Basic Blade Services Software on ATCA-7370

Programmer's Reference

P/N: 6806800P57E

October 2019

SMART™ Embedded Computing

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 The text itself may not be published commercially in print or electronic form, edited, translated, or otherwise altered without the permission of SMART EC.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

Table of Contents

List of Figures

List of Figures

List of Tables

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 gives an overview of ATCA-7370.

Chapter 2, Installing the Basic Blade Services Software on page 19 describes the procedure to install the BBS.

Chapter 3, Linux Distribution Description on page 31 describes the Linux distribution.

Chapter 4, Firmware Upgrade Facility on page 37 describes the firmware upgrade facility.

Chapter 5, Network Management Utility on page 49 describes the network management utility.

Chapter 6, Hardware Platform Management on page 51 describes the hardware platform management.

Chapter 7, Board Control Module on page 85 describes the board control module.

Chapter 8, Kernel and Root File System Config using WRL 4.3 on page 87 describes the Kernel and root file system.

Chapter 9, Using Development Kit BLSV API on page 89 describes how to use BLSV API Development kit.

Chapter A, Related Documentation on page 91 lists the related documents of ATCA-7370.

Abbreviations

This document uses the following abbreviations:

Abbreviation	Definition
API	Application Programming Interface
AdvancedTCA	Advanced Telecommunications Computing Architecture
ATCA	Advanced Telecommunications Computing Architecture
BBS	Basic Blade Services
BIOS	Basic Input Output System
BLSV	Blade Services
BSP	Board Support Package

Abbreviation	Definition
CGL	Carrier Grade Linux
DHCP	Dynamic Host Configuration Protocol
FCU	FUF Command Line Utility
FMS	Fault Management System
FPGA	Field Programmable Gate Array
FRI	Firmware Recovery Image
FRU	Field Replaceable Unit
FUF	Firmware Upgrade Facility
GPIO	General Purpose Input/Output
HPI	Hardware Platform Interface
HPM	Hardware Platform Management
IPMB	Intelligent Platform Management Bus
IPMC	Intelligent Platform Management Controller
IPMI	Intelligent Platform Management Interface
LUN	Logic Unit Number
MAC	Media Access Control
NMU	Network Management Utils
OEM	Original Equipment Manufacturer
OSDL	Open Source Development Labs
PCI	Peripheral Component Interconnect
PCIx	PCI Express
PICMG	PCI Industrial Computers Manufacturers Group
PXE	Preboot Execution Environment
RPM	RedHat Package Manager
RTM	Rear Transition Module
SAS	Serial Attached SCSI
SATA	Serial ATA

Abbreviation	Definition
SCSI	Small Computer System Interface
SDR	Sensor Data Record
SMI	Serial Management Interface
SNMP	Simple Network Management Protocol
SSD	Solid State Disk
SSH	Secure Shell
TAR	Tape Archive
TFTP	Trivial File Transfer Protocol

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

Notation	Description
..	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
	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

See the table below for manual revisions and changes.

Part Number	Date	Description
6806800P57E	October 2019	Rebrand to SMART Embedded Computing
6806800P57D	May 2016	Added *Updating Software* on page 31, *Linux Services Initialization*, *RC Scripts* on page 36, *Modifying the Configuration of the Artesyn-Supplied CGL Kernel* on page 33, *Building Kernel and Root File System* on page 109, *Board Control Module* on page 105, and *Using Development Kit BLSV API* on page 111.
6806800P57C	June 2014	Re-branded to Artesyn.
6806800P57B	February 2014	Updated Figure BBS Architecture on page 17, updated Software Building Blocks on page 17, updated Package Information, Configuring TFTP, DHCP and PXE, Installation Procedures, updated Chapter 3, Linux Distribution Description, Chapter 4, Firmware Upgrade Facility, Chapter 6, Hardware Platform Management, Chapter 7, Board Control Module, and Appendix A, Related Documentation. Updated Figure Software Levels of the HPM Architecture on page 58, Chapter 5, Network Management Utility, Chapter 8, Kernel and Root File System Config using WRL 4.3, and created a new chapter Chapter 9, Using Development Kit BLSV API.
6806800P57A	July 2012	Initial version

Introduction

1.1 Overview

The Basic Blade Services (BBS) software provides a set of services that support the blade on which the software is installed. BBS includes:

Board Support Package (BSP) to build Wind River® Linux 4.3 for the ATCA-7370 blade.

Several custom hardware management functions for the unique hardware of the blade and firmware upgrading facilities.

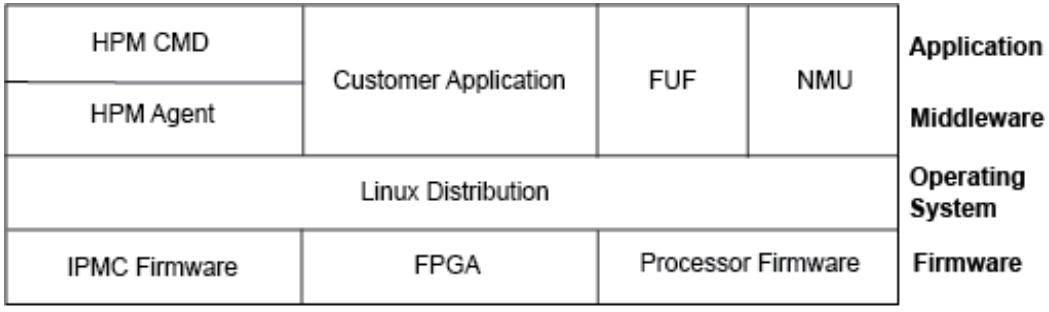
A set of management routines for Linux and all hardware interfaces. Management access includes support of a local console interface based on a standard Linux command shell.

1.2 Software Building Blocks

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

The following figure depicts the BBS software architecture.

Figure 1-1 BBS Architecture



HPM: Hardware Platform Management
FUF: Firmware Upgrade Facility
FPGA: Field Programmable Gate Array
IPMC: Intelligent Peripheral Management Controller

BBS for the ATCA-7370 consists of the following main software and services:

- Firmware Upgrade Facility
  The Firmware Upgrade Facility (FUF) provides a uniform way to upgrade firmware on SMART EC blades, regardless on which flash locations the firmware is stored. FUF upgrades the BIOS firmware as well as the Intelligent Platform Management Controller (IPMC) firmware. The FUF currently consists of a Firmware Upgrade Command Line Utility (FCU), flash device drivers, and specially prepared firmware recovery image files. The FUF can be used on switch and node blades.
- Network Management Utility (NMU)
  The NMU tool is used to query Small Form-factor Pluggable (SFP) information that are connected to the board. To use this functionality, the nmucmd tool must be installed and the ixgbe_sfp module must be loaded.
- Linux Operating System
  Wind River Enterprise Linux 4.3 (Carrier Grade Linux) is the operating system for BBS blades and modules. The operating system comes with kernel 2.6.34.15. The BBS installation scripts will activate various Linux services (above the kernel).
- Hardware Platform Management
  Hardware Platform Management (HPM) in AdvancedTCA systems is based on Intelligent Platform Management Interface (IPMI) specification. IPMI commands can be complex and cumbersome. Using a certain set of commands, HPM facilitates the blade or module-level hardware management.

Installing the Basic Blade Services Software

2.1 Overview

SMART Embedded Computing provides software images, including software updates, to its licensed customers. In order to obtain the latest BBS software versions, contact your local sales representative.

Following are the three methods of installing/booting the BBS software on ATCA-7370:

- Booting diskless client through network
- Installing and booting from SATA/SAS hard disk
- Installing and booting from USB stick

On ATCA-7370 the SAS-HDD and SATA-HDD can reside in the RTM when an ARTM-ATCA-736X blade is installed. As an option a Solid State Disk (SSD) with SATA interface (SATA cube) can be placed on the front board.

For all these options you need to set up an external TFTP server to retrieve the required BBS files. Furthermore you need to do some initial configurations. The following table provides the main set up and configuration steps of the three installation/boot options. The detailed procedures can be found in the following sections.

Table 2-1 BBS Installation/Boot Options - Main Set-Up and Configuration Steps

Installation/Boot Option	Main Set-Up and Configuration Steps
Booting Diskless Client. Refer to section *2.3.1 on page 25*.	1. Set up and configure external TFTP boot server 2. Configure DHCP server 3. Configure PXE boot options 4. Configure ATCA-7370 BIOS to boot from network
Installing and booting from SATA/SAS hard disk. The hard disk can be located on the RTM or locally (as SATA cube) on the front board. Refer to section *2.3.2 on page 26*.	1. Set up and configure external TFTP boot server 2. Configure DHCP server 3. Configure PXE boot options 4. Configure ATCA-7370 BIOS for network boot 5. Boot BBS initrd image 6. Install BBS image on hard disk (via install script) 7. Configure ATCA-7370 BIOS to boot from hard disk

Table 2-1 BBS Installation/Boot Options - Main Set-Up and Configuration Steps

Installation/Boot Option	Main Set-Up and Configuration Steps
Installing and booting from USB stick.	1. Set up and configure external TFTP boot server 2. Configure DHCP server 3. Configure PXE boot options 4. Configure ATCA-7370 BIOS for network boot 5. Boot BBS initrd image 6. Install BBS image on USB stick (via install script) 7. Configure ATCA-7370 BIOS to boot from USB stick

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

2.1.1 Installation Scripts

The following table describes the installation scripts required for installing the operating system and blade utilities for ATCA-7370. These installation scripts require a TFTP and a DHCP server to download the installation files.

Table 2-2 Installation Scripts

	flashrfsrc
Installed packages	Kernel, RFS and additional packages
Features	Simple partition layout and autoconfig

2.1.2 Package Information

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

The BBS distribution contains the following packages:

Table 2-3 BBS Distribution Packages

Description	File Name
Linux Kernel	`kernel`
Ramdisk image for netboot	`ramdisk.image.gz`
Hard Disk Installation	

Table 2-3 BBS Distribution Packages (continued)

Description	File Name
Root file system including BBS packages for hard disk installation	`atca7370-cgl-glibc_cgl-dist.tar.bz2`
Check sum of files and for the hard-disk installation	`files.sha1sum`
Boot command line for pxelinux	`default.bbs-atca7370`
Initrd needed to boot from harddisk or flash	`initrd0.img`
Kernel symbol table	`System.map`

The following table describes hardware components:

Table 2-4 Hardware Content

Item	Model	Description
Blade	ATCA-7370-0GB	ATCA BLADE, DUAL INTEL XEON E5-2648L 8-CORE (1.8 GHZ), 0GB, 10GB SUPPORT (ROHS 6/6)
Blade	ATCA-7370-0GB-S	ATCA BLADE, SINGLE INTEL XEON E5-2648L 8-CORE (1.8 GHZ), 0GB, 10GB SUPPORT (ROHS 6/6)
RTM	RTM-ATCA-7360	RTM FOR THE ATCA-736x PRODUCT SERIES, 6XGBE, 2XSAS, 1XSLOT FOR OPTIONAL HDD (ROHS 6/6)
RTM	RTM-ATCA-7360-L	RTM FOR THE ATCA-736x PRODUCT SERIES, 2XGBE, 2XSAS, 1XSLOT FOR OPTIONAL HDD (ROHS 6/6)
RTM	RTM-ATCA-736X-10G	RTM FOR THE ATCA-736x PRODUCT SERIES

2.1.3 Root file system BBS packages

The root file system contains the following BBS packages:

- `hpmcmd`
- `hpmagent`
- `nmucmd`
- `fcu`
- `IPMC firmware for the Single CPU board`
- `IPMC firmware for Dual CPU board`

- BIOS and FPGA firmware
- netdevrenamer

You can find the packages in the ATCA7370-custom-layer.tgz archive within the wrlinux4/RPMS directory.

