	Base Interface Ekeys: <EkeyList> Fabric Interface EKeys: <EkeyList> Update Channel Interface EKeys: <EkeyList>" where <EkeyList> is a list of ekey settings for the interface such as Ekey1 : enabled EKey2 : disbaled EKey3 : no set	N/A

Table 25. Dataitem Keywords for All Locations Except Chassis and System (Sheet 4 of 4)

Dataitem	Description	Get/Set	CLI Get Output	Valid Set Values
busedekeys	Get a list of Bused EKeys and who owns them.	Get	Metallic Test Bus Pair #1: Token Owned: Yes/No Owner's IPMBAddress: IPMBAddress Metallic Test Bus Pair #2: Token Owned: Yes/No Owner's IPMBAddress: IPMBAddress Sync Clock Group #1: Token Owned: Yes/No Owner's IPMBAddress: IPMBAddress Sync Clock Group #2: Token Owned: Yes/No Owner's IPMBAddress: IPMBAddress Sync Clock Group #3: Token Owned: Yes/No Owner's IPMBAddress: IPMBAddress where "Owner's IPMBAddress" is displayed when "Token Owned" is set to "Yes".	N/A
totalfrus	Used to query the total number of FRUs in a particular location. Once the number of FRUs for the location is known, the FRU can be specified by the format "-l location:fru#". Not specifying the ":fru#" part will direct the command to FRU ID 0	Get	integer number	N/A
frudeactivationpolicy	Get/Set the deactivation policy of the FRU. In PICMG 3.0 ECN 1 this refers to the deactivation locked bit. The FRU can be specified by the format "-l [location:fru#]. Not specifying the ":fru#" will direct the command to FRU 0.	Both	1 - Locked bit is set 0 - Locked bit is not set	1- Set the locked bit 0 - Clear the locked bit
ipmicommand		Set	Command Response string on success or error code on failure.	Command request string
rawsel	Used to list SEL in raw format.	Get	"Listing of raw format SEL log of the location. The listing is of the format: <Entry1>\n\n<Entry2>… Where : <EntryM> is of the format: <Timestamp in Linux date format>\n\t<SensorName>\t<EventDescription>"	N/A

Table 26. Dataitem Keywords for Chassis Location

dataitem	Description	Get/Set	CLI Get Output	Valid Set Values
fanspeed	Used to get or set the fan speed of all fans in the chassis. Value is percent of the maximum fan speed. See Section 16, “Fan Control and Monitoring” on page 132 for more information.	Both	The percentage of the max speed, "Emergency Shut Down", or "Local Control". For example, 80 for 80% of the max speed.	A numerical value between 0 and 100 (i.e., “70”), “localcontrol”, or “emergencyshutdown” (localcontrol is not supported on the MPCHC0001 chassis fan tray.)
location	This is used to get or set the Location field in the chassis FRU and is sent out as a part of SNMP and UDP alerts. This is only used with the chassis location.	Both	"Shelf Address: <address>" Where: <address> is a space-separated list of two-digit, hex numbers if the address’ type/len byte is 0, decoded string otherwise	Location String less than 16 characters in length.

Table 27. Dataitem Keywords for Cmm Location (Sheet 1 of 7)

dataitem	Description	Get/ Set	CLI Get Output	Valid Set Values
listlocations	Used to find out all the locations that can be queried. This list can contain both present and non-present locations.	Get	All possible locations in the shelf e.g. “cmm blade1 blade2 blade3 blade4 blade5 blade6 blade7 blade8 blade9 blade10 blade11 blade12 blade13 blade14 FanTray PEM1 PEM2 chassis system”	N/A
listpresent	Used to query the locations that are currently present in the shelf that the CMM can communicate with.	Get	All the present locations in the shelf. It is the subset of listlocation. e.g.: “cmm blade1 blade2 FanTray PEM1 PEM2 chassis system”	N/A
clearsel	Clears the event log of the entire shelf. cmmset -d clearsel -v clear	Set	N/A	clear
powerbudget	Get information about the overall power budget, how much is used, how much available	Get	"Shelf Power Budget Distribution: Feed #1 = A Watts Feed #2 = B Watts Feed #3 = B Watts ... Feed #N = D Watts"	N/A

Table 27. Dataitem Keywords for Cmm Location (Sheet 2 of 7)

dataitem	Description	Get/ Set	CLI Get Output	Valid Set Values
alarmcutoff	Retrieve or set the state of the Telco Alarm cutoff. When enabled, it silences the Telco alarm for active events and blinks the event LEDs on the CMM. This dataitem is only valid when used with the cmm as the location and is used to set the alarm cutoff or get its value.	Both	"Telco Alarm Cutoff is <enabled/ disabled>."	1 = Set cut off 0 = Unset cut off
alarmtimeout	Retrieve or set the timeout value in minutes for the Telco Alarm cutoff. This is the amount of time before the alarm cutoff will automatically become unset if the user doesn't unset it themselves. This dataitem is only valid when used with the cmm as the location and is used to set the alarm timeout or get its value.	Both	"Timeout is <timeoutvalue> minutes."	Number of minutes: 0-1000. Value of 0 disables the time-out.
criticalled minorled majorled	Used only with the CMM location to turn on or off the critical, major and minor leds. When used with cmmset, a –v value of 1 turns the LED on while a 0 turns it off.	Both	"1" if the LED is On "0" if the LED is Off	1 - Turn On LED 0 - Turn Off LED
Ethernet	Included for backward compatibility only. The mapping of the command for existing dataitem is: Ethernet = EthernetA	Both	"front" or "rear" or "backplane" For example, bash-2.04# cmmget -d ethernet cmm1ethernetA: front cmm2ethernetA: front	"Front" – Set direction to the front panel. "Back" – Set direction to the rear IO panel card.

EthernetA
EthernetB
	Used only with the CMM location to change the eth0/ eth1 direction to either the front panel, the rear panel IO card, or backplane.
	The mapping of the command for existing dataitems are:
	EthernetA = cmm1EthernetA + cmm2EthernetA
	EthernetB = cmm1EthernetB + cmm2EthernetB
		Both	"Front" or "Rear" or "Backplane". For example,
			bash-2.04# cmmget -d ethernetA
			cmm1ethernetA: front
			cmm2ethernetA: front
			bash-2.04# cmmget -d ethernetB
			cmm1ethernetB: front
			cmm2ethernetB: front
				"Front", "Rear", or "Backplane"

Table 27. Dataitem Keywords for Cmm Location (Sheet 3 of 7)

dataitem	Description	Get/ Set	CLI Get Output	Valid Set Values
cmm1EthernetA cmm1EthernetB cmm2EthernetA cmm2EthernetB	Used only with the CMM location to change the eth0/ eth1 direction to either the front panel, the rear panel IO card, or backplane on CMM1 and/or CMM2.	Both	"Front" or "Rear" or "Backplane". For example, bash-2.04# cmmget -d cmm1ethernetA cmm1ethernetA: front bash-2.04# cmmget -d cmm1ethernetB cmm1ethernetB: front bash-2.04# cmmget -d cmm2ethernetA cmm2ethernetA: front bash-2.04# cmmget -d cmm2ethernetB cmm2ethernetB: front	"Front", "Rear", or "Backplane"
version	The version of the CMM software.	Get	"Version: [Generation].[SRA].[Patch].[Build]" where Generation: Firmware Generation. SRA: Release in that Generation. Patch: Patch number. Build: Build number. E.g. Version:5.1.0.11	N/A
update	Used with the cmmget command to update the CMM firmware on the CMM. In a redundant system, updates should only be done on one CMM at a time in order to maintain chassis management. Refer to Section 23, “Updating CMM Software” on page 204 for additional information.	Set	N/A	"<cmm image location> ftp:<hostname or IP address>:username :password"
redundancy	Returns the list of CMMs in the shelf and their status.	Get	CMM 1: Present (active) * CMM 2: Not Present (standby) * = The CMM you are currently logged into.	N/A
failover	Used with cmmset from the active CMM to force a failover to the standby. This will only complete successfully if the standby CMM is in a state where it can handle a failover.	Set	N/A	“1” = Failover to standby CMM with equal or newer firmware version. “any”= Failover to standby CMM regardless of firmware version.
rmcpenable	Enable/Disable RMCP interface.	Both	"1" - RMCP Enabled "0" - RMCP disabled	"1" - RMCP Enable "0" - RMCP disable

Table 27. Dataitem Keywords for Cmm Location (Sheet 4 of 7)

dataitem	Description	Get/ Set	CLI Get Output	Valid Set Values
snmpenable	Used to set or query SNMP trap enabled status.	Both	SNMP traps are <enabled/disabled>.	1=Enable 0=Disable enable disable
snmptrapaddress[1-5]	Get or Set the machine's IP address that will receive SNMP traps from a location. Up to five addresses can be set. Default is 0.0.0.0 for all 5. Example: cmmset -l cmm -d snmptrapaddress3 -v 10.10.241.105	Both	"SNMP trap address: IpAddress" where IpAddress is of the format A.B.C.D	<IpAddress> Where: <IpAddress> is a Valid IP address in the form: A.B.C.D
snmptrapcommunity	Get or Set the SNMP trap community name. Example: cmmget -l cmm -d snmptrapcommunity Returns: SNMP trap community: publiccmm	Both	"SNMP trap community: communityValue"	<CommunityName > Where <CommunityName > is any Valid SNMP community name 64 characters or less. Ex: publiccmm
snmptrapport	Get or Set the TCP/IP port that the SNMP trap will be sent to. The default is 162.	Both	"SNMP trap port: portNumber"	Valid port number 0-65535
snmptrapversion	Retrieves or sets SNMP trap version. This is either v1 or v3.	Both	"SNMP trap version: v1/v3"	"v1" or "v3"
airfilterruntimelimit	Returns the uppercritical limit. Note: It uses the sensor to display the runtime value in days since the last reset. To retrieve the uppernoncritical limit use the command: cmmget -t "filter run time" -d uppernoncritical (or -d thresholdsall)	Both	<uppercritical limit> Days	1. Disable eventing on air filter run time -v 0 2. Enable eventing on air filter and set the uppercritical limit to (xxx) days, and it also sets the upper non-critical value to 90% of the uppercritical. -v xxx 3. Enable eventing on air filter and set the uppercritical limit to (xxx) days and the upper non-critical to (yyy) days -v xxx,yyy xxx and yyy must be an integer between 1 and 255.
resetairfilterruntime	Resets the air filter runtime to 0. The set is supported to allow the user to set the run time to zero when the filter is replaced.	Set	N/A	1 - Only accepts a value of 1

Table 27. Datаitem Keywords for Cmm Location (Sheet 5 of 7)

dataitem	Description	Get/ Set	CLI Get Output	Valid Set Values
syncuserledstate	Gets/Sets whether the LED state is synced between the active and standby CMM.	Both	"Yes" or "No"	“Yes” or “No”

powersequence	Used to get/set the power sequence order, Power Sequencing Delay, ShelfManagerControlledActivation in the CDM.
	Note: The power sequencing delay is in tenths of a second to delay before powering up any other FRU after powering this FRU. The value of the power sequencing delay is between 0 and 63. Shelf Manager Controlled Activation determines if the Shelf Manager activates the FRU residing at this location when it reaches M2.
		Both	It will be in INI format displayed on the console as follows:
			[Settings]
			EntryCount=xxx
			[Power Sequence 1]
			Location=cmm1
			FRUDeviceID=0
			ShelfManagerControlledActivation=Yes
			DelayBeforeNextPowerOn=0
			[Power Sequence 2]
			… …
			[Power Sequence xxx]
			Location=blade12
			FRUDeviceID=0
			ShelfManagerControlledActivation=No
			DelayBeforeNextPowerOn=0
				INI file with its path, such as "-v /home/ PowerSeq.INI"

loginmessage	Used to customize the login screen message by allowing user to add the OEM name.	Both	The OEM string that is displayed at the login screen	"OEMWelcomeMessage" where the OEMWelcomeMessage is the message that will appear at the login screen Max length = 63 characters
cmdlineprompt	Used to customize the bash prompt by allowing user to add the OEM name.	Both	The OEM Name string to be prepended to the $PS1 variable	"OEMName" where the OEMName is the string that will appear at the beginning of the bash prompt. Max length = 63 characters
FaultLEDColor	Get/Set the color of the fault/ health LED on the CMM fronted FRUs (Filter Tray, CDM) to be used when an error is reported. Does not affect CMM Health LED.	Both	"amber" or "red"	“amber” or “red”

Table 27. Dataitem Keywords for Cmm Location (Sheet 6 of 7)

dataitem	Description	Get/ Set	CLI Get Output	Valid Set Values
AdminState	Used to set or query the adminstrative state of the PMS as a whole or an individual monitored process. A target of “PmsGlobal” will get/set the state of the PMS as a whole. A target of “PmsProc[#]” will get/ set the unique state of an individual process, where # is the unique process number for the process. A target of "PmsPie[#]" will get/set the unique state of an PIE, where # is the unique pie number. AdminState is CMM-specific and is not synced between CMMs. It allows individual control of each CMMs adminstate and can be set on either active or standby CMM.	Both	"1:Unlocked" or "2:Locked"	1 = Unlocked 2 = Locked
RecoveryAction	Used to set or query the recovery action of a PMS monitored process. This is only valid for a target of "PmsProc[#]. Where "#" is the unique number for the process.	Both	"1:No Action", "2:Process Restart", "3: Failover and Restart", or "4:Failover and Reboot"	1 = No Action 2 = Process Restart 3 = Failover and Restart 4 = Failover and Reboot
EscalationAction	Used to set or query the process restart escalation action. This is only valid for a target of "PmsProc[#]. Where "#" is the unique number for the process.	Both	"1:No Action", "2:Failover and Reboot"	1 = No Action 2 = Failover and Reboot
ProcessName	Used to query the process name and associated command line arguments for a monitored process. A target of "PmsProc[#]” will retrieve the name of an individual process, where "#" is the unique number for the process. "PmsPie[#]" will retrieve the path and command line arguments, of the PIE to be executed periodically.	Get	"<Process_Name> <Command_Line_Arguments>"	N/A
OpState	Used to query the operational state of a monitored process. An operational state of disabled indicates that the process has failed and cannot be recovered. This is valid for a target of "PmsProc[#]” and “PmsGlobal”, where "#" is the unique number for the process, and PmsGlobal refers to the OpState for all of PMS. This is also valid for a target of Pie[#].	Get	"1:Enabled", "2:Disabled"	N/A

Table 27. Dataitem Keywords for Cmm Location (Sheet 7 of 7)

dataitem	Description	Get/ Set	CLI Get Output	Valid Set Values
standbycmmreboot	Used to request to reboot the standby CMM from the active CMM.	Set	N/A	1 - Request to reboot the standby CMM
feedcount	Get the power feed count for that location. Determines number of feed targets (i.e., feed1) for that location. See Section 7.7, “Power Feed Targets” on page 69.	Get	Integer number	N/A
failoveronredundancy	Used to set the failover configuration flag	Both	automatic, manual	automatic, manual
syncuserscripts	Used to set the direction of synchronization for home scripts when the CMM versions differ.	Both	“upgrade”, “downgrade”, “always”, “equal”	upgrade, downgrade, always, equal
snmptrapformat	Uset to get/set SNMP trap format	Both	“1” - text “2” -raw “3”-text+raw	“1” - text “2” - raw “3” - text+raw
snmpsendunrecognizedevents	Used to get/set the option if to send unrecognized events. Used only when snmptrapformat is set to 1.	Both	0 - don't send 1 - send	“0” - don't send “1” - send
selformat	Used to get/set the option if to send unrecognized events. Used only when snmptrapformat is set to 1.	Both	“0” - don't send “1” - send	“0” - don't send “1” - send
seldisplayunrecognizedevents	Used to get/set the option if to display unrecognized events in SEL. Used only when selformat is set to 1.	Both	“0” - don't display “1” - display	“0” - don't display “1” - display
temperaturelevel	Used to query the current temperature level of the fantray.	Get	“Normal”, “Minor”, “Major”, “Critical”	N/A

Table 28. Dataitem Keywords for System Location

dataitem	Description	Get/ Set	CLI Get Output	Valid Set Values
unhealthylocations	Used to query which locations have active health events	Get	"Critical : CritList Major : MajList Minor : MinList" where each list is a list of locations having that level of health events (space separated)	N/A
clearmajor	Clear major alarm LED on the active CMM.	Set	N/A	1 - Only accepts a value of 1
clearminor	Clear minor alarm LED on the active CMM.	Set	N/A	1 - Only accepts a value of 1

Table 29. Dataitem Keywords for FantrayN Location

dataitem	Description	Get/ Set	CLI Get Output	Valid Set Values
minorlevel	Used to set or query the minorlevel for the fantray.	Both		Any value between the normallevel and the majorlevel of the fantray.
normallevel	Used to set or query the normallevel for the fantray.	Both		Any value between the minimumsetting and the minorlevel of the fantray.
control	Used to set or query the control mode of the fantray.	Both		EmergencyShutdown fantray CMM defaultcontrol
defaultcontrol	Used to set or query the defaultcontrol mode of the fantray.	Both		fantray CMM
restoredefaults	Used to restore the cooling table defaults of the fan tray to the vendor defaults or code defaults.	Set	N/A	true
minimumsetting	Used to query the minimum setting of the fantray returned via the getfantray properties IPMI command.	Get		N/A
maximumsetting	Used to query the maximum setting of the fantray returned via the getfantray properties IPMI command.	Get		N/A
recommendedsetting	Used to query the recommended setting of the fantray returned via the getfantray properties IPMI command.	Get		N/A
currentfanlevel	Used to query the current cooling level of the fantray.	Get	0	N/A

8.4.5.2 Target Dataitem Lists

When a target is specified, there is usually a slightly different set of dataitems specifically for that target. Refer to Section 8.4.4, “Target Parameter: -t” on page 78 for more information on the target parameter. Table 30 lists the possible dataitems used with various targets.

Table 30. Dataitem Keywords Used with the Target Parameter (Sheet 1 of 4)

dataitem	Description	Get/ Set	CLI Get Output	Valid Set Values
listdataitems	Lists the available dataitems for that target.	Get	Listing of all valid data items that can be issued for the specified location or target	N/A
health	Returns the health of the target and if any events exist. The returned values will be one of OK, minor, major, or critical.	Get	"Location/Target has no/minor/major/ critical problems"	N/A
healthevents	Returns the specific health events that are occurring on the target if any exist.	Get	List of currently active events. E.g. "Major Event : +12V_B Lower critical going low asserted Major Event : +12V_A Lower critical going low asserted"	N/A
current	The current value of a sensor.	Get	"The current value is currentValue [Units]"	N/A
thresholdsall	All thresholds of a sensor. This includes lower non-recoverable, lower critical, lower non-critical, upper non-critical, upper critical, and upper non-recoverable.	Get	"Upper Non-recoverable: ThresholdValue [Units] Upper Critical: ThresholdValue [Units] Upper Non-critical: ThresholdValue [Units] Lower Non-critical: ThresholdValue [Units] Lower Critical: ThresholdValue [Units] Lower Non-recoverable: ThresholdValue [Units]" If a certain threshold is not supported, the ThresholdValue will display "Not Supported"	N/A
uppernonrecoverable uppercritical uppernoncritical lowernoncritical lowercritical lowernonrecoverable	Used to query individual thresholds for a value based sensor, such as temperature or voltage.	Get	One of the following: “Upper Critical: ThresholdValue [Units]” “Upper Non-critical: ThresholdValue [Units]” “Lower Non-critical: ThresholdValue [Units]” “Lower Critical: ThresholdValue [Units]” “Lower Non-recoverable: ThresholdValue [Units]” “Lower Non-recoverable: ThresholdValue [Units]”	N/A

Table 30. Dataitem Keywords Used with the Target Parameter (Sheet 2 of 4)

dataitem	Description	Get/ Set	CLI Get Output	Valid Set Values
criticalaction majoraction minoraction normalaction	Used to configure user-defined actions when events occur. This dataitem is used with a target (–t) parameter specified sensor and a value (-v) parameter. When an event happens for that particular sensor, then the script defined in the –v parameter will be executed. The script to be executed must be located in the /home/scripts/ directory on the CMM and the /home/scripts path should be omitted when specifying the script. Example: cmmset -l blade9 -t +5V -d minoraction -v “powerdownblade 9” In this example, /home/scripts/powerdownblade will be executed with a parameter of 9 when the +5V sensor on blade1 generates a minor event.	Both	If set, the full path of the script e.g. /home/scripts/EventScript. If not set, output is ““ (null).	"<ScriptName> arg1 arg2 …argN" Where Script name (not full path) is the script file name and arg1-argN are the parameters to the script. Use “none” to remove an existing entry.
eventaction	Used to trigger a script based on event code of a health event. Refer to Section 18, “CMM Scripting” on page 164	Set	N/A	“<event code>:<ScriptName> arg1 arg2...argN” Where event code is the event code associated with the event to associate with the script. ScriptName (not full path) is the name of the script file, and arg1..argN are any parameters required with the script. Use “<eventcode>:none” to remove an existing entry.
ledcolorprops	Gets a FRU LED’s valid color set. This command returns a comma separated list of supported colors, the default local control color, and the default override color. This command should be issued before a ledstate set command. Implements the Get LED Color Capabilities command. See PICMG 3.0 table 3-24.	Get	Color properties of the LED "<ledtarget> supports <colors> Default local control color is <colorList> Default override color is <color>" Where: <ledtarget> is One of the valid LEDs (hsled, led1, led2, led3, userled1-userled251) <colorList> is Comma-separated list of <color> items <color> is one of blue, red, green, amber, orange, white	N/A

Table 30. Dataitem Keywords Used with the Target Parameter (Sheet 3 of 4)

dataitem	Description	Get/ Set	CLI Get Output	Valid Set Values
ledstate	Gets or Sets a FRU LED's state. The Get returns the LED's mode, one of {localcontrol, override, or lamptest}, and a function message. Implements the Get/ Set FRU LED State commands. See PICMG 3.0 tables 3-26 for Get and 3-25 for Set. Set syntax model: cmmset -l <location> -t <LED> -d ledstate -v <function>, <function options> Example: cmmset -l cmm -t "userled1" -d ledstate -v blink,300,700,green This sets the CMM's user1 LED to blinking green with an off duration of 300 ms and an on duration of 700 ms.	Both	"<ledtarget> is in <LEDmode> mode <function message>" where <LEDMode> is one of localcontrol/ override/ lamptest <function message> is one of the following, depending on the LED's current function: If LED is off: function is off If LED is on: function is on color is <color> If LED is blinking: function is blink off time is <offtime> ms on time is <ontime> ms color is <color> If LED is under lamp test: duration is <duration> ms <Color> is one of blue, red, green, amber, orange, white <Offtime> is Time in milliseconds that the LED is in the off cycle of a blink <Ontime> is Time in milliseconds that the LED is in the on cycle of a blink <Duration> is The duration of the lamp test in milliseconds	Functions: off, on, blink, lamptest, localcontrol Accepted values: <offtime>, <ontime>,<color>, <duration> Refer to Section 12.5, "Setting the State of the User LEDs" on page 125 for more information.

Table 30. Dataitem Keywords Used with the Target Parameter (Sheet 4 of 4)

dataitem	Description	Get/ Set	CLI Get Output	Valid Set Values
maxexternalavailable current	Get/Set the field in the CDM regarding the max external available current. Only used with the feedN target. e.g. cmmget -l cmm -t feed1 -d maxexternalavailablecurrent See Section 7.7, "Power Feed Targets" on page 69.	Both	current in Amps with 1 decimal point	current in Amps with 1 decimal point.
maxinternalcurrent	Get the field in the CDM regarding the max internal available current. Only used with the feedN target. e.g. cmmget -l cmm -t feed1 -d maxinternalcurrent See Section 7.7, "Power Feed Targets" on page 69.	Get	current in Amps with 1 decimal point	N/A
minexpectedoperating voltage	Get/Set the field in the CDM regarding the max expected operating voltage. Only used with the feedN target. e.g. cmmget -l cmm -t feed1 -d minexpectedoperatingvoltage See Section 7.7, "Power Feed Targets" on page 69.	Both	voltage value in string between -36 to -72v	voltage value in string between -36 to -72v

8.4.6 Value Parameter: -v

The `value` parameter specifies the new value for a `dataitem`. This parameter is required for all **cmmset** commands and is only used for **cmmset** commands. Valid `value` parameters are shown in with their corresponding `dataitem`s in the data item tables listed above.

8.4.7 Sample CLI Operations

Sample CLI Operations can be found in Appendix A, "Example CLI Commands".

8.5 Generating a System Status Report

The CLI includes an executable script (cmmdump) that is used to generate a system status report for use in communicating system health and configuration information to technical support personnel. This is useful in helping technical support successfully troubleshoot any issues that may be affecting the system. Cmmdump outputs system information to the screen by default or to a file. To send the output to a file use the following command:

```
cmmdump > [filename]
```

The filename should refer to a file that is in a valid directory (i.e. /home/cmmdump.txt).The file can then be retrieved off the CMM using FTP (see Section 8.3.7, “FTP into the CMM” on page 76).

Resetting the Password 9

It may become necessary at some point to reset the CMM password to its default of ***cmmrootpass***. The CMM has one on board dip switch labeled S2-1 to perform this action. Refer to the *Intel® NetStructure™ MPCMM0001 Hardware Technical Product Specification* for the location of the switch. Setting the switch and powering up the CMM will cause the password to reset to its default. The CMM then needs to be removed and the switch then needs to be turned off again.

9.1 Resetting the Password in a Dual CMM System

In redundant systems containing dual CMMs, one active, one standby, the password should be reset on the standby CMM. Once reset to its default, the default password will synchronize itself to the active CMM. This prevents the need to perform the reset on both CMMs and a failover.

1. Open the ejector latch on the standby CMM and wait for the blue hot swap LED to illuminate, indicating the CMM is safe to remove from the system.
2. Remove the standby CMM from the chassis.
3. Set dip switch S2-1 to “on”. The dip switch has a label indicating which way is on.
4. Re-insert the CMM into the system and allow the CMM to fully boot (blue light will go off when fully booted).
5. An OK health event will occur indicating that the passwords on both CMMs have been reset and were synched from the standby CMM. A SEL entry will be recorded, and a trap will be sent out.
6. Once at the login prompt, the password should now be reset to its default of ***cmmrootpass***.
7. Login to the active CMM to ensure the password was reset.
8. Open the ejector latch on the standby CMM and wait for the blue hot swap LED to illuminate, indicating the CMM is safe to remove from the system.
9. Remove the standby CMM from the chassis.
10. Set dip switch S2-1 back to original “off” position.
11. Re-insert the CMM into the system and allow the CMM to fully boot (blue light will go off when fully booted).
12. Login to the CMM and operate as normal.
13. Use the **passwd** command on the active CMM to change the CLI password if desired. The new password will sync to the standby.

9.2 Resetting the Password in a Single CMM System

For nonredundant systems that contain only a single CMM, resetting the password will require removing the CMM. This will cause any boards that are power controlled by the CMM become unmanaged. Care should be taken to safely shut down boards in the system prior to removing the CMM.

1. Safely shut down and power off boards being power controlled by the CMM.
2. Remove the CMM from the system.
3. Set dip switch S2-1 to “on”. The dip switch has a label indicating which way is on.
4. Re-insert the CMM into the system and allow the CMM to fully boot (blue light will go off when fully booted).
5. Once at the login prompt, the password should now be reset to its default of ***cmmrootpass***.
6. Login to the CMM to ensure the password was reset.
7. Remove the CMM from the system.
8. Set dip switch S2-1 back to its original “off” position.
9. Re-insert the CMM into the system and allow the CMM to fully boot (blue light will go off when fully booted).
10. Login to the CMM and operate as normal.
11. Use the **passwd** command on the active CMM to change the CLI password if desired.

Sensor Types 10

10.1 CMM Sensor Types

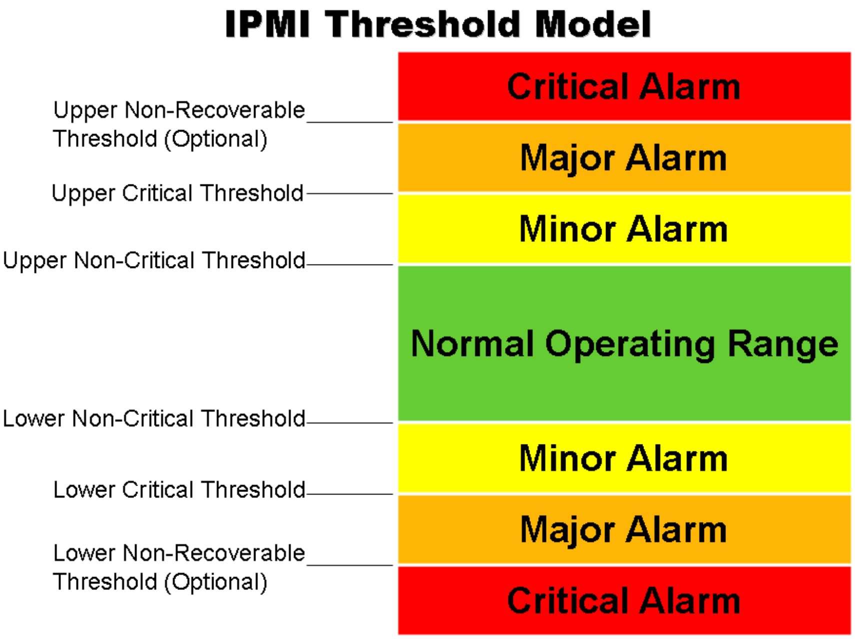
The CMM provides access to and can log events from different IPMI sensor types. These sensors can be both threshold-based sensors or discrete sensors, depending on the type of reading and events they generate. For more information on sensors and sensor types, refer to the “Intelligent Platform Management Interface Specification v1.5”, available on http://www.intel.com.

10.2 Threshold-Based Sensors

Threshold-based sensors are sensors that generate or change an event status based on comparing a current value to a threshold value for a given hardware monitor device. Examples of threshold-based sensors are temperature, voltage, and fan tachometer sensors.

10.2.1 Threshold-Based Sensor Events

Threshold-based sensors generate events when a current value for a device becomes greater than or less than a given threshold value. The IPMI specification defines six thresholds that can be assigned to a given sensor. The sensor will generate an event when its current reading goes above or below any of these thresholds. The severity of the event generated depends on which threshold is crossed. These six thresholds and their associated event severity are as follows:



10.3 CMM Voltage/Temp Sensor Thresholds

Table 31 shows the threshold sensors present on the CMM with the Upper Non-Recoverable, Upper Critical, Upper Non-Critical, Lower Non-Critical, Lower Critical, and Lower Non-Recoverable thresholds for each sensor. UseTable 34 on page 109 to determine the severity for an event that crosses a specific threshold on a sensor.

Table 31. CMM Voltage and Temp Sensor Thresholds

Sensor Name	UNR	UC	UNC	LNC	LC	LNR
VBAT	N/A	3.494 (3.463)	3.292 (3.26)	2.496 (2.527)	1.997 (2.028)	N/A
VTT DDR	N/A	1.427 (1.404)	N/A	N/A	1.053 (1.076)	N/A
+2.5V	N/A	2.621 (2.597)	N/A	N/A	2.363 (2.387)	N/A
+3.3V	N/A	3.477 (3.443)	N/A	N/A	2.976 (3.01)	N/A
+5V	N/A	5.486 (5.434)	5.304 (5.252)	4.68 (4.732)	4.472 (4.524)	N/A
+12V	N/A	15.12 (14.994)	13.041 (12.915)	11.088 (11.214)	8.064 (8.19)	N/A
CPU Core V	N/A	1.357 (1.334)	N/A	N/A	1.229 (1.252)	N/A
Brd Temp	100 (97)	67 (64)	60 (57)	10 (13)	N/A	N/A
CPU Temp	80 (77)	67 (64)	63 (60)	10 (13)	5 (8)	N/A
FilterTrayTemp 1	80 (77)	40 (37)	30 (27)	2 (5)	-2 (1)	N/A
FilterTrayTemp 2	80 (77)	40 (37)	30 (27)	2 (5)	-2 (1)	N/A
Filter Run Time	N/A	192 (189)	128 (125)	N/A	N/A	N/A

Note: Value in parenthesis is the de-assert value.

10.4 Discrete Sensors

Discrete sensors are sensors that have a predefined set of states. An example of a discrete sensor would be the FRU Hot Swap sensor type, which monitors the hot swap state of a FRU. The Hot Swap sensor for a FRU will always be in one of the predefined hot swap states (M0-M7).

10.4.1 Discrete Sensor Events

Discrete sensors can generate events on state changes for various sensors in the system. The severity of the event is determined by the CMM.

Health Events 11

This section describes the health events that are associated with CMM events and includes the syntax and severity of these strings. Refer to Section 10, “Sensor Types” on page 101 for more information on the different type of sensors.

Note: For this section, “health event” (two words) refers to any event that is generated that reports the health of a sensor. “healthevents” (one word) refers specifically to the “healthevents” dataitem or the output of that dataitem (healthevents query).

11.1 Syntax of Health Event Strings

Health event string is a string that is associated with the particular health event occurring. Active events are displayed in a healthevents query using the following command in the CLI:

```
cmmget -l [location] [-t [target]] -d healthevents
```

Active health events are also returned when healthevents queries are executed over SNMP. In addition, all health events are logged in the SEL and sent out as SNMP Traps.

Note: SEL entries and SNMP traps do not include the severity of the event. Only healthevents query results display the severity of an event.

11.1.1 Healthevents Query Event Syntax

The following is the syntax of a string returned by a healthevents query for an associated active health event. The \n represents newline characters (seen as <0A> in SNMP queries).

```
[Timestamp]\n
[Severity] Event : [SDR Sensor Name] [Health Event String]: [Event Type]\n
```

- Timestamp is in the format: `[Day] [Month] [Date] [HH:MM:SS] [Year].` For example, Thu Dec 11 22:20:03 2003
- Severity is either Minor, Major, or Critical.
- SDR Sensor Name is the name given to the sensor in the Sensor Data Record (SDR).
- Health Event Strings are listed in Section 11.4, “List of Possible Health Event Strings” on page 108.
- Event Type is either “Assertion Event” or “Deassertion Event” stating whether it is an event that is being asserted or one that is being deasserted. This is not applied to threshold-based sensor events since the string already states event type.

11.1.2 SEL Event Syntax

The following is the syntax of each SEL entry for an associated health event. The \n represents newline characters and the \t represents tab characters.

```
[Timestamp]\n
```

```
\t[SDR Sensor Name]\t[Health Event String]: [Event Type]\n\n
```

- Timestamp is in the format: `[Day] [Month] [Date] [HH:MM:SS] [Year].` For example, Thu Dec 11 22:20:03 2003
- SDR Sensor Name is the name given to the sensor in the Sensor Data Record (SDR).
- Health Event Strings are listed in Section 11.4.
- Event Type is either “Assertion Event” or “Deassertion Event” stating whether it is an event that is being asserted or one that is being deasserted. This is not applied to threshold-based sensor events since the string already states event type.

11.1.3 SEL Sensor Types

The following table describes the Sensor Types supported on the CMM and the Sensor Type Codes used for SEL entries.

Note: When entering event codes to associate action scripts, the decimal value should be used. The hex values are given for reference only.

Table 32. CMM SEL Sensor Information

Sensor Name	Sensor Type Code		Descriptions
	Hex	Decimal	
Temperature	01h	1	Threshold exceeded for upper critical, upper noncritical, lower critical and lower non-critical thresholds. Refer to Table 31, “CMM Voltage and Temp Sensor Thresholds” on page 102 for sensor thresholds data.
Voltage	02h	2	Voltage exceeded upper critical, upper non-critical, lower critical and lower non-critical thresholds. Refer to Table 31, “CMM Voltage and Temp Sensor Thresholds” on page 102 for sensor thresholds data.
Filter Run Time	0xC0	192	Filter Run Time
CMM Redundancy	0xD1	209	CMM Redundancy Sensor Type. Several event types are classified under this sensor type and can use the same event offsets for generating events. Refer to table below.
BIST	0xD8	216	Built-In Self Test Sensor
CMM Status	0xD9	217	CMM Status ready or not ready sensor type
PMS Fault	0xDA	218	Sensor type for Process Monitoring events
PMS Info	0xDB	219	Target for PMS global data
FRU Hot Swap	0xF0	240	AdvancedTCA* Hot Swap Sensor
IPMB-0 Snsr [1-16]	0xF1	241	IPMB 0 sensors
Datasync Status	0xDE	222	Datasync Status Sensor

11.1.4 SNMP Trap Event Syntax

The following is the syntax of each SNMP trap for an associated event:

```
Time : [TimeStamp] , Location : [ChassisLocation] , Chassis Serial # :
[ChassisSN] , Board : [Location] , Sensor : [SDR Sensor Name] , Event :
[Health Event String]: [Event Type} , Event Code : [Event Code]
```

- Timestamp is in the format: `[Day] [Month] [Date] [HH:MM:SS] [Year].` For example, Thu Dec 11 22:20:03 2003
- ChassisLocation is the chassis location information recorded in the chassis FRU.
- ChassisSN is the chassis serial number records in the chassis FRU.
- Location is the location where the sensor generating the event is located (i.e., CMM)
- SDR Sensor Name is the name given to the sensor in the Sensor Data Record (SDR).
- Health Event Strings are listed in Section 11.4, "List of Possible Health Event Strings" on page 108.
- Event Type is either "Assertion Event" or "Deassertion Event" stating whether it is an event that is being asserted or one that is being deasserted. This is not applied to threshold-based sensor events since the string already states event type.
- Event Code is a code unique to that event. The codes for each event can be found below in Table 11.4, "List of Possible Health Event Strings" on page 108.

11.2 Sensor Targets

Available sensors for a location can be retrieved with the listtargets dataitem argument of the cmmget command. To view a list of sensor targets on the cmm using the listtargets command:

```
cmmget -l cmm -d listtargets
```

List of targets for the CMM can be found in Table 19, "Remote Procedure Calls (RPC)" on page 174.

For complete lists of sensors on other components (i.e., voltage sensors on blade1), refer to the corresponding Technical Product Specification for that product.

Table 33. Sensor Targets (Sheet 1 of 2)

Sensor Target	Description
Brd Temp	Board Temperature Sensor
CPU Temp	CPU Temperature Sensor
FilterTrayTemp [1/2]	Filter Tray Temperature Sensor
CPU Core V	CPU Core Voltage Sensor
VBAT	Battery Sensor
VTT DDR	VTT Sensor
+2.5V	+2.5 Volt Sensor
+3.3V	+3.3 Volt Sensor
+5V	+5 V olt Sensor
+12V	+12 Volt Sensor
CDM [1/2]	CDM 1 or 2 Sensor

Table 33. Sensor Targets (Sheet 2 of 2)

Sensor Target	Description
Air Filter	Air Filter Sensor
Filter Tray	Filter Tray Sensor
Filter Run Time	Filter Tray Run Time Sensor
BIST	BIST Sensor
CMM Status	CMM Status Sensor
DataSync Status	Datasync Sensor
FRU Hot Swap	FRU Hot Swap Sensor
Filter Tray HS	Filter Tray Hot Swap Sensor
IPMB-0 Snsr [1-16]	IPMB-0 Sensors
FRU	FRU Sensor
all_leds	all_leds sensor
hsled	Hot Swap LED Sensor
userled[1-4]	User LED Sensors
feed[1-4]	Power Feed Sensors
PmsGlobal	Process Monitoring Global Sensor
PmsProc[2-180]	Process Monitoring Process Sensors

11.3 Healthevents Queries

Healthevents queries, whether done from the CLI or through SNMP, output the same event strings and syntax for active events. This section explains the various types of output that are retrieved from healthevents queries. For more information on using the healthevents dataitem, refer to Section 8, “The Command Line Interface (CLI)” on page 71.

11.3.1 HealthEvents Queries for Individual Sensors

Executing a healthevents query on a particular sensor target returns all active healthevents for that sensor target in a concatenated string. One sensor may have multiple events. For example, running the following healthevents query on the CMM BIST sensor:

```
cmmget -l cmm -t BIST -d healthevents
```

might return multiple events that are active on the BIST sensor in a concatenated string like this:

```
Mon Feb 2 19:51:05 2004
Major Event : BIST RTC Not Working
Mon Feb 2 19:51:09 2004
Major Event : BIST Both Ethernet interfaces are not working.
```

11.3.2 HealthEvents Queries for All Sensors on a Location

Executing a healthevents query on the “cmm” location in the CLI without a target specified (*cmmget -l cmm -d healthevents*) returns all the healthevents for all CMM sensors in a concatenated string. This includes all BIST, LAN, Telco Alarm, Voltage, and Temp sensors on the CMM. This ability to retrieve all healthevents on a location also applies to the chassis, bladeN, FantrayN and PemN* locations.

11.3.3 No Active Events

When a healthevents query is executed in the CLI on a target that doesn’t have active events a string will be returned that is a single line string with no timestamp or severity. Only this string will be returned and won’t be concatenated with any other strings. For example, doing a healthevents query from the CLI of a location or target that doesn’t have any active healthevents will return the following string:

“[Target] has no problems.” - i.e., “cmm has no problems” or “Brd Temp has no problems”

Executing a healthevents query through SNMP on a target with no active events returns different values than the CLI does. When a healthevents query is executed in SNMP on a location or a target (i.e., in SNMP, the cmmHealthEvents object or cmmSensorHealthEvents object) that has no active events, the value returned is a zero length string.

Note: Attempting to execute a healthevents query on the “FRU” target will return a “CLI Invalid Data Item Error.” string since “FRU” is not a sensor. This also applies to LED targets and FeedN targets.

11.3.4 Not Present or Non-IPMI Locations

Doing a healthevents query of a blade or powersupply, or a target on a blade or powersupply, that is not present or non-IPMI compliant will return one of the following:

“BPM Blade Not Present.” - if querying an empty blade slot.

“BPM Non IPMI Blade.” - if querying a blade that is present but doesn’t support IPMI.

11.4 List of Possible Health Event Strings

The following tables list strings, codes, and severities for sensor events generated on cmm1 (cmm in left slot), cmm2 (cmm in right slot), chassis, fantrays, PEM1 (PEM on left when looking from front of chassis) and PEM2 (PEM on right when looking from front of chassis). All listed healthevents are logged in a SEL entry, sent out as an SNMP Trap, and show up in a healthevents query while the event is active. OK Health Events are displayed in healthevents queries for a limited time when they are asserted.

For each event, the event code that is returned with each event in SNMP is listed. SNMP traps sent from CMM include the event code with each trap. Users can search for and customize their SNMP applications to use the event code to search for and perform actions on an event instead of needing to use the entire event string.

The event codes below can also be used with the eventaction dataitem to trigger scripts based on when individual events occur. Eventaction uses the base-10 version of the below hexadecimal codes. For more information on eventaction scripting refer to Section 18, “CMM Scripting” on page 164.

The table for threshold-based sensors is common to other threshold-based sensors on other components, e.g., voltage, temp, current).

