Basic Blade Services Software on PrAMC-7311

Programmer's Reference

P/N: 6806800P89G

December 2019

Trademarks

The stylized "S" and "SMART" is a registered trademark of SMART Modular Technologies, Inc. and “SMART Embedded Computing” and the SMART Embedded Computing logo are trademarks of SMART Modular Technologies, Inc. All other names and logos referred to are trade names, trademarks, or registered trademarks of their respective owners. These materials are provided by SMART Embedded Computing as a service to its customers and may be used for informational purposes only.

Disclaimer*

SMART Embedded Computing (SMART EC) assumes no responsibility for errors or omissions in these materials. **These materials are provided "AS IS" without warranty of any kind, either expressed or implied, including but not limited to, the implied warranties of merchantability, fitness for a particular purpose, or non-infringement.** SMART EC further does not warrant the accuracy or completeness of the information, text, graphics, links or other items contained within these materials. SMART EC shall not be liable for any special, indirect, incidental, or consequential damages, including without limitation, lost revenues or lost profits, which may result from the use of these materials. SMART EC may make changes to these materials, or to the products described therein, at any time without notice. SMART EC makes no commitment to update the information contained within these materials.

Electronic versions of this material may be read online, downloaded for personal use, or referenced in another document as a URL to a SMART EC website. The text itself may not be published commercially in print or electronic form, edited, translated, or otherwise altered without the permission of SMART EC.

It is possible that this publication may contain reference to or information about SMART EC products, programming, or services that are not available in your country. Such references or information must not be construed to mean that SMART EC intends to announce such SMART EC products, programming, or services in your country.

Limited and Restricted Rights Legend

If the documentation contained herein is supplied, directly or indirectly, to the U.S. Government, the following notice shall apply unless otherwise agreed to in writing by SMART Embedded Computing.

Use, duplication, or disclosure by the Government is subject to restrictions as set forth in subparagraph (b)(3) of the Rights in Technical Data clause at DFARS 252.227-7013 (Nov. 1995) and of the Rights in Noncommercial Computer Software and Documentation clause at DFARS 252.227-7014 (Jun. 1995).

SMART Embedded Computing, Inc.

2900 S. Diablo Way, Suite 190

Tempe, Arizona 85282

USA

*For full legal terms and conditions, visit *www.smartembedded.com/ec/legal*

Table of Contents

About this Manual 11

1 Introduction 17

1.1 Overview 17
1.2 Software Building Blocks 17

2 Installing the BBS Software 19

2.1 Overview 19
2.1.1 Package Information 19
2.1.2 Available BBS Packages 19
2.1.3 Accessing the PrAMC-7311 via Serial Console 20
2.1.4 Installation Overview 20
2.2 Configuring TFTP, DHCP and PXE 21
2.2.1 Setting Up and Configuring the TFTP Server 21
2.2.2 Configuring DHCP 22
2.2.3 Configuring PXE 24
2.3 Installation Procedures 25
2.3.1 Booting the PrAMC-7311 Using a Ramdisk 25
2.3.1.1 Creating a Ramdisk USB boot device 26
2.3.1.2 Creating a Bootable Disk Device 27
2.3.2 Diskless Client Network Boot Installation 34
2.4 Upgrading the Software 35
2.5 Adapting the BBS Software to Customer's Needs 35
2.5.1 Modifying the Root File System Images 35
2.5.2 Modifying the Configuration of the Linux Kernel 35
2.6 Serial-Over-LAN 36
2.6.1 SOL Channels 36
2.6.2 Configuring LAN Interfaces 36
2.6.2.1 Set LAN Configuration Parameters Command 36
2.6.3 Configuring SOL Parameters 37
2.6.3.1 Set SOL Configuration Parameters Command 37
2.6.4 Configuring User Accounts 37
2.6.4.1 IPMI Messaging Support Commands 38
2.6.5 SOL Session Setup with IPMITOOL 38
2.6.5.1 Setting up an SOL Session 38
2.6.6 Exiting the SOL Session 39

2.6.7 Console Port Baud Rate . . . 39
2.6.8 Console Port Flow Control . . . 39

3 Firmware Upgrade Facility . . . 41

3.1 Overview . . . 41
3.2 Firmware Recovery Image Files . . . 41
3.3 Backup Concept . . . 41
3.4 fcu—Firmware Upgrade Command-Line Utility . . . 42
3.5 Upgrading a Firmware Image . . . 45
3.5.1 BIOS Upgrade . . . 45
3.5.2 IPMC Upgrade . . . 47

4 Linux Distribution Description . . . 49

4.1 Distribution Description . . . 49
4.2 Login . . . 49
4.3 Network Services Configuration . . . 49

5 Hardware Platform Management . . . 51

5.1 Overview . . . 51
5.2 hpmagentd—HPM Agent Daemon . . . 53
5.3 hpm—Start-Up Script . . . 54
5.4 hpm—Shutdown and Reboot Scripts . . . 55
5.5 hpmcmd—HPM Command Utility . . . 55
5.5.1 Command Overview . . . 57
5.5.2 Supported Commands . . . 58
5.5.2.1 bye . . . 59
5.5.2.2 bootbankget . . . 59
5.5.2.3 bootbankset . . . 60
5.5.2.4 cmd . . . 60
5.5.2.5 deviceid . . . 61
5.5.2.6 exit . . . 61
5.5.2.7 frudata . . . 62
5.5.2.8 fruinfoget . . . 62
5.5.2.9 fruinv . . . 63
5.5.2.10 fruread . . . 64
5.5.2.11 get-shut-tout . . . 64

5.5.2.12 help . . . 65
5.5.2.13 ipmbaddress . . . 65
5.5.2.14 ipmcdevice . . . 65
5.5.2.15 ledget . . . 65
5.5.2.16 ledprop . . . 66
5.5.2.17 ledset . . . 66
5.5.2.18 loglevelget . . . 67
5.5.2.19 loglevelset . . . 68
5.5.2.20 macaddress . . . 68
5.5.2.21 partnumber . . . 69
5.5.2.22 physlotnumber . . . 69
5.5.2.23 portget . . . 69
5.5.2.24 portset . . . 70
5.5.2.25 posttypeget . . . 71
5.5.2.26 posttypeset . . . 71
5.5.2.27 quit . . . 72
5.5.2.28 rebootpath . . . 72
5.5.2.29 sdr . . . 72
5.5.2.30 sdr_dump . . . 73
5.5.2.31 sdrinfo . . . 73
5.5.2.32 sendamc . . . 74
5.5.2.33 sendcmd . . . 74
5.5.2.34 set-shut-tout . . . 75
5.5.2.35 shelfaddress . . . 75
5.5.2.36 shutdownpath . . . 76
5.5.2.37 slotnumber . . . 76
5.5.2.38 version . . . 76
5.5.2.39 watchdog . . . 77

A Related Documentation . . . 79

A.1 SMART Embedded Computing Documentation . . . 79
A.2 Related Specifications . . . 79
A.3 Additional Resources . . . 80

Table of Contents

List of Figures

Figure 1-1 BBS Architecture 17
Figure 5-1 Software Levels of the HPM Architecture 52

List of Figures

List of Tables

Table 2-1 BBS Ramdisk and Kernel Image Files 19
Table 2-2 BBS Module Files 20
Table 2-3 BBS Installation/Boot Options - Main Set Up and Configuration Steps 21
Table 2-4 SOL Channels 36
Table 2-5 User Accounts 37
Table 5-1 Command Overview 57
Table A-1 SMART EC Embedded Computing Publications 79
Table A-2 Related Specifications 79
Table A-3 Additional Resources 80

List of Tables

About this Manual

Overview of Contents

This manual is divided into the following chapters and appendices.

Chapter 1, Introduction on page 17

Chapter 2, Installing the BBS Software on page 19

Chapter 3, Firmware Upgrade Facility on page 41

Chapter 4, Linux Distribution Description on page 49

Chapter 5, Hardware Platform Management on page 51

Appendix A, Related Documentation on page 79

Abbreviations

This document uses the following abbreviations:

Abbreviation	Definition
AMC	Advanced Mezzanine Card
API	Application Programming Interface
AdvancedTCA	Advanced Telecommunications Computing Architecture
ATCA	Advanced Telecommunications Computing Architecture
BBS	Basic Blade Services
BIOS	Basic Input Output System
CGL	Carrier Grade Linux
CMC	Common Mezzanine Card
DAT	Domain Alarm Table
DHCP	Dynamic Host Configuration Protocol
DoS	Denial of Service
ECC	Embedded Communications Computing
ER	Expected Responder
EST	Eastern Standard Time
EVQ	Event Queue

Abbreviation	Definition
FCU	FUF Command Line Utility
FM	Fault Management
FPGA	Field Programmable Gate Array
FRI	Firmware Recovery Image
FRU	Field Replaceable Unit
FUF	Firmware Upgrade Facility
FWH	Firmware Hub
GPIO	General Purpose Input/Output
HPI	Hardware Platform Interface
HPM	Hardware Platform Management
IDE	Integrated Device Electronics
IPM	Intelligent Platform Management
IPMB	Intelligent Platform Management Bus
IPMC	Intelligent Platform Management Controller
IPMI	Intelligent Platform Management Interface
LSP	Linux Support Package
LT	Lower Threshold
LUN	Logic Unit Number
MAC	Media Access Control
MIB	Management Information Base
MMC	Module Management Controller
NTP	Network Time Protocol
OEM	Original Equipment Manufacturer
OSDL	Open Source Development Labs
PC	Personal Computer
PCI	Peripheral Component Interconnect
PCIEX	PCI Express

Abbreviation	Definition
PEM	Power Entry Module
PICMG	PCI Industrial Computers Manufacturers Group
PMC	PCI Mezzanine Card
PrPMC	Processor PMC
PXE	Preboot Execution Environment
RDR	Resource Data Record
RMCP	Remote Monitoring and Control Protocol
RPM	RedHat Package Manager
RTM	Rear Transition Module
SAF	Service Availability Forum
SAS	Serial Attached SCSI
SATA	Serial ATA
SCSI	Small Computer System Interface
SCXB	System Controller Switching Blade
SDR	Sensor Data Record
SIT	Simple Internet Transition
SMI	Serial Management Interface
SNMP	Simple Network Management Protocol
SOL	Serial Over Lan
SSH	Secure Shell
SSU	Synchronization Supply Unit
TAR	Tape Archiver
TBD	To Be Defined
TCP	Transmission Control Protocol
TFTP	Trivial File Transfer Protocol
UDP	User Datagram Protocol
UT	Upper Threshold

Conventions

The following table describes the conventions used throughout this manual.