2.1.4 Accessing the ATCA-7370 via Serial Console

To access the ATCA-7370 via serial console, the default serial port settings are as follows:

- 9600 baud
- No parity
- Eight data bits
- One stop bit
- Flow control: xon/xoff
- Emulated terminal type: VT100

If you wish to access Linux via a Linux shell, the default account login is **root** with the password **root**. For more information, refer *Login on page 31*.

2.2 Configuring TFTP, DHCP and PXE

For BBS installation and boot options, you need to 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. All related steps are described in the following section.

2.2.1 Configuring DHCP

The DHCP configuration file on an TFTP server (for example ATCA-F120 or an external TFTP server) resides in /etc/dhcpd.conf. Make sure this file contains the following entries (IP addresses may be different in your configuration):

```
#
# Sample dhcpd configuration file
#
#

allow bootp;
allow booting;
authoritative;
filename "pxelinux.0";
```

```
ddns-update-style ad-hoc;

option domain-name "booting.com";

option subnet-mask 255.255.255.0;
default-lease-time 600;
max-lease-time 7200;

#Base 1 interfaces
subnet 192.168.21.0 netmask 255.255.255.0 {
   range 192.168.21.100 192.168.21.125;
   option broadcast-address 192.168.21.255;
}


#Base 2 interfaces
subnet 192.168.22.0 netmask 255.255.255.0 {
   range 192.168.22.100 192.168.22.125;
   option broadcast-address 192.168.22.255;
}
```

Restart DHCP service on your Linux DHCP server by issuing the following commands and make sure your DHCP service starts successfully against your configuration files:

#/etc/init.d/dhcp stop

#/etc/init.d/dhcp start

2.2.2 Configuring PXE

PXE determines which kernel and root file system image a blade gets from the server. The PXE environment as well as the bootable images usually reside in the `/tftpboot` directory on the server. The initial boot file is called `pxelinux.0` and the PXE configuration directory is in the `/tftpboot/pxelinux.cfg`. The default configuration file is called `/tftpboot/pxelinux.cfg/default`.

Example default file:

```
DEFAULT ATCA7370/kernel ramdisk_size=2100000 console=ttyS0,115200n8
initrd=ATCA7370/ramdisk.image.gz root=/dev/ram0 ip=none ro
pci=lastbus=255 quiet nopat
```

In this configuration, the same images are served to all blades in the chassis. In order to distinguish between blades and to serve different images, you can use different default files and link them to different MAC addresses of different blades.

Depending on the particular BBS release, an example default file for the ATCA-7370 may be contained in the BBS package (check the release notes applicable to your blade release). This file contains all required kernel parameters. In order to use the default file, you need to link it to the MAC address of the ATCA-7370 as described below.

Example:
The following example shows how to set up the PXE environment for an ATCA-7370 blade. This is done by creating a new default file and linking it to the MAC address of the ATCA-7370 boot Ethernet interface, which is `00:80:42:1d:da:07` in the example.

PXE expects that the file name should be prefixed with "01" and all the characters in the file name are lower case letters.

Setting up the PXE Environment

Proceed as follows:

1. Make a new subdirectory in `/tftpboot`
   `#mkdir -p /tftpboot/ATCA7370`
2. Copy the corresponding boot image and RPMs to this directory.
3. Set up a new default files in `/tftpboot/pxelinux.cfg`, for example default.atca7370.
   The contents of default.atca7370 are:
   ```
   DEFAULT ATCA7370/kernel ramdisk_size=2100000 console=ttyS0,115200n8
   initrd=ATCA7370/ramdisk.image.gz root=/dev/ram0 ip=none ro
   pci=lastbus=255 quiet nopat
   ```
4. Link the MAC address of the blade to its boot default file, for example:
   `#cd /tftpboot/pxelinux.cfg`
   `#ln -s default.atca7370 01-00-80-42-1d-da-07`

2.3 Installation Procedures

The following subsections list the different BBS installation procedures.

2.3.1 Configuring ATCA-7370 for Diskless Client Boot of the BBS Software

This section describes the steps you need to take for performing diskless client boot of the BBS software.

Configuring BIOS for Diskless Client Boot

1. Connect to the blade via the serial interface.
2. Power up or reboot the blade.
3. Quickly hold down the **<F2>** key on your keyboard until the BIOS menu appears.
4. Select **ADVANCED** on the top menu.
5. Scroll down to **BOOT FEATURES** by using the arrow keys.
6. Press **<ENTER>** key.
7. Make sure that the following settings are enabled:
   `Base-Interface Network boot` or `Front Panel Network Boot` (depending on the interface you want to boot from).
   If any of these settings is disabled, enable the setting(s) and press **<F10>** key or select **Exit Saving Changes**. This will save the new settings and restart the BIOS. After the restart, press **<F2>** key to enter BIOS again and continue with the BIOS configuration.
8. Depending on which interface you want to boot from, put **Base Network 1, Base Network 2**, or **FrontPanel Network** to the first position of the **Boot priority** order list.
9. Save and exit.

Rebooting the Blade

1. Reboot the blade via:
   - Shelf manager
   - Opening and closing the lower handle switch on the face plate
   - Pressing the reset button on the face plate
2. Observe that the blade is getting a DHCP address and is loading the kernel and ramdisk image:
   ```
   Try to load: pxelinux.cfg/<address>
   boot:
   Loading <blade/module>/kernel.............
   Loading <blade/module>/ramdisk-image.gz....
   ```
3. Once the blade has fully come up, log on to the serial console as **root** with the default password **root**.

2.3.2 Installing BBS Software on Disk Drive

This section describes how to install and boot the BBS software from hard disk or USB drive. The BBS software can be installed on the following hard disk types:

- SAS/SATA hard disk drive installed on RTM or
- SATA cube on front board (optional)

The installation process starts with the booting of an initial ramdisk via network. The initial ramdisk is then used to copy (via TFTP) and interactively install the kernel, the root file system, and other BBS software on the disk.

The following procedures describe these steps in detail.

Configuring BIOS for Diskless Network Boot

1. Connect to the blade via the serial interface.
2. Power up or reboot the blade.
3. Quickly hold down the **<F2>** key on your keyboard until the BIOS menu appears.
4. Select **ADVANCED** on the top menu.
5. Scroll down to **BOOT FEATURES** by using the arrow keys.
6. Press **<ENTER>** key.
7. Make sure that the following settings are enabled:
   `Base-Interface Network boot` or `Front Panel Network Boot` (depending on the interface you want to boot from).
   If any of these settings is disabled, enable the setting(s) and press **<F10>** key or select **`Exit Saving Changes.`** This will save the new settings and restart the BIOS. After the restart, press **<F2>** key to enter BIOS again and continue with the BIOS configuration.
8. Depending on which interface you want to boot from, put `Base Network 1, Base Network 2,` or **`FrontPanel Network`** to the first position of the **`Boot priority`** order list.
9. Save and exit.

Disk Installation

Use `/opt/bladservices/tools/flashrfsrc` script to install the disk.

To install the disk, copy the `atca7370-cgl-glibc_cgl-dist.tar.bz2, initrd0.img, kernel, system.map, files.shalsum` files from Release/RFS directory to external tftp server. Identify the `flashrfsrc` script and follow the instructions. After the installation, you have to configure the BIOS Boot Order to select your hard disk or USB device.

2.4 Updating Software

Software updates are usually delivered as rpm-files. To install the files on the disk of the blade, copy the new rpm file to the blade, stop the application using this rpm, remove the original files (using the `rpm -e` <package> command) and install the newly copied rpm (using the command `rpm -Uvh` <package-name>).

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 NetBoot root file system
- Modifying the hard disk installation
- Modifying the hard disk installation procedure
- Modifying the Configuration of the SMART EC-Supplied CGL Kernel

2.5.1 Modifying the NetBoot Root File System

The netBoot root file system is stored in the file `ramdisk.image.gz` on the TFTP server. In order to change the system's behavior regarding network booting blades, you have to modify the root file system.

As root:

```
# cd /tftpboot/<blade or module to be modified>
# mkdir mnt
# gunzip ramdisk.image.gz
# mount -o loop ramdisk.image mnt
# pushd mnt
# ..... /* make all modifications and enhancements: delete, add or change
files*/
# popd
# unmount mnt
# gzip -9 ramdisk.image
```

The blade will now boot the modified root file system.

2.5.2 Modifying Hard Disk Installation

The hard disk installation can be changed after the blade has been installed or by modifying the file `atca7370-cgl-glibc_cgl-dist.tar.bz2` prior to the installation. After modifying this file, you have to compute and add the sha1 checksum of the modified root file system to the files.sha1sum in the installation directory on the TFTP server.

The example below shows how to change the default login behavior.

```
# cd /tftpboot/...   (cd to the directory containing the atca7370-cgl-
glibc_cgl-dist.tar.bz2 you want to modify)
# mkdir rootfs
# cd rootfs
# tar xzf ../atca7370-cgl-glibc_cgl-dist.tar.bz2
* Make your modifications and enhancements to the root filesystem in the
current directory
# tar czf ../atca7370-cgl-glibc_cgl-dist.tar.bz2 .
# cd ..
# sha1sum atca7370-cgl-glibc_cgl-dist.tar.bz2
*Correct the sha1sum for atca7370-cgl-glibc_cgl-dist.tar.bz2  in
files.sha1.sum
```

2.5.3 Modifying the Hard Disk Installation Procedure

The hard disk installation procedure is based on the `files.sha1sum` in the installation directory on the TFTP server. All packages which are copied to the blade during installation are listed in the `files.sha1sum` together with their sha1sum. The standard installation process accepts rpm, tar, and tgz files and all files that have "kernel" in the file name.

The packages from `files.sha1sum` are installed in the same sequence as listed in the file `files.sha1sum`. The installation process re-calculates the sha1sum of the packages on the blade and compares it to the sha1sum determined by `files.sha1sum`. This ensures a protection against errors and faults during file transmission. You will be notified in case of mismatch. In that case, you have to repeat the installation procedure.

The root file system must precede the rpm files in the files.sha1sum file.

2.5.4 Modifying the Configuration of the SMART EC-supplied CGL Kernel

The current kernel configuration of a running ATCA-7370 installation can be retrieved using the Linux command `zcat /proc/config.gz` or from `atca7370/wrlinux-4/templates/board/atca7370/linux/atca7370-cgl.cfg`

To modify the configuration of the CGL kernel supplied by SMART Embedded Computing, consult your local SMART EC sales representative for assistance and further information.

Linux Distribution Description

3.1 Distribution Description

The BBS for the ATCA-7370 is based on latest Wind River Enterprise Linux 4.3, which is a Linux distribution built on Linux 2.6.34.15 kernel technology.

User should always upgrade to latest Wind River Enterprise Linux 4.3.

3.2 Reliability

The hard disk installation is configured to use the journaling file system `ext3`. In this distribution majority of errors that are caused due to improper shutdown are fixed automatically during the boot process. Catastrophic errors that cannot be fixed automatically will yield to a command prompt, allowing the super user to execute the `fsck` command on the affected partition.