11.4.1 All Locations

Table 34. Threshold-Based Sensors: Voltage, Temp, Current, Fan

Event String	Event Code		Event Severity
	Hex	Decimal	
"Lower non-critical going low asserted"	0x010	16	Minor
"Lower non-critical going high asserted "	0x011	17	Minor
"Lower critical going low asserted"	0x012	18	Major
"Lower critical going high asserted"	0x013	19	Major
"Lower non-recoverable going low asserted"	0x014	20	Critical
"Lower non-recoverable going high asserted"	0x015	21	Critical
“Upper non-critical going low asserted"	0x016	22	Minor
“Upper non-critical going high asserted"	0x017	23	Minor
"Upper critical going low asserted"	0x018	24	Major
"Upper critical going high asserted"	0x019	25	Major
"Upper non-recoverable going low asserted"	0x01A	26	Critical
"Upper non-recoverable going high asserted"	0x01B	27	Critical
"Lower non-critical going low deasserted"	0x01C	28	OK
"Lower non-critical going high deasserted"	0x01D	29	OK
"Lower critical going low deasserted"	0x01E	30	OK
"Lower critical going high deasserted"	0x01F	31	OK
"Lower non-recoverable going low deasserted"	0x020	32	OK
"Lower non-recoverable going high deasserted"	0x021	33	OK
"Upper non-critical going low deasserted"	0x022	34	OK
"Upper non-critical going high deasserted"	0x023	35	OK
"Upper critical going low deasserted"	0x024	36	OK
"Upper critical going high deasserted"	0x025	37	OK
"Upper non-recoverable going low deasserted"	0x026	38	OK
"Upper non-recoverable going high deasserted"	0x027	39	OK

Table 35. Hot Swap Sensor: Filter Tray HS, FRU Hot Swap

Event String	Event Code		Event Severity
	Hex	Decimal	
“[FRU] transitioned from [M0-M7] to [M0-M7] Due to Normal State Change: [Asserted,Deasserted]”	N/A	N/A	Major (if M0-M0) else OK
“[FRU] transitioned from [M0-M7] to [M0-M7] Due to Unknown Reasons: [Asserted,Deasserted]”	N/A	N/A	Major (if M0-M0) else OK
“[FRU] transitioned from [M0-M7] to [M0-M7] Due to Operator changing the handle switch: [Asserted,Deasserted]”	N/A	N/A	Major (if M0-M0) else OK
“[FRU] transitioned from [M0-M7] to [M0-M7] Due to Programmatic action: [Asserted,Deasserted]”	N/A	N/A	Major (if M0-M0) else OK
“[FRU] transitioned from [M0-M7] to [M0-M7] Due to Information Provided by User/System: [Asserted,Deasserted]”	N/A	N/A	Major (if M0-M0) else OK
“[FRU] transitioned from [M0-M7] to [M0-M7] Due to Surprise Extraction: [Asserted,Deasserted]”	N/A	N/A	Major (if M0-M0) else OK
“[FRU] transitioned from [M0-M7] to [M0-M7] Due to Communication Failure caused by Local Malfunction: [Asserted,Deasserted]”	N/A	N/A	Major (if M0-M0) else OK
“[FRU] transitioned from [M0-M7] to [M0-M7] Due to Invalid Hardware Address: [Asserted,Deasserted]”	N/A	N/A	Major (if M0-M0) else OK
“[FRU] transitioned from [M0-M7] to [M0-M7] Due to Invalid Hardware Address: [Asserted,Deasserted]”	N/A	N/A	Major (if M0-M0) else OK
“[FRU] transitioned from [M0-M7] to [M0-M7] Due to Command by Shelf Manager with Set FRU Activation: [Asserted,Deasserted]”	N/A	N/A	Major (if M0-M0) else OK
“[FRU] transitioned from [M0-M7] to [M0-M7] Due to Communication Failure: [Asserted,Deasserted]”	N/A	N/A	Major (if M0-M0) else OK
“[FRU] State M0 - Not Installed”	0x0130	304	N/A
“[FRU] State M1 - Inactive”	0x0131	305	N/A
“[FRU] State M2 - Activation Request”	0x0132	306	N/A
“[FRU] State M3 - Activation in Progress”	0x0133	307	N/A
“[FRU] State M4 - Active”	0x0134	308	N/A
“[FRU] State M5 - Deactivation Request”	0x0135	309	N/A
“[FRU] State M6 - Deactivation in Progress”	0x0136	310	N/A
“[FRU] State M7 - Communication Lost: Asserted”	0x0137	311	Major

Table 36. IPMB Link State Sensor: IPMB-0 Snsr [1-16]

Event String	Event Code		Event Severity
	Hex	Decimal	
"IPMB-A disabled, IPMB-B disabled"	0x0140	320	Major
"IPMB-A enabled, IPMB-B disabled"	0x0141	321	Major
"IPMB-A disabled, IPMB-B enabled"	0x0142	322	Major
"IPMB-A enabled, IPMB-B enabled"	0x0143	323	OK

Table 37. System Firmware Progress Event Strings (System Firmware Progress)

Event String	Event Code		Event Severity
	Hex	Decimal	
"System Firmware Error: Unspecified error occurred"	0x050	592	OK
"System Firmware Error: No system memory installed"	0x251	593	Major
"System Firmware Error: No usable system memory found"	0x252	594	Major
"System Firmware Error: Unrecoverable hard disk/ATAPI/IDE device"	0x253	595	Major
"System Firmware Error: Unrecoverable system-board failure"	0x254	596	Major
"System Firmware Error: Unrecoverable diskette subsystem failure"	0x255	597	Major
"System Firmware Error: Unrecoverable hard disk controller failure"	0x256	598	Major
"System Firmware Error: Unrecoverable PS/2 or USB keyboard failure"	0x257	599	Major
"System Firmware Error: Removable boot media not found"	0x258	600	Major
"System Firmware Error: Unrecoverable video controller failure"	0x259	601	Major
"System Firmware Error: No video device detected"	0x25A	602	Major
"System Firmware Error: Firmware (BIOS) ROM corruption detected"	0x25B	603	Major
"System Firmware Error: CPU voltage mismatch"	0x25C	604	Major
"System Firmware Error: CPU speed matching failure"	0x25D	605	Major
"System Firmware Progress: Unspecified error occurred"	0x260	608	OK
"System Firmware Progress: Memory initialization"	0x261	609	OK
"System Firmware Progress: Hard disk initialization"	0x262	610	OK
"System Firmware Progress: Secondary processor(s) initalization"	0x263	611	OK
"System Firmware Progress: User authentication"	0x264	612	OK
"System Firmware Progress: User-initiated system setup"	0x265	613	OK
"System Firmware Progress: USB resource configuration"	0x266	614	OK
"System Firmware Progress: PCI resource configuration"	0x267	615	OK
"System Firmware Progress: Option ROM initialization"	0x268	616	OK
"System Firmware Progress: Video initialization"	0x269	617	OK
"System Firmware Progress: Cache initialization"	0x26A	618	OK
"System Firmware Progress: SM Bus initialization”	0x26B	619	OK
"System Firmware Progress: Keyboard controller initialization”	0x26C	620	OK
"System Firmware Progress: Embedded/Management controller initialization”	0x26D	621	OK
"System Firmware Progress: Docking station attachment”	0x26E	622	OK

Table 37. System Firmware Progress Event Strings (System Firmware Progress)

Event String	Event Code		Event Severity
	Hex	Decimal	
"System Firmware Progress: Enabling docking station”	0x26F	623	OK
"System Firmware Progress: Docking station ejection”	0x270	624	OK
"System Firmware Progress: Disabling docking station”	0x271	625	OK
"System Firmware Progress: Calling OS wake-up vector”	0x272	626	OK
"System Firmware Progress: Starting OS boot process”	0x273	627	OK
"System Firmware Progress: Baseboard or motherboard initialization”	0x274	628	OK
"System Firmware Progress: Floppy initialization”	0x275	629	OK
"System Firmware Progress: Keyboard test”	0x276	630	OK
"System Firmware Progress: Pointing device test”	0x277	631	OK
"System Firmware Progress: Primary processor initialization”	0x278	632	OK
"System Firmware Error: Timer count read/write error”	0x280	640	Critical
"System Firmware Error: CMOS battery error”	0x281	641	Major
"System Firmware Error: CMOS diagnosis error”	0x282	642	Major
"System Firmware Error: CMOS checksum error”	0x283	643	Major
"System Firmware Error: CMOS memory size error”	0x284	644	Major
"System Firmware Error: RAM read/write test error”	0x285	645	Critical
"System Firmware Error: CMOS date/time error”	0x286	646	Major
"System Firmware Error: Clear CMOS jumper”	0x287	647	OK
"System Firmware Error: Clear password jumper”	0x288	648	OK
"System Firmware Error: Manufacturing jumper”	0x289	649	OK
"System Firmware Error: Microcontroller in update”	0x28A	650	Major
"System Firmware Error: Microcontroller response failure”	0x28B	651	Major
"System Firmware Error: Event Log full”	0x28C	652	OK
"System Firmware Hang: Unspecified error occurred”	0x460	1120	OK
"System Firmware Hang: Memory initialization”	0x461	1121	Major
"System Firmware Hang: Hard disk initialization”	0x462	1122	Major
"System Firmware Hang: Secondary processor(s) initialization”	0x463	1123	Major
"System Firmware Hang: User authentication”	0x464	1124	Major

Table 37. System Firmware Progress Event Strings (System Firmware Progress)

Event String	Event Code		Event Severity
	Hex	Decimal	
"System Firmware Hang: User-initiated system setup”	0x465	1125	Major
"System Firmware Hang: USB resource configuration”	0x466	1126	Major
"System Firmware Hang: PCI resource configuration”	0x467	1127	Major
"System Firmware Hang: Option ROM initialization”	0x468	1128	Major
"System Firmware Hang: Video initialization”	0x469	1129	Major
"System Firmware Hang: Cache initialization”	0x46A	1130	Major
"System Firmware Hang: SM Bus initialization”	0x46B	1131	Major
"System Firmware Hang: Keyboard controller initialization”	0x46C	1132	Major
"System Firmware Hang: Embedded/Management controller initialization”	0x46D	1133	Major
"System Firmware Hang: Docking station attachment”	0x46E	1134	Major
"System Firmware Hang: Enabling docking station”	0x46F	1135	Major
"System Firmware Hang: Docking station ejection”	0x470	1136	Major
"System Firmware Hang: Disabling docking station”	0x471	1137	Major
"System Firmware Hang: Calling OS wake-up vector”	0x472	1138	Major
"System Firmware Hang: Starting OS boot process”	0x473	1139	Major
"System Firmware Hang: Baseboard or motherboard initialization”	0x474	1140	Major
"System Firmware Hang: Floppy initialization”	0x475	1141	Major
"System Firmware Hang: Keyboard test”	0x476	1142	Major
"System Firmware Hang: Pointing device test”	0x477	1143	Major
"System Firmware Hang: Primary processor initialization”	0x478	1144	Major

Table 38. Watchdog 2 Sensor Event Strings (Sheet 1 of 2)

Event String	Event Code		Event Severity
	Hex	Decimal	
"Timer expired status only”	0x350	848	OK
"Hard reset”	0x351	849	OK
"Power down”	0x352	850	OK
"Power cycle”	0x353	851	OK
"Timer interrupt generated”	0x354	852	OK

Table 38. Watchdog 2 Sensor Event Strings (Sheet 2 of 2)

Event String	Event Code		Event Severity
	Hex	Decimal	
"None interrupt type BIOS FRB2 timer”	0x355	853	OK
"None interrupt type BIOS/POST timer”	0x356	854	OK
"None interrupt type OS Load timer”	0x357	855	OK
"None interrupt type SMS/OS timer”	0x358	856	OK
"None interrupt type OEM timer”	0x059	857	OK
"None interrupt type unspecified timer”	0x35A	858	OK
"SMI interrupt type BIOS FRB2 timer”	0x35B	859	OK
"SMI interrupt type BIOS/POST timer”	0x35C	860	OK
"SMI interrupt type OS Load timer”	0x35D	861	OK
"SMI interrupt type SMS/OS timer”	0x35E	862	OK
"SMI interrupt type OEM timer”	0x35F	863	OK
"SMI interrupt type unspecified timer”	0x360	864	OK
"NMI interrupt type BIOS FRB2 timer”	0x361	865	OK
"NMI interrupt type BIOS/POST timer”	0x362	866	OK
"NMI interrupt type OS Load timer”	0x363	867	OK
"NMI interrupt type SMS/OS timer”	0x364	868	OK
"NMI interrupt type OEM timer”	0x365	869	OK
"NMI interrupt type unspecified timer”	0x366	870	OK
"Messaging interrupt type BIOS FRB2 timer”	0x367	871	OK
"Messaging interrupt type BIOS/POST timer”	0x368	872	OK
"Messaging interrupt type OS Load timer”	0x369	873	OK
"Messaging interrupt type SMS/OS timer”	0x36A	874	OK
"Messaging interrupt type OEM timer”	0x36B	875	OK
"Messaging interrupt type unspecified timer”	0x36C	876	OK
"Unspecified interrupt type BIOS FRB2 timer”	0x36D	877	OK
"Unspecified interrupt type BIOS/POST timer”	0x36E	878	OK
"Unspecified interrupt type OS Load timer”	0x36F	879	OK
"Unspecified interrupt type SMS/OS timer”	0x370	880	OK
"Unspecified interrupt type OEM timer”	0x371	881	OK
"Unspecified interrupt type unspecified timer”	0x372	882	OK

11.4.2 CMM Location

Table 39. CMM Redundancy

Event String	Event Code		Event Severity
	Hex	Decimal	
"Regained"	0x0D0	208	OK
"Established"	0x0D1	209	OK
"Lost due to switch failure"	0x0D2	210	Major
"Lost due to unhealthy signal"	0x0D3	211	Major
"Lost due to CMM removal"	0x0D5	213	Major
"Lost due to CMM reboot or halt"	0x0D6	214	Major
"Lost due to critical event on the CMM"	0x0D7	215	Major
"Lost due to inability to communicate with the other CMM over its management bus."	0x0D8	216	Major
"Lost due to incompatible firmware versions on the CMMs. Please upgrade the CMMs to the same version."	0x0EF	239	Major

Table 40. CMM Trap Connectivity (CMM [1-2] Trap Conn)

Event String	Event Code		Event Severity
“Regained”	0x0D0	208	OK
"Lost due to failed network connectivity between the CMM and the Primary SNMP trap destination (SNMPTrapAddress1)."	0x0D4	212	Major

Table 41. CMM Failover (Sheet 1 of 2)

Event String	Event Code		Event Severity
	Hex	Decimal	
“Failover occurred."	0x0D9	217	OK
“Failover occurred because of switch failure.”	0x0DA	218	OK
“Failover occurred because of failed network connectivity between the CMM and the Primary SNMP trap destination.”	0x0DB	219	OK
“Failover occurred because of critical CMM health events.”	0x0DC	220	OK
“Failover occurred because of a bad hardware signal from the other CMM.”	0x0DD	221	OK
“Failover occurred because it was forced by the user.”	0x0DE	222	OK
“Failover cannot occur because the other CMM has a bad switch.”	0x0E0	224	OK
“Failover cannot occur because the other CMM has lost network connectivity with its Primary SNMP trap destination.”	0x0E1	225	OK
“Failover cannot occur because the other CMM has critical health events.”	0x0E2	226	OK

Table 41. CMM Failover (Sheet 2 of 2)

Event String	Event Code		Event Severity
	Hex	Decimal	
“Failover cannot occur because the other CMM is not responding over its management bus.”	0x0E3	227	OK
“Failover cannot occur because the critical items have not been synced.”	0x0E4	228	OK
“Failover cannot occur because the other CMM has a bad hardware signal.”	0x0E5	229	OK
“Failover occurred because the active CMM had older firmware than the newly active CMM."	0x0F0	240	OK
“Failover cannot occur because the Standby is not present.”	0x0F1	241	OK
“Failover cannot occur because the standby has an older version of the firmware.”	0x0F2	242	OK
“Failover cannot occur because the Standby failover state discovery is not finished.”	0x0F3	243	OK
"Failover occurred because it was initiated by Process Monitoring."	0x018F	399	OK

Table 42. CMM Synchronization (Sheet 1 of 2)

Event String	Event Code		Event Severity
	Hex	Decimal	
“Could not copy the /etc/CMM.cfg file to the standby CMM."	0x0E6	230	OK
“Could not copy the /etc/passwd and /etc/shadow files to the standby CMM.”	0x0E7	231	OK
“Default password files synced to active CMM. Passwords have been reset to default.”	0x0E8	232	OK
“Could not copy the /etc/sel_* files to the standby CMM.”	0x0E9	233	OK
“Could not copy the /etc/snmpd.conf and /etc/var/snmpd.conf files to the standby CMM.”	0x0EA	234	OK
“Could not copy the SDR file to the standby CMM.”	0x0EB	235	OK
“Could not copy the /etc/scripts directory to the standby CMM.”	0x0EC	236	OK
“All files successfully synced.”	0x0ED	237	OK
"Could not copy the PmsSync.ini file to the standby CMM."	0x0EE	238	OK
“Could not copy the cmm.ini file to the standby CMM.”	0x0F4	244	OK
“Could not copy the command privilege file to the standby CMM.”	0x0F5	245	OK
“Could not copy the command prompt file to the standby CMM.”	0x0F6	246	OK
“Could not copy the login message files to the standby CMM.”	0x0F7	247	OK
“Could not copy the SDR repository information to the standby CMM.”	0x0F8	248	OK
“Could not copy the EKEY information to the standby CMM.”	0x0F9	249	OK
“Could not copy the power and hotswap information to the standby CMM.”	0x0FA	250	OK

Table 42. CMM Synchronization (Sheet 2 of 2)

Event String	Event Code		Event Severity
	Hex	Decimal	
“Could not copy the FRU manager information to the standby CMM.”	0x0FB	251	OK
“Could not copy the IPMB state information to the standby CMM.”	0x0FC	252	OK
“Could not copy the user LED information to the standby CMM.”	0x0FD	253	OK
“Could not copy the cooling state information to the standby CMM.”	0x0FE	254	OK
“Could not copy the sensors state information to the standby CMM.”	0x0FF	255	OK
"Could not copy the PmsShadowSync.ini file to the standby CMM."	0x01A2	418	OK

Table 43. BIST Event Strings (Sheet 1 of 2)

Event String	Event Code		Event Severity
	Hex	Decimal	
“Event-log area full.”	0x071	113	Critical
“RedBoot image corrupted. Using golden image.”	0x072	114	Major
“Backup RedBoot image corrupted.”	0x073	115	Critical
“FPGA image corrupted. Using backup image.”	0x074	116	Major
“Backup FPGA image corrupted.”	0x075	117	Critical
“OS image corrupted.”	0x076	118	Critical
“Base memory test failed.”	0x077	119	Critical
“Extended memory test failed.”	0x078	120	Critical
“FPGA1 firmware outdated.”	0x079	121	Minor
“FPGA2 firmware outdated.”	0x07A	122	Minor
“FPGA 1+2 version mismatched.”	0x07B	123	Minor
“RTC date is invalid.”	0x07C	124	Major
“RTC time is invalid.”	0x07D	125	Major
“RTC is not working.”	0x07E	126	Critical
“One of the Ethernet interfaces is not working.”	0x07F	127	Minor
“Both Ethernet interfaces are not working.”	0x080	128	Major
“CMM Boot.”	0x081	129	OK
“IPMB bus [0-29] busy/not ready.”	0x082 - 0x09F	130-159	Major
“Software update successful.”	0x0A0	160	OK
“Update of RedBoot failed.”	0x0A1	161	Critical
“Update of FPGA failed.”	0x0A2	162	Critical
“Update of Bluecat OS failed.”	0x0A3	163	Critical

Table 43. BIST Event Strings (Sheet 2 of 2)

Event String	Event Code		Event Severity
	Hex	Decimal	
“Update of /etc failed.”	0x0A4	164	Critical
“Restore of /etc files failed.”	0x0A5	165	Critical
“Software update failed.”	0x0A6	166	Critical
“FPGA re-programmed 2 times and no further lockup detected.”	0x0B4	180	Minor
“FPGA re-programmed 3 times and no further lockup detected.”	0x0B5	181	Minor
“FPGA re-programming has failed.”	0x0B6	182	Critical
“FPGA re-programmed more than 3 times and lockup still detected.”	0x0B7	183	Critical
“Dynamic FLASH image corruption detected.”	0x0B8	184	Critical
“Static Redboot image is corrupt.”	0x0B9	185	Critical
“Static BlueCat image is corrupt.”	0x0BA	186	Critical
“Static FPGA image is corrupt.”	0x0BB	187	Critical
"CPLD firmware outdated."	0x0BC	188	Minor
"RTC battery power lost."	0x0BE	189	Critical
"I2C failure while reading RTC."	0x0BF	190	Critical
"PBI frequency was not 33MHz. Now recovered after the reset."	0x0150	336	OK

Table 44. Chassis Data Module (CDM [1,2])

Event String	Event Code		Event Severity
	Hex	Decimal	
“Removed”	0x0120	288	OK
"Inserted"	0x0121	289	OK

Table 45. Datasync Status

Event	Event Code		Event Severity
	Hex	Decimal	
“Initial Data Synchronization complete”	0x420	1056	OK

Table 46. CMM Status Event Strings (CMM Status)

Event String	Event Code		Event Severity
	Hex	Decimal	
“CMM is not ready.”	0x400	1024	Minor
“CMM is ready.”	0x401	1025	OK

Table 46. CMM Status Event Strings (CMM Status)

Event String	Event Code		Event Severity
	Hex	Decimal	
“CMM is Active”	0x402	1026	OK
“CMM is Standby”	0x403	1027	OK
“CMM ready timed out”	0x404	1028	Minor

Table 47. Process Monitoring Service Fault Event Strings (PMS Fault)

Event String	Event Code		Event Severity
	Hex	Decimal	
"Process existence fault; attempting recovery"	0x0170	368	User-Configurable
"Process integrity fault; attempting recovery"	0x0171	369	User-Configurable
"Thread watchdog fault; attempting recovery"	0x0172	370	User-Configurable
"Process existence fault; monitoring disabled”	0x0173	371	User-Configurable
"Process integrity fault; monitoring disabled"	0x0174	372	User-Configurable
"Thread watchdog fault; monitoring disabled"	0x0175	373	User-Configurable
"Excessive reboots/failovers; all process monitoring disabled"	0x0176	374	User-Configurable
"Recovery successful"	0x0177	375	OK
“Monitoring initialized”	0x0178	376	OK

Note: For more information on Process Monitoring Service commands refer to Section 6, “Process Monitoring and Integrity” on page 41.

Table 48. Process Monitoring Service Info Event Strings (PMS Info)

Event String	Event Code		Event Severity
	Hex	Decimal	
"Take no action specified for recovery"	0x0179	377	User-Configurable
"Attempting process restart recovery action"	0x017A	378	User-Configurable
"Attempting process failover & restart recovery action"	0x017B	379	User-Configurable
"Attempting process failover & reboot recovery action"	0x017C	380	User-Configurable
"Take no action specified for escalated recovery"	0x017D	381	User-Configurable
"Attempting failover & reboot escalated recovery action"	0x017E	382	User-Configurable
"Process restart recovery failure"	0x017F	383	User-Configurable
"Failover & reboot recovery failure"	0x0180	384	User-Configurable
"Recovery failure due to excessive restarts"	0x0181	385	User-Configurable
"Failover & reboot escalated recovery failure"	0x0182	386	User-Configurable
"Internal fault detected; monitoring disabled"	0x0183	387	User-Configurable

Note: PMS Info events are only written to the SEL, not SNMP traps, healthevents, LEDs, or telco alarms.

11.4.3 Chassis Location

Table 49. Chassis Events

Event String	Event Code		Event Severity
	Hex	Decimal	
"cmm.sif file not found. Cannot scan CMM sensors."	0x0101	257	Major
"chassis.sif file not found. Cannot scan Chassis sensors."	0x0102	258	Major

11.5 IPMI Error Completion Codes

Whereas versions of firmware prior to 5.2 gave a single error message for any IPMI communication issue, Version 5.2 firmware and above implements more granular IPMI error code reporting, allowing the return of the exact IPMI error code that is returned by the IPMI message response. This is specified in the IPMI 1.5 specification as the first byte in the data field of the IPMI response. This allows more accurate analysis to be done in regards to the IPMI communication between the CMM and a IPMI sensor/device.

The following table shows the IPMI Completion codes from the IPMI 1.5 specification as well as their respective enumeration and error message given by the CMM.

Table 50. IPMI Error Completion Codes and Enumerations

Code	Error Message
C0h	“Node Busy”
C1h	“Invalid Command”
C2h	"Command Invalid for Given LUN"
C3h	"Timeout while processing command"
C4h	“Out of space”
C5h	"Reservation Canceled or Invalid Reservation ID"
C6h	"Request Data Truncated"
C7h	"Request Data Length Invalid"
C8h	"Request Data Field Length Limit Exceeded"
C9h	"Parameter Out of Range"
CAh	"Cannot Return Number of Requested Data Bytes"
CBh	"Requested Sensor, Data, or Record not present"
CCh	"Invalid Data Field in Request"
CDh	"Command illegal for specified sensor or record type"
CEh	"Command response could not be provided"
CFh	"Cannot execute duplicated request"
D0h	"Command response could not be provided. SDR Repository in update mode"
D1h	"Command response could not be provided. Device in firmware update mode"
D2h	"Command response could not be provided. BMC initialization or initialization agent in progress"
D3h	"Destination unavailable"
D4h	"Cannot execute command.”
D5h	"Cannot execute command. Command, or request parameter(s), not supported in present state"
FFh	"Unspecified Error"
01h – 7Eh	"OEM Error"

11.5.1 Configuring IPMI Error Completion Codes

A configuration parameter is available to allow for the enhanced IPMI error code reporting. A new configuration variable “IMBErrorCodeReporting” is defined in /etc/cmm.cfg file. It will have two values 0 (disabled) and 1 (enabled). The default value of “IMBErrorCodeReporting” is disabled (0). To turn on the enhanced IPMI error code reporting, the “IMBErrorCodeReporting” variable should be set to 1.

11.5.2 IPMI/IMB Error Message Format

When “IMBErrorCodeReporting” is set to 0 in /etc/cmm.cfg, the current error message is displayed whenever IMB completion error occurs:

```
"IMB ERROR Completion Code Error"
```

When “IMBErrorCodeReporting” is set to 1 in /etc/cmm.cfg, then the following message will be displayed:

```
IMB ERROR Completion Code Error: [IPMI Error Code] : [Error String]
```

Where:

IPMI Error Code: Ascii String of the IPMI error code returned in the IPMI response. The IPMI codes are listed in Table 50, “IPMI Error Completion Codes and Enumerations” on page 121.

Error String: Error message for the corresponding error code as displayed in Table 50, “IPMI Error Completion Codes and Enumerations” on page 121

Example 1.

If the CMM receives error code D3h (Destination unavailable) from a FRU, the following message will be displayed:

```
"IMB ERROR Completion Code Error: D3h : Destination unavailable"
```

Front Panel LEDs 12

The CMM has nine LEDs on the front panel of each unit for displaying system health, hot swap state, and other information. They include:

- Three system health LEDs
- One CMM Health LED
- One CMM HotSwap State LED
- Four User-Definable LEDs (A-D)

For more information on CMM LEDs refer to the *Intel® NetStructure™ MPCMM0001 Hardware Technical Product Specification.*

12.1 LED Types and States

The CMM can retrieve and, in some cases, set values for LEDs on the CMM, fan trays, PEMs, and blades in the chassis. Refer to Section 8, "The Command Line Interface (CLI)" on page 71 for more information. The following tables list the default values for the LEDs on the CMM. Other locations will likely have different LED properties that can be retrieved through the CMM.

12.1.1 Alarm LEDs

There are three alarm LEDs on the CMM corresponding to the minor (!), major (!!) and critical (!!!) alarm states. The LEDs are amber when on. Each LED has the following meaning:

Table 51. System Health LED States

LED (Symbol)	Status	Description
Minor Alarm (!)	Off	No Minor Alarm active
	On	Minor Alarm active
	Flashing	Minor Alarm active, but silenced
Major Alarm (!!)	Off	No Major Alarm active
	On	Major Alarm active
	Flashing	Major Alarm active, but silenced
Critical Alarm (!!!)	Off	No Critical Alarm active
	On	Critical Alarm active
	Flashing	Critical Alarm active, but silenced

The Alarm LEDs on the CMM can be turned on or off using the minorled, majorled, and criticalled CMM dataitems. This also affects the relay contacts. The value is 1 or 0. For example, the command to turn the criticalled off is:

```
cmmset -d criticalled -v 0
```

12.1.2 Health LED

Each CMM maintains a single health LED (✚) to provide the status of the CMM.

Table 52. CMM Health LED States

Color	Description
Off	No power to CMM
Solid Green	Normal operation, power okay
Blinking Green	CMM in standby mode
Solid Red	Attention status (CMM is unhealthy due to critical power error)

12.1.3 Hot Swap LED

Each CMM maintains a single blue hot swap LED (⇙⇗) to provide the status of the CMM itself. The hotswap LED cannot have its state set or changed and is readable only.

Table 53. CMM Hot Swap LED States

Color	Description
Off	In use
Long Blink	Searching for CMM (900ms on, 100ms off)
Solid Blue	Ready to remove
Short Blink	Preparing for extraction (100ms on, 900ms off)

12.1.4 User Definable LEDs

Each CMM provides four LEDs (A, B, C, D on the front panel and userled [1, 2, 3, 4] in the CLI, respectively) that can be controlled via the CMM. Each LED can be off, green, yellow, or red.

During the boot process the user LEDs sequentially blink off to indicate boot progress. The user LEDs will be off by the time the CMM software is fully loaded. Once the CMM is up, the administrator can control the LED through standard interfaces or via programmatic control.

12.2 Retrieving a Location’s LED properties

The properties for a location’s LED control status and the number of User LEDs a location supports can be retrieved using the command:

```
cmmget -l [location] -d ledproperties
```

12.3 Retrieving Color Properties of LEDs

The valid colors that an LED supports and the default color properties for that LED can be retrieved using the command:

```
cmmget -l [location] -t [LED] -d ledcolorprops
```

12.4 Retrieving the State of LEDs

The state of an LED on a location can be retrieved using the command:

```
cmmget -l [location] -t [LED] -d ledstate
```

12.5 Setting the State of the User LEDs

The state of the User LEDs on the CMM or other FRUs can be changed using the ledstate set command. Here is the syntax and the table below lists the function values and options:

```
cmmset -l [location] -t [LED] -d ledstate -v [function],[function options]
```

Example:

```
cmmset -l cmm -t "userled1" -d ledstate -v blink,300,700,green
```

This sets the CMM's user1 LED to blinking green with an off duration of 300 ms and an on duration of 700 ms. The following table list functions that can be used when setting the state of LEDs:

The all_leds target can also be used to set all User LEDs on the FRU to the specified value(s). For example, to set all four user LEDs to 'on' and the color 'green' use the following command:

```
cmmset -l cmm -t 'all_leds' -d ledstate -v on,green
```

Note: The ledstate dataitem on the all_leds target is a set-only item. Doing a cmmget -d ledstate on all_leds target will return an error.

Table 54. Ledstate Functions and Function Options

[function] values	description	[function options]
`off`	Turns the target LED(s) off.	no options
`on`	Turns the target LED(s) on.	`[color]`
`blink`	Makes the target LED(s) blink at the specified rate.	`[offtime],[ontime],[color]`
`lamptest`	Puts the target LED(s) into lamp test mode for the specified time.	`[duration],[color]`
`localcontrol`	Puts the target LED(s) back into normal operation.	no options

The following are possible values for function options:

- `color` = one of blue, red, green, amber, orange, white, default, nochange
- `offtime` = the time in milliseconds that the LED is in the off cycle of a blink. Granularity=10 ms Min=10 ms Max=2500 ms
- `ontime` = the time in milliseconds that the LED is in the on cycle of a blink. Granularity=10 ms Min=10 ms Max=2500 ms
- `duration` = the duration of the lamp test in milliseconds. Granularity=100 ms Min=100 ms Max=12700 ms

12.6 LED Boot Sequence

During the boot process, the user LEDs will change in a pattern, as described in Table 55, "LED Event Sequence", to indicate boot progress. The user LEDs will be off by the time the CMM software is fully loaded. Once the CMM is up, the administrator can control the LED through standard interfaces or via programmatic control. The following table describes the sequence of events following the insertion of the CMM and the corresponding LED state for each event.

Table 55. LED Event Sequence

Event	Health LED	Hot Swap LED	LEDA	LEDB	LEDC	LEDD
Initial insertion/ power on, with ejector latch closed	Flash red, then solid green	Solid blue	Off	Off	Off	Off
Redboot initialization	Solid green	Momentarily off, then solid blue	Solid green	Off	Off	Off
Redboot initialization finished, user script running	Solid green	Solid blue	Solid green	Solid green	Off	Off
Linux initialization finished, OS at init level 1	Solid green	Solid blue	Solid green	Solid green	Solid green	Off
CMM init script running, command handler app loaded, CMM at M1	Solid green	Momentarily off, then solid blue	Momentarily off, then solid green	Momentarily off, then solid green	Momentarily off, then solid green	Solid green
Initial CMM initialization finished (i.e., FRU election), CMM at M2	Solid green	Long blink, 100 ms. Off, 900 ms solid blue	Off	Off	Off	Off
CMM at M3 or M4	Solid green	Off	Off	Off	Off	Off

Node Power Control 13

The CMM controls power to the nodes of a chassis. The CMM can power up, power down, and reset a board in a particular slot and can be used to query the operational state of a board at any time. The CMM also manages the overall power budget of the shelf as well as power budget of each power feed.

The active CMM is responsible for power management when there are two CMMs operating in a redundant mode. Critical power management data is kept in sync at all times between the active and standby CMM.

13.1 Node Operational State Management

The CMM manages FRU insertions, extractions, and the operational states and state transitions of the nodes in a shelf based on Section 3.2.4 of the PICMG 3.0 Specification.

13.2 Obtaining the Power State of a Board

The CMM can obtain the power state information of a board at any time by issuing the following command:

```
cmmget -l bladeN -d powerstate
```

Where N is the number of the physical slot number in which the blade you are querying resides. This command will give information on whether the blade is present, the power state, and the board state.

13.3 Controlling the Power State of a Board

The CMM can power off a board, power on a board, or reset the board.

13.3.1 Powering Off a Board

The following command will power off a board:

```
cmmset -l bladeN -d powerstate -v poweroff
```

N is the physical slot number where the blade to be powered down resides. Once issued, the command will ask for confirmation by entering “y” before continuing.

13.3.2 Powering On a Board

The following command will power on a board:

```
cmmset -l bladeN -d powerstate -v poweron
```

N is the number of the physical slot the blade to be reset resides in.

13.3.3 Resetting a Board

The following command will reset a board:

```
cmmset -l bladeN -d powerstate -v reset
```

N is the number of the physical slot where the blade to be reset resides. Once issued, the command will ask for confirmation by entering “y” before continuing.

Electronic Keying Manager 14

Electronic Keying (EKeying) is used in the AdvancedTCA* architecture to dynamically implement a specific fabric interconnect in a fabric agnostic backplane. The PICMG* 3.0 specification calls out two types of EKeying: point-to-point and bused.

14.1 Point-to-Point EKeying

Point-to-point EKeying is used to set up a specific fabric interconnect and protocol between two end points when a board is inserted into the chassis.

In point-to-point EKeying, the CMM queries the topology of the interconnects in the shelf from the shelf FRU multi-records, determines each board’s EKeys from the Board FRU multi-records and attempts to find the best match possible between the two interconnected end-points. Once the match is made, the CMM informs each of the entities to enable its interconnect and which protocol to use. If no match is found, the two end points are informed to disable their interconnect.

14.2 Bused EKeying

Bused EKeying is used to manage control of the bused resources provided by an AdvancedTCA shelf. These resources include the Synchronization Clock Interface and the Metallic Test Bus.

With bused EKeying, the CMM grants control of a specific resource to a single requesting board. Only one board can control a resource at any given time. The CMM controls the resources through the use of tokens. A board can request the token for a particular resource from the CMM at any time. If the CMM has possession of the token for that resource, it grants the token to the requesting board. If the CMM does not have possession of the token, the requesting board is notified, and the token owner is notified that it will need to release the token as soon as possible.

14.3 EKeying CLI Commands

The CLI on the CMM includes the following dataitems used with the cmmget command to obtain EKeying information for the system: Refer to Section 8, “The Command Line Interface (CLI)” on page 71 for more information on these CLI dataitems.

- grantedboardekeys - Retrieves the EKeys that have been granted to the board.
- busedekeys - Retrieves a list of Bused EKeys and how owns them.

Examples:

```
cmmget -l blade7 -d grantedboardekeys
```

```
cmmget -l cmm -d busedekeys
```

CDMs and FRU Information 15

15.1 Chassis Data Module

There are two chassis data modules (CDMs) in a chassis for redundancy.Each CDM has 2 EEPROMs containing the FRU information for the chassis. The CDM FRU data is also cached on the CMMs at boot time.

The CDM stores serial number and asset information about chassis and provides PICMG 3.0 shelf FRU information, such as number of slots, slot connection/routing information (for electronic keying), and max power per feeds.

15.2 FRU/CDM Election Process

When the CMMs first boot up, there is an CDM election process to determine the valid CDM contents. At least one valid set of CDM data must exist for the election to succeed. A CDM data set is considered valid if the Active CMM can read it and the checksum computes

During the initial bootup process, the CMM needs to elect which CDM's FRU information to use to retrieve critical chassis information. The election is a simple majority process:

1. Three matching sets of CDM data win the election.(3-1).
2. Two matching sets of CDM data win if the other two are split (2-1-1) or if one is missing (2-1-x)
3. If matched sets are split 2-2, CDM1 wins
4. If there are no matched data sets, then CDM priority is determined by first available, valid data. The priority is determined in the following order:
   - CDM1
   - CDM2
   - Local cache (/etc/cmm/sfrucache)
   - Remote cache (/usr/sfrucache)

The highest-ranking member of the winning "team" in a 3-1, 2-1-1, or 2-1-x result is elected the defining CDM authority

15.3 FRU Information

The CMM can query the entire FRU of a device, entire areas of a FRU, or individual fields in the different areas of the FRU.

For detailed information on FRU requirements, please refer to the PICMG 3.0 Specification.

15.4 FRU Query Syntax

The format for querying the FRU of a particular location is:

```
cmmget -l [location] -t fru -d [dataitem]
```

Where location is the component for which the FRU information is to be retrieved from, and dataitem is the field(s) of the FRU which will be retrieved.

Table 56 lists the various FRU data items and the information they retrieve.

Note: The chassis* data items in Table 56 are only available with the location “chassis”.

Note: When listing all FRU information for the location “chassis”, there is a location field listed consisting of “xxxxx.”, which is not changeable. The correct chassis location information is kept in the Shelf Address record. Use the location dataitem on the chassis location to get and set the chassis location field. For example, *cmmget -l chassis -d location*. Refer to Section 8, “The Command Line Interface (CLI)” on page 71 for more information.

Table 56. Dataitems Used With FRU Target (-t) to Obtain FRU Information

Dataitem	Description
all	Returns all FRU information for the location.
boardall	Lists all board area FRU information for the location.
boarddescription	Returns the description in the FRU board area for the location.
boardmanufacturer	Lists the manufacturer field in the FRU board area for the location.
boardpartnumber	Lists the part number field in the FRU board area for the location.
boardserialnumber	Lists the serial number field in the FRU board area for the location.
boardfrufileid	Gets the FRU information version for that device.
boardmanufacturedatetime	Lists the manufacture date and time field in the FRU board area for the location.
productall	Lists all product area FRU information for the location.
productdescription	Returns the description in the FRU product area for the location.
productmanufacturer	Lists the manufacturer field in the FRU product area for the location.
productmodel	Lists the model field in the FRU product area for the location.
productpartnumber	Lists the part number field in the FRU product area for the location.
productserialnumber	Lists the serial number field in the FRU product area for the location.
productrevision	Lists the revision field in the FRU product area for the location.
productmanufacturedatetime	Lists the manufacture date and time field in the FRU product area for the location.
chassisall	Lists all chassis area FRU information for the location.
chassispartnumber	Lists the part number field in the FRU chassis area for the location.
chassisserialnumber	Lists the serial number field in the FRU chassis area for the location.
chassistype	List the type field in the FRU chassis area for the location.
listdataitems	Displays a list of all FRU dataitems that can be queried for the FRU target.

Fan Control and Monitoring 16

The CMM controls the fan speeds and the fan tray LEDs. In a healthy state (no events), the LED is driven to green color. If any of the fan tray sensors (temperature, voltages, fan tachs) are in an unhealthy state, the LED is driven to red or amber (red by default).

16.1 Automatic Fan Control

The CMM will drive the fans to full speed (100 percent) under the following condition:

- An upper critical (major event) or upper non-recoverable (critical event) temperature threshold is violated in the platform, including blades.
- If both CMMs are removed from the chassis.

Note: If communication is lost between the CMM and the fantray, it takes approximately two minutes for the fantray to reset, which will drive the fanspeed to 100 percent.

16.2 Querying Fan Tray Sensors - FantrayN location

To query the fan tray and fan tray sensors, use *fantrayN* as the location (-l) of the cmmget command. For example, to query the current RPM value of Chassis Fan 1 in the fantray on the MPCHC0001 chassis, issue the command:

```
cmmget -l fantray1 -t “Chassis Fan 1” -d current
```

16.3 Fantray Cooling Levels

The fantray supports a range of ***cooling levels*** that it can operate at. By AdvancedTCA specification, when queried via IPMI, the fantray returns its maximum cooling level, minimum cooling level and a recommended cooling level (for normal operation). The AdvancedTCA specification explains that all fantrays must support all cooling levels between its minimum and maximum levels by increments of one unit.

The fantray can only run at one cooling level at a time. The cooling level that a fantray is running at is its ***current cooling level***.

The cooling levels do not represent RPMs because each cooling unit may not actually contain fans. The CMM Cooling Manager isn’t even aware of how the fantrays cool the shelf. It simply knows that to increase the cooling output of the fantray it should use a higher cooling level. Each fantray may (and most likely will) have different minimum, maximum and recommended normal cooling levels.

16.4 CMM Cooling Manager Temperature Status

When in the CMM Control Mode, the Cooling Manger is responsible for changing the fantray’s current cooling level upon temperature events. It does so by switching between four temperature statuses. At any moment the Cooling Manager can only operate at one temperature status, it is called the current temperature status. The four temperature statuses are normal, minor, major and critical. The Cooling Manager changes its current temperature status depending on what, if any, temperature events it has received. Here is how the Cooling Manager determines its current temperature status.

- Normal– There is currently no asserted temperature event.
- Minor– There is at least one asserted minor temperature event.
- Major— There is at least one asserted major temperature event.
- Critical— There is at least one asserted critical temperature event.

The user can read the temperature status with the following command:

```
cmmget –l cmm –d temperaturelevel
```

16.5 CMM Cooling Table

When the Cooling Manager's current temperature status changes it also changes the fantray's current cooling level. The Cooling Manager uses a mapping table called the ***cooling table*** to map each temperature status to a cooling level.

When the current temperature status changes the Cooling Manager retrieves the cooling level corresponding to that temperature status. It then changes the fantray's current cooling level to that value.

Two of the cooling levels in the cooling table can be configured by the user, however there are limitations.

Table 57. CMM Cooling Table

Temperature Status	Sample Cooling Level	User Definable?
Critcal	100	No
Major	100	No
Minor	76	Yes
Normal	72	Yes

This data structure is synchronized between the active and standby CMMs.

16.5.1 Setting Values in the Cooling Table

There are commands for assigning cooling levels to the normal and minor temperature statuses in the cooling table.

To assign a cooling level to the normal level of fantrayN, the following command is used:

```
cmmset –l fantrayN -d normallevel –v [cooling level]
```

To assign a cooling level to the minor level of fantrayN, the following command is used:

```
cmmset –l fantrayN -d minorlevel –v [cooling level]
```

N: The number of the fan tray being addressed.

These values will take effect immediately after they are entered. That means that if fantrayN's current temperature status is normal and the user sets the normallevel the current cooling level will change immediately.

There are the limits on the possible cooling level values for the temperature statuses. The major status and the critical status are both always set to the maximum cooling level and are not configurable. In addition the following rule is always enforced.

Minimum cooling level <= normallevel <= minorlevel <= majorlevel = criticallevel = maximum cooling level

16.6 Control Modes for Fan Trays

There are three modes of control that a fantray may operate at: CMM, FanTray or Emergency Shutdown. The fantray can only run at one control mode at a time. The control mode that the fantray is running at is its current control mode. The user has the ability to change the current control mode of each fantray in the shelf.

16.6.1 CMM Control Mode

The CMM Control Mode is the mode in which the CMM has complete control over the fantray's current cooling level. The user has the ability to configure how the Cooling Manager manages each fantray by changing the values in the cooling table. In the CMM Control Mode the cooling manager uses the cooling table to determine which cooling level to use for the current temperature status. The user will be able to change to this mode with the following command:

```
cmmset –l fantrayN -d control –v cmm
```

Where:

N: The number of the fan tray being addressed.

16.6.2 Fantray Control Mode

The ATCA Specification defines a mode called local control where the fantray determines its own cooling level. This control mode is optional. Not all fan trays will support local control.

To change to this local control mode, the control mode on the CMM is changed to “fantray control”.

While a fantray is in “fantray control” any changes the user makes to the cooling table will not take effect immediately. They will be saved and take effect if the fantray's current mode becomes the CMM Control Mode.

The user may change to this mode with using the following command:

```
cmmset –l fantrayN -d control –v fantray
```

Note: The control mode can only be fantray mode if there are no temperature events in the chassis.

16.6.3 Emergency Shutdown Control Mode

The Emergency Shutdown Control Mode causes the fantray to stop cooling the system. A fantray will stay in this mode until the user changes the current control mode to one of the other two modes. The user may change to this mode with the following command:

```
cmmset –l fantrayN -d control –v emergencyshutdown
```

Where:

N: The number of the fan tray being addressed.

16.6.4 User Initiated Mode Change

To change the control mode of a fantray, the user may use the command:

```
cmmset –l fantrayN -d control –v [ CMM | fantray | EmergencyShutdown |
defaultcontrol]
```

N:	The number of the fan tray being addressed.
CMM:	Sets the fan tray to CMM fan tray control mode.
fantray:	Sets the fan tray to local fan tray control mode.
EmergencyShutdown:	Sets the fan tray to Emergency Shutdown mode.
deafultcontrol:	Sets the fan tray to the default control mode.

If the user selects the defaultcontrol value the current control mode will be the default control mode.

16.6.5 Automatic Mode Change

There is only one scenario that will cause a fantray’s current control mode to change automatically. This is if the fantray is in the Fantray Control Mode and the Cooling Manager receives an asserted temperature event. In this case we do not want the fantray controlling itself. We want the Cooling Manager to immediately increase the fantray’s current cooling level. Therefore the Cooling Manager will switch the current cooling mode of all fantrays out of the fantray control mode and into the CMM control Mode. The fantrays will stay in the CMM Control Mode until the user specifies otherwise. If scenario occurs a SEL event will be logged and a SNMP Trap will be dispatched.

16.7 Getting Temperature Statuses

There are two commands for getting temperature statuses.

To retrieve the cooling level assigned to the normal status for fantrayN.

```
cmmget –l fantrayN -d normallevel
```

To retrieve the cooling level assigned to the minor status for fantrayN.

```
cmmget –l fantrayN -d minorlevel
```

N: The number of the fan tray being addressed.

The user is not able to set the majorlevel or the criticallevel and therefore he will not be able to retrieve these values either.

16.8 Fantray Properties

There are also commands to retrieve the fantray’s settings. These properties are maximum cooling level, minimum cooling level and recommended cooling level. The cooling manager gets the fan speed properties by performing an IPMI get fantray properties call on the fantray. This method is defined by the PICMG 3.0 specification. The user can use these values to determine how he configures the temperature statuses.