Notation	Description
0x00000000	Typical notation for hexadecimal numbers (digits are 0 through F), for example used for addresses and offsets
0b0000	Same for binary numbers (digits are 0 and 1)
bold	Used to emphasize a word
Screen	Used for on-screen output and code related elements or commands. `Sample of Programming used in a table (9pt)`
`Courier + Bold`	Used to characterize user input and to separate it from system output
Reference	Used for references and for table and figure descriptions
File > Exit	Notation for selecting a submenu
<text>	Notation for variables and keys
[text]	Notation for software buttons to click on the screen and parameter description
...	Repeated item for example node 1, node 2, ..., node 12
. . .	Omission of information from example/command that is not necessary at the time
..	Ranges, for example: 0..4 means one of the integers 0,1,2,3, and 4 (used in registers)
|	Logical OR
	Indicates a hazardous situation which, if not avoided, could result in death or serious injury
	Indicates a hazardous situation which, if not avoided, may result in minor or moderate injury

Notation	Description
	Indicates a property damage message
	Indicates a hot surface that could result in moderate or serious injury
	Indicates an electrical situation that could result in moderate injury or death
	Indicates that when working in an ESD environment care should be taken to use proper ESD practices
	No danger encountered, pay attention to important information

Summary of Changes

Part Number	Date	Description
6806800P89G	December 2019	Rebranded to SMART Embedded Computing
6806800P89F	April, 2015	Updated Chapter 2, *Installing the Basic Blade Services Software*.
6806800P89E	June, 2014	Rebranded to Artesyn.
6806800P89D	October, 2013	Updated *Serial-Over-LAN* on page 38.
6806800P89C	June, 2013	Updated Chapter 2, *Serial-Over-LAN,* on page 38.
6806800P89B	December, 2012	Added *Booting the PrAMC-7311 Using a Ramdisk* on page 26 and *Serial-Over-LAN* on page 38.
6806800P89A	September, 2012	Initial version.

Introduction

1.1 Overview

The Basic Blades Services (BBS) software provides a set of services that support the module on which the software is installed.

BBS includes:

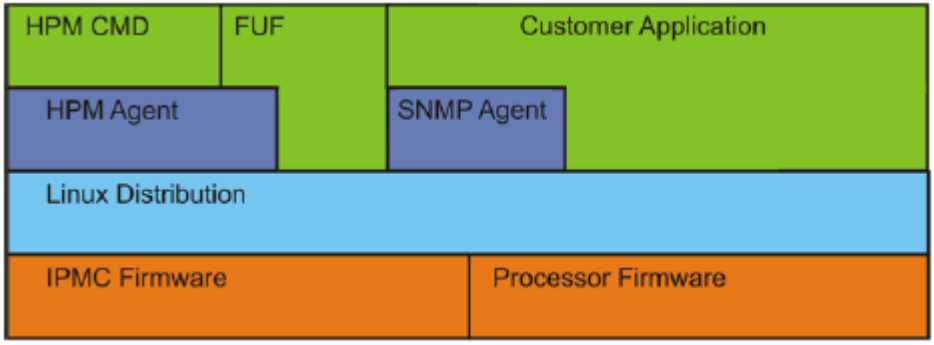
- Linux Operating System
- Several custom hardware management functions for the unique hardware of the module
- A set of management routines for Linux and all hardware interfaces. Management access includes support for SNMP and a local console interface based on a standard Linux command shell

1.2 Software Building Blocks

BBS services include a common set of functionality which is available for all AdvancedTCA blades and AMC modules, and a unique set of functionality which is tailored to a particular blade or module.

The figure below depicts the architecture of the BBS software.

Figure 1-1 BBS Architecture



BBS consists of the following software and services:

Firmware Upgrade Facility

The Firmware Upgrade Facility (FUF) provides a uniform way to upgrade firmware on SMART EC blades and AMC modules, regardless on which flash locations the firmware is stored. FUF upgrades the BIOS firmware as well as the IPMC firmware. The FUF currently consists of a Firmware Upgrade Command Line Utility (FCU), flash device drivers, and supports specially prepared firmware recovery image (FRI) files as well as HPM.1 compatible firmware images. The FUF can be used on switch and node blades and on AMC modules.

Linux

The PrAMC-7311 BBS described in this manual is based either on Linux or WindRiver 3.0.3 RCPL21. Various Linux services (above the kernel) will be activated by the BBS installation scripts.

Hardware Platform Management

HPM in AdvancedTCA systems is based on Intelligent Platform Management Interface specification (IPMI). IPMI commands can be complex and cumbersome. Using a certain set of commands, HPM facilitates the blade or module-level hardware management.

SNMP Agent

As each BBS module is individually managed, the default installation script installs and initializes the Net-SNMP agent. Note that the Net-SNMP agent is available, but not initialized by the default installation.

Installing the BBS Software

2.1 Overview

SMART EC provides software images, including software updates, to its licensed customers. To obtain the software, please contact your local SMART EC sales representative.

2.1.1 Package Information

BBS software is packaged with the RPM and is installed as part of the standard installation. In general, you will not need to install or upgrade an individual package.

The following rpm commands are useful to review package information.

Command	Description
`rpm -qa`	List all installed packages. Use **`rpm -qa | grep hpi`** to list only HPI packages.
`rpm -ql package-name`	List the content of a package, where package-name is the name of a specific package, for example, `rpm -ql openhpi`
`rpm -qi package-name`	List information about a package, where package-name is the name of a specific package, for example, `rpm -qi openhpi`
`rpm -qf <path to file>`	Finds out which RPM a file belongs to.

For more information about the rpm command, see its `man` page.

2.1.2 Available BBS Packages

The BBS distribution contains a Wind River 3.0.3 RCPL21 kernel and a ramdisk image containing the root file system and BBS RPMs. After booting, this image contains scripts that can be used to configure and populate an attached hard drive or USB flash device. There is no prompting for system configuration information except for date and time. The following table lists the files and their names.

Table 2-1 BBS Ramdisk and Kernel Image Files

File	Description
kernel	Wind River 3.0.3 RCPL21

Table 2-1 BBS Ramdisk and Kernel Image Files (continued)

File	Description
initramfs.cpio.gz	Ramdisk image contains root file system, BBS RPMs (preinstalled BBS RPMs) and so on. It is used for booting the PXE. For more information, see *Installation Overview on page 20*.

The files listed in the next table are contained in the above image files. You may have to access these files if you want to perform modifications or create a file system from scratch.

All files have release numbers in their names. Refer to the release notes applicable to your system release for details about current version numbers.

Table 2-2 BBS Module Files

File	Description
bbs-pramc7311-ipmc-boot-<VERSION>.rpm bbs-pramc7311-ipmc-<VERSION>.rpm	MMC Firmware images
bbs-pramc7311-bios-<VERSION>.rpm	BIOS

2.1.3 Accessing the PrAMC-7311 via Serial Console

In most procedures described in the following sections you need to invoke Linux commands or configure BIOS settings. In order to do this, you need to access the PrAMC-7311 via the face plate serial port. If using a serial console or terminal emulator, the serial port settings are: 115200 baud, no parity, eight data bits, and one stop bit.

If you wish to access Linux via a Linux shell, the account login is **`root`** with the password **`root`**.

2.1.4 Installation Overview

For installing/booting the BBS software on the PrAMC-7311, use the Diskless client boot via network method.

For this method, you need to set up an external TFTP server to retrieve the required BBS files and do some initial configurations. The following table provides an overview of the main steps that you need to perform. The detailed procedures can be found in the following sections.

Table 2-3 BBS Installation/Boot Options - Main Set Up and Configuration Steps

Installation/Boot Option	Main SetUp and Configuration Steps
Diskless client boot	Set up and configure external TFTP boot server Configure DHCP server Configure PXE boot options Configure PrAMC-7311 BIOS to boot from network

2.2 Configuring TFTP, DHCP and PXE

The following installation procedures assume that you have placed the new BBS files on a TFTP server. As one of the first steps, you will set up and configure a TFTP server. For the diskless client and hard disk installation/boot option, you need to configure the system's DHCP server and configure PXE boot options.

2.2.1 Setting Up and Configuring the TFTP Server

It is customary to place TFTP files in a /tftpboot directory.

Creating the /tftpboot Directory and Copying the Target Files

To create the expected directory structure and copy the needed files, follow these steps.

1. On the host, create a **/tftpboot** directory if it does not already exist.
   ```
   mkdir /tftpboot
   ```
2. Copy the files for your module to the **/tftpboot** directory. Depending on the installation/boot option, the required files are the following.

Installation/Boot	File	Description
Diskless client	kernel	Linux kernel
	initramfs.cpio.gz	Ram disk image

Next you will configure the TFTP server. The procedure assumes that the TFTP server is under the control of xinetd. For any other servers, this procedure will be different. It is essential that your server provides TFTP support and that the files are copied to the directories as specified in *Setting Up and Configuring the TFTP Server on page 21*.

Configuring a TFTP Server

To configure TFTP as root on the host, complete the following steps:

1. Create (or edit) the file **/etc/xinetd.d/tftp**. This file might be located somewhere else in your system, therefore, the first line might be different. Use the following format:

```
#/etc/xinetd.d/tftp
service tftp
{
   socket_type = dgram
   wait = yes
   user = root
   log_on_success += USERID
   log_on_failure += USERID
   server = /bin/in.tftpd
   server_args = -r blksize /tftpboot
   disable = no
   protocol = udp
}
```

2. Create the directory **/tftpboot** and add the required files as described in *Setting Up and Configuring the TFTP Server on page 21*.
3. Stop any TFTP daemons that have not timed out. Enter the following command to do so:

```
killall in.tftpd
```

4. Enter the following command to have **xinetd** re-read its configuration file:

```
/etc/rc.d/init.d/xinetd restart
```

Your TFTP server is now configured.

2.2.2 Configuring DHCP

To configure DHCP, you edit the configuration file of the DHCP server. At `/etc/` location, in the `dhcpd.conf` file, provide the MAC and IP address of the PrAMC-7311.

One way to determine the MAC address of the interface on your PrAMC-7311 is by letting BIOS attempt to PXE boot.

The screen displays information, for example:

```
Intel (R) Boot Agent GE v1.2.40
Copyright (C) 1997-2006, Intel Corporation

CLIENT MAC ADDR: 00 01 AF 18 F2 68
```

In this example, the MAC address would be 00:01:AF:18:F2:68.

The configuration file should look similar to the following extract.

```
#
# DHCP server sample configuration file
#

allow booting;
allow bootp;

#
#   A declaration for each subnet in your network that requires
DHCP service
#  Note that your DHCP server needs an interface addressed to each
of these #   subnets
#   These declarations can be used to define options common to all
blades in
#   that subnet
#

subnet 192.168.1.0 netmask 255.255.255.0
{
    option broadcast-address 192.168.1.255;
    option routers 192.168.1.253;
}

<Additional subnet declarations as required>

#
#  Blade declarations. One for each blade requiring DHCP service
#  Blades are identified by their ethernet address
#  7201 (and other boards using BIOS) should use filename
"pxelinux.0"
#      and will have further configuration information contained in
#  /tftpboot/pxelinux.cfg
#  6201 (and other boards using MotLoad) do not need a filename
```

```
#       as it can be specified in the individual board's firmware
boot command
#

host 7311_blade1
{
    hardware ethernet 00:01:AF:18:F2:68;
    fixed-address 192.168.1.1;
    filename "pxelinux.0";
}


<Additional blade declarations as required>
```

On Linux servers, the DHCP service is usually started with #**/etc/init.d/dhcp start**

Make sure the DHCP daemon is running:

```
# ps -aef|grep dhcp
```

The following is displayed if the daemon is running:

```
.../usr/bin/dhcpd...
```

Restart the daemon to apply the changes made to the DHCP configuration file:

```
# /etc/init.d/dhcp restart
```

2.2.3 Configuring PXE

Once BIOS has obtained an IP address using DHCP, it downloads and executes the file `pxelinux.0` configured in the DHCP entry. This utility will return to the server to get further information required to boot the Linux kernel. It looks for this information in the directory `/tftpboot/pxelinux.cfg`.