3.3 Login

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

If you use a serial console or terminal emulator, the serial/RTM port settings are 115200 baud, no parity, 8 data bits, and 1 stop bit.

If you use secure shell server, it starts in run levels 2–5 and listens on all the Ethernet interfaces. If you use Secure Shell (SSH), refer to *Network Services Configuration on page 33* for default IP address assignments.

The following table lists available default login accounts.

Login Name	Password	Description
root	root	Privileged user account

3.4 Linux Services Initialization

The following table describes the generic Linux run levels. *RC Scripts on page 32* describes the services configured to start in the various Linux run levels. By default, the blade first runs in run-level 'S' and then boots into run-level '3' as configured by the factory.

Table 3-1 Generic Linux Run Levels

Run-level	Description
S	Startup
0	Halt System
1	Single user mode
2	Multi user mode
3	Multi user mode with network (default)
4	Not used
5	Not used
6	Reboot system

3.5 RC Scripts

In addition to the rc-scripts of the Wind River Linux 4.3 Linux configuration, the following start/stop scripts are added to ATCA-7370.

Table 3-2 RC Scripts

Run-level	Script Name	Description
rc0.d	`K01watchdog`	stops the software watchdog
	`K05hpm`	stops the hpmagent
rc2.d	`S03boardctrl`	inserts boardctrl.ko module
	`S04hpm`	starts the hpmagent
	`S05sethost`	sets hostname
rc3.d	`S03boardctrl`	inserts boardctrl.ko module
	`S04hpm`	starts the hpmagent

Table 3-2 RC Scripts (continued)

Run-level	Script Name	Description
	S05sethost	sets hostname
	S06netdev-rename	starts the netdevrenamer tool
	S10network	Activates/Deactivates all network interfaces configured to start at boot time
	S91watchdog	starts software watchdog
	S92qat	start Intel QAT
rc6.d	K01watchdog	stops the software watchdog
	K05hpm	stops the hpmagent

3.6 Network Services Configuration

The following sections describe the default configuration for network services.

3.6.1 Network Device Renaming

The BBS tool netdevrenamer which renames the ethernet interfaces from eth1, 2, 3… to the corresponding name of the interface location like base1, fabric2, front1 or rtm2. The tool consists of the program netdevrenamer and a configuration file `/opt/bladeservices/etc/netdev-config`. The config file follows the well-known XML syntax and may be easily changed for your purposes. Each interface has a ifcfg element which hold the lines for the `ifcfg-<interface>` which is generated by the netdevrenamer. In this XML element section you can add, change and delete lines to setup the interface as you need it. The following lines show the base1 interface as an example:

```
<device name="base1">
<location pci="0000:08:00.0" rtm-name="artm-736x-10G,artm-736x-10G-SP"/>
<location pci="0000:03:00.0" />
<ifcfg path="/etc/sysconfig/network-scripts">
BOOTPROTO=dhcp
HWADDR=[address]
ONBOOT=yes
</ifcfg>
</device>
```

Note that you must not change or delete the line `HWADDR=[address]`. Otherwise, the renaming does not work.

The following table specifies the Ethernet devices supported by ATCA-7370.

Device Name	Description	Speed	Location	IP address
base1	Base Interface 1	1GbE	Base blade -> Backplane	Obtained by the DHCP client request.
base2	Base Interface 2	1GbE	Base blade -> Backplane	Obtained by the DHCP client request.
fabric1	Fabric Interface 1	10GbE	Base blade -> Fabric Interface on Backplane	No IP address assigned.
fabric2	Fabric Interface 2	10GbE	Base blade -> Fabric Interface on Backplane	No IP address assigned.
front	Front Panel Interface	1GbE	Base blade	No IP address assigned.
update	Update channel	1GbE	Base blade	No IP address assigned.

3.7 Tools

This section describes CPUSpeed tools that can be used to change the processor performance governors.

3.7.1 Performance Tool

The performance tool, CPUSpeed allows to change the processor performance governors and the core frequency (for userspace governor) on a per core base. It utilizes data stored in the `/sys/device/system/cpu` directory. The following table describes various governors.

Governor	Description
Performance	Core is running with maximum frequency.
Ondemand	Cores in idle state are running at lowest frequency. When the core is changed to the utilized state, the frequency of the core is changed to maximum.
Powersave	Core is running with minimum frequency.
Userspace	Core frequency can be adjusted by the user (in steps).

If the P-States are limited by BIOS the required driver is not loaded and therefore the CPUSpeed tool can not work.

CPUSpeed supports the following options:

Option	Description
-d	Dump CPU Frequency/Governor Info
-h	Help
-p	Print supported governors
-s	Set governor/frequency. It supports the following options: -c : Specifies the core. Valid values are 0 .. 15. Omitting this option means, all cores. -f : Specifies the frequency. Valid values are 1200000..2100000. This parameter is ignored except for 'userspace governor'. -g : Specifies governors, such as performance, powersave, ondemand, and userspace.

Example:

```
root@ATCA7370-9:~#:/opt/bladeservices/tools#
/opt/bladeservices/tools/cpuspeed
####### CPU Frequency Info #######
Number Of Cores: 16
MinFrequency:    1200000
MaxFrequency:    2100000
Available Governors: ondemand userspace powersave performance
####### Frequency Info Per Core #######
Core:  Governor:     CurrentFrequency:
    0  userspace      1200000
    1  userspace      1200000
    2  userspace      1200000
    3  userspace      1200000
    4  userspace      1200000
    5  userspace      1200000
    6  userspace      1200000
    7  userspace      1200000
    8  userspace      1200000
    9  userspace      1200000
   10  userspace      1200000
```

```
11  userspace      1200000
12  userspace      1200000
13  userspace      1200000
14  userspace      1200000
15  userspace      1200000
```

Firmware Upgrade Facility

4.1 Overview

The Firmware Upgrade Facility (FUF) provides a uniform way to upgrade firmware on SMART EC hub blades and node blades. It consists of a Firmware Upgrade Command-line Utility (FCU) and specially prepared firmware recovery image files. On the ATCA-7370 FUF allows you to upgrade the following firmware types:

- BIOS firmware
- IPMC firmware
- FPGA image

4.2 Firmware Recovery Image Files

FCU supports specially prepared firmware recovery image (FRI) files as well as firmware images in the HPM.1 format. HPM.1 is a PICMG standard to upgrade IPMCs.

By default, the image files for the current hardware configurations are loaded as part of the BBS software in `/opt/bladeservices/rom` when the blade-specific firmware support packages are installed.

The following table describes the firmware recovery image (FRI) files as well as firmware images in the HPM.1 format.

Component	FCU Device Name	Type	Currently Supported	Upgrade Image Filename
BIOS	atca-7370-cpu	FRI	Yes	`atca7370-bios-<version>.fri`
IPMC Bootloader SINLGE CPU	H8S-AMCc B/L	HPM.1	Yes	`atca7370s_em_ibbl_<version>.hpm`
IPMC Bootloader DUAL CPU	H8S-AMCc B/L	HPM.1	Yes	`atca7370_em_ibbl_<version>.hpm`
IPMC Firmware SINGLE CPU IPMC	H8S-AMCc F/W	HPM.1	Yes	`atca7370s_em_ipmc_<version>.hpm`
IPMC Firmware DUAL CPU IPMC	H8S-AMCc F/W	HPM.1	Yes	`atca7370_em_ipmc_<version>.hpm`

Component	FCU Device Name	Type	Currently Supported	Upgrade Image Filename
IPMC FRU Information	H8S-AMCc F/I	HPM.1	No	N/A
IPMC Carrier FRU Information	H8S-AMCc F/C	HPM.1	No	N/A
Management Engine Firmware	MEFW	HPM.1	No	N/A
FPGA	FPGA	HPM.1	Yes	atca7370_em_fpga_<version>.hpm
ARTM 7360	AVR-AMCm F/W	HPM.1	Yes	artm7360_ipmc_<version>.hpm
ARTM 7360	AVR-AMCm B/L	HPM.1	Yes	artm7360_ibbl_<version>.hpm
ARTM 7360-L	AVR-AMCm F/W	HPM.1	Yes	artm7360_ipmc_<version>.hpm
ARTM 7360-L	AVR-AMCm B/L	HPM.1	Yes	artm7360_ibbl_<version>.hpm
ARTM 736x 10G SP	AVR-AMCm F/W	HPM.1	Yes	artm-736x-10G-SP-<version>.hpm
ARTM 736x 10G SP	AVR-AMCm B/L	HPM.1	Yes	artm-736x-10G-SP-<version>.hpm
ARTM 736x 10G	AVR-AMCm F/W	HPM.1	Yes	artm-736x-10G-<version>.hpm
ARTM 736x 10G	AVR-AMCm B/L	HPM.1	Yes	artm-736x-10G-<version>.hpm

4.3 Backup Concept

The firmware for the ATCA-7370 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 on the ATCA-7370 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 allows you to upgrade an inactive bank, that is, it allows you to upgrade only the bank from which the blade has not booted. This means that you need to reboot the blade in case you want to upgrade both banks.

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 ATCA-7370. Please note that the BootLoader firmware image is not part of the root file system by default, as the BootLoader update must be performed on SMART EC request only.

Each time the IPMC firmware is upgraded, the active firmware version is kept in flash memory and the backup firmware version is overwritten by the new one. Once the new IPMI firmware is uploaded to the IPMC, FCU activates 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.

BIOS BootBlock shall verify BIOS integrity by checksum calculation. If the active BIOS is found corrupt a switch to the backup BIOS shall occur (initiated by BIOS BootBlock). The switch shall also occur in case the BootBlock code is missing or corrupt. This BIOS independent switch is triggered by IPMC watchdog logic.

The following sample output displays the information regarding BIOS, IPMI, and FPGA. Depending on your setup, you may get a different output.

```
fcu -q
********************[[[[[ REPORT BEGIN ]]]]]********************
Operation: Query
#00 Device   : H8S-AMCc F/W
Bank #0 -      Active Version: 2.00.02000002
Bank #1 -    Rollback Version: 2.00.02000002
#01 Device   : H8S-AMCc B/L
Bank #0 -      Active Version: 2.00.02000002
#02 Device   : H8S-AMCc F/I
Bank #0 -      Active Version: 2.00.02000002
#03 Device   : H8S-AMCc F/C
#04 Device   : FPGA
Bank #0 -      Active Version: 10.00.00000000
Bank #1 -    Rollback Version: 10.00.00000000
#05 Device   : MEFW
Bank #0 -      Active Version: 2.01.02000000
```

```
#06 Device   : atca-7370-cpu
Bank #1 -      Active Version: 1.0.00000001
Bank #0 -    Rollback Version: 1.0.00000001
Bank marked for next use: #1
********************[[[[[ REPORT END ]]]]]********************
```

4.4 FCU - Firmware Upgrade Command-line Utility

The FCU represents the upgrade capabilities of firmware devices on a board. If the board is managed by an IPMC, then FCU requests IPMC to determine the board type. A board may have multiple devices which are low-level hardware components like BIOS, FPGA, and IPMC and so on. FCU abstracts firmware upgrade operations to a common set of operations for all devices. The FCU user does not need to know internals of a device.

FCU provides the following upgrade operations:

- Query the device to return the firmware version and other information.
- Validate the firmware image.
- Verify whether the image is applicable on the target device.
- Upgrade the device with the given firmware image.
- Mark a bank of the device to become active after the next reset or power cycle.
- Activate a bank of the device immediately.