To return the minimum cooling level that the fantray supports:

```
Cmmget –l fantrayN -d minimumsetting
```

To return the maximum cooling level that the fantray supports:

```
Cmmget –l fantrayN -d maximumsetting
```

To return the fantray’s recommended cooling level.

```
Cmmget –l fantrayN -d recommendedsetting
```

N: The number of the fan tray being addressed.

Note: Values returned are in units. These are not necessarily percentages, which have been used in versions of firmware prior to 5.2. For further information please refer to the PICMG 3.0 Specification.

16.9 Retrieving the Current Cooling Level

There is a command to allow the user to get the current cooling level. This command is:

```
cmmget –l fantrayN –d currentfanlevel
```

N: The number of the fan tray being addressed.

This command will query fantrayN and return to the user it’s current cooling level. If the fantray is in fantray control mode the cooling level selected by the fantray will be returned. If the fantray is in emergencyshutdown mode “0” will be returned.

16.10 Fantray Insertion

When a new fantray is installed in the shelf the Cooling Manager on the active CMM will be notified that a fantray was inserted. When the Cooling manager on the active CMM is notified of this condition it needs to determine the fantray’s control mode and it needs to populate the cooling table.

This situation also occurs when the chassis is power-cycled. In this case the each fantray in the chassis will notify the active CMM’s Cooling Manager that it has been inserted. The Cooling Manager will then read that /etc/cmm/fantray.cfg file and determine appropriate values using the following conventions.

16.11 Default Cooling Values

The CMM determines the fan tray cooling values from the /etc/cmm/fantray.cfg file. The CMM uses the defaults from the file in the following order of precedence:

1. User Defaults
2. Vendor Defaults
3. Code Defaults

16.11.0.1 Setting User Defaults and Defaultcontrol through the CLI

During normal operation the user may use these commands to set the default values.

```
cmmset –l fantrayN -d defaultcontrol –v [ CMM | fantray ]
```

```
cmmset –l fantrayN -d normallevel –v [value to set the normal level]
```

```
cmmset –l fantrayN -d minorlevel –v [value to set the minor level]
```

N: The number of the fan tray being addressed.

The defaultcontrol dataitem sets the default control mode to be used for that specific fantray if it is ever removed and then re-inserted. The other two commands set the values for the temperature statuses in the cooling table and are also saved for user defaults.

16.11.1 Vendor Defaults

If user specified defaults are not configured the Cooling Manager attempts to use vendor specified defaults.

The vendor specified defaults are stored in the /etc/cmm/fantray.cfg file. They are not configurable by the user. These defaults do not take the fantray's location into account, they are the defaults for a particular type of fantray not a particular fantray.

With the vendor defaults all fantrays of type X will use these vendor defaults if the user hasn't set any user defaults.

The vendor MUST set both the normal and minor default levels. If both are not set the code defaults will be used.

- Manufacturer_ID.product_ID.minorlevel=VALUE
- Manufacturer_ID.product_ID.normallevel=VALUE
- Manufacturer_ID.product_ID.control=VALUE

The /etc/cmm/fantray.cfg file contains other information that the Cooling Manager uses and only the vendor setting should ever be edited. The vendor settings should only be edited by the vendor and the end user should never touch this file. The manufacturer_ID must be identical to the decimal representation of bytes 8, 9 and 10 of the "Get Device ID" command as defined by the IPMI specification. The product_ID must be identical to the decimal representation of bytes 11 and 12 of the "Get Device ID" command as defined by the IPMI specification.

The manufacturer_ID should be four digits and the product_ID should be three digits. (e.g. 0157.870.minorlevel = 88)

16.11.2 Structure of /etc/cmm/fantray.cfg

All default values will persist in the /etc/cmm/fantray.cfg file. These will be stored in key=value pairings. These are the formats of those entries in the /etc/cmm/fantray.cfg file.

Example 2. Sample /etc/cmm/fantray.cfg file

```
0157.870.minorlevel=76

0157.870.normallevel=72

0157.870.control=fantray
```

16.11.3 Code Defaults

The code defaults are determined using a simple algorithm and values gathered by issuing the Get Fan Speed Properties command to the fan tray. Using the returned Maximum Speed Level, Minimum Speed Level and Normal Operating Level the Cooling Manager makes the following assignments. The code defaults are determined as follows:

— Normallevel = Normal Operating Level

— Criticallevel, majorlevel = Maximum Speed Level

— Minorlevel = [(Maximum Speed Level – Normal Operating Level) / 2] + Minimum Speed Level

If there aren’t any user or vendor specified defaults configured then the Cooling Manager will put all fantrays into CMM Control Mode and it will use this algorithm to populate the cooling table.

16.11.4 Restoring Defaults

Aside from using defaults during fantray insertion defaults can be used by the user to reset the cooling table. The user can use the command.

```
cmmset –l fantrayN -d restoredefaults –v true
```

N: The number of the fan tray being addressed.

This command will use the vendor defaults if possible, or the code default if not, to reset the cooling table. In addition to restoring the cooling table, all saved user defaults will be cleared.

The same command will reset the control mode. If there is a specified vendor control mode it will be used, if not the CMM Control Mode will be used to set the current control mode. The user specified default control mode will also be cleared.

16.12 Firmware Upgrade/Downgrade

When a fan tray’s firmware is upgraded or downgraded the user who is changing the firmware should update the vendor defaults in the /etc/cmm/fantray.cfg file on both CMMs and then upgrade/downgrade the firmware. When the firmware is upgraded/downgraded the fan tray will go through its M-states and when it returns to the M4 state the Cooling Manager will re-read the /etc/cmm/fantray.cfg file and determine the values to be used in the cooling table. If the new firmware on the fan tray has a different minimum cooling level or a different maximum cooling level the Cooling Manager can no longer trust the user defaults in the /etc/cmm/fantray.cfg file.

The Cooling Manager keeps track of a fan tray’s properties in the /etc/cmm/fantray.cfg file so that it can determine when a firmware upgrade/downgrade would affect the user defaults. When a fan tray is inserted into the chassis the Cooling Manager checks the /etc/cmm/fantray.cfg to see if there are recorded minimum and maximum values for that fan tray at that location. It will compare the recorded minimum and maximum to the minimum and maximum values that the inserted fan tray returns. If they differ from the Cooling Manager it removes all of the user defaults from the /etc/cmm/fantray.cfg file.

16.13 Chassis vs. Fantray

In versions of firmware prior to 5.2, all cooling operations are performed on the chassis location (-l). If there is more than one fantray in the shelf the Cooling Manager will apply all settings to every fantray. Also, if the user gets the current fan speed he is returned the average fan speed of all of the fantrays in the shelf.

In firmware version 5.2, all cooling operations will be performed on specific fantray locations (i.e., cmmget -l fantray1 -d listdataitems). With the exception of the deprecated operations from 5.1 there will be no fantray operations performed on the chassis location.

16.14 Legacy Method of Querying/Setting Fan Speed

The following section describes the legacy method of fan control, which was the only method supported in versions of firmware prior to 5.2. The following method has been left in for legacy support only, and is not a recommended method for doing fan speed control. This method will be removed from the next generation of firmware.

Note: While the fanspeed setting in the CLI supports any integer between 0-100, the PICMG Set Fan Level command for the MPCHC0001 chassis only supports integers between 20 and 100. This means 0 in the CLI fanspeed dataitem corresponds to 20 on the PICMG command, 50 to 60, and 100 to 100.

By default, the CMM sets the fan speed to a value based on the Recommended Normal Operating Level (Maximum Sustained Speed Level)–a value returned by the fan tray in response to the PICMG Get Fan Speed Properties command. For example, in the MPCHC0001 chassis, the Maximum Sustained Speed Level is 70, which translates to 62 percent in the CLI.

The user can set the fan speed to a value between 0 and 100 percent of the full speed through any of the supported interfaces. Setting the fan speed from the CMM changes the fan speeds for all of the fans in the chassis. Setting the fan speed to 0 sets the fans to PICMG fan level of 20.

The following command can be used to set the fan speed for all fans:

```
cmmset -l chassis -d fanspeed -v [0-100]
```

emergencyshutdown and localcontrol are additional settings that can be used with the fanspeed dataitem. emergencyshutdown will shut down the fans and localcontrol will restore local control of fanspeed to those fan trays that support local control.

Note: localcontrol is not supported on the MPCHC0001 chassis fan tray.

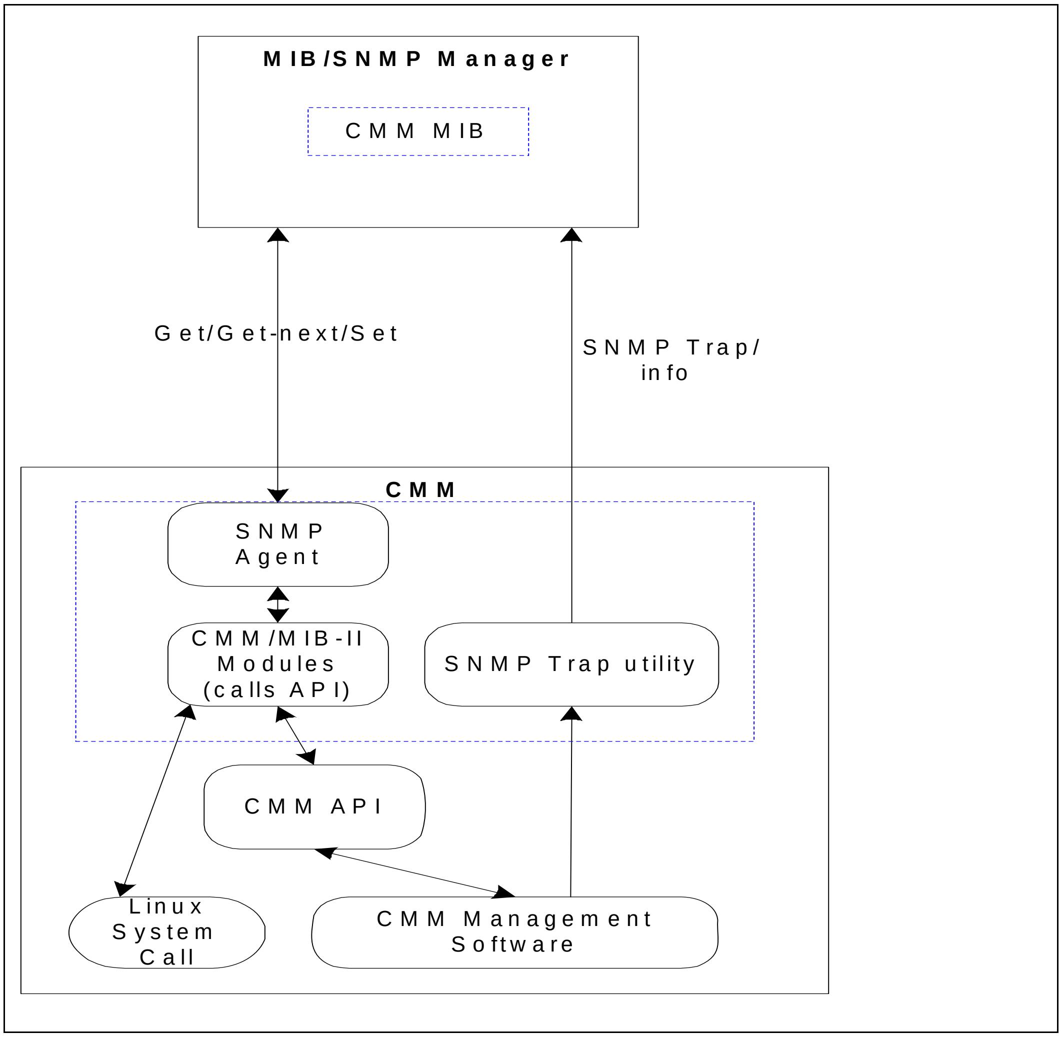
SNMP 17

The CMM supports v1 and v3 of the Simple Network Management Protocol (SNMP). The CMM can support SNMP queries and send SNMP traps in either v1 or v3. The SNMP interface on the CMM very closely mirrors that of the CLI in both syntax and function.

Note: Like the CLI, SNMP commands should be issued to the active CMM. The standby CMM will only respond to commands for using the CMM location parameter.

The following diagram shows the high level layout of the SNMP/MIB components and the flow of the data via SNMP protocol.

Figure 3. High Level SNMP/MIB Layout



17.1 CMM MIB

The CMM comes with a text file (mpcmm0001.mib) that describes the CMM and platform objects to be managed. A remote application such as an SNMP/MIB manager can compile and read this file and utilize this information to manage the sensor devices on CMM, chassis, and server blades currently present. The mpcmm0001.mib is located in the */usr/local/cmm* directory in the CMM. Users can utilize FTP or RCP to get this file from the CMM.

17.2 MIB Design

The CMM supports the following MIB II objects. The writeable objects can be set in their respective fields in the /etc/snmpd.conf file. Only the objects described in this section can be customized for the CMM. Other unlisted MIB II objects will work.

Table 58. MIB II Objects - System Group

Object	Syntax	Access	Value
sysDscr	DisplayString	read-only	“Linux mpcmm0001 2.4.2-1 #156 [DATE] armv5” Date is the actual current date.
sysObjectID	OBJECT IDENTIFIER	read-only	iso(1).org(3).dod(6).internet(1).private(4).enterprises(1).intel(343).products(2).Server-Management(10).Chassis-Management(3).mpcmm0001(2)
sysContact	DisplayString	read-write	max. 128 bytes
sysName	DisplayString	read-write	Default:”mpcmm0001”
sysLocation	DisplayString	read-only	max. 128bytes

Table 59. MIB II - Interface Group

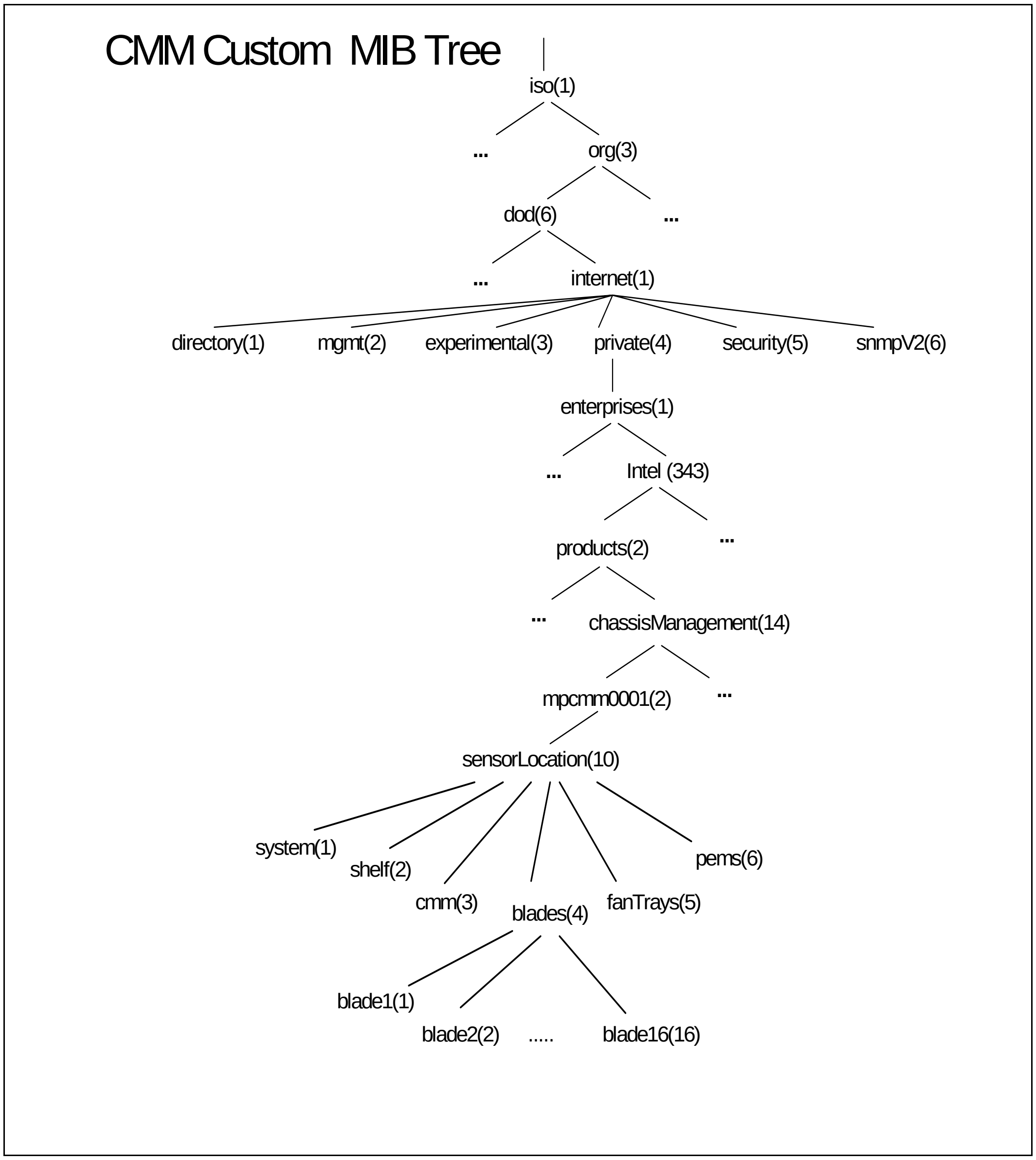
Object	Syntax	Access	Value
ifDscr	DisplayString	read-only	10/100BASE-TX

17.2.1 MIB Tree

The MIB tree representing the locations of the custom MIB is shown in Figure 4, “CMM Custom MIB Tree” on page 142.

Next to each object is the object identifier number in parentheses () for each object in the MIB tree. For example, using the tree below it can be calculated that the full OID for blade16 location on the CMM is 1.3.6.1.4.1.343.2.14.2.10.4.16 by starting at the top of the tree and following the numbers down to blade16.

Figure 4. CMM Custom MIB Tree



17.2.2 CMM MIB Objects

All the objects below can be found in the mpcmm0001.mib file. Please refer to that file for the latest updated list of supported MIB objects. It is intended to mirror the CLI cmmget and cmmset command dataitems as much as possible for each location.

Table 60. System Location (1.3.6.1.4.1.343.2.14.2.10.1)

OID	Object	Syntax	Access	Value
1	systemHealth	INTEGER	read-only	Health information about a particular location. This will be one of the following: (0-OK, 1-Minor, 2-Major, 3-Critical)
2	systemUnhealthyLocation	DisplayString	read-only	Used with the System location to find out what blades are having problems.
3	systemListDataItems	DisplayString	read-only	Used to find out what data items are available on a Location.
4	systemClearMajor	INTEGER	write-only	Setting this object will clear the Major Events LED on the active CMM (1-set)
5	systemClearMinor	INTEGER	write-only	Setting this object will clear the Minor Events LED on the active CMM (1-set)

Table 61. Shelf Location (Equivalent to Chassis) (1.3.6.1.4.1.343.2.14.2.10.2)

OID	Object	Syntax	Access	Value
1	shelfListTargets	DisplayString	read-only	Used to find out what targets are available on a location. (e.g., Getting the list of sensors in the chassis).
2	shelfListDataItems	DisplayString	read-only	Used to find out what data items are available on a target or location.
3	shelfFanSpeed	DisplayString	read-write	Used to get or set the fan speed in the chassis. Percent of the maximum fan speed that the chassis supports. Values can be multiples of 10 from 20 to 100, or 0xFE for emergency shutdown.
4	shelfLocation	DisplayString	read-write	This is used to get or set the Location field in the chassis FRU and is sent out as a part of SNMP and UDP alerts. This is only used with the chassis location.
5	shelfHealth	INTEGER	read-only	Health information about a particular Location. This will be one of the following: (0-OK, 1-Minor, 2-Major, 3-Critical).
6	shelfHealthEvents	DisplayString	read-only	Events that contribute to the Health of the Location. This is a list of events happening on the Location.
7	shelfIPMICommandRequest	DisplayString	read-write	Used to send any IPMI command to this location.
8	shelfIPMICommandResponse	DisplayString	read-only	Used to read the response to the above IPMI command.

Table 62. ShelfTable/shelfEntry (1.3.6.1.4.1.343.2.14.2.10.2.50.1) (Sheet 1 of 2)

OID	Object	Syntax	Access	Value
1	shelfTarget	OCTET STRING	none	index
23	shelfMinorAction	DisplayString	write-only	Used to configure user defined actions when minor events occur.
24	shelfMajorAction	DisplayString	write-only	Used to configure user defined actions when major events occur.
25	shelfCriticalAction	OCTET STRING	write-only	Used to configure user defined actions when critical events occur.
26	shelfNormalAction	OCTET STRING	write-only	Used to configure user defined actions when normal events occur.
27	shelffruBoardDescription	DisplayString	read-only	Used to read the Board Description from the FRU.
28	shelffruBoardManufacturer	DisplayString	read-only	Used to read the Board Manufacturer from the FRU.
29	shelffruBoardPartNumber	DisplayString	read-only	Used to read the Board Part Number from the FRU.
30	shelffruBoardSerialNumber	DisplayString	read-only	Used to read the Board Serial Number from the FRU.

Table 62. ShelfTable/shelfEntry (1.3.6.1.4.1.343.2.14.2.10.2.50.1) (Sheet 2 of 2)

OID	Object	Syntax	Access	Value
31	shelffruBoardManufactureDate	DisplayString	read-only	Used to read the Board Manufacture Date from the FRU.
32	shelffruProductDescription	DisplayString	read-only	Used to read the Product Description from the FRU.
33	shelffruProductManufacturer	DisplayString	read-only	Used to read the Product Manufacturer from the FRU.
34	shelffruProductPartNumber	DisplayString	read-only	Used to read the Product Part Number from the FRU.
35	shelffruProductSerialNumber	DisplayString	read-only	Used to read the Product Serial Number from the FRU.
36	shelffruProductManufactureDate	DisplayString	read-only	Used to read the Product Manufacture Date from the FRU.
37	shelffruProductModel	DisplayString	read-only	Used to read the Product Model from the FRU.
38	shelffruProductRevision	DisplayString	read-only	Used to read the Product Revision from the FRU.
39	shelffrushelfPartNumber	DisplayString	read-only	Used to read the shelf Part Number from the FRU.
40	shelffrushelfSerialNumber	DisplayString	read-only	Used to read the shelf Serial Number from the FRU.
42	shelffrushelfType	DisplayString	read-only	Used to read the shelf Type from the FRU.

Table 63. Cmm Location (1.3.6.1.4.1.343.2.14.2.10.3) (Sheet 1 of 3)

OID	Object	Syntax	Access	Value
1	softwareVersion	DisplayString	read-only	The version of the CMM software.
2	cmmRedundancy	DisplayString	read-only	cmm redundancy information.
3	rmcpEnabled	INTEGER	read-write	Enable or disable the RMCP Service: disable(0), enable(1)
4	cmmGrantedBoardEkeys	DisplayString	read-only	Get the Ekeys that have been granted to the location.
5	cmmTotalFrus	DisplayString	read-only	display the total number of FRUs.
6	cmmPICMGProperties	DisplayString	read-only	
7	cmmDeviceID	DisplayString	read-only	
8	cmmPresence	INTEGER	read-only	Used to find out if a location is present in the chassis.1 - Present 0 - Not present
9	cmmListTargets	DisplayString	read-only	Used to find out what targets are available on a location (e.g., getting the list of sensors in the chassis).
10	cmmListDataItems	DisplayString	read-only	Used to find out what data items are available on a target or location.
11	cmmHealth	INTEGER	read-only	Health information about a particular location. This will be one of the following: (0-OK, 1-Minor, 2-Major, 3-Critical)
12	cmmHealthEvents	DisplayString	read-only	Events that contribute to the health of the location. This is a list of events happening on the location.
13	cmmSNMPEnable	INTEGER	read-write	Used to set or query SNMP trap enabled status.
14 - 18	CmmSNMPTrapAddressN	IP Address	read-write	The IP address of the machine that will receive SNMP traps from a location.
19	cmmSNMPTrapCommunity	DisplayString	read-write	SNMP community name (max 64 characters).
20	cmmSNMPTrapVersion	DisplayString	read-write	v1 or v3
21	cmmSNMPTrapPort	DisplayString	read-write	SNMP trap port.
22	cmmFaultLEDColor	DisplayString	read-write	Get/Set the color of the fault/health LED on the CMM fronted FRUsto be used when an error is reported. Values: “amber” or “red”.
23	cmmAlarmCutoff	INTEGER	read-write	The Telco Alarm cutoff. When enabled, it silences the Telco alarm for active events and blinks the event LEDs on the CMM. This dataitem is only valid when used with the cmm as the location and is used to set the alarm cutoff or get its value. 1 - Set, 0 - Clear.

Table 63. Cmm Location (1.3.6.1.4.1.343.2.14.2.10.3) (Sheet 2 of 3)

OID	Object	Syntax	Access	Value
24	cmmAlarmTimeOut	INTEGER	read-write	The timeout value in minutes for the Telco Alarm cutoff. This is the amount of time before the alarm cutoff will automatically become unset if the user doesn't unset it themselves. A value of 0 will disable the timeout. This dataitem is only valid when used with the cmm as the location and is used to set the alarm timeout or get its value.
25	cmmCriticalLED	INTEGER	read-write	The LED on the CMM faceplate that indicates presence of critical events in the system. 1 - Set, 0 - Clear
26	cmmMajorLED	INTEGER	read-write	The LED on the CMM faceplate that indicates presence of major events in the system. 1 - Set, 0 - Clear
27	cmmMinorLED	INTEGER	read-write	The LED on the CMM faceplate that indicates presence of minor events in the system.1 - Set, 0 - Clear
28	cmmFailover	INTEGER	write	Used to induce a failover from the active cmm to the standby cmm. 1 - Enable, 2 - any
29 - 34	cmmsEthernetA cmmsEthernetB cmm1EthernetA cmm1EthernetB cmm2EthernetA cmm2EthernetB	INTEGER	read-write	When get, it displays the Eth0 or Eth1 connectors of both CMMs, CMM1, or CMM2. When set, it routes the selected Connection between the front or rear connectors. Valid Values: front / rear
35	cmmListLocations	DisplayString	read-only	list of Locations information
36	cmmListPresent	DisplayString	read-only	list of Present Entities information
37	cmmPowerBudget	DisplayString	read-only	
38	cmmBusedEkeys	DisplayString	read-only	
39	cmmClearSel	INTEGER	write	To clear the SEL set the object with 1 to clear. Other value - has no effect
40	cmmPowerSequence	DisplayString	read-write	List of boards/delay pairs indicating the power sequence of the boards in the Chassis
41	cmmPowerState	DisplayString	read-write	This is used to find out the current power state of the CMM. Format text - Location: CurrentState (Mx) where Mx is the ATCA-defined M state of the location. Settable value:reset, poweron, poweroff
42	cmmFeedCount	DisplayString	read-only	Get the power feed count
43	cmmFRUExtractionNotify	INTEGER	write	This is an Intel extension command for PICMG 3.0. Used to notify the Shelf Manager that a FRU has been extracted from the shelf. Valid value: 1

Table 63. Cmm Location (1.3.6.1.4.1.343.2.14.2.10.3) (Sheet 3 of 3)

OID	Object	Syntax	Access	Value
44	cmmUpdate	DisplayString	write	Perform the CMM firmware update. The set value is in the following format: <cmm image location> ftp:<hostname or IP address>:username:password.
45	cmmResetAirFilterRuntime	INTEGER	write	This allow the user to set the runtime to zero when the filter is replaced. 1 - reset, other value no change.
46	cmmAirFilterRunTimeLimit	DisplayString	read-write	Returns the upper critical limit. Refer to CLI section for more info.
47	cmmSyncuserledstate	DisplayString	read-write	Get/Set whether the LED state is synced between the active and the standby CMM. “Yes” or “No”.
48	cmmLoginMessage	DisplayString	read-write	Used to customize the login screen message by allowing the user to add the OEM name.
49	cmmCmdLinePrompt	DisplayString	read-write	Used to customize the bash prompt by allowing the user to add the OEM name.
50	standbyCMMReboot	INTEGER	read-write	Used to reboot the standby CMM. 1 - Reboot
55	cmmIPMICommandRequest	DisplayString	read-write	Used to send any IPMI command to this location.
56	cmmIPMICommandResponse	DisplayString	read-only	Used to read the response to the above IPMI command.
57	cmmFailoverOnRedundancy	DisplayString	read-write	Get/Set of the failover on redundancy flag.Settable values are: 'automatic' or 'manual'
58	cmmSyncUserScripts	DisplayString	read-write	Get/Set the User Scripts Synchronization flag.Settable values are: 'equal' or 'upgrade' or 'downgrade' or 'always'"
59	cmmTemperatureLevel	DisplayString	read-only	Used to get the temperature level of the system.
62	cmmSNMPTrapFormat	INTEGER	read-write	Get/Set SNMP trap format. 1 - text, 2 - raw, 3 - text + raw
63	cmmSNMPSendUnrecognizedEvents	INTEGER	read-write	Get/Set SNMPSendUnrecognizedEvents. 0 - disable, 1 - enable
64	cmmSELFormat	INTEGER	read-write	Get/Set SEL format. 1 - text, 2 - raw, 3 - text + raw
65	cmmSELDisplayUnrecognizedEvents	INTEGER	read-write	Get/Set SELDisplayUnrecognizedEvents. 0 - disable, 1 - enable

Table 64. CmmTable/cmmEntry (1.3.6.1.4.1.343.2.14.2.10.3.51.1) (Sheet 1 of 2)

OID	Object	Syntax	Access	Value
1	cmmTarget	DisplayString	none	index
2	cmmCurrent	DisplayString	read-only	The current value of a sensor.
3	cmmThresholdsAll	DisplayString	read-only	All thresholds of a sensor.
4	cmmUpperCritical	DisplayString	read-only	Upper critical thresholds
5	cmmUpperNonCritical	DisplayString	read-only	upper non-critical thresholds
6	cmmLowerNonCritical	DisplayString	read-only	lower non-critical thresholds
7	cmmLowerCritical	DisplayString	read-only	lower critical thresholds
8	cmmUpperNonRecoverable	DisplayString	read-only	upper non recoverable thresholds
9	cmmLowerNonRecoverable	DisplayString	read-only	lower non recoverable thresholds
10	cmmMaximumExternalAvailableCurrent	DisplayString	read-write	Each unit is 100mA
11	cmmMaximumInternalAvailableCurrent	DisplayString	read-only	Each unit is 100mA
12	cmmMinimumExpectedOperatingVoltage	DisplayString	read-write	Each unit is 0.5 Volt
13	cmmSensorHealth	INTEGER	read-only	Health information about a particular location/sensor. This will be one of the following: (0-OK, 1-Minor, 2-Major, 3-Critical)
14	cmmSensorHealthEvents	DisplayString	read-only	Events that contribute to the health of the location/sensor. This is a list of events happening on the location.
15	cmmMinorAction	OCTET STRING	read-write	Used to configure user defined actions when minor events occur.
16	cmmMajorAction	OCTET STRING	read-write	Used to configure user defined actions when major events occur.
17	cmmCriticalAction	OCTET STRING	read-write	Used to configure user defined actions when critical events occur.
18	cmmNormalAction	OCTET STRING	read-write	Used to configure user defined actions when normal events occur.
35	cmmfruBoardDescription	DisplayString	read-only	Used to read the Board Description from the FRU.
36	cmmfruBoardManufacturer	DisplayString	read-only	Used to read the Board Manufacturer from the FRU.
37	cmmfruBoardPartNumber	DisplayString	read-only	Used to read the Board Part Number from the FRU.
38	cmmfruBoardSerialNumber	DisplayString	read-only	Used to read the Board Serial Number from the FRU.
39	cmmfruBoardManufactureDate	DisplayString	read-only	Used to read the Board Manufacture Date from the FRU.
40	cmmfruProductDescription	DisplayString	read-only	Used to read the Product Description from the FRU.
41	cmmfruProductManufacturer	DisplayString	read-only	Used to read the Product Manufacturer from the FRU.
42	cmmfruProductPartNumber	DisplayString	read-only	Used to read the Product Part Number from the FRU.

Table 64. CmmTable/cmmEntry (1.3.6.1.4.1.343.2.14.2.10.3.51.1) (Sheet 2 of 2)

OID	Object	Syntax	Access	Value
43	cmmfruProductSerialNumber	DisplayString	read-only	Used to read the Product Serial Number from the FRU.
44	cmmfruProductManufactureDate	DisplayString	read-only	Used to read the Product Manufacture Date from the FRU.
45	cmmfruProductModel	DisplayString	read-only	Used to read the Product Model from the FRU.
46	cmmfruProductRevision	DisplayString	read-only	Used to read the Product Revision from the FRU.
47	cmmEventAction	DisplayString	read-write	Used to query or associate a script to a particular event action cod.

Table 65. CmmFruTable/cmmFruEntry (1.3.6.1.4.1.343.2.14.2.10.3.52.1)

OID	Object	Syntax	Access	Value
1	cmmFruNumber	INTEGER	none	index
3	cmmFruHotSwapState	DisplayString	read-only	Retrieve the hot swap state of the FRU from the CMM’s internally known hot swap states. Returns M state number (0-7)
4	cmmFruLedProperties	DisplayString	read-only	Find out what LEDs the FRU supports.
5	cmmFruPowerLevels	DisplayString	read-only	Returns the power levels available for a FRU and number of watts drawn by each.
6	cmmFruPresentPowerLevel	DisplayString	read-write	Get/Set the power level of an entity.
7	cmmFRUActivation	INTEGER	read-write	Get/Set the activation state to either activate or deactivate the FRU. The Deactivate is the same as a Graceful Shutdown
8	cmmFRUActivationPolicy	INTEGER	read-write	Get/Set the locked bit of the activation policy
9	cmmFRUControl	INTEGER	write-only	"0" – Cold reset "1" – Warm reset "2" – Graceful reboot "3" – Issue diag. Interrupt
10	cmmFRUDeactivationPolicy	INTEGER	read-only	Get /Set the locked bit of the deactivation policy.

Table 66. CmmFruTargetTable (1.3.6.1.4.1.343.2.14.2.10.3.53.1)

OID	Object	Syntax	Access	Value
1	cmmFruNumber	INTEGER	none	index 1
2	cmmFruTarget	DisplayString	none	index 2
3	cmmFruLedColorProps	DisplayString	read-only	Finds out what colors are supported by the LED.
4	cmmFruLedState	DisplayString	read-write	Get/Set the FRU LED state.

Table 67. CmmPmsTable/cmmPmsEntry (1.3.6.1.4.1.343.2.14.2.10.3.54.1) (Sheet 1 of 2)

OID	Object	Syntax	Access	Value
1	cmmPmsTarget	INTEGER	none	index
2	cmmPmsAdminState	DisplayString	read-write	The process administrative state defines if monitoring of the process will be provided. Possible values: 1-Unlocked, 2-Locked.
3	cmmPmsRecoveryAction	DisplayString	read-write	The process recovery action defines how a faulty process will be recovered. Possible values: 1-No Action, 2-Process Restart, 3-Failover and Process Restart, 4 - Failover and Reboot.

Table 67. CmmPmsTable/cmmPmsEntry (1.3.6.1.4.1.343.2.14.2.10.3.54.1) (Sheet 2 of 2)

OID	Object	Syntax	Access	Value
4	cmmPmsEscalationAction	DisplayString	read-write	The process escalation action defines how a faulty process that fails to restart will be recovered. Possible values: 1-No Action, 2-Failover and Reboot.
5	cmmPmsProcessName	DisplayString	read-only	Display the monitored process' name and its associated command line arguments.
6	cmmPmsOpState	DisplayString	read-only	Display the operational state. An operational state of disabled indicates that the process is faulty but cannot be recovered. Possible values: 1-Enabled, 2-Disabled.
7	cmmPmsHealth	INTEGER	read-only	Health information about a particular monitored process. Possible values: 0-OK, 1-Minor, 2-Major, 3-Critical.
8	cmmPmsHealthEvents	DisplayString	read-only	Display the monitored process' health events.

Table 68. Blade# Location (1.3.6.1.4.1.343.2.14.2.10.4.[1-16])

OID	Object	Syntax	Access	Value
1	b#GrantedBoardEKeys	DisplayString	read-only	E keys
2	b#FRUExtractionNotify	INTEGER	write	Used to notify that a FRU has been extracted from the shelf. Value = “1”
3	b#PICMGProperties	DisplayString	read-only	The version Information.
4	b#DeviceID	DisplayString	read-only	Device ID
6	b#Powerstate	INTEGER	read-write	The power state of a blade. This is used to find out if a blade is present, healthy, on/off or to reboot, shutdown and turn on a blade. reset - Reset a blade poweron - Power on a blade poweroff - Power off a blade offline - Set the blade in the offline mode activate - set the board in active mode.
7	b#Presence	INTEGER	read-only	Used to find out if a location is present in the chassis. 0 - Not present, 1 - Present.
8	b#ListTargets	DisplayString	read-only	Used to find out what targets are available on a location (e.g., Getting the list of sensors in the chassis).
9	b#ListDataItems	DisplayString	read-only	Used to find out what data items are available on a target or location.
10	b#Health	INTEGER	read-only	Health information about a particular location. This will be one of the following: (0-OK, 1-Minor, 2-Major, 3-Critical).
11	b#HealthEvents	DisplayString	read-only	Events that contribute to the health of the location. This is a list of events happening on the location.

Table 68. Blade# Location (1.3.6.1.4.1.343.2.14.2.10.4.[1-16])

OID	Object	Syntax	Access	Value
12	b#TotalFrus	DisplayString	read-only	Displays the total number of FRUs
13	b#IPMICommandRequest	DisplayString	read-write	Used to send any IPMI command to this location.
14	b#IPMICommandResponse	DisplayString	read-write	Used to read the response to the above IPMI command.

Table 69. Blade#TargetTable/blade#TargetEntry (1.3.6.1.4.1.343.2.14.2.10.4.[1-16].51.1) (Sheet 1 of 2)

OID	Object	Syntax	Access	Value
1	b#Target	DisplayString	none	index
2	b#Current	DisplayString	read-only	The current value of a sensor.
3	b#ThresholdsAll	DisplayString	read-only	All thresholds of a sensor.
4	b#UpperCritical	DisplayString	read-only	Upper critical thresholds
5	b#UpperNonCritical	DisplayString	read-only	Upper non-critical thresholds
6	b#LowerNonCritical	DisplayString	read-only	Lower non-critical thresholds
7	b#LowerCritical	DisplayString	read-only	Lower critical thresholds
8	b#UpperNonRecoverable	DisplayString	read-only	Upper non recoverable thresholds
9	b#LowerNonRecoverable	DisplayString	read-only	Lower non recoverable thresholds
14	b#SensorHealth	INTEGER	read-only	Health information about a particular location/sensor. This will be one of the following: (0-OK, 1-Minor, 2-Major, 3-Critical).
15	b#SensorHealthEvents	DisplayString	read-only	Events that contribute to the health of the location/sensor. This is a list of events happening on the location.
16	b#MinorAction	OCTET STRING	read-write	Used to configure user defined actions when minor events occur.
17	b#MajorAction	OCTET STRING	read-write	Used to configure user defined actions when major events occur.
18	b#CriticalAction	OCTET STRING	read-write	Used to configure user defined actions when critical events occur.
19	b#NormalAction	OCTET STRING	read-write	Used to configure user defined actions when normal events occur.
20	b#fruBoardDescription	DisplayString	read-only	Used to read the Board Description from the FRU.
21	b#fruBoardManufacturer	DisplayString	read-only	Used to read the Board Manufacturer from the FRU.
22	b#fruBoardPartNumber	DisplayString	read-only	Used to read the Board Part Number from the FRU.
23	b#fruBoardSerialNumber	DisplayString	read-only	Used to read the Board Serial Number from the FRU.
24	b#fruBoardManufactureDate	DisplayString	read-only	Used to read the Board Manufacture Date from the FRU.

Table 69. Blade#TargetTable/blade#TargetEntry (1.3.6.1.4.1.343.2.14.2.10.4.[1-16].51.1) (Sheet 2 of 2)

OID	Object	Syntax	Access	Value
25	b#fruProductDescription	DisplayString	read-only	Used to read the Product Description from the FRU.
26	b#fruProductManufacturer	DisplayString	read-only	Used to read the Product Manufacturer from the FRU.
27	b#fruProductPartNumber	DisplayString	read-only	Used to read the Product Part Number from the FRU.
28	b#fruProductSerialNumber	DisplayString	read-only	Used to read the Product Serial Number from the FRU.
29	b#fruProductManufactureDate	DisplayString	read-only	Used to read the Product Manufacture Date from the FRU.
30	b#fruProductModel	DisplayString	read-only	Used to read the Product Model from the FRU.
31	b#fruProductRevision	DisplayString	read-only	Used to read the Product Revision from the FRU.
32	b#EventAction	OCTET STRING	read-write	

Table 70. Blade#FruTable/blade#FruEntry (1.3.6.1.4.1.343.2.14.2.10.4.[1-16].52.1)

OID	Object	Syntax	Access	Value
1	b#FruNumber	INTEGER	none	Index
3	b#FruHotSwapState	DisplayString	read-write	The hot swap state as read from the FRU and the CMM's internally known hot swap state.
4	b#FruLedProperties	DisplayString	read-only	Find out what LEDs the FRU supports.
5	b#FruPowerLevels	DisplayString	read-only	Returns the power levels available for a FRU and the number of watts drawn by each.
6	b#FruPresentPowerLevel	DisplayString	read-write	Get/Set the power level of an entity.
7	b#FRUActivation	INTEGER	read-write	Get/Set the activation state to either activate or deactivate the FRU. The Deactivate is the same as a Graceful Shutdown
8	b#FRUActivationPolicy	INTEGER	read-write	Get/Set the locked bit of the activation policy
9	b#FRUControl	INTEGER	write-only	Set the FRU to do things like Cold Reset, Warm Reset, etc. See AdvancedTCA for possibilities.
10	b#FRUDeactivationPolicy	INTEGER	read-write	

Table 71. Blade#FruTargetTable/blade#FruTargetEntry (1.3.6.1.4.1.343.2.14.2.10.4.[1-16].53.1)

OID	Object	Syntax	Access	Value
1	b#FruNumber	INTEGER	none	Index 1
2	b#Target	DisplayString	none	Index 2
3	b#FruLedColorProps	DisplayString	read-only	Finds out what colors are supported by the LED.
4	b#FruLedState	DisplayString	read-write	Get/Set the FRU LED state.

Table 72. [FanTray/pem]Table/[fanTray/pem]Entry (1.3.6.1.4.1.343.2.14.2.10.[5/6].51.1)

OID	Object	Syntax	Access	Value
1	[fanTray/pem]Number	INTEGER	none	index
2	[fanTray/pem]FRUExtractionNotify	INTEGER	write	Used to notify that a FRU has been extracted from the shelf. Valid value: 1
3	[fantray/pem]TotalFrus	DisplayString	read-only	Display the total number of FRUs.
4	[fanTray/pem]PICMGProperties	DisplayString	read-only	The version Information.
5	[fanTray/pem]DeviceID	DisplayString	read-only	
6	[fanTray/pem]Presence	INTEGER	read-only	Used to find out if a location is present in the chassis. 0 - Not present, 1 - Present
7	[fanTray/pem]ListTargets	DisplayString	read-only	Used to find out what targets are available on a location (e.g., getting the list of sensors in the chassis).
8	[fanTray/pem]ListDataItems	DisplayString	read-only	Used to find out what data items are available on a target or location.
9	[fanTray/pem]Health	INTEGER	read-only	Health information about a particular location. This will be one of the following: (0-OK, 1-Minor, 2-Major, 3-Critical).
10	[fanTray/pem]HealthEvents	DisplayString	read-only	Events that contribute to the health of the location. This is a list of events happening on the location.
11	[fanTray/pem]IPMICommandRequest	DisplayString	read-write	Used to send any IPMI command to this location.
12	[fanTray/pem]IPMICommandResponse	DisplayString	read-only	Used to read the response to the above IPMI command.
13	fanTrayMinorLevel	DisplayString	read-write	The cooling to be used when at the minor level.
14	fanTrayNormalLevel	DisplayString	read-write	The cooling to be used when at the minor level.
15	fanTrayControl	DisplayString	read-write	The fantray's control mode.
16	fanTrayDefaultControl	DisplayString	read-write	The fantray's default control mode.
17	fanTrayRestoreDefaults	DisplayString	write-only	Restore the fantray's control mode to the default setting.
18	fanTrayMinimumSetting	DisplayString	read-only	The fantray's minimum setting.

Table 72. [FanTray/pem]Table/[fanTray/pem]Entry (1.3.6.1.4.1.343.2.14.2.10.[5/6].51.1)

OID	Object	Syntax	Access	Value
19	fanTrayMaximumSetting	DisplayString	read-only	The fantray's maximum setting.
20	fanTrayRecommendedSetting	DisplayString	read-only	The fantray's recommended setting.
21	fanTrayCurrentFanLevel	DisplayString	read-only	The fantray's current fan level.