First, it checks for a file name using the Ethernet hardware address, all in lower case hexadecimal with dash separators; the file name has the `01-<MAC address>` format. For example, if the Ethernet address is 00:01:AF:17:5E:90 it would search for the file name 01-00-01-af-17-5e-90.

Next, it looks for a file name based on its assigned IP address in upper case hexadecimal. For example, if the assigned IP address is 192.168.1.1, it checks for a file named C0A80101. If that file is not found, it will remove one trailing hexadecimal digit of the hexadecimal IP address file name and iteratively try down to just the first digit (C0A8010, then C0A801, then C0A80, ... C).

If none of these module-specific files exist, it looks for a file named **default**. Module-specific files allow each module (or a group of modules) to boot using different kernel and file system images. The contents of any of these files follows a similar format:

```
DEFAULT <kernel file name> initrd=<file system image name>
console=ttyS0,9600 root=/dev/ram0 <other options>
```

where:

<kernel file name> includes the path along with the kernel file name in the /tftpboot directory. For example, in the /tftpboot location, if there is a PrAMC-7311 directory that contains the kernel and ramdisk, the <kernel file name> is /PrAMC-7311/kernel.

<file system image name> includes the path along with the system image file in the /tftpboot directory. For example, in the /tftpboot location, if there is a PrAMC-7311 directory that contains Linux root file system image, then the <file system image name> is /PrAMC-7311/initramfs.cpio.gz.
<other options> are any additional desired Linux command line options.

For a diskless PrAMC-7311, the file would normally contain:

```
DEFAULT /PrAMC-7311/kernel initrd=/PrAMC-7311/initramfs.cpio.gz
console=ttyS0,115200n8 rdinit=/sbin/init selinux=0
```

2.3 Installation Procedures

You can use the following methods for installation/boot.

Booting the PrAMC-7311 Using a Ramdisk

Diskless Client Network Boot Installation

The detailed steps are described in the following sections.

2.3.1 Booting the PrAMC-7311 Using a Ramdisk

This section describes about booting the PrAMC-7311 using the ramdisk USB boot device.

Make sure that you have set up a TFTP server on a host and create a directory structure for booting images and installing files. For more information, see *Configuring TFTP, DHCP and PXE*.

2.3.1.1 Creating a Ramdisk USB boot device

The following is an example for creating a ramdisk boot device on a USB device:

```
root@localhost:/root> flashrc
************************************************************
**  Installation of RAMDISK Boot on a disk or flash device **
************************************************************
The following disks are a available:
/dev/sda: 500.1 GB
/dev/sdb: 1015 MB
/dev/sdc: 4009 MB
Select the device of the onboard Flash where you want to have the
filesystem installed (e.g.
sdc) []: sdc
Verifying USB onboard flash drive device: sdc...
 Device:  ok
 Checking for necessary commands...

       awk..............[ exists ]
     mount..............[ exists ]
    umount..............[ exists ]
     mkdir..............[ exists ]
     rmdir..............[ exists ]
        rm..............[ exists ]
        cp..............[ exists ]
        mv..............[ exists ]
      date..............[ exists ]
      grep..............[ exists ]
        dd..............[ exists ]
       sed..............[ exists ]
  ifconfig..............[ exists ]
    mke2fs..............[ exists ]
    reboot..............[ exists ]
    sfdisk..............[ exists ]
       cut..............[ exists ]
       seq..............[ exists ]
   sha1sum..............[ exists ]
        tr..............[ exists ]
      tftp..............[ exists ]
   ntpdate..............[ exists ]

 Necessary commands found, safe to continue....
```

```
Caution: Installation will erase all data on the device /dev/sdc!
 Do you wish to begin the installation? [y/n] y
------
Verifying size of /dev/sdc...    [OK]
Creating a partition, please wait ...
Formatting......Done
Which TFTP server do you wish to use? [xxx.xxx.xxx.xxx] <Your TFTP Server>
TFTP Server active.

 What is the installation files directory?
[] builds/PrAMC7311 <Subdirectory under /tftpboot>

Downloading kernel from /tftpboot/builds/PrAMC7311
Are you sure you want to continue connecting (yes/no)? yes
....Done.
Downloading initramfs.cpio.gz from /tftpboot/builds/PrAMC7311
....Done.
 Setting up boot device...
....Done.
 Unmounting local filesystems...Now you should be able to boot from flash
Make sure that boot order in BIOS is set correctly
root@localhost:/root> shutdown -r now
```

2.3.1.2 Creating a Bootable Disk Device

The following is an example for creating bootable device on a SATA or other storage device with at least 30GB of space.

```
root@F140-7311-09:/root> hdiskrc
*************************
** Disk Installation   **
*************************
 Checking for necessary commands...

       awk..............[ exists ]
     chroot.............[ exists ]
      mount.............[ exists ]
     umount.............[ exists ]
        tar.............[ exists ]
       gzip.............[ exists ]
      mkdir.............[ exists ]
      rmdir.............[ exists ]
```

```
          rm..............[ exists ]
          cp..............[ exists ]
          mv..............[ exists ]
        date..............[ exists ]
       chmod..............[ exists ]
       chown..............[ exists ]
        grep..............[ exists ]
          dd..............[ exists ]
        stty..............[ exists ]
         sed..............[ exists ]
  fsck.ext3..............[ exists ]
   ifconfig..............[ exists ]
     mke2fs..............[ exists ]
     mkswap..............[ exists ]
     reboot..............[ exists ]
      fdisk..............[ exists ]
     sfdisk..............[ exists ]
    hwclock..............[ exists ]
      blkid..............[ exists ]
        cut..............[ exists ]
       tftp..............[ exists ]
        seq..............[ exists ]
    sha1sum..............[ exists ]
         tr..............[ exists ]
    ntpdate..............[ exists ]
   smartctl..............[ exists ]

 Necessary commands found, safe to continue....
 ------
The following disks are a available:
/dev/sda: 500.1 GB
/dev/sdb: 1015 MB
/dev/sdc: 4009 MB

 ------
 Select the disk/flash device where you want to have the filesystem
installed (
.g. sdc) []: sda
Verifying disk device: /dev/sda...done.

 Do you wish to begin the installation? [y/n] y
```

```
Verifying hard drive size of /dev/sda...   500  [OK]
Swapspace: 3942 Megabyte
Checking that no-one is using this disk right now ...
OK

Disk /dev/sda: 60801 cylinders, 255 heads, 63 sectors/track
Old situation:
Units = mebibytes of 1048576 bytes, blocks of 1024 bytes, counting from 0

   Device Boot Start   End    MiB    #blocks   Id  System
/dev/sda1         0+  1004-  1005-   1028159+  83  Linux
/dev/sda2   *  1004+  6008-  5005-   5124735   83  Linux
/dev/sda3      6008+ 11013-  5005-   5124735   83  Linux
/dev/sda4     11013+ 476937- 465925- 477106402+   5  Extended
/dev/sda5     11013+ 16017-  5005-   5124734+  83  Linux
/dev/sda6     16017+ 21022-  5005-   5124734+  83  Linux
/dev/sda7     21022+ 24968-  3946-   4040347   82  Linux swap / Solaris
/dev/sda8     24968+ 28968-  4001-   4096574+  83  Linux
/dev/sda9     28968+ 29486-   518-    530144+  83  Linux
/dev/sda10    29486+ 30004-   518-    530144+  83  Linux
  New situation:
Units = mebibytes of 1048576 bytes, blocks of 1024 bytes, counting from 0
   Device Boot Start   End    MiB    #blocks   Id  System
/dev/sda1         0+  1004-  1005-   1028159+  83  Linux
/dev/sda2   *  1004+  6008-  5005-   5124735   83  Linux
/dev/sda3      6008+ 11013-  5005-   5124735   83  Linux
/dev/sda4     11013+ 476937- 465925- 477106402+   5  Extended
/dev/sda5     11013+ 16017-  5005-   5124734+  83  Linux
/dev/sda6     16017+ 21022-  5005-   5124734+  83  Linux
/dev/sda7     21022+ 24968-  3946-   4040347   82  Linux swap / Solaris
/dev/sda8     24968+ 28968-  4001-   4096574+  83  Linux
/dev/sda9     28968+ 29486-   518-    530144+  83  Linux
/dev/sda10    29486+ 30004-   518-    530144+  83  Linux
Successfully wrote the new partition table

Re-reading the partition table.

If you created or changed a DOS partition, /dev/foo7, say, then use dd(1)
to zero the first 512 bytes: dd if=/dev/zero of=/dev/foo7 bs=512 count=1
(See fdisk(8).)
Formatting...1...2...3...5...6...Done
```

```
Creating Swap...Done

What is the installation files directory?
[] builds/PrAMC7311 <Subdirectory under /tftpboot>
Downloading files.sha1sum from /tftpboot/builds/PrAMC7311
....Done.
 Downloading kernel from /tftpboot/builds/PrAMC7311....
Done.
 Downloading initramfs.cpio.gz from /tftpboot/builds/PrAMC7311....
Done.
 Do you wish to set the TimeZone? [Y/n]
Do you wish to use NTP to set the current time? [Y/n] n
 Do you wish to set the date manually? [Y/n] n
 Default date will be used.
 Checking SHA1 Checksum for kernel...Good
 Flushing kernel...Done
 Checking SHA1 Checksum for initramfs.cpio.gz...Good
Done
echo "Timezone set to: America/Phoenix" in /etc/profile

 -------
 Beginning final configuration
-------
 Choose a hostname for this machine []
Set Root password
 New UNIX password:
BAD PASSWORD: it is too short
Retype new UNIX password:
passwd: password updated successfully

 Root password set.

Done.
 Setting up boot device...
starting SmartCtl on the HDD: /dev/sdasmartctl version 5.38 [x86_64-wrs-
linux-g
u] Copyright (C) 2002-8 Bruce Allen
Home page is http://smartmontools.sourceforge.net/

=== START OF ENABLE/DISABLE COMMANDS SECTION ===
SMART Enabled.
```

```
smartctl version 5.38 [x86_64-wrs-linux-gnu] Copyright (C) 2002-8 Bruce
Allen
Home page is http://smartmontools.sourceforge.net/

=== START OF INFORMATION SECTION ===
Device Model:     TOSHIBA MK5061GSYB
Serial Number:    42H9P1HCT
Firmware Version: ME0A
User Capacity:    500,107,862,016 bytes
Device is:        Not in smartctl database [for details use: -P showall]
ATA Version is:   8
ATA Standard is: Exact ATA specification draft version not indicated
Local Time is:    Wed May 2 02:24:09 2012 GMT
SMART support is: Available - device has SMART capability.
SMART support is: Enabled

=== START OF READ SMART DATA SECTION ===
SMART overall-health self-assessment test result: PASSED

General SMART Values:
Offline data collection status:  (0x82) Offline data collection activity
                                        was completed without error.
                                 Auto Offline Data Collection: Enabled.
Self-test execution status:      (0) The previous self-test routine
complete
                                  without error or no self-test has ever
                                        been run.
Total time to complete Offline
data collection:                (120) seconds.
Offline data collection
capabilities:                   (0x5b) SMART execute Offline immediate.
                                 Auto Offline data collection on/off sup
ort.
                                     Suspend Offline collection upon new
                                        command.
                                        Offline surface scan supported.
                                        Self-test supported.
                                    No Conveyance Self-test supported.
                                        Selective Self-test supported.
SMART capabilities:            (0x0003) Saves SMART data before entering
```

```
                                        power-saving mode.
                                        Supports SMART auto save timer.
Error logging capability:        (0x01) Error logging supported.
                                        General Purpose Logging supported.
Short self-test routine
recommended polling time:        (   2) minutes.
Extended self-test routine
recommended polling time:        ( 128) minutes.
SCT capabilities:              (0x003d) SCT Status supported.
                                        SCT Feature Control supported.
                                        SCT Data Table supported.

SMART Attributes Data Structure revision number: 16
Vendor Specific SMART Attributes with Thresholds:
ID# ATTRIBUTE_NAME          FLAG     VALUE WORST THRESH TYPE      UPDATED
WHEN
FAILED RAW_VALUE
  1 Raw_Read_Error_Rate     0x000b   100   100   050    Pre-fail  Always
       0
  2 Throughput_Performance  0x0005   100   100   050    Pre-fail  Offline
       0
  3 Spin_Up_Time            0x0027   100   100   001    Pre-fail  Always
       2566
  4 Start_Stop_Count        0x0032   100   100   000    Old_age   Always
       9
  5 Reallocated_Sector_Ct   0x0033   100   100   050    Pre-fail  Always
       0
  7 Seek_Error_Rate         0x000b   100   100   050    Pre-fail  Always
       0
  8 Seek_Time_Performance   0x0005   100   100   050    Pre-fail  Offline
       0
  9 Power_On_Hours          0x0032   098   098   000    Old_age   Always
       906
 10 Spin_Retry_Count        0x0033   100   100   030    Pre-fail  Always
       0
 12 Power_Cycle_Count       0x0032   100   100   000    Old_age   Always
       6
191 G-Sense_Error_Rate      0x0032   100   100   000    Old_age   Always
       0
192 Power-Off_Retract_Count 0x0032   100   100   000    Old_age   Always
       0
```

```
193 Load_Cycle_Count        0x0032   100   100   000    Old_age   Always
       737
194 Temperature_Celsius     0x0022   100   100   000    Old_age   Always
       24 (Lifetime Min/Max 21/30)
196 Reallocated_Event_Count 0x0032   100   100   000    Old_age   Always
       0
197 Current_Pending_Sector  0x0032   100   100   000    Old_age   Always
       0
198 Offline_Uncorrectable  0x0030  100  100  000   Old_age  Offline
 0
199 UDMA_CRC_Error_Count    0x0032   200   200   000    Old_age   Always
       0
220 Disk_Shift              0x0002   100   100   000    Old_age   Always
       211
222 Loaded_Hours            0x0032   100   100   000    Old_age   Always
       102
223 Load_Retry_Count        0x0032   100   100   000    Old_age   Always
       0
224 Load_Friction           0x0022   100   100   000    Old_age   Always
       0
226 Load-in_Time            0x0026   100   100   000    Old_age   Always
       351
240 Head_Flying_Hours       0x0001   100   100   001    Pre-fail  Offline
       0

SMART Error Log Version: 1
No Errors Logged

SMART Self-test log structure revision number 1
No self-tests have been logged. [To run self-tests, use: smartctl -t]


SMART Selective self-test log data structure revision number 1
 SPAN  MIN_LBA  MAX_LBA  CURRENT_TEST_STATUS
    1        0        0  Not_testing
    2        0        0  Not_testing
    3        0        0  Not_testing
    4        0        0  Not_testing
    5        0        0  Not_testing
Selective self-test flags (0x0):
After scanning selected spans, do NOT read-scan remainder of disk.
```

```
If Selective self-test is pending on power-up, resume after 0 minute
delay.

 Unmounting local filesystems...done.
root@localhost:/root> shutdown -r now
```