4.4.1 Query Operation

Using the Query operation, FCU returns 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. Query shows all banks of a device. One of these banks is the active version, which means that the device was booted with the firmware installed in that bank. The device might have a second bank which contains the rollback version. You can switch to the rollback version with help of the Activate operation or with the help of the Mark operation in combination with a reboot or power-cycle.

If the device supports the Mark operation, then the Query operation shows bank which is marked for next use. Furthermore, FCU shows the capabilities of a device when you specify the `--details` option. Capabilities are the set of FCU operations which are supported and whether the device has manual or automatic rollback or has an self test implemented. The following example shows the BIOS device of ATCA-7370:

```
fcu -q -d atca-7370-cpu --details
********************[[[[[ REPORT BEGIN ]]]]]********************
```

```
Operation: Query
#08 Device: atca-7370-cpu
Bank #1 - Active Version: 1.3.00000002
Bank #0 - Rollback Version: 1.2.00000010
Bank marked for next use:#1

 Capabilities:
Upgrade operation: (x)
Mark operation: (x)
Activate operation: ( )
Compare operation: (x)
Service affected: ( )
Manual Rollback: (x)
Automatic Rollback: ( )
Self Test: ( )

********************[[[[[ REPORT END]]]]]********************
```

In the first line, the number of the device and its name are printed. In Bank1, the active version is stored and this bank is also marked for next use. Bank0 contains the rollback version. The device supports the upgrade, mark, and compare operation. The activate operation is not supported. The version is always a combination of decimal numbers.

4.4.2 Show Operation

The Show operation does not access any device. It only operates with the firmware image and it shows the meta data which is part of the image. Furthermore, it validates the firmware image to compare the checksum part of the meta data against the checksum of the embedded raw image. The output of the Show operation is similar to that of the Query operation.

```
fcu -sf atca-7370-bios-1.3.0.fri

********************[[[[[ REPORT BEGIN ]]]]]********************

Operation: Show
Manufacturer: EMERSON
Board: atca-7370
#00 Device: atca-7370-cpu
Bank #0- Version:1.3.00000000

********************[[[[[ REPORT END]]]]]********************
```

Additionally, Show operation prints the name of the manufacturer and the board to which the firmware of this image is compatible with. You can see only one bank, because a firmware image does not have multiple banks.

4.4.3 Mark Operation

The Mark operation selects the bank which is used after the next reset or reboot. With Mark operation, you have to specify the name of the device and the bank. In combination with the Mark operation and a following reset of the blade you can perform a manual rollback of a device. Note that, not every device supports this operation.

4.4.4 Activate Operation

The Activate operation is similar to Mark operation. It also sets a bank to become active. But, with the Activate operation, the bank is activated immediately. Not every device supports this operation. IPMCs which have the HPM.1 must support the Activate operation on the rollback bank and the deferred bank. If the rollback bank is activated by this operation, a manual rollback is performed.

4.4.5 Compare Operation

With the Compare operation, you can compare firmware images on the target with the images in the upgrade file. For this, you have to specify the firmware image file and the bank which you want to compare with.

4.4.6 Upgrade Operation

The Upgrade operation uploads the firmware image to the device. To have a valid firmware image in a bank, FCU tries to protect the active bank from being overwritten with a new firmware image. For the devices like IPMC, this protection is done by the IPMC firmware itself by communicating with FCU during the image upload. The IPMC firmware selects the bank to write the new image to. For other devices like BIOS, which does not have an upgrade intelligence, FCU selects the rollback bank. FCU determines this bank by reading an IPMI sensor which knows the active bank. You have to specify the firmware image with the Upgrade operation.

4.4.7 Verify Operation

The Verify operation checks if the firmware image is applicable on a device of the blade. It prevents you from writing a firmware image on to a non-compatible device. The Verify operation is always done before an Upgrade operation. So, you do not have to specify it explicitly when you perform an upgrade.

4.4.8 Command-Line Options

`fcu -- help`

USAGE

`fcu [operations] [operands]`

OPERATIONS

Table 4-1 Operation Command Description

Operation	Short Option	Description
`--query`	`-q`	Sets to perform a query operation.
`--upgrade`	`-u`	Upgrades the unused version of firmware. This operation requires the `--file` flag.
`--help`	`-h`	Displays this help message.
`--show`	`-s`	Displays information about the target which is included in the given upgrade file. This operation requires the `--file` flag.
`--verify`	`-v`	Performs verification of an upgrade file. This operation does NOT install the upgrade image.This operation requires the `--file` flag to be set.
`--mark`	`-m`	Marks the specified bank as next to boot. This operation depends on the `--bank` and the `--device` flag. This operation will not cause a power cycle of the blade.
`--activate`	`-r`	Activates the specified bank, This operation depends on the `--bank` and the `--device` flag.
`--compare`	`- c`	Compares the operation firmware with the image specified by the `--file` flag.
`--version`		Displays the version of this utility.
`--print-product-table`		Prints the table of products which are known by FCU.
`--match-product-table`		Performs the operation when image and target device match by their names in the product table.

OPERANDS

Table 4-2 Operands Command and Syntax description

Command	Syntax	Description
-d	--device=<device name>	Device to perform operation on.
	--details	Print the details of the device.
-f	--file=<filename>	Name of the firmware file.
-b	--bank=<bankletter>	Bank letter for Mark or Compare command.
	--level=[0-7]	Severity level of logging. 7 logs everything; default is 5.
	--log=ARG	File name for logging.
	--product-table=ARG	File with product table.
-t	--target=ipmbAddr[:mmcAddr]	Address of the IPMC or MMC device.

VALID FLAG COMBINATIONS

```
--query
--query --device=<device name>
--query --details
--upgrade --file=<filename>
--help
--show --file=<filename>
--verify  --file=<filename>
--verify --upgrade --file=<fileName>
--mark --bank=<bankletter> --device=<device-id>
--activate --bank=<bankletter> --device=<device-id>
--level=7
--log=/tmp/fcu.log
--compare --file=<filename> --bank=0
--version
--print-product-table
--product-table=fcu-product.tbl
-uf <filename> -t8a
--upgrade --file=<filename> --target=8c:62
--upgrade --match-product-table
```

4.4.9 The Product Table

fcu identifies a board by retrieving its manufacturer ID and product ID by a `GetDeviceId` IPMI command. The combination of these both IDs is unique and is mapped to a product name. For example, atca-7370. You will find the file in the Root File System in the following location: `/opt/bladeservices/etc/blsv-product-table`. This file stores the mapping in XML format. An entry for the atca-7370 is **`<product manufacturer="EMERSON" id="0x2009" name="atca-7370"/>`**.

To find the product table, fcu looks up the environment variable BLSV_PRODUCT_TABLE. You can overwrite the environment variable by specifying the option `--product-table` to point to another product table file when you call fcu in any operation.

Imagine you have Dual-CPU variant of the atca-7370 but it runs the IPMC firmware of the Single-CPU board. So you want to repair the mistake by invoking `fcu -uf <ipmc-image` for dual cpu>. In that step fcu prevents the upgrade, but it writes a note: target and image have different manufacturer/product ID combinations but would match by their names located in the blsv-product-table.

If you want to process the operation, you have to specify the option --match-product -table. In the product table, both CPU variants have the same product name which means that a match would be possible. To perform the upgrade, specify **`fcu -uf <ipmc-image`** for dual cpu> --match-product-table.

4.5 Upgrading a Firmware Image

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

NOTICE

The upgrade fails if the following is not taken into consideration:

Upgrade only one bank at a time, then reboot and verify the upgrade using the query option. If the upgrade fails and both banks become corrupted for any reason, the ATCA-7370 will be rendered inoperable.

Every activation stage would activate the latest uploaded firmware.

4.5.1 BIOS Upgrade

The BIOS can only be upgraded from the ATCA-7370 on which the BIOS is running. You have to upgrade the BIOS by using `fcu` command.

4.5.1.1 Upgrading the BIOS Firmware with FCU Utility

Follow these steps to upgrade the BIOS. The shown file names and paths are only meant as an example and should be replaced with file names and paths applicable to your configuration.

Upgrading with FRI format BIOS Firmware Image

1. Query the current BIOS firmware images on the blade.
   `fcu -qd atca-7370-cpu`
2. Show the version of the new BIOS file (to verify that it has actually a newer version than the already installed BIOS):
   `fcu --show -f /opt/bladeservices/rom/atca7370_em-bios_<version>.fri`
3. Upgrade the firmware image to the backup bank and mark the backup bank as the bank with which the CPU blade will boot up next time:
   `fcu --upgrade --file /opt/bladeservices/rom/atca7370_em-bios_<version>.fri`
   and
   `fcu --mark --bank=<bank number of rollback version> -d atca-7370-cpu`

NOTE: 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. Reset your payload CPU and the blade should be able to boot up with your newly upgraded BIOS firmware. You may double check the new BIOS firmware images on the blade with:
   `fcu -qd atca7370-cpu`

Upgrading with HPM.1 format BIOS Firmware Image

In case both banks of the BIOS are corrupted you can recover the board from another atca-7370 board in the chassis. For that purpose you need a BIOS image in HPM.1 format. First you have to know the IPMB address of the damaged board. On the other atca-7370, call `hpmcmd -c slotmap` to see the IPMB addresses of all slots. Then you can perform the upgrade with `fcu -uf atca-7370-bios-<version>.hpm -t <ipmb address>`. After the upgrade you can try to boot the damaged atca-7370 again to see if the BIOS comes up.

4.5.2 IPMC/MMC Firmware, Bootloader, and FRU Data Upgrade

Upgrading the IPMC Firmware

Follow these steps to upgrade an IPMC. The shown file names and paths are only meant as an example and should be replaced with file names and paths applicable to your configuration.

1. Query the current IPMC firmware images on the blade.
   fcu -q -d "H8S-AMCc F/W
2. Show the version of the new
   fcu --show -f /opt/bladeservices/rom/atca7370_em_ipmc_<version>.hpm
3. Upgrade the firmware image:
   f**cu --upgrade -f**
   /opt/bladeservices/rom/atca7370_em_ipmc_<version>.hpm
   Once the new IPMI firmware is uploaded, fcu activates 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.
4. Query the new image to ensure that the version information is correct:
   fcu -qd "H8S-AMCcF/W"

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

4.5.3 FPGA Upgrade

Upgrading the FPGA Firmware with FCU

The ATCA-7370 uses two pieces of EEPROM, which contains the FPGA firmware. The one is active, and other is inactive for the backup.

The following procedure describes how to upgrade the FPGA image stored in the user-programmable EEPROM. The shown file names and paths are only meant as an example and should be replaced with file names and paths applicable to your configuration.

1. Query the current FPGA firmware images on the blade.
   fcu -q -d FPGA
2. Show the version of the new FPGA file (to verify that it has actually a newer version than the already installed image):
   fcu --show -f /opt/bladeservices/rom/atca7370_em_fpga_<version>.hpm
3. Upload the firmware image.**fcu --upgrade --file**
   /opt/bladeservices/rom/atca7370_em_fpga_<version>.hpm
   or
   fcu -uf /opt/bladeservices/rom/atca7370_em_fpga_<version>.hpm

This upgrades the inactive user programmable FPGA EEPROM with specified FPGA image.

The whole blade will be power-cycled which is triggered by IPMC when the above upgrade was performed successfully.