Table 73. [FanTray/pem]TargetTable/[fanTray/pem]TargetEntry (1.3.6.1.4.1.343.2.14.2.10.[5/6].52.1) (Sheet 1 of 2)

OID	Object	Syntax	Access	Value
1	[fanTray/pem]Number	INTEGER	none	Index 1
2	[fanTray/pem]Target	DisplayString	none	Index 2
3	[fanTray/pem]Current	DisplayString	read-only	The current value of a sensor.
4	[fanTray/pem]ThresholdsAll	DisplayString	read-only	All thresholds of a sensor.
5	[fanTray/pem]UpperCritical	DisplayString	read-only	Upper critical thresholds
6	[fanTray/pem]UpperNonCritical	DisplayString	read-only	Upper non-critical thresholds
7	[fanTray/pem]LowerNonCritical	DisplayString	read-only	Lower non-critical thresholds
8	[fanTray/pem]LowerCritical	DisplayString	read-only	Lower critical thresholds
9	[fanTray/pem]UpperNonRecoverable	DisplayString	read-only	Upper non recoverable thresholds
10	[fanTray/pem]LowerNonRecoverable	DisplayString	read-only	Lower non recoverable thresholds
14	[fanTray/pem]SensorHealth	INTEGER	read-only	Health information about a particular location/sensor. This will be one of the following: (0-OK, 1-Minor, 2-Major, 3-Critical).
15	[fanTray/pem]SensorHealthEvents	DisplayString	read-only	Events that contribute to the health of the location/sensor. This is a list of events happening on the location.
16	[fanTray/pem]MinorAction	OCTET STRING	read-write	Used to configure user defined actions when minor events occur.
17	[fanTray/pem]MajorAction	OCTET STRING	read-write	Used to configure user defined actions when major events occur.
18	[fanTray/pem]CriticalAction	OCTET STRING	read-write	Used to configure user defined actions when critical events occur.
19	[fanTray/pem]NormalAction	OCTET STRING	read-write	Used to configure user defined actions when normal events occur.
20	[fanTray/pem]fruBoardDescription	DisplayString	read-only	Used to read the Board Description from the FRU.
21	[fanTray/pem]fruBoardManufacturer	DisplayString	read-only	Used to read the Board Manufacturer from the FRU.
22	[fanTray/pem]fruBoardPartNumber	DisplayString	read-only	Used to read the Board Part Number from the FRU.
23	[fanTray/pem]fruBoardSerialNumber	DisplayString	read-only	Used to read the Board Serial Number from the FRU.
24	[fanTray/pem]fruBoardManufactureDate	DisplayString	read-only	Used to read the Board Manufacture Date from the FRU.

Table 73. [FanTray/pem]TargetTable/[fanTray/pem]TargetEntry (1.3.6.1.4.1.343.2.14.2.10.[5/6].52.1) (Sheet 2 of 2)

OID	Object	Syntax	Access	Value
25	[fanTray/pem]fruProductDescription	DisplayString	read-only	Used to read the Product Description from the FRU.
26	[fanTray/pem]fruProductManufacturer	DisplayString	read-only	Used to read the Product Manufacturer from the FRU.
27	[fanTray/pem]fruProductPartNumber	DisplayString	read-only	Used to read the Product Part Number from the FRU.
28	[fanTray/pem]fruProductSerialNumber	DisplayString	read-only	Used to read the Product Serial Number from the FRU.
29	[fanTray/pem]fruProductManufactureDate	DisplayString	read-only	Used to read the Product Manufacture Date from the FRU.
30	[fanTray/pem]fruProductModel	DisplayString	read-only	Used to read the Product Model from the FRU.
31	[fanTray/pem]fruProductRevision	DisplayString	read-only	Used to read the Product Revision from the FRU.
32	[fanTray/pem]EventAction	OCTET STRING	read-write	Used to associate a script to a particular event.

Table 74. [FanTray/pem]FruTable/[fanTray/pem]FruEntry (1.3.6.1.4.1.343.2.14.2.10.[5/6].53.1)

OID	Object	Syntax	Access	Value
1	[fanTray/pem]Number	INTEGER	none	index 1
2	[fanTray/pem]FruNumber	INTEGER	none	index 2
3	[fanTray/pem]FruShelfManagerControlled Activation	INTEGER	read-write	Get/Set whether the Shelf Manager activates the FRU or not. none(0), activate(1).
4	[fanTray/pem]FruHotSwapState	DisplayString	read-write	The hot swap state as read from the FRU and the CMM's internally known hot swap state.
5	[fanTray/pem]FruLedProperties	DisplayString	read-only	Find out what LEDs the FRU supports.
6	[fanTray/pem]FruPowerLevels	DisplayString	read-only	Returns the power levels available for a FRU and the number of watts drawn by each.
7	[fanTray/pem]FruPresentPowerLevel	DisplayString	read-write	Get/Set the power level of an entity.
8	[fanTray/pem]FRUActivation	INTEGER	read-write	Get/Set the activation state to either activate or deactivate the FRU. The Deactivate is the same as a Graceful Shutdown.
9	[fanTray/pem]FRUActivationPolicy	INTEGER	read-write	Get/Set the locked bit of the activation policy.
10	[fanTray/pem]FRUControl	INTEGER	write-only	Set the FRU to do things like Cold Reset, Warm Reset, etc. See the AdvancedTCA specification for possibilities.
11	[fanTray/pem]FRUDeactivationPolicy	INTEGER	read-write	Get/Set FRU Deactivation Policy.

Table 75. [FanTray/pem]FruTargetTable/[fanTray/pem]FruTargetEntry (1.3.6.1.4.1.343.2.14.2.10.[5/6].54.1)

OID	Object	Syntax	Access	Value
1	[fanTray/pem]Number	INTEGER	none	index 1
2	[fanTray/pem]FruNumber	INTEGER	none	index 2
2	[fanTray/pem]Target	DisplayString	none	index 3
4	[fanTray/pem]FruLedColorProps	DisplayString	read-only	Finds out what colors are supported by the LED.
5	[fanTray/pem]FruLedState	DisplayString	read-write	Get/Set the FRU LED state.

17.3 SNMP Agent

The SNMP agent (snmpd) listens to SNMP v1 queries (gets/sets) by default, evokes the corresponding MIB Module to process the request, and sends the SNMP response with return data to the SNMP/MIB manager. The agent can also be configured to respond to v3 queries. The SNMP agent in the CMM is implemented to support SNMP-get, SNMP-get-next, and SNMP-set for MIB-II and the CMM MIB objects.

All SNMP “set” queries are logged in the command log file.

Note: Changing IP addresses when using SNMP v3 will require a restart of the snmpd process. This is due to SNMP v3 using the IP address to generate the encryption keys. To restart the snmpd process follow steps 2 through 4 in Section 17.3.2.

17.3.1 Configuring the SNMP Agent Port

The SNMP agent is set up to use port 161 by default. The port can be configured to use a different port number by adding the following line to the snmpd.conf file:

```
agentaddress port_number
```

17.3.2 Configuring the Agent to Respond to SNMP v3 Requests

The agent can also be configured to respond to SNMP v3 queries, in which case it checks the security, decodes the request, then evokes the corresponding MIB Module to process the request, and sends the SNMP response with return data to the SNMP/MIB manager. SNMP v3 adds support for strong authentication and private communication.

To change the SNMP agent to respond to SNMP v3 queries:

1. Copy /etc/snmpdv3.conf to /etc/snmpd.conf by issuing the command:

```
cp /etc/snmpdv3.conf /etc/snmpd.conf
```

2. Find the process ID of the snmpd process by issuing the command:

```
ps -ax
```

3. Look for the snmpd process. There may be more than one, so use the lowest process ID, which is the first instance of the process. The listing will look similar to this:

```
121 ?        SN     0:00 /usr/sbin/snmpd -c /etc/snmpd.conf
```

4. Restart the snmpd agent by issuing the following command:

```
kill -s SIGHUP [snmpd_process_id]
```

Where snmpd_process_id is the process ID of the snmpd process found in the step above.

17.3.3 Configuring the Agent Back to SNMP v1

To reconfigure the agent back to SNMP v1, follow the same steps as above substituting the /etc/snmpdv1.conf for /etc/snmpdv3.conf, using the command:

```
cp /etc/snmpdv1.conf /etc/snmpd.conf
```

17.3.4 Setting up an SNMP v1 MIB Browser

By default the community name for the SNMP agent on the CMM is ‘public’ for both read and write. This can be changed by editing the /etc/snmpd.conf file on the CMM. The SNMP MIB browser needs to match the community name (default ‘public’) for both read and write.

17.3.5 Setting up an SNMP v3 MIB Browser

To manage the CMM via SNMP v3 MIB browser/manager (mg-soft, net-snmp, etc.), the user needs to configure the browser with the following parameters:

1. Load and compile the MPCMM0001.MIB file
2. Set the SNMP v3 security parameters:
   - Set SNMP v3 agent user
     At default, User: **root**
   - Set the MD5 Authentication password: **cmmrootpass**
   - Set the DES Encryption password: **cmmrootpass**

17.3.6 Changing the SNMP MD5 and DES Passwords

To change the MD5 Authentication and DES Encryption passwords for the SNMP interface on the CMM, use one of the following methods:

Method 1:

1. Edit the file /etc/var/snmpd.conf on the active CMM and add the following line:

```
createUser root MD5 cmmrootpass DES cmmrootpass
```

This line allow the creation of user root with MD5 authentication password as cmmrootpass, and DES encryption password as cmmrootpass.

2. Add more lines for more users if needed.

3. Restart SNMP agent

Method 2:

Use the 'snmpusm' utility from a Redhat* Linux* host which has net-snmp packet install. You can find the usage under http://www.net-snmp.org.

17.4 SNMP Trap Utility

The SNMP trap utility is utilized by the CMM Management software to send SNMP trap messages to a remote application regarding any abnormal system events. When enabled, the CMM will issue SNMP v1 traps on port 162. The CMM can also be configured to issue SNMP v3 traps. The SNMP trap port can also be configured.

Note: If the “snmptrapversion” value is not in the cmm.cfg file or if the cmm.cfg file doesn't exist the firmware will default to Trap Version 3.

17.4.1 Configuring the SNMP Trap Port

To configure the SNMP Trap Port to a different port number, issue the command:

```
cmmset -l cmm -d SNMPTrapPort -v [Port Number]
```

Where Port Number is the desired SNMP Trap Port number.

17.4.2 Configuring the CMM to Send SNMP v3 Traps

To configure the CMM to send SNMP v3 Traps, issue the command:

```
cmmset -l cmm -d SNMPTrapVersion -v v3
```

17.4.3 Configuring the CMM to Send SNMP v1 Traps

To configure the CMM to send SNMP v1 Traps, issue the command:

```
cmmset -l cmm -d SNMPTrapVersion -v v1
```

17.5 Configuring and Enabling SNMP Trap Addresses

The CMM allows up to five SNMP trap addresses destinations, SNMPTrapAddress1-5. In redundant CMM systems, **SNMP Trap Address 1 MUST be set to a valid IP address on the network that the CMM can ping.** This is used as a test of network connectivity as well as being the first SNMP Trap Address.

When the CMM is configured to send SNMP v3 traps, it is recommended that only one SNMPTrapAddress is configured due to the large amount of traps that can be generated on a loaded system.

17.5.1 Configuring an SNMP Trap Address

To configure an SNMP trap address, issue the command:

```
cmmset -l cmm -d SNMPTrapAddressN -v [IP address]
```

where N is the number of the trap address from 1 to 5 that is being set, and IP address is the IP address of the trap receiver.

17.5.2 Enabling and Disabling SNMP Traps

SNMP Trap addresses are enabled by default. To disable SNMP traps, issue the command:

```
cmmset -l cmm -d SNMPenable -v disable
```

To enable SNMP trap addresses, issue the command:

```
cmmset -l cmm -d SNMPenable -v enable
```

To check the status of SNMP traps, issue the command:

```
cmmget -l cmm -d SNMPenable
```

17.5.3 Alerts Using SNMP v3

The CMM utilizes the SNMPtrap utility to send the SNMP v3 ‘trap’ message**.** For receiving the SNMP v3 Trap, the remote application such as the trap listener needs to:

1. Set the SNMP v3 Trap User. The default trap user is **root**.
2. Set the MD5 Authentication password. The default MD5 Authentication password: is **publiccmm**.
3. Set the DES Encryption password. The default DES Encryption password is **publiccmm**.

Note: To change the passwords (MD5 and DES) for SNMP v3 trap, change the snmpTrapCommunity from the CLI interface using “`cmmset -d snmpTrapCommunity`” or from the SNMP manager console.

17.5.4 Alert Using UDP Alert

The CMM Management software also sends the alert via UDP port 10000. The IP address to receive this alert is the same as the SNMP Trap IP addresses. The UDP alert is enabled by default when the user enables SNMP traps and disabled when SNMP Traps are disabled. See Section 17.5.2 for enabling and disabling SNMP Traps.

Note: The UDP Alert portion of the SNMP trap functionality is being deprecated. This function has become obsolete and will be removed in a future revision of firmware.

17.6 SNMP Security

17.6.1 SNMP v1 Security

SNMP version 1 utilizes the community name for authentication. If the SNMP manager/client sends a request message containing the community name that doesn't match the community name set in the SNMP Agent, the SNMP agent will respond with the authentication failure message. This community name is not encrypted during transmission.

17.6.2 SNMP v3 Security - Authentication Protocol and Privacy Protocol

The CMM supports the highest security level for SNMP v3. MD5 is used for the authentication protocol in the CMM. The DES is used for the privacy protocol in the CMM. When in this mode, the user needs to specify each password (authKey, privKey) for these protocols. The SNMP v3 packet is securely encrypted during the transmission. This is the default security level of the CMM when configured for SNMP v3.

The following fields are defined to handle all SNMP v3 security levels:

Table 76. SNMP v3 Security Fields For Traps

Security Name	User Name	Default Value:
SecurityName	User name	root
AuthProtocol	authentication type	MD5
AuthKey	authentication password	**publiccmm**
PrivProtocol	privacy type	DES
PrivKey	privacy password	**publiccmm**

Table 77. SNMP v3 Security Fields For Queries

SecurityName	user name	Default Value:
SecurityName	User name	root
AuthProtocol	authentication type (MD5)	MD5
AuthKey	authentication password	**cmmrootpass**
PrivProtocol	privacy type (DES)	DES
PrivKey	privacy password	**cmmrootpass**

17.7 SNMP Trap Descriptions

The list of possible SNMP trap strings is listed in Section 11, "Health Events" on page 104.

17.8 Snmpd.conf File

For more information regarding SNMP configuration and the snmpd.conf file, please visit:

http://www.net-snmp.org/man/snmpd.conf.html

CMM Scripting 18

18.1 CLI Scripting

In addition to calling the CLI directly, commands can easily be called through scripts using “bash” shell scripting. These scripts could be tailored to create a single command from several CLI commands or to give more detailed information. For example, you may want to display all of the fans and their speeds on the server. A script could be written that would first call the CLI to find out what fan trays are present. Next it would find out what fan sensors are in each fan tray. Finally it would call the CLI to get the current speeds of each of the fans.

Scripts can be written directly on the CMM and should be saved on the CMM as a file in flash in the /home/scripts directory.

Script files need to have execute permissions set.

18.1.1 Script Synchronization

Script files are not synchronized automatically when changed (by deleting, renaming, altering its chown/chmod/chgrp permissions, etc) in the /home/scripts directory on the active CMM. The Linux “touch” command should be issued to the /home/scripts directory on the active CMM to synchronize the scripts.

Scripts are automatically synchronized on any full synchronization (chassis power up, insertion of a new CMM) as well as when they are edited (e.g., using vi and writing the file) or copied.

Scripts need to be deleted from both CMMs manually. Deleting a script on the active CMM does not automatically delete the script on the standby.

Note: Scripts are always synchronized from the active CMM to the standby CMM, not standby to active. Adding, editing, or modifying scripts on the standby CMM should be done with caution, and will need to be manually copied to the active CMM.

18.2 Event Scripting

Health events triggered on the CMM can be used to execute scripts stored locally in the /home/scripts directory. Any level of an event can be used as a trigger: normal, minor, major, and critical. The CLI command for associating a script with an event is:

```
cmmset –l [location] –t [target] –d [action type] –v [script]
```

Location is the component in the chassis that the health event is associated with.

Target is the sensor to be triggered on.

Action type is NormalAction, MinorAction, MajorAction, or CriticalAction, based upon the severity of the event to be triggered on.

Script is the script file or executable in the /home/scripts directory to be run including parameters to be sent to the script. The script and parameters should be enclosed in quotes. The *script* parameter should not include the /home/scripts path.

Note: If applicable, remember to set a script for when a sensor returns to normal (NormalAction). A script will not automatically stop running when a sensor returns to a normal setting (no alarms or events).

For example, if you wanted to run a blade powerdown script called "powerbladedown" stored in /home/scripts when the ambient temperature triggered a major event for a blade 4, the command would be:

```
cmmset –l blade4 –t "Ambient Temp" –d MajorAction –v "bladepowerdown 4"
```

In this example, the /home/scripts/bladepowerdown script or executable will be executed with "4" as a parameter when blade4's Ambient Temp sensor generates a major health event.

18.2.1 Listing Scripts Associated With Events

The scripts that are associated with events are put into the /etc/actionscripts.cfg file. To display a list of scripts associated with events, cat the /etc/actionscripts.cfg file.

For further information on event scripting commands and its arguments, see Section 8, "The Command Line Interface (CLI)" on page 71.

18.2.2 Removing Scripts From an Associated Event

To remove scripts from occurring for an event in which they have been associated, issue the following command:

```
cmmset –l [location] –t [target] –d [action type] –v none
```

Location is the component in the chassis that the health event is associated with.

Target is the sensor that was being triggered on.

Action type is NormalAction, MinorAction, MajorAction, or CriticalAction based upon the severity of the event that was being triggered on.

18.3 Setting Scripts for Specific Individual Events

The CMM allows scripts to be associated with specific events which may not necessarily be health related, such as the assertion of a threshold sensor. This allows any single event that can occur on the CMM to have an associated script, or scripts with it.

18.3.1 Event Codes

To allow the user to set scripts based on any event, a unique event code is assigned to each event that can occur on the CMM. The list of events and the codes associated with each event is detailed in Section 11, "Health Events" on page 104.

18.3.2 Setting Event Action Scripts

Setting event action scripts can be done using any of the standard CMM interfaces (CLI, SNMP, RPC). The format for the CLI command is as follows:

```
cmmset  -l [location] -t [Sensor Name] -d EventAction -v [Event
Code]:[script to be run]
```

This setting gets written to the actionscripts.cfg file and is synced to the standby CMM. It is persistent across boots.

18.4 Running CMM Event Scripts on CMM State Transitions (Active/Standby/Ready/Not Ready)

Event scripts can be set to run when a CMM becomes active, and to cease running when a CMM becomes the standby CMM. The script will begin running on a newly active CMM following any kind of failover.

The following is a list of the events and their associated event ID's to be used for associating a script.

Table 78. CMM State Transition Events and Event IDs

Event ID	Event Name
1024	CMM is Not Ready
1025	CMM is Ready
1026	CMM is Active
1027	CMM is Standby
1028	CMM ready timed out

The Active event is an EventAction fired as a result of the SEL entry when the CMM transitions to the Active state. Similarly, the Standby event is an EventAction fired as a result of the SEL entry when the CMM transitions to the Standby state. These are not health related events.

The "CMM not ready" and "CMM ready" events are triggered internally by CMM firmware. It will log "CMM not Ready" when the CMM first becomes active. Once all internal status bits are set by firmware, it will log "CMM is Ready" and deassert the "Not Ready" state. The status bits set by the hardware are described below.

If the CMMReadyTimeout is set to a positive number of seconds, then an event is asserted if the CMM has not become ready with in that time. This is deasserted once the CMM is Ready or goes to Standby. This is a minor health related event.

18.4.1 Sensor Data Bits

The CMM Status sensor is a normal discrete sensor, defined in the CMM's SDR file, and it has an entry in the sensor table. The sensor status bits are assigned as follows:

Table 79. CMM Status Sensor Data Bits

Bit	Bit Name	Explanation
0	RUNNING	Set when the Active/Standby election has taken place, and reset at a restart of the CMM.
1	ACTIVE	Set when CMM is Active
2	bENUMERATION	Set when the initial blade discovery is finished
3	WRAPPERS	Set when the WrapperProcesses are running
4	SNMP	Set when SNMP daemon’s tables are initially populated
14	TIMEOUT	Set when CMM exceeds a timeout waiting to become ready

The Running bit is used to be sure the Active/Standby election has taken place and the remaining status bits are valid. All bits are initialized to 0 on CMM startup and bRUNNING is set to 1 by the election process.

When the active election has taken place, the CMM will transition to active or to standby mode. These will set or clear bACTIVE and log either CMM_Active or CMM_Standby in the SEL. The SEL events will trigger the SNMP traps and launch any attached EventAction scripts.

bENUMERATION will be set either by the initial powerup scan of the FRUs, or by a Re-Enumeration. The firmware will start checking the startup of the WrapperProcesses; bWRAPPERS is set when the WPs are up. Once they are up, the SNMPD is watched for its report on the tables being filled out, and bSNMP will then be set. If a timeout value has been set and this process takes longer then that timeout, bTIMEOUT will be set; it is cleared again once all the others are set and the CMM is fully ready.

These bits, except for bRUNNING, will be cleared when CMM goes to standby.

18.4.2 Retrieving the Value of the Data Sensor Bits

To retrieve the state of the CMM status sensor bits, the following CLI command is used:

```
cmmget –l cmm –t “cmm status” –d current
```

Output of the CLI command is in the form:

```
The current value is 0xNNNN

CMM is [Active/Standby]

CMM enumeration is completed

CMM is [Read/Not Ready, waiting for bladeY]
```

Where:

NNNN: Is the hex representation of the CMM status sensor bits.

Y: The number of the blade whose wrapper process has not started.

These strings will be driven directly by the value of the identified status bits in the sensor. In addition, when in the Not Ready state, information about which blades are not yet running will also be printed. As with other CMM sensor data, this item is queryable in the Standby CMM.

18.4.3 CMMReadyTimeout Value

The CMMReadyTimeout value is a 16 bit unsigned value signifying the number of seconds in which the CMM should become Ready after becoming Active. If the CMM takes longer than this amount of time, a CMM Ready timed out event will be asserted. This is a health related event.

The CMMReadyTimeout value will be set and queryable from the CLI using the "CMM Status" target and CMMReadyTimeout data item. By default the CMMReadyTimeout is set to 0, or no timeout (infinite). A default timeout in CMM will be set to infinite timeout.

This value will be stored in the file /etc/cmm.cfg with the string "CMMReadyTimeout = n", thus made permanent and copied to the redundant CMM. When an infinite timeout is set, the CMMReadyTimeout entry in CMM.CFG file is removed.

The new value will be applied to the next Ready/Not Read cycle. It will not affect the current ready timeout scan. This is, the timeout value will be read when a "Not Ready" is triggered.

18.4.3.1 Setting the CMMReadyTimeout Value

The CMMReadyTimeout value can be set using the CLI command:

```
cmmset -l cmm -t “CMM Status” -d CMMReadyTimeout -v [Number of Seconds to
Timeout]
```

18.4.3.2 Retrieving the CMMReadyTimeout Value

The CMMReadyTimeout value can be retrieved using the CLI command:

```
cmmget -l cmm -t “CMM Status” -d CMMReadyTimeout
```

The output of the command will be in the form:

```
CMMReadyTimeout is NN seconds.
```

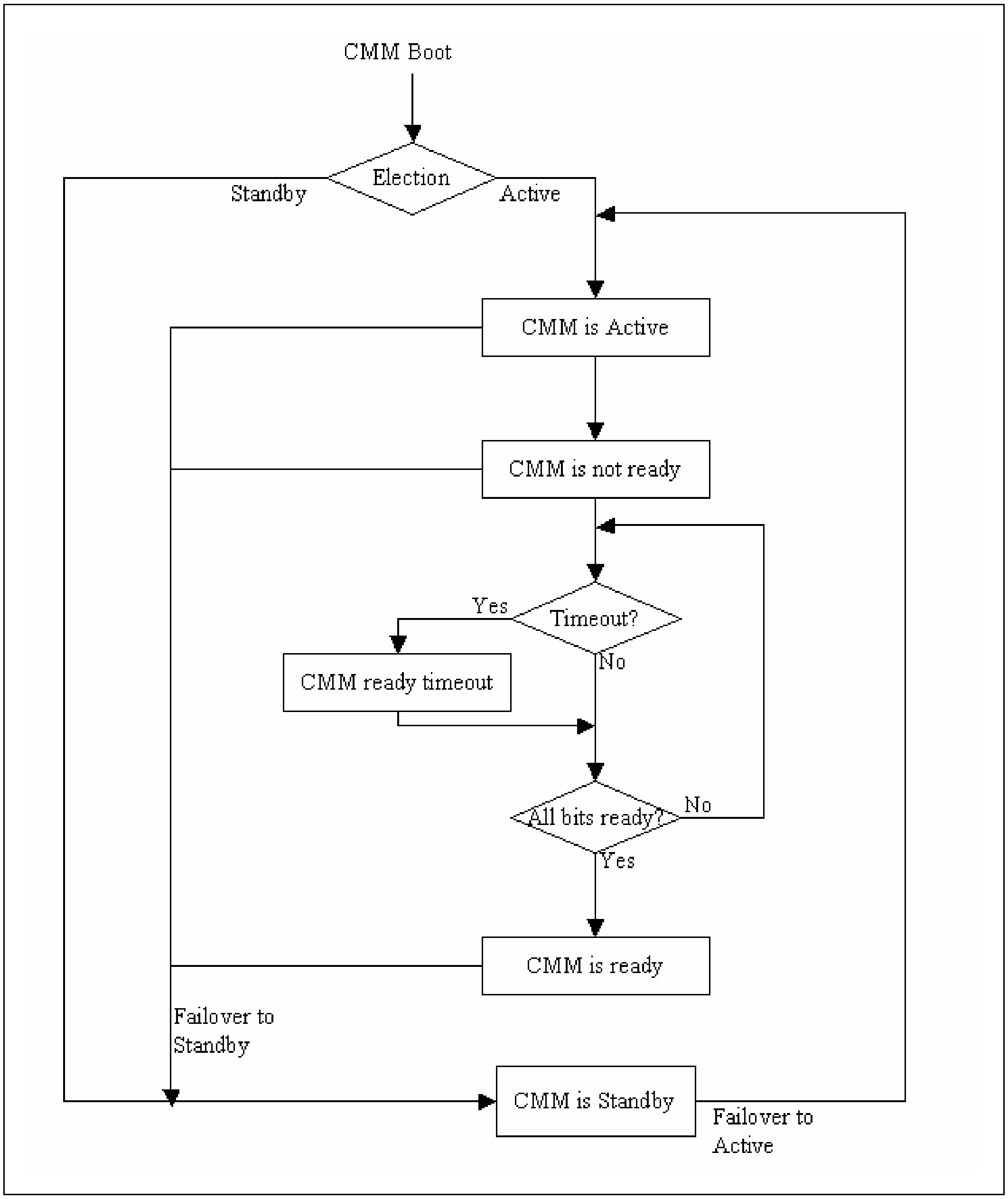
Where NN is the number of seconds the timeout value is set for.

Note: Triggering a user script off of CMMReadyTimeout may not occur in the amount of time reflected in the CMMReadyTimeout value. There may be a gap between when the CMMReadyTimeout occurs and the actual script runs.

18.4.4 CMM State Transition Model

The following diagram illustrates the state transitions for the CMM Status.

Figure 5. CMM Status State Diagram



18.5 FRU Control Script

The CMM will check for a file in /home/scripts called frucontrol before it activates or deactivates any FRU. If that file does not exist, the CMM automatically activates or deactivates all FRUs. If it does exist, the CMM attempts to execute the program, and that is all. It is then up to the /home/

scripts/frucontrol file to do the proper activation or deactivation at some later point. The option for this script is useful for having a separate entity or system manager to be in charge of activating and deactivating FRUs instead of the CMM.

Note: The activation and deactivation of the CMM itself is not controlled by the frucontrol script.

18.5.1 Command line arguments

When the CMM invokes the frucontrol script, it passes the following arguments:

frucontrol <IPMB Address> <FRU ID> <Previous State> <Current State>

Where:

IPMB Address: A valid IPMB address which identifies one of the FRUs on the chassis. The value passed in is in hex, but the '0x' is stripped off. For example, if the IPMB address of a blade is "0x8C", then the argument passed in would be "8c" and not "0x8c". The case of letters is small.

FRU ID: FRU id of any device fronted by the location identified by the IPMB address. Typically the value passed is "0", but it could be "1", "2", "3 "etc.

Previous State: One of the valid M states. A value of "1" through "7" is used to represent states M1 through M7 respectively.

<Current State>: Same as <Previous State>

18.5.2 Sample frucontrol file

```
#!/bin/sh

# Copyright (c) 2004-2005, Intel Corporation.

# This script is furnished under license and may only be used or copied in

# accordance with the terms of that license. No license, express or
implied,

# by estoppel or otherwise, to any intellectual property rights is granted
by

# this document. The Software is subject to change without notice, and
should

# not be construed as a commitment by Intel Corporation to market,
license,

# sell or support any product or technology. Unless otherwise provided for
in

# the license under which this Software is provided, the Software is
provided

# AS IS, with no warranties of any kind, express or implied. Except as

# expressly permitted by the Software license, neither Intel Corporation
```

```
nor

# its suppliers assumes any responsibility or liability for any errors or

# inaccuracies that may appear herein. Except as expressly permitted by
the

# Software license, no part of the Software may be reproduced, stored in a

# retrieval system, transmitted in any form, or distributed by any means
without

# the express written consent of Intel Corporation.


# Command line arguments passed to this script

#

# $1 = IPMB address

# $2 = FRU ID

# $3 = Previous state (1 - M1, 2 - M2, ...)

# $4 = Current state  (1 - M1, 2 - M2, ...)

#


DEVICE=""

case $1 in

    98) DEVICE=blade13:fru$2

          ;;

    90) DEVICE=blade11:fru$2

          ;;

    8c) DEVICE=blade10:fru$2

          ;;

    42) DEVICE=fantray1:fru$2

          ;;

    62) DEVICE=pem1:fru$2

          ;;

    64) DEVICE=pem2:fru$2

          ;;
    *) echo Invalid device IPMB address  >> /var/log/debug.log
          ;;
esac

#
# ACTIVATION
#
date >> /var/log/frucontrol.log
if [ "$4" = "2" ] ; then
    echo activating device $DEVICE >> /var/log/frucontrol.log
    while true
        do
            cmmset -l $DEVICE -d fruactivation -v 1
            if [ $? -eq 0 ] ; then
                break
            fi
        done
    echo done activating device $DEVICE >> /var/log/frucontrol.log
    cmmget -l $DEVICE -d hotswapstate
fi

#
# DEACTIVATION
#
if [ "$4" = "5" ] ; then
    echo deactivating device $DEVICE >> /var/log/frucontrol.log
    while true
    do
        cmmset -l $DEVICE -d fruactivation -v 0
        if [ $? -eq 0 ] ; then
```

```
            break
        fi
    done
    echo done deactivating device $DEVICE >> /var/log/frucontrol.log
fi
date >> /var/log/frucontrol.log
```

Remote Procedure Calls (RPC) 19

The CMM can be administered by custom remote applications via remote procedure calls (RPC). RPCs provide all of the functionality of the CLI.

RPC calls are useful for managing the CMM from:

- An administrator’s machine via the house network
- Another blade in the same chassis as the CMM, via the chassis backplane network
- An application running on the CMM itself

Note: System Event Log (SEL) information is currently not available through the RPC interface.

19.1 Setting Up the RPC Interface

Before you can use RPC calls in a custom application, you must obtain the following RPC source code files (contact your Intel representative):

- rcliapi.h
- rcliapi_xdr.c
- rcliapi_clnt.c
- cli_client.c
- cli_client.h

The first three files should be compiled and linked into the application under development. These files implement the RPC calling subsystem for use in an application.

The file cli_client.c contains a small sample program for calling the CMM through RPC. These files currently support Linux* and VxWorks*.

The last file, cli_client.h, contains declarations and function prototypes necessary for interfacing with the RPC calling subsystem. Include the file in a ‘#include’ statement in all application files that make RPC calls.

19.2 Using the RPC Interface

The RPC interface may be used to manage the CMM whether the calling application is on a remote network, on a blade in the same chassis as the CMM, or even running on the CMM itself.

The following two functions are defined by the RPC subsystem for calling the CMM through RPC:

- GetAuthCapability()
- ChassisManagementApi()

19.2.1 GetAuthCapability()

The following is the calling syntax for GetAuthCapability():

```
int GetAuthCapability(
        char*   pszCMMHost,
        char*   pszUserName,
        char*   pszPassword
        );
```

Parameters
pszCMMHost:[in] IP Address or hostname of CMM
pszUserName:[in] A valid CMM user name
pszPassword:[in] Password corresponding to *pszUserName*

Return Value
>0 Authentication successful. The return value is the authentication code.

-1 Invalid username / password combination.

E_RPC_INIT_FAILRPC initialization failure.

E_RPC_COMM_FAILRPC communication failure.

GetAuthCapability is used to authenticate the calling application with the remote CMM. The remote CMM will not respond to RPC communications until the application has successfully authenticated. To authenticate, the application will need to pass the CMM's current IP address, username and password to GetAuthCapability(). The default username and password are 'root' and 'cmmrootpass'. When the authentication is successful, GetAuthCapability() will return an authentication code for use with all further RPC communication.

Note: Clients will need to re-authenticate to the CMM whenever the CMM is reset. Re-authentication is necessary when ChassisManagementApi() returns E_ECMM_SVR_AUTH_CODE_FAIL.

19.2.2 ChassisManagementApi()

The following is the calling syntax for ChassisManagementApi():

```
int ChassisManagementApi(
        char*            pszCMMHost,
        int              nAuthCode,
        unsigned int     uCmdCode,
        unsigned char*   pszLocation,
        unsigned char*   pszTarget,
        unsigned char *  pszDataItem,
        unsigned char *  pszSetData,
        void **          ppvbuffer,
        unsigned int *   uReturnType
        );
```

Parameters

pszCMMHost [in] IP Address or DNS hostname of CMM.

nAuthCode[in] Authentication code returned by GetAuthCapability().

uCmdCode[in] The command to be executed (CMD_GET or CMD _SET as defined in cli_client.h).

pszLocation[in] The location that contains the data item that *uCmdCode* acts upon, such as system, CMM, or blade1.

pszTarget[in] The target that contains the attribute that *uCmdCode* acts upon, such as the sensor name as listed in the Sensor Data Record (SDR). When not applicable, use "NA" (such as when *pszDataItem* is an attribute of the *pszLocation* rather than *pszTarget.*

pszDataItem[in] The attribute that *uCmdCode* acts upon, either an attribute of *pszLocation* or *pszTarget*.

pszSetData[in] The new value to set. When not applicable, use "NA".

ppvbuffer[out] A pointer to the buffer containing the returned data.

uReturnType[out] The type of data that *ppvbuffer* points to (see the defines in cli_client.h).

The value definitions of the return codes may be found in Table 80, "Error and Return Codes for the RPC Interface" on page 177.

Once the application has authenticated, it may proceed to get and set CMM parameters by calling ChassisManagementApi(). For each call to ChassisManagementApi(), the calling application must pass in the authentication code returned from GetAuthCapability(). The get and set commands available through ChassisManagementApi() are the same as those available on the command-line interface with cmmget and cmmset.

Note: SEL information is currently not available through the RPC interface.

Table 80. Error and Return Codes for the RPC Interface (Sheet 1 of 4)

Code #:	Error Code String	Error Code Description
0	E_SUCCESS	Success
1	E_BPM_BLADE_NOT_PRESENT	Blade isn't in the chassis.
2	E_ECMM_SVR_COMMAND_UNSUPPORTED	ECMM_SVR: Unsupported Command Error.
3	E_CLI_MSG_SND	CLI Send Message Error.
4	E_CLI_INVALID_TARGET	Not a valid -t parameter.
5	E_CLI_INVALID_LOCATION	Not a valid -l location.
6	E_CLI_INVALID_DATA_ITEM	Not a valid -d parameter.
7	E_CLI_INVALID_SET_DATA	Not a valid -v parameter.
8	E_CLI_INVALID_REQUEST	CLI Invalid Request Error.
9	E_CLI_MSG_RCV	CLI Receive Message Error.
10	E_CLI_NO_MORE_DATA	No data found to retrieve.
11	E_CLI_DATA_TYPE_UNSUPPORTED	CLI Data Type Unsupported.
12	E_ECMM_CLIENT_CONNECT_ERROR	ECMM_CLIENT: RPC Connect Error.
13	E_ECMM_SVR_AUTH_CODE_FAIL	Invalid auth code passed to RPC interface.
14	E_CLI_STANDBY_CMM	Operation cannot be performed on standby CMM.
15	E_WP_INITIALIZING	The CMM is Initializing and Not Ready. Wait a few seconds and try again.
16	E_BPM_NON_IPMI_BLADE	Blade does not support IPMI.
17	E_BPM_STANDBY_CMM	BPM operation cannot be performed on standby CMM.
18	E_BPM_NO_MORE_DATA	Couldn't delete a board from the drone mode list.
19	E_BPM_INVALID_SET_DATA	Not a valid -v parameter.
20	E_CLI_INVALID_BUFFER	Internal CMM Error
21	E_CLI_INVALID_CMM_SLOT	Internal CMM Error
22	E_CLI_NO_MSGQ_KEY	Internal CMM Error
23	E_CLI_NO_MSGQ	Internal CMM Error
24	E_CLI_NO_MSGQ_LOCK	Internal CMM Error
25	E_CLI_NO_MSGQ_UNLOCK	Internal CMM Error
26	E_CLI_FILE_OPEN_ERROR	Internal CMM Error
27	E_CLI_CFG_WRITE_ERROR	CMM Config File Error.
28	E_IMB_NO_MSGQ	Internal CMM Error
29	E_IMB_NO_MSGQ_KEY	Internal CMM Error
30	E_IMB_SEND_TIMEOUT	Internal CMM Error
31	E_IMB_DRIVER_FAILURE	Internal CMM Error
32	E_IMB_REQ_TIMEOUT	A blade is not responding to IPMI requests.
33	E_IMB_RECEIVE_TIMEOUT	A blade is not responding to IPMI requests.

Table 80. Error and Return Codes for the RPC Interface (Sheet 2 of 4)

Code #:	Error Code String	Error Code Description
34	E_IMB_COMPCODE_ERROR	An IPMI request returned with a non successful completion code. User should try the command again.
35	E_IMB_INVALID_PACKET	Invalid IPMI response. Blade may be returning invalid data.
36	E_IMB_INVALID_REQUEST	Invalid IPMI response. Blade may be returning invalid data.
37	E_IMB_RESPONSE_DATA_OVERFLOW	Invalid IPMI response. Blade may be returning invalid data.
38	E_IMB_DATA_COPY_FAILED	Internal CMM Error
39	E_IMB_INVALID_EVENT	Internal CMM Error
40	E_IMB_OPEN_DEVICE_FAILED	Internal CMM Error
41	E_IMB_MMAP_FAILED	Internal CMM Error
42	E_IMB_MUNMAP_FAILED	Internal CMM Error
43	E_IMB_RESP_LEN_ERROR	Invalid IPMI response. Blade may be returning invalid data.
44	E_NEM_SNMPTRAP_ERROR	Error setting snmp trap parameters. Retry command
45	E_NEM_SYSTEMHEALTH_ERROR	Internal CMM Error.
46	E_NEM_GETHEALTH_ERROR	Internal CMM Error.
47	E_NEM_SNMPENABLE_ERROR	Internal CMM Error.
48	E_NEM_SENSOR_HEALTH_ERROR	Internal CMM Error.
49	E_NEM_FILTER_SEL_ERROR	Internal CMM Error.
50	E_NEM_INITIALIZE_ERROR	Internal CMM Error.
51	E_NEM_SENSOR_EVENT	Internal CMM Error.
52	E_NEM_SENSOR_ERROR	Internal CMM Error.
53	E_NEM_SNMP_PROCESS_EVENT_ERROR	Internal CMM Error.
54	E_NEM_SNMP_DEST_ADDR_ERROR	SNMP Trap address that the user is setting is invalid.
55	E_NEM_SNMP_COMMUNITY_STRING_ERROR	SNMP Community that user is setting is invalid.
56	E_NEM_SNMP_TRAP_VERSION_ERROR	SNMP Trap version that the user is setting is invalid.
57	E_NEM_SNMP_TRAP_PORT_ERROR	SNMP Trap port that the user is setting is invalid.
58	E_NEM_SNMP_CFG_ERROR	Cannot read SNMP config parameter. Config file may be corrupted.
59	E_NEM_SEND_SNMP_TRAP_ERROR	Internal CMM Error.
60	E_SFS_INVALID_TRANSACTION	Internal CMM Error.
61	E_SFS_LOCK_SDR	Can't read SDRs. Blade may be busy, try again.
62	E_SFS_ENTITY_ID	Internal CMM Error.
63	E_SFS_DEVICE_LOCATOR_NULL	Internal CMM Error.

Table 80. Error and Return Codes for the RPC Interface (Sheet 3 of 4)

Code #:	Error Code String	Error Code Description
64	E_SFS_NO_MEMORY	Internal CMM Error.
65	E_SFS_UNSUPPORTED_DEVICE	Internal CMM Error.
66	E_SFS_RESPONSE_LENGTH	Internal CMM Error.
67	E_SFS_RESPONSE_DATA	Internal CMM Error.
68	E_SFS_POWER_SUPPLY_FRU	Internal CMM Error.
69	E_SFS_PATTERN_FOUND	Internal CMM Error.
70	E_SFS_SEMAPHORE_FAILED	Internal CMM Error.
71	E_SFS_CALLBACK_NOT_FOUND	Internal CMM Error.
72	E_SFS_END_OF_DATA	Internal CMM Error.
73	E_SFS_NO_SEL_ENTRY	Internal CMM Error.
74	E_SHEM_INTERNAL_ERROR	Internal CMM Error.
75	E_SHEM_INVALID_DATA_ITEM	Not a valid -d parameter.
76	E_SHEM_STANDBY_CMM	Can't execute this command on the standby CMM.
77	E_SNSR_STATUS_UNSUPPORTED	Internal CMM Error.
78	E_SNSR_UNSUPPORTED	Internal CMM Error.
79	E_SNSR_CATEGORY	Internal CMM Error.
80	E_SNSR_NO_MEMORY	Internal CMM Error.
81	E_SNSR_NOT_FOUND	Internal CMM Error.
82	E_SNSR_ACTION_UNSUPPORTED	Internal CMM Error.
83	E_SNSR_NON_FIRMWARE	Internal CMM Error.
84	E_SNSR_SHARE_CODE	Internal CMM Error.
85	E_SNSR_LOW_STORAGE	Internal CMM Error.
86	E_SNSR_EVENT_TYPE	Internal CMM Error.
87	E_SNSR_INVALID_REQUEST	Internal CMM Error.
88	E_SNSR_OS_ERROR	Internal CMM Error.
89	E_SNSR_PROCESSOR_NOT_PRESENT	Internal CMM Error.
90	E_SNSR_THRESHOLD_UNSUPPORTED	The sensor being queried doesn't support a particular threshold.
91	E_SNSR_CAPABILITY_UNSUPPORTED	Internal CMM Error.
92	E_SNSR_SCANNING_DISABLED	Internal CMM Error.
93	E_SNSR_MAX_RETRIES	Internal CMM Error.
94	E_SNSR_TRIGGER_TYPE	Internal CMM Error.
95	E_SNSR_STATE	Internal CMM Error.
96	E_SNSR_EVENT_DEREGISTER	Internal CMM Error.
97	E_SNSR_SEL_EVENT_FUNCTION	Internal CMM Error.
98	E_SNSR_BASE_INDEX	Internal CMM Error.
99	E_SNSR_PRESENCE_DETECTED	Internal CMM Error.

Table 80. Error and Return Codes for the RPC Interface (Sheet 4 of 4)

Code #:	Error Code String	Error Code Description
100	E_SNMP_CMD_UNSUPPORTED	Internal CMM Error.
101	E_SNMP_ERROR	Internal CMM Error.
102	E_SNSR_VALUE_OUT_OF_RANGE	Internal CMM Error.
103	E_SNSR_AUTH_ERROR	Internal CMM Error.
104	E_WP_INITIALIZE_LIBS	Internal CMM Error.
105	E_WP_CFG_READ_ERROR	CMM config file may be corrupted.
106	E_WP_CFG_WRITE_ERROR	CMM config file may be corrupted.
107	E_WP_THRESHOLD_UNSUPPORTED	The sensor being queried doesn't support a particular threshold.
108	E_WP_INVALID_TARGET	The sensor doesn't support a "current" value. This happens when querying a current value on a discrete sensor type.
109	E_WP_INVALID_LOCATION	Not a valid -l location.
110	E_WP_INVALID_DATA_ITEM	Not a valid -d parameter.
111	E_WP_INVALID_SET_DATA	Not a valid -v parameter.
112	E_WP_CMD_UNSUPPORTED	Not a supported command.
113	E_WP_STANDBY_CMM	Can't execute this command on the standby CMM.
114	E_WP_I2C_ERROR	Internal CMM Error.
115	E_FT_SEM_GET_FAILURE	Internal CMM Error.
116	E_DRONE_NOT_FOUND	Internal CMM Error.
117	E_INTERNAL_ERROR	Internal CMM Error.
118	E_BPM_PWR_SUPPLY_NOT_PRESENT	Internal CMM Error
119	E_NEM_INTERNAL_FAILURE	Internal CMM Error.
120	E_WP_CMM_RESET	CMM Reset.
121	E_UPDATE_INPROGRESS	Firmware update in progress.
122	E_CLI_INVALID_GET_DATA_ITEM	Not a valid getdataitem.
123	E_CLI_INVALID_SET_DATA_ITEM	Not a valid setdataitem.
124	E_SNSR_UPDATE_INPROGRESS	Sensor update in progress.
125	E_WP_SNSR_EVN_DESCRIPTION_NOT_FOUND	Sensor event description not found.

Note: Error codes 100 and 101 will never be returned by the CMM when using RPC. If you receive these error codes, they are most likely being returned by the RPC client. Consult your RPC client documentation or code to find the description of these errors.