2.3.2 Diskless Client Network Boot Installation

The PrAMC-7311 can be booted as diskless client from a boot server using `pxeboot`. In order to configure the PrAMC-7311 accordingly, the following preparative steps are required.

1. Configure BIOS
2. Reboot the module

Configuring BIOS and Reboot

To configure BIOS, proceed as follows:

1. Connect to the PrAMC-7311 via the serial console interface (see *Accessing the PrAMC-7311 via Serial Console* on page 20).
2. Power up or reboot the PrAMC-7311 and hold down the <F2> key on your keyboard until the BIOS menu appears.
3. Select `Boot` on the top menu.
4. Select the network boot device. Depending on your system configuration, this device is named similar to the following: PCI LAN: IBA GE Slot xx01 v1240. Make sure that this device is enabled, which means it is not part of the *Excluded from boot order* list.
5. Move the network boot device with the <+> key so it is the first enabled device.
6. Save and exit.

   It takes about 30 seconds to reboot.

While the module is booting, observe that it is getting a DHCP address and then loading the kernel and ramdisk image:

```
Trying to load: pxelinux.cfg/00000000-0000-0000-0000-000000000000
Trying to load: pxelinux.cfg/xx-xx-xx-xx-xx-xx
Trying to load: pxelinux.cfg/0AF741BB
boot:
Loading PrAMC-7311/kernel.............................
Loading PrAMC-7311/initramfs.cpio.gz.....................
```

2.4 Upgrading the Software

To upgrade your BBS software, you need to place the new images on the TFTP server.

To upgrade the BBS software for **diskless clients**, you simply reboot and the new BBS files are automatically booted from the TFTP server.

2.5 Adapting the BBS Software to Customer's Needs

The BBS software structure allows a maximum flexibility with regards to customer's adaptations. Software packages can easily be installed into or removed from existing installations.

The following adaptations are possible:

- Modifying the root file system images
- Modifying the Configuration of the Linux Kernel

2.5.1 Modifying the Root File System Images

The root file systems are stored in initramfs file, intitramfs.cpio.gz. In order to add/remove/modify files contained in the root file system, proceed as described in the following procedure. The procedure assumes that the image has been placed on the TFTP server. It also assumes ramdisk.image.gz as file name, but you may simply replace this file name with initrd.gz. The procedure is identical.

As root:

```
# cd /tftpboot/<blade or module to be modified>
# mkdir tmp
# cd tmp
# gzip -dc ../initramfs.cpio.gz | cpio  -i -d -H newc --no-absolute-
filenames
## make modifications
# find . | cpio -o -H newc | gzip > ../new_intramfs.cpio.gz
```

You may now boot or reinstall the updated image.

2.5.2 Modifying the Configuration of the Linux Kernel

To modify the kernel of the chosen Linux distribution, you need to purchase development software from the chosen Linux vendor.

2.6 Serial-Over-LAN

Serial-Over-LAN (SOL) is a feature defined by IPMI specification that allows you to access the serial port, typically the system console port, through Ethernet remotely.

PrAMC-7311 supports SOL over the Ethernet ports to access the payload console (that is, /dev/ttyS1). Additional configurations of MMC firmware are required to use the Serial-Over-LAN.

2.6.1 SOL Channels

PrAMC-7311 supports three SOL channels (since 1.2.0 MMC firmware) as shown in the following table. However, only one SOL session is allowed at a time.

Table 2-4 SOL Channels

SOL/LAN Channel #	Ethernet Device	Description
1	eth0	Front Panel RJ45
2	eth1	AMC Port 0
3	eth2	AMC Port 1

2.6.2 Configuring LAN Interfaces

In general, like Linux, the IPMC/MMC firmware should configure the Ethernet port(s) to enable the LAN interface(s) to use the SOL feature. Such configurations include the MAC address, IP address, network mask, gateway MAC address, and gateway IP address.

As PrAMC-7311 MMC shares the MAC address with the payload (as supported by the Intel I350 Ethernet controller), only IP address, network mask, gateway MAC address, and gateway IP address need to be configured.

2.6.2.1 Set LAN Configuration Parameters Command

The standard `Set LAN Configuration Parameters` command is used to configure the LAN interface.

The ipmitool has a set of user friendly commands, that help to configure the LAN interfaces. Following are the examples, to set IP address, netmask, gateway MAC address, and gate IP address.

```
ipmitool lan set 1 ipaddr 192.168.1.100
ipmitool lan set 1 netmask 255.255.255.0
ipmitool lan set 1 defgw macaddr 01:23:45:67:89:ab
```

```
ipmitool lan set 1 defgw ipaddr 192.168.1.1
```

`ipmitool lan set` command displays the usage of the commands.

NOTE: Not all command options are supported by the IPMC/MMC firmware.

2.6.3 Configuring SOL Parameters

By default, the SOL is disabled. It needs to be explicitly enabled before setting up an SOL session.

2.6.3.1 Set SOL Configuration Parameters Command

The standard `Set SOL Configuration Parameters` command enables the SOL feature.

Following is the example that enables the SOL:

```
ipmitool sol set enabled true 1
```

`ipmitool sol set` command displays the usage of commands.

2.6.4 Configuring User Accounts

User accounts are necessary for SOL access. PrAMC-7311 MMC firmware has three preconfigured user accounts. You can enable or disable the user account, modify the user account name and password.

Table 2-5 User Accounts

User ID	User Name	Password	Privilege Level
1	-	-	User, no SOL access
2	soluser	solpasswd	User, SOL access
3	rmcp	rmcp	User, SOL access

'_' means no user name and password; that is, no authentication.

2.6.4.1 IPMI Messaging Support Commands

The standard IPMI messaging support commands are used to configure the user accounts.

The following are the examples, to set user name and password, and disable the account.

```
ipmitool user set name 2 sol
ipmitool user set password 2 SOL
ipmitool user disable 2
```

`ipmitool user` command displays the usage of the commands

2.6.5 SOL Session Setup with IPMITOOL

IPMITOOL is a well-known tool that supports SOL. The following are some usage examples. For more information, refer to IPMITOOL manual and/or help message.

2.6.5.1 Setting up an SOL Session

To set up the SOL session, one of the cipher suites needs to be specified. PrAMC-7311 supports four cipher suites: 0, 1, 2, and 3. When encryption (i.e. cipher suite 1, 2, or 3) is used, it is required to specify the K_G using the -k option. The default K_G is gkey. You can change it using the Set Channel Security Keys command. For more information, refer to IPMI v2.0 specification.