Network Management Utility

5.1 Overview

The Network Management Utils tool can be used for the supervision of SFP devices and enable/disable the laser of SFP. The command `nmucmd -c show trx` displays all SFPs that are connected to the board. In order to use this, the `ixgbe_sfp` driver must be loaded.

```
nmucmd -c help
```

Table 5-1 Command and Related Description

Command	Description
`help`	Contains list of commands.
`show`	Prints information about different network entities.
`trx`	Modifies transceiver settings.
`version`	Displays the version of the tool.

The `show` command displays transceiver information which is retrieved from the EEPROM of the SFP. Therefore, the SFP must be compliant to the SFF-8472 specification. With the `trx` command, you can modify transceiver settings which are accessible in the EEPROM.

5.2 The Show Command

The `nmucmd` tool is based on the NMU API, which is part of the BLSV API. In the NMU world, an SFP is part of the hierarchical organization among switches and ports. A board may have multiple switches and a switch may have multiple ports. A SFP is always connected to a port. In case of `atca-7370` there isn't any switch chip on the board, but the NMU defines a virtual one. You can see it with `nmucmd -c show switch`.

```
nmucmd -c show switch
```

```
ixgbe: Intel 10G NIC
```

To get the information about an SFP, type **`nmucmd -c show <switch name> [portnumber] [property]`**. The port-number is one-based. Without the port number, you get all the SFPs that are associated with the switch. This property is optional and you get a list of the possible properties by calling `nmucmd -c help show`. The vendor name, temperature, serial number are the examples for a property. Without the property option you will see all the properties of a SFP. For the user it is difficult to find out which switch and port number combination is associated to which network interface. So, `nmucmd` supports to specify either the switch and port number or only the name of the network interface.

Examples:

```
nmucmd -c show trx ixgbe 1
```

or

```
nmucmd -c show trx rtm1
Transceiver info at port 1:
eth-device-name : rtm1
vendor-name : FINISAR CORP
vendor-part-number : FTLX8571D3BCL
vendor-revision : A
vendor-serial-number : AHE0DDQ
vendor-date : 04/04/2010
connector-type : LC
ethernet-code : 10G BASE-SR
encoding : 64b/66b Line Code
bitrate : 10300 MBd
tx-state : Enabled
rx-state : Available

internal-temperature : 33.636719 degrees Celsius
internal-supply-voltage : 3309400 uV
laser-bias-current : 7784 uA
capability-tx-fault : supported
capability-soft-tx-fault : supported
capability-tx-disable : supported
capability-soft-tx-disable : supported
capability-rx-los : not supported
capability-soft-rx-los : not supported
nmucmd -c show trx rtm1 vendor-part-number
vendor-part-number   : FTLX8571D3BCL
nmucmd -c show trx rtm2 tx-state
tx-state     : Enabled
```

5.3 The trx Command

With the `trx` command you can modify SFP settings. To get a list of settings you call `nmucmd -c help trx`.

Examples:

Turn off the transmitter of the SFP: `nmucmd -c trx rtm1 tx-state off`

Hardware Platform Management

6.1 Overview

Hardware management in ATCA systems is based on the Intelligent Platform Management Interface (IPMI) specification. IPMI commands can be complex and cumbersome. To facilitate blade-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, for example reading the FRU Inventory data. An HPM command can be the unifier for OEM IPMI commands that are different on different blade types, for example reading the CPU boot bank. For a catalogue of supported IPMI commands of the blade, refer to the respective IPMI manual.

The HPM package consists of:

- HPM daemon, `hpmagentd`
- Command line utility, `hpmcmd`
- Script framework for managing shutdown, reboot, and local ekeying events

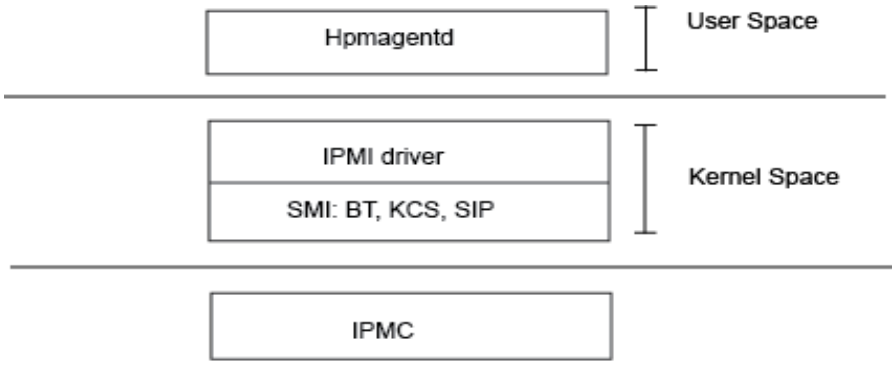
The HPM daemon is responsible to wait for events from the IPMC to perform a graceful shutdown/reboot of the operating system and to react when the link state of a channel's port is changed.

The utility `hpmcmd` displays the response of commands on the console in a human-readable format. HPM commands include:

- Retrieving and modifying FRU data
- Reading and controlling status of IPMI-controlled LEDs
- Communicating local slot location information
- Retrieving the event messages from the SEL of the IPMC

The hpmagentd and hpmcmd make use of OpenIPMI driver to talk to the local IPMC. The following figure shows the software levels that are involved in the HPM architecture:

Figure 6-1 Software Levels of the HPM Architecture



BT: Block Transfer Interface
SIP: Serial Interface Protocol
SMI: System Maintainence Interface
KCS: Keyboard Control Style

The System Management Interface (SMI) driver provides the low level interface for talking to the IPMC and could be a KCS driver or Block Transfer (BT) driver or other. If you need more information about the software aspects of the blade IPM controller, refer to the respective IPMI manual.

6.2 hpmagentd - HPM Agent Daemon

6.2.1 Description

The hpmagentd is the service to process events from the local IPMC. For any incoming event, it calls the respective script which is part of the hpmagent package. Event data is passed to a script by command line arguments. You can modify a script to fulfill your requirements. The following events are handled by the daemon:

Graceful Shutdown - When the IPMC receives an FRU activation request to deactivate an FRU, then it redirects the command to the hpmagentd through the IPMI driver. The hpmagentd invokes the shutdown script which is located in /opt/bladeservices/bin/hpmshutdown. By default within the script shutdown -h now is called which initiates a shutdown of the Linux immediately. Note that the IPMC powers down the processor in any case after a certain time. You may adjust this time with the Graceful Shutdown Timeout parameter of the IPMC, which can be set with a SetSystemBootOption IPMI command.

Graceful Reboot - On receiving an FRU control request to gracefully reboot the payload, the IPMC sends the command to the hpmagentd. The daemon invokes the reboot script which is available at /opt/bladeservices/bin/hpmreboot.

Ekeying Events - When the IPMC modifies the link state of a port, then it notifies the hpmagentd about that change. If the link goes down, then the script hpmekeydown is invoked otherwise the daemon calls hpmekeyup. Both scripts are placed in /opt/bladeservices/bin. By default these scripts are empty and do not do anything. To identify which port was changed, the hpmagentd passes an argument to an hpmekeyup script in the format: First the interface is specified: BC for Base Channel, FC for Fabric Channel, UC for Update Channel, and AMC for AMC Channel. The channel number and the port numbers of the channel are specified in the below example.

For example, the ports 1,2,3,4 of the channel 1 of a base interface is changed, then the argument looks like: "BC1.1,2,3,4".

6.2.2 Deployment

By default, the HPM daemon is installed in /opt/bladeservices/bin. With the hpmagentd binary, the scripts hpmreboot, hpmshutdown, hpmekeyup, and hpmekeydown are stored in that directory. Additionally, there are init script hpm to start and stop the daemon and the script hpmvar which exports some important variables to /etc/default/hpmvars to describe the board.

Synopsis

```
hpmagentd [options]
```

Parameters

--log =<file name>
You may specify a log file for the daemon with this option. If you do not use it, then the hpmagentd logs to the syslog.

-l --level
Specifies the level of message logging, where level is one of the standard syslog levels.

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

Log Level	Description
6	Information
7	Debug

`-v --version`
Displays the version of the daemon

`-L --disable-led`
Disable the LED management

`-r --reboot-script=<script>`
Use the specified Blade Reboot script. Default is `/opt/bladeservices/bin/hpmreboot`

`-s --shutdown-script=<script>`
Use the specified Blade Shutdown script. Default is `/opt/bladeservices/bin/hpmshutdown`

`-u --ekey-up=<script>`
Called when a port is enabled. Default is `/opt/bladeservices/bin/hpmekeyup`

`-d --ekey-up=<script>`
Called when a port is disabled. Default is `/opt/bladeservices/bin/hpmekeydown`

`-h --help`
Displays the help message

`-i --dont-daemonize`
Run interactively

6.2.3 hpm - Init.d Script

The `hpm init.d` script allows to start, stop, and restart the `hpmagentd`. It can be linked to a run level to automatically start the daemon at Linux boot time and to stop it when Linux shuts down.

Synopsis

```
hpm { start | stop | restart | status }
```

Parameters

`Start` - Starts the `hpmagentd`

`Stop` - Stops the `hpmagentd`

`Restarts` - Stops and Starts the `hpmagentd` again

`Status` - Reports if the `hpmagentd` is currently running or not

6.3 hpmcmd - HPM Command Utility

6.3.1 Overview

The HPM command utility talks directly to the IPMC through the IPMI driver which is part of the Linux kernel. It takes care of translating the user-friendly commands into the elaborated IPMI commands that the IPMC is able to understand. Those IPMI commands are transferred to the local IPMC. The HPM command utility can be started in interactive mode, where a prompt is displayed and the user enters commands; or it can process a single command.

By default, the hpmcmd binary is installed in `/opt/bladeservices/bin` directory. hpmcmd prints log messages to the syslog. By default, the log level is set to the severity error. You can modify the log severity by setting the environment variable HPMCMD_LOG_LEVEL. For example to set severity debug: export HPMCMD_LOG_LEVEL=7. For a list of log severities, refer to the table in *Deployment on page 53*.

Synopsis

`hpmcmd [options]`

Parameters

`-c`
Processes a single command

`--help -h`
Displays this help message

`-v`
Verbose mode for some commands
Some commands like `fruinfoget`, print more details if this options is given.
Commands which do not support the verbose option ignores it.

`-t`
Sends the command to a remote target.
For more information, refer *Target Addressing with hpmcmd on page 56*. If this option is not given, then the command goes to the local IPMC.

`-p`
Changes the prompt when `hpmcmd` runs in interactive mode.

6.3.2 Target Addressing with hpmcmd

Using the `-t` option, you can send commands to other IPMCs or MMCs which participate on an IPMB. It has the following syntax: `-t <IPMB address>[: MMC address]`. The addresses must be set in hex format.

To send the command to an AMC attached on this blade use:
`-t0:72 or -tlc:c0`

To send the command to another IPMC type:
`-t 92`

To send the command to an MMC, which is attached on another blade in the shelf specify:
`-t 82:72`

6.3.3 Command Overview

The following table lists all commands from the hpmcmd program available on ATCA-7370. You can display this list and a short command description using the help command (see section *help on page 68*). A detailed description of the commands is given in section *Supported Commands on page 58*.