19.2.3 ChassisManagementApi() Threshold Response Format

The following table documents the format of ChassisManagementApi() queries that return data of type DATA_TYPE_ALL_THRESHOLDS.

Table 81. Threshold Response Formats

Dataitem	Return format	Example
thresholdsall	Data is returned in the THRESHOLDS_ALL structure as defined in cli_client.h. All structure fields are valid. If a particular threshold is not supported, the structure field will contain an empty string. Each supported and valid field is a null-terminated string. **Syntax:** [*Value*] [*Units*] /n /0	5.400 Volts 5.200 Volts 5.100 Volts 4.600 Volts 4.800 Volts 4.900 Volts
uppernonrecoverable uppercritical uppernoncritical lowernonrecoverable lowercritical lowernoncritical	Data is returned in the THRESHOLDS_ALL structure defined in cli_client.h. Only the structure field corresponding to the dataitem requested is valid. If a particular threshold is not supported, the structure field will contain an empty string. A valid field is a null-terminated string. **Syntax:** [*Value*] [*Units*] /n /0	5.160 Volts

19.2.4 ChassisManagementApi() String Response Format

The following table documents the format of ChassisManagementApi() queries that return data of type DATA_TYPE_STRING.

Table 82. String Response Formats (Sheet 1 of 5)

Dataitem	Return Format	Example
current	Null-terminated string showing the current value of a sensor. **Syntax:** Value [Units] /0	23.000 Celsius
Ethernet	Null-terminated string showing the orientation of the eth0 Ethernet port: **Syntax:** [*front/back*] /0	front
fanspeed	Null-terminated string giving the current CMM fan speed setting: **Syntax:** [*0 or 80 or 100*] /0	80

Table 82. String Response Formats (Sheet 2 of 5)

Dataitem	Return Format	Example
healthevents	List of human-readable health events. Lines are separated by linefeeds with a null-terminator at the end. "(null)" or "" if there are no healthevents Refer to Section 10, "Sensor Types" on page 101 **Syntax:** [*Critical/Major/Minor*] Event: [*Health String*] /n /0	Minor Event: +3.3 V Upper non-critical going high asserted
ListDataItems	List of available dataitems. Lines are separated by linefeeds and a null-terminator at the end. **Syntax:** [*Dataitem*] /n /0	presence listtargets listdataitems health healthevents sel snmpenable snmptrapcommunity snmptrapaddress1 snmptrapaddress2 snmptrapaddress3 snmptrapaddress4 snmptrapaddress5 redundancy powerstate
ListTargets	List of available targets. Targets represent the sensor data records (SDRs) for a particular component. Lines are separated by linefeeds with a null-terminator at the end. **Syntax:** [*Sensor Name*] /n /0	Brd Temp +1.5 V +2.5 V +3.3 V +5 V
ListLocations	List of available locations in the system. Except for the CMM, locations are displayed as integers according to the following: 1-14 = blade[1-14] 15 = Fantray1 16 = PEM1 17 = PEM2 CMM = CMM (only one CMM displayed)	CMM 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17
location	Null-terminated string containing the user-specified physical location of the CMM, 20 characters maximum. **Syntax:** [*Location String*] /0	Server room 3

Table 82. String Response Formats (Sheet 3 of 5)

Dataitem	Return Format	Example
powerstate	Human readable powerstate information containing the target blade powerstate information. Lines are separated by linefeeds with a null-terminator at the end. **Syntax:** Board is [*present or not present*] /n Board has [*not*] been powered up by the CMM /n Board is in [*active or offline*] mode. /n /0	Board is present Board has been powered up by the CMM Board is in active mode.
redundancy	Human-readable redundancy information containing the current CMM redundancy status. Lines are separated by linefeeds with a null-terminator at the end. **Syntax:** CMM 1: [*Present or Not Present*] ([*active or standby]*) [** or no star*] /n CMM 2: [*Present or Not Present*] ([*active or standby*) [** or no star*] /n * = The CMM you are logged into. /n /0	CMM 1: Present (active) * CMM 2: Not Present (standby) * = The CMM you are logged into.
slotinfo	Human-readable slot information, containing a list of System slots, Peripheral slots, Busless slots, and Occupied slots. If there are no slots in a particular category, "None" is reported. Lines are separated by linefeeds with a null-terminator at the end. Each colon is followed by one tab (for Peripheral and Busless slots) or two tabs (for System and Occupied slots) and a space-delimited list of slot numbers. **Syntax:** System Slot(s): [*None or slot numbers*] /n Peripheral Slot(s): [*None or slot numbers*] /n Busless (Switch) Slot(s): [*None or slot numbers*] /n Occupied Slot(s): [*None or slot numbers*] /n /0	System Slot(s): None Peripheral Slot(s): 2 3 4 5 6 7 8 13 14 15 16 17 18 19 20 21 Busless (Switch) Slot(s): 2 19 20 21 Occupied Slot(s): 2 5 21
snmptrapaddress[*1..5*]	Null-terminated string containing a dotted-quad IP address **Syntax:** XXX.XXX.XXX.XXX /0	10.10.240.81
snmptrapcommunity	Null-terminated string containing the snmptrapcommunity name **Syntax:** [*SNMP Trap Community Name String*] /0	publiccmm
snmptrapport	Null-terminated string showing the SNMP trap port. **Syntax:** [*port number*] /0	161

Table 82. String Response Formats (Sheet 4 of 5)

Dataitem	Return Format	Example
snmptrapversion	Null-terminated string showing the version of SNMP traps the CMM is currently set for. **Syntax:** [v1 *or* v3] /0	v3
unhealthylocations	Human-readable unhealthy blade, cmm and/or chassis information, containing a list of blades, cmms and/or chassis with a health status of Critical, Major, and Minor. If there are no items in a particular category, "None" is reported. Lines are separated by linefeeds with a null-terminator at the end. Each colon is followed by one space and a space-delimited list of blade numbers and the word "chassis" for chassis health conditions. **Syntax:** Critical: [*None or Blade# and/or chassis and/or cmm*] /n Major: [*None or Blade# and/or chassis and/or cmm*] /n Minor: [*None or Blade# and/or chassis and/or cmm*] /n /0	Critical: None Major: None Minor: 8 chassis
version	Null-terminated string containing the version of the CMM firmware. **Syntax:** X.X.X.XXXX /0	5.1.0.117
AdminState	"1:Unlocked" or "2:Locked"	Used to set or query the administrative state of PMS as a whole, an individual monitored process. A target of "PmsGlobal" will set the state of the PMS as a whole. A target of "PmsProc[#] will set the state of an individual process. Where "#" is the unique number for the process. AdminState is CMM-specific and is not synced between CMMs. It allows individual control of each CMMs adminstate and can be set on either active or standby CMM.
RecoveryAction	"1:No Action", "2:Process Restart", "3:Failover and Restart", or "4:Failover and Reboot"	Used to set or query the recovery action of a PMS monitored process. This is only valid for a target of "PmsProc[#]. Where "#" is the unique number for the process.

Table 82. String Response Formats (Sheet 5 of 5)

Dataitem	Return Format	Example
EscalationAction	"1:No Action", "2:Failover and Reboot"	Used to set or query the process restart escalation action. This is only valid for a target of "PmsProc[#]. Where "#" is the unique number for the process.
ProcessName	"<Process_Name> <Command_Line_Arguments>"	Used to query the process name and associated command line arguments for a monitored process. A target of "PmsProc[#] will retrieve the name of an individual process. Where "#" is the unique number for the process.
OpState	"1:Enabled", "2:Disabled"	Used to query the operational state of a monitored process. An operational state of disabled indicates that the process has failed and cannot be recovered. This is only valid for a target of "PmsProc[#]. Where "#" is the unique number for the process.

19.2.5 ChassisManagementApi() Integer Response Format

The following table documents the format of ChassisManagementApi() queries that return data of type DATA_TYPE_INT.

Table 83. Integer Response Formats

Dataitem	Return format	Example
alarmcutoff	Integer value indicating alarm cutoff status: 0 = alarm not cut off 1 = alarm cut off	0
alarmtimeout	Integer value indicating the alarm cutoff time-out in minutes.	5
health	Integer value corresponding to the health of the location queried: 0 = OK 1 = Minor 2 = Major 3 = Critical	2
presence	Integer value corresponding to the absence or presence of the location queried: 0 = Not present 1 = Present **NOTE:** If a blade is not present, ChassisManagementApi() returns E_BLADE_NOT_PRESENT.	1
snmpenable	Integer value indicating SNMP status: 0 = disabled 1 = enabled	0

19.2.6 FRU String Response Format

Querying an individual FRU field returns a null terminated string with a single linefeed.

Table 84. FRU Data Items String Response Format

Data Item	Description
all	Null-terminated string containing all FRU information for the location.
boardall	Null-terminated string containing all board area FRU information for the location.
boarddescription	Null-terminated string containing the description field in the FRU board area for the location.
boardmanufacturer	Null-terminated string containing the manufacturer field in the FRU board area for the location.
boardpartnumber	Null-terminated string containing the part number field in the FRU board area for the location.
boardserialnumber	Null-terminated string containing the serial number field in the FRU board area for the location.
boardmanufacturedatetime	Null-terminated string containing the manufacture date and time field in the FRU board area for the location.
productall	Null-terminated string containing the product area FRU information for the location.
productdescription	Null-terminated string containing the description field in the FRU product area for the location.
productmanufacturer	Null-terminated string containing the manufacturer field in the FRU product area for the location.
productmodel	Null-terminated string containing the model field in the FRU product area for the location.
productpartnumber	Null-terminated string containing the part number field in the FRU product area for the location.
productserialnumber	Null-terminated string containing the serial number field in the FRU product area for the location.
productrevision	Null-terminated string containing the revision field in the FRU product area for the location.
productmanufacturedatetime	Null-terminated string containing the manufacturer date and time field in the FRU product area for the location.
chassisall	Null-terminated string containing all chassis area FRU information for the location.
chassispartnumber	Null-terminated string containing the part number field in the FRU chassis area for the location.
chassisserialnumber	Null-terminated string containing the serial number field in the FRU chassis area for the location.
chassislocation	Null-terminated string containing the location field in the FRU chassis area for the location.
chassistype	Null-terminated string containing the type field in the FRU chassis area for the location.
listdataitems	Null-terminated string containing the list of all of the FRU dataitems that can be queried for the FRU target.

19.3 RPC Sample Code

Sample code for interfacing with the CMM through RPC is available in the file cli_client.c. The sample code compiles into a command-line executable for use with Linux or a .o file for use with VxWorks. To select the appropriate target, remove the comment from the appropriate #define in the source code.

The sample code first authenticates to the CMM through GetAuthCapability(). When the authentication is successful, the user's command-line arguments (for Linux) or calling parameters (for VxWorks) are passed to the CMM through ChassisManagementApi(). The return code is then checked and the result is printed to the console.

19.4 RPC Usage Examples

Table 85 presents examples of using RPC calls to get and set fields on the CMM. Data returned by RPC calls are returned in the *ppvbuffer* and *uReturnType* parameters to ChassisManagementApi().

Table 85. RPC Usage Examples (Sheet 1 of 3)

Example	ChassisManagementApi() [in] Parameters	ChassisManagementApi() [out] Parameters
Get the chassis temperature.	*pszCMMHost*: localhost *uCmdCode*: CMD_GET *pszLocation*: Chassis *pszTarget*: TempSensorName *pszDataItem*: current	*uReturnType*: DATA_TYPE_STRING *ppvbuffer*: A null-terminated string of the format: Value [Units]
Get the fan tray presence.	*pszCMMHost*: localhost *uCmdCode*: CMD_GET *pszLocation*: fantray1..3 *pszTarget*: NA *pszDataItem*: presence	*uReturnType*: DATA_TYPE_INT *ppvbuffer*: Integer value indicating presence 1 = Present 0 = Not Present
Get the CPU temperature of blade 5.	*pszCMMHost*: localhost *uCmdCode*: CMD_GET *pszLocation*: blade5 *pszTarget*: CPUTempSensorName *pszDataItem*: current	*uReturnType*: DATA_TYPE_STRING *ppvbuffer*: A null-terminated string of the format: Value [Units]
Determine if a certain blade is present.	*pszCMMHost*: localhost *uCmdCode*: CMD_GET *pszLocation*: blade[1-n] *pszDataItem*: presence	*uReturnType*: DATA_TYPE_INT *ppvbuffer*: Present The call to ChassisManagementApi() returns E_BLADE_NOT_PRESENT if the selected blade is not present.
Get all thresholds for the +3.3 V sensor on blade 2.	*pszCMMHost*: localhost *uCmdCode*: CMD_GET *pszLocation*: blade2 *pszTarget*: 3.3vSensorName *pszDataItem*: ThresholdsAll	*uReturnType*: DATA_TYPE_ALL_THRESHOLDS *ppvbuffer*: A THRESHOLDS_ALL structure as defined in cli_client.h

Table 85. RPC Usage Examples (Sheet 2 of 3)

Example	ChassisManagementApi() [in] Parameters	ChassisManagementApi() [out] Parameters
Get the overall system health.	*pszCMMHost*: localhost *uCmdCode*: CMD_GET *pszLocation*: system *pszDataItem*: health	*uReturnType*: DATA_TYPE_INT *ppvbuffer*: Integer value denoting health state 0 = OK 1 = Minor 2 = Major 3 = Critical
Get a list of blades with problems.	*pszCMMHost*: localhost *uCmdCode*: CMD_GET *pszLocation*: system *pszDataItem*: unhealthylocations	*uReturnType*: DATA_TYPE_STRING *ppvbuffer*: List of all blades with problems
Get the temp1 sensor's health on blade 5.	*pszCMMHost*: localhost *uCmdCode*: CMD_GET *pszLocation*: blade5 *pszTarget*: Temp1SensorName *pszDataItem*: health	*uReturnType*: DATA_TYPE_INT *ppvbuffer*: Integer value denoting health state 0 = OK 1 = Minor 2 = Major 3 = Critical
Get the CMM's overall health.	*pszCMMHost*: localhost *uCmdCode*: CMD_GET *pszLocation*: CMM *pszDataItem*: health	*uReturnType*: DATA_TYPE_INT *ppvbuffer*: Integer value denoting health state 0 = OK 1 = Minor 2 = Major 3 = Critical
Get a blade's overall health.	*pszCMMHost*: localhost *uCmdCode* CMD_GET *pszLocation*: blade[1..n] *pszDataItem*: health	*uReturnType*: DATA_TYPE_INT *ppvbuffer*: Integer value denoting health state 0 = OK 1 = Minor 2 = Major 3 = Critical
Get the version of software on the CMM.	*pszCMMHost*: localhost *uCmdCode*: CMD_GET *pszLocation*: CMM *pszDataItem*: version	*uReturnType*: DATA_TYPE_STRING *ppvbuffer*: A human-readable null-terminated version string.
Power off one of the blades.	*pszCMMHost*: localhost *uCmdCode*: CMD_SET *pszLocation*: blade[1-19] *pszDataItem*: powerstate *pszSetData*: poweroff	*uReturnType*: not used *ppvbuffer*: not used The return code from ChassisManagementApi() indicates success or failure.
Power on one of the blades.	*pszCMMHost*: localhost *uCmdCode*: CMD_SET *pszLocation*: blade[1-19] *pszDataItem*: powerstate *pszSetData*: poweron	*uReturnType*: not used *ppvbuffer*: not used The return code from ChassisManagementApi() indicates success or failure.

Table 85. RPC Usage Examples (Sheet 3 of 3)

Example	ChassisManagementApi() [in] Parameters	ChassisManagementApi() [out] Parameters
Reset a blade.	*pszCMMHost*: localhost *uCmdCode*: CMD_SET *pszLocation*: blade[1-19] *pszDataItem*: powerstate *pszSetData*: reset	*uReturnType*: not used *ppvbuffer*: not used The return code from ChassisManagementApi() indicates success or failure.
Determine what sensors are on blade 3.	*pszCMMHost*: localhost *uCmdCode*: CMD_GET *pszLocation*: blade3 *pszDataItem*: ListTargets	*uReturnType*: DATA_TYPE_STRING *ppvbuffer*: A list of sensor names as defined in the SDR.
Determine what may be queried or set on a blade.	*pszCMMHost*: localhost *uCmdCode*: CMD_GET *pszLocation*: blade3 *pszDataItem*: ListDataItems	*uReturnType*: DATA_TYPE_STRING *ppvbuffer:* A list of commands to be used as data items.
Determine what may be queried on the blade4 +3.3 V sensor.	*pszCMMHost*: localhost *uCmdCode*: CMD_GET *pszLocation*: blade4 *pszTarget*: +3.3SensorName *pszDataItem*: ListDataItems	*uReturnType*: DATA_TYPE_STRING *ppvbuffer*: A list of commands to be used as data items.
Enable the SNMP Traps.	*pszCMMHost*: localhost *uCmdCode*: CMD_SET *pszLocation*: chassis *pszDataItem*: SNMPEnable *pszSetData*: enable	*uReturnType*: not used *ppvbuffer*: not used The return code from ChassisManagementApi() indicates success or failure.
Set the SNMP Target.	*pszCMMHost*: localhost *uCmdCode*: CMD_SET *pszLocation*: chassis *pszDataItem*: SNMPTrapAddress[1-5] *pszSetData*: 134.134.100.34	*uReturnType*: not used *ppvbuffer*: not used The return code from ChassisManagementApi() indicates success or failure.
Set the SNMP Community.	*pszCMMHost*: localhost *uCmdCode*: CMD_SET *pszLocation*: chassis *pszDataItem*: SNMPCommunity *pszSetData*: public	*uReturnType*: not used *ppvbuffer*: not used The return code from ChassisManagementApi() indicates success or failure.
Set the Telco Alarm on.	*pszCMMHost*: localhost *uCmdCode*: CMD_SET *pszLocation*: CMM *pszDataItem*: TelcoAlarm *pszSetData*: 1	*uReturnType*: not used *ppvbuffer*: not used The return code from ChassisManagementApi() indicates success or failure.
Light Major LED on the CMM.	*pszCMMHost*: localhost *uCmdCode*: CMD_SET *pszLocation*: CMM *pszDataItem*: MajorLED *pszSetData*: 1	*uReturnType*: not used *ppvbuffer*: not used The return code from ChassisManagementApi() indicates success or failure.

RMCP 20

The RMCP (Remote Management Control Protocol) section of the Alert Standard Format Specification version 2.0 section 3.2.1 allocates two UDP ports for the RMCP server, ports 623 (the Primary RMCP port) and 664 (the Secondary RMCP or Secure port). This implementation of the RMCP server listens for the RMCP packet from the Ethernet interface on only on port 623. When the RMCP packet arrives, the RMCP server decodes the RMCP message. It checks the RMCP packet:

- If it is an invalid version or not valid IPMI RMCP packet, it will drop the packet
- If the message is RMCP ping, it returns the RMCP pong message.

If the session is invalid, not available, duplicated, or out of order, or slots full, it sends the return RMCP error message to the RMCP client.

If the RMCP packet is a valid, it is forwarded on through the CMM interface to the appropriate destination. If the CMM receives an appropriate IPMI response from the RMCP destination, the CMM will return the IPMI response in a properly formatted RMCP message back to the RMCP client over Ethernet.

20.1 RMCP References

Intelligent Platform Management Specification V1.5 Revision 1.1

Alert Standard Format (ASF) Specification V2.0 (DMTF document DSP0136)

PICMG 3.0 Advanced Telecommunications Computing Architecture (AdvancedTCA) vD1.0 December 18,2002

20.2 RMCP Modes

The RMCP server on the CMM may be configured to operate in one of two different modes. The Set RMCP Mode command is used to set the mode. The RMCP modes are shown in Table 86:

Table 86. RMCP Modes

RMCP Mode	Description
Full Access	The RMCP feature functionality is fully operational and a RMCP client can initiate a session regardless of the host /server power state and operating system health.This is the default system setting.
Disabled	Disables the RMCP functionality. In this mode, the RMCP server discards the requests received over the network.

20.3 RMCP User Privilege Levels

There are five privilege levels defined in the IPMI1.5 spec.

1. Callback level
2. User level
3. Operator level
4. Administrator level
5. OEM Proprietary level

Callback Level has the most restricted privileges, and OEM Proprietary Level has the least restricted privilege. The RMCP server provides the user and password support associated with these five privilege levels.

A user level requestor is not allowed to issue a request with a higher privilege level IPMI command. For example, a user level requestor cannot issue a request such as *Clear SEL* that needs Operator privileges.

The user name, password, and privilege level can be set using the following IPMI commands:

- *Set User Name (See IPMI specification V1.5 section 18.25 for details)*
- *Set User Password (See IPMI specification V1.5 section 18.27 for details)*
- *Set User Access (See IPMI specification V1.5 section 18.23 for details)*

Currently only user ***root*** is supported.

20.4 RMCP Discovery

According to the IPMI 1.5 specification, RMCP client uses Ping/Pong messages to discover the existence of an RMCP server. To support the discovery mechanism, the RMCP server supports:

- RMCP/ASF Presence Ping Message
- RMCP/ASF Pong message

For the Pong message, the CMM will convey:

- IANA Enterprise number.
- Supported Entities: IPMI supported and ASF version 1.0

20.5 RMCP Session Activation

The CMM will support multiple RMCP sessions.

1. To activate a session, the RMCP client sends a *Get Channel Authentication Capabilities* command packet with Authentication Type = none (in clear). The response packet will contain information regarding which type of challenge/response authentication is available.
2. The RMCP client requests a session challenge by issuing a *Get Session Challenge* request, also with Authentication Type = none. The request contains information indicating what type of authentication type the RMCP client wants to use. This must be one of the supported types

returned by the *Get Channel Authentication Capabilities* command. The response packet will contain a challenge string and a Session ID.

3. The RMCP client activates the session by issuing an *Activate Session* request. The Activate Session packet is typically authenticated. For message-digest algorithms, the packet includes a signature (AuthCode) that is a hash of the challenge, the Session ID, the password, and the message data using one of the supported algorithms from the *Get Channel Authentication Capabilities* command. The client also sets the initial value for the outbound sequence number that it wants the RMCP server to use for packets it sends to the console.
4. The RMCP server returns a response confirming that the Session has been successfully activated. It also returns the Session ID to be used for the remainder of the session, and the initial inbound session sequence number that it wants the RMCP client to use for subsequent messages it sends to the RMCP server for that session. The *Activate Session* response is also authenticated (signed) in the same manner as the request. This allows the RMCP client to validate that it has a correct Session ID. Note that IPMI does not support switching authentication algorithms 'mid stream'. The algorithm used with the *Activate Session* command is the algorithm that will be used for subsequent authenticated messages for the session. The exception to this is that the 'none' authentication type is allowed if options such as 'Per-Message Authentication' and/or 'User Authentication' are disabled.
5. At this point the session is active. The RMCP client can send a *Close Session* request to terminate an active session. The RMCP server will return the *Close Session response* to acknowledge the client request.

During a **session-active** phase:

- Administrator-level requests must be sent as secured (authenticated) messages using the authentication type that was requested in the *Activate RMCP Session* message request.
- Authentication type cannot be changed. Secured messages with authentication type other than that requested in the *Activate RMCP Session* message request will be silently discarded.
- The RMCP server sends the response using the same authentication type that was used in the request.
- The RMCP server implements a session-active phase expiration timer. The server will terminate the session if it does not receive any valid secured message request for a time since last valid secured message request was received. The client, in this case, should reestablish the session initiating with the *Activate RMCP Session* message-request. The following table shows the RMCP Session Timers.

Table 87. RMCP Session Timers

RMCP Session Phase	Time-out Interval
Activation	120 Seconds
Active Session	120 Seconds

20.6 RMCP Port Numbers

RMCP messages are sent via UDP datagrams over Ethernet. The RMCP server communicates on management port 623 (26Fh) for handling RMCP requests.

A secondary port 664 (298h) is used when encryption is necessary.

20.7 IPMB Slave Addresses

In the current RMCP server implementation in the BMC, embedded 'IPMI message' within a RMCP message is bundled with IPMB protocol. The 'slave address' required by this protocol should be set to 20h to address the BMC. The RMCP client may use any of the address shown in the range specified below as its slave Address. Only EVEN values are allowed. i.e., the least significant bit of the slave address is always '0'.

Table 88. RMCP Slave Addresses

Nodes	Value
RMCP Server Slave Address	20h
RMCP Client Slave Address	C0h-CEh

20.8 CMM RMCP Configuration

To communicate with the CMM RMCP server a RMCP client must have the following information:

- the RMCP server's IP address
- the User Name, which is initially set to ***root***
- the User Password, which is initially set to ***cmmrootpass***
- and set the RMCP mode to "Full Access"

All of these were discussed in the CLI section above.

20.9 IPMI Commands Supported by CMM RMCP

The following IPMI commands are supported via RMCP by the CMM. To configure privileges for the commands see Section 20.10, “Configuring IPMI Command Privileges” on page 196 below.

Table 89. IPMI Commands Supported by CMM RMCP (Sheet 1 of 3)

Command Type	Defined In	Command
IPMI Device Global	IPMI 1.5 Spec	Get Device ID
		Get Self Test Results
BMC Device and Messaging Commands	IPMI 1.5 Spec	Set BMC Global Enables
		Get BMC Global Enables
		Clear Message Flags
		Get Message Flags
		Get Message
		Send Message
		Get Channel Authentication Capabilities
		Get Session Challenge
		Activate Session
		Set Session Privilege Level
		Close Session
		Get Session Info
		Get AuthCode
		Set Channel Access
		Get Channel Access
		Get Channel Info
		Set User Access
		Get User Access
		Set User Name
		Get User Name
		Set User Password
Chassis Device Commands	IPMI 1.5 Spec	Get Chassis Capabilities
		Get Chassis Status
		Chassis Control
		Get POH Counter
Event Commands	IPMI 1.5 Spec	Get Event Receiver
		Set Event Receiver

Table 89. IPMI Commands Supported by CMM RMCP (Sheet 2 of 3)

Command Type	Defined In	Command
Sensor Device Commands	IPMI 1.5 Spec	Get Device SDR Info
		Get Device SDR
		Reserve Device SDR Repository
		Get Sensor Hysteresis
		Get Sensor Threshold
		Get Sensor Event Enable
		Re-arm Sensor Events
		Get Sensor Event Status
		Get Sensor Reading
FRU Device Commands	IPMI 1.5 Spec	Get FRU Inventory Area Info
		Read FRU Data
		Write FRU Data
SDR Repository Commands	IPMI 1.5 Spec	Get SDR Repository Info
		Reserve SDR Repository
		Get SDR
		Partial Add SDR
		Delete SDR
		Clear SDR Repository
SEL Device Commands	IPMI 1.5 Spec	Get SEL Info
		Get SEL Allocation Info
		Reserve SEL
		Get SEL Entry
		Add SEL Entry
		Clear SEL
		Get SEL Time
		Set SEL Time
LAN Device Commands	IPMI 1.5 Spec	Set LAN Configuration Parameters
		Get LAN Configuration Parameters

Table 89. IPMI Commands Supported by CMM RMCP (Sheet 3 of 3)

Command Type	Defined In	Command
AdvancedTCA™	PICMG 3.0 Spec	Get PICMG Properties
		Get Address Info
		Get Shelf Address Info
		Set Shelf Address Info
		FRU Control
		Get FRU LED Properties
		Get LED Color Capabilities
		Set FRU LED State
		Get FRU LED State
		Set IPMB State
		Set FRU Activation Policy
		Get FRU Activation Policy
		Set FRU Activation
		Get Device Locator Record ID
		Get Port State
		Set Power Level
		Get Power Level

20.10 Configuring IPMI Command Privileges

The cmdPrivillege.ini file located in the /etc directory of the CMM allows the configuration of privileges allowed for each IPMI command on the CMM.

Note: The CMM or cmd_hand must be restarted for the changes to the cmdPrivillege.ini to take effect.

The format of the file is as follows:

```
[IPMI]

NumNetFunc=23

NetFunc[Net Function1]=[Net Function Name 1]

NetFunc[Net FunctionN]=[Net Function Name N]


[Net Function Name 1]

NumCMD=Y

[Command Byte 1]=[Privelege Level]

[Command Byte Y-1]=[Privelege Level]
```

Where:

[Net Function1..n]: TheIPMI Net Function in hex of the corresponding command.

[Net Function Name 1..n]: The name used to identify its corresponding subsection.

Y: The max number of commands defined in the subsection plus 1.

[Command Bye 1..(Y-1)]: The command byte of the Net Function defined in this section.

Privilege Level: The privilege given to that command byte defined as follows:

- C: Callback, callback only
- U: User, browse only
- O: Operator, can change state
- A: Admin, can change transport
- D: Disabled, no access

20.10.1 Sample cmdPrivillege.ini file

```
[IPMI]

NumNetFunc=23

NetFunc00=Chassis

NetFunc02=Bridge


[Chassis]

NumCMD=16

00=U

01=U

02=O

03=O

04=O

05=A
```

20.11 Completion Codes for the RMCP Messages

The following table lists the completion codes for RMCP Messages. Please refer to the IPMI 1.5 Specification for more detail.

Table 90. RMCP Message Completion Codes

Code	Description
00	Success
C0	Busy
C1	Invalid Command
C2	Command invalid for a given LUN
C7	Request data length invalid
C8	Requested data field length limit exceeded. (too long)
C9	Requested Offset (in the data) Out of Range
CB	Not Found
CC	Invalid field in the Request
CD	Illegal Command
10	RMCP Session/User Authentication Failed
11	RMCP Session Active
12	RMCP Session in Authentication Phase

Command and Error Logging 21

21.1 Command Logging

All **CMMSET** commands from all of the CMM interfaces (CLI, RPC, and SNMP) are logged by the CMM in the file */var/log/user.log*. The size of the *user.log* file is 500 KBytes. When the command log becomes full, the log file is compressed and archived using gzip, then stored in */home/log*. The filename format for the log files is *user.log.N.gz*, where N is the number of the log file from 1 to 4. The CMM will keep up to four archives of log files. If the log file becomes full and there are already four existing command log archives, the oldest archive will be deleted to make room for the newest archive.

Note: Archived files should **NEVER** be decompressed on the CMM because the resulting prolonged flash file writing could disrupt normal CMM operation and behavior. The files should be transferred and decompressed on a different machine. Files can be decompressed by any application that supports the decompression of gzip (.gz) file types.

Note: The /home/log directory should not be deleted or change. the CMM requires that directory exist to log errors.

21.2 Error Logging

The error.log and debug.log files are archived to maintain error logging in the event either log gets full and to prevent any loss of log data. This information is useful for technical support personnel.

21.2.1 Error.log File

Error logging information for the CMM is logged in the file */home/log/error.log*. The size of the *error.log* file is 500 KBytes. When *error.log* becomes full, the log file is compressed and archived using gzip, then stored in */home/log*. The filename format for the log files is *error.log.N.gz*, where N is the number of the log file from 1 to 4. The CMM will keep up to four archives of log files. If the log file becomes full and there are already four existing *error.log* archives, the oldest archive will be deleted to make room for the newest archive.

21.2.2 Debug.log File

Debug information for the CMM is logged in the file */var/debug.log*. The size of the *debug.log* is 200 KBytes. When the *debug.log* becomes full, the log file is compressed and archived using gzip, then stored in */var/log*. The filename format for the log files is *debug.log.N.gz*, where N is the number of the log file from 1 to 4. The CMM will keep up to four archives of log files. If the log file becomes full and there are already four existing *debug.log* archives, the oldest archive will be deleted to make room for the newest archive.

21.3 Cmmdump Utility

The cmmdump utility is a script that dumps the current system state of the CMM, the value of several configuration and log files, and the output of several cmmget commands. This utility is currently designed to be executed from a command prompt on the CMM. The output is sent to stdout and can be re-directed to a file to log the data.

The purpose of cmmdump is to capture the debug information from the CMM system. This debug information can be used by support personnel to debug CMM and system issues. It will log all important system information that can be helpful to localize a problem.

Application Hosting 22

The CMM allows applications to be hosted and run locally. This is useful as a method for adding small custom management utilities to the CMM.

22.1 System Details

The CMM runs a customized version of embedded BlueCat* Linux* 4.0 on an Intel® 80321 with Intel® XScale® microarchitecture. Development support for BlueCat Linux is available on the web at: http://www.lynuxworks.com.

22.2 Startup and Shutdown Scripts

The CMM is capable of running user created scripts automatically on boot up or shutdown.This can be accomplished by editing the /etc/rc.d/init.d/userScripts file with a text editor. This is a standard shell file just like /etc/rc.d/init.d/cmm. By default it just echos "Starting user services" and "Shutting down user services". The user can add scripts or what they want to the file. The Ramdisk version of /etc/inittab calls "/etc/rc.d/init.d/userScripts start" right after its starts the cmm software. When you type reboot, it calls "/etc/rc.d/init.d/userScripts stop”.

22.3 System Resources Available to User Applications

Since the CMM has firmware of its own running at all times, user applications must adhere to certain resource and directory constraints to avoid disrupting the CMM’s operation. Specifically, restrictions are placed on an application’s consumption of file system storage space, RAM, and interrupts. Exceeding these guidelines may interfere with proper operation of the CMM.

22.3.1 File System Storage Constraints

The following file system locations are available for storage by user applications. Storing files in locations other than those explicitly listed here is not supported.

22.3.1.1 Flash Storage Locations

Applications should not perform excessive amounts of flash file I/O at runtime because this will impair performance of the CMM.

/etc - Useful for storing configuration files. Do not add more than 300 KBytes to /etc.

/etc/rc.d/init.d - Useful for storing startup scripts. Do not add more than 300 KBytes to /etc.

/home - Useful for storing application binaries. Do not add more than 8 Mbyes to /home.

22.3.1.2 Flash Memory Map

Below is the list of images on CMM flash with flash address ranges and the size of the images.

Table 91. Flash #1

Image Name	Flash Address Range	Image Size
Backup RedBoot image	0xF0000000 - 0xF003FFFF	256KB
RedBoot image	0xF0040000 - 0xF007FFFF	256KB
FPGA image	0xF0080000 - 0xF00BFFFF	256KB
Backup FPGA image	0xF00C0000 - 0xF00FFFFF	256 KB
OS image	0xF0100000 - 0xF0FFFFFF	153060KB

Table 92. Flash #2

Image Name	Flash Address Range	Image Size
OS image	0xF1000000 - 0xF1FFFFFF	16MB

Table 93. Flash #3

Image Name	Flash Address Range	Image Size
FFS2 (/home partition)	0xF2000000 - 0xF2FFFFFF	16MB

Table 94. Flash #4

Image Name	Flash Address Range	Image Size
FFS (/etc partition)	0xF3000000 - 0xF37FFFFF	8MB
Reserved	0xF3800000 - 0xF3FBFFFF	8MB – 256KB
Configuration area	0xF3FC0000 – 0xF3FDFFFF	128KB
FIS directory	0xF3FE0000 - 0xF3FFFFFF	128KB

22.3.1.3 RAM-Disk Storage Locations

Caution: Files in this location are stored in RAM and will be lost during CMM reboots.

Due to the constraints of writing to flash memory, larger file operations such as decompressing an archive should be performed on RAM-disk in the following directory:

/usr/local/cmm/temp - Useful for storing temporary files. Applications should make a subdirectory for use with their temporary files. Do not add more than 5 MBytes to this location.

22.3.2 RAM Constraints

Up to 32 MBytes of RAM are available for user applications. RAM usage at runtime, on average, should not exceed this quantity.

22.3.3 Interrupt Constraints

User applications should not use interrupts. All interrupts are reserved for use by the CMM firmware.

22.4 RAM Disk Directory Structure

The following directories are stored on a RAM-disk. These directories store information critical to the CMM and are size constrained. These directories should not be modified in any way.

/usr – Executables including the CMM executables (/usr/local/cmm/bin). Also contains some libraries and temp files.

/lib - Shared library files and drivers

/bin – Linux binary files like ping, mount

/dev – Device driver files

/proc – Special linux area that contains info about memory, drivers, etc.

/root – Empty root user area.

/sbin – Binary files like ifconfig, route

/tmp – Empty

/var – Temp location for log files and other temp files.

Updating CMM Software 23

When new CMM updates are available, they are packaged in a zip file and posted to the Intel developer web site located at:

http://www.intel.com/design/network/products/cbp/atca/mpcmm0001.htm

Please follow the instructions provided with the software update package to perform the update.

The CMM is capable of having its firmware and critical system files updated when new update packages become available. The update process allows these updates to occur remotely without losing the active CMM in a redundant configuration.

23.1 Key Features of the Firmware Update Process

- Updates can be done remotely over the front or back Ethernet ports on the CMM
- Current CMM configuration data is preserved across the update
- Critical CMM data such as the SEL and command history is preserved across an update
- Redundant CMMs can be updated without interrupting management of the chassis
- Update files are verified and checked for corruption
- Update components have associated version numbers
- Update events are logged to the SEL
- Updates can be triggered using the CLI
- Update packages can be located locally on the CMM or pulled from a mounted NFS or remote FTP server

23.2 Update Process Architecture

The update process consists of the following components:

- User Client – The client is used to trigger the update process, and can be located anywhere on the network.
- Update Package – The update package contains the new software components and other files necessary for the update. The update package can be pulled from a remote server, or be pushed locally onto the CMM.
- CMM Update Request Handler – The update request handler is CMM software that processes incoming update requests from and responses to the various interfaces provided by the CMM.
- CMM Update Script – The update script is stored on the CMM and is used to process the contents of the update package and perform the update.
- Update Package Server (Optional) – The update package server can be used to store update packages remotely from the CMM. This can be an FTP or NFS server.

23.3 Critical Software Update Files and Directories

The following table is a list of files and directories important to the software update process.

Table 95. List of Critical Software Update Files and Directories

File or Directory Name:	Description:
/usr/local/cmm/temp	Mount point for ramdisk
/usr/local/cmm/temp/update/package	Directory into which the update package is copied and unzipped. The update process will delete and recreate this directory. If it is desired to copy the update package to the CMM before updating, then put the package in /usr/local/cmm/temp/update. **NOTE:** Do NOT place the update files into the /usr/local/cmm/temp/update/package directory. The update process uses this directory and will erase and overwrite files placed here.
/usr/local/cmm/temp/update/etc	Backup copy of /etc is located here
[package file].zip	Zip file containing update package files
[package file].md5	MD5 checksum of the [package file].zip file
/etc/versions	At the end of an update, this file contains a list of the package version, and each component version
/etc/versions.<version>	The update process will move the "old" /etc/versions file by appending a file extension to it which is the package version string - i.e., .. /etc/versions.5.1.0.0117
/home/update/scripts/K*	These optional scripts are supplied by the user. The update process will execute them just prior to shutting down the cmm applications. The primary function of these scripts is to allow the user a method of shutting down any user process which are using the flash file systems.
/home/update/scripts/S*	These optional scripts are supplied by the user. The update process will execute these scripts just before processing the save list - which copies files and directories from /usr/local/cmm/temp/update/etc to /etc
/home/update/backup	If the update direction is "backward", then files and directories that are priority 2 will be copied here instead of /etc
/usr/local/cmm/temp/update/update.log	Log file to which output is sent while the update process is operating - this is on a ramdisk.
/etc/cmm/update.log	The log file is copied when the update process completes successfully.

23.4 Update Package

The update package is available at:

http://www.intel.com/design/network/products/cbp/atca/mpcmm0001.htm

It contains the components listed in Table 96 on the following page.

Table 96. Contents of the Update Package

File Name	Description
CMM_RB.bin	Redboot image to be stored in flash
CMM_FPGA.bin	FPGA image
CMM_FFS.bin	/etc image
CMM_OS.bin	OS image
README	Text file containing release notes for the update package
Update_Metadata	File containing info on the update package and how it should be installed on the CMM
Utilities	Sub-directory containing any utilities that might be required for the install.
validationfile	File containing MD5 checksums for the files and scripts present in the update package.
saveList	List of configuration files in /etc to be preserved across software updates.
Other	Other components such as FRU files that may be necessary for the update.
version_history	List of all versions of firmware available for the platform in sequential order, with the newest versions at the end of the list.
mpcmm0001.mib	MIB file for SNMP.

The update package can be placed locally on the CMM in the /usr/local/cmm/temp directory, or it can reside on a server on the network.

The files and directories that make up the Update Package are packaged and delivered as a .zip file. Associated with the .zip file is a file containing the md5 sum of the .zip file. For example:

- [*PackageFileName]*.zip file containing all files and subdirectories of the update package
- [*PackageFileName]*.md5 file containing md5 checksum of PackageFileName.zip

Arguments for the location of the update package will be given in the CLI command, which can be used to point to a remote server or a local directory. The update script will then remove the /usr/local/cmm/temp/update/package directory and recreate it with the new package files.

Note: If an NFS server is mounted to the CMM, the argument in the update script will be similar to a file located locally on the CMM.

If the package fails to copy or transfer to /usr/local/cmm/temp/update/package, then the update process will terminate.

23.4.1 Update Package File Validation

The first level of update package validation is checking that the md5sum of the zip file matches the md5sum in the [*PackageFileName*].md5 file of the update package.

Update packages are validated during the update process by computing the md5sum of each file and comparing against the checksums stored in the checksums stored in the validation_file included with the update package.

The update package is also checked to ensure it is valid for the platform where it is being used. For example, an update package with firmware for a CompactPCI* CMM will not work on an AdvancedTCA* CMM.

23.4.2 Update Firmware Package Version

The firmware update package has a firmware version associated with the entire package. This firmware version is the same version that can be retrieved using the CLI command:

```
cmmget -l cmm -d version
```

To determine if the update is a new, old or same version, the update package will contain a version_history file which contains a list of all software builds that have occurred, listed in sequential order. Newer builds are at the bottom of the list.

23.4.3 Component Versioning

The components contained in the update package will contain versioning information that will allow the update process to ensure the updated components installed on the CMM are all compatible at the end of an update and match those contained in the update package.

If an update package contains a component version that is already installed on the CMM, then that component will not be updated. A method is also provided to force all components to update regardless of the version installed.

Version info for files installed on the CMM can be obtained through the various interfaces and is stored in the /etc/versions file. Version info for the files in the update package will also be available.

The CMM checks the md5sum of /etc/versions during boot. In the event that the checksum fails, the CMM applications will fail to start and a message will appear on the console as follows “Startup of CMM applications has terminated due to detection of an inconsistent software load.” This functionality exists to prevent a CMM with an incorrect /etc/versions file, which may have occurred during an incomplete update, from synchronizing incorrectly with the other CMM.

23.5 saveList and Data Preservation

The update process will preserve user configuration data as well as critical system configuration information across updates. These files are located in the /etc directory. The list of files or directories to be preserved across updates is contained in the software update package and is called SaveList. Each entry in the saveList will contain a directory or file to be saved as well as a priority assigned to it. Priority for the file can be either a 1 or a 2, and is used to determine if certain files or directories should be saved when updating to an earlier firmware version.

Priority 1 is assigned to files or directories that should be saved in all cases of an update, including going forward or backward in firmware version. Priority 1 files are considered files critical to the CMM operation and access to the CMM, such as Ethernet configuration.

Priority 2 is assigned to files or directories which should not be saved if an update is being performed to an earlier version of firmware.

During an update, the CMM will copy over the current files in /etc to ram disk in /usr/local/cmm/temp/update/etc. The CMM then uses the saveList file to determine which configuration files or directories to copy back into the new /etc partition. When updating to a previous version of firmware, the CMM makes a backup of any files designated priority 2 in the save list to flash in the /home/update/backup/etc directory.

Table 97. SaveList Items and Their Priorities

File	Priority
/etc/passwd	1
/etc/shadow	1
/etc/cmm/sel_*	1
/etc/cmm/cmm_sel	1
/etc/snmp*.conf	1
/etc/ifcfg-eth*	1
/etc/HOSTNAME	1
/etc/ftp*	1
/etc/group	1
/etc/profile	1
/etc/versions*	1
/etc/issue	1
/etc/issue.net	1
/etc/pump.conf	1
/etc/*.cfg	2
/etc/cmm/sensors.ini	2
/etc/cmm/fantray.cfg	2

Note: Performing an update to an earlier version of the firmware will cause a loss of some configuration data (such as SNMP trap address configuration which is stored in /etc/cmm.cfg) because files with a priority of 2 are not synchronized on updates to earlier versions.

23.6 Update Mode

To perform the update, the CMM will enter an update mode to prevent interference from other processes that may write to the flash and unmount the JFFS2 drives that may need to be re partitioned.

In the update mode, all CMM applications will be stopped. Any user-defined processes that are running must also be stopped. Hooks in the update process will be provided to allow these processes to be stopped by calling user scripts located in /home/update/scripts/ if they exist. The update process will execute any scripts that match the following pattern: /home/update/scripts/K*. Scripts will be execute in alpha-numeric sort order. Getting a directory listing using the “ls” command will display the order the scripts will run. User scripts should follow the convention of returning a “0” if successful and a non-zero for failure. The update process will not fail if a script returns a failure; however, the update process will fail if the flash drives can not be unmounted when necessary.

Before unmounting the JFFS2 drives (/etc and /home), /etc will be copied to ram disk /usr/local/cmm/temp/update/etc to retain user and system configuration information.

Because all CMM applications are stopped, systems running with redundant CMMs will lose redundancy during the firmware update process.

23.7 Update_Metadata File

The Update_Metadata file included in the update package is used by the update process to determine the platform, firmware package version, files, sequence, update method, location, and any other data required to update the individual components in the update package.

23.8 Firmware Update Synchronization/Failover Support

The following CMM versions and corresponding update directions will include support for heterogeneous synchronization.The CMM will continue to support behavior from firmware prior to version 5.2 for scenarios where a pre-configured newer CMM is introduced in a chassis with an older CMM.