Following are the example commands for setting up the SOL session with IPMITOOL:

1. connecting with cipher suite 0 (no encryption) from the remote site:
   ```
   ipmitool -I lanplus -H 192.168.1.100 -U sol -P SOL -C 0 sol activate
   ```
2. connecting with cipher suite 1 from the remote site:
   ```
   ipmitool -I lanplus -H 192.168.1.100 -U sol -P SOL -C 3 -k gkey sol
   activate
   ```
3. Changing K_G to newkey from the local payload:
   ```
   ipmitool raw 6 0x56 1 1 1 0x6e 0x65 0x77 0x6b 0x65 0x79 0 0 0 0 0
   0 0 0 0 0 0 0 0 0
   ```

A new key may be used then to set up new SOL connections:

```
ipmitool -I lanplus -H 192.168.1.100 -U sol -P SOL -C 3 -k newkey
sol activate
```

2.6.6 Exiting the SOL Session

Like a telnet session, a special escape string, **~.**, is used to exit from the SOL connection. Type **~?** in the SOL session for more information on escape string

NOTICE

Due to the PrAMC-7311 hardware limitation, payload must be activated to start the SOL session. When payload gets deactivated, the existing SOL session will be disconnected.

2.6.7 Console Port Baud Rate

By default, the MMC firmware configures SOL console port baud rate as 115200. The standard Set SOL Configuration Parameters command may be used to modify the baud rate.

The SOL baud rate shall not be changed alone or it will break the SOL.

The customer shall change the following:

MMC side SOL console serial baud rate

BIOS side SOL console serial baud rate

OS side SOL console serial baud rate at the same time to the same baud rate setting.

Changing the flow control may break the SOL, and block the payload service.

The following is the example command to configure the SOL serial port baud to 9600 (non-volatile).

```
ipmitool sol set non-volatile-bit-rate 9.6.1.
```

2.6.8 Console Port Flow Control

By default, the MMC firmware enables the software flow control over the SOL console port. It may be reconfigured to no flow control or hardware flow control with the standard Set Serial Configuration command.

NOTE: The Set Serial Configuration command configures the flow control as well as the non-volatile baud rate for all SOL channels. The new configuration does not take effects until the next SOL session.

The following are the example commands to configure the flow control (non-volatile):

Example1:

Configure the SOL serial port with no flow control, 115200 baud

```
ipmitool raw 12 0x10 8 0 1
ipmitool raw 12 0x10 8 7 0 0x0a
ipmitool raw 12 0x10 8 0 2
```

Example2:

Configure the SOL serial port with hardware flow control, 115200 baud

```
ipmitool raw 12 0x10 8 0 1
ipmitool raw 12 0x10 8 7 0x40 0x0a
ipmitool raw 12 0x10 8 0 2
```

Example3:

Configure the SOL serial port with software flow control, 115200 baud

```
ipmitool raw 12 0x10 8 0 1
ipmitool raw 12 0x10 8 7 0x80 0x0a
ipmitool raw 12 0x10 8 0 2
```

Firmware Upgrade Facility

3.1 Overview

The Firmware Upgrade Facility (FUF) provides a uniform way to upgrade firmware on hub blades, node blades, and AMC modules. It consists of a Firmware Upgrade Command-line Utility (FCU), flash device drivers, and specially prepared firmware recovery image files.

3.2 Firmware Recovery Image Files

The PrAMC-7311 supports the HPM.1 IPMI standard. FCU works with specially prepared firmware recovery image (FRI) files or with HPM.1 compatible files. By default, the firmware files for the current hardware configurations are loaded as part of the BBS software in /opt/bladeservices/rom when the module-specific firmware support packages are installed. The following firmware packages are currently available:

Filename	Description
pramc-7311-cpu.fri	BIOS firmware image for PrAMC-7311
pramc-7311-ipmc-boot-<version>.hpm	IPMC bootloader
pramc-7311-ipmc-<version>.hpm	IPMC firmware

3.3 Backup Concept

The BIOS firmware for the PrAMC-7311 is stored in redundant, persistent memory devices. This allows the firmware image in one bank to serve as a backup for the other bank. This is particularly useful for firmware upgrades.

During normal operation, the CPU determines which bank to boot from based on a chip select signal controlled by the IPMC. This bank is considered the active boot device. FCU will only allow you to upgrade an inactive device. It determines whether a device is active or inactive by querying the IPMC to learn which device is marked to be used at boot. Because you can change the active device with the FCU mark operation, active status does not necessarily indicate which device was used on the last boot. It simply represents which device is set for use on the next boot.

The IPMC firmware consists of a boot loader as well as an active and a stand-by IPMI firmware. The boot loader maintains both the active and stand-by firmware in the flash memory of the PrAMC-7311. Both the boot loader as well as the IPMI firmware images can be upgraded.

Each time the IPMC firmware is upgraded, the most recent firmware version is kept in flash memory and the older firmware version is overwritten by the new one. Once the new IPMI firmware is programmed, the IPMC resets itself to boot from the new image. The boot loader validates the new IPMC firmware. Provided the IPMC can power up successfully the current image is made active and the previously active image is made backup. In case of power-up failures, the boot loader automatically recovers from crisis and boots from the previous image.

3.4 fcu—Firmware Upgrade Command-Line Utility

Description

The Firmware Upgrade Command-line Utility (FCU) allows you to:

- Query the current versions of firmware installed on a module and determine which firmware devices are active
- Verify that a specified upgrade image is sound and compatible with the current hardware
- Upgrade a firmware image
- Mark a device to be used as the boot source on the next reset
- Display detailed information about a firmware upgrade image file

By default, the FCU binary executable is installed in /opt/bladeservices/bin. This directory has been added to the PATH environment variable.

FCU works in conjunction with device drivers created specifically for the flash devices on SMART EC modules..

The FCU verify and upgrade operations require specially prepared FRI files or HPM.1 compatible image files. For more information, see *Firmware Recovery Image Files on page 41*.

Synopsis

```
fcu --help [-t<slave address>]
fcu --version
fcu -q [-d <device-id>]
fcu -v -f <filename>
fcu -u -f <filename>
fcu -a -f <filename>
fcu -s <filename>
fcu -m -b <bank-letter> -d <device-id>
```

Parameters

`-a`
`--full-upgrade`

This option is a shortcut for performing the verify, upgrade, and mark operations. The file option -f is required. This option should not be combined with other operations.

`-b <bank-letter>`
`--bank=<bank-letter>`

Specifies the flash bank (A or B, for example) to mark for next boot, where <bank-letter> is the letter designating a specific bank. This option is used with the mark operation. Use the query option -q to list available banks.

`-d <device-id>`
`--device=<device-id>`

Specifies a target firmware device, where <device-id> is the name of the device. This option is used with the mark or query operations. Device ID values vary by hardware. You can display supported devices on a given module by using **`fcu --help`**. Currently supported values are listed in the following table.

Device ID	Description
pramc-7311-cpu	CPU firmware device for PrAMC-7311
pramc-7311-hpm.1-ipmc	IPMC firmware device for PrAMC-7311

`-f <filename>`
`--file=<filename>`

Specifies the FRI or HPM.1 image file, where <filename> is the complete path and filename of the file. This option is used with the verify and upgrade operations.

`--force`

This option allows the installation of images with non-matching part-number and part-revision FRU data fields. This option should be used with extreme caution only because installing an incompatible image on a device may render it inoperable.

`--help`

Displays a brief message describing command usage. It also displays a list of the devices supported on the module.This option is exclusive and should not be used with other options.

`-m`
`--mark`

Tells FCU to set the boot select so that on the next boot the specified firmware bank will be active. When mark is combined with the upgrade operation, there is no need to specify a bank; the bank just upgraded will be marked. Otherwise, you must specify a bank and a device.

Currently, the mark operation only supports CPU firmware devices.

`-q`
`--query`

Tells FCU to return firmware information for a specific device (if used with -d) or information about all firmware devices. The query operation is exclusive and is not intended to be combined with other operations.

`-s`
`--show`

Display information about the target which the selected upgrade file corresponds to. Use this command to display the current firmware version status of the target device.

`-u`
`--upgrade`

Indicates FCU to upgrade the currently inactive bank of the device specified by the target FRI file. The file option -f is required. The upgrade operation may be combined with the verify and mark operations.

`-v`
`--verify`

Informs FCU to verify the image file specified by the required -f option. This operation verifies that the specified file is sound and compatible with the current hardware. The verify operation may be combined with the upgrade and mark operations.

`--version`

Displays version information for the utility. This option is exclusive and should not be used with other options.

Usage

Some FCU options can be combined. Some options are exclusive. The following list describes the valid option combinations:

`--full --file=<filename>`

`--help`

`--mark --bank=<bank-letter> --device=<device-id>`

```
--query
--query --device=<device-id>
--upgrade --file=<filename>
--show --file=<filename>
--upgrade --mark --file=<filename>
--verify --file=<filename>
--verify --upgrade --file=<filename>
--verify --upgrade --mark --file=<filename>
--version
```

Multi-character options may be abbreviated so long as they are unique. For example, `--full` is equivalent to `--full-upgrade`. Typing `--ver`, however, will not work since it matches both `--verify` and `--version`.

Single-character options may be combined without repeating the hyphen, as in these examples:

```
fcu –vf /opt/bladeservices/rom/<filename>
fcu –q –d <device-id>
fcu –q –d-t 0x90 <device-id>
fcu –mb a –d <device-id>
```

Options are not case-sensitive. For example, `--help` is equivalent to `--HeLp`. However, option arguments, such as filename and device ID, are case-sensitive.

When upgrading firmware, it is strongly recommended that you upgrade only one device at a time. While FCU performs many checks during upgrade to ensure success, if something goes wrong and both firmware banks become corrupted, the module will be inoperable.

3.5 Upgrading a Firmware Image

This section describes recommended procedures for upgrading firmware devices. The procedures for upgrading BIOS and IPMC differ slightly.

3.5.1 BIOS Upgrade

The BIOS can only be upgraded from the module on which the BIOS is running. You have to upgrade the BIOS by using fcu. The hpmcmd command cannot be used to upgrade the BIOS.

Upgrading the BIOS for PrAMC-7311

Follow these steps to upgrade the BIOS.

1. Query the current BIOS firmware images on the module.

   `fcu -qd <device-id>`
   where <device-id> is the name of the device to be upgraded.

2. Verify the integrity of the upgrade file.

   `fcu --verify -f <path+filename>`

   where <path+filename> is the complete path to the upgrade file, for example:

   `fcu --verify -f /opt/bladeservices/rom/pramc-7311-cpu.fri`

   Make sure the upgrade image is newer than the current firmware image.

3. Upgrade the firmware image,
   `fcu --upgrade -f <path+filename>`

   where <path+filename> is the complete path to the upgrade file, for example:
   `fcu --upgrade -f /opt/bladeservices/rom/pramc-7311-cpu.fri`

FCU writes the new image and then reads back the image and performs a binary compare to ensure that the write was successful. If the upgrade was not successful, you will see an error message. Try the upgrade again. If it is still not successful, contact your SMART EC representative.

4. Query the new image to ensure that the version information is correct,
   `fcu -qd <device-id>`

   where <device-id> is the name of the upgraded device, for example:
   `fcu -qd pramc-7311-cpu`

5. Mark the new image as active so that it will be used for the next boot, for example:
   `fcu --mark -b <bank-letter> -d <device-id>`

   where <bank-letter> is the letter of the upgraded bank and <device-id> is the name of the upgraded device, for example:
   `fcu --mark -b a -d pramc-7311-cpu`

3.5.2 IPMC Upgrade

Upgrading IPMC for PrAMC-7311

Follow these steps to upgrade an IPMC:

1. Query the current IPMC firmware images on the module.

   `fcu -qd <device-id>`
   where <device-id> is the name of the device to be upgraded.