Table 6-1 Command Overview

Command	Description
bootbankget	Gets the bootbank to boot from
bootbankset	Sets the bootbank to boot from
bootparamerase	Erase boot parameter value
bootparamget	Get boot parameter value
bootparamset	Set a boot parameter value
chinfo	Retrieve channel information
cmd	Executes any IPMI command
deviceid	Gets the Device Id
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

Table 6-1 Command Overview (continued)

Command	Description
fruread	Allows to read x number of bytes from the FRU
fruwrite	Allows to write x number of bytes from the FRU
fwprogevent	Sends a Firmware Progress Sensor Event
help	Gets list of commands
ipmbaddress	Gets the IPMB address
ipmcstatus	Gets the IPMC status
Lancfgget	Get LAN configuration parameter
Lancfgset	Set LAN configuration parameter
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
macaddress	List the MAC addresses
partnumber	Gets the board part number
physlotnumber	Gets the board physical slot number
portget	Shows the current state E-Key governed intfs
portset	Enables/Disables ports in a channel
posttypeget	Gets the posttype to run at boot
posttypeset	Sets the posttype to run at boot
sdr	Shows the SDR records
sdr_dump	Shows the SDR records in raw format
sdrinfo	Shows SDR information
sel	Shows the SEL records

Table 6-1 Command Overview (continued)

Command	Description
selclear	Erases all contents from the SEL
selinfo	Shows SEL information
sendamc	Sends an IPMI request to an MMC behind a remote IPMC
sendcmd	Sends an IPMI request to an IPMB address IPMC
serialoutputget	Determines which serial output source goes to a particular serial port connector
serialoutputset	Select the serial output source of the serial port connector
shelfaddress	Gets the Shelf Address String
shelfslots	Prints the number of slots in the shelf
shelftype	Gets the Shelf Type from the Shelf FRU (Board Product Name)
slotmap	Prints the slotmap of the shelf
slotnumber	Shows the board slot number
solcfgget	Get SOL configuration parameter
solcfgset	Set SOL configuration parameter
version	Shows the `hpmcmd` version
watchdog	Controls Payload WDT functionality

6.3.4 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` option (that is, interactive mode), you simply enter these commands at the HPM command prompt.

6.3.4.1 bootbankget

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

Firmware for the CPU on SMART EC ATCA blades 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 blade 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 blade. On the ATCA-7370, 0 selects the SPP processor and 1 addresses the GPP.

Example

`hpmcmd -c bootbankget 0`

BANK1

6.3.4.2 bootbankset

This command sets the boot bank for a particular CPU from which the blade 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 blade. On the ATCA-7370, 0 selects the SPP processor and 1 addresses the GPP.

`newBootBank`
Can be set to `BANK1, BANK2...`

Example

`hpmcmd -c bootbankset 1 bank1`

6.3.4.3 bootparamerase

This command erases the boot parameter.

Synopsis

```
bootparamerase section [name] [-t ipmbAddr[:mmcAddr]]
```

Parameters

section
Can have value as USER, DEFAULT, TEST, or OS_PARAM.

name
Specifies name of the parameter.

t
Sends the command to ipmbAddr:mmcAddr.

6.3.4.4 bootparamget

This command gets the boot parameter value.

Synopsis

```
bootparamget section [name] [-t ipmbAddr[:mmcAddr]]
```

Parameters

section
Can have value as USER, DEFAULT, TEST, or OS_PARAM.

name
Specifies name of the parameter.

t
Sends the command to ipmbAddr:mmcAddr.

6.3.4.5 bootparamset

This command sets the boot parameter value.

Synopsis

```
bootparamset section name=value [-t ipmbAddr[:mmcAddr]]
```

Parameters

section
Can have value as USER, TEST, or OS_PARAM.

name
Specifies name of the parameter.

t
Sends the command to ipmbAddr:mmcAddr.

6.3.4.6 chinfo

Retrieves information about an IPMI Channel.

Synopsis

`chinfo <channel>`

Parameters

`channel`
Channel number

Example

`hpmcmd -c chinfo 1`

```
Channel Medium Type   : Asynch. Serial/Modem (RS-232)
Channel Protocol Type : TMode
Session Support       : session-less
Active Session Count  : 0
Protocol Vendor ID    : 00400A
```

`hpmcmd -c chinfo 0`

```
Channel Medium Type   : IPMB (I2C)
Channel Protocol Type : IPMB-1.0
Session Support       : session-less
Active Session Count  : 0
Protocol Vendor ID    : 001BF2
```

`hpmcmd -c chinfo 5`

```
Channel Medium Type   : OEM
Channel Protocol Type : OEM
Session Support       : session-less

Active Session Count  : 0
Protocol Vendor ID    : 0065CD
```

`hpmcmd -c chinfo 4`

```
Channel Medium Type   : System Interface (KCS, SMIC, or BT)
Channel Protocol Type : KCS
Session Support       : session-less
Active Session Count  : 0
Protocol Vendor ID    : 001BF2
```

6.3.4.7 cmd

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 command`
The ipmi command specifies the sequence of hexadecimal bytes as entered using the ipmicmd tool from the OpenIPMI library. The ipmi command can have value, such as:
`0f 00 XX ZZ W1 W2 ... Wn`

In this example:
`XX` specifies netfunct in hexadecimal.
`ZZ` specifies the command number, as stated in the IPMI/PICMG spec.
`W1` to `Wn` specifies the data bytes according to the command supports.

`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`>, where `SA` is the slave address.

`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 AMC attached on this blade. MMC address of the AMC is 7a:
`hpmcmd -c cmd 7 7a 0 6 1`

6.3.4.8 deviceid

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`.

Example

`hpmcmd -c deviceid`

```
DEVICEID INFORMATION

--------------------

Device Id           =  0x00
Device Revision     =  0x00
Device Mode         =  Normal Operation; Supports Device SDR
Firmware Revision   =  2.00
IPMI Version        =  1.5
Device Support      =  IPMB Evnt Gen; FRU; SEL; Sensor;
Manufacturer ID     =  0x000065CD
Product ID          =  0x002B
Auxiliary Revision  =  0x00000013
```

6.3.4.9 frudata

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

Synopsis

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

Parameters

`fruid`
Is 0 for the SPP, 1 for the DMC0, 2 for DMC1, 3 for DMC2, 4 for the GPP, 5 for the RTM

`-t`
Sends the command to `ipmbAddr:mmcAddr.`

Example

```
hpmcmd -c frudata 0
01 00 00 01 0b 13 00 e0 01 0a 19 3a 2d 7c c7 45
4d 45 52 53 4f 4e d7 50 43 41 2c 41 54 43 41 2d
38 33 31 30 2d 49 41 2d 54 35 2d 43 30 31 c7 45
30 32 46 43 34 39 cb 30 31 30 36 38 33 30 48 30
```

```
35 4c cc 66 72 75 2d 69 6e 66 6f 2e 69 6e 66 c1
00 00 00 00 00 00 00 4a 01 08 19 c7 45 4d 45 52
53 4f 4e d3 41 54 43 41 2d 38 33 31 30 2d 49 41
2d 54 35 2d 43 30 31 cb 30 31 30 36 38 33 30 48
30 35 4c c4 52 31 2e 31 c7 45 30 32 46 43 34 39
c0 c0 c1 00 00 00 00 6a c0 02 2a 42 d2 cd 65 00
01 01 04 11 06 01 ec 9e cd 02 d1 a1 11 06 01 ec
9e cd 02 d1 a2 11 06 01 ec 9e cd 02 d1 a3 05 06
.....
```

6.3.4.10 fruinfoget

This command retrieves information from the specified FRU.

Synopsis

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

Parameters

`fruid`
Is 0 for the SPP, 1 for the DMC0, 2 for DMC1, 3 for DMC2, 4 for the GPP, 5 for the RTM.

`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 area manufacturer
bproductname	Board area product name
bserialnumber	Board area serial number
bpartnumber	Board area part number
pmanufacturer	Product area manufacturer
pproductname	Product area product name
ppartnumber	Product area part number
pversion	Product area version
pserialnumber	Product area serial number
passettag	Product area asset tag

`-v`
Verbose mode to get point-to-point connectivity information when no specific field is requested.

`-t`
Sends the command to `ipmbAddr:mmcAddr`.

Example

```
hpmcmd -c fruinfoget 0
Common Header:
    Format Version = 1
Board Info Area:
    Version   = 1
    Language Code   = 25
   Mfg Date/Time          = Jun 22 10:02:00 2011 (8138042 minutes since
1996)
    Board Manufacturer       = EMERSON
    Board Product Name       = PCA,ATCA-7370
    Board Serial Number      = E02FC49
    Board Part Number        = 0106830H05L
    FRU Programmer File ID   = fru-info.inf
Product Info Area:
    Version   = 1
    Language Code            = 25
    Manufacturer Name        = EMERSON
    Product Name             = ATCA-7370
    Product Part / Model#    = 0106830H05L
    Product Version          = R1.1
    Product Serial Number    = E02FC49
    Asset Tag                =
    FRU Programmer File ID   =


Multi Record Area:
    OEM MAC Addresses Record (ID=0x01)
        Version = 1
    PICMG Board Point-to-Point Connectivity Record (ID=0x14)
        Version = 0
    AMC Carrier Information Table Record (ID=0x1a)
        Version = 0
    AMC Carrier Activation and Current Management Record (ID=0x17)
        Version = 0
```

```
AMC Carrier Point-to-Point Connectivity Record (ID=0x18)
    Version = 0
AMC Point-to-Point Connectivity Record (ID=0x19)
   Version = 0
```

6.3.4.11 fruinv

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 SPP, 1 for the DMC0, 2 for DMC1, 3 for DMC2, 4 for the GPP, 5 for the RTM.

`-t`
Sends the command to `ipmbAddr:mmcAddr`.

Example

`hpmcmd -c fruinv 0`

```
FruSize = 1024
Accessed Units = Bytes
```

6.3.4.12 fruread

This command gets nBytes of fruid from the startAddress of the specified FRU.

Synopsis

`fruread <fruid> <startAddress> <nBytes> [-t ipmbAddr[:mmcAddr]]`

Parameters

`fruid`
Is 0 for the SPP, 1 for the DMC0, 2 for DMC1, 3 for DMC2, 4 for the GPP, 5 for the RTM.

`startAddress`
Is the starting address for reading the fruid.

`nBytes`
Number of bytes to read in decimal.

`-t`
Sends the command to `ipmbAddr:mmcAddr`.

Example

```
hpmcmd -c fruread 0 0 280
01 00 00 01 0b 13 00 e0 01 0a 19 3a 2d 7c c7 45
4d 45 52 53 4f 4e d7 50 43 41 2c 41 54 43 41 2d
38 33 31 30 2d 49 41 2d 54 35 2d 43 30 31 c7 45
30 32 46 43 34 39 cb 30 31 30 36 38 33 30 48 30
35 4c cc 66 72 75 2d 69 6e 66 6f 2e 69 6e 66 c1
00 00 00 00 00 00 00 4a 01 08 19 c7 45 4d 45 52
53 4f 4e d3 41 54 43 41 2d 38 33 31 30 2d 49 41
2d 54 35 2d 43 30 31 cb 30 31 30 36 38 33 30 48
30 35 4c c4 52 31 2e 31 c7 45 30 32 46 43 34 39
c0 c0 c1 00 00 00 00 6a c0 02 2a 42 d2 cd 65 00
01 01 04 11 06 01 ec 9e cd 02 d1 a1 11 06 01 ec
9e cd 02 d1 a2 11 06 01 ec 9e cd 02 d1 a3 05 06
01 ec 9e cd 02 d1 a4 c0 02 32 c3 49 5a 31 00 14
00 01 e3 5a 0e 32 65 0f 9b 86 62 4a c0 d7 a3 e5
72 d2 01 11 00 00 02 11 00 00 41 2f 10 00 41 21
00 00 42 2f 10 00 42 21 00 00 81 01 0f 00 c0 02
08 57 df 5a 31 00 1a 00 02 01 01 c0 02 0c 97 9b
5a 31 00 17 00 32 00 05
```

6.3.4.13 fruwrite

This command allows to write hexadecimal byte values to `fruid` starting at `startAddr`.