Table 98. CMM Update Directions

CMM Versions	Update Direction Supported
4.x – 5.1	Upgrade/Downgrade with behavior prior to 5.2
4.x – 5.2	Upgrade only **Note**: the corresponding downgrade scenario (5.2->4.x), though possible from a CMM update point of view does not guarantee stable data synchronization behavior
5.1 – 5.2	Upgrade/Downgrade
5.2 – 6.1 (+/-)	Upgrade/Downgrade

The CMM will seamlessly synchronize all data items irrespective of version differences. User scripts (/home/scripts/*) is configurable to control its synchronization between differing CMM versions. Between same versions, this data will be synchronized.

Failover to a newer CMM firmware version (after an update and any other redundancy establishment) is configurable - automatic/manual.

The CMM also allows a build independent failover, allowing failover even to older firmware versions.

23.9 Automatic/Manual Failover Configuration

The CMM firmware now provides the user control over CMM failover when redundancy is established following a firmware update. The user can now decide if the CMM should or should not automatically failover to a newer CMM firmware version.

A new failover configuration flag - -"FailoverOnRedundancy" in the CMM configuration file (/etc/cmm.cfg) decides if an automatic or manual failover is desired. The configuration flag can be modified using the cmmget/cmmset commands. This flag can be read/set through any of the CMM interfaces (i.e., CLI, SNMP and RPC).

The setting of this flag will be in effect only starting the next time redundancy is established.

Note: The regular CMM health monitoring failover will continue to be in effect irrespective of the value of this flag. Other conditions such as CMM health, switch health (CompactPCI only), network connectivity that determine an automatic failover will continue to be in effect. This means that even if the failover configuration flag is marked "manual", if any of the above condition is met, automatic failover may occur.

23.9.1 Setting Failover Configuration Flag

To set the value of the failover configuration flag, the following CLI command can be used:

```
cmmset -l cmm -d failoveronredundancy -v [manual/automatic]
```

Where:

Manual: Upon establishing redundancy between the two CMM's, failover does NOT occur automatically to the CMM with the newer firmware version. This is the ***default value***.

Automatic: Upon establishing redundancy, failover to the CMM with the newer firmware version occurs automatically.

23.9.2 Retrieving the Failover Configuration Flag

To retrieve the value of the failover configuration flag, the following CLI command is used:

```
cmmget -l cmm -d failoveronredundancy
```

Where the return value will be: automatic, manual or an error

23.10 Single CMM System

When updating CMMs in a system that only contains a single CMM, once the CMM enters the update mode, any components in that system that the CMM manages power to may power down because the CMM applications are stopped. Consult your system and board documentation to determine the effect of updating or removing the CMM from the system.

23.11 Redundant CMM Systems

In systems with redundant CMMs, the update should be done first on the standby CMM. It is strongly recommended that an update is not performed on the active CMM.

Once the CMM enters the update mode, the chassis will lose redundancy for the duration of the update process. After the update is complete, initiate a failover from the active to the standby and update the second CMM which is now the standby.

Note: CMMs in a redundant system must only be updated one at a time, otherwise the chassis will become unmanaged.

23.12 CLI Software Update Procedure

The CLI supports a command for an update request. The syntax of the command is as follows:

```
cmmset -l cmm -d update -v “[Update Package path and filename] [force]
```

```
[overwrite] [ftp:server:user:password]”
```

Where:

Update Package path and filename - The path and file name of the update package file without the .zip or .md5 extension. For example: “usr/local/cmm/temp/CMM_P00.09f”

force - Optional argument that causes all components to update regardless of version. This option cause the environment variable FORCE_UPDATE to be set to TRUE.

overwrite - Optional argument that prevents data in the saveList from being preserved. Also, data restore user scripts will not be executed. This option cause the environment variable FORCE_OVERWRITE to be set to TRUE.

Note: When the overwrite option is not used when updating to 5.2 firmware, the following line needs to be added to /etc/profile to remove warnings using SNMP:

```
export MIBS=""
```

The following arguments are used if the update package is located on a remote FTP server. If *ftp* is supplied as an argument, *server* and *user* are also required. *Password* is optional. If *password* is not supplied, then *ftp* will prompt for a password.

ftp - Optional argument used to indicate that the update package resides on a remote FTP server. If this argument is given, the arguments for *server*, *user*, and *password* must also be given.

server - Optional argument that is the name or IP address of a remote FTP server containing the firmware update package.

user and password - Optional argument that is the username and password for the FTP server.

Note: The -v argument can be up to 64 characters long.

The command returns a 0 if the update request is successful, and non-zero if an error occurs.

23.13 Hooks for User Scripts

The update process provides hooks for user scripts in two places during the update process. User scripts should follow the convention of returning a "0" if successful and a non-zero for failure. The user scripts will be invoked as a forked sub-shell.

23.13.1 Update Mode User Scripts

Entering update mode requires that all JFFS2 drives be unmounted, which requires that all processes using these drives be stopped. Prior to entering update mode, the update process will call user scripts located in /home/update/scripts/ if they exist. The update process will execute any scripts that match the following pattern: /home/update/scripts/K*. No arguments are passed to these scripts. The update process will not fail if a script returns a failure, however, the update process will fail if the flash drives can not be unmounted when necessary. A message indicating if the script was successful or failed will be output to the terminal. See Section 23.6, “Update Mode” on page 208 for more information on the Update Mode.

23.13.2 Data Restore User Scripts

The update process will also execute user scripts following the update of the new /etc file system to flash. The update process will execute any user scripts that match the following pattern: /home/update/scripts/S*. Arguments will be passed that the scripts can use. These arguments are:

arg1 = Update Component Name: ["BlueCat", "etc_jffs", "RedBoot", "fpga"]

Note: For the user scripts, this argument will always be "etc_jffs".

arg2 = Update Direction: ["forward", "backward", "same"]

Note: "forward" means that the Update Package version is newer than the installed version. "same" means they are the same, and "backward" means the Update Package version is older then the installed version.

arg3 = Installed Package Version: i.e, "5.1.0.0117". This is the version of the software that is installed on the cmm.

arg4 = Update Package Version: i.e., "5.1.0.0117". This is the version of the update package software.

The update script will print a message to the terminal indicating whether the script was successful or failed. The failure of one of these scripts does not cause the overall update process to fail.

These scripts can be used to perform any user updates that need to coincide with the newly installed version.

Note: In prior versions of this document, the following examples showed how to use the user scripts to replace a script in the /etc/scripts directory. Going forward, all user scripts should be stored in /home/scripts. The directory /etc/scripts will be a link to /home/scripts. As a general rule, users should only put scripts and files under the /home directory.

23.13.3 Example Task—Replace /home/scripts/myScript

The following example update scenario demonstrates replacing /home/scripts/myScript with a newer version (if updating “forward”) or an older version (if updating “backward”).

- There are three CMM builds - call them A, B, and C. The CMM is currently installed with build B. Build A is older than B and C is newer.
- If the CMM is updated to build C, then the user script /home/scripts/myScript needs to be updated to a newer version.
- Similarly, if the CMM is reverted to build A, then the user script /home/scripts/myScript needs to be replaced with an older version.
- The old version of myScript is placed on the CMM as: /home/stagingarea/myScript.old
- The new version of myScript is placed on the CMM as: /home/stagingarea/myScript.new
- The Data Restore User Script written to manage /home/scripts/myScript is placed on the CMM as: /home/update/scripts/S10updateMyScript

A simple example of the S10updateMyScript could look like the following:

```
#!/bin/bash
```

```
direction=$2

if [ "$direction" = "forward" ] ; then
cp /home/stagingarea/myScript.new /home/scripts/myScript
elif [ "$direction" = "backward" ] ; then
cp /home/stagingarea/myScript.old /home/scripts/myScript
fi

exit $?
```

When the update process executes, and the Data Restore User Scripts are executed (during the update of the /etc partition), then the script S10updateMyScript will be executed and /home/script/myScript will be updated accordingly.

Variation on the above example:

- In this case, the different versions of myScript are put on the CMM (by the user) as:
  - — /home/stagingarea/myScript.A
  - — /home/stagingarea/myScript.B
  - — /home/stagingarea/myScript.C

Note: If the CMM has build A installed, the "cmmget -d version" command will return the string "A" - and so on for builds B and C.

Another example of S10updateMyScript :

```
#!/bin/bash
newversion=$4
cp /home/stagingarea/myScript.$newversion /home/scripts/myScript
exit $?
```

23.14 Update Process

1. Client initiates an update request via CLI command
2. CMM validates the update request
   - — CMM is not already doing an update
   - — Enough space is present in ramdisk and flash for the update
3. If update request is valid then
   - — Continue
4. Else
   - — Exit
5. If FTP arguments supplied then
   - — Retrieve package files (.zip, .md5) from FTP server
   - — Exit if error occurs
6. Else
   - — Copy package files (.zip, .md5) to package directory on ramdisk (/usr/local/cmm/temp/update/package)

   - Exit if error occurs

7. Validate that .zip file matches checksum in .md5 file
   - Exit if no match
8. Unzip the .zip file in the package directory on ramdisk
9. Validate that all files in the unzipped package match the md5 checksum in the validationFile.
   - Exit if any files fail
10. Validate that the package matches the CMM platform ("cpci" or "atca")
    - Exit if mismatch
11. Transition to Update Mode
    - Call any user supplied scripts in /home/update/scripts/K*
    - A call to /etc/rc.d/init.d/userScripts stop
    - Shutdown CMM apps (/etc/rc.d/init.d/cmm stop)
    - Copy /etc to /usr/local/cmm/temp/update/etc
    - Unmount /etc and /home
    - Exit if any of these filesystems cannot be unmounted
12. Update Components (listed in UpdateMetadata file)
    - If a component update fails
      Stop updating components
      Attempt to remount /etc and /home
      Exit the update process, but do not reboot
    - If the component is "etc_jffs" then
      Remount /etc and /home
      Call any user supplied scripts in /home/update/scripts/S*
      Process saveList - copying files and directories from /usr/local/cmm/temp/update/etc to /etc
13. If the process has been successful so far then
    - Remount /etc and /home
    - Update the /etc/versions file - backing up the old one as "/etc/versions.<version>"
    - Copy the update.log file from ramdisk to /etc/cmm/update.log
    - Reboot CMM

If an update fails to complete, the CMM will not load the application stack upon rebooting. The following message will be displayed upon failure:

```
Startup of CMM applications has terminated due to detection of an
inconsistent software load.
```

The software update must be executed again to solve the problem.

23.15 Update Process Status and Logging

During the update process, status is sent to stdout as it executes. Output will also be appended to the file /usr/local/cmm/temp/update/update.log. This file will be copied to /etc/cmm/update.log at the completion of the update process.

Status output will be of the format:

MM/DD/YY HH:MM:SS [process[pid]]: [Message String]

23.16 Update Process Sensor and SEL Events

Once the CMM enters the update mode in which no CMM processes are active, events cannot be entered into the SEL directly. While in this mode, the CMM will enter events into the EEPROM. Once the system reboots into the OS, the CMM will log the events from the EEPROM into the SEL.

23.17 Redboot* Update Process

The firmware can also be updated through Redboot. This update is done at a pre-OS level, meaning that the update is executed before the OS loads. This method requires updating over TFTP through the eth0 Ethernet port and must be done locally.

Because this process can completely erase the flash and operates in a pre-OS environment, it can be used as a failsafe to recover from failed firmware updates done from the command line interface.

Use the following instructions to perform a firmware update using this method, but please also refer to any instructions included in the update package and in the Specification Update for additional information.

23.17.1 Required Setup

To update the flash on the CMM using the Redboot* method, the following setup is required:

1. A host computer running Microsoft* Windows XP, 2000 or NT4.0 operating systems
2. The above computer should be configured with a static IP address of 192.168.100.20 and a subnet mask of 255.255.255.0 if connected directly or on the same subnet as the CMM.
3. It is highly recommended that the subnet be isolated. The only two network entities on the subnet should be the CMM and the Windows computer.
4. If the update is to be run over a live network, an unused IP address on the network will be needed.
5. The host computer should be connected via serial port to the console port on the CMM to monitor the update process as it occurs.

23.17.2 Update Procedure

Read through the following steps completely before starting the update.

1. Extract the contents of the .zip file into a folder (e.g. "C:\CMM\") on the host PC.
2. Open a DOS Command Line Prompt on the Windows* PC and change directory to the above folder (e.g., C:\CMM).
3. To execute the update, execute the update_cmm.bat on the host with the following options:
   a. The MAC address of the connected interface on the CMM is the first command line parameter. To find this out, use the command *ifconfig eth0* on the CMM CLI to find out the MAC address of eth0. The MAC address must be with dash separated values (not colon).
   b. A second argument “FFS2” must be used when upgrading/downgrading to a different version. Note that this will reset the passwords, IP addresses and other custom Linux* environment settings to factory defaults and clear the event log entries.

Example 3. Example update_cmm command when directly connected or on the same subnet:

```
update_cmm 00-a0-b7-d8-d8-5d FFS2
```

   c. By default, the update script assumes that the host computer and the CMM are on the network 192.168.100.x. If your host computer is on a different network, provide another command line argument to the update_cmm batch file. This second argument should be an unused IP address on the same network the host computer is on followed by the FFS2 argument.

Example 4. Example update-cmm command when connected over a live network.

```
update_cmm 00-a0-b7-d8-d8-5d 192.168.100.50 FFS2
```

4. Before pressing “Enter” and executing the command line, reset the CMM by pressing in and holding the front panel reset button for 5 seconds. The blue hot swap indicator will light to indicate the CMM will reset. Release the front panel reset button and the CMM will reset.
5. Simultaneously press enter and execute the update_cmm command line.
6. At this point some versions of the update_cmm batch file will prompt if you want to terminate the batch file. To continue you must enter “no” and press enter.

Note: Due to the nature of TFTP, the timing for the CMM to capture the TFTP request from the update client can sometimes be missed. This may lead the update_cmm batch file to prompt for the CMM to be reset again. This is normal and the update will continue as before once the batch file resumes.

Note: Occasionally the batch file will not detect the TFTP timeout, and will bet stuck in one of the update steps. This is usually indicated by a string of endless dots/periods on the update client screen. If this happens, hit ctrl-c on the update client, which will prompt to terminate the batch file. At this point you can hit “y” to terminate the update process and start over, OR to continue the update process, enter “n”, reset the CMM, and then press enter. The update process will continue from the step where it got stuck previously.

Updating Shelf Components 24

Certain elements of the shelf that are managed with an IPM device can have their FRU information and firmware updated either locally or remotely through the CMM. Components on the shelf that can be updated include the CDMs, the fan trays, and the PEMs. The CMM can also be used to update firmware on blades in the chassis.

Instructions on updating components in the Intel® NetStructure™ MPCHC0001 chassis (including the CDMs, PEMs, and fan trays) can be found in the *Intel® NetStructure™ MPCHC0001 Technical Product Specification*.

For instructions on updating the firmware in the Intel® NetStructure™ MPCBL0001 SBC, please refer to the *Intel® NetStructure™ MPCBL0001 Technical Product Specification.*

These documents and the firmware for these components can be found on the Intel web site at:

http://www.intel.com/design/network/products/cbp/index.htm

IPMI Pass-Through 25

25.1 Overview

The IPMI Pass-through feature allows IPMI commands to be sent directly to any device in the chassis through the CMM without being processed by lower layers of the CMM software stack. This allows local or remote devices such as other SBCs in the platform to send unfiltered IPMI commands directly to other devices in the platform through the CMM.

The command is available through any of the CMM interfaces (i.e., CLI, SNMP and RPC) and will function even when the blade may not be present or unable to communicate via IPMI.

25.2 Command Syntax and Interface

This command is set only.

cmmset -l [location] -d IPMICommand -v [Command Request String]

Where:

Location: Any

Command Request String: Command Request string (defined below)

25.2.1 Command Request String Format

This is a space delimited string containing the data for the command to be sent and has the following format:

```
-v “netfn [lun] cmd [data_0 …. data_n]”
```

Where:

netfn: A decimal or 0xY hexadecimal number specifying the Net Function of the IPMI request. The number must be an even integer greater than or equal to 0 and less than 62.

lun: A decimal or 0xY hexadecimal number specifying the destination LUN of the IPMI request. This number must be an integer greater then or equal to 0 and less than or equal to 3. The number must also be immediately preceded by the uppercase or lowercase letter L (i.e. L0 or l0). (This argument is optional and defaults to LUN 0 if not provided.)

cmd: A decimal or 0xY hexadecimal number specifying the command number of the IPMI request. The number must be an integer greater than or equal to 0 and less than or equal to 255.

data_n: Space separated decimal and/or 0xY hexadecimal numbers specifying the IPMI request’s data. These numbers must be integers greater than or equal to 0 and less than or equal to 255. There cannot be more than 25 data items.

The request string will only be validated for the format and ranges specified above. Any further validation of the command or data is left up to the receiver. If the range or format validation fails the error code of E_CLI_INVALID_SET_DATA will be returned.

Note: Please refer to the IPMI Specification for further details on IPMI commands and the values described above.

25.2.2 Response String

On a successful transmission, this is a space delimited string containing the data returned as the response to the IPMI request. All the values will be decimal integers. There will always be at least one number returned and is the completion code of the command. The number and meaning of rest of the numbers is dependent on the command sent.

Upon a failure in transmission an error code will be returned from the CLI API. It’s left up to the caller to interpret this code.

25.2.3 Usage Examples

25.2.3.1 CLI Usage Example

Sending an AdvancedTCA* Get PICMG Properties command to LUN 0 of blade 7:

```
# cmmset -l blade7 -d IPMICommand -v "0x2c L0 1 0"

0 0 71 142 255 0 4 0
```

25.2.3.2 RCP Usage Example

Sending an AdvancedTCA* Get PICMG Properties command to LUN 0 of blade 7:

```
# cli_client –s MPCMM0001 –m set -l blade7 –d IPMICommand -v “0x2c 1 L0 0”
0 0 71 142 255 0 4 0
```

25.3 SNMP

Because SNMP sets cannot return data the IPMI pass-through functionality will be split into two SNMP objects under each location: IPMICommandReq and IPMICommandRes (OIDs TBD).

IPMICommandReq will be Read-Write. On a read (get) it will return a string (initially empty) containing the last successful request performed via SNMP. On a write (set) it will return whether the IPMI command was successfully sent and the response was successfully received.

IPMICommandRes will be Read-Only and will return the response string of the last successful IPMICommand. In order to differentiate between requests, the response string will also be followed by the request string separated by “#”.

25.3.1 Usage Example

Sending IPMI Get Device ID request to the CMM:

```
# snmpget […] […].cmmIPMICommandRequest
[…].cmmIPMICommandRequest=””
# snmpget […] […].cmmIPMICommandResponse
[…].cmmIPMICommandResponse=””
# snmpset […] […].cmmIPMICommandRequest s “6 1”
OK
# snmpget […] […].cmmIPMICommandRequest
[…].cmmIPMICommandRequest=”6 1”
# snmpget […] […].cmmIPMICommandResponse
[…].cmmIPMICommandResponse=”0 32 129 5 2 81 255 87 1 0 65 8 0 0 0 0 # 6 1”
```

FRU Update Utility 26

26.1 Overview

This utility is intended to be used to update any FRU information in an AdvancedTCA* or CompactPCI chassis. It will be able to update functional system FRU information with a FRU provided in the update package; or it can be used to repair damaged FRU information. It can also view all FRU information in the chassis.

The utility is able to read and write to all FRU devices in the chassis, including but not limited to:

- CMM
- fan trays
- power supplies / PEMs
- single board computers (SBCs)
- CompactPCI* shelf FRU
- CDMs (on the AdvancedTCA* chassis)

Update features are controlled by configuration file (.CFG), allowing automation of the update process as well as being able to specify specific fields in a FRU to update. The configuration files specify what input FRU files and target locations should be used in the update, and also provides a mechanism for querying the chassis and user for specific information during the update process.

26.2 FRU Update Architecture

The FRU update utility is responsible for replacing existing FRU information with valid FRU information contained in a vendor supplied FRU file as well as recovering corrupted FRUs on a system. The utility will support both non-interactive and interactive operation of the update (as determined by the configuration file). It will also be capable of displaying the contents of the FRU or writing the contents of a FRU to a file.

Access to the various FRUs in the chassis depends on the state of the active and standby CMM software, and whether the FruUpdate utility is being run on the active or standby CMM. The following table describes which FRUs in the system will be accessible based on these conditions.

Table 99. Platform FRU Accessibility of the FRU Update Utility

State of CMMs / Where Utility Is Run	Chassis FRU	CMM FRU	Remote IPMCs	Fan Trays (CPCI Only)
Active Up/ Standby Up/ Run from Active	Available	Available	Available	Available
Active Up/ Standby Up/ Run from Standby	Unavailable	Available	Unavailable	Unavailable
Active Up/ Standby Down/ Run from Active	Available	Available	Available	Available
Active Up/ Standby Down/ Run From Standby	Available	Unavailable	Unavailable	Unavailable
Active Down/ Standby Down/ Run from Standby	Available	Available	Unavailable	Available

26.3 FRU Update Process

The FRU update process is controlled by the configuration file (not including forced FRU recovery). The configuration file is capable of querying the system and user for information. The configuration file can be modified to adjust how the FRU update proceeds. Because the update process is controlled by a user modifiable file, the utility will perform a three part update process to ensure a proper update is performed.

The first step is to read the file and validate the syntax of the commands. If there are any errors in the configuration file, an error message will be displayed stating which line in the file had an error and what that error was. The utility will exit after the first error has occurred.

The second step is to process the commands in the configuration file and validate them to the hardware. This step will perform probing, gather user input, and read the FRU devices in the chassis and perform the first verification of the FRUs. All update files will also be read and validated at this time.

When validating the FRU file to use for the update, the utility will compare the IPMI FRU header in the update file to the IPMI FRU header in the system hardware. If the headers are not equal, the utility will exit with an error. This is a measure to ensure that the correct FRU file was chosen for the location. If the user desires more verification that the FRU is for the correct device, probing commands can be used in the configuration file to validate product IDs, manufacturer name, and other fields.

The third step is to commit the changes to the system. The FRUs are written in the order they are specified in the configuration file. If any FRU fails to write, the utility will exit and display an error to why the FRU failed to update; none of the FRUs due to be updated after the failed FRU will be updated. All FRUs successfully written before the failure will remain updated.

If, while writing to the FRU, the device times out with a communication error or the FRU write command fails, the utility will attempt the command two more times. The communication layer will also attempt each command three times before returning an error, for a total of nine retries per IPMI command before exiting.

26.4 FRU Recovery Process

If the user is attempting to recover a corrupted FRU, they will use the forced FRU recovery option. With this option, no FRU verification will be done with the target device’s FRU before the update happens. No header comparison will be performed, and the target FRU information will not be read and validated.

When doing a forced FRU recovery, all information on the target FRU WILL BE LOST and will be replaced by the data in the recovery FRU file, including protected fields.

A FRU file to force onto the target device needs to be provided. The data in this FRU file will be written onto the target device.

26.5 FRU Verification

FRUs will be verified twice during the update process, once before any action is taken on the FRU, and once more after all bytes have been written to the FRU. If the verification fails when first reading the FRU from the system, the FRU will not be modified, and the update process will exit. This first verification will not take place on a force FRU recovery.

If verification fails after the FRU is written to the system, the utility will attempt at most two more writes of the FRU to the system. If none of the write attempts is successful, the utility will exit with an error and no further FRU's will be updated. Any FRU updated before the error occurred will be updated.

26.6 FRU Display

Displaying the FRU data is done from the command line. The utility will be able to view binary files, ASCII FRU files, and FRU devices. ASCII FRU files are determined by file extension; all ASCII FRU files must have a '.FRU' extension and binary files should not use that extension.

FRU devices will be read and validated before displaying the information. If any part of the binary file or FRU device is invalid, none of the FRU data will be displayed.

Areas of the FRU that will be displayed are: header, internal, chassis, board, product, and multi record areas. The header and internal areas will be displayed as hex data. The chassis, board, and product areas will be decoded according to the FRU storage definition spec. And the multi record areas will be decoded into header and data segments, the header will be decoded and the data will be displayed in hex.

26.7 Setting the Library Path And Invoking the Utility

Because the utility uses shared libraries that are not part of the firmware image libraries, the user must specify additional search locations for the FRU update libraries. Usually this is the current directory '.'.

To run the utility from the directory containing the utility:

```
LD_LIBRARY_PATH=. ./fruUpdate <commands>.
```

To run the utility from a path that is not the current working directory, you must point the load library path to the directory with the utility:

```
LD_LIBRARY_PATH=/home/fruUpdate /home/fruUpdate/fruUpdate <commands>.
```

26.8 FRU Update Command Line Interface

The syntax for the command line of the FRU Update utility is as follows:

LD_LIBRARY_PATH=. ./fruUpdate [/argument paramter]

Arguments can be preceded by either a '-' or a '/' character. Arguments have both a long and short name which can be used.The table below lists the valid arguments, both long and short names, for the FRU Update utility as well as their corresponding parameters.

Table 100. FruUpdate Utility Command Line Options

Argument (Short Name)	Parameter (Required/Optional)	Description
Update (u)	Name of the configuration file to use for the update process. (Required)	Specifies the update configuration file to use.
ForceRecovery (fr)	Name of the .FRU or .BIN file to use for the forced recovery process. (Required)	Specifies the name of the FRU file to use for a forced recovery. This requires the 'location' switch to be supplied as well. Data from the target FRU is not preserved when updated this way.
Location (l)	Location of the target device for the force and view commands. Formatted as Location:DeviceID:Bus. The location name is required and should match a valid location in the system or the IPMB address of the FRU. If no device ID is supplied, 0 is used. If no bus is supplied, 0 is used and the utility will determine the correct bus.	Specifies the location to use for force, view, get, and dump commands; not valid with the 'update' switch.
Inventory (i)	None	Get the FRU version inventory of the targeted update FRU configuration file. Requires the 'update' option to supply the file. This provides a means to test a configuration file by listing the files that would be used in the FRU update without modifying the system FRU contents.
Login	Username:Password Username is required, Password is optional, and if not supplied, the utility will prompt for it and echo '*'s instead of the characters.	This gives the username and password for authenticating a connection.
View (v)	The name of the input file to view; can be binary or an ASCII FRU file. If the extension is '.fru' the file is considered to be an ASCII FRU file. (Optional)	View the contents of the FRU; this will decode as many fields as possible in the FRU. This 'location' switch is optional and is not allowed if the parameter is supplied.
Get (g)	The name of the binary output file to create. (Required)	Get the contents of the FRU and write them to a file. This requires the 'location' option also be specified. The contents of the FRU are validated as they are read, and no file is created if the FRU is invalid.

26.9 Using the Location Switch

The location switch allows valid cmmget and cmmset locations to be specified in other commands. The location can also be an IPMB address of the device, such as '0x20'. If access to a FRU on a device other than the FRU at the logical device ID 0, they can specify the device ID after the location name, separated by a colon. For example, "-l blade3:1" would access the FRU at logical device ID 1 on blade3.

The location can also include a bus address. The bus address is needed to access the correct power supply in a CompactPCI* chassis if the IPMB address is used instead of the location name. When the bus is defined, the location and device ID must also be defined. For example, "-l 0x54:0:26" would access IPMB address 0x54, device ID 0, on IPMB bus 26.

26.10 Updating the FRU

The “update” (u) switch is used to update a FRU. This argument requires a configuration file to be specified. The update will then be performed according to the contents of the configuration file.

Example:

```
LD_LIBRARY_PATH=. ./fruUpdate /u ChassisFruUpdate.cfg
```

“FRU Update Configuration File” on page 227 describes the format of the configuration file and how to update FRU information using the configuration file.

26.11 Getting the Inventory

The ‘inventory’ command displays the versions of the FRUs for the selected target update file. The user must also specify a configuration file with the ‘update’ switch. The inventory will list the versions of the FRUs that will be updated as if the update would actually occur.

Example:

```
LD_LIBRARY_PATH=. ./fruUpdate /i /u ChassisFruUpdate.cfg
```

26.12 Viewing the Contents of the FRU

The contents of a FRU can be viewed with the ‘view’ option. The user must specify a location to view using the ‘location’ switch, unless viewing a file. If the user wishes to view the contents of a file, the file name must follow the ‘view’ switch. The input file can be either an ASCII FRU file or a binary FRU image. The utility will detect automatically which is being viewed.

Example:

```
LD_LIBRARY_PATH=. ./fruUpdate /v /l chassis
```

26.13 Getting the Contents of the FRU

The contents of a FRU can be written to a binary file image with the ‘get’ option. The file name to write the contents to must be specified. The user must also specify a location to get using the ‘location’ switch. The contents of the FRU are validated as they are read.

Example:

```
LD_LIBRARY_PATH=. ./fruUpdate /g Board10.FRU /l blade10
```

26.14 Dumping the Contents of the FRU

The contents of a FRU can be written to a binary file image with the ‘dump’ commandl. The file name to write the contents to must be specified. Optionally the number of bytes to write from the FRU can also be specified. If the number of bytes to write is 0, the utility will attempt to determine the maximum size of the FRU device and write the entire contents. The user must also specify a location to get using the ‘location’ switch. The contents of the FRU are not validated as they are read.

Example:

```
LD_LIBRARY_PATH=. ./fruUpdate /d output.bin 12 /l chassis
```

FRU Update Configuration File 27

27.1 Configuration File Format

This section specifies the format for the configuration file used to assist in determining the configuration of the product. The configuration for a product consists of information about the product that can be used to select appropriate Field Replaceable Unit (FRU) information to be loaded into non-volatile storage areas. The information about the product is obtained by identifying all the boards, sub-assemblies, and components in the product.

The configuration file directs the load utility via a system of commands, probes, requests, and prompts. The configuration file tells the load utility how to discover the configuration of a product via probing and user queries. As the Load Utility processes the configuration file, tag values (sensor ID and descriptive strings) are accumulated and saved. The entire file is processed prior to the FRU files being programmed.

In programming the FRU areas, the appropriate areas must be specified after the FRU name is given. If no FRU areas are specified, then the default procedure is to disregard all FRU areas in the FRU file and leave what was previously programmed in the non-volatile storage device.

27.2 File Format

The configuration file format is ASCII text and is editable. The file consists of commands, data, and comment fields. There are two data types, strings and numbers. The numeric data values will be represented in decimal unless specified otherwise by a leading ‘0x’. Strings are consecutive non white space characters or a double quoted series of printable characters. These strings and numeric values are formed into fields that make up commands, requests, and prompts. The commands, requests, and prompts can be formed into combinations and used to obtain the information necessary to establish a system’s configuration. A line in the configuration file can be commented out by placing the comment characters (“//”) at the beginning of the line. The comment characters are effective until the end-of-line character is reached. An example configuration file is shown at the end of this document. Comment characters can also be used in the middle of the line to comment out all data on the line after the comment characters.

The extension for the configuration file will be ‘cfg’, case does not matter. If the file does not have the ‘cfg’ extension, it shall cause an error in the utility.

27.3 String Constraints

- All lines in the file are allowed to be unlimited in length and are terminated with a linefeed.
- All strings with spaces must be enclosed in double quotes. It is recommended that all strings be enclosed in double quotes.
- The double quote character is not valid *within* a string (that is, in any position except the enclosing double quote marks).
- If a null ASCII string is desired, then two double quotes must be inserted as such, “”.

- As the configuration file is parsed, on each line the correct number of quotes is checked and when incorrect, flagged as an error prior to checking for the correct number of arguments.
- White space is defined as space, tab, and Carriage Return-Line Feed characters (CR-LF).

27.4 Numeric Constraints

- All numbers are considered decimal unless denoted by the hexidecimal prefix string.
- If as number starts with “0x”, then number is interpreted as hexidecimal.

27.5 Tags

Tags are a means for the configuration file to control the path for users of the file. Tags are a Boolean (true / false) value represented by a string. When the tag is set, a string for that tag is added to a utility wide (master) list of set tags. When a tag is cleared, all tags with that string are cleared from the master list of tags, thus making the tag false. Only tags present in the master list are true, all other tags are false. Tags are only set, cleared, and checked during the processing phase of the configuration file. The order in which the tags are set, cleared, and checked is very important. A tag that is set after it is checked by a statement will be false for the statement.

27.6 Control Commands

These commands are used to control the execution path of the configuration file. This is done with the setting, clearing, and checking of tags. The control commands are the following: IFSET, ELSE, ENDIF, SET, CLEAR, CFGNAME, ERRORLEVEL, PROBE, FOUND, PROMPT, YES, NO, MENUTITLE, MENU, and MENUPROMPT. The commands PROMPT, YES, NO, MENUTITLE, MENU, and MENUPROMPT are covered in Section 27.6, “Control Commands” on page 228, and PROBE and FOUND are covered in Section 27.7, “Probing Commands” on page 230.

27.6.1 IFSET

The IFSET statement will take an unlimited number of tags for its arguments, with an implicit AND between each argument. Each IFSET statement must have a matching ENDIF statement, and if desired, can have one ELSE statement between the two. If the tags required by the IFSET command are set during the processing of the configuration file, the commands following the IFSET command and up to the first ELSE or ENDIF command will be processed and executed. If the tags are not set, the commands up to the next ELSE or ENDIF command are skipped.

Example:

```
SET TRUE

IFSET “TRUE”

// Do this stuff

ELSE

// Skip this stuff

ENDIF
```

Additionally there is wild card support for numbers in a tag string. When the '#' character is used in the tag string following an IFSET statement, it will match any number found in that position of any of the tags previously saved in the tag string link list.

For example, when screening for various versions of a platform, you can use the following IFSET statement:

```
IFSET  “MPCHC000#”
```

27.6.2 ELSE

The ELSE command has no parameters. It defines the alternate path of commands to run if the previous unmatched IFSET command failed when checking the tags. The ELSE command also signals the end of the commands that will be run when the matching IFSET command succeeds. If an ELSE command does not have a matching IFSET and ENDIF, the program should exit and notify the user. The commands following the ELSE command up to the matching ENDIF command will be processed and executed if the matching IFSET fails. If the matching IFSET command passed, then the commands following the ELSE command and up to the matching ENDIF are skipped.

Example:

```
CLEAR FALSE

IFSET “FALSE”

// Skip this stuff

ELSE

// Do this stuff

ENDIF
```

27.6.3 ENDIF

The ENDIF command signals the end of the commands controlled by an IFSET or ELSE command. If an ENDIF command does not have a matching IFSET or ELSE command, the program should exit and notify the user.

27.6.4 SET

The SET command adds a tag to the master list to be checked by IFSET commands. If the tag already exists in the list, it is added again. When a tag is set, it remains set for the remainder of the processing phase, is has no affect of IFSET commands prior to it in the file, only those that follow, or until cleared.

Example:

```
SET   “tag”
```

27.6.5 CLEAR

The CLEAR command removes a tag to the master list to be checked by IFSET commands. If the tag does not exist in the list, nothing happens. If multiple tags exist, all are deleted.

Example:

```
CLEAR   “tag”
```

27.6.6 CFGNAME

The CFGNAME command identifies a configuration file to insert at this point in the current file. The referenced configuration file is treated as an extension of the current configuration file at the location of command. The results of the commands in either configuration file will affect all configurations files. If a tag is set is the upper file and then cleared in the referenced file, it remains cleared at the end of the referenced file.

This command does not support pathnames in the referenced file name. It will verify that a series of configuration files does not create a cyclic inclusion of files. That is, file A includes B and B includes A. The same file name can be used multiple times in a file as long as they don’t included cyclic inclusions, so A can include B twice.

Example:

```
CFGNAME   “fwupdate.cfg”
```

27.6.7 ERRORLEVEL

The ERRORLEVEL command causes the load utility to exit immediately and return the error code specified. Returning an error level of zero shall make the utility exit with a successful error code immediately. The zero errorlevel can be used to terminate a configuration file early without causing an application error. An errorlevel of 0 does not perform updates to the FRU.

Example:

```
ERRORLEVEL   01
```

27.7 Probing Commands

Probing is a method used to determine the system configuration by querying the system. Probing allows you to determine the system’s ID string, firmware version, and a targeted FRU’s version. The FOUND command always follows the PROBE command and determines which tag is set when the probe succeeds or fails. These tags can then be used by the IFSET command to change the execution path. Only IFSET commands following the probe will be able to check the tags set by the probe command.

27.7.1 PROBE

The following table shows the valid PROBE command types that will be covered and the required parameters for each. The individual types will be covered in more detail following the table.

Table 101. Probe Command Parameters

Probe Type	Parameter	Description of Parameter
any	0	Error, invalid command, too few arguments
SYSTEM	1	System ID string to compare equal too
FRUVER	1+	Version strings to check ranges for
BMCVER	1+	Version strings to check ranges for

27.7.2 SYSTEM

When the “SYSTEM” argument is used, then the utility reads the currently selected FRU’s board area part number as a string and compares that to the argument string that follows the SYSTEM argument. If the two strings are the same (case insensitive), then the probe is TRUE. If they don’t compare, then the probe is FALSE. This test can be used to ensure that the correct configuration files are used on the appropriate system.

You cannot probe the FRU version without first specifying a FRU to probe using the FRUNAME and FRUADDRESS commands. You can read more on these commands in section Section 27.8, “Update Commands ” on page 233.

Example:

```
PROBE “SYSTEM” “MPCHC0001”

FOUND “ATCA” “NO ATCA”
```

27.7.3 FRUVER

Probing the FRU version is used to determine if a compatible version the currently selected FRU exist on the system. The probe will get the FRU board ‘FRU file ID’ value and compare it to the list of versions that follow in the parameters. The probe arguments allow a major and minor version in the string, but in practice the FRU versions are a single number. The version only needs to be within one of the multiple ranges that can be supplied in the probe arguments. Both the argument strings and the system string are converted from a string to a decimal number to perform the compare.

You cannot probe the FRU version without first specifying a FRU to probe using the FRUNAME and FRUADDRESS commands. You can read more on these commands in Section 27.8, “Update Commands ” on page 233.

Example:

```
PROBE FRUVER “104” “106-107” “112+”

FOUND “FRUMATCH” “FRUINVALID”
```

There are three ways to define a FRU version range. As a single version - a number in quotes, as a defined range - two numbers separated by a dash ‘-‘, and as an infinite range - a number followed by a plus sign ‘+’. The format of the FRU version is “AA.BB”, where “AA” and “BB” are decimal numbers.

27.7.4 BMCVER

Probing the BMC version is used to determine if a compatible version of firmware exist on the system. The version that is queried is the controller at IPMB address 0x20. The probe will get the BMC operational code version from the system and compare it to the list of versions that follow in the parameters. The BMC version is retrieved by an IPMI call, so the version is limited to the value that can be returned in the IPMI response message. The probe arguments only allow a major and minor version in the string. The version only needs to be within one of the multiple ranges that can be supplied in the probe arguments.

There is also an optional method to probe for the BMC boot code version. By supplying the keyword “BOOT” as the first argument, the probe will return the boot code version of the BMC.

Example:

```
PROBE BMCVER “05.05” “06.01-09.01” “12.00+”// Op code version

FOUND “BMCMATCH” “BMCINVALID”


PROBE BMCVER “BOOT” “00.09”// Boot code version

FOUND “BOOTMATCH” “BOOTINVALID”
```

There are three ways to define a BMC version range. As a single version - a number in quotes, as a defined range - two numbers separated by a dash ‘-‘, and as an infinite range - a number followed by a plus sign ‘+’. The format of the BMC version is “AA.BB”, where “AA” is a hexadecimal number version and “BB” is a BCD number.

27.7.5 FOUND

The FOUND command is only valid after a PROBE command. It defines what flags are set if the probe is successful or fails. The FOUND command accepts two arguments, the second is optional. The first argument is the tag to set if the probe is successful, and the second is the tag to set if the probe fails.

```
PROBE   “SYSTEM” “SE7501WV20”

FOUND   “SWV2”
```

The first argument for this probe is “FRU_VERSION”, the second argument will be the version string to be checked for. If the FRU’s board area FRU file id field is the same as the FRU version string, the probe will set the found tag. The string compare for the version is not case sensitive.

Example:

```
PROBE FRUVER “103” “104-105” “106+”// board FRU file ID

FOUND “FRUMATCH” “FRUINVALID”
```

27.8 Update Commands

The configuration file allows the user to update the FRUs by specifying a target source file. And allow more customization of how the update should proceed with more commands. If the load utility fails at any of the updates, it should exit immediately and report the error. The update commands do not support path names; no error will be generated if a pathname exist.

27.8.1 FRUNAME

The FRUNAME command specifies the file name of the FRU to update the system with. Alternately, the name of the FRU file can also be “SYSTEM”, and in that case the FRU will be loaded from the system. In either case, the contents of the FRU specified are not modified unless specified with FRUADDRESS, FRUAREA, and FRUFIELD commands.

The SYSTEM FRU name should be used when you don’t want any inputs from a FRU file and want to change a field by typing in the value or reading from an environment variable.

Example:

```
FRUNAME “MPCHC001.FRU”

FRUADDRESS chassis

FRUAREA “PRODUCT”

FRUFIELD “P#” // Input from MPCHC001.FRU file

FRUFIELD “S#” “@STDIN:ASCII” // Input from user thru standard input
```

The configuration file allows the user to select the areas and fields to update through using FRUADDRESS, FRUAREA, and FRUFIELD. By default, no FRU areas are programmed, even if a FRUNAME is given. If the user wants the entire FRU file programmed, then they need to specify all the FRU areas and fields to be programmed.

To update more than one FRU in a configuration, you need to identify a second FRU by using the FRUNAME command again. All FRUADDRESS, FRUAREA, and FRUFIELD commands above the command in the file will have no affect on this new FRU entity. You will need to specify a new address and fields to update.

Example:

```
FRUNAME “MPCHC001.FRU”

FRUADDRESS chassis

FRUAREA “PRODUCT”

FRUFIELD “P#”


FRUNAME SYSTEM

FRUADDRESS cmm

FRUAREA “BOARD”
```

```
FRUFIELD “S#” “@STDIN:ASCII”
```

27.8.2 FRUADDRESS

The FRUADDRESS command specifies an alternate address to program the FRU file to, by default the address in the file is used. If the SYSTEM FRU name is used, this is required by the user to correct outcome. Multiple FRUADDRESS commands are allowed for a FRUNAME command, but only the last FRUADDRESS command will be used. The FRUADDRESS command has three different addressing formats.

FRUADDRESS commands do not need to be immediately after the FRUNAME command to change the address of that FRU entity. The FRUADDRESS is applied to the FRUNAME as the file is read in and validated. **This means that the FRUADDRESS command ignores control statements and should not be used inside IFSET commands, because it won’t have the correct effect.** When a FRUADDRESS command is read from the file, the utility will find the last FRUNAME identified and modify the address of the FRU to match the FRUADDRESS command. This way, when the FRU is processes in the second phase of the utility, the FRUADDRESS is set and the utility can start to make changes to the FRU without having to parse the remainder of the file.

Format one takes only one argument, it is the logical address of the FRU device or the alias name of the FRU.

Example:

```
FRUADDRESS   “01”
```

```
FRUADDRESS “CHASSIS”
```

Format two takes two arguments, the address of the device (or the alias name of the FRU), and the device ID of the FRU.

Example:

```
FRUADDRESS   “0x92” “00”
```

```
FRUADDRESS   “CHASSIS” “03”
```

Format three takes 3 arguments, the address of the device (or the alias name of the FRU), the device ID of the FRU, and the IPMB bus number.

Example:

```
FRUADDRESS   “BLADE9” “00” 25
```

27.8.3 FRUAREA

The FRUAREA command must be preceded by a FRUNAME command. This command identifies the area of the FRU that the FRUFIELD commands are to update because multiple areas have the same size fields in them. The valid area selections are listed in Table 102.

Table 102. FRU Area String Specifications

FRU Area ASCII Strings
“HEADER”
“INTERNALUSE”
“CHASSIS”
“BOARD”
“PRODUCT”
“MULTIREC”

The HEADER area is not selectable for writing; it is listed as an option to allow future use for comparisons. The internal use area is unique in that it has no fields, and can only be modified as a whole. When the ‘FRUAREA INTERNALUSE’ command is used, the internal use area in the FRU file targeted by the FRUNAME command is written over the internal area of the FRU in the system. Because the utility requires that the headers between the two FRUs be identical, we can be assured that the internal use areas are the same, and we need not do anything special to support different sizes of the areas.

Example:

```
FRUNAME SYSTEM

FRUADDRESS chassis

FRUAREA PRODUCT

...

FRUAREA BOARD

...
```

27.8.4 MULTIREC

The MULTIREC area is the only area selector that requires parameters to be passed to it. The parameters select which multi record to manipulate and how to do so. Source records can replace, insert before, and append after the selected destination record. You can also delete a destination record from the FRU. The source record is the data to be used after the update takes place (from the FRU designated by the FRUNAME command); the destination record is the currently existing data in the system. The formats for the parameters to the command are in the following table.

Table 103. Multi-Record Selection Parameters

Parameter Number	Parameter Value	Description
1	APPEND, INSERT, REMOVE, REPLACE	Operation to perform for the selected record.
2	"PICMG_ID", "RECORD_ID"	Method to use for selecting the source and destination records.
3	Number	Destination record ID to be selected.
4	Number	Destination record count to be selected, 1 is the first record.
5	Number	Source record ID to be selected.[1]
6	Number	Source record count to be selected, 1 is the first record.[1]

NOTE:

1. Not allowed for the REMOVE operation.

The destination and source record IDs are based on the second parameter, the selector type. If the selector type is ‘RECORD ID’ the utility should look at the OEM record ID in the record header. If the selector type is ‘PICMG ID’ the utility should validate that the record is a PICMG record and then compare to the PICMG record ID field. When multiple IDs match the selector ID, it is necessary to be able to select each record; this can be done with the record count parameter. The record count parameter indicates which number of the record that matched should be selected, 1 is the first record discovered in the area. If the utility cannot find a record that matches the ID and count of the selectors, it shall exit with an error and not perform any updates to the FRU.