2. Verify the integrity of the upgrade file,

   `fcu --verify -f <path+filename>`
   where <path+filename> is the complete path to the upgrade file, for example:
   `fcu --verify -f /opt/bladeservices/rom/pramc-7311-ipmc.fri`

   Make sure the upgrade image is newer than the current firmware image.

3. Upgrade the firmware image,

   `fcu --upgrade -f <path+filename>`
   where <path+filename> is the complete path to the upgrade file, for example:
   `fcu --upgrade -f /opt/bladeservices/rom/pramc-7311-ipmc.fri`

   FCU writes the new image and then reads back the image and performs a binary compare to ensure that the write was successful. If the upgrade was not successful, you will see an error message. Try the upgrade again. If it is still not successful, contact your SMART EC representative.

4. Query the new image to ensure that the version information is correct,

   `fcu -qd <device-id>`
   where <device-id> is the name of the upgraded device, for example:
   `fcu -qd pramc-7311-ipmc`

If the version you have just installed now is the active image, the upgrade was successful.

Linux Distribution Description

4.1 Distribution Description

The BBS software for the PrAMC-7311 is based on Windriver 3.0.3 RCPL21. The kernel version is 2.6.27.57-grsec-WR3.0.3bc_cgl.

4.2 Login

A Linux shell can be accessed via the face plate serial port.

You need a serial console for initial login, and then you can configure the network services to comply with your network specifications.

- If you use a serial console or terminal emulator, the serial port settings are 115200 baud, no parity, eight data bits, and one stop bit.
- If you use SSH, see *Network Services Configuration on page 49* for default IP address assignments.

The available login account is **root** with the password **root**.

4.3 Network Services Configuration

The following Ethernet interfaces are configured at startup:

Interface	IP Address
eth0	Front Panel
eth1	Base
eth2	Fabric
eth3	Base
eth4	Fabric

A secure shell server starts in run levels 2–5 and listens on all Ethernet interfaces.

Hardware Platform Management

5.1 Overview

Hardware management is based on the Intelligent Platform Management Interface (IPMI) specification. IPMI commands can be complex and cumbersome. To facilitate module-level management, SMART EC provides the Hardware Platform Management (HPM) package that provides a set of commands that are based on IPMI commands but which are easier to use than the IPMI command itself. An HPM command can encapsulate a sequence of IPMI commands. An HPM command can be the unifier for OEM IPMI commands that are different on different module types, for example reading the BIOS boot bank. For a catalogue of supported IPMI commands of the module, refer to the respective IPMI manual.

Since HPM provides full access to the underlying IPMI subsystem, HPM commands should be used with special care. Improper usage of HPM commands may cause system corruption or system failure. Examples of improper usage are modifying FRU data, change blue led state, change e-keying, and so on.

The HPM package consists of

- HPM daemon called hpmagentd
- Command line client called hpmcmd
- Script framework for managing shutdown and reboot events

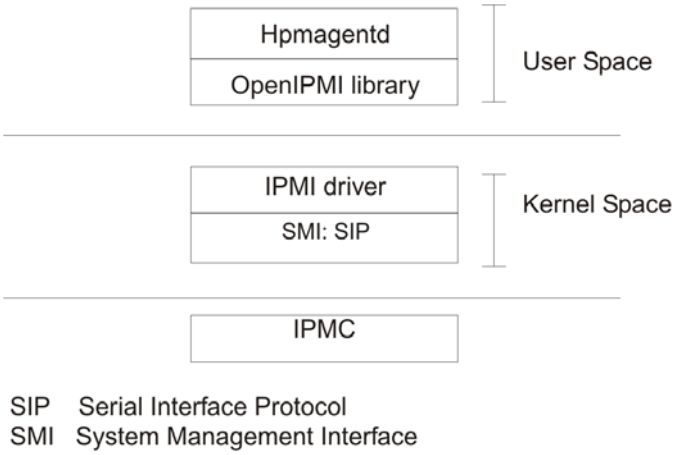
The hpmcmd sends a given HPM command to the hpmagentd and displays the received response on the console. The hpmagentd executes the incoming HPM commands and returns the result to a hpmcmd client.

HPM commands include:

- Retrieving FRU data
- Reading and controlling status of IPMI-controlled LEDs
- Executing shutdown and reboot scripts in response to cold reset or graceful reboot requests
- Communicating local slot location information

The hpmagentd makes use of OpenIPMI to talk to the local IPMC. OpenIPMI consists of two main parts: A device driver that goes into the Linux kernel, and a user-level library. The following picture shows the software levels that are involved in the HPM architecture:

Figure 5-1 Software Levels of the HPM Architecture



The System Management Interface (SMI) driver provides the low level interface for talking to the IPMC. The communication is based on a serial interface protocol.

If you need more information about the software aspects of the blade and module IPM controller, refer to the respective IPMI manual.

5.2 hpmagentd—HPM Agent Daemon

Description

The HPM agent daemon handles local communication to the intelligent platform management controller (IPMC) on a module using the SMI. This SMI gets set up by the OpenIPMI driver.

By default, the hpmagentd binary executable is installed in */opt/bladeservices/bin/*. This directory has been added to the PATH environment variable.

This daemon has an init script called hpm that will start the daemon in run level 3 with the default settings.

When hpmagentd receives a graceful reboot or shutdown alert from the IPMC, it will call the respective script to run the reboot or shutdown sequence.

Synopsis

```
hpmagentd [-l log-level] [-r reboot-script] [-s shutdown-script]
hpmagentd {-i | -u | -h | -v}
```

Parameters

`-l log-level`

Specifies the level of message logging, where log-level is one of the standard syslog levels:

Log Level	Description
0	Emergency
1	Alert
2	Critical
3	Error
4	Warning
5	Notice (default)
6	Information
7	Debug

`-r reboot-script`

Specifies a graceful reboot script that will be called when a module graceful reboot request is received by the MMC, where reboot-script is the complete path and filename of the target script. The default is /opt/bladeservices/bin/hpmreboot (see *hpm—Shutdown and Reboot Scripts on page 55*).

`-s shutdown-script`

Specifies a shutdown script that will be called when a module shutdown request is received by the MMC, where shutdown-script is the complete path and filename of the target script. The default is */opt/bladeservices/bin/hpmshutdown* (see *hpm—Shutdown and Reboot Scripts* on page 59).

`-i`

hpmagentd runs interactively, that is it will not run as daemon.

`-u | -h`

Displays a brief message about command usage.

`-v`

Displays the version of hpmagentd

5.3 hpm—Start-Up Script

Description

An HPM agent init script, hpm, allows you to start, stop, and restart the HPM agent daemon using the agent's default option settings. By default, this script is installed in the /opt/bladeservices/etc/init.d directory during installation of the BBS software. It is also linked into the system start-up directories to automatically start the HPM agent when the system boots.

Synopsis

```
hpm {start | stop | restart | force-reload}
```

Parameters

`start`

Starts the hpm agent daemon.

`stop`

Terminates the hpm agent daemon.

`restart`

Terminates and then starts the hpm agent daemon.

`force-reload`

Terminates and then starts the hpm agent daemon.

5.4 hpm—Shutdown and Reboot Scripts

Description

At any time during normal operation, a shelf manager may issue a shutdown (FRU Activation Deactivate) or graceful reboot (FRU Control Reboot) request to the MMC on a given module. The MMC then forwards this information to the HPM agent. The HPM agent listens for such requests from the MMC. When it receives a request, it calls the respective script to run the reboot or shutdown sequence. In case of a shutdown indication, all running processes should be notified about the shutdown. In case of a reboot notification, the payload is responsible for invoking the reboot procedure. The MMC is not involved in this process. This allows processes currently running on the module to prepare for shutdown. After the notification, it takes roughly 30 seconds before the payload is powered off.

Two default scripts, hpmshutdown and hpmreboot, are installed by default in the /opt/bladeservices/bin directory. Currently, these scripts simply print a banner indicating they have run and then issue `shutdown -h now` (hpmshutdown script) or `reboot` (hpmreboot script).

You may modify the default scripts to suit the needs of your system application or create new scripts. If you create new scripts, use -s and -r options when starting hpmagentd to specify the new locations and names of the scripts. You may also need to update the hpm start up script in /opt/bladeservices/etc/init.d/hpm.

Synopsis

```
hpmshutdown
hpmreboot
```

5.5 hpmcmd—HPM Command Utility

Description

The HPM command utility accepts commands from the user and forwards them to the HPM agent.

Only one HPM command can be outstanding with the HPM agent at any particular moment. This means that even though multiple instances of hpmcmd can be started, the HPM agent will handle only one command at a time. Once a command is sent, the hpmcmd program waits until the answer from the HPM agent is received or until a time-out occurs.

The HPM command utility can be started in interactive mode, where a prompt is displayed and the user enters commands; it can read in a file of commands; or it can process a single command.

By default, the hpmcmd binary executable is installed in /opt/bladeservices/bin. During installation of the BBS software, this directory is added to the PATH environment variable.

If you do not provide any options you will see the following prompt once the program starts running:
`hpmcmd>`

From there you can start executing commands.

Synopsis

```
hpmcmd [-p new-prompt] [-o output] [-i input | -c command]
hpmcmd [--prompt new_prompt] [--output_file output] [--input_file

 input | -cmd_line command]
```

Parameters

`-p new-prompt`

Specifies the prompt you would like to have for the hpmcmd interactive mode, where new-prompt is any string. The default prompt is `hpmcmd>`. This option should not be combined with the -r or -c options.

`-i input-file`

Specifies the name of a file with HPM commands, where input-file is the complete path and filename of the target file, a standard ASCII file with one command per line (comments are not supported). The default is Standard Input (stdin). This option should not be combined with the -c option.

Once it has executed all commands in the file, hpmcmd terminates.

`-o output-file`

Specifies the name of an output file, where output-file is the complete path and filename of the target file. The default is Standard Output (stdout).

`-c command`

This option executes a single command and terminates, where command is one of the supported commands. This allows you to use the arrow history functions supported in the base shell; a history is not available inside the hpmcmd program. This option should not be combined with the -i option.

If this option is combined with -o, -c should be last option entered, since all arguments that follow -c on the command line will be considered part of the command.

5.5.1 Command Overview

The following table lists all commands from the hpmcmd program and shows for which blades/modules they are available. You can display this list and a short command description using the help command (see section *help on page 65*). A detailed description of the commands is given in section *Supported Commands on page 58*.