Synopsis

`fruwrite <fruid> <startAddress> <hexval1> [hexval2] [...] [hexval16] [-t ipmbAddr[:mmcAddr]]`

Parameters

`fruid`
Is 0 for the SPP, 1 for the DMC0, 2 for DMC1, 3 for DMC2, 4 for the GPP, 5 for the RTM.

`startAddress`
Starting address for writing.

`hexval1 .. hexvalN`
Is the hexadecimal value to write.

`-t`
Sends the command to `ipmbAddr:mmcAddr`.

6.3.4.14 fwprogevent

This command sends a Firmware Progress Sensor Event to the Shelf Manager SEL. Refer IPMI specifications for details on values.

Synopsis

```
fwprogevent <data1> <data2> <data3>
```

Parameters

`data1`
Stores hexadecimal value as; 00 for Error, 01for Hang, and 02 for Progress.

`data2`
Stores hexadecimal value as; 00-0D for Error, 00-19 for Hang or Progress.

`data3`
Stores hexadecimal value as; FF unless an OEM data2 is specified.

6.3.4.15 help

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

Synopsis

```
help
```

6.3.4.16 ipmbaddress

This command retrieves the blade IPMB address.

Synopsis

```
ipmbaddress
```

Example

```
hpmcmd -c ipmbaddress
ipmbaddress is 0x92
```

6.3.4.17 ipmcstatus

This command retrieves the status of given IPMC.

Synopsis

```
ipmcstatus [-t ipmbAddr]
```

Parameters

`-t`
Specifies the target with `ipmbAddr`.

Example

`hpmcmd -c ipmcstatus`

```
IPMC Mode                =  NORMAL
Payload Control          =  Enabled
IPMC Outstanding Events  =   None
```

6.3.4.18 Lancfgget

Get LAN configuration parameter

Synopsis

`lancfgget <channel> [param]`

Parameters

`channel`
channel number

`param`
- auth-type-support|
- auth-type-enables
- ip-addr
- ip-addr-src
- mac-addr
- subnet-mask
- ipv4-header-params
- primary-rmcp-port
- secondary-rmcp-port
- bmc-generated-arp-control
- gratuidous-arp-interval
- default-gateway-addr
- default-gateway-mac-addr
- backup-gateway-addr
- backup-gateway-mac-addr
- community-string
- num-destinations
- destination-type
- destination-addr
- vlan-id
- vlan-prio

rmcp-cipher-support
rmcp-ciphers
rmcp-priv-levels
dst-addr-vlan-tags

Example

```
hpmcmd -c lancfgget 5
```

```
IP Address          : 211.21.168.192
Subnet Mask         : 0.255.255.255
Default Gateway     : 0.0.0.0
MAC Address         : ec:9e:cd:02:d1:a4
```

6.3.4.19 Lancfgset

Set LAN configuration parameter

Synopsis

```
lancfgset <channel> <param> <value>
```

Parameters

channel
channel number

param
- ip-addr
- subnet-mask
- default-gateway-addr

Example

```
hpmcmd -c lancfgset 4 ip-addr 192.168.24.10
```

6.3.4.20 ledget

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 blade, 5 for the RTM.

`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 blade.

`-t`
Sends the command to `ipmbAddr:mmcAddr`.

Example

```
hpmcmd -c ledget 0 led1

Current State = OVERRIDE

State Function/(ms) Duration(ms)      Color
---------------------------------------------
Override         On       Always        Red
```

6.3.4.21 ledprop

This command displays the FRU LED properties under IPMC control.

Synopsis

```
ledprop <fruid>
```

Parameters

`fruid`
Is 0 for the main board and 5 for the RTM.

Example

```
hpmcmd -c ledprop 0

FRU LEDs under IPMC control:

LED0 = BLUE
LED1 = RED or AMBER
LED2 = GREEN
```

6.3.4.22 ledset

This command controls the override state of a specific FRU LED.

Synopsis

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

Parameters

`fruid`
Is 0 for the main blade and 5 for the rear transition module.

`led`
Is BLUE for the hot swap LED or LEDN for FRU LED<n>. <n> is a number between 0 and the maximum FRU LEDs supported by the blade.

`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 300ms.
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. The default duration is 5000ms.

`offms`
Specifies OFF duration in milliseconds. It can have value from 10ms to 2500ms in the 10ms increments. It is valid only if the operation is BLINK.

`onms`
Specifies ON duration in milliseconds. It can have value from 10ms to 2500ms in the 10ms increments. It is valid only if the operation is BLINK.

`color`
LED0 = BLUE
LED1 = RED or AMBER
LED2 = GREEN (if supported by IPMC)
LED3 = AMBER (if supported by IPMC)

`-t ipmbAddr`
Sends the command to `ipmbAddr:mmcAddr`.

Example

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

6.3.4.23 loglevelget

This command retrieves the current `hpmagentd` log level. The log level of hpmcmd can be set with the environment variable `HPMCMD_LOG_LEVEL`, for example export HPMCMD_LOG_LEVEL=7 to set debuglevel. All log messages are sent to the syslog.

LogLevels

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

Synopsis

```
loglevelget
```

Example

```
hpmcmd -c loglevelget
5
```

6.3.4.24 macaddress

This command retrieves a list of available MAC addresses.

Synopsis

```
macaddress [fruid]
```

Parameters

```
Fruid
```

Example

```
hpmcmd -c macaddress 1
Multi-Tpye Interface       MAC Addr :  ec:9e:cd:02:d1:9e
Multi-Tpye Interface       MAC Addr :  ec:9e:cd:02:d1:9f
Front or Real Panel Interface MAC Addr :  ec:9e:cd:02:d1:a0
```

6.3.4.25 partnumber

This command retrieves the part number (FRU 0) of the main blade.

Synopsis

```
partnumber
```

Example

hpmcmd -c partnumber

6.3.4.26 physlotnumber

This command retrieves the physical slot number in which the blade is plugged in.

Synopsis

physlotnumber

6.3.4.27 portget

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] [devid]

Parameters

interface
Valid values are:
BASE | FABRIC | UPDATE | AMC

channel
It is a number from 1 to the maximum number of channels supported for the interface. Node blades usually support 2 Base and 2 Fabric channels, and switch blades support 16 Base, 15 Fabric, and 1 Update channels.

devid
For AMC only: it is an on-carrier device ID that identifies the on-carrier device to which the desired channel is connected.

Example

hpmcmd -c portget

```
STATE      INTERFACE     CHANNEL     LINKTYPE      LINKEXT     GROUP    PORTS
-----------------------------------------------------------------

ENABLED    BASE        1       BASE          0         0          0
DISABLED   BASE        2       BASE          0         0          0
DISABLED   FABRIC      1       ETHER         1         0      0,1,2,3
ENABLED    FABRIC      1       ETHER         0         0          0
DISABLED   FABRIC      2       ETHER         1         0        1,2,3
DISABLED   FABRIC      2       ETHER         0         0          0
DISABLED   UPDATE      1       OEM           0         0          0
DISABLED   AMC 00h     0       PCIEXPRESS    0         0          0
```

6.3.4.28 portset

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> <typeX> <ports> <oper> [devid] [-t ipmbAddr[:mmcAddr]]

Parameters

`intf`
Valid values are:
`BASE | FABRIC | UPDATE | AMC`

chan
It is a number from 1 to the maximum number of channels supported for the interface. Node blades usually support 2 Base and 2 Fabric channels, and switch blades support 16 Base, 15 Fabric, and 1 Update channels.

`grpid`
Specifies the group id. Always 0 according to the current shelf FRU information.

`type`
Valid values are:
`BASE | ETHER | EXPRESS | INFINI | STAR | OEM`

`typeX`
Valid values are:
`0` (for 1000Base-BX)
`1` (for 10GBase-BX4)
`2` (for FC-PI)

`ports`
A sequence of ports to act on.
For base and update channels, port is always 0.
For fabric channels, port can specify up to 4 ports as specified in PICMG 3.1:
Option 1: 0 (for port 0)
Option 2: 01 (for ports 0,1)
Option 3: 0123 (for ports 0,1,2,3)
Option 7: 3 (for port 3)

`oper`
Valid values are `DISABLE or ENABLE`.

`devid`
For AMC only: it is an on-carrier device ID that identifies the on-carrier device to which the desired channel is connected.

`-t ipmbAddr`
Sends the command to `ipmbAddr:mmcAddr.`

Example

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

6.3.4.29 posttypeget

This command retrieves the postType to which the board is currently set to run at boot time, for the specified CPU.

Synopsis

`posttypeget <payload_cpu_selector>`

Parameters

`payload_cpu_selector`
The specified CPU is set to postType to run. Is 0 for the GPP, 1 for the SPP.

Example

`hpmcmd -c posttypeget 1`

`LONG`

6.3.4.30 posttypeset

This command sets the postType to which the board is currently set to run at boot time, for the specified CPU.

Synopsis

`posttypeset <payload_cpu_selector> <newPostType>`

Parameters

`payload_cpu_selector`
It is an integer between 0 and number of CPU devices supported per board.

`newPostType`
Valid values are: `SHORT | LONG.`

6.3.4.31 sdr

This command shows the SDR records.

Synopsis

`sdr`

Example

hpmcmd -c sdr

```
recID 1: full sensor record
owner is IPMB 20 sensor num 10 on lun 00 channel 00
logical entity: power module - instance 61
SBC +1.05V Vtt : voltage : threshold
recID 2: full sensor record
owner is IPMB 20 sensor num 11 on lun 00 channel 00
logical entity: power module - instance 61
SBC +1.1V : voltage : threshold
recID 3: full sensor record
owner is IPMB 20 sensor num 12 on lun 00 channel 00
logical entity: power module - instance 61
SBC +1.2V SAS : voltage : threshold
.
.
.
recID 74: OEM sensor record
```

6.3.4.32 sdr_dump

This command shows the SDR records in binary and hexadecimal 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"
```

6.3.4.33 sdrinfo

This command shows the SDR information.

Synopsis

```
sdrinfo
```

Example

hpmcmd -c sdrinfo

```
SDR Information:
LUN 0 has 031 sensors; static sensor population
LUN 1 has 000 sensors;
LUN 2 has 000 sensors;
LUN 3 has 000 sensors;
```

6.3.4.34 sel

Show the SEL records.

Synopsis

```
Sel
```

Example

hpmcmd -c sel

```
0x0023: Event: at: Sep 21 14:19:12 2011; from:(0x8c,0,0); sensor:(0xda,31);
event:(0x6f,asserted): a0 00 01
0x0024: Event: at: Sep 21 14:19:12 2011; from:(0x8c,0,0);sensor:(0xda,31);
event:(0x6f,asserted): a0 81 00
0x0025: Event: at: Sep 21 14:19:12 2011; from:(0x8c,0,0); sensor:(0xda,31);
event:(0x6f,asserted): a0 00 01
0x0026: Event: at: Sep 21 14:19:12 2011; from:(0x8c,0,0); sensor:(0xda,31);
event:(0x6f,asserted): a0 00 01
0x0027: Event: at: Sep 21 14:19:12 2011; from:(0x8c,0,0); sensor:(0xda,31);
event:(0x6f,asserted): a0 41 00
0x0028: Event: at: Sep 21 14:19:12 2011; from:(0x8c,0,0); sensor:(0xda,31);
event:(0x6f,asserted): a0 81 00
0x0029: Event: at: Sep 21 14:19:12 2011; from:(0x8c,0,0); sensor:(0xda,31);
event:(0x6f,asserted): a0 18 00
0x002A: Event: at: Sep 21 14:19:13 2011; from:(0x8c,0,0); sensor:(0xda,31);
event:(0x6f,asserted): a0 81 00
```

6.3.4.35 selinfo

Show the SEL information.

Synopsis