Example:

```
FRUNAME SYSTEM

FRUADDRESS CHASSIS

FRUAREA MULTIREC REPLACE PICMG_ID 0x12 1 0x12 1

FRUAREA MULTIREC REMOVE RECORD_ID 0xC1 1
```

27.8.5 FRUFIELD

The FRUFIELD command must be preceded by a FRUAREA command. This command specifies what and where the input value for this field is to come from when updating the FRU. If the same field definition occurs multiple times in an area, then the modification made by the last field definition will be the one used, and no error will be displayed to the user. Also, the field must exist in the system FRU that is to be updated, the configuration file is only capable of modifying existing FRU fields not adding new fields. The valid field selections for an area are listed in Table 104 below. The table below lists the description of the abbreviated field name and (the FRUFIELDs that the field is allowed in).

Table 104. FRU Field First String Specifications

FRU Field First ASCII String	String Description
“CT”	**Chassis Type** (Chassis Area) The input string must represent a number. Regardless of the input type specified in the configuration file. This is because the chassis type field is an encoded value and is limited in range. The utility shall limit the input to the valid range and interpret the value correctly when decoding.
“MN”	**Manufacturer Name** (Board and Product Areas)
“PN”	**Product Name** (Board and Product Areas)
“P#”	**Part Number** (Chassis, Board and Product Areas)
“S#”	**Serial Number** (Chassis, Board and Product Areas)
“PV”	**Product Version** (Product Area)
“AT”	**Asset Tag** (Product Area)
“ID”	**Manufacturer ID** (Board and Product Areas)
“MD”	**Manufacturing Date & Time** (Board Area). Date and time are supplied by the Server, therefore no arguments are allowed following “MD”. This field shall not be updated by the utility; as so, the utility should error when this field is selected for an update.
“AMx”	**Additional Manufacturing Info** (Chassis, Board and Product Areas) There can be multiple instances of this field, where ‘x’ represents a decimal number from 1-9.

If you specify the “MD” field, then the date and time will be updated from the server. If you add any arguments after the “MD” field, then an error will be displayed.

Table 105 defines the maximum length of the fields that shall be allowed by the utility. The load utility may force the input to be shorter if there is not enough room in the FRU area to support the maximum length of the FRU field.

Table 105. FRU Field Maximum Allowed Lengths (Sheet 1 of 2)

FRU Fields	FRU Field Max Allowed Length
Product Manufacturer	1Fh bytes
Product Name	1Fh bytes
Product Part Number	17h bytes
Product Version Number	1Fh bytes
Product Serial Number	1Fh bytes
Product Version	1Fh bytes
Product FRU Field ID	1Fh bytes
Product Custom Fields (AMx)	1Fh bytes
Board Manufacturer	1Fh bytes
Board Product Name	1Fh bytes
Board Part Number	17h bytes
Board Serial Number	1Fh bytes
Board FRU Field ID	1Fh bytes
Board Custom Fields (AMx)	1Fh bytes
Chassis Type	1h byte

Table 105. FRU Field Maximum Allowed Lengths (Sheet 2 of 2)

Chassis Part Number	17h bytes
Chassis Serial Number	1Fh bytes
Chassis Custom Fields (AMx)	1Fh bytes
Product Asset Tag	1Fh bytes

Whether entering information by way of STDIN, an environment variable, or from a file, if an input length of zero is entered, then the utility will treat it the same as an empty string. An empty string will be entered into the selected FRU field, thus wiping out any previously stored input. If the environment variable does not exist, then a length of zero will be used and a warning message displayed.

Table 106. FRU Field Second String Specification

NONE	If a second string does not exist, then the string in FRU file is used by default.
“”(empty string)	An empty string will clear the value currently in the FRU field.
“@FILE:TYPE:NAME:#”	**@FILE** designator lets the utility know that it needs to obtain the string from a file designated by NAME. NAME may be a complete path, minus the drive letter, and up to 64 bytes, the filename is in standard DOS format. If it is desired to specify another drive, then the “@ENVFILE” argument should be used. NAME may also be IFICS, this lets the utility know it needs to find the IFICS file to get the string from it. The ‘#’ specifies which line of the file to insert as the string, the first line in the file is 1. If no line number is supplied, then an error will occur. TYPE specifies what coding designation the ASCII string from the file needs to be translated to, but only ASCII is supported. If another valid TYPE is specified, then it will be ignored and ASCII will be used, no error message will be generated. Possible TYPE designations specified in Table 107. Valid input terminates with any ASCII character entered outside the bounds of 20h to 7Fh.
“@ENV:TYPE:NAME”	**@ENV** designator lets the utility know it needs to obtain the ASCII string from the environment variable specified by NAME. TYPE specifies what coding designation the ASCII string from the file needs to be translated to, but only ASCII is supported. If another valid TYPE is specified, then it will be ignored and ASCII will be used, no error message will be generated. Possible TYPE designations are specified in Table 107. Valid input terminates with any ASCII character entered outside the bounds of 20h to 7Fh. If the environment variable does not exist, then a warning message is displayed and a input length on zero is used.
“@ENVFILE:TYPE:NAME:#”	**@ENVFILE** designator lets the utility know it needs to obtain a DOS drive, path and filename string from the environment variable specified by NAME. TYPE specifies what coding designation the ASCII string from the file needs to be translated to, but only ASCII is supported. If another valid TYPE is specified, then it will be ignored and ASCII will be used, no error message will be generated. Valid input terminates with any ASCII character entered outside the bounds of 20h to 7Fh. If the environment variable does not exist, then a error message is displayed and the application terminates.
“@STDIN:TYPE”	**@STDIN** designator lets the utility know it needs to obtain the ASCII string from operator input. TYPE specifies what coding designation the ASCII string from the file needs to be translated to. Possible TYPE designations are specified in Table 107. Valid input terminates with any ASCII character entered outside the bounds of 20h to 7Fh.

Table 107 defines the valid types that can be used as the 'TYPE' option in FRU field 2nd string specifications. The load utility is not required to support all of these types.

Table 107. Type Code Specification

Type	Description	FRUSDR Version
ASCII	Straight 8 bit ASCII	Ver 2.0 and later
ASCII6	6 bit ASCII packed bytes	Not supported
BCD	Binary Coded Decimal.	Ver 2.0 and later. Support for BCD only exists when using Stdin and in the original FRU file.
BIN	Binary bytes Note: Binary data will be displayed in HEX format.	Not supported
UNICODE	Unicode	Not supported

It's a lot to take in, so here are a few examples, you can also look at the example file at the end of this document.

Example:

```
FRUNAME “MPCHC001.FRU”

FRUADDRESS CHASSIS

FRUAREA CHASSIS

FRUFIELD S# // gets the value from the MPCHC001.FRU file


FRUAREA BOARD

FRUFIELD MN “Intel” // puts ‘Intel’ in the field

FRUFIELD S# “@stdin:ascii” // user types in the value


FRUAREA PRODUCT

// Gets the value from an environment variable “prod_ser_num”.

FRUFIELD S# “@env:ascii:prod_ser_num”

// Gets the value from the 4th line of the ASCII file ‘file_name’.

FRUFIELD P# “@file:ascii:file_name:4”
```

27.8.6 Input of Data

When taking in data from a file, an environment variable or from standard in, the input type is always ASCII. If in the configuration file, the type is specified as something other than ASCII, then it will be read in as ASCII and converted to the correct type. The maximum number of characters accepted as input does not change based on the input type. For example, if a maximum ASCII length for a field is 31 characters, then 31 input characters is also the maximum length for BCD, Binary, and ASCII6 input, therefore using different types does not allow more characters to be stored, but rather the same number of characters to be stored using fewer bytes.

BCD characters have a few special cases that need to be handled; this is how the utility should solve them. If only one or two BCD characters are entered, then they will not be accepted because when translated they take only one byte, and a field length of one byte is not allowed by spec. If an odd number of BCD characters are input, then they will be pre-appended with a zero in order to make the length even. Only even length BCD numbers are valid because two numbers are stored in each BCD byte.

Example 1:

BCD as entered in a FRU file:

99 12 34 56 78

Same BCD value as entered using an environment variable, file or stdin:

9912345678

Example 2:

Odd number of BCD characters entered.

12345

This input will be converted to “012345”

In the case where FRU fields are stored as UNICODE, fewer characters are stored. If 31 ASCII characters are allowed, then only 15 UNICODE characters will be permitted.

27.9 Display Commands

These commands display text to the user and do nothing else, they are: DISPLAY and CONFIGURATION. In order to display a blank line, you must display a space enclosed in quotes (“ ”). An empty string is not sufficient.

27.9.1 DISPLAY

The Display command displays a line of text to the user. If the text is more than one word, it must be encapsulated in double quotes. In order to display a CR-LF, you must use two separate DISPLAY commands. The text to display should be no longer than 80 ASCII characters to fit correctly on the screen, but the line to display can be of infinite length. In order to display a blank line, you must display a space enclosed in quotes (“ ”). An empty string is not sufficient.

Example:

```
DISPLAY “This is a line of text to display.”

DISPLAY “This is the second line of text to display.”

DISPLAY WORD
```

27.9.2 CONFIGURATION

The CONFIGURATION command is used to display the file configuration. It takes up to three parameters; the first parameter is a description of the file, such as “FRU configuration”. The second parameter is the target platform, this platform ID is not validated, so it is possible to have an incorrect target platform listed as part of this command. The third parameter is the file version; the version string has no defined format, and is displayed to the user as a string. This command requires only the first argument. It is recommended that the first non-comment line of all configuration file contain this command, so that users of the file understand the target of this file.

Example:

```
CONFIGURATION   “FRU Configuration” “MPCHC0001” “Version 5”
```

27.9.3 Input Commands

These commands require the user to input data for the process to continue. The input commands are: MENUTITLE/MENU/MENUPROMPT, PROMPT/YES/NO, and FRUFIELD. FRUFIELD has multiple input types and is covered in chapter Section 27.8.5. For the PROMPT and MENUTITLE commands to display a blank line, you must display a space enclosed in quotes (“ ”). An empty string is not sufficient.

27.9.4 MENU

The MENU command presents the user with a list of answers for a question. The maximum number of answers for a menu is 9, 1-9. The utility has the option to allow more menu items if desired, but the selection must be limited to a single key press. The MENU command allows two parameters. First is the tag to set if the option is chosen, and second is a prompt to show for that answer. The second parameter is not required; if no prompt is defined for the menu option, the tag string will be shown instead.

Example:

```
MENUTITLE   “What platform is this?”

MENU   “atca”   “Advanced TCA”
MENU   “cpci”   “Compact PCI”

MENU   “unknown”

MENUPROMPT


IFSET atca

// do atca stuff

ENDIF

IFSET cpci

// do cpci stuff

ENDIF

IFSET unknown

// quit the update with an error

ERRORLEVEL21

ENDIF
```

27.9.5 MENUTITLE

The MENUTITLE command displays a line of text before displaying the options for the menu. This command is optional, and if this command is used it must immediately precede a MENU command.

27.9.6 MENUPROMPT

The MENUPROMPT command signals the load utility that all the menu options are defined for a menu and user input needs to select an option. The utility shall limit the input to the specified range of menu options, and will not let the user proceed until a option is selected.

27.9.7 PROMPT

The PROMPT command is for asking the user for yes/no questions. The only valid input from the user for a prompt is ‘Y’, ‘y’, ‘N’, or ‘n’. The PROMPT command has two optional secondary commands: YES and NO. These commands define the tag to be set based on the yes or no answer. If the command for the answer does not exist, nothing happens. For instance answering no for the example has no effect.

Example:

```
PROMPT   “Is there a fan plugged into fan header #3?”

YES      “Fan3”
```

27.9.8 YES

The YES command requires that a PROMPT or NO command directly precedes it. It defines what tag is to be set when the user answers yes to the previous PROMPT. If preceded by a NO command, that NO command must be preceded by a PROMPT command.

Example:

```
PROMPT   “Is there a fan plugged into fan header #3?”

YES      “Fan3”

NO       “NO_Fan3”
```

27.9.9 NO

The NO command requires that a PROMPT or YES command directly precedes it. It defines what tag is to be set when the user answers no to the previous PROMPT. If preceded by a YES command, that YES command must be preceded by a PROMPT command.

Example:

```
PROMPT   “Is there a fan plugged into fan header #3?”

NO       “NO_Fan3”

YES      “Fan3”
```

27.10 Command Quick Reference

The following table gives a brief overview of the commands and their expected inputs, the probe commands are found in the next table.

Table 108. Command Quick Reference (Sheet 1 of 3)

Commands and Arguments	Description
//	Comment – indicates that all text following this command, up to the new line character, should be ignored.
CFGNAME	References a configuration file at this point during the read, process, and execution tasks.
ASCII String	Name of the CFG file to reference.
CLEAR	Clears the tag from the master tag list when the command is processed.
ASCII String	Tag string to clear from the master tag list.
CONFIGURATION	Descriptive name for the configuration file, this message is displayed to the user when the command is processed.
ASCII String	Description of the file’s purpose - will be displayed to the user.
ASCII String	Target platform - will be displayed to the user. (Optional)
ASCII String	Version information - will be displayed to the user. (Optional)
DISPLAY	Displays text to the user when processing the file.
ASCII String	Text to display to the user.

Table 108. Command Quick Reference (Sheet 2 of 3)

ELSE	Processes and executes the commands between the ELSE and the matching ENDIF command if the matching IFSET command failed to find all the tags in the master tag list.
ENDIF	Indicates the end of the matching IFSET or ELSE command’s affect.
ERRORLEVEL	Exits the load utility when processed, exiting the application with the specified error code.
Number	Error level to exit the application with.
FOUND	When processing the file, sets the TRUE or FALSE tag for the related PROBE command.
ASCII String	Tag to set in master list if TRUE was returned from the probe.
ASCII String	Tag to set in master list if FALSE was returned from the probe. (Optional)
FRUADDRESS	**Format 1: 1 argument.** Changes the destination address of the FRU file loaded by the last FRUNAME command.
Number	IPMB address of the FRU, use device id of 0.
FRUADDRESS	**Format 2: 1 arguments.** Same as format 1.
ASCII String	Location of target device or alias of device, use device ID of 0. Example: BLADE9, PEM1, CMM
FRUADDRESS	**Format 3: 2 arguments.** Same as format 1.
ASCII String	Location of target device or alias of device. Example: BLADE9, PEM1, CMM
Number	Device ID of FRU
FRUADDRESS	**Format 4: 2 arguments.** Same as format 1.
Number	IPMB address of the FRU.
Number	Device ID of FRU
FRUADDRESS	**Format 5: 3 arguments.** Same as format 1.
ASCII String	Location of target device or alias of device. Example: BLADE9, PEM1, CMM
Number	Device ID of FRU
Number	Bus number of device.
FRUADDRESS	**Format 6: 3 arguments.** Same as format 1.
Number	IPMB address of the FRU.
Number	Device ID of FRU
Number	Bus number of device.
FRUAREA	**Format 1: 1 arguments.** Indicates what FRU area the following FRUFIELD commands are to modify.
ASCII String	Area of the FRU to work with.
FRUAREA	**Format 2: 5-7 arguments.** Indicates how to modify the multi-record area of the FRU.
“MULTIREC”	Area of the FRU to work with.
ASCII String	Multi record operation to perform: ‘APPEND’, ‘INSERT’, ‘REMOVE’, ‘REPLACE’
ASCII String	Multi record select method: ‘PICMG_ID’, ‘RECORD_ID’
Number	Destination multi record ID
Number	Destination multi record ID number. When multiple records have the same ID, this identifies which number of the matching records to select.
Number	Source multi record ID, *not allowed with REMOVE operation.
Number	Source multi record ID number. When multiple records have the same ID, this identifies which number of the matching records to select, *not allowed with REMOVE operation.

Table 108. Command Quick Reference (Sheet 3 of 3)

FRUFIELD	Specifies what field of the selected FRU area to modify and what or where the data is.
ASCII String	FRU field name, see Table 104, “FRU Field First String Specifications” on page 237.
ASCII String	Optional argument to indicate the FRU field input or where the input should be redirected from. If not specified, the input will come from the FRU file specified by FRUNAME. See Section 27.8.1 for redirection information.
FRUNAME	Specifies the designated FRU file to use for updating the selected fields.
ASCII String	Name of the FRU file to use as the source of the FRU update. Alternatively, “SYSTEM” can be used to indicate the system FRU is to be used as the source.
ASCII String starts with “i:”	The file name of the binary image to use instead of reading from the system. (Optional)
ASCII String starts with “o:”	The file name of the binary image to use instead of writing to the system. (Optional)
IFSET	Processes and executes the commands between the IFSET and the matching ELSE or ENDIF command if all the tags listed after the command are found in the master tag list.
ASCII String(s) (1 +)	One or more tags to check for in the master tag list, implicit AND with the tags.
MENU	Specifies a tag and message to be displayed to the user as part of a multiple choice question during the processing of the file.
ASCII String	Tag to set in the master tag list if the user selects this MENU option.
ASCII String	String to display for this menu choice, if this is not the tag string is shown instead. (Optional)
MENUPROMPT	Indicates the end of the menu options and the utility needs to prompt the user for an answer before the utility can continue processing the file. A valid answer is needed to continue.
MENUTITLE	Optional command that immediately precedes the MENU command. This displays a message before the MENU strings when processing the file.
ASCII String	Text to display to the user.
NO	Sets a tag if the user answered ‘n’ to the preceding PROMPT command.
ASCII String	Tag to set in the master tag list.
PROBE	Probes the system for configuration information when processing the file, see Table 109, “Probe Arguments Quick Reference” on page 246.
PROMPT	Displays text to the user during processing of the file and forces the user to enter ‘y’ or ‘n’.
ASCII String	Text to display to the user.
SET	Sets the tag in the master tag list when the command is processed.
ASCII String	Tag string to set in the master tag list.
YES	Sets a tag if the user answered ‘y’ to the preceding PROMPT command.
ASCII String	Tag to set in the master tag list.

Table 109 gives a brief overview of the various PROBE arguments and their expected inputs. All arguments in this table follow the PROBE command on the same line in the file.

Table 109. Probe Arguments Quick Reference

Probe Arguments	Description
BMCVER	**Format 1: 1+ arguments.** Probes the system for the BMC operational code version and verifies that if falls within the range of versions supplied in the arguments.
ASCII String (1+)	BMC version ranges to check against the system version. Three formats: ‘X’ – one version, ‘X-Y’ – inclusive range, and ‘X+’ – inclusive infinite range.
BMCVER	**Format 2: 2+ arguments.** Probes the system for the BMC boot code version and verifies that if falls within the range of versions supplied in the arguments.
“**BOOT**”	Key word signaling that this probe is for the boot code version.
ASCII String (1+)	BMC version ranges to check against the system version. Three formats: ‘X’ – one version, ‘X-Y’ – inclusive range, and ‘X+’ – inclusive infinite range.
FRUVER	Probes the current FRUNAME or FRUADDRESS for the FRU’s board ‘FRU file ID’ value and verifies that if falls within the range of versions supplied in the arguments.
ASCII String (1+)	FRU version ranges to check against the system version. Three formats: ‘X’ – one version, ‘X-Y’ – inclusive range, and ‘X+’ – inclusive infinite range.
SYSTEM	Probes the current FRUNAME or FRUADDRESS for the FRU’s board ‘Part Number’ value and verifies that it matches the length and value (case insensitive) of the next parameter.
ASCII String	String to compare to the board ‘Part Number’.

27.11 Example Configuration File

The following three configuration files are an example package that could be used to update multiple systems to various versions the FRU (upgrade / downgrade). The file in the ‘master’ section is the one launched by the utility; the other files are referenced by the ‘master’. The menu prompts the user for which version of the FRU to update to, and proceeds to call the version update regarding that choice.

27.11.1 Chassis Update Version 0

```
CONFIGURATION "CDM FRU update" "MPCHC0000" "Ver 4.x"


// Set the FRU name so we can probe the FRU version info

FRUNAME     "MPCHC000.fru"

FRUADDRESS  chassis


PROBE SYSTEM CHUGACH

FOUND SYS_CORRECT

PROBE SYSTEM MPCHC0001

FOUND SYS_CORRECT
```

```
IFSET SYS_CORRECT

// Skip the else for the wrong system

ELSE

DISPLAY"Incorrect platform for FRU update"

// This will exit the file

ERRORLEVEL 14

ENDIF


//
*************************************************************************

// Validate the current version of the FRU is something we know about, if
not

// exit the update process with instructions on how to go from here.


// Later version than current update

PROBE FRUVER "108+"

FOUND FRUVER_LATER


IFSET FRUVER_LATER

DISPLAY "CDM FRU version is later than all known versions, no update will
be performed."

DISPLAY "Run the FRU update associated with the version of CMM firmware
running."

ERRORLEVEL 15

ENDIF


// Earlier version than all known

PROBE FRUVER "0-102"

FOUND FRUVER_EARLIER


IFSET FRUVER_EARLIER
```

```
DISPLAY "CDM FRU version is earlier than all known versions, no update can
be performed."

ERRORLEVEL 15

ENDIF

//
**************************************************************************



PROBE FRUVER "103"

FOUND FRUVER_CORRECT



PROBE FRUVER ""

FOUND FRUVER_NEEDED



IFSET FRUVER_CORRECT

DISPLAY "CDM FRU information is up to date, no update will be performed"

// error level 0 will exit now with successful return code

ERRORLEVEL 0

ELSE

IFSET FRUVER_NEEDED

// just update the version number

FRUAREA     "BOARD"

FRUFIELD    "ID"

ELSE

// need to do a downgrade - mark it

SET FRU_DOWNGRADE

ENDIF

ENDIF



IFSET FRU_DOWNGRADE

// update the FRU version

FRUAREA     "BOARD"
```

```
FRUFIELD    "ID"

// replace the first picmg power distribution record
FRUAREA "MULTIREC” REPLACE PICMG_ID 0x11 1 0x11 1
// remove the 2nd picmg power record
FRUAREA "MULTIREC" REMOVE PICMG_ID 0x11 2
// replace the shelf power management record
FRUAREA "MULTIREC" REPLACE PICMG_ID 0x12 1 0x12 1
// replace the shelf IP connection record
FRUAREA "MULTIREC" REPLACE PICMG_ID 0x13 1 0x13 1
// remove the ipmb link entry record to the end of the records
FRUAREA "MULTIREC” REMOVE PICMG_ID 0x15 1
ENDIF
```

27.11.2 Chassis Update Version 1

```
CONFIGURATION "CDM FRU update" "MPCHC0001" "Ver 5.1 and Ver 5.2"

// Set the FRU name so we can probe the FRU version info
FRUNAME     "MPCHC001.fru"
FRUADDRESS chassis

PROBE SYSTEM CHUGACH
FOUND SYS_CORRECT
PROBE SYSTEM MPCHC0001
FOUND SYS_CORRECT

IFSET SYS_CORRECT
// Skip the else for the wrong system
ELSE
DISPLAY "Incorrect platform for FRU update"
// This will exit the file
```

```
ERRORLEVEL 14

ENDIF


//
*************************************************************************

// Validate the current version of the FRU is something we know about, if
not

// exit the update process with instructions on how to go from here.


// Later version than current update

PROBE FRUVER "108+"

FOUND FRUVER_LATER


IFSET FRUVER_LATER

DISPLAY "CDM FRU version is later than the version to update to, no update
can be performed."

ERRORLEVEL 0

ENDIF


// Earlier version than all known

PROBE FRUVER "0-102"

FOUND FRUVER_EARLIER


IFSET FRUVER_EARLIER

DISPLAY "CDM FRU version is earlier than all known versions, no update can
be performed."

ERRORLEVEL 15

ENDIF

//
*************************************************************************


PROBE FRUVER ""

FOUND ALL_UPDATES
```

```
PROBE FRUVER "103"

FOUND ALL_UPDATES


PROBE FRUVER "104-105"

FOUND IPMB_LINK_UPDATE


// the update of the FRU to version 107 requires a power record update for
versions 104 and later

IFSET IPMB_LINK_UPDATE

SET POWER_UPDATE

ENDIF


// To go from 106 to 107, we need to update the power distribution records

PROBE FRUVER "106"

FOUND POWER_UPDATE


// Version 107 is the latest, no update required.

PROBE FRUVER "107"

FOUND FRUVER_107


IFSET FRUVER_107

DISPLAY "CDM FRU information is up to date, no update will be performed"

// error level 0 will exit now with successful return code

ERRORLEVEL 0

ELSE

// always update the FRU version

FRUAREA     "BOARD"

FRUFIELD    "ID"


IFSET ALL_UPDATES
```

```
// replace the first picmig power distribution record
FRUAREA "MULTIREC" REPLACE PICMG_ID 0x11 1 0x11 1
// append the new record after the first one
FRUAREA "MULTIREC" APPEND PICMG_ID 0x11 1 0x11 2
// replace the shelf power management record
FRUAREA "MULTIREC" REPLACE PICMG_ID 0x12 1 0x12 1
// replace the shelf IP connection record
FRUAREA "MULTIREC" REPLACE PICMG_ID 0x13 1 0x13 1
// append the ipmb link entry record to the end of the records
FRUAREA "MULTIREC” APPEND PICMG_ID 0x15 1
ENDIF

IFSET IPMB_LINK_UPDATE
// replace the ipmb link entry record
FRUAREA "MULTIREC" REPLACE PICMG_ID 0x15 1 0x15 1
ENDIF

IFSET POWER_UPDATE
// replace the first picmig power distribution record
FRUAREA "MULTIREC" REPLACE PICMG_ID 0x11 1 0x11 1
// replace the second picmig power distribution record
FRUAREA "MULTIREC" REPLACE PICMG_ID 0x11 2 0x11 2
ENDIF
ENDIF
```

Unrecognized Sensor Types 28

28.1 System Events Overview

When a System Event is recorded in the CMM's System Event Log (SEL), it contains 16 bytes. Below is an overview of the meaning of the bytes. For more information see Table 26-1 in the IPMI 1.5 specification.

Record ID – (2 bytes) The record ID of the SEL record.

Record Type – (1 byte) The type of the record. For 1.5 the values are 02h = System Event Record, C0h – DFh = OEM Timestamped, and E0h – FFh = OEM non-timestamped. However, for future proofing, the CMM must be able to report any record type from 00h – FFh. The ability to translate bytes below this point only applies to System Event Record and OEM Timestamped records. For OEM non-timestamped, all remaining bytes are OEM specified.

Timestamp – (4 bytes) The time the event was received by the CMM. The ability to translate bytes below this point only applies to System Event Record. For OEM timestamped, all remaining bytes are OEM specified.

Generator ID – (2 bytes) Specifies what entity in the shelf generated the event. Normally this will be the IPMB address of the entity.

EvM Format – (1 byte) The event message format. The CMM supports 03h and 04h formats.

Sensor Type – (1 byte) Sensor Type Code for the sensor that generated the event. See Table 36-3 in the IPMI 1.5 specification for the Sensor Type Codes.

Sensor Number – (1 byte) The number of the sensor that generated the event.

Event Dir – (high bit of 1 byte) Defines if the event is being deasserted (1b) or asserted (0b).

Event Type Code – (low 7 bits of 1 byte) Type of trigger for the event. There are three general categories: A) Generic, B) Sensor Specific, and C) OEM. The Generic category has values of 01h – 0Ch. The Sensor Specific is defined as 6Fh. The OEM range is 70h – 7Fh. However, for future proofing, the CMM must be able to report any Event Type from 00h to FFh. See Sections 36-1, 36-2, and 36-3 of IPMI 1.5 spec for more information.

Event Data 1 – (1 byte) How to interpret the bits in this byte depends on the class of the sensor. There are three general classes: A) Threshold, B) Discrete, and C) OEM. Generally speaking, the high nibble of this byte defines what type of data is in Event Data 2 and 3. Bits 7:6 define Data 2 usage and 5:4 define Data 3 usage. A value of 00b means that data byte is not used. A value of 01b for threshold sensors means that the data byte 2 or 3 contains the trigger reading or threshold value, respectively. A value of 01b for a discrete or OEM class sensor means that data byte 2 contains the previous state/severity and for data byte 3 it is a reserved value. A value of 10b means that the respective data byte contains OEM data. A value of 11b is reserved for OEM sensors and for the other two means that a sensor-specific offset is held in the data byte.

The low nibble defines an offset within the threshold, discrete, or OEM tables.

Event Data 2 and 3 – (2 bytes) Event data field contents which varies depending on how Event Data 1's high nibble is set.

The CMM uses the above 16 bytes of data from a SEL entry to produce human readable output. There are two general categories that an event can fall into: 1) data that the CMM code base recognizes, and 2) data that is unrecognized by the CMM.

For unrecognized events the CMM does not have enough encoded knowledge to translate the event. For instance, all events with a **Record Type** of OEM timestamped or non-timestamped fall into this category. In addition, any standard IPMI events that are not currently supported by the CMM translation code also fall into this category.

Beginning in firmware version 5.2 the CMM has the ability to display and trap both recognized and unrecognized events.

28.2 System Events— SNMP Trap Support

All CMM generated traps have a standard "header" that is shown below. The first four items (i.e., Time, Location, Chassis Serial #, and Board) represent information that does not necessarily have to come from the event itself. These pieces of information are helpful in tracing the trap back to its source.

28.2.1 SNMP Trap Header Format

```
Time : TimeStamp , Location : ChassisLocation , Chassis Serial # :
ChassisSerialNumber , Board : Location
```

Where:

- ***TimeStamp*** is in the format: [Day] [Month] [Date] [HH:MM:SS] [Year]. For example, Thu Apr 14 22:20:03 2005
- ***ChassisLocation*** is the chassis location information recorded in the chassis FRU.
- ***ChassisSerialNumber*** is the chassis serial number recorded in the chassis FRU.
- ***Location*** is the location where the sensor generating the event is located (i.e., CMM)

The next portion can be controlled by a CMM variable to turn it on or off. This section provides the text interpretation of the event.

28.2.2 SNMP Trap ATCA Trap Text Translation Format

```
Sensor : SDRSensorName , Event : HealthEventString , Event Code :
EventCodeNumber
```

Where:

- **SDRSensorName:** The name given to the sensor in the Sensor Data Record (SDR).
- **HealthEventString:** The CMM's translation of the event.
- **EventCodeNumber:** A hexadecimal number that uniquely defines the event. This will be in the format x%X.

The final portion that an SNMP Trap might contain is the "raw" portion of the trap. This reports the original sixteen bytes of the System Event as ASCII upper case hex bytes.

```
Raw Hex : [ 12 34 56 78 9A … (16 bytes hex) ]
```

28.3 SNMP Trap Raw Format

SNMPTrapFormat, in the cmm.cfg file controls whether the “text” portion or the “raw” portion of a trap is sent along with the “header”. The header is always sent. If the SNMPTrapFormat variable is equal to 1 (text) the output is header plus text. The next figure shows what the output will look like.

Figure 6. SNMPTrapFormat = 1

```
Time : TimeStamp , Location : ChassisLocation , Chassis Serial # :
ChassisSerialNumber , Board : Location , Sensor : SDRSensorName , Event :
HealthEventString , Event Code : EventCodeNumber
```

If SNMPTrapFormat is 2 (raw), the output is header plus raw. The next figure shows the output.

Figure 7. SNMPTrapFormat = 2

```
Time : TimeStamp , Location : ChassisLocation , Chassis Serial # :
ChassisSerialNumber , Board : Location , Raw Hex : [ 12 34 56 78 9A … (16
bytes hex) ]
```

A third value of 3 (text&raw) is used to indicate header, text, and raw. The example output is shown in the following table.

Figure 8. SNMPTrapFormat = 3

```
Time : TimeStamp , Location : ChassisLocation , Chassis Serial # :
ChassisSerialNumber , Board : Location , Sensor : SDRSensorName , Event :
HealthEventString , Raw Hex : [ 12 34 56 78 9A … (16 bytes hex) ]
```

28.3.1 SNMP Trap Control

To assist with backward compatibility a variable exists called SNMPSendUnrecognizedEvents. This controls whether the CMM should not send SNMP traps for unrecognized SEL events (a value of 0) or begin sending SNMP traps for unrecognized events (a value of 1). The default value is 0.

Table 110. Results of Variable Settings

	SNMPTrapFormat Control		
	1 (text)	2 (raw)	3 (text&raw)
Recognized Event	Standard output only	Header + Raw display	Standard output + Raw information. Helps in cases where the event is partially translated in the text portion.
Unrecognized Event	*SNMPSendUnrecognizedEvents =0* Not sent	Header + Raw display	Header, Text, and Raw output. The Text portion will simply state that the CMM could not translate the event.
	SNMPSendUnrecognizedEvents =1 Useful in allowing the user to see that there are unrecognized events; however, it does not give enough information to understand the event.		

28.3.2 System Events— SEL Support

When a user lists the contents of a SEL with the *cmmget* command, the format for the SEL has three possible parts: the header, the translated text, and the raw output.

All CMM displayed SEL entries have a standard header that is shown below. For the SEL, this includes only the Timestamp followed by a newline.

28.3.2.1 SEL Header Format

`TimeStamp``\n`

TimeStamp has two possibilities:

1. A SEL event that has a timestamp (recognized System Event Records and OEM timestamped events). It is in the format: [Day] [Month] [Date] [HH:MM:SS] [Year]. For example, Thu Apr 14 22:20:03 2005.
2. OEM non-timestamped sensors. That format will be: Date/time unknown.

28.3.2.2 SEL Text Translation Format

The next portion of the SEL entry can be enabled or disabled as described later in this section. This provides the text interpretation of the event. Its format is shown below:

```
\tSDRSensorName\tHealthEventString
```

Where:

- ***SDRSensorName*** is the name given to the sensor in the Sensor Data Record (SDR).
- ***HealthEventString*** is the CMM's translation of the event.

28.3.2.3 SEL Raw Format

The final portion that SEL display might contain is the "raw" portion of the trap. This reports the original sixteen bytes of the System Event as ASCII, upper case, hex bytes.

```
\tRaw Hex : [ 12 34 56 78 9A … (16 bytes hex) ]
```

Notice the square brackets have a space after the open bracket and before the close bracket. This is intended to help parsing for any scripting languages.

At the end of the SEL display, there is always two trailing newlines.

28.3.3 Configuring SEL Format

When the command "*cmmget [–l location] –d sel*" used, a variable SELFormat in the cmm.cfg file is used. This variable controls whether the "text" portion or the "raw" portion of the SEL display is output along with the "header". The header is always displayed.

28.3.3.1 SELFormat = 1 (text) Example Output

If the SELFormat variable is equal to 1 (text) the output is header plus text.

```
TimeStamp\n

\tSDRSensorName\tHealthEventString\n\n
```

28.3.3.2 SELFormat = 2 (raw) Example Output

If SELFormat is 2 (raw), the output is as below.

```
TimeStamp\n

\tRaw Hex : [ 12 34 56 78 9A … (16 bytes hex) ]\n\n
```

If SELFormat is 3 (text&raw), the example output is shown below:

28.3.3.3 SELFormat = 3 (raw&text) Example Output

```
TimeStamp\n

\tSDRSensorName\tHealthEventString\tRaw Hex : [ 12 34 56 78 9A … (16 bytes
hex) ]\n\n
```

28.3.3.4 System Events – SEL Display Control

In older firmware, the CMM always displays events for recognized events and never displays unrecognized events. To assist with backward compatibility a new configuration variable has been added, SELDisplayUnrecognizedEvents, that controls whether the CMM should continue with the old behavior (a value of 0) or begin displaying unrecognized events as well (a value of 1).

Warranty Information 29

29.1 Intel[®] NetStructure™ Compute Boards and Platform Products Limited Warranty

Intel warrants to the original owner that the product delivered in this package will be free from defects in material and workmanship for two (2) year(s) following the latter of: (i) the date of purchase only if you register by returning the registration card as indicated thereon with proof of purchase; or (ii) the date of manufacture; or (iii) the registration date if by electronic means provided such registration occurs within 30 days from purchase. This warranty does not cover the product if it is damaged in the process of being installed. Intel recommends that you have the company from whom you purchased this product install the product.

THE ABOVE WARRANTY IS IN LIEU OF ANY OTHER WARRANTY, WHETHER EXPRESS, IMPLIED OR STATUTORY, INCLUDING, BUT NOT LIMITED TO, ANY WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, ANY WARRANTY OF INFRINGEMENT OF ANY OTHER PARTY'S INTELLECTUAL PROPERTY RIGHTS, OR ANY WARRANTY ARISING OUT OF ANY PROPOSAL, SPECIFICATION OR SAMPLE.

This warranty does not cover replacement of products damaged by abuse, accident, misuse, neglect, alteration, repair, disaster, improper installation or improper testing. If the product is found to be otherwise defective, Intel, at its option, will replace or repair the product at no charge except as set forth below, provided that you deliver the product along with a return material authorization (RMA) number (see below) either to the company from whom you purchased it or to Intel. If you ship the product, you must assume the risk of damage or loss in transit. You must use the original container (or the equivalent) and pay the shipping charge. Intel may replace or repair the product with either a new or reconditioned product, and the returned product becomes Intel's property. Intel warrants the repaired or replaced product to be free from defects in material and workmanship for a period of the greater of: (i) ninety (90) days from the return shipping date; or (ii) the period of time remaining on the original two (2) year warranty.

This warranty gives you specific legal rights and you may have other rights which vary from state to state. All parts or components contained in this product are covered by Intel's limited warranty for this product. The product may contain fully tested, recycled parts, warranted as if new.

29.2 Returning a Defective Product (RMA)

Before returning any product, contact an Intel Customer Support Group to obtain either a Direct

Return Authorization (DRA) or Return Material Authorization (RMA). Return Material

Authorizations are only available for products purchased within 30 days.

Return contact information by geography:

29.3 For the Americas

Return Material Authorization (RMA) credit requests e-mail address: requests.rma@intel.com

Direct Return Authorization (DRA) repair requests e-mail address: uspss.repair@intel.com

DRA on-line form: http://support.intel.com/support/motherboards/draform.htm

Intel Business Link (IBL): http://www.intel.com/ibl

Telephone No.: 1-800-INTEL4U or 480-554-4904

Office Hours: Monday - Friday 0700-1700 MST Winter / PST Summer

29.3.1 For Europe, Middle East, and Africa (EMEA)

Return Material Authorization (RMA) e-mail address - emea.fs@intel.com

Direct Return Authorization (DRA) for repair requests e-mail address: emea.fs@intel.com

Intel Business Link (IBL): http://www.intel.com/ibl

Telephone No.: 00 44 1793 403063

Fax No.: 00 44 1793 403109

Office Hours: Monday - Friday 0900-1700 UK time

29.3.2 For Asia and Pacific (APAC)

RMA/DRA requests email address: apac.rma.front-end@intel.com

Telephone No.: 604-859-3111 or 604-859-3325

Fax No.: 604-859-3324

Office Hours: Monday - Friday 0800-1700 Malaysia time

Return Material Authorization (RMA) requests e-mail address: rma.center.jpss@intel.com

Telephone No.: 81-298-47-0993 or 81-298-47-5417

Fax No.: 81-298-47-4264

Direct Return Authorization (DRA) for repair requests, contact the JPSS Repair center.

E-mail address: sugiyamakx@intel.co.jp

Telephone No.: 81-298-47-8920

Fax No.: 81-298-47-5468

Office Hours: Monday - Friday 0830-1730 Japan time

If the Customer Support Group verifies that the product is defective, they will have the Direct Return Authorization/Return Material Authorization Department issue you a DRA/RMA number to place on the outer package of the product. Intel cannot accept any product without a DRA/RMA number on the package. Limitation of Liability and Remedies

INTEL SHALL HAVE NO LIABILITY FOR ANY INDIRECT OR SPECULATIVE DAMAGES (INCLUDING WITHOUT LIMITING THE FOREGOING, CONSEQUENTIAL, INCIDENTAL AND SPECIAL DAMAGES) ARISING FROM THE USE OF OR INABILITY TO USE THIS PRODUCT, WHETHER ARISING OUT OF CONTRACT, NEGLIGENCE, TORT, OR UNDER ANY WARRANTY, OR FOR INFRINGEMENT OF ANY OTHER PARTY'S INTELLECTUAL PROPERTY RIGHTS, IRRESPECTIVE OF WHETHER INTEL HAS ADVANCE NOTICE OF THE POSSIBILITY OF ANY SUCH DAMAGES, INCLUDING, BUT NOT LIMITED TO LOSS OF USE, BUSINESS INTERRUPTIONS, AND LOSS OF PROFITS. NOTWITHSTANDING THE FOREGOING, INTEL'S TOTAL LIABILITY FOR ALL CLAIMS UNDER THIS AGREEMENT SHALL NOT EXCEED THE PRICE PAID FOR THE PRODUCT. THESE LIMITATIONS ON POTENTIAL LIABILITIES WERE AN ESSENTIAL ELEMENT IN SETTING THE PRODUCT PRICE. INTEL NEITHER ASSUMES NOR AUTHORIZES ANYONE TO ASSUME FOR IT ANY OTHER LIABILITIES.

Some states do not allow the exclusion or limitation of incidental or consequential damages, so the above limitations or exclusions may not apply to you.

Customer Support 30

30.1 Customer Support

This chapter offers technical and sales assistance information for this product. Information on returning an Intel® NetStructure™ product for service is in the following chapter.

30.2 Technical Support and Return for Service Assistance

For all product returns and support issues, please contact your Intel product distributor or Intel Sales Representative for specific information.

30.3 Sales Assistance

If you have a sales question, please contact your local Intel NetStructure Sales Representative or the Regional Sales Office for your area. Address, telephone and fax numbers, and additional information is available at the following web site:

http://www.intel.com/network/csp/sales/

Intel Corporation
Telephone (in U.S.) 1-800-755-4444
Telephone (Outside U.S.) 1-973-993-3030
FAX 1-973-967-8780

Certifications 31

The Intel® NetStructure™ MPCMM0001 Chassis Management Module has the following approvals:

- UL/cUL 60950
- EN/IEC 60950
- EN55022 Class A
- EN55024
- FCC CFR47 Part 15 Class A
- VCCI
- AS/NZS3548
- BSMI

Agency Information 32

32.1 North America (FCC Class A)

FCC Verification Notice

This device complies with Part 15 of the FCC Rules. Operation is subject to the following two conditions: (1) this device may not cause harmful interference, and (2) this device must accept any interference received, including interference that may cause undesired operation.

For questions related to the EMC performance of this product, contact:

Intel Corporation
5200 N.E. Elam Young Parkway
Hillsboro, OR 97124
1-800-628-8686

This equipment has been tested and found to comply with the limits for a Class A digital device, pursuant to Part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment. This equipment generates, uses, and can radiate radio frequency energy if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. Operation of this equipment in a residential area is likely to cause harmful interference in which case the use will be required to correct the interference at his own expense.

32.2 Canada – Industry Canada (ICES-003 Class A) (English and French-translated below)

CANADA – INDUSTRY CANADA

Cet appareil numérique respecte les limites bruits radioélectriques applicables aux appareils numériques de Classe A prescrites dans la norme sur le matériel brouilleur: “Appareils Numériques”, NMB-003 édictée par le Ministre Canadian des Communications.

(English translation of the notice above) This digital apparatus does not exceed the Class A limits for radio noise emissions from digital apparatus set out in the interference-causing equipment standard entitled “Digital Apparatus,” ICES-003 of the Canadian Department of Communications.

32.3 Safety Instructions (English and French-translated below)

32.3.1 English

CAUTION: This equipment is designed to permit the connection of the earthed conductor of the d.c. supply circuit to the earthing conductor at the equipment. See installation instructions. If this connection is made, all of the following conditions must be met:

-This equipment shall be connected directly to the DC supply system earthing electrode conductor or to a bonding jumper from an earthing terminal bar or bus to which the DC supply system earthing electrode conductor is connected.

-This equipment shall be located in the same immediate area (such as adjacent cabinets) as any other equipment that has a connection between the earthed conductor of the same DC supply circuit and the earthing conductor, and also the point of earthing of the DC system. The DC system shall not be earthed elsewhere.

-The DC supply source shall be located within the same premises as this equipment.

-Switching or disconnecting devices shall be in the earthed circuit conductor between the DC source and the point of connection of the earthing electrode conductor.

32.3.2 French

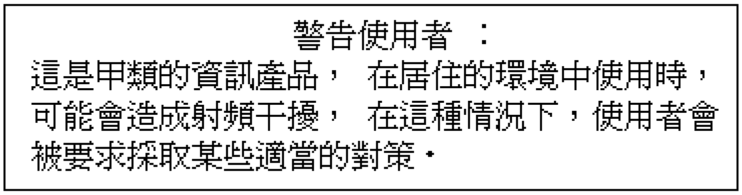
Cet appareil est conçu pour permettre le raccordement du conducteur relié à la terre du circuit d’alimentation c.c. au conducteur de terre de l’appareil. Cet appareil est conçu pour permettre le raccordement du conducteur relié à la terre du circuit d’alimentation c.c. au conducteur de terre de l’appareil. Pour ce raccordement, toutes les conditions suivantes doivent être respectées:

- Ce matériel doit être raccordé directement au conducteur de la prise de terre du circuit d’alimentation c.c. ou à une tresse de mise à la masse reliée à une barre omnibus de terre laquelle est raccordée à l’électrode de terre du circuit d’alimentation c.c.