Table 5-1 Command Overview

Command	Description
bootbankget	Gets the bootbank to boot from
bootbankset	Sets the bootbank to boot from
bye	Exit the hpmCmd program
cmd	Execute any IPMI command
deviceid	Gets the Device Id
exit	Exit the hpmcmd program
frudata	Allows to get FRU info in hex numbers
fruinfoget	Gets string fields from the FRU
fruinv	Allows to get the FRU size and addressable units
fruread	Allows to read x number of bytes from the FRU
get-shut-tout	Get payload shutdown timeout in milliseconds
help	List of hpmcmd commands.
ipmbaddress	Shows the local board IPMB address
ipmcdevice	Shows the payload interface to the IPMC
ledget	Gets the state of a specific FRU LED
ledprop	Get the LED properties for this FRU
ledset	Controls the state of a specific FRU LED
loglevelget	Gets the hpmagentd log level
loglevelset	Sets the hpmagentd log level(0-7)
macaddress	Lists the MAC adresses

Table 5-1 Command Overview (continued)

Command	Description
partnumber	Gets the board part number
physlotnumber	Lists the physical slot location
portget	Shows the current state E-Key governed intfs
portset	Enables/Disables ports in a channel
posttypeget	Gets the POST type to run at boot
posttypeset	Sets the POST type to run at boot
quit	Exits the hpmcmd program
rebootpath	Gets hpmagentd reboot script path
sdr	Shows the SDR records
sdr_dump	Shows the SDR records in binary and hex format
sdrinfo	Shows the SDR information
sendamc	Sends any IPMI command to a remote AMC or MMC of a remote IPMC IPMB-L
sendcmd	Sends an IPMI request to the IPMC
set-shut-tout	Sets payload shutdown timeout in milliseconds
shelfaddress	Gets the shelf address string
shutdownpath	Gets hpmagentd shutdown script path
slotnumber	Shows the board logical slot number
version	Shows the hpmcmd version and the hpmagentd version
watchdog	Control Payload WDT functionality

5.5.2 Supported Commands

This section lists the supported commands. All commands are case insensitive. The examples illustrate the use of hpmcmd in single-command mode (-c). If you start hpmcmd without the -c or -i options (that is, interactive mode), you simply enter these commands at the HPM command prompt.

Some of the hpm commands can be sent to a remote IPMC by specifying the -t option. This option is not mandatory. If it is not specified, the command is sent to the local IPMC.

The following commands support the -t option:

- deviceid
- frudata
- fruinfoget
- fruinv
- fruread
- ledget
- ledset
- macaddress
- partnumber

The following sections describe the available commands.

5.5.2.1 bye

Description

This command ís for exiting the hpmcmd program when running in interactive mode.

Synopsis

bye

5.5.2.2 bootbankget

Description

This command retrieves the boot bank which is currently marked as active for the CPU specified by payload_cpu_selector.

Firmware for the CPU is stored in redundant, persistent memory devices. This allows the firmware image in one bank to serve as a backup for the other bank. During normal operation, the CPU on a module determines which bank to boot from based on a GPIO signal controlled by the IPMC. This bank is considered the active boot device.

Because you can change the “active” device with the hpmcmd bootbankset command, active status does not necessarily indicate which device was used on the last boot. It simply represents which device is set to be used on the next boot.

Synopsis

bootbankget <payload_cpu_selector>

Parameters

`payload_cpu_selector`

Is an integer between 0 and the number of CPU devices supported on the module.

On the PrAMC-7311 the only valid value for payload_cpu_selector is 0.

Example

```
hpmcmd -c bootbankget 0
```

5.5.2.3 bootbankset

Description

This command sets the boot bank for a particular CPU from which the module is supposed to boot.

Synopsis

```
bootbankset <payload_cpu_selector> <newBootBank>
```

Parameters

`payload_cpu_selector`

Is an integer between 0 and the number of CPU devices supported on the module.

`newBootBank`

Can be set to BANK0 or BANK1

Example

```
hpmcmd -c bootbankset 0 bank1
```

5.5.2.4 cmd

Description

This command allows you to enter commands understood by the IPMC. Commands are entered as a sequence of hexadecimal numbers as defined in the *IPMI 1.5 Specification*.

Synopsis

```
cmd <ipmi address> <netfn cmd> <cmd data>
```

Parameters

`ipmi address`

The IPMI address specifies the IPMC that receives the command, it can be the local IPMC or another IPMC on the IPMB. The IPMI address for the local IPMC consists of <f LUN> where f is the BMC channel number. The IPMI address for a remote IPMC consists of <0 SA LUN>.

`netfn cmd`

Identifies the command type.

`cmd data`

Specifies the message data associated with the command.

Example

GetDeviceId command to the local IMPC:
`hpmcmd -c cmd f 0 6 1`

GetDeviceId command to the remote IPMC on address 9a:
`hpmcmd -c cmd 0 9a 0 6 1`

GetDeviceId command to the remote IPMC on address 7a:
`hpmcmd -c cmd 0 7a 0 6 1`

5.5.2.5 deviceid

Description

This command retrieves the raw IPMI Get Device ID response and decodes the IPMI message.

Synopsis

```
deviceid -t [ipmbAddr[:mmcAddr]]
```

Parameters

`-t`

Sends the command to ipmbAddre:mmcAddr. ipmbAddr is the string lc if it is a local mmcAddr.

Example

```
hpmcmd -c deviceid
```

5.5.2.6 exit

Description

This command is for exiting the hpmcmd program when running in interactive mode.

Synopsis

```
exit
```

5.5.2.7 frudata

Description

This command dumps the content of the FRU data in hexadecimal format.

Synopsis

```
frudata <fruid> [-t ipmbAddr[:mmcAddr]]
```

Parameters

`fruid`

Is 0 for the main module.

`-t`

Sends the command to ipmbAddr:mmcAddr. ipmbAddr is the string lc if it is a local mmcAddr.

Example

```
hpmcmd -c frudata 0
```

5.5.2.8 fruinfoget

Description

This command retrieves information from the specified FRU.

Synopsis

```
fruinfoget <fruid> [field] [-v] [-t ipmbAddr[:mmcAddr]]
```

Parameters

`fruid`

Is `0` for the main module. On ATCA-7107, ATCA-7221, and ATCA-F120, `fruid` is `1` for the rear transition module. On ATCA-C110, `fruid` is `1`, `2`, `3`, and `4` for AMCs on bay 1, 2, 3, and 4 respectively and 5 for the rear transition module.

`field`

Is one of the following data fields. If no field is specified, it retrieves the whole fruinfo for that FRU.

Field	Description
bmanufacturer	Board manufacturer
bproductname	Board product name
bserialnumber	Board serial number
bpartnumber	Board part number
pmanufacturer	Product manufacturer
pproductname	Product product name
ppartnumber	Product part number
pversion	Product version number
pserialnumber	Product serial number
passettag	Product inventory asset identifier

`-v`

Verbose mode to get point-to-point connectivity information where no specific field is requested.

`-t`

Sends the command to ipmbAddr:mmcAddr. ipmbAddr is the string lc if it is a local mmcAddr.

Example

```
hpmcmd -c fruinfoget 1 bmanufacturer
```

The following example for fruinfoget is without fields and -v option.

```
hpmcmd -c fruinfoget 0
```

5.5.2.9 fruinv

Description

This command retrieves the FRU size and the addressable unit for the specified FRU.

Synopsis

```
fruinv <fruid> [-t ipmbAddr[:mmcAddr]]
```

Parameters

`fruid`

Is 0 for the main module. On ATCA-7107, ATCA-7221, and ATCA-F120, fruid is 1 for the rear transition module (if supported). On ATCA-C110, fruid is 1, 2, 3, and 4 for AMCs on bay 1, 2, 3, and 4 respectively and 5 for the rear transition module.

`-t`

Sends the command to ipmbAddr:mmcAddr. ipmbAddr is the string lc if it is a local mmcAddr.

Example

```
hpmcmd -c fruinv 0
```

5.5.2.10 fruread

Description

This command gets a range of data from the specified FRU.

Synopsis

```
fruread <fruid> <startAddress> <nBytes>
```

Parameters

`fruid`

Is 0 for the main module.

`startAddress`

Is the starting address in decimal.

`nbytes`

Number of bytes to read in decimal; cannot exceed 16 because of IPMI message size limitations.

Example

```
hpmcmd -c fruread 0 0 8
```

5.5.2.11 get-shut-tout

Description

Gets the payload shutdowqn timeout in milliseconds.

Synopsis

```
get-shut-tout
```

5.5.2.12 help

Description

This command lists the available commands from the hpmcmd program with a brief explanation about the command.

Synopsis

```
help
```

5.5.2.13 ipmbaddress

Description

This command retreives the module IPMB address.

Synopsis

```
ipmbaddress
```

5.5.2.14 ipmcdevice

Description

This command retrieves the payload tty device.

Synopsis

```
ipmcdevice
```

5.5.2.15 ledget

Description

This command gets information about a specified LED controlled by the IPMC.

Synopsis

```
ledget <fruid> <led> [-t ipmbAddr[:mmcAddr]]
```

Parameters

`fruid`

Is 0 for the main module.

`led`

Is BLUE for the hot swap LED or LEDN for FRU LED<n>. <n> is a number between 1 and the maximum FRU LEDs supported by the module.

`-t`

Sends the command to ipmbAddr:mmcAddr. `ipmbAddr` is the string lc if it is a local mmcAddr.

Example

```
hpmcmd -c ledget 0 led1
```

5.5.2.16 ledprop

Description

This command displays the FRU LED properties under IPMC control.

Synopsis

```
ledprop <fruid>
```

Parameters

`fruid`

0 for the module.

Example

```
hpmcmd -c ledprop 0
FRU LEDs under IPMC control:
LED0 = BLUE
LED1 = RED or AMBER
LED2 = GREEN
```

5.5.2.17 ledset

Description

This command controls the override state of a specific FRU LED. The RTM FRU LEDs reflect the state of the main blade (FRU 0) LEDs. Therefore, overriding the state to something different than the main FRU LED state will not have any effect.

The blue LED is the only one that can be controlled separately.

Synopsis

```
ledset <fruid> <led> <operation> [offms] [onms] [color] [-t
ipmbAddr]
```

Parameters

`fruid`

Is 0 for the main module.

`led`

Is BLUE for the hot swap LED or LEDN for FRU LED<n>. <n> is a number between 1 and the maximum FRU LEDs supported by the module.

`operation`

ON = enable override state and turn LED on.
OFF = enable override state and turn LED off.
BLINK = enable override state and blink LED; off_duration and on_duration specify the blink duration; the default on and off duration is 300 ms.
LOCAL = cancel override state and restore LED control to the IPMC, that is, local state.
TEST = run lamp test for specified on_duration, then restore prior state.

`offms`

10–2500 in 10-millisecond increments; only valid if operation is BLINK

onms

Only valid if operation is BLINK or TEST:
If operation is BLINK, 10–2500 in 10-millisecond increments
If operation is TEST, 100-12800 in 100-millisecond increments

`color`

LED0 = BLUE
LED1 = RED
LED2 = GREEN
LED3 = AMBER

`-t ipmbAddr`

Sends the command to ipmbAddr.

Example

```
hpmcmd -c ledset 0 led1 on
```

5.5.2.18 loglevelget

Description

This command retrieves the current `hpmagentd` log level. See loglevelset for more detail.

Synopsis

```
loglevelget
```

Example

```
hpmcmd -c loglevelget
Loglevel 5 (NOTICE)
```

5.5.2.19 loglevelset

Description

This command sets the level of message logging for hpmagentd.

Synopsis

```
loglevelset <newLogLevel>
```

Parameters

`newLogLevel`

Is one of the standard syslog levels:

Level	Description
0	Emergency
1	Alert
2	Critical
3	Error
4	Warning
5	Notice
6	Information
7	Debug

Example

```
hpmcmd -c loglevelset 7
```

5.5.2.20 macaddress

Description

This command retrieves a list of available MAC addresses.

Synopsis

```
macaddress [-t ipmbAddr]
```

Parameters

`-t ipmbAddr`

Sends the command to ipmbAddr.

5.5.2.21 partnumber

Description

This command retrieves the part number of the main module.

Synopsis

```
partnumber [-t ipmbAddr]
```

Parameters

`-t ipmbAddr`

Sends the command to ipmbAddr.

Example

```
hpmcmd -c partnumber
```

5.5.2.22 physlotnumber

Description

This command retrieves the physical slot number in which the blade is plugged in. It is only applicable to AdvancedTCA system environments.

Synopsis

```
physlotnumber
```

5.5.2.23 portget

Description

This command shows the current state of interfaces governed by e-keying. If no channel is specified, portget returns data for all channels in the specified interface. If neither interface nor channel are specified, portget will return data for all interfaces.

Synopsis

```
portget [interface] [channel]
```

Parameters

`interface`

The only valid value for PrAMC-7311 is AMC.

`channel`

Is an integer in the following range:
0 and 1 for Ethernet links for Base
2 and 3 for SATA storage links
4 and 5 for PCI-Express links

Example

```
hpmcmd -c portget AMC 0
```

5.5.2.24 portset

Description

This command enables and disables ports in a channel. The following table lists the valid values for each parameter.

Synopsis

```
portset <intf> <chan> <grpid> <type> <ports> <oper>
```

Parameters

`intf`

Valid value for the PrAMC-7311 is: AMC

`chan`

Is an integer in the following range:
0 and 1 for Ethernet links for Base
2 and 3 for SATA storage links
4 and 5 for PCI-Express links

`grpid`

0 for Ethernet links and SATA storage links
0 or 2 for PCI-Express links

`type`

Valid values are:

Valid Value	Description
ETHER	for fabric interface
STORAGE	for storage links

`ports`

0 for Ethernet links
0123 for PCI-Express links

oper

Valid values are DISABLE or ENABLE.

Example

```
hpmcmd -c portset base 1 0 base 0 0 enable
```

5.5.2.25 posttypeget

Description

Gets the POST type to run at boot

Synopsis

```
posttypeget <payload_cpu_selector>
```

Parameters

payload_cpu_selector

The particular CPU specified is set to posttype to run.

5.5.2.26 posttypeset

Description

Sets the POST type to run at boot

Synopsis

```
posttypeset <payload_cpu_selector> <newPostType>
```

Parameters

payload_cpu_selector

Integer between 0 and number of CPU devices supported per board.

newPostType

POST type. Supported values are: SHORT and LONG

5.5.2.27 quit

Description

This command is for exiting the hpmcmd program when running in interactive mode.

Synopsis

```
quit
```

5.5.2.28 rebootpath

Description

This command retrieves the path and filename of the current hpmagentd rebbot script.

Synopsis

```
rebootpath
```

Example

```
hpmcmd -c rebootpath
/opt/bladeservices/bin/hpmreboot
```

5.5.2.29 sdr

Description

This command shows the SDR records.

Synopsis

```
sdr
```

Example

```
hpmcmd -c sdr
recID 0: management controller device locator record
    I2C slave addr:  42
    Channel number:  00
    Power state:     06
    Global init:     0C
    Capabilities:    2D
    Entity Id:       PICMG front board
    Entity instance: 60
    OEM:             00
    Id string:       ATCA-7301
```

```
recID 1: full sensor record
       owner is IPMB 84 sensor num 00 on lun 00 channel 00
       logical entity: PICMG front board - instance 60
       Hot Swap Carrier : FRU hot swap : sensor-specific discrete

recID 2: full sensor record
       owner is IPMB 84 sensor num 01 on lun 00 channel 00
       logical entity: AMC - instance 65
       Hotswap_AMC1 : FRU hot swap : sensor-specific discrete
```

5.5.2.30 sdr_dump

Description

This command shows the SDR records in binary and hex format.

Synopsis

```
sdr_dump
```

Example

```
hpmcmd -c sdr_dump
   SDR Records:
   01 00 51 01 39 20 00 10 14 61 7f 69 02 01 04 22 "..Q.9 ...a.i..."""
   04 22 12 12 00 04 00 00 33 00 00 00 00 c0 07 cd "."......3......Í"
   d0 ca ff 00 00 d8 00 00 c2 00 01 01 00 00 00 ce "..............Î"
   53 42 43 20 2b 31 2e 30 35 56 20 56 74 74 "SBC +1.05V Vtt"
   .
   .
   .
   61 67 65 20 45 "age E"
```

5.5.2.31 sdrinfo

Description

This command shows the SDR information.

Synopsis

```
sdrinfo
```

Example

```
hpmcmd -c sdrinfo
```

```
SDR Information:
LUN 0 has 066 sensors; static sensor population
LUN 1 has 000 sensors; static sensor population
LUN 2 has 000 sensors; static sensor population
LUN 3 has 000 sensors; static sensor population
```

5.5.2.32 sendamc

Description

This command allows sending any of the commands supported in the IPMI spec to a remote AMC or MMC of a remote IPMC IPMB-L.

Synopsis

```
sendamc <IPMBaddress> <MMCaddress> <netfn> <cmd> <data0> <datan>
```

Parameters

`IPMBaddress`

Destination IPMB address in hex digits.

`MMCaddress`

Destination MMC address in hex digits.

`netfn`

IPMI request net function in hex digits.

`cmd`

IPMI request command in hex digits.

`data0-datan`

IPMI request data bytes. if any, in hex digits.

5.5.2.33 sendcmd

Description

This command allows a user to send any of the commands supported in the IPMI spec to a remote IPMC.

Synopsis

```
sendcmd <IPMBaddress> <netfn> <cmd> <data0> ... <dataN>
```

Parameters

`IPMBaddress`

Destination IPMB address in hex digits.

`netfn`

IPMI request net function in hex digits.

`cmd`

IPMI request command in hex digits

`data0 ... dataN`

IPMI request data bytes. if any, in hex digits.

Example

```
hpmcmd -c sendcmd 90 06 59
07 59 C1
```

5.5.2.34 set-shut-tout

Description

Sets the payload shutdown timeout in milliseconds.

Synopsis

```
set-shut-tout <timeout>
```

Parameters

`timeout`

Timeout value expressed in milliseconds.

5.5.2.35 shelfaddress

Description

This command retrieves the shelf address string from the shelf FRU.

Synopsis

```
shelfaddress
```

Example

```
hpmcmd -c shelfaddress
01
```

5.5.2.36 shutdownpath

Description

This command retrieves the path and filename of the current hpmagentd shutdown script.

Synopsis

```
shutdownpath
```

Example

```
hpmcmd -c shutdownpath
/opt/bladeservices/bin/hpmshutdown
```

5.5.2.37 slotnumber

Description

This command retrieves the logical slot number of the slot where the module is plugged in.

Synopsis

```
slotnumber
```

Example

```
hpmcmd -c slotnumber
ATCA Carrier Slot: 6, Bay ID: 8
```

5.5.2.38 version

Description

This command retrieves the version of the hpmcmd software and sends a request to get the version of the hpmagent daemon that is running. Once the information is gathered, it is printed.

Synopsis

```
version
```

Example

```
hpmcmd -c version
hpmcmd version bbs 3.1.0 build X
hpmagentd version bbs 3.1.0 build X
```

5.5.2.39 watchdog

Description

This command is used handle the payload BMC watchdog.

Synopsis

```
watchdog set <tmr_use> <tmr_action> <pre_timeout> <flags>
<lsb_val> <msb_val>
watchdog set default
watchdog get
watchdog start
watchdog stop
watchdog reset
```

Parameters

`set`

Possible values are

tmr_use	dont_stop stop
tmr_action	no_action hard_reset power_cycle power_down
pre_timeout	0-255
flags	clear dont_clear
lsb_val	0-255
msb_val	0-255

Related Documentation

A.1 SMART Embedded Computing Documentation

The documentation listed is referenced in this manual. Technical documentation can be found by using the Documentation Search at *https://www.smartembedded.com/ec/support/* or you can obtain electronic copies of SMART EC documentation by contacting your local sales representative.

Table A-1 SMART EC Embedded Computing Publications

Document Title	Publication Number
PrAMC-7311 Installation and Use	6806800P34
ATCA-F125 Installation and Use	6806800J94
ATCA-F120 Installation and Use	6806800D06
ATCA-F140 Installation and Use	6806800M67

A.2 Related Specifications

For additional information, refer to the following table for related specifications. As an additional help, a source for the listed document is provided. Please note that, while these sources have been verified, the information is subject to change without notice.

Table A-2 Related Specifications

Document Title	Source
IPMI Specifications *http://www.intel.com/design/servers/ipmi*	
IPMI Spec v1.5, Document Revision 1.1, February 20, 2002	Intel Corporation, Hewlett-Packard, DEC, NEC
IPMI v1.5 Addenda, Errata, and Clarifications, Addendum Document Revision 5, January 29, 2004	Intel Corporation, Hewlett-Packard, DEC, NEC
Intelligent Platform Management Interface Specification v1.0, Document Revision 1.1, November 15 1999	Intel Corporation, Hewlett-Packard, NEC, Dell
IPMI Implementer's Guide, Draft Version 0.7, September 16, 1998	Intel Corporation
IPMI Platform Management FRU Information Storage Definition V1.0, September 27, 1999	Intel Corporation

Table A-2 Related Specifications (continued)

Document Title	Source
PCI Industrial Computer Manufacturers Group (PICMG) Specifications *http://www.picmg.org*	
PICMG 3.0 Revision 1.0 Advanced Telecommunications Computing Architecture (AdvancedTCA) Base Specification, December 2002	PICMG
PICMG 3.1 Revision 1.0 Specification Ethernet/Fibre Channel for AdvancedTCA Systems, January 2003	PICMG
Service Availability Forum Specifications *http://www.saforum.org*	
SAI-HPI-B.01.01 Hardware Platform Interface Specification	SA Forum
SAI-AIS-A.01.01 Application Interface Specification	SA Forum
SAI-HPI-SNMP-B.01.01	SA Forum
SAIM-HPI-B.01.01-ATCA SAF HPI-to-AdvancedTCA Mapping Specification	SA Forum

A.3 Additional Resources

The following table lists additional resources which may be useful in working with SMART EC's AdvancedTCA systems.

Table A-3 Additional Resources

Resource	Source
OpenHPI open source software project *http://openhpi.org*	
OpenHPI 1.0 Manual	OpenHPI
OpenHPI NetSNMP Subagent Development Manual	OpenHPI
Net-SNMP *http://net-snmp.sourceforge.net/*	
Pigeon Point Systems *http://www.pigeonpoint.com*	
IPM Sentry Shelf-External Interface Reference	Pigeon Point Systems

Table A-3 Additional Resources (continued)

Resource	Source
IPM Sentry Shelf Manager User Guide	Pigeon Point Systems
OpenIPMI *http://openipmi.sourceforge.net/*	
Initramfs *http://en.gentoo-wiki.com/wiki/Initramfs*