```
Sel
```

Example

hpmcmd -c selinfo

```
SEL version: 1.5
Number of log entries: 1023
Free space: 0 bytes
Events have been dropped due to lack of space in the SEL
Last addition timestamp: Nov 15 08:40:40 2011
Last erase timestamp: Jan  1 00:00:00 1970
```

6.3.4.36 selclear

Erases all contents of the SEL.

Synopsis

```
Selclear
```

Example

hpmcmd -c selclear

6.3.4.37 serialoutputget

Determines which serial output source goes to a particular serial port connector.

Synopsis

```
serialoutputget <Serial Connector Type> <Instance Number>
```

Parameters

```
Serial Connector Type
  0  Face Plate Connector
  1  Backplane Connector
  2  On-board Connector
  3  On-board Device (route to another chipset)

Instance Number
```
zero-based instance number

Example

hpmcmd -c serialoutputget 0 1

```
Serial Output Selector: 1
```

6.3.4.38 serialoutputset

Select the serial port output source for a serial port connector.

Synopsis

```
serialoutputset <Serial Connector Type> <Instance Number> <Serial Output
Selector>
```

Parameters

`Serial Connector Type`
- `0` Face Plate Connector
- `1` Backplane Connector
- `2` On-board Connector
- `3` On-board Device (route to another chipset)

`Instance Number`
zero-based instance number

`Serial Output Selector`
an integer value >= 0

6.3.4.39 sendamc

This command allows to send any of the commands supported in the IPMI specifications 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.

6.3.4.40 sendcmd

This command allows a user to send any of the commands supported in the IPMI specifications 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 1
Completion code: 0x00 (0) Success
Response data  : 00 80 01 20 02 2D CD 65 00 1A 20 00 00 00 24
```

6.3.4.41 shelfaddress

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

Synopsis

shelfaddress

Example

```
hpmcmd -c shelfaddress
01
```

6.3.4.42 shelfslots

This command retrieves the total number of blade slots in the shelf.

Synopsis

```
shelfslots
```

Example

```
hpmcmd -c shelfslots
14 slots
```

6.3.4.43 shelftype

This command retrieves the shelf FRU (IPMB 20) Board Area Product Name (FRU 254).

Synopsis

`shelftype`

Example

`hpmcmd -c shelftype`

```
CHS1406
```

6.3.4.44 slotmap

This command prints a slotmap table for the shelf the blade is installed in.

Synopsis

`slotmap`

Example

`hpmcmd -c slotmap`

```
------------------------------------------------------------
Physical Slot: 01 02 03 04 . 05 06 07 08 09 10 . 11 12 13 14
Logical Slot: 01 03 05 07 . 09 11 13 04 06 08 . 10 12 14 02
IPMB Address: 82 86 8A 8E . 92 96 9A 88 8C 90 . 94 98 9C 84
------------------------------------------------------------
```

6.3.4.45 slotnumber

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

Synopsis

`slotnumber Parameters`

Example

`hpmcmd -c slotnumber`

```
2
```

6.3.4.46 solcfgget

Get SOL configuration parameter

Synopsis

`solcfgget <channel> [param]`

Parameters

channel
channel number

param
- enable
- authentication
- char-settings
- retry
- nonvolatile-bit-rate
- volatile-bit-rate
- payload-channel
- payload-port

Example

```
hpmcmd -c solcfgget 5
```

```
Enabled                         : false
Non-Volatile Bit Rate       (kps): 9.6
Volatile Bit Rate           (kps): 9.6
Payload Port                    : 623
```

6.3.4.47 solcfgset

Set SOL configuration parameter.

Synopsis

```
solcfgset <channel> <param> <value>
```

Parameters

channel
channel number

param
- force-encryption true|false
- force-authentication true|false
- privilege-level user|operator|administrator|oem
- char-accumulate-interval 1-1275 (ms)
- char-send-threshold 0-255
- retry-count 0-255
- retry-interval 0-2550 (ms)
- non-volatile-bitrate 9.6|19.2|38.4|57.6|115.2
- volatile-bitrate 9.6|19.2|38.4|57.6|115.2
- port 0-255

6.3.4.48 version

This command prints the version of the hpmcmd software.

Synopsis

`version`

Example

```
hpmcmd -c version
3.3.1
```

6.3.4.49 watchdog

This command is used to 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

Value	Description
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

Board Control Module

7.1 Overview

Board control is a kernel module which provides access to the board FPGA. The board control module creates a boardinfo directory in the `/proc` file system that contains general information about ATCA-7370. The following table describes the information of the boardinfo directory.

File	Description	Sample output
`board_name`	Shows the board name, as provided by the BIOS.	`ATCA-7370-0GB`
`board_version`	Shows the board version, as provided by the BIOS.	`0106861M01A`
`bios_version`	Shows the BIOS version.	`1.3.7 X64`
`board_releasedate`	Shows release date of BIOS.	`04/17/2014`
`board_serialnumber`	Shows the serial number of the board, as provided by the BIOS.	`9244225`

`fpga`	Shows additional FPGA information.	`Board Module Version: ATCA7370`
		`FPGA version: 0x0A`
		`--- Flash Control Register -----`
		`--- Serial Line Routing ------`
		`COM1 to FacePlate and COM2 to RTM`

File	Description	Sample output
summary	Shows a summary of the board state (FPGA registers) and BIOS provided information.	Board Vendor: ARTESYN Board Name: ATCA-7370-0GB Board Version: 0106861M01A Board Serial Number: 9244225 BIOS Vendor: Emerson BIOS Version: 1.3.7 X64 BIOS Release Date:04/17/2014 Memory Module: Device/Bank: J8K3/CHANNEL A DIMM 0 Size: 4096 Mbyte Data Width: 64 Bit Manufacturer: CE00 Memory Module: Device/Bank: J8K1/CHANNEL B DIMM 0 Size: 4096 Mbyte Data Width: 64 Bit Manufacturer: CE00 Memory Module: Device/Bank: J4F3/CHANNEL D DIMM 0 Size: 4096 Mbyte Data Width: 64 Bit Manufacturer: CE00 Memory Module: Device/Bank: J5A2/CHANNEL F DIMM 0 Size: 4096 Mbyte Data Width: 64 Bit Manufacturer: CE00 IPMI Interface Type: 1 KCS Keyboard Control Style) IPMI Spec Rev: 2.0 I2C Slave Addr: 0x98 NV Stor.Dev.Addr: 0x0 Base Addr: 0x00000CA3 IRQ: 0x66

Kernel and Root File System Config using WRL 4.3

8.1 Building Kernel and Root File System

This section provides an introduction to build the Linux kernel and Root File System for an ATCA-7370 with Wind River WRL 4.3. Follow these steps to build Kernel and Root File System:

1. Extract the `ATCA7370-custom-layer.tgz` layer file to a location in which you want to build.
2. Adjust the `WIND_HOME` variable in env.sh to point to your WRL 4.3 installation path.
3. After you sourced the `env.sh` file, start building by calling `make` command. After successful completion of the build process, you will find the Kernel and Root File System in the `atca7370_glibc_cgl_build/export` directory.

The `atca7370` layer contains the BBS packages in the `wrlinux-4/RPMS` directory and the corresponding Makefiles to include with them into the Root File System are visible in `wrlinux-4/dist` directory.

The kernel patches are located in the `wrlinux-4/templates/board/atca7370/linux` directory.

8.1.1 Applying glibc dns Patch (CVE-2015-7547)

Prerequisite

Upgrade to WRL4.3 RCPL32.

The following patches needs to be applied, which can be obtained from the link below: (contact Wind River for latest link)
https://knowledge.windriver.com/en-us/000_Products/000/010/030/000/000_Wind_River_Linux_4.3_Security_Alert_for_glibc_getaddrinfo%28%29_stack-based_buffer_overflow_--_CVE-2015-7547

1. 0001-fix-build_libc-regression-in-4.4a-466-toolchain.patch
   This patch should be applied to RCPL32 layer in your WRL installation directory. Follow the procedure described in the above mentioned link.
2. CVE-2015-7547-wr4.patch
   To apply CVE-2015-7547-wr4.patch follow the steps described below:
   – Assuming that atca-7370 layer is extracted to `/home/atca7370.` Copy the CVE-2015-7547-wr4.patch to the extracted atca-7370 layer.
   Example: `cp CVE-2015-7547-wr4.patch /home/atca7370/`

– Edit the Makefile in atca7370 layer (`/home/atca7370/Makefile`) as follows

```
$(TARGET_ROOTFS)$(TARGET_KERNEL):
$(BUILDDIR)/scripts/autoconf.mk.orig
rm -f $(ROOTFSVERSIONFILE)
sh gen_blade_release.sh $(BUILDDIR)/scripts/ >>
$(ROOTFSVERSIONFILE)
cp $(ROOTFSVERSIONFILE) $(BUILDDIR)/filesystem/fs/etc/blade-release
cd $(BUILDDIR); make -C build glibc.patch
cp CVE-2015-7547-wr4.patch $(BUILDDIR)/build/glibc-2.11
d $(BUILDDIR)/build/glibc-2.11; patch -p2 < CVE-2015-7547-wr4.patch
make -C $(BUILDDIR) fs
```

– Do the steps 2 and 3 of section *Building Kernel and Root File System on page 87*

Using Development Kit BLSV API

9.1 Using BLSV API Development Kit

The following steps describe how to use the BSLV API Development Kit:

Extract the **DEV_KIT/blsv-api-<version>.tgz** archive to a location where you have access to WRL 4.3 installation.

The documentation of the BLSV API is available in the HTML format in the doc/html directory. From that path, open index.html file through your browser.

You also find some examples on FUPI and NMU which are available in the examples directory. To build the examples, you have to adjust your PATH variable. It must include the directories to find the cross-compilers.

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 Documentation

Document Title	Publication Number
ATCA-7370 Data Sheet	ATCA-7370-DS
ATCA-7370 Installation and Use	6806800P54
RTM-ATCA-736X Installation and Use	6806800L47

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 V.2.0	Intel Corporation, Hewlett-Packard, DEC, NEC
IPMI Platform Management FRU Information Storage Definition V1.0, September 27, 1999	Intel Corporation
PCI Industrial Computer Manufacturers Group (PICMG) Specifications http://www.picmg.org	
PICMG 3.0 Revision 2.0 Advanced Telecommunications Computing Architecture (AdvancedTCA) Base Specification	PICMG
Hardware Platform Management IPM Controller Firmware Upgrade Specification	PICMG

A.3 References

The following table lists references documentations for which the BBS software is implemented.

Table A-3 References

Document Title	Source
Embedded SW Delivery Format Description for ATCA HW Platform	Jarmo Kant, NSN

A.4 Additional Resources

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

Table A-4 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
IPM Sentry Shelf Manager User Guide	Pigeon Point Systems
OpenIPMI http://openipmi.sourceforge.net/