- Les appareils dont les conducteurs de terre respectifs sont raccordés au conducteur de terre du même circuit d’alimentation c.c. doivent être installés à proximité les uns des autres (p.ex., dans des armoires adjacentes) et à proximité de la prise de terre du circuit d’alimentation c.c. Le circuit d’alimentation c.c. ne doit comporter aucune autre prise de terre. matériel. - Il ne doit y avoir

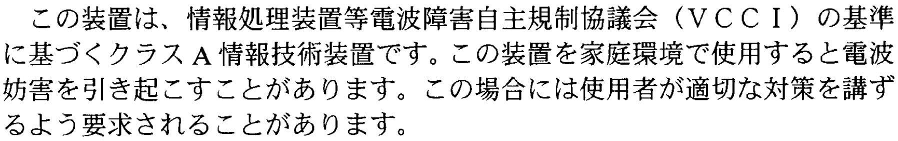
– La source d’alimentation du circuit c.c. doit être située dans la même pièce que le aucun dispositif de commutation ou de sectionnement entre le point de raccordement au conducteur de la source d’alimentation c.c. et le point de raccordement à la prise de terre.

32.4 Taiwan Class A Warning Statement



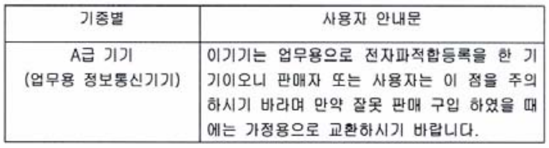
警告使用者 ：
這是甲類的資訊產品，在居住的環境中使用時，
可能會造成射頻干擾，在這種情況下，使用者會
被要求採取某些適當的對策•

32.5 Japan VCCI Class A



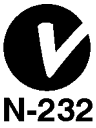
　この装置は、情報処理装置等電波障害自主規制協議会（ＶＣＣＩ）の基準に基づくクラスＡ情報技術装置です。この装置を家庭環境で使用すると電波妨害を引き起こすことがあります。この場合には使用者が適切な対策を講ずるよう要求されることがあります。

32.6 Korean Class A



기종별	사용자 안내문
A급 기기 (업무용 정보통신기기)	이기기는 업무용으로 전자파적합등록을 한 기기이오니 판매자 또는 사용자는 이 점을 주의하시기 바라며 만약 잘못 판매 구입 하였을 때에는 가정용으로 교환하시기 바랍니다.

32.7 Australia, New Zealand



Safety Warnings 33

Caution: Review the following precautions to avoid personal injury and prevent damage to this product or products to which it is connected. To avoid potential hazards, use the product only as specified.

Read all safety information provided in the component product user manuals and understand the precautions associated with safety symbols, written warnings, and cautions before accessing parts or locations within the unit. Save this document for future reference.

AC AND/OR DC POWER SAFETY WARNING: The AC and/or DC Power cord is the unit's main AC and/or DC disconnecting device, and must be easily accessible at all times. Auxiliary AC and/or DC On/Off switches and/or circuit breaker switches are for power control functions only (NOT THE MAIN DISCONNECT).

IMPORTANT: See installation instructions before connecting to the supply.

For AC systems, use only a power cord with a grounded plug and always make connections to a grounded main. Each power cord must be connected to a dedicated branch circuit.

For DC systems, this unit relies on the building's installation for short circuit (over-current) protection. Ensure that a Listed and Certified fuse or circuit breaker no larger than 72VDC, 15A is used on all current carrying conductors. For permanently connected equipment, a readily accessible disconnect shall be incorporated in the building installation wiring. For permanent connections, use copper wire of the gauge specified in the system's user manual.

The enclosure provides a separate Earth ground connection stud. Make the Earth ground connection prior to applying power or peripheral connections and never disconnect the Earth ground while power or peripheral connections exist.

To reduce the risk of electric shock from a telephone or Ethernet* system, connect the unit's main power before making these connections. Disconnect these connections before removing main power from the unit.

RACK MOUNT ENCLOSURE SAFETY: This unit may be intended for stationary rack mounting. Mount in a rack designed to meet the physical strength requirements of NEBS GR-63-CORE and NEBS GR 487. Disconnect all power sources and external connections prior to installing or removing the unit from a rack.

System weight may be minimized prior to mounting by removing all hot-swappable equipment. Mount your system in a way that ensures even loading of the rack. Uneven weight distribution can result in a hazardous condition. Secure all mounting bolts when rack mounting the enclosure.

Warning: Verify power cord and outlet compatibility: Use the appropriate power cords for your power outlet configurations. Visit the following web site for additional information: http://kropla.com/electric2.htm.

Warning: Avoid electric overload, heat, shock, or fire hazard: Only connect the system to a to a properly rated supply circuit as specified in the product user manual. Do not make connections to terminals outside the range specified for that terminal. See the product user manual for correct connections.

Warning: Avoid electric shock: Do not operate in wet, damp, or condensing conditions. To avoid electric shock or fire hazard, do not operate this product with enclosure covers or panels removed.

Warning: Avoid electric shock: For units with multiple power sources, disconnect all external power connections before servicing.

Warning: Power supplies must be replaced by qualified service personnel only.

Caution: System environmental requirements: Components such as Processor Boards, Ethernet Switches, etc., are designed to operate with external airflow. Components can be destroyed if they are operated without external airflow. External airflow is normally provided by chassis fans when components are installed in compatible chassis. Never restrict the airflow through the unit's fan or vents. Filler panels or air management boards must be installed in unused chassis slots. Environmental specifications for specific products may differ. Refer to product user manuals for airflow requirements and other environmental specifications.

Warning: Device heatsinks may be hot during normal operation: To avoid burns, do not allow anything to touch heatsinks.

Warning: Avoid injury, fire hazard, or explosion: Do not operate this product in an explosive atmosphere.

Caution: Lithium batteries. There is a danger of explosion if a battery is incorrectly replaced or handled. Do not disassemble or recharge the battery. Do not dispose of the battery in fire. When the battery is replaced, the same type (CR2032) or an equivalent type recommended by the manufacturer must be used. Used batteries must be disposed of according to the manufacturer's instructions.

Warning: Avoid injury: This product may contain one or more laser devices that are visually accessible depending on the plug-in modules installed. Products equipped with a laser device must comply with International Electrotechnical Commission (IEC) 60825.

33.1 Mesures de Sécurité

Veuillez suivre les mesures de sécurité suivantes pour éviter tout accident corporel et ne pas endommager ce produit ou tout autre produit lui étant connecté. Pour éviter tout danger, veillez à utiliser le produit conformément aux spécifications mentionnées.

Lisez toutes les informations de sécurité fournies dans les manuels de l'utilisateur des produits composants et veillez à bien comprendre les mesures associées aux symboles de sécurité, aux avertissements écrits et aux mises en garde avant d'accéder à certains éléments ou emplacements de l'unité. Conservez ce document comme outil de référence.

AVERTISSEMENT CONCERNANT LA SÉCURITÉ DE L'ALIMENTATION C.A. ET/OU C.C. : le câble d'alimentation C.A. et/ou C.C. constitue le dispositif de déconnexion principal de l'alimentation électrique de l'unité et doit être facilement accessible à tous moments. Les commutateurs de marche/arrêt C.A. et/ou C.C. et/ou les commutateurs disjoncteurs auxiliaires permettent uniquement de contrôler l'alimentation (ET NON LA DÉCONNEXION PRINCIPALE).

IMPORTANT : reportez-vous aux instructions d'installation avant de connecter le bloc d'alimentation.

Pour les systèmes C.A., utilisez uniquement un câble d'alimentation avec une prise de terre et établissez toujours les connexions à une prise secteur mise à la terre. Chaque câble d'alimentation doit être connecté à un circuit terminal dédié.

Pour les systèmes C.C., la protection de cette unité repose sur les coupe-circuits (surintensité) du bâtiment. Assurez-vous d'utiliser un fusible ou un disjoncteur répertorié et certifié ne dépassant pas 72 VCC et 15 A pour tous les conducteurs de courant. Pour les équipements connectés en permanence, un sectionneur facilement accessible doit être incorporé au câblage du bâtiment. Pour les connexions permanentes, utilisez des câbles en cuivre d'un calibre conforme à celui spécifié dans le manuel de l'utilisateur du système.

Le boîtier fournit un connecteur de mise à la terre séparé. Établissez la connexion à la terre avant de mettre le système sous tension ou de connecter des périphériques. Veillez à ne jamais déconnecter la mise à la terre tant que le système est sous tension ou si des périphériques sont connectés.

Pour réduire le risque d'un choc électrique en provenance d'un téléphone ou d'un système Ethernet*, connectez l'alimentation principale de l'unité avant d'établir ces connexions. De même, déconnectez-les avant de couper l'alimentation principale de l'unité.

SÉCURITÉ DU BOÎTIER POUR UN MONTAGE EN BAIE : cette unité peut être destinée à un montage en baie stationnaire. Le montage en baie doit satisfaire aux exigences sur la résistance physique des normes NEBS GR-63-CORE et NEBS GR 487. Déconnectez toutes les sources d'alimentation et les connexions externes avant d'installer ou de supprimer l'unité d'une baie.

Minimisez la masse du système avant le montage en retirant l'équipement permutable à chaud. Assurez-vous que le système est réparti de manière uniforme sur la baie. Une distribution inégale de la masse du système peut présenter des risques. Fixez tous les boulons lors de l'installation du boîtier dans une baie.

Avertissement : vérifiez que le câble d'alimentation et la prise sont compatibles. Utilisez les câbles d'alimentation correspondant à la configuration de vos prises de courant. Pour de plus amples informations, visitez le site Web suivant : http://kropla.com/electric2.htm.

Avertissement : évitez toute forme de surcharge, chaleur, choc électrique ou incendie. Connectez uniquement le système à un circuit d'alimentation dûment répertorié conformément aux spécifications du manuel de l'utilisateur du produit. N'établissez pas de connexions à des terminaux en dehors des limites spécifiées pour ce terminal. Reportez-vous au manuel de l'utilisateur du produit pour les connections adéquates.

Avertissement : évitez les chocs électriques. N'utilisez pas ce produit dans des endroits humides, mouillés ou provoquant de la condensation. Pour éviter tout risque de choc électrique ou d'incendie, n'utilisez pas ce produit si les couvercles ou les panneaux du boîtier ne sont pas en place.

Avertissement : évitez les chocs électriques. Pour les unités comportant plusieurs sources d'alimentation, déconnectez toutes les sources d'alimentation externes avant de procéder aux réparations.

Avertissement : les blocs d'alimentation doivent être remplacés exclusivement par des techniciens d'entretien qualifiés.

Attention : exigences environnementales du système : les composants tels que les cartes de processeurs, les commutateurs Ethernet, etc., sont conçus pour fonctionner avec un flux d'air externe. Les composants peuvent être détruits s'ils fonctionnent dans d'autres conditions. Le flux d'air externe est généralement produit par les ventilateurs des châssis lorsque les composants sont installés dans des châssis compatibles. Veillez à ne jamais obstruer le flux d'air alimentant le ventilateur ou les conduits de l'unité. Des boucliers ou des panneaux de gestion de l'air doivent être installés dans les connecteurs inutilisés du châssis. Les spécifications environnementales peuvent varier d'un produit à un autre. Veuillez-vous reporter au manuel de l'utilisateur pour déterminer les exigences en matière de flux d'air et d'autres spécifications environnementales.

Avertissement : les dissipateurs de chaleur de l'appareil peuvent être chauds lors d'un fonctionnement normal. Pour éviter tout risque de brûlure, veillez à ce que rien n'entre en contact avec les dissipateurs de chaleur.

Avertissement : évitez les blessures, les incendies ou les explosions. N'utilisez pas ce produit dans une atmosphère présentant des risques d'explosion.

Attention : les batteries au lithium. Celles-ci peuvent exploser si elles sont incorrectement remplacées ou manipulées. Veillez à ne pas désassembler ni à recharger la batterie. Veillez à ne pas jeter la batterie au feu. Lors du remplacement de la batterie, utilisez le même type de batterie (CR2032) ou un type équivalent recommandé par le fabricant. Les batteries usagées doivent être mises au rebut conformément aux instructions du fabricant.

Avertissement : évitez les blessures. Ce produit peut contenir un ou plusieurs périphériques laser visuellement accessibles en fonction des modules plug-in installés. Les produits équipés d'un périphérique laser doivent être conformes à la norme IEC (International Electrotechnical Commission) 60825.

33.2 Sicherheitshinweise

⚠ Lesen Sie bitte die folgenden Sicherheitshinweise, um Verletzungen und Beschädigungen dieses Produkts oder der angeschlossenen Produkte zu verhindern. Verwenden Sie das Produkt nur gemäß den Anweisungen, um mögliche Gefahren zu vermeiden.

Lesen Sie alle Sicherheitsinformationen in den Benutzerhandbüchern der zu dem Produkt gehörenden Komponenten und machen Sie sich mit den Hinweisen zu den Sicherheitssymbolen, schriftlichen Warnungen und Vorsichtsmaßnahmen vertraut, ehe Sie Teile oder Stellen des Geräts anfassen. Bewahren Sie dieses Dokument gut auf, um später darin nachlesen zu können.

SICHERHEITSWARNUNG FÜR WECHSELSTROM UND/ODER GLEICHSTROM: Die Stromversorgung des Gerätes wird über das Wechselstrom- und/oder Gleichstromkabel unterbrochen und muss daher jederzeit leicht zugänglich sein. Zusätzliche Ein-/Aus-Schalter für Wechselstrom und/oder Gleichstrom und/oder Leistungsschalter dienen lediglich der Steuerung der Stromversorgung (NICHT ABER DER UNTERBRECHUNG DER STROMVERSORGUNG).

WICHTIG: Lesen Sie vor dem Anschließen der Stromversorgung die Installationsanweisungen!

Wechselstromsysteme: Verwenden Sie nur ein Stromkabel mit geerdetem Stecker und verbinden Sie dieses immer nur mit einer geerdeten Steckdose. Jedes Stromkabel muss an einen eigenen Stromkreis angeschlossen werden.

Gleichstromsysteme: Dieses Gerät basiert auf dem im Gebäude installierten Schutz vor Kurzschlüssen (Netzüberlastung). Stellen Sie sicher, dass für alle stromführenden Leiter eine zertifizierte Sicherung oder ein Leistungsschalter mit nicht mehr als 72V Gleichstrom, 15A verwendet wird. Für Geräte, die ständig angeschlossen sind, sollte in der Gebäudeverkabelung ein leicht zugänglicher Trennschalter installiert werden. Für eine permanente Verbindung verwenden Sie Kupferdraht der im Benutzerhandbuch des Systems angegebenen Stärke.

Das Gehäuse verfügt über einen eigenen Erdungs-Verbindungsbolzen. Stellen Sie die Erdungsverbindung her, ehe Sie das Stromkabel oder Peripheriegeräte anschließen, und trennen Sie die Erdungsverbindung niemals, so lange Strom- und Peripherieverbindungen angeschlossen sind.

Um die Gefahr eines durch ein Telefon oder Ethernet*-System bedingten elektrischen Schlags zu verringern, schließen Sie das Stromkabel des Geräts an, ehe Sie diese Verbindungen einrichten. Trennen Sie diese Verbindungen, ehe Sie die Hauptstromversorgung des Geräts unterbrechen.

SICHERHEITSHINWEISE BEI GESTELLMONTAGE: Dieses Gerät kann stationär in einem Gestell angebracht werden. Das Gestell muss den Anforderungen an eine physische Stärke laut NEBS GR-63-CORE und NEBS GR 487 entsprechen. Trennen Sie vor der Installation oder dem Abbau des Geräts in einem Gestell alle Strom- und externen Verbindungen.

Das Gewicht des Systems kann vor dem Einbau verringert werden, indem man alle während des Betriebs austauschbaren Elemente entfernt. Achten Sie darauf, das System so aufzustellen, dass das Gestell gleichmäßig belastet wird. Eine ungleiche Verteilung des Gewichts kann gefährlich werden. Befestigen Sie alle Sicherungsbolzen, wenn Sie das Gehäuse in einem Gestell montieren.

Warnung: Überprüfen Sie, ob Stromkabel und Steckdose kompatibel sind: Verwenden Sie die Ihrer Stromkonfiguration entsprechenden Stromkabel. Weitere Informationen finden Sie auf folgender Website: http://kropla.com/electric2.htm.

Warnung: Vermeiden Sie elektrische Überlastung, Hitze, elektrischen Schlag oder Feuergefahr: Schließen Sie das System nur an einen den Spezifikationen des Produkt-Benutzerhandbuchs entsprechenden Stromkreis an. Stellen Sie keine Verbindung zu Terminals her, die nicht den jeweiligen Spezifikationen entsprechen. Für die korrekten Verbindungen siehe das Benutzerhandbuch des Produkts.

Warnung: Vermeiden Sie einen elektrischen Schlag: Unterlassen Sie den Betrieb in nassen, feuchten oder kondensierenden Betriebsumgebungen. Um die Gefahr eines elektrischen Schlags oder eines Feuers zu vermeiden, betreiben Sie dieses Produkt nicht ohne Gehäuse oder Abdeckungen.

Warnung: Vermeiden Sie einen elektrischen Schlag: Trennen Sie bei Geräten mit mehreren Stromquellen vor der Wartung alle externen Stromverbindungen.

Warnung: Netzteile dürfen nur von qualifizierten Servicemitarbeitern ausgewechselt werden.

Vorsicht: Anforderungen an die Systemumgebung: Komponenten wie Prozessor-Boards, Ethernet-Schalter usw. sind auf den Betrieb mit externer Luftzufuhr ausgelegt. Diese Komponenten können bei Betrieb ohne externe Luftzufuhr beschädigt werden. Wenn die Komponenten in einem kompatiblen Gehäuse installiert sind, wird Luft von außen normalerweise durch Gehäuselüfter zugeführt. Blockieren Sie niemals die Luftzufuhr der Gerätelüfter oder -ventilatoren. In ungenutzten Gehäusesteckplätzen müssen Füllelemente oder Luftsteuerungseinheiten eingesetzt werden. Die Betriebsbedingungen können zwischen den verschiedenen Produkten variieren. Für die Anforderungen an die Belüftung und andere Betriebsbedingungen siehe die Benutzerhandbücher der jeweiligen Produkte.

Warnung: Die Kühlkörper des Geräts können sich während des normalen Betriebs erhitzen: Um Verbrennungen zu vermeiden, sollte jeder Kontakt mit den Kühlkörpern vermieden werden.

Warnung: Vermeiden Sie Verletzungen, Feuergefahr oder Explosionen: Unterlassen Sie den Betrieb dieses Produkts in einer explosionsgefährdeten Betriebsumgebung.

Vorsicht: Lithiumbatterien. Bei unsachgemäßem Austausch oder Umgang mit Batterien besteht Explosionsgefahr. Zerlegen Sie die Batterie nicht und laden Sie diese nicht wieder auf. Entsorgen Sie die Batterie nicht durch Verbrennen. Beim Auswechseln der Batterie muss dasselbe oder ein der Händlerempfehlung gleichwertiges Modell verwendet werden (CR2032). Gebrauchte Batterien müssen entsprechend den Anweisungen des Herstellers entsorgt werden.

Warnung: Vermeiden Sie Verletzungen: Dieses Produkt kann ein oder mehrere Lasergeräte enthalten, die abhängig von den installierten Plug-In-Modulen optisch zugänglich sind. Mit einem Lasergerät ausgestattete Produkte müssen der International Electrotechnical Commission (IEC) 60825 entsprechen.

33.3 Norme di Sicurezza

Leggere le norme seguenti per prevenire lesioni personali ed evitare di danneggiare questo prodotto o altri a cui è collegato. Per evitare qualsiasi pericolo potenziale, usare il prodotto unicamente come indicato.

Leggere tutte le informazioni sulla sicurezza fornite nella guida per l'utente relativa al componente e comprendere le norme associate ai simboli di pericolo, agli avvisi scritti e alle precauzioni da adottare prima di accedere a componenti o aree dell'unità. Custodire il presente documento per usi futuri.

AVVISO DI SICUREZZA RELATIVO ALL'ALIMENTAZIONE IN C.A. E/O C.C. Il cavo di alimentazione in c.a. e/o c.c. rappresenta il dispositivo principale per interrompere l'alimentazione in c.a. e/o c.c. dell'unità e deve sempre essere facilmente accessibile. Gli interruttori di accensione/spegnimento ausiliari per l'alimentazione in c.a. e/o c.c. hanno l'unico scopo di controllare l'alimentazione (NON INTERROMPONO L'ALIMENTAZIONE PRINCIPALE).

IMPORTANTE: prima di collegare l'unità alla fonte di alimentazione, leggere le istruzioni di installazione.

Per i sistemi CA, usare solo un cavo di alimentazione con una spina provvista di una messa a terra e collegarsi sempre a prese provviste di una messa a terra. Ogni cavo di alimentazione deve essere collegato ad un circuito derivato dedicato.

Per i sistemi CC, la presente unità può usufruire dell'eventuale installazione integrata nell'edificio per la protezione contro i cortocircuiti (sovratensione). Assicurarsi della presenza di un fusibile o di un circuito derivato non superiore a 72 V c.c., 15 A, certificato e conforme alla normativa in vigore, in tutti i conduttori portanti. Per gli apparecchi collegati in modo permanente, è necessario inserire nel circuito dell'edificio un interruttore ad accesso immediato. Per i collegamenti permanenti, usare il filo di rame del diametro specificato nella guida per l'utente relativa al sistema.

Il materiale fornito comprende un perno per il collegamento della messa a terra. Assicurare il collegamento della messa a terra prima di alimentare l'unità o prima di collegarla alle periferiche e non scollegare mai la messa a terra quando l'unità è alimentata o collegata a periferiche.

Per ridurre il rischio di scariche elettriche da parte della linea telefonica o dalla rete Ethernet*, collegare l'unità all'alimentazione principale prima di effettuare tale collegamento. Rimuovere i collegamenti prima di togliere l'alimentazione principale all'unità.

NORME DI SICUREZZA PER LE UNITÀ MONTATE IN UN RACK. Questa unità può essere alloggiata in modo permanente in un rack. Il montaggio in rack deve essere conforme ai requisiti di resistenza fisica delle norme NEBS GR-63-CORE e NEBS GR 487.Prima di installare o rimuovere l'unità da un rack, rimuovere tutte le fonti di alimentazione e i collegamenti esterni.

Prima di effettuare il montaggio, è possibile ridurre il peso complessivo del sistema togliendo tutte le apparecchiature sostituibili a caldo. Montare il sistema in modo da garantire una distribuzione uniforme del peso nel rack. Una distribuzione irregolare del peso può essere pericolosa. Avvitare fino in fondo tutti i bulloni durante l'installazione dell'unità in un rack.

Avvertenza: verificare il cavo di alimentazione e la compatibilità con la presa di corrente. Usare i cavi di alimentazione compatibili con il tipo di presa di corrente. Per ulteriori informazioni, visitare il sito Web all'indirizzo seguente: http://kropla.com/electric2.htm.

Avvertenza: evitare sovraccarichi elettrici, calore diretto, scosse e possibili cause di incendio. Collegare il sistema solo ad una rete elettrica la cui tensione nominale corrisponda al valore indicato nella guida per l'utente. Non collegarlo a fonti di alimentazione con valori di tensione esterne a quanto specificato per il sistema. Per ulteriori informazioni sul corretto collegamento, consultare la guida per l'utente del prodotto.

Avvertenza: evitare le scosse elettriche. Non usare l'apparecchio in ambienti umidi o in presenza di condensa. Per evitare scosse elettriche o possibili cause di incendio, non adoperare il prodotto senza le custodie o i pannelli appositi.

Avvertenza: evitare le scosse elettriche. Prima di intervenire su unità con più fonti di alimentazione, rimuovere tutti i collegamenti all'alimentazione esterna.

Avvertenza: far sostituire i componenti di alimentazione solo da personale tecnico qualificato.

Attenzione: rispettare i requisiti ambientali del sistema. I componenti come le schede di processore, i commutatori Ethernet, ecc., sono progettati per funzionare in presenza di un flusso di aria proveniente dall'esterno, in assenza del quale rischiano di danneggiarsi irrimediabilmente. In genere, il flusso di aria esterno viene generato da appositi ventilatori installati contemporaneamente ai componenti nello chassis compatibile. Non ostacolare mai il flusso di aria convogliato dal ventilatore e dai condotti dell'unità. I pannelli di copertura o le schede per il controllo dell'aria devono essere installati negli alloggiamenti vuoti dello chassis. I requisiti ambientali possono variare a seconda del prodotto. Per ulteriori informazioni sui requisiti del flusso di aria e sugli altri requisiti ambientali, consultare la guida per l'utente del prodotto.

Avvertenza: i dissipatori di calore possono scaldarsi durante il funzionamento normale. Per evitare bruciature o danni, evitare il contatto del dissipatore di calore con qualsiasi altro elemento.

Avvertenza: evitare lesioni, possibili cause di incendio o di esplosione. Non usare il prodotto in un'atmosfera in cui sussiste il rischio di esplosione.

Attenzione: le batterie al litio. La sostituzione o l'uso non corretto della batteria comporta un rischio di esplosione. Non smontare né ricaricare la batteria. Non gettare la batteria nel fuoco. Per la sostituzione, usare il tipo di batteria identico (CR2032) o equivalente consigliato dal costruttore. Le batterie usate devono essere smaltite rispettando le istruzioni del costruttore.

Avvertenza: evitare le lesioni. Questo prodotto può contenere uno o più dispositivi laser accessibili alla vista, a seconda dei moduli installati. I prodotti provvisti di un dispositivo laser devono essere conformi alla norma 60825 della Commissione elettrotecnica internazionale (IEC).

33.4 Instrucciones de Seguridad

Examine las instrucciones sobre condiciones de seguridad que siguen para evitar cualquier tipo de daños personales, así como para evitar perjudicar el producto o productos a los que esté conectado. Para evitar riesgos potenciales, utilice el producto únicamente en la forma especificada.

Lea toda la información relativa a seguridad que se incluye en los manuales de usuario de los distintos componentes y procure familiarizarse con los distintos símbolos de seguridad, advertencias escritas y normas de precaución antes de manipular las distintas piezas o secciones de la unidad. Guarde este documento para consultarlo en el futuro.

AVISO DE SEGURIDAD SOBRE LA ALIMENTACIÓN DE CA O CC El cable de alimentación de CA o CC constituye el dispositivo principal de desconexión de la alimentación de CA o CC, y debe permanecer accesible en todo momento. Los interruptores auxiliares de encendido y apagado de CA o CC y los disyuntores sólo tienen una función de control de la alimentacion (Y NO LA DE DESCONEXIÓN PRINCIPAL).

IMPORTANTE: Consulte las instrucciones de instalación antes de conectar la unidad a la alimentación.

En el caso de sistemas de CA, utilice sólo cables de alimentación con enchufe con toma de tierra, y realice siempre conexiones a una toma con toma de tierra. Cada uno de los cables de alimentación deberá estar conectado a una derivación dedicada.

En el caso de sistemas de CC, la unidad dependerá de la instalación existente en el edificio para la protección frente a cortocircuitos (sobreintensidades). Asegúrese de que todos los conductores que transporten corriente empleen un fusible o disyuntor homologado y certificado con una capacidad que no supere los 72V de CC ni 15A. En el caso de los equipos que vayan a permanecer conectados de manera constante, en la instalación eléctrica del edificio deberá estar incluida una desconexión de fácil acceso. Para conexiones permanentes, emplee cable de cobre del calibre especificado en el manual de usuario del sistema.

El chasis incluye aparte una clavija de conexión a tierra. Realice la conexión a tierra antes de suministrar corriente o realizar cualquier tipo de conexión de periféricos; no desconecte nunca la toma de tierra mientras la corriente esté presente o existan conexiones con periféricos.

Para reducir los riesgos de descargas eléctricas a través de un teléfono o un sistema de Ethernet*, conecte la alimentación principal de la unidad antes de realizar este tipo de conexiones. Desconecte estas conexiones antes de desconectar la alimentación principal de la unidad.

PROCEDIMIENTOS DE SEGURIDAD PARA EL CHASIS DE MONTAJE EN BASTIDOR: Esta unidad puede estar preparada para su montaje en un bastidor estático. Un montaje de este tipo deberá realizarse en un bastidor que cumpla con los requisitos de robustez de las normas NEBS GR-63-CORE y NEBS GR 487. Desconecte cualquier tipo de alimentación y conexiones externas antes de instalar la unidad en un bastidor o desmontarla.

Puede desmontar todos los equipos de intercambio en caliente para reducir el peso del sistema antes del montaje en bastidor. Asegúrese de montar el sistema de forma que el peso quede distribuido uniformemente en el bastidor. Una distribución irregular del peso podría generar riesgos. Asegúrese de fijar todos los tornillos de montaje en el bastidor.

Advertencia: Compatibilidad del cable y la toma: Utilice los cables adecuados para la configuración de tomas de corriente con que cuente. Si necesita más información, visite el sitio web siguiente: http://kropla.com/electric2.htm.

Advertencia: Evite sobrecargas eléctricas, calor y riesgos de descarga eléctrica o incendio: Conecte el sistema sólo a un circuito de alimentación que tenga el régimen apropiado, según lo especificado en el manual de usuario del producto. No realice conexiones con terminales cuya capacidad no se ajuste al régimen especificado para ellos. Consulte el manual de usuario del producto para que las conexiones que realice sean las correctas.

Advertencia: Evite descargas eléctricas: No haga funcionar el sistema en condiciones de humedad, mojado o si se produce condensación de la humedad. Para evitar descargas eléctricas o posibles incendios, no permita que el aparato funcione con sus tapas o paneles del chasis desmontados.

Advertencia: Evite descargas eléctricas: En el caso de unidades que cuenten con varias fuentes de alimentación, desconecte las conexiones con alimentación externa antes de proceder a realizar labores de mantenimiento.

Advertencia: La sustitución de fuentes de alimentación sólo debe ser realizada por personal de mantemiento cualificado.

Precaución: Requisitos de entorno para el sistema: Los componentes del tipo de placas de procesador, conmutadores de Ethernet, etc., están concebidos para funcionar en condiciones que permitan el paso de aire. Los componentes pueden averiarse si funcionan sin que circule el aire en su entorno. La circulación del aire suele estar facilitada por los ventiladores incorporados en el armazón cuando los componentes están instalados en armazones compatibles. Nunca interrumpa el paso del aire por los ventiladores or los respiraderos. Los paneles de relleno y las placas para el control de la circulación del aire deben instalarse en ranuras del chasis que no estén destinadas a ningún otro uso. Las características técnicas relativas al entorno pueden variar entre productos. Consulte los manuales de usuario del producto si necesita conocer sus necesidades en términos de circulación de aire u otras características técnicas.

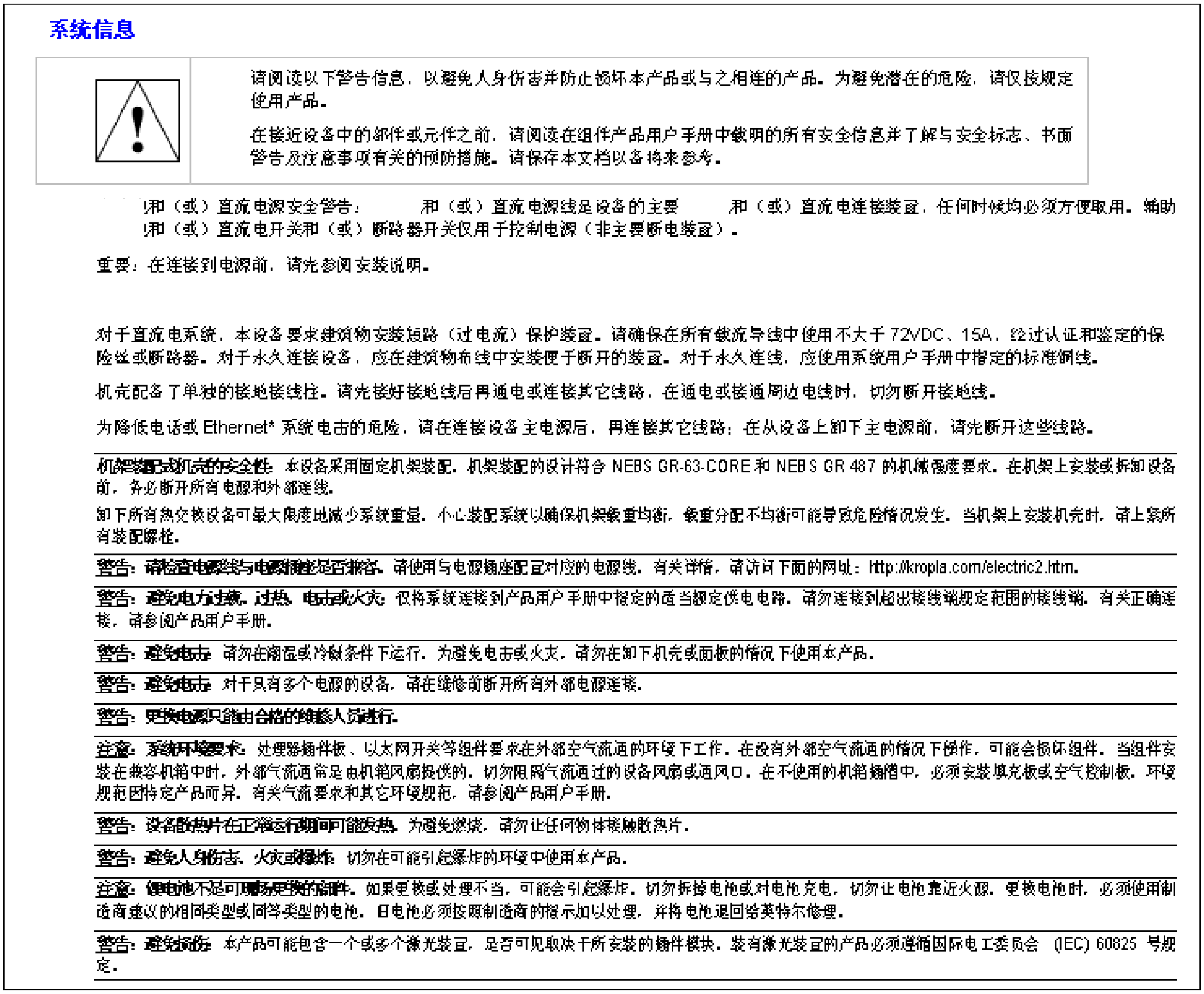
Advertencia: En condiciones de funcionamiento normales, los disipadores de calor pueden recalentarse. Evite que ningún elemento entre en contacto con los disipadores para evitar quemaduras.

Advertencia: Riesgos de daños, incendio o explosión: No permita que el aparato funcione en una atmósfera que presente riesgos de explosión.

Precaución: Las baterías de litio. Si las baterías no se manipulan o cambian correctamente, exite riesgo de explosión. No desmonte ni recargue la batería. Nunca tire las baterías al fuego. Al cambiar la batería, es preciso utilizar el mismo tipo (CR2032) o un tipo equivalente que haya sido recomendado por el fabricante. Las baterías utilizadas deben desecharse según las instrucciones del fabricante.

Advertencia: Daños personales: Este producto puede contener uno o varios dispositivos láser, que estarán a la vista dependiendo de los módulos enchufables que se hayan instalado. Los productos provistos de un dispositivo láser deben ajustarse a la norma 60825 de la International Electrotechnical Commission (IEC).

33.5 Chinese Safety Warning



系统信息

请阅读以下警告信息，以避免人身伤害并防止损坏本产品或与之相连的产品。为避免潜在的危险，请仅按规定使用产品。

在接近设备中的部件或元件之前，请阅读在组件产品用户手册中载明的所有安全信息并了解与安全标志、书面警告及注意事项有关的预防措施。请保存本文档以备将来参考。

和（或）直流电源安全警告： 和（或）直流电源线是设备的主要 和（或）直流电连接装置，任何时候均必须方便取用。辅助 和（或）直流电开关和（或）断路器开关仅用于控制电源（非主要断电装置）。

重要：在连接到电源前，请先参阅安装说明。

对于直流电系统，本设备要求建筑物安装短路（过电流）保护装置。请确保在所有载流导线中使用不大于 72VDC、15A，经过认证和鉴定的保险丝或断路器。对于永久连接设备，应在建筑物布线中安装便于断开的装置。对于永久连线，应使用系统用户手册中指定的标准铜线。

机壳配备了单独的接地接线柱。请先接好接地线后再通电或连接其它线路，在通电或接通周边电线时，切勿断开接地线。

为降低电话或 Ethernet* 系统电击的危险，请在连接设备主电源后，再连接其它线路；在从设备上卸下主电源前，请先断开这些线路。

机架装配或机壳的安全性： 本设备采用固定机架装配。机架装配的设计符合 NEBS GR-63-CORE 和 NEBS GR 487 的机械强度要求。在机架上安装或拆卸设备前，务必断开所有电源和外部连线。

卸下所有热交换设备可最大限度地减少系统重量。小心装配系统以确保机架载重均衡，载重分配不均衡可能导致危险情况发生。当机架上安装机壳时，请上紧所有装配螺栓。

警告：请检查电源线与电源插座是否兼容。 请使用与电源插座配置对应的电源线。有关详情，请访问下面的网址：http://kropla.com/electric2.htm。

警告：避免电力过载、过热、电击或火灾： 仅将系统连接到产品用户手册中指定的适当额定供电电路。请勿连接到超出接线端规定范围的接线端。有关正确连接，请参阅产品用户手册。

警告：避免电击： 请勿在潮湿或冷凝条件下运行。为避免电击或火灾，请勿在卸下机壳或面板的情况下使用本产品。

警告：避免电击： 对于具有多个电源的设备，请在维修前断开所有外部电源连接。

警告：更换电源只能由合格的维修人员进行。

注意：系统环境要求： 处理器插件板、以太网开关等组件要求在外部空气流通的环境下工作。在没有外部空气流通的情况下操作，可能会损坏组件。当组件安装在兼容机箱中时，外部气流通常是由机箱风扇提供的。切勿阻隔气流通过的设备风扇或通风口。在不使用的机箱插槽中，必须安装填充板或空气控制板。环境规范因特定产品而异。有关气流要求和其它环境规范，请参阅产品用户手册。

警告：设备散热片在正常运行期间可能发热。 为避免烧伤，请勿让任何物体接触散热片。

警告：避免人身伤害、火灾或爆炸： 切勿在可能引起爆炸的环境中使用本产品。

注意：锂电池不是可现场更换的部件。 如果更换或处理不当，可能会引起爆炸。切勿拆掉电池或对电池充电，切勿让电池靠近火源。更换电池时，必须使用制造商建议的相同类型或同等类型的电池。旧电池必须按照制造商的指示加以处理，并将电池退回给英特尔修理。

警告：避免损伤： 本产品可能包含一个或多个激光装置，是否可见取决于所安装的插件模块。装有激光装置的产品必须遵循国际电工委员会 (IEC) 60825 号规定。

Example CLI Commands A

The following table shows examples of most CLI operations.

Note: The variable "*N*" (as in blade*N* or fantray*N*) represents the chassis slot number of the device being acted on (such as Blade5 or fantray1). Please refer to chassis documentation for slot, fan, fan tray, and power supply location and information.

Table 111. Example CLI Commands (Sheet 1 of 3)

Use Case	CLI Command	Return
Get fan tray presence.	cmmget –l fantray1 –d presence	Fantray1 is present. Fantray1 is not present.
Get the baseboard temperature of blade 5 when the sensor is named "ambient temp".	cmmrget –l blade5 –t *"ambient temp"*–d current	Current Temperature of "ambient temp" on blade 5.
Find out if blade 5 is present.	cmmget –l blade5 –d presence	Blade5 is present. Blade5 is not present.
Get all thresholds for the +3.3 V sensor on blade 2.	cmmget –l blade2 –t "+3.3V" –d thresholdsall	Returns thresholds for upper non-recoverable, upper critical, upper non-critical, lower non-critical, lower critical, and lower non-recoverable.
Get the overall system health.	cmmget –l system –d health	Indicates if the entire system is healthy, or has health events.
Get a list of blades with problems	cmmget –l system –d unhealthylocations	List of locations with active health events.
Get the CMM's overall health.	cmmget –l cmm –d health	Indicates if the CMM is healthy or if there are active health events on the CMM.
Display the SEL of the CMM	cmmget –l cmm –d sel	Displays a list of SEL entries (if any) for the CMM.
Get the version of firmware on the CMM	cmmget –l cmm –d version	Displays the version of firmware currently installed on the CMM.
Find out what sensors are on blade 3.	cmmget –l blade3 –d listtargets	A list of sensor names as defined in their SDRs on blade 3.
List the data items that can be queried or set on blade 4.	cmmget –l blade4 –d listdataitems	A list of dataitems for blade 4.
Find out what can be queried on blade4's +3. 3V sensor.	cmmget –l blade4 –t "+3.3V" –d listdataitems	A list of commands to be used as dataitems for the +3.3 V.
Get the FRU's hot swap state.	cmmget –l blade5 -d hotswapstate	The intern M state information: Blade5 Hot Swap state is M[4].

Table 111. Example CLI Commands (Sheet 2 of 3)

Use Case	CLI Command	Return
Query what LEDs are supported by a FRU	cmmget -l blade5 -d ledproperties	LED support information: Blade5 has control of hsled, led1, led2, Blade5 support 0 user leds
Query the color support of an LED1 on a FRU	cmmget -l blade5 -t led1 -d ledcolorprops	Information on the colors supported by an LED: led1 supports red,green,amber Default local control color is green Default override color is red
Query the current state of LED1 on a FRU	cmmget -l blade5 -t led1 -d ledstate	Current LED state information: led1 is in localcontrol mode function is on color is green
Set LED1 on a FRU to blink green, 900ms off, 100ms on	cmmset -l blade5 -t led1 -d ledstate -v blink,900,100,green	Success or failure
Get a FRU's supported power levels	cmmget -l blade5 -d powerlevels	The FRU's steady state power levels: AdvancedTCA FRU Power Levels: Power Level 1 = 152 watts
Query the CMM for it's power budget	cmmget -l cmm -d powerbudget	The shelf's power budget: Shelf Power Budget Distribution: Feed #1 = 1322 Watts Feed #2 = 1322 Watts Feed #3 = 1474 Watts Feed #4 = 1474 Watts
Query the shelf's address/location	cmmget -l chassis -d location	The shelf address information: Shelf Address: lab bench
Set the shelf's address/location	cmmset -l chassis -d location -v "[location]"	Success or Failure
Query the FRU's activation policy	cmmget -l bladeN -d fruactivationpolicy	Information about whether the "Locked Bit" is set or not: "Blade11 activation locked bit is set. If in M1 Blade 11 cannot transition to M2 until unlocked" or "Blade11 activation locked bit is not set. Blade11 can transition from M1 to M2"
Set the FRU activation policy	cmmset -l bladeN -d fruactivationpolicy -v [0-1]	Success or Failure
Activate or Deactivate the FRU	cmmset -l bladeN -d fruactivation -v [0-1]	Success or Failure

Table 111. Example CLI Commands (Sheet 3 of 3)

Use Case	CLI Command	Return
Exercise control over the FRU payload	cmmset -l bladeN -d frucontrol -v [0-3] Where: 0 = Cold reset 1 = Warm reset 2 = Graceful reboot 3 = Issue diag. Interrupt	Success or Failure
Retrieve the device's hardware revision, FW/SW revision, and sensor and event interface command spec revision information	cmmget -l cmm -d deviceid	Refer to Section 8, "The Command Line Interface (CLI)" on page 71
Query maximum FRU Device ID supported by the IPM controller	cmmget -l cmm -d picmgproperties	Refer to Section 8, "The Command Line Interface (CLI)" on page 71
Customize the OEM login message	cmmset -l cmm -d loginmessage -v "My CMM"	Success or Failure
Customize the OEM command line prompt	cmmset -l cmm -d cmdlineprompt -v "My CMM"	Success or Failure

Data Sheet Reference B

This appendix provides links to data sheets, standards, and specifications for the technology designed into the CMM.

B.1 Intel® AdvancedTCA* Product Information

Information, collateral, and software updates can be found for all Intel AdvancedTCA* products at:

http://www.intel.com/technology/atca/index.htm

B.2 AdvancedTCA*

Current AdvancedTCA Specifications can be purchased from PICMG* for a nominal fee. Short form specifications in Adobe Acrobat format (PDF) are also available on PICMG's website at:

http://www.picmg.org/pdf/PICMG_3_0_Shortform.pdf

B.3 IPMI

The Intelligent Platform Management Initiative is described and current specifications can be found at:

http://developer.intel.com/design/servers/ipmi/spec.htm

B.4 Intel® IOP312 Processor Chipset

For more information about the Intel® IOP310 Processor Chipset, see the following Web site:

http://developer.intel.com/design/iio/docs/iop310.htm

Customer Support C

This appendix offers technical and sales assistance information for this product as well as information on returning an Intel® NetStructure™ product for service.

C.1 Technical Support and Return for Service Assistance

For all product returns and support issues, please contact your Intel product distributor or Intel Sales Representative for specific information.

C.2 Sales Assistance

If you have a sales question, please contact your local Intel NetStructure Sales Representative or the Regional Sales Office for your area. Address, telephone and fax numbers, and additional information is available at Intel's website, located at:

http://www.intel.com/network/csp/sales/

Intel Corporation
Telephone (in U.S.) 1-800-755-4444
Telephone (Outside U.S.) 1-973-993-3030

C.3 Open Source CD

For acquiring the Open Source CD associated with our current firmware release, please contact your Intel Sales Representative for specific information.