System Management Interface Based on HPI-B (Centellis 2000/2100/4411)

User Guide

P/N: 6806800P20F

January 2020

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 The text itself may not be published commercially in print or electronic form, edited, translated, or otherwise altered without the permission of SMART EC.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

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, System Management Interface on page 15 provides an overview on HPI-B in SMART EC AdvancedTCA systems.

Chapter 2, Software Installation and Configuration on page 19 describes how to install and configure HPI-B clients and daemons.

Chapter 3, Developing Applications on page 43 describes the necessary steps in order to build HPI-B client applications.

Chapter 4, Using HPI-B on page 45 describes in detail which HPI-B features are supported and discusses some specific features of the SMART EC HPI-B implementation.

Appendix A, Example Applications on page 97 briefly describes HPI-B example applications, which are delivered as part of the SMART EC HPI-B distribution.

Appendix B, Related Documentation on page 123 provides references to other related documentation.

Abbreviations

This document uses the following abbreviations:

Abbreviation	Description
AMC	Advanced Mezzanine Module
ATCA	Advanced Telecom Computing Architecture
FRU	Field Replaceable Unit
FUMI	Firmware Update Management Instrument
HPI	Hardware Platform Interface
IPMI	Intelligent Platform Management Interface
PICMG	PCI Industrial Computer Manufacturers Group
RDR	Resource Data Record
RPT	Resource Presence Table
SAF	Service Availability Forum
SAM	Shelf Management Alarm Module

Abbreviation	Description
SMI	System Management Interface
ShMC	Shelf Management Controller

Conventions

The following table describes the conventions used throughout this manual.

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

Notation	Description
	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

This manual has been revised and replaces all prior editions.

Part Number	Publication Date	Description
6806800P20A	April 2012	Initial release
6806800P20B	June 2012	Added the following tables: *HPI Tags on page 21* *Plugin Specific Parameters on page 27* *HPI Restart Daemon RDR on page 73* *HPI Restart Daemon State on page 74* *IPMI Command RDR on page 74* *IPMI Command Control on page 74* *IPMI Command State on page 75* *Boot Bank RDR on page 75* *Boot Bank Control on page 76* *Boot Bank State on page 76* *SOL Configuration RDR on page 77* *SOL Configuration Control on page 77* *SOL Configuration State on page 78* Updated *Limitations with Respect to HPI-B AdvancedTCA Mapping Specification on page 46*.
6806800P20C	June 2014	Re-branded to Artesyn template.
6806800P20D	September 2015	Updated tables *HPI Tags* and *Parameters of Cooling System*.
6806800P20E	October 2015	Changed the title of the manual.
6806800P20F	January 2020	Rebrand to SMART Embedded Computing template.

System Management Interface

1.1 Introduction

SMART Embedded Computing provides an SAF Hardware Platform Interface (HPI) as part of its AdvancedTCA platforms. HPI provides an industry standard interface to monitor and control highly available telecommunications system platforms. The ability to monitor and control these platforms is provided through a consistent and standard set of programmatic interfaces that are targeted for adoption by the telecom building block industry to significantly reduce product time-to-market and development costs while retaining or enhancing total system/network availability.

HPI provides the interface between the middleware software solution stack and the hardware solution stack, allowing portability of middleware software building blocks across many different hardware platforms and portability of hardware platforms across many different middleware software building blocks.

This guide describes the HPI-B implementation targeted at the SMART EC AdvancedTCA Centellis 2000/Centellis 2100 and Centellis 4411 platforms.

1.2 Standard Compliances

The SMART EC HPI-B implementation is compliant to the following standards.

Table 1-1 HPI-B Standards Supported by SMART EC HPI-B Implementation

Standard	Description
SAI-HPI-B.03.02	HPI-B base specification. It abstracts hardware platform characteristics into a data model consisting of entities and resources.
SAIM-HPI-B.03.02-xTCA	HPI-B-AdvancedTCA mapping specification. It provides a vendor independent hardware platform view of an AdvancedTCA system.
Firmware Update Management Instrument (FUMI), specified in HPI-B-03.02	Mechanism which allows remote firmware upgrade via HPI-B without service interruption. Includes a fall-back feature.

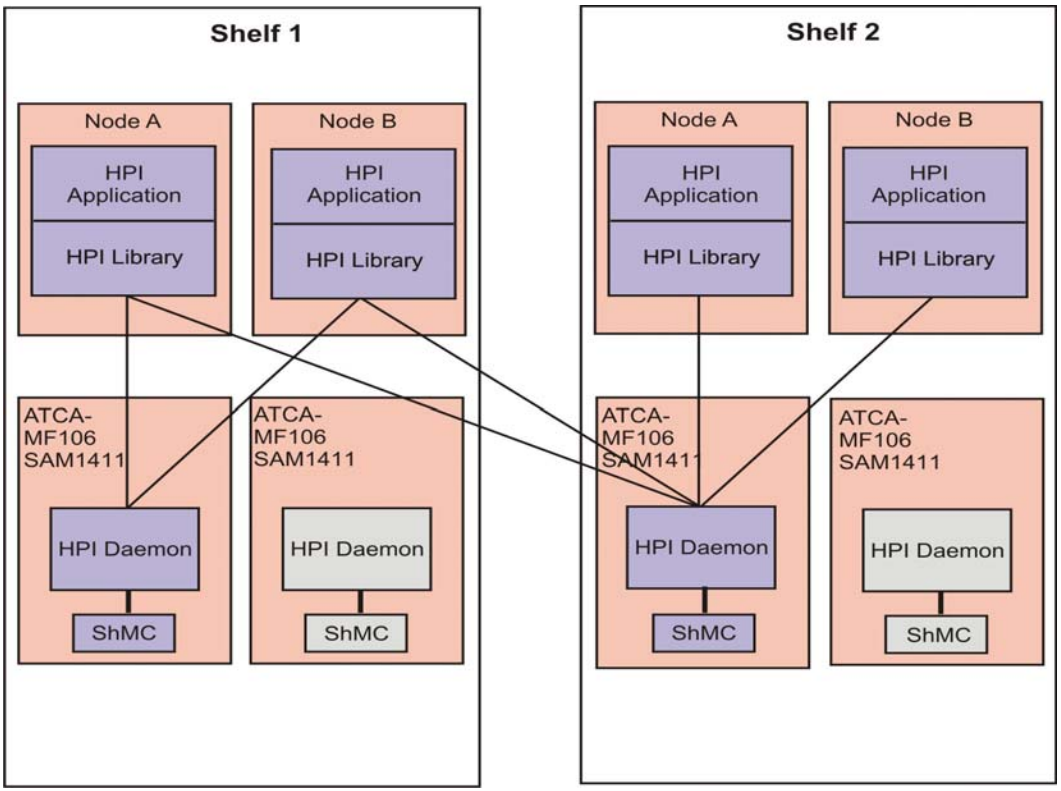
Although the standards listed above are fulfilled, some specific limitations apply. For details, refer to *Section 4.2, Limitations on page 45*.

1.3 Architecture

The SMART EC HPI implementation is provided in the form of a client-server architecture. In the Centellis 2000/Centellis 2100, the server, called HPI daemon, runs on the ATCA-MF106 and in the Centellis 4411 environment it runs on the SAM1411 blades. In the client, which is constituted by the HPI library and an application which links that library, runs on any node within a shelf. The communication between HPI daemon and HPI client is realized by an IP-based remote HPI communication protocol.

An HPI client may access one or more HPI daemons, and on the other hand, an HPI daemon may be accessed by one or more HPI clients. The following figure illustrates this.

Figure 1-1 Overview of HPI Usage in Systems



1.3.1 HPI Library

The HPI library is the primary user interface. It is intended to be used by applications that wish to control and monitor HPI managed components, such as ATCA shelves, blades and other FRUs. The HPI library is delivered as shared and as static library and has to be linked with an application.

There are two types of HPI libraries available:

- Single shelf library
- Multi shelf library

The single shelf library supports the communication with one HPI daemon only. Since it is only used internally and is not intended to be used by user applications, it will not be described any further in this manual. The multishelf library, on the other hand, supports the communication with one up to several HPI daemons. This becomes necessary if you wish to deploy redundancy in one HPI-B based shelf management system or if you wish to manage several shelves. The multishelf library is the library which you should build your applications on, it is the official interface to customer applications.

Details about supported combinations of CPU architecture/Linux distribution are given in *Chapter 2, Software Installation and Configuration on page 19*

1.3.2 HPI Daemon

As previously mentioned, the HPI daemon runs either on the ATCA-MF106, or SAM1411 blades and its main tasks are:

- Provide a single access point to control and monitor hardware components in a shelf
- Map information provided by the underlying Shelf Manager to HPI

1.4 High Availability

Both HPI daemons in a shelf are operated in cold stand-by mode. This means that while one HPI daemon is active, the other is deactivated.

A supervisor which runs on both ATCA-MF106/SAM1411 blades monitors the activity of the peer ATCA-MF106/SAM1411 and of the corresponding switch blade and it will make sure that under normal conditions the ATCA-MF106/SAM1411 corresponding to the currently active switch is also active.

If the supervisor detects that either a switchover or failover was performed between the two switch blades or if it detects that the previously active ATCA-MF106/SAM1461 has failed, it will perform a switchover of the ATCA-MF106/SAM1461 blades and the previously deactivated HPI daemon will become active. Note that a switchover between the HPI-B daemons can also be triggered by the user through OEM HPI controls.

No data is replicated between the two HPI daemons. The only exception are HPI resource IDs. They are written into a file and read in again by the newly started up HPI daemon after a failover. This file is replicated from the active Shelf Manager to the standby when HPI daemon starts up or when a resource is added/deleted from the file, while the daemon is running.

An HPI application that wishes to access an HPI daemon has two options to do this: access the HPI daemon via the physical or via the virtual IP address assigned to the ATCA-MF106/SAM1411 blades. For further details refer to *Section 2.3.2, Configuring HPI Clients on page 34*.

Software Installation and Configuration

2.1 Overview

This section describes how to set-up HPI-B daemons and clients and how to install all files needed to develop client applications. Generally, all files are delivered in the form of RPMs. The content of an RPM is reflected in its naming scheme.

The following table lists all RPM files needed to run clients and for their development. The placeholder "architecture" stands for the supported CPU architecture of the respective blade where the client is to run and can be any of the following:

- x86
  Intel IA-32 bit blades
- x86_64
  Intel IA-64-bit blades
- ppc_<xxx>
  PowerPC based blades

Table 2-1 RPM Files for HPI-B Clients

RPM File Name	Description
`bbs-hpib-<version>-1.<architecture>-<distribution>-<OS>.rpm`	This RPM is the HPI-B base package. It contains shared libraries to be used by HPI-B clients and daemons, as well as compiled example applications and client configuration files. This package is required both by HPI-B daemons and clients.
`bbs-hpib-daemon-<version>-1.<architecture>-<distribution>-<OS>.rpm`	This RPM contains all files which are related to the HPI-B daemon: executables, libraries and configuration files.

2.2 Installing and Configuring the HPI-B Daemon

This section describes how to install and configure the HPI-B daemon.

2.2.1 Installing the HPI-B Daemon

The HPI-B daemon is part of the ATCA-MF106/SAM1411 Linux root file system and is, together with the root file system, preinstalled on the ATCA-MF106/SAM1411.

You can upgrade the ATCA-MF106/SAM1411 software manually. The respective procedure is described in the *ATCA-MF106 Installation and Use* and *SAM1411 Installation and Use*.

The following table lists all files/directories on the ATCA-MF106/SAM1411 which are related to the HPI-B daemon.

Table 2-2 Overview of HPI-B Directories and Files on ATCA-MF106/SAM1411

Directory/Files	Description
`/usr/bin`	Contains HPI-B daemon files.
`/etc/bbs-hpib`	Contains configuration files used to configure the HPI daemon as well as client libraries. See *Configuring the HPI Daemon on page 21* for details on configuring the HPI daemon.
`/etc/init.d/bbs-hpib`	Daemon start/stop script.

2.2.2 Configuring the HPI Daemon

At start-up, each HPI daemon reads the following configuration file: `/etc/bbs-hpib/bbs-hpib.conf`.

Generally, you should not change any values. All settings are set to appropriate values.

Table 2-3 HPI Tags

Tag	Description	Allowed Values	Default in Config. File	Built-in Default
`OPENHPI_INTERACTIVE`	interactive daemon with command line interface	`"YES | NO"`	`[NO]` - non-interactive `YES` - interactive	`NO`
`OPENHPI_CONFIG_ REPLICATION`	replicate this configuration file to standby	`"YES | NO"`	`[NO]` - no replication `YES` - replicate	`NO`
`OPENHPI_DEL_SIZE_LIMIT`	sets the maximum size for the domain event log	`<0…N>`	`[1000]` - max size is 1000 `0` - unlimited	`10000`
`OPENHPI_ON_EP`	Sets the entity path on which the application is running. This entity path will be returned when `SaHpiResourceIdGet()` is called	`"{ADVANCEDTCA_CHASSIS,<shelf_id>}"`	`[{ADVANCEDTCA_CHASSIS,0}]`	`{ADVANCEDTCA_CHASSIS,0}`
`OPENHPI_EVT_QUEUE_LIMIT`	sets the maximum number of events that are allowed in the session's event queue	`<0…N>`	`[1000]` - max size is 1000 `0` - unlimited	`10000`
`OPENHPI_DEL_SAVE`	sets whether the domain event log will be persisted to disk or not. The event log is written to `OPENHPI_VARPATH` value	`"YES | NO"`	`[NO]` - not saved `[YES]` - saved to `OPENHPI_VARPATH`	`NO`

Table 2-3 HPI Tags (continued)

Tag	Description	Allowed Values	Default in Config. File	Built-in Default
OPENHPI_DAT_SIZE_LIMIT	Sets the maximum size for the alarm table	<0…N>	[1000] - max size is 1000 0 - unlimited	1000
OPENHPI_DAT_USER_LIMIT	sets the maximum number of user type alarm entries allowed in the alarm table	<0…N>	[100] - max size is 100	100
OPENHPI_LOG_ON_SEV	sets the lowest severity level an event must meet to be logged in the domain event log	(Order: Highest to Lowest) "CRITICAL | MAJOR | MINOR | INFORMATIONAL| OK | DEBUG"	[INFORMATIONAL] – all events with INFORMATIONAL and above severity are logged CRITICAL | MAJOR | MINOR |OK | DEBUG – Events with given severity and above that are logged.	INFORMATIONAL
OPENHPI_AUTO_INSERT_ TIMEOUT	Sets the default auto-insert timeout. If the resource is not shelf manager auto activated, it starts an auto insert timer, otherwise the resource stays in insertion state until user explicitly issues saHpiResourceActiveSet API. (This information is taken from PICMG Fru Activation and Power Descriptor record)	"block | <0…N>"	[block] - auto insertion policy never executed until user issues saHpiResourceActiveSet() <0…N> Any positive value would give timeout in nanoseconds.	block

Table 2-3 HPI Tags (continued)

Tag	Description	Allowed Values	Default in Config. File	Built-in Default
OPENHPI_AUTO_EXTRACT_TIMEOUT	Sets the default autoextract timeout. If the resource is not shelf manager auto deactivated, it starts an auto extract timer otherwise the resource stays in extraction state until user explicitly issues saHpiResourceInactiveSet API (This information is taken from PICMG Fru Activation and Power Descriptor record)	"block | <0…N>"	[60000000000] - 60000 milli-seconds block - auto extraction policy never executed until user issues saHpiResourceInactiveSet()	block
OPENHPI_PATH	Search path for plugins	Any valid search path	[/opt/bladeservices/lib/bbs-hpib]	/opt/bladeservices/lib/bbs-hpib
OPENHPI_VARPATH	path to store user events	Any valid path	[/var/lib/bbs-hpib] - If OPENHPI_DEL_SAVE is YES, then user events are stored in this path by default	/var/lib/bbs-hpib
OPENHPI_CONNECTION_PING_INTERVAL	Connection ping interval for the clients	<0…N>	[500] - 500 milliseconds is the default ping interval	1000 ms
OPENHPI_CONNECTION_PING_TIMEOUT	Connection timeout for clients	<0…N>	[5000] - 5000 milliseconds is the default timeout	5000

Table 2-3 HPI Tags (continued)

Tag	Description	Allowed Values	Default in Config. File	Built-in Default
OPENHPID_MAX_NUMBER_OF_CONNECTIONS	maximum number of HPI clients	<0…N>	[30] - when the client connections reach 30, any further connections are rejected	30
OPENHPI_CONNECT_LOCAL_ONLY	Secure connections, local host connection only	"YES | NO"	[NO]– Clients can connect from any outside host that is reachable YES – Clients can connect only through loopback address	NO
OPENHPI_START_SNMP	Start SNMP after HPI-B daemon discovery	"YES | NO"	[YES] – snmp is started after HPI-B discovery NO – snmp is not started	YES
OPENHPI_SNMP_SCRIPT	SNMP script to run if OPENHPI_START_SNMP is YES	<Path to snmp script>	[/etc/init.d/hpibsnmp] – default snmp script to be executed	/etc/init.d/hpibsnmp
OPENHPI_DAEMON_PORT	Openhpi daemon port	port number	[4743] - default port for HPI-B daemon	4743
OPENHPI_LOGFILE	Path and Name of the log file	Any valid name	[/var/log/bbs-hpib] - log file name starts with bbs-hpib Ex: bbs-hpib00.log	/var/log/bbs-hpib
OPENHPI_LOGFILE_MAX	Sets the maximum number of log files to be created	<0…N>	[3] - By default, 3 log files are created	2

Table 2-3 HPI Tags (continued)

Tag	Description	Allowed Values	Default in Config. File	Built-in Default
OPENHPI_LOGFILE_MAX_SIZE	Log file maximum size	<0…N>	[500000] - 500 KB is default maximum size	500000
OPENHPI_LOGFLAGS	Flags that control logging	"stdout | stderr | file | simplefile | syslog | prefix | threadid"	[file|prefix] - log into a file with prefix(<time> [threadid] <log level> {facility}) in each line logged Stdout - log to standard output Stderr - standard error Simplefile - only one file for logging and its name is OPENHPI_LOGFILE Syslog - log into syslog Threadid - logging of threadid is optional. With this option, threadid is also logged	stderr
OPENHPI_LOG_ERROR_ FACILITIES	Facilities that should be logged at error level	List of facilities mentioned in the config file	[all] - By default, all the facilities are logged at error level To mention more than one facility, use | Ex: core|daemon	all
OPENHPI_LOG_WARNING_ FACILITIES	Facilities that should be logged at warning level	List of facilities mentioned in the config file	[all] - By default, all the facilities are logged at warning level	all

Table 2-3 HPI Tags (continued)

Tag	Description	Allowed Values	Default in Config. File	Built-in Default
OPENHPI_LOG_INFO_ FACILITIES	Facilities that should be logged at info level	List of facilities mentioned in the config file	[all] - By default, all the facilities are logged at info level	all
OPENHPI_LOG_DEBUG_ FACILITIES	Facilities that should be logged at info level	List of facilities mentioned in the config file	[] - By default, no facilities are logged at debug level User has two options to change facilities at any log level: 1.Modifying config file - the changes are persistent and are reflected after restarting the hpib daemon 2.Using hpilog application - changes take effect immediately but are not persistent and removed once the daemon is restarted	""
OPENHPI_LOGTHREAD	Separate thread for logging	"0 | 1"	[1] - start a separate thread for logging [0] - no separate thread If tag is not mentioned, no thread is started	0

Table 2-3 HPI Tags (continued)

Tag	Description	Allowed Values	Default in Config. File	Built-in Default
OPENHPI_CRASHFILE	Crash handler log file	Any valid path	[/var/log/bbs-hpibdcrash.log] - default file into which stack trace of all the running threads is dumped when a crash occurs	/var/log/bbs-hpibdcrash.log
OPENHPI_HA_IP	IP address for redundant hpi-b daemon	HA IP	[] - empty	0.0.0.0
OPENHPI_HA_HWADDR_OFFSET	Hardware address of the first shelf manager	<8,9>	[8] - default value	8
OPENHPI_HA_NUM_PORTS	Number of ports to open per IP	<1…N>	[1] - only 1 port opened	5
OPENHPI_LOG_SEL_LOGS_IN_SYSLOG	Logging SEL logs into syslog in the same format as in Shelf Manager's SEL	"true | false"	[false] - not logged true - logged into syslog	false

Table 2-4 Plugin Specific Parameters

Tag	Description	Allowed values	Default in Config. file	Built-in Default
entity_root	Root entity value	{<chassis description> , <0…N>}	[{ADVANCEDTCA_CHASSIS,0}] - Any entity's path would be starting with the above string	Can be evaluated in 2 ways" 1.Uses shelf address as the integer 2.If step 1 fails, then uses 0

Table 2-4 Plugin Specific Parameters (continued)

Tag	Description	Allowed values	Default in Config. file	Built-in Default
`name`	Interface name for ipmi connection	`"smi | lan"`	`[smi]`– connection over SMI interface `lan` – connection over LAN	`"smi"` - if HPI is running on shelf manager `"lan"` - if HPI is running on management controllers other than shelf manager
`DomainTag`	Tag for the domain	Any valid tag	`[ATCA Shelf]`	`ATCA Shelf`
`sel_filename`	System event log file name	Any valid file name	`[/var/lib/bbs-hpib/shelf_sel.db]` Commented out by default	NA
`max_number_of_sel_entries`	Maximum number of SEL entries	`<0…N>`	`[100]` Commented out by default	NA
`IpmiConnectionTimeout`	IPMI Connection timeout	`<0…N> msecs`	`[1500]` - 1500 milliseconds	NA
`AtcaConnectionTimeout`	IPMI driver/smi timeout	`<0…N> msecs`	`[1500]` - 1500 milliseconds	`1000`
`MaxOutstanding`	Maximum number of outstanding IPMI commands	`<0…N>`	`[10]` - 10 commands by default. After this max is reached, any other request is awaited.	`0`

Table 2-4 Plugin Specific Parameters (continued)

Tag	Description	Allowed values	Default in Config. file	Built-in Default
PtpEKeyingMode	E-keying mode	<0,1,2>	[2] – Full e-keying. To do the e-key on both local and remote ports 1 – Simple e-keying. To do the e-key on only local ports 0 – Disable the e-keying	Full e-keying
ResourcePowerOffTimeout	A timeout after which the resource FRU Power State will be set to initial state. This timer is started soon after executing the Power Off/Power Cycle/hotswap operations on the resource.	<0…N>	[60000] - 60000 ms	60000
ResourcePowerOnTimeout	A timeout after which the resource FRU Power State will be set to initial state. This timer is started soon after executing the Power On/hotswap operations on the resource.	<0…N>	[20000] - 20000 ms	20000
clear_sel	Clear the SEL after reading events from it	"yes | no"	[yes] – clear the SEL No – Do not clear the SEL	yes
RmcpEnable	Enables/disables the RMCP server	"true | false"	[true] – enables the rmcp server false – disables the rmcp server	false

Table 2-4 Plugin Specific Parameters (continued)

Tag	Description	Allowed values	Default in Config. file	Built-in Default
RmcpPort	Port on which RMCP service runs	Standard port	[623]	623
RmcpMaxConnection	Maximum number of connections active. After this count is reached, any new connections are rejected	<0…N>	[10] – 10 connections	10
RmcpConnectionTimeout	RMCP connection timeout	<0…N>	[60000] - 60000 msecs	60000 msecs
RmcpSessionTimeout	RMCP Session Timeout	<0…N>	[60000] - 60000 msecs	60000 msecs
RmcpPefAlerting	RMCP PEF alerting	"true | false"	[true] - supports PEF alerting false - does not support	false
RmcpPerMsgAuth	RMCP per msg authentication	"true | false"	[true] - authentication supported	false
RmcpPrivLimit	Privilege needed for using RMCP service	"callback | user | operator | admin"	[admin] - highest privilege	admin
RmcpAllowedAuthsCallback	Allowed authentication mechanisms for privilege level 'callback	"none | md2 | md5 | straight"	"[none, md2, md5, straight]"	none
RmcpAllowedAuthsUser	Allowed authentication mechanisms for privilege level 'user	"none | md2 | md5 | straight"	"[none, md2, md5, straight]"	none

Table 2-4 Plugin Specific Parameters (continued)

Tag	Description	Allowed values	Default in Config. file	Built-in Default
RmcpAllowedAuthsOperator	Allowed authentication mechanisms for privilege level 'operator	"none | md2 | md5 | straight"	"[none, md2, md5, straight]"	none
RmcpAllowedAuthsAdmin	Allowed authentication mechanisms for privilege level 'operator	"none | md2 | md5 | straight"	"[none, md2, md5, straight]"	none
RmcpGuid	RMCP group user id	16 hex bytes	[01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16] - default guid	All zeros (16 bytes)
RmcpUser[X] X is an integer Ex: RmcpUser2, RmcpUser3, RmcpUser4 etc	RMCP users	<U><P><Pr><MS><A> U - username P - Password Pr - Privilege MS - Maximum number of sessions A - Authentication	[RmcpUser[2,3,4,5,6]] are available in config file Ex: RmcpUser6 = "Administrator Administrator admin 64 none md2 md5 straight"	NA
SelMaxEntries	Maximum number of entries in the SEL	<0…N>	[240]	1024
SelCleanPolicy	Policy to retain/remove the SEL entries once maximum is reached	"true | false"	[false] - Will not add the SEL entries to the SEL, once the SEL is full true - Removes the oldest sel entry when the entries reaches the SelMaxEntries	false

Table 2-4 Plugin Specific Parameters (continued)

Tag	Description	Allowed values	Default in Config. file	Built-in Default
Script<old_state><new_state> Ex: ScriptInitializeAloneScriptInitializeActiveetc	State change script that gets executed when the HPI state changes from old_state to new_state. This script is responsible for setting virtual active IPs in response to the state change	"/opt/bladeservices/etc/bbshpib/ StateChange"	[/opt/bladeservices/etc/bbs-hpib/StateChange]	/opt/bladeservices/etc/bbs-hpib/StateChange
FumiEnabled	Enabling FUMI plugin Note: All the following FUMI related tags are not applicable when FumiEnabled is set to false	"true | false"	[true] - Load the FUMI plugin, supported on ATCAM100/ ATCA-MF105 false - FUMI plugin is not supported for this ATCA board	true
FumiTimeout	FUMI timeout	<100..900000>	[300000] - 300 ms	300 ms
FumiRemoveLocalImageAfterUpgrade	Remove/Retain local image after upgrade	"yes | no | true | false | on | off"	<true|on|yes> – remove image after upgrade	true
FumiPreserveOemRecords	Preserve/Don't preserve OEM records after upgrade	"yes | no | true | false | on | off"	<true|on|yes> - do not preserve <false|off|no> - preserve Default value is true	true
PowerOnSequencing	To enable/disable power on sequencing. Specific feature of M100 shelf manager	<0,1>	[1] – enable 0 – disable	1

Table 2-4 Plugin Specific Parameters (continued)

Tag	Description	Allowed values	Default in Config. file	Built-in Default
PowerBudgetComputation	Power budget computation	<0,1>	[1] - Follow the Power Budget algorithm before allocating the power 0 - No power budget algorithm is followed	1
HPI_OEM_DISCRETE_SENSOR_DEF_SEVERITY	HPI-B Daemon will set the severity for the HPI events of non-threshold sensors based on the value configured for this tag. Note: If this tag is set to 3, then change the OPENHPI_LOG_ON_SEV tag to "INFORMATIONAL" instead of "MINOR" so that it logs events of "INFORMATIONAL" severity also into DEL.	<0,1,2,3>	[3] - informational 0 - critical 1 - major 2 - minor Commented by default, Please uncomment in the config file for the tag to take effect.	0 - critical daemon will assign the default "critical" severity for the events of non-threshold sensors (discrete/OEM)

SOL and cooling related options are not described in this table. For more information on this, refer to *Section 4.6, Cooling Management on page 78* **and** *Section 4.7, Serial Over LAN Configuration on page 87*.

2.3 Setting Up HPI Clients

This section describes how to install/configure HPI clients on node blades.

2.3.1 Installing HPI Clients

Procedure

In order to install an HPI-B client package on a node blade, proceed as follows.

1. Connect to the node blade where you wish to install the HPI-B package.
2. Copy the RPM file that you wish to install to the node blade. Refer to *Table 2-1 on page 19* for details on available RPM files for your particular node blade.
3. If applicable, enter **`rpm -e <Old HPI-B client package name>`** to uninstall an already installed client package.
4. Enter **`rpm -i <New HPI-B client RPM file name>`**
   This installs the HPI-B library package.

The following table lists the directories and their content available on the blade after installing the package.

Table 2-5 Overview of HPI-B Directories and Files on Node Blades

Directory	Description
`/opt/bladeservices/lib`	Contains example applications and shared libraries needed to run clients
`/opt/bladeservices/bin`	Contains precompiled example applications. They are controlled via the command line and can easily be identified through the prefix "hpi" in their names. Use the -h option to display supported command line parameters.
`/opt/bladeservices/bbs-hpib`	Contains configuration files used to configure HPI client libraries. See *Configuring HPI Clients on page 34* for details.

2.3.2 Configuring HPI Clients

Before running your client, you need to configure the multishelf library:
`/opt/bladeservices/etc/bbs-hpib/bbs-hpibmultishelf.conf`

Most of the entries should be left as they are. They have been set to values that are appropriate for most operations. The only settings that need to be adapted are those which are related to the HPI daemons that the multishelf library wishes to access. The following table shows the expected syntax of such entries.

Table 2-6 Multishelf Library Configuration File - HPI Daemon Entries

Entry	Description
`[Shelf<Domain Name>]`	This indicates the start of the definition of an HPI daemon. The chosen domain name appears as name of the Shelf Management Resource and is used as Domain Tag. See *Shelf Management Resource on page 54*.
`Daemon=<IP address of HPI daemon>`	This is the IP address used to access an HPI daemon. Details are given in the following subsections.
`Port=<port number>`	This is the port number. The HPI daemon uses 4743 as port.

In the following sections we will describe some typical configuration scenarios in Centellis 2000/2100/4411 systems.

Typically you will want to specify both HPI daemons in the configuration file so that in case of a failure of the currently active HPI daemon, you can continue with the previously deactivated daemon. There are two ways to specify the HPI daemons in the configuration file. You can use either the virtual or the physical IP addresses of the corresponding ATCA-MF106/SAM1411 blades. Both options are explained in the following section.

2.3.2.1 Accessing the HPI Daemons via Physical IP Addresses

The following table lists the physical IP addresses assigned to the ATCA-MF106 and SAM1411 blades.

Table 2-7 Physical IP Addresses of ATCA-MF106/SAM1411 Blades

System	Viewed from the front of the shelf	
	Physical IP Address of Left ATCA-MF106/SAM1411	**Physical IP Address of Right ATCA-MF106/SAM1411**
Centellis 2000/ Centellis 2100	172.17.<shelfID>.8 (connection to left switch) 172.18.<shelfID>.8 (connection to right switch) Access is possible either via the backplane or via the ATCA-MF106 face plate connector UPLINK ETH, which provides an uplink to the system's base interface.	172.17.<shelfID>.9 (connection to left switch) 172.18.<shelfID>.9 (connection to right switch) Access is possible either via the backplane or via the ATCA-MF106 face plate connector UPLINK ETH, which provides an uplink to the system's base interface.
Centellis 4411	192.168.21.8 (connection to left switch) 192.168.22.8 (connection to right switch)	192.68.21.9 (connection to left switch) 192.168.22.9 (connection to right switch)

If you want to access the HPI daemons via the physical IP addresses of the ATCA-MF106/SAM1411 blades, then you need to specify two physical IP addresses corresponding to the two ATCA-MF106/SAM1411 blades in your configuration file. Therefore, in a Centellis 4411 system for example, the configuration file could contain the following two entries.

```
[ShelfFrodo1]
Daemon=192.168.21.8
Port=4743

[ShelfFrodo2]
Daemon=192.168.22.9
Port=4743
```

At start-up, your application would typically first determine which HPI daemon is currently active (for example by opening a session with the daemon), and then establish a connection with the active daemon. In order to monitor the status of an established connection, you can use OEM HPI sensors. Refer to *Section 4.4, Working with the Multishelf Library on page 48* for further details.

If the application detects, that the connection to the active daemon got lost, it would establish a connection to the previously deactivated daemon and continue with that daemon. Since the HPI resource IDs were preserved in a file, the application can continue with the same resource IDs. Apart from resource IDs, however, no other data is replicated and therefore preserved.

The activation of a previously deactivated HPI daemon may take up to several minutes. Your application should take this into account when establishing a connection with a previously deactivated HPI daemon.

2.3.2.2 Accessing the HPI Daemons via Virtual IP Addresses

In Centellis 2000/2100/4411 systems, you can access the ATCA-MF106/SAM1411 blades via virtual IP addresses. One virtual IP address is automatically linked to the currently active ATCA-MF106/SAM1411 blade. The following table shows which IP address this is for the different systems.

Table 2-8 Virtual IP Addresses Corresponding to Active ATCA-MF106/SAM1411 Blade

System	Virtual IP of Active ATCA-MF106/SAM1411
Centellis 2000/ Centellis 2100	172.16.<shelfID>.171 Access is possible either via the backplane or via the ATCA-MF106 face plate connector UPLINK ETH, which provides an uplink to the system's base interface.
Centellis 4411	192.168.20.171

If you want to access the HPI daemon via the virtual IP address, then you need to specify the respective IP address in your configuration file. In a Centellis 4411 context for example, an entry might look like this:

```
[ShelfFrodo]
Daemon=192.168.20.171
Port=4743
```

It is important to note that the HPI-B library does not automatically switch between the two interfaces towards the two switch blades when a switchover or failover between the two switches occurs.

This becomes a problem in situations when one switch blade fails and the connection to the currently active ATCA-MF106/SAM1411 is interrupted. In such situations it becomes necessary that an underlying mechanism, such as for example a bonding driver, switches to the second interface and so that the active ATCA-MF106/SAM1411 can be accessed via the second interface.

Before accessing the HPI daemon via the virtual IP address, you must make sure that such a switching mechanism exists. Refer also to the respective system guides for information about the system network topologies.

Once the previously described switching mechanism exists, your application can access the virtual IP address and open a connection. Since the virtual IP address is used, the application would automatically be connected with the active HPI daemon. If the application detects that the connection to the HPI daemon gets lost, it would reconnect to the same virtual IP address again and this time the application would automatically be connected to the newly started up, previously deactivated HPI daemon. Since the HPI resource IDs were preserved in a file, the application can continue with the same resource IDs. Apart from resource IDs, however, no other data is replicated and therefore preserved.

2.4 Installing and Configuring an SNMP Agent

The SNMP agent is intended to run on the ATCA-MF106/SAM1411 and is bundled with the ATCA-MF106/SAM1411 Basic Blade Services (BBS) software. This means that under normal conditions there is no need to install the SNMP software manually. If for some reason you need to manually install an SNMP agent on an ATCA-MF106/SAM1411, follow the installation instructions given below.

You may want to change the default configuration of an SNMP agent. All necessary steps are described below as well.

For more information on using SNMP agent, refer to *HPI-B Subagent User Guide*.

2.4.1 Installing an SNMP Agent

Procedure

The SNMP agent requires the following software images.

Table 2-9 SNMP Agent - Required Software Images

Software Image	Description
SNMP subagent package	This is an RPM with the following file name scheme: `bbs-hpib-snmp-<version>-1.<architecture>-<distribution>-<OS>.rpm` It can be obtained from SMART EC and contains SNMP agent binaries, config files as well as MIB files.
net-snmp 5.1.2 or higher	This is an open source suite which contains files needed by the SNMP master agent as well as useful SNMP command line tools. It is part of the ATCA-MF106/SAM1411 BBS software and therefore there should be no need to install it manually.
HPI-B client package	Since the SNMP subagent is an HPI-B client application, it relies on the HPI-B client package to be installed. For details about how to install and configure an HPI-B client package, refer to *Setting Up HPI Clients on page 34*.

Make sure that the net-snmp 5.1.2 or higher package as well as the HPI-B client base package are installed on the ATCA-MF106/SAM1411 where you wish to install the SNMP agent. In order to install the SNMP subagent package, proceed as follows:

1. Connect to the ATCA-MF106/SAM1411 blade where you wish to install the HPI-B SNMP subagent package.
2. Copy the package file to the ATCA-MF106/SAM1411.
3. If applicable, enter **`rpm -e <Old HPI-B SNMP subagent package name>`** to uninstall an existing SNMP subagent package
4. Enter **`rpm -i <New HPI-B SNMP subagent package name>`** to install the new package.

After the installation of all required packages, the SNMP agent related files can be found in the following directories on the ATCA-MF106/SAM1411:

Table 2-10 Overview of SNMP Agent Related Directories and Files

Directory	Description
`/opt/bladeservices/bin`	Contains HPI-B sub-agent daemon binaries
`/opt/bladeservices/share/snmp/mibs`	Contains MIB file SAI-HPI-SNMP-MIB-B.01.01.mib used by HPI-B SNMP subagent.

Table 2-10 Overview of SNMP Agent Related Directories and Files (continued)

Directory	Description
`/opt/bladeservices/etc/snmp`	Configuration file for Hpi-B subagent
`/usr/share/snmp/mibs`	Contains mib file SAF-TC-MIB.mib used by HPI-B SNMP subagent and Saf application
`/etc/init.d`	Contains HPI-B SNMP daemon start/stop scripts

2.4.2 Configuring an SNMP Agent

Once the SNMP agent files are installed and the single shelf library used by the SNMP subagent is configured appropriately as described in the previous sections, you need to configure the SNMP master agent as follows.

Configuring the SNMP Master Agent

The following configurations can be grouped into the following categories:

- General configuration
- SNMP V2 and V3 related configuration
- Trap message configuration

1. Open the SNMP master agent configuration file `/etc/snmp/snmpd.conf` in a text editor
2. Make sure that the file contains the following entries:

   ```
   master agentx
   AgentXTimeout 600
   AgentXRetries 20
   ```

   The values specified for `AgentXTimeout` and `AgentXRetries` should have at least the values 600 and 20.
3. In order to configure the SNMP master agent for SNMP V2, add the following line to `snmpd.conf`:

   ```
   com2sec notConfigure default rwcommunity
   rwcommunity public
   ```

   To test the SNMP V2 configuration, save the configuration file and perform a SNMP walk by entering the following command at the ATCA-MF106/SAM1411 command line:

```
 #  snmpwalk -v 2c -c public localhost -m
/opt/bladeservices/share/snmp/mibs/SAI-HPI-SNMP-MIB-B.01.01.mib
hpib0101
```

4. In order to configure the SNMP master agent for SNMP V3, add the following line to snmpd.conf:

   ```
   createUser LocalUser MD5 "LocalUserPassword" DES
   "localUserPassword"
   rwuser LocalUser
   ```

   To test the SNMP V3 configuration, save the configuration file and perform a SNMP walk by entering the following command at the ATCA-MF106/SAM1411 command line:

   snmpwalk -v3 -n "" -l AuthPriv -u "LocalUser" -a MD5-A "LocalUserPassword" -x DES -X "LocalUserPassword" -m /opt/bladeservices/share/snmp/mibs/SAI-HPI-SNMP-MIB-B.01.01.mib localhost hpib0101

5. If you are using SNMP V2 and want to enable SNMP traps to be sent to a host, enter the following line to snmpd.conf

   ```
   trap2sink <Host IP> public
   ```

   <Host IP> is the IP address of the host where you want the traps to be sent to.

6. If you are using SNMP V3 and want to enable SNMP traps to be sent to a host, enter the following line to snmpd.conf

   ```
   trapsess -e <Destination snmp v3 device engine-id>  -v 3   -n "" -
   l <Authorization/Encryption detail>  -u <User-name> -a MD5 -A
   "<Authorization-Password>" -x DES -X "<Encryption Password>"
   <Dest-Ip>:162
   ```

   This creates a session with the host where the traps are to be sent to (as required by SNMP V3). An example entry might look as follows:

   ```
   trapsess -e 0x800000000102030405  -v 3   -n "" -l AuthPriv  -u
   LocalUser  -a MD5 -A "LocalUserPassword" -x DES -X
   "LocalUserPassword"  192.168.21.1:162
   ```

Developing Applications

3.1 Overview

This chapter describes how to develop applications that make use of the HPI-B library.

Depending on the CPU architecture of the target system where you want to run your HPI-B application and on the operating system, different RPM files are delivered which contain include files and static libraries needed for the application development. The naming scheme used for these files is as follows: `bbs-hpib-devel-<version>-1.<architecture>-<distribution>-<os>.rpm`

In order to run your HPI-B clients, you furthermore need to install the HPI-B client base package applicable to the blade where the client is running. See *Setting Up HPI Clients on page 34*.

The HPI-B client base package contains compiled example applications which illustrate the use of HPI-B controls. For these example applications the source codes and an example make file are available as well. You may want to use the source code and the make file as a starting point for developing your own applications. For further details refer to *Appendix A, Example Applications on page 97*.

3.2 Building the Application

If your development system is based on the same operating system/CPU architecture environment as the target system, then you can simply install the RPM files on the target system. If the development system is based on another operating system/CPU architecture environment and you consequently intend to do cross-compilation, then the RPM files should be converted to the `cpio` format and then extracted, using the standard Linux `rpm2cpio` tool.

In order to do this, you would for example enter the following at the command prompt:

1. **`cd <working directory>`**
2. **`rpm2cpio <rpm file> | cpio -id`**

After extracting the RPM or CPIO files, you obtain the following directories with the following contents.

Table 3-1 Development RPMs - Directory Structure

Directory	Content
`/opt/bladeservices/include/bbs-hpib`	Include files
`/opt/bladeservices/lib` or `/opt/bladeservices/lib64`	Static libraries

Using HPI-B

4.1 Overview

This chapter provides information which is necessary when writing applications that are based on the SMART Embedded Computing HPI-B distribution. It lists limitations with respect to the HPI-B specification and describes extensions which were added by SMART EC.

4.2 Limitations

This section describes those HPI-B features which the SMART EC HPI-B implementation does not support.

4.2.1 Limitations with Respect to HPI-B Base Specification

The following limitations apply with respect to the compatibility with the SAI-HPI-B.01.02 specification. Note that these limitations apply to the current and also to all future SMART EC HPI-B releases. There are no plans to implement these features in the future.

- Limited `saHpiIdrAreaAdd()` call
  If the space is available, the function `saHpiIdrAreaAdd()` adds an OEM Inventory Area including two pre-defined fields as multi-record with a maximum size of 255 Byte. The first one is a read-only field containing the ManufacturerID (3 bytes). The second field (252 bytes) can be updated by the user. The SMART EC HPI-B implementation of `saHpiIdrAreaAdd()` does not support the creation of other types of Inventory Areas.
- Limited `saHpiIdrAreaAddById()` call
  If the space is available, the function `saHpiIdrAreaAddById()` adds an OEM Inventory Area with a specified area Id including two pre-defined fields as multi-record with a maximum size of 255 Byte. The first one is a read-only field containing the ManufacturerID (3 bytes). The second field (252 bytes) can be updated by the user. The SMART EC HPI-B implementation of `saHpiIdrAreaAddById()` does not support addition of an area with id zero or the creation of other types of Inventory Areas.
- Limited `saHpiIdrAreaDelete()` call
  SMART EC HPI-B does not allow deleting Inventory Area with `saHpiIdrAreaDelete()`, except OEM Multi Records not specified by the PICMG ATCA and AMC specifications.
- Limited `saHpiHotSwapIndicatorStateSet()` call
  The API is not supported even though the Hotswap Indicator Flag is True for the resource.

- Limited `SaHpiIdrFieldSet()` call
  In case of OEM multi records, SMART EC HPI-B always assumes DataType=SAHPI_TL_TYPE_BINARY and Language=SAHPI_LANG_UNDEF for the field on which the API is being called.
- Limited `saHpiResourceResetStateSet()` call
  SMART EC HPI-B does not support Reset Action SAHPI_RESET_ASSERT.
- Unsupported `saHpiParamControl()` call
  SMART EC HPI-B does not support `saHpiParamControl()`.
- Unsupported resource event log
  SMART EC HPI-B does not support resource event logs.
- Unsupported Unicode character set
  SMART EC HPI-B does not support the Unicode character set.
- Unsupported annunciator functionality
  SMART EC HPI-B does not support annunciator functionality. Our platforms do not have these features.
- Unsupported `SaHpiIdrFieldAdd()` and `SaHpiIdrFieldAddById()` call
  SMART EC HPI-B does not allow adding field in an inventory area with the above APIs.
  To add a new field, add a new OEM area and use the second field of the area.
- Optional APIs `saHpiInitialize()` and `saHpiFinalize()` are not supported.
- FUMI explicit banks are not supported, only logical bank is used for FUMI APIs.
- APIs `saHpiFumiBackupStart()`, `saHpiFumiBankCopyStart()`, `saHpiBankBootOrderstart()`, `saHpiTargetVerifyStart()`, and `saHpiFumiRollbackstart()` are not supported.
- FUMI APIs `saHpiFumiUpgradeCancel()` and `saHpiFumiCleanup()` cannot preempt the image activation process as it can result in image corruption. We do not stop the activation process once it is started.

4.2.2 Limitations with Respect to HPI-B AdvancedTCA Mapping Specification

The following limitations apply with respect to the compatibility with the HPI-B-AdvancedTCA mapping specification SAIM-HPI-B.03.02-xTCA. Note that these limitations apply to the current and also to all future SMART EC HPI-B releases. There are no plans to implement these features in the future.

- Only physical slot numbers are supported
  SMART EC HPI-B only supports physical slot numbers in entity paths

- Only "shall" and "should" requirements are supported
  SMART EC HPI-B only supports the "shall" and "should" requirements of the HPI-to-AdvancedTCA mapping specification SAIM-HPI-B.03.02-xTCA. "May" requirements might not be supported.
- Following Inventory records are not exposed to the HPI users through the HPI Inventory APIs:
  - Radial IPMB-0 Link Mapping Record (PICMG Record Id 15h), version 1
  - PICMG Form Factor Information Record (PICMG Record Id 2Eh)
  - Shelf Fan Geography Record (PICMG Record Id 1Bh)
  - Shelf Configuration Information Inventory
  - LED Description Record Fields (PICMG Record Id 2Fh) for AdvancedTCA FRUs.
- E-Keying sensors for the Synchronization Clock Bus and the Metallic Test Bus are not supported.
- According to the mapping spec, maximum power capability sensor is deprecated and provided only for backward compatibility.
- The Max value of Assigned Power Sensors for the Managed FRU slots shall not be set. HPI users can check the Max value for the Assigned Power Sensor of the Managing FRU slots to determine the total power capability of that slot along with all slots being managed under the same hardware address.

4.3 Backward Compatibility

This section describes the various aspects that HPI-B user should make a note while migrating from HPI-B 02.01 based applications to HPI-B 03.02, or in using the HPI-B 02.01 applications with the HPI-B 03.02 daemon. The following points could impact existing HPI-B client applications.

It is recommended for the user to note the differences in the specification and then modify the client applications suitably.

- Some management instruments and resources that were mandatory in SAIM-HPI-B.01.01-ATCA are either made optional or removed in the latest specification.

- Some of the Resource Data Record (RDR) data has been modified to better describe the management instruments in this specification. This modification includes changing entity paths, sensor types, and control output types on various management instruments. HPI-B user applications compliant to HPI-B 02.01 specification will receive data as per the B03.02 xTCA mapping specification. These applications need to be modified to interpret the data correctly.
- The symbolic names of all defined values are changed. A user program compiled with previous header files should still be usable with HPI library implementations that are built using the new symbols defined in this specification. We have taken care of this part by back porting the Xtca prefix to Atca Prefix.

Following table describes the backward compatibility:

Table 4-1 Backward Compatibility Matrix

HPI-B Daemon version	HPI-B Library version	Compatibility
B.03.02	B.02.01	Yes, functionality as B.03.02 daemon specs compliance. Refer to the backward compatibility notes in the Mapping specification.
B.02.01	B.03.02	Yes, Only B.02.01 functionality.

4.4 Working with the Multishelf Library

The multishelf HPI library allows your application to connect to and manage several shelves at the same time. For this purpose the multishelf library provides several HPI controls which allow the application to manage the connection to shelves and also to dynamically add and remove shelves to the HPI environment. This section provides all the information that you need to know in order to use these HPI controls and work with the HPI multishelf library.

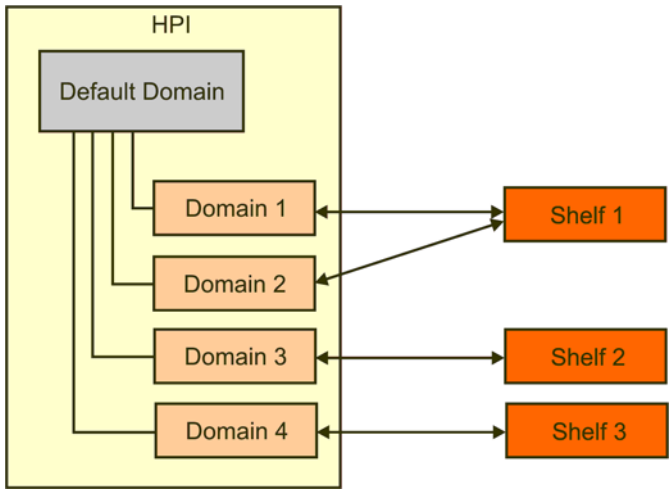
4.4.1 Overview

HPI uses the concept of domains. Generally, a domain represents one shelf. Furthermore there is a default domain. It acts as a container for all other domains and does itself not represent actual hardware.

It is possible for multiple domains to represent the same physical shelf. This is for example the case in typical AdvancedTCA systems which often provide two shelf managers with an HPI daemon running on each of them.

The following figure illustrates an example configuration with four domains and three shelves.

Figure 4-1 Multishelf Library - Representation of Shelves as Domains



Any FRUs available in a shelf are represented as HPI resources together with Resource Data Records (RDRs) corresponding to that FRU. Whenever a FRU is added to or removed from a shelf, the corresponding HPI resource/RDR is added/removed from the HPI domain.

4.4.2 Accessing HPI Domains

In order to access an HPI domain, you must open a session via the HPI call `saHpiSessionOpen( )` and provide as first parameter the domain ID of the corresponding HPI domain. How to obtain the domain ID is described later within this section about the multishelf HPI library.

If you want to access several shelves, then you need to open several sessions simultaneously, one session for each HPI domain which represents a shelf. It is also possible to open several sessions for one HPI domain/shelf only as well.

When the connection to a shelf is lost, all running HPI calls which access the corresponding domain return immediately with the error code `SA_HPI_ERR_NO_RESPONSE`. All open sessions for the affected domain are automatically closed by the multishelf library. In the meantime, the library tries to regain access to the shelf. As soon as the connection is reestablished, the domain is recreated and the application can open another session and access the domain again. Whenever a domain is created or removed, an HPI event from the HPI Communication State sensor is generated in the default domain (see *Connection State Sensor on page 55* and *HPI Domain Events on page 56*).

HPI events are handled domain wide. This means that HPI events from a shelf or FRUs in that shelf are only visible and can only be received within the session that corresponds to that domain.

4.4.3 How Domains and Shelves are Represented

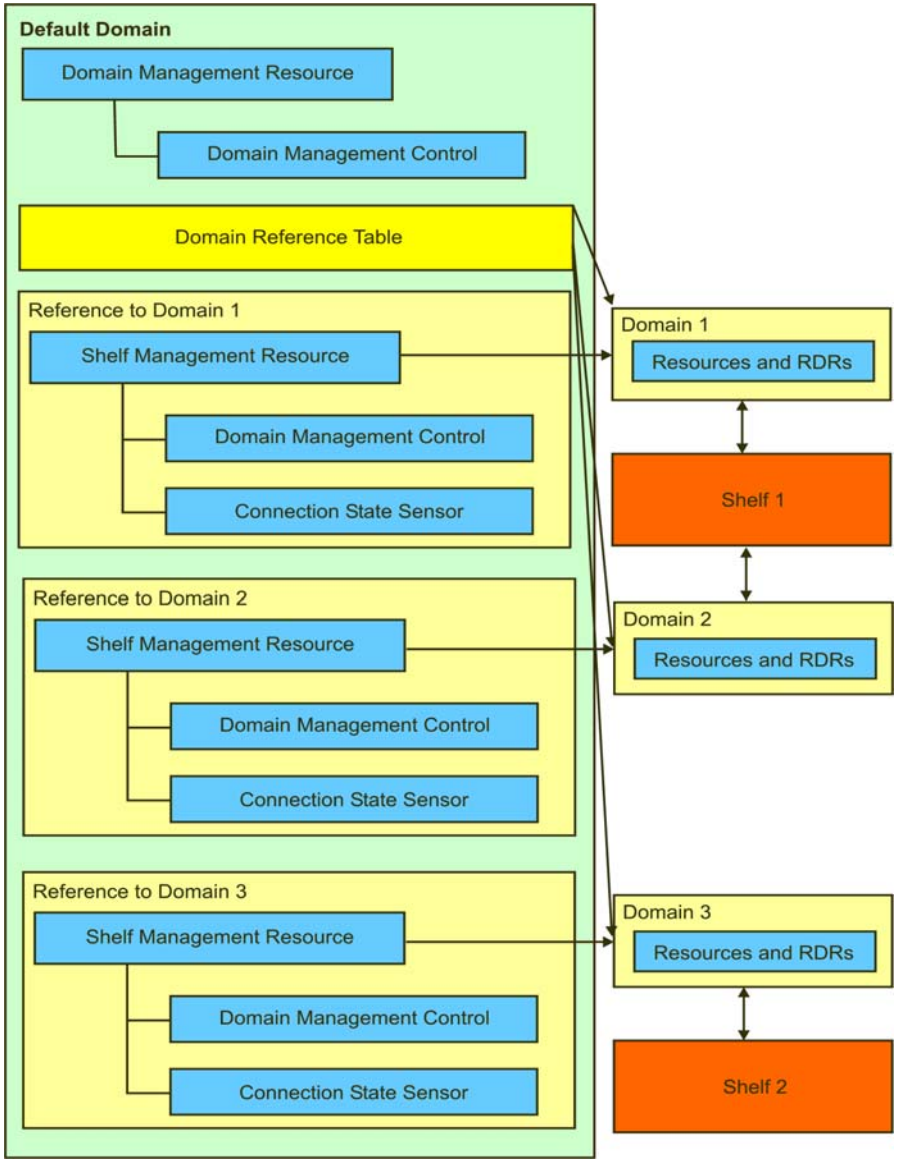
As previously mentioned, each HPI implementation has at least the default domain. It has the ID 0 assigned to it.

Starting with HPI-B, the default domain contains a Domain Reference Table, which contains references to all related domains and may be used by applications for discovery of available domains in the current configuration. For more information about the Domain Reference Table, refer to the HPI-B specification document of the SAI-HPI-B.02.01 standard.

In the SMART EC HPI-B implementation, the default domain furthermore contains multiple HPI resources which handle connected shelves and their corresponding HPI domains. These HPI resources were defined and added by SMART EC and are called Domain Management Resource and Shelf Management Resource.

The Domain Management Resource contains one HPI control and allows applications to add/remove HPI domains/shelves to the HPI environment. The Shelf Management Resource contains one HPI control and one HPI sensor and acts as reference to connected daemons. There is one Shelf Management Resource for each connected daemon. The following figure shows an example configuration with the HPI resources, controls and sensors which are related to the handling of multiple shelves/domains in it.

Figure 4-2 HPI Multishelf Library - Overview of Related HPI Resources and Controls



In the following section, the Domain Management Resource and the Shelf Management resource will be described in detail. A description of typical usage examples/scenarios will be given after that.

4.4.3.1 Domain Management Resource

The Domain Management Resource acts as container for the Domain Management Control and is defined as follows.

Table 4-2 Definition of Domain Management Resource

SaHpiRptEntryT	Value
`EntryId`	Assigned by HPI
`ResourceId`	Assigned by HPI
`ResourceInfo`	`0` for all values
`ResourceEntity`	`{RACK,0}` this can be changed with the multishelf library configuration file
`ResourceCapabilities`	`SAHPI_CAPABILITY_RESOURCE | SAHPI_CAPABILITY_RDR | SAHPI_CAPABILITY_CONTROL`
`ResourceSeverity`	`SAHPI_MAJOR`
`DomainId`	`0`
`IdString`	Domain management

This Domain Management control is only writable, not readable, and allows the application to dynamically add and remove domains.

Adding/removing a domain using this HPI control has the same effect as adding/removing a shelf by adding/removing an entry in the multishelf library configuration file. Therefore, whenever you use this HPI control to add/remove a domain, the software automatically updates the configuration file as well.

The RDR and the HPI control are defined as follows.

Table 4-3 Domain Management Control RDR

SaHpiRdrT	Value
`RecordId`	Assigned by HPI
`RdrType`	`SAHPI_CTRL_RDR`
`Entity`	The same entity like Domain Management Resource
`RdrTypeUnion`	Define in *Table 4-4*.
`IdString`	`MOTHPI_CTRL_NAME_DOMAIN_MANAGEMENT`

Table 4-4 Domain Management Control

SaHpiCtrlRecT	Value
Num	MOTHPI_CTRL_NUM_DOMAIN_MANAGEMENT
Ignore	SAHPI_FALSE
OutputType	SAHPI_CTRL_OEM
Type	SAHPI_CTRL_TYPE_OEM
TypeUnion - Oem -Mld	MOTHPI_MANUFACTURER_ID_MOTOROLA
TypeUnion - Oem -ConfigData	0
TypeUnion - Oem - Default - Mld	0
TypeUnion - Oem - Default - BodyLength	0
TypeUnion - Oem - Default - Body	0
Oem	0

Table 4-5 Domain Management Control State

SaHpiCtrlStateT	Value
Type	SAHPI_CTRL_TYPE_OEM
StateUnion - Oem - Mld	MOTHPI_MANUFACTURER_ID_MOTOROLA
StateUnion - Oem - BodyLength	Depends on the length of the resource name
StateUnion - Oem - Body	Sequence of n bytes, named [0] ... [n], with the following definitions: [0] - Command 0 = Get state 1 = Create domain (for set state) 2 = Remove domain (for set state) [1] - [4] - IP address (little endian order) [5][6] - port (little endian order) [7] ... [n] - Domain name as null-terminated string.

4.4.3.2 Shelf Management Resource

The shelf management resource represents one daemon. For each configured daemon, the default domain creates one shelf management resource. The shelf management resource is defined as follows.

Table 4-6 Shelf Management Resource

SaHPIRptEntryT	Value
`EntryId`	Assigned by HPI
`ResourceId`	Assigned by HPI
`resourceInfo`	`0` for all values
`ResourceEntity`	`{SYS_MNGMT_SOFTWARE domain Id}`

`ResourceCapabilities`	`SAHPI_CAPABILITY_RESOURCE | SAHPI_CAPABILITY_RDR | SAHPI_CAPABILITY_CONTROL | SAHPI_CAPABILITY_SENSOR`
	When the shelf is reachable:
	`SAHPI_CAPABILITY_DOMAIN`

`ResourceSeverity`	`SAHPI_MAJOR`
`DomainId`	Domain Id when the shelf is reachable
`IdString`	Domain name

The shelf management resource contains the following two RDRs:

- Domain management control
  Read-only HPI control which allows the application to obtain the IP address, port, domain name and other connection parameters of a particular shelf
- Connection status sensor
  Contains information about the current status of the connection to a shelf

The definitions of both HPI controls are given in the following.

4.4.3.2.1 Domain Management Control

This HPI control is only readable. It has the same structure and definition as the HPI controls defined in the following tables: *Domain Management Control on page 53*, *Domain Management Control State on page 53* and *Domain Management Control RDR on page 52*.

4.4.3.2.2 Connection State Sensor

This HPI sensor represents the status of the connection to a shelf. When the connection is interrupted, an HPI event of type SAHPI_ES_OFF_LINE is generated. When the connection is reestablished, an HPI event of type SAHPI_ES_ON_LINE is generated. An application can access a shelf only when the state of this sensor is SAHPI_ES_ON_LINE. The definition of this sensor is given in the following tables.

Table 4-7 Connection State RDR

SaHpiRdrT	Value
RecordId	Assigned by HPI
RdrType	SAHPI_SENSOR_RDR
Entity	The same entity as domain management resource
RdrTypeUnion	MOTHPI_SENSOR_NAME_DOMAIN_CONNECTION

Table 4-8 Domain Connection Sensor

SaHpiSensorRecT	Value
Num	MOTHPI_SENSOR_NUM_DOMAIN_CONNECTION
Type	SAHPI_CHASSIS
Category	SAHPI_EC_AVAILABILITY
EventCtrl	SAHPI_SEC_PER_EVENT
Events	SAHPI_ES_ON_LINE | SAHPI_ES_OFF_LINE
Ignore	SAHPI_FALSE
DataFormat - ReadingFormats	SAHPI_SRF_EVENT_STATE
DataFormat - IsNumeric	SAHPI_FALSE
DataFormat - IsThreshold	SAHPI_FALSE
Oem	0

Table 4-9 Domain Connection Sensor Reading

SaHpiSensorReadingT	Value
ValuesPresent	SAHPI_SRF_EVENT_STATE
EventStatus - SensorStatus	SAHPI_SENSTAT_EVENTS_ENABLED | SAHPI_SENSTAT_SCAN_ENABLED

Table 4-9 Domain Connection Sensor Reading (continued)

SaHpiSensorReadingT	Value
AssertEvents	SAHPI_ES_ON_LINE for a connection to a shelf SAHPI_ES_OFF_LINE when there is no connection to a shelf

4.4.3.3 HPI Domain Events

Depending on the connection status, the Domain Connection sensor can throw events. The following table provides details.

Table 4-10 Domain Connection Sensor Events

Event	Description
SAHPI_ES_OFF_LINE	No connection
SAHPI_ES_ON_LINE	Connection is established. Only now is the domain of the shelf accessible.

4.4.4 Discovering Shelves and Domains

As previously mentioned, one way to discover available domains is to use the Domain Reference Table. For further details, refer to the SAI-HPI-B.02.01 specification document.

Alternatively, you can use the Shelf Management Resources defined in the default domain. Any shelves that were added to the HPI environment are represented by one Shelf Management Resource with the entry ResourceCapability set to SAHPI_CAPABILITY_DOMAIN. So in order to discover shelves/domains, you simply need to parse the default domain for HPI resources with the entry ResourceCapability set to SAHPI_CAPABILITY_DOMAIN. Then you can obtain the respective domain ID by reading the entry DomainId and open a session to the shelf using the obtained domain ID.

4.4.5 Adding and Removing Shelves and Domains

All shelves which are used by the HPI multishelf library are listed in the multishelf library configuration file. There are two ways of adding and removing shelves:

- Manually by editing the multishelf configuration file. Changes become effective after the next restart in this case.
- Dynamically by using the HPI domain management control

For a description of how to manually edit the multishelf library configuration file, refer to *Configuring HPI Clients on page 34*.

A description of how to dynamically add and remove shelves, is given in the following two subsections.

4.4.5.1 Adding Shelves and Domains

In order to dynamically add a shelf and create a domain, you must invoke the HPI function `saHpiControlStateSet()` and provide the following parameters.

Parameter	Value
`SessionId`	Handle to session context.
`ResourceId`	Resource ID of the addressed resource.
`CtrlNum`	Number of the control for which the state is being set.
`CtrlState`	Pointer to control state as described in table *Domain Management Control State on page 53*. In this control state, fill the field body as follows:, `[0] : 1` (stands for "create domain") `[1] - [4] :` IP address of shelf you want to add (little endian order) `[5] - [6] :` port of shelf (little endian order) `[7] - [n] :` Domain name

After calling `saHpiControlStateSet()`, a new resource with the domain name you supplied as parameter is created. Additionally, a hot swap event of type ACTIVE for the created resource is sent. The newly created shelf is also added to the HPI multishelf configuration file as a new entry.

All shelves must have different IP addresses.

4.4.5.2 Removing Shelves and Domains

In order to dynamically remove a shelf/domain, you must invoke the HPI function `saHpiControlStateSet()` and provide the following parameters.

Parameter	Value
`SessionId`	Handle to session context.
`ResourceId`	Resource ID of the addressed resource.
`CtrlNum`	Number of the control for which the state is being set.

Parameter	Value
CtrlState	Pointer to control state as described in *Domain Management Control State on page 53*. In this control state, fill the field body as follows:, [0] : 2 (stands for "remove domain") [1] - [4] : IP address of shelf you want to add (little endian order) [5] - [6] : port of shelf (little endian order) [7] - [n] : Domain name

If the domain/shelf specified in the call exists, the domain/shelf is removed, an HPI domain removed event is generated and any sessions that may be open to this domain are closed. Furthermore the domain shelf resource in the HPI domain is removed and a hot swap even of type "NOT_PRESENT" is generated.

4.5 SMART EC Extensions

The following describes features which are not specified in the HPI-B specifications, but were added by SMART EC or which are specific to the SMART EC HPI-B implementation.

4.5.1 HPI Controls for Domain and Shelf Management

This refers to the Domain Management Resource and Shelf Management Resource which were previously explained. Both are SMART EC-specific extensions. See *Section 4.4, Working with the Multishelf Library on page 48*.

4.5.2 IPMI System Boot Options Support

A FRU may have a payload which is capable of booting an operating system. Usually, a boot firmware, such as BIOS or U-Boot, is started after the payload is powered up or reset. Via the System Boot Options Control you can set some options for the boot firmware. The boot firmware will read these settings from the IPMC.

In order to set or get the system boot options, you need to use the HPI Boot Option control.

This control maps the IPMI commands **`Set System Boot Options`** and **`Get System Boot Options`** to HPI.

The Boot Option Control is only available for AdvancedTCA front blades, AMC modules and the shelf manager if the respective IPMC supports the `Set System Boot Option` IPMI command. Refer to the Intelligent Platform Management Interface Specification v2.0, section 28.12 Set System Boot Options Command and 28.13 Get System Boot Options Command, for further details. Furthermore refer to the respective IPMI Programmer's Reference manuals of the respective blades/modules.

Table 4-11 Boot Option RDR

SaHpiRdrT	Value
RecordId	Assigned by HPI
RdrType	SAHPI_CTRL_RDR
Entity	The same entity as resource
RdrTypeUnion	Defined in next table
IdString	MOTHPI_CTRL_NAME_BOOT_OPTION

Table 4-12 Boot Option Control

SaHpiCtrlRecT	Value
Num	MOTHPI_CTRL_NUM_BOOT_OPTION
Ignore	SAHPI_FALSE
OutputType	SAHPI_CTRL_OEM
Type	SAHPI_CTRL_TYPE_OEM
TypeUnion.Oem.Mid	MOTHPI_MANUFACTURER_ID_MOTOROLA
Oem	0

Table 4-13 Boot Option State

SaHpiCtrlStateT	Value
Type	SAHPI_CTRL_TYPE_OEM
StateUnion.Oem.Mid	MOTHPI_MANUFACTURER_ID_MOTOROLA

Table 4-13 Boot Option State (continued)

SaHpiCtrlStateT	Value
`StateUnion.Oem.BodyLength`	Get operations: When input parameter: 3 When output parameter: Total length of the response data - 2 Set operations: Total length of the request data
`StateUnion.Oem.Body`	Get operations: When input parameter: Byte 0: Parameter selector Byte 1: Set selector Byte 2: Block selector When output parameter: Response data from the GetSystemBootOptions IPMI command without the first two bytes Set operations: Request data for the IPMI SetSystemBootOptions command

Example: On most SMART EC blades, the BIOS software is stored twice on a flash device, BIOS bank 1 and BIOS bank 2. The bank to boot from can be selected with the System Boot Options Control.

You have to select parameter 96 with the Boot Option Select Control and then set the BIOS 1 (0) or BIOS 2 (1) with the Boot Option Control.

The following example shows how to set BIOS 2 for a resource with the ID 120 using an example program which is delivered together with the HP-B development package:

```
hpibootoptions -r 120 96 1
```

The example program hpibootoptions, which is available in the base RPM package, shows how an option can be set or got.

4.5.3 POST Type Control

This HPI control allows you to set/get the Power-On Self Test (POST) type of the blade. Two POST types are configurable: long POST and short POST. Refer to the respective hardware user manual of the blade for details about both POST types. The definition of the respective HPI controls is given in the following tables.

Table 4-14 POST Type HPI Control RDR

SaHpiRdrT	Value	Notes
RecordId	Assigned by HPI implementation	Unique identifier for the RDR
RdrType	SAHPI_CTRL_RDR	
RdrTypeUnion.CtrlRec.Num	MOTHPI_CTRL_NUM_POSTTYPE	
RdrTypeUnion.CtrlRec.OutputType	SAHPI_CTRL_OEM	Indicates that this control does not correlate to any of the given control output types, and describes a generic control output.
RdrTypeUnion.CtrlRec.Type	SAHPI_CTRL_TYPE_DISCRETE	
RdrTypeUnion.CtrlRec.TyoeUnion.Discrete.Default	0x0	
RdrTypeUnion.CtrlRec.Oem	Assigned by HPI implementation	
IdString	MOTHPI_CTRL_NAME_POSTTYPE	

Table 4-15 POST Type HPI Control Sate Values

SaGPISateT	Value	Notes
Type	SAHPI_CTRL_TYPE_DISCRETE	
StateUnion.Discrete	Bits 7..0: 0: Short POST 1: Long POST Bits 15..8: CPU complex number Bits 31..16: Reserved	The POST type value is specified in the least significant byte. The CPU number (or SET Selector byte) is normally zero. However, for blades that support more than one CPU complex, the processor complex is identified with this field.

4.5.4 Shelf-Specific Mappings

The following table contains some notes on shelf-specific usage/mapping of HPI controls.

Table 4-16 Shelf-Specific Mapping of HPI Controls

Feature	Note
Shelf geographical address	Centellis 2000/Centellis 2100/4411 Here the shelf geographical address can only be changed via HPI-B/IPMI. This is the only way to modify it.
Shelf alarm LEDs	Centellis 2000/Centellis 2100 The shelf alarm LEDs are controlled through the LED controls which are attached to the HPI-B shelf resource. Centellis 4411 The shelf alarm LEDs are located on the Alarm Display Panel at the front of the shelf and are controlled by the Minor Alarm, Major Alarm, and Critical Alarm controls associated with the shelf manager resources.

4.5.5 HPI Logging Support

The actions taken by the HPI daemon are written to a log file. The path and name of the log file is defined in the configuration file `bbs-hpib.conf`.

To control what kind of information is written to the log file, the log control can be used.

Refer to the header file `MotorolaHpi.h` for used defines.

Table 4-17 Log RDR

SaHpiRdrT	Value
`RecordId`	Assigned by HPI
`RdrType`	`SAHPI_CTRL_RDR`
`Entity`	The same entity as logical shelf resource
`RdrTypeUnion`	Defined in next table
`IdString`	`MOTHPI_CTRL_NAME_LOG` found in `MotorolaHpi.h`

Table 4-18 Log Control

SaHpiCtrlRecT	Value
Num	MOTHPI_CTRL_NUM_LOG
Ignore	SAHPI_FALSE
OutputType	SAHPI_CTRL_OEM
Type	SAHPI_CTRL_TYPE_OEM
TypeUnion.Oem.MId	MOTHPI_MANUFACTURER_ID_MOTOROLA
TypeUnion.Oem.ConfigData	
TypeUnion.Oem.Default.MId	MOTHPI_MANUFACTURER_ID_MOTOROLA
TypeUnion.Oem.Default.BodyLength	0
TypeUnion.Oem.Default.Body	0
Oem	0

Table 4-19 Log Control State

SaHpiCtrlStateT	Value
Type	SAHPI_CTRL_TYPE_OEM
StateUnion.Oem.MId	MOTHPI_MANUFACTURER_ID_MOTOROLA
StateUnion.Oem.BodyLength	40 + length of log file name

Table 4-19 Log Control State (continued)

SaHpiCtrlStateT	Value
StateUnion.Oem.Body	MOTHPI_LOG_ERROR_FACILITIES_OFFSET - facilities for that error logging is enabled MOTHPI_LOG_WARNING_FACILITIES_OFFSET - facilities for that warning logging is enabled MOTHPI_LOG_INFO_FACILITIES_OFFSET - facilities for that info logging is enabled MOTHPI_LOG_DEBUG_FACILITIES_OFFSET - facilities for that debug logging is enabled The logging facilities are defined in the file log_utils.h, enum oh_log_fac. MOTHPI_LOG_CURRENT_IDX_OFFSET - index of currently used log file [MOTHPI_LOG_OFFSET_PROPERTIES] - output locationdLogStdout - stdoutdLogStderr - stderrdLogFile - log file [MOTHPI_LOG_NUM_FILES_OFFSET] - number of logfile created [MOTHPI_LOG_MAX_FILE_SIZE_OFFSET] - maximum logfile size before creating a new one. This is a 32 bit field in MSB byte order. Use GetUInt32 in byte_utils.h to get host byte order. [MOTHPI_LOG_FILENAME_OFFSET] - log file name.

4.5.6 HPI Daemon Redundancy Sensor

The SMART EC HPI-B implementation provides an OEM sensor which indicates the redundancy state of the HPI daemon. As previously mentioned, both HPI daemons in a shelf are operated in cold stand-by mode, i.e. one HPI daemon is active, while the other daemon is deactivated. A supervisor which runs on both ATCA-MF106/SAM1411 blades furthermore monitors the activity of the currently active daemon, and if it detects a failure, the supervisor activates the previously stand-by daemon and switches over to it. The following table provides possible states which are indicated by the HPI daemon redundancy sensor.

Table 4-20 HPI Daemon Redundancy Sensor - Redundancy States

State	Description
MOTHPI_ES_FULLY_REDUNDANT	There is one active and one deactivated daemon. The deactivated daemon is ready to be activated in case of a failure.

Table 4-20 HPI Daemon Redundancy Sensor - Redundancy States (continued)

State	Description
MOTHPI_ES_NON_REDUNDANT_SUFFICIENT_RESOURCES	There is one active HPI daemon, but no deactivated daemon which could be activated in case of a failure. A possible reason might be that there is only one ATCA-MF106/SAM1411 in the shelf.
No state	All HPI daemons have failed or are deactivated. A possible reason could be that there is no ATCA-MF106/SAM1411 blade in the shelf.

The RDR of this sensor is similar to the RDR of the Virtual Shelf Manager Redundancy sensor described in the HPI-to-AdvancedTCA mapping specification.

The sensor's RDR is defined as follows.

Table 4-21 HPI Daemon Redundancy Sensor RDR

MotHpiRdrT	Value	Notes
RecordId	Assigned by HPI implementation	
RdrType	SAPHPI_SENSOR_RDR	
Entity	Same as Virtual Shelf Manager Resource	
IsFru	SAPHI_FALSE	Per the rules stated in [HPI], the IsFru flag is not applicable when the Management Instrument is directly associated with the Resource, as is the case here.
RdrTypeUnion.SensorRec.Num	MOTHPI_SENSOR_NUM_DAEMON_REDUNDANCY	
RdrTypeUnion.SensorRec.Type	SAHPI_OPERATIONAL	
RdrTypeUnion.SensorRec.Category	SAHPI_EC_REDUNDANCY	
RdrTypeUnion.SensorRec.EnableCtrl	True if the HPI User can enable or disable the Sensor via saHpiSensorEnableSet()	

Table 4-21 HPI Daemon Redundancy Sensor RDR (continued)

MotHpiRdrT	Value	Notes
RdrTypeUnion.SensorRec.EventCtrl	SAHPI_SEC_PER_EVENT | SAHPI_SEC_READ_ONLY_MASKS | SAHPI_SEC_READ_ONLY	
RdrTypeUnion.SensorRec.Events	SAHPI_ES_FULLY_REDUNDANT | SAHPI_ES_NON_REDUNDANT_SUFFICIENT_RESOURCES | SAHPI_ES_NON_REDUNDANT_INSUFFICIENT_RESOURCES	
RdrTypeUnion.SensorRec.DataFormat.IsSupported	SAHPI_FALSE	
RdrTypeUnion.SensorRec.DataFormat.ReadingType	Not Applicable	
RdrTypeUnion.SensorRec.DataFormat.BaseUnits	Not Applicable	
RdrTypeUnion.SensorRec.DataFormat.ModifierUnits	Not Applicable	
RdrTypeUnion.SensorRec.DataFormat.ModifierUse	Not Applicable	
RdrTypeUnion.SensorRec.DataFormat.Percentage	Not Applicable	
RdrTypeUnion.SensorRec.DataFormat.Range.Flags	Not Applicable	
RdrTypeUnion.SensorRec.DataFormat.Range	Not Applicable	
RdrTypeUnion.SensorRec.DataFormat.AccuracyFactor	Not Applicable	
RdrTypeUnion.SensorRec.ThresholdDefn.IsAccessible	SAHPI_FALSE	
RdrTypeUnion.SensorRec.ThresholdDefn.ReadThold, RdrTypeUnion.SensorRec.ThresholdDefn.WriteThold, RdrTypeUnion.SensorRec.ThresholdDefn.Nonlinear	These values are not meaningful, because the IsAccessible flag is set to SAHPI_FALSE.	

Table 4-21 HPI Daemon Redundancy Sensor RDR (continued)

MotHpiRdrT	Value	Notes
RdrTypeUnion.SensorRec.OEM	Assigned by HPI implementation	
IdString	MOTHPI_SENSOR_NAME_DAEMON_REDUNDANCY	

Table 4-22 HPI Daemon Redundancy Sensor Reading

SaHpiSensorReadingT	Value	Notes
IsSupported	SAHPI_FALSE	This Sensor supports only event states and no reading value.
Type	Not Applicable.	
Value	Not Applicable.	

Table 4-23 HPI Daemon Redundancy Sensor Event State

MotHpiEventStateT	Value	Notes
EventState	SAHPI_ES_FULLY_REDUNDANT | SAHPI_ES_NON_REDUNDANT_SUFFICIENT_RESOURCES	

4.5.7 Active and Stand-By HPI Daemon Sensor

The SMART EC HPI-B implementation provides two sensors which allow to detect which of the two HPI daemons in a system is the currently active, and which is the currently deactivated one. There are two sensors: one which represents the currently active HPI daemon, and one which represents the currently deactivated daemon. Both sensors are SMART EC OEM sensors, but the structure of their RDR is similar to the RDR of the standard Active and Standby Shelf Manager sensors which are described in the HPI-to-AdvancedTCA mapping specification.

Both Sensors are logical Sensors which do not have counterparts in the IPMI or AdvancedTCA realm. Applicable events and readings are generated by the HPI implementation.

Both Sensors are modeled as Presence Sensors.

Table 4-24 Active and Stand-By HPI Daemon Sensors RDRs

SaHpiRdrT	Value	Notes
RecordId	Assigned by HPI implementation	
RdrType	SAHPI_SENSOR_RDR	
Entity	Same as Virtual Shelf Manager Resource	
IsFru	SAPHI_FALSE	Per the rules stated in [HPI], the IsFru flag is not applicable when the Management Instrument is directly associated with the Resource, as is the case here.
RdrTypeUnion.SensorRec.Num	MOTHPI_SENSOR_ACTIVE_DAEMON_NUM MOTHPI_SENSOR_STANDBY_DAEMON_NUM	MOTHPI_SENSOR_ACTIVE_DAEMON_NUM corresponds to the currently active daemon MOTHPI_SENSOR_STANDBY_DAEMON_NUM corresponds to the currently deactivated daemon
RdrTypeUnion.SensorRec.Type	SAHPI_ENTITY_PRESENCE	
RdrTypeUnion.SensorRec.Category	SAHPI_EC_PRESENCE	
RdrTypeUnion.SensorRec.EnableCtrl	True if the HPI User can enable or disable the Sensor via saHpiSensorEnableSet()	
RdrTypeUnion.SensorRec.EventCtrl	SAHPI_SEC_PER_EVENT | SAHPI_SEC_READ_ONLY_MASKS | SAHPI_SEC_READ_ONLY	

Table 4-24 Active and Stand-By HPI Daemon Sensors RDRs (continued)

SaHpiRdrT	Value	Notes
RdrTypeUnion.SensorRec.Events	SAHPI_ES_PRESENT | SAHPI_ES_ABSENT	
RdrTypeUnion.SensorRec.DataFormat.IsSupported	SAHPI_TRUE	Note that this Sensor will report the ResourceId of the resident Resource, via a Sensor value.
RdrTypeUnion.SensorRec.DataFormat.ReadingType	SAHPI_SENSOR_READING_TYPE_UINT64	Although ResourceIDs are 32 bits, we need to use the Uint64 Sensor reading type for this Sensor. The least significant 32 bits are populated with the ResourceId, the most significant 32 bits are reserved and set to 0.
RdrTypeUnion.SensorRec.DataFormat.BaseUnits	SAHPI_SU_UNSPECIFIED	
RdrTypeUnion.SensorRec.DataFormat.ModifierUnits	SAHPI_SU_UNSPECIFIED	
RdrTypeUnion.SensorRec.DataFormat.ModifierUse	SAHPI_SMUU_NONE	
RdrTypeUnion.SensorRec.DataFormat.Percentage	SAHPI_FALSE	
RdrTypeUnion.SensorRec.DataFormat.Range.Flags	00h	No flags are set, indicating that no range is defined.
RdrTypeUnion.SensorRec.DataFormat.Range	Not Applicable	
RdrTypeUnion.SensorRec.DataFormat.AccuracyFactor	0.00	

Table 4-24 Active and Stand-By HPI Daemon Sensors RDRs (continued)

SaHpiRdrT	Value	Notes
RdrTypeUnion.SensorRec.ThresholdDefn.IsAccessible	SAHPI_FALSE	
RdrTypeUnion.SensorRec.ThresholdDefn.ReadThold, RdrTypeUnion.SensorRec.ThresholdDefn.WriteThold, RdrTypeUnion.SensorRec.ThresholdDefn.Nonlinear	These values are not meaningful, because the IsAccessible flag is set to SAHPI_FALSE.	
RdrTypeUnion.SensorRec.OEM	Assigned by HPI implementation	
IdString	MOTHPI_SENSOR_ACTIVE_DAEMON_NAME MOTHPI_SENSOR_STANDBY_DAEMON_NAME	

Table 4-25 Active and Stand-By HPI Daemon Sensors Reading

SaHpiSensorReadingT	Value
isSupported	SAHPI_TRUE
Type	SAHPI_SENSOR_READING_TYPE_UINT64
Value	ResourceId in least significant 32 bits if one is present, otherwise SAHPI_UNSPECIFIED_RESOURCE_ID

Table 4-26 Active and Stand-By HPI Daemon Sensors Event State

SaHpiEventStateT	Value
EventState	SAHPI_ES_PRESENT | SAHPI_ES_ABSENT

4.5.8 HPI Daemon Switchover Control

This HPI control handles the switchover between the active and deactivated HPI-B daemon. The location (active/deactivated) of the HBI daemon cannot be controlled directly. Instead, this HPI control can be used to initiate a switchover. After the switchover is completed, if eventing is enabled for the active HPI-B daemon sensor, the HPI Implementation will send an event with the Resource Id of the new Resource hosting the active HBP daemon.

Note that the initiation of a switchover is meaningful only when the state of the HPI daemon redundancy sensor is SAHPI_ES_FULLY_REDUNDANT. If there is no standby HPI daemon present, an attempt to initiate a switchover will return the error code SA_ERR_HPI_INVALID_REQUEST.

Table 4-27 HPI Daemon Failover Control RDR

SaHpiRdrT	Value	Notes
RecordId	Assigned by HPI implementation	
RdrType	SAHPI_CTRL_RDR	
Entity	Same as Shelf Manager Resource	
IsFru	SAHPI_FALSE	
RdrTypeUnion.CtrlRec.Num	MOTHPI_CTRL_SWITCHOVER_DAEMON_NUM	
RdrTypeUnion.CtrlRec.Type	SAHPI_CTRL_TYPE_DIGITAL	
RdrTypeUnion.CtrlRec.OutputType	SAHPI_CTRL_GENERIC	
RdrTypeUnion.CtrlRec.TypeUnion.Digital.Default	SAHPI_CTRL_STATE_OFF	
RdrTypeUnion.CtrlRec.DefaultMode.Mode	SAHPI_CTRL_MODE_MANUAL	
RdrTypeUnion.CtrlRec.DefaultMode.ReadOnly	SAHPI_TRUE	
RdrTypeUnion.CtrlRec.WriteOnly	SAHPI_FALSE	
RdrTypeUnion.CtrlRec.OEM	Assigned by HPI implementation	
RdrTypeUnion.CtrlRec.IdString	MOTHPI_CTRL_SWITCHOVER_DAEMON_NAME	

Table 4-28 HPI Daemon Failover Control States

SaHpiCtrlStateT	Value	Notes
Type	SAHPI_CTRL_TYPE_DIGITAL	
StateUnion.Digital	SAHPI_CTRL_STATE_PULSE_ON | SAHPI_CTRL_STATE_ OFF	

4.5.9 Failed Slot Restore Control

"Failed Slot Restore Control" control enables the user to initiate a re-discovery of the resource in the slot and sets the `ResourceFailed` flag in RPT to `false`. The `ResourceFailed` flag in the RPT Entry of slot resource is set to `true` if the discovery of the resource in the slot fails, or the resource in the slot is in failed state. The `ResourceFailed` flag of the slot resource can be set to `false` in any one of the following conditions:

Resource in failed state is removed using `saHpiResourceFailedRemove()`.

If any of the resource is inserted into the failed slot.

An example application `hpislotrestore` can be used to set and get the new control "Failed Slot Restore" state.

Table 4-29 Failed Slot Restore Control RDR

SaHpiRdrT	Value	Notes
`RecordId`	Assigned by HPI implementation.	
`RdrType`	`SAHPI_CTRL_RDR`	
`Entity`	Entity path of the slot resource.	
`IsFru`	`SAHPI_FALSE`	
`RdrTypeUnion.CtrlRec.Num`	`MOTHPI_CTRL_FAILED_SLOT_RESTORE_NUM`	
`RdrTypeUnion.CtrlRec.Type`	`SAHPI_CTRL_TYPE_DIGITAL`	
`RdrTypeUnion.CtrlRec.OutputType`	`SAHPI_CTRL_GENERIC`	
`RdrTypeUnion.CtrlRec.TypeUnion.Digital.Default`	`SAHPI_CTRL_STATE_OFF`	
`RdrTypeUnion.CtrlRec.DefaultMode.Mode`	`SAHPI_CTRL_MODE_MANUAL`	
`RdrTypeUnion.CtrlRec.DefaultMode.ReadOnly`	`SAHPI_TRUE`	
`RdrTypeUnion.CtrlRec.WriteOnly`	`SAHPI_FALSE`	
`RdrTypeUnion.CtrlRec.OEM`	Assigned by HPI implementation.	
`RdrTypeUnion.CtrlRec.IdString`	`MOTHPI_CTRL_FAILED_SLOT_RESTORE_NAME`	

Table 4-30 Failed Slot Restore Control States

SaHpiCtrlStateT	Value	Notes
Type	SAHPI_CTRL_TYPE_DIGITAL	
StateUnion.Digital	SAHPI_CTRL_STATE_PULSE_ON | SAHPI_CTRL_STATE_OFF	

4.5.10 HPI Restart Daemon Control

HPI Restart Daemon control enables the user to restart the hpi-b daemon. hpirestartdaemon client application makes use of this control to restart the daemon.

Table 4-31 HPI Restart Daemon RDR

SaHpiRdrT	Value
RecordId	Assigned by HPI implementation
RdrType	SAHPI_CTRL_RDR
Entity	Entity path of 'Daemon Resource' Ex: {ADVANCEDTCA_CHASSIS,6} {SYS_MGMNT_SOFTWARE,0}
RdrTypeUnion.CtrlRec.Num	MOTHPI_CTRL_NUM_HPI_RESTART_DAEMON
RdrTypeUnion.CtrlRec.Type	SAHPI_CTRL_TYPE_DIGITAL
RdrTypeUnion.CtrlRec.OutputType	SAHPI_CTRL_GENERIC
RdrTypeUnion.CtrlRec.TypeUnion.Digital.Default	SAHPI_CTRL_STATE_OFF
RdrTypeUnion.CtrlRec.DefaultMode.Mode	SAHPI_CTRL_MODE_MANUAL
RdrTypeUnion.CtrlRec.DefaultMode.ReadOnly	SAHPI_TRUE
RdrTypeUnion.CtrlRec.WriteOnly	SAHPI_FALSE
RdrTypeUnion.CtrlRec.Oem	Assigned by HPI implementation.
RdrTypeUnion.CtrlRec.IdString	MOTHPI_CTRL_NAME_HPI_RESTART_DAEMON

Table 4-32 HPI Restart Daemon State

SaHpiCtrlStateT	Value
Type	SAHPI_CTRL_TYPE_DIGITAL
StateUnion.Digital	SAHPI_CTRL_STATE_PULSE_ON | SAHPI_CTRL_STATE_OFF

4.5.11 IPMI Command Control

IPMI command control is used to execute IPMI commands on a given resource id. This control is created for all the FRUs in the shelf. hpiipmi client application makes use of this control to get the IPMI commands executed.

Table 4-33 IPMI Command RDR

SaHpiRdrT	Value
RecordId	Assigned by HPI implementation
RdrType	SAHPI_CTRL_RDR
Entity	Entity path of the FRU Ex: {ADVANCEDTCA_CHASSIS,6}{FAN_TRAY_SLOT,1}{COOLING_UNIT,1}
RdrTypeUnion	Defined in next table
IdString	MOTHPI_CTRL_NAME_BOOT_OPTION

Table 4-34 IPMI Command Control

SaHpiCtrlRecT	Value
Num	MOTHPI_CTRL_NUM_IPMI_COMMAND
Type	SAHPI_CTRL_TYPE_OEM
OutputType	SAHPI_CTRL_OEM
TypeUnion.Oem.MId	OHHPI_MANUFACTURER_ID_MOTOROLA
TypeUnion.Oem.ConfigData	
TypeUnion.Oem.Default.MId	OHHPI_MANUFACTURER_ID_MOTOROLA
TypeUnion.Oem.Default.BodyLength	0
TypeUnion.Oem.Default.Body	0

Table 4-34 IPMI Command Control (continued)

SaHpiCtrlRecT	Value
DefaultMode.Mode	SAHPI_CTRL_MODE_MANUAL
DefaultMode.ReadOnly	SAHPI_TRUE
WriteOnly	SAHPI_FALSE
Oem	0

Table 4-35 IPMI Command State

SaHpiCtrlStateT	Value
Type	SAHPI_CTRL_TYPE_OEM
StateUnion.Oem.MId	OHHPI_MANUFACTURER_ID_MOTOROLA
StateUnion.Oem.BodyLength	IPMI command request/response length (max - 255) Ex: When we issue a `Get Device ID` command like, `hpiipmi -d 1 -r 43 0 6 1` BodyLength would be 3 (0 6 1) For the response, the BodyLength would be 16 bytes
StateUnion.Oem.Body	IPMI command request/response bytes Request - all 0's Response - ipmi command response bytes

4.5.12 Boot Bank Control

Boot Bank control is used to switch the boot bank of Motorola/SMART EC specific dual-flash-bank boards. This control is created for all front blades, AMCs, RTMs, and Shelf Managers. hpibootbanks client application makes use of this control to set/get boot bank parameters.

Table 4-36 Boot Bank RDR

SaHpiRdrT	Value
RecordId	Assigned by HPI implementation
RdrType	SAHPI_CTRL_RDR
Entity	Entity path of the managing FRU which owns the control Ex: {ADVANCEDTCA_CHASSIS,6}{SHELF_MANAGER_SLOT,1}{SHELF_MANAGER,1}

Table 4-36 Boot Bank RDR (continued)

SaHpiRdrT	Value
RdrTypeUnion	Defined in next table
IdString	MOTHPI_CTRL_NAME_BOOT_BANK

Table 4-37 Boot Bank Control

SaHpiCtrlRecT	**Value**
Num	MOTHPI_CTRL_NUM_BOOT_BANK
Type	SAHPI_CTRL_TYPE_OEM
OutputType	SAHPI_CTRL_OEM
TypeUnion.Oem.MId	OHHPI_MANUFACTURER_ID_MOTOROLA
TypeUnion.Oem.ConfigData	
TypeUnion.Oem.Default.MId	OHHPI_MANUFACTURER_ID_MOTOROLA
TypeUnion.Oem.Default.BodyLength	0
TypeUnion.Oem.Default.Body	0 (255 times)
DefaultMode.Mode	SAHPI_CTRL_MODE_MANUAL
DefaultMode.ReadOnly	SAHPI_FALSE
WriteOnly	SAHPI_FALSE
Oem	0

Table 4-38 Boot Bank State

SaHpiCtrlStateT	**Value**
Type	SAHPI_CTRL_TYPE_OEM
StateUnion.Oem.MId	OHHPI_MANUFACTURER_ID_MOTOROLA
StateUnion.Oem.BodyLength	2 (max - 255)
StateUnion.Oem.Body	For Set or Get, the body will be Offset 0 - processor id Offset 1 - bank id

4.5.13 SOL Configuration Control

SOL configuration control is used to configure SOL parameters. This control can be used to configure an IPMI user, user privileges, LAN channel parameters, and SOL parameters. `hpisol` example client application makes use of this control to set/get SOL parameters. Refer to section *Section 4.7, Serial Over LAN Configuration on page 87*, for more information on SOL configuration.

Table 4-39 SOL Configuration RDR

SaHpiRdrT	Value
`RecordId`	Assigned by HPI implementation
`RdrType`	`SAHPI_CTRL_RDR`
`Entity`	Entity path of the managing FRU which owns the control Ex:`{ADVANCEDTCA_CHASSIS,6}{PHYSICAL_SLOT,1}{PICMG_FRONT_BLADE,1}`
`RdrTypeUnion`	Defined in next table
`IdString`	`MOTHPI_CTRL_NAME_SOL_CONFIG`

Table 4-40 SOL Configuration Control

SaHpiCtrlRecT	Value
`Num`	`MOTHPI_CTRL_NUM_SOL_CONFIG`
`Type`	`SAHPI_CTRL_TYPE_OEM`
`OutputType`	`SAHPI_CTRL_OEM`
`TypeUnion.Oem.MId`	`OHHPI_MANUFACTURER_ID_MOTOROLA`
`TypeUnion.Oem.ConfigData`	
`TypeUnion.Oem.Default.MId`	`OHHPI_MANUFACTURER_ID_MOTOROLA`
`TypeUnion.Oem.Default.BodyLength`	`0`
`TypeUnion.Oem.Default.Body`	`0 (255 times)`
`DefaultMode.Mode`	`SAHPI_CTRL_MODE_MANUAL`
`DefaultMode.ReadOnly`	`SAHPI_FALSE`
`WriteOnly`	`SAHPI_FALSE`
`Oem`	`0`

Table 4-41 SOL Configuration State

SaHpiCtrlStateT	Value
Type	SAHPI_CTRL_TYPE_OEM
StateUnion.Oem.MId	OHHPI_MANUFACTURER_ID_MOTOROLA
StateUnion.Oem.BodyLength	18 (max - 255)
StateUnion.Oem.Body	Parameters 0(SetInProgress) 1(SOL Enable) 2(SOL authentication) 3(Character accumulate interval and character send threshold) 4(SOL Retry) 5(SOL non-volatile bit rate) 6(SOL volatile bit rate) 7(SOL payload channel) 8(SOL payload port) For more information, refer to the *IPMIv2 spec, section 26, SOL commands, Table 26-5, SOL Configuration Parameters*.

4.6 Cooling Management

As mentioned earlier, the HPI daemon software of the ATCA-MF106/SAM1411 performs shelf management functions. One of the most important functions is cooling management. Details are given in the following sections.

4.6.1 Cooling Management Performed by ATCA-MF106/SAM1411

Cooling management as performed by the ATCA-MF106/SAM1411 HPI daemon software includes the monitoring of operating temperatures within a shelf, and depending on the data obtained, controlling the cooling fans in order to ensure proper cooling of all components in a shelf. Apart from proper cooling, another important goal of cooling management is to keep the noise caused by the fans to a minimum. This is important to meet NEBS requirements.

4.6.1.1 Basic Functionality

For the cooling management, the software considers all IPMI temperature sensors in the shelf. This includes sensors on FRUs, such as blades and AMC modules, but also inlet/outlet sensors of the shelf. The latter are combined to a single virtual difference sensor.

After system start-up, the fans are operated at minimum speed and the shelf management software starts to monitor all temperature sensors.

As soon as a sensor reaches or exceeds the upper minor threshold, the software initializes and starts a virtual Proportional-Integral-Derivative (PID) controller for the respective sensor. The PID uses the current sensor temperature as input and the fan control level as output. In the case of the virtual difference sensor, the input is a predefined air difference temperature profile. The algorithm used within the PID keeps the measured temperature at the upper minor threshold.

If more than one sensor has reached the temperature warning level and therefore several PIDs are active at the same time, the software combines the several outputs of the PIDs into one output which is used to control the fans.

Whenever one or more temperature sensors have reached the upper non-recoverable level, the fan speed is always set to the maximum, i.e. 100%.

Whenever one or more sensors have reached the upper critical level, the software increases the fan speed by a certain percentage, calculated by an internal algorithm. This can typically be observed as a sudden increase of the fan noise.

Configuration option: The configuration parameter **`Cooling_DeactOnUpperNonRecoverTemp`** forces to power off all blades and drop fan speed after receiving nonrecoverable thermal events from two different resources. This option is available on C2000/C2100 chassis only.

If individual fans fail, the software compensates this by increasing the fan speed of the remaining fans accordingly.

Sensors located on FRUs which have been extracted are no longer considered for the cooling management. Furthermore, if a sensor is temporarily not accessible, the software freezes its last state and resumes the sensor control based on the frozen data as soon as the connection has been reestablished.

4.6.1.2 Configuring the Cooling Management

The behavior of the cooling management can be configured via entries in the configuration file `/etc/bbs-hpib/bbs-hpib.conf`. All entries which are related to the cooling management start with the prefix `Cooling_`.

The cooling system can be influenced by the parameters listed in the *Table 4-42*.

Reboot the Shelf Manager or restart the HPI daemon with `sv_stop` and `sv_start` commands in order to activate the modified configuration. Assure that the active Shelf Manager works with the modified configuration file.

Table 4-42 Parameters of Cooling System

Parameter Name	Default Value	Description
`OPENHPI_LOG_DEBUG_FACILITIES`	“”	"ipmicooling" enables logging. It enables the shelf manager to log the cooling related information into files `/var/log/bbs-hpib0X.log` where `X` can be in the range of `0` to `OPENHPI_LOGFILE_MAX` as mentioned in the `/etc/bbs-hpib/bbs-hpib.conf` configuration file.
`Cooling_DeactOnUpperNonRecoverTemp` <valid only for C2000/C2100 systems>	“1”	Enables the function to power off all blades and drop fan speed after receiving upper non recoverable (UNR) thermal events from two different resources in the C2000/C2100 shelf.
`Cooling_FanMinimumLevel`	“12”	Default fan speed in %.
`Cooling_InletOutletLowerTemperature`	“20”	Lower temperature point for inlet/outlet control
`Cooling_InletOutletUpperTemperature`	“40”	Upper temperature point
`Cooling_InletOutletLowerSetPoint`	“10”	Set point at upper temperature
`Cooling_InletOutletUpperSetPoint`	“25”	Set point at lower temperature Range 25 to 15, most effect at 15. Influences the difference between in- and outlet temperature.
`Cooling_InltOutltMinimumSensorTemperature`	“18”	Below this inlet temperature the inlet/outlet related control is disabled
`Cooling_BoardSetPointOffset`	“0”	The temperature difference to the upper-non-critical Threshold of a blade. E. g. threshold at 85, a 5 would result in holding the temperature at 80. Allowed range is 0 to 5
`Cooling_BoardConsiderHysteresis`	“0”	“1” subtracts a hysteresis from the set point

Table 4-42 Parameters of Cooling System (continued)

Parameter Name	Default Value	Description
`Cooling_BoardMinNumControlled`	"10"	Minimum number of sensors to be polled after a temperature event
`Cooling_BoardMaxNumControlled`	"10"	Maximum number of sensors to be polled Both should not be changed
`Cooling_FanConsiderFailureEvent`	"0"	A "1" treats a fan, which had sent a lower critical event, as broken and would increase the speed of the remaining fans accordingly. Default controls the speed related to measured temperatures only.

If the debug logs are enabled for the cooling algorithm, the HPI-B daemon log files will have the following log messages:

```
04:14:21.732 <debug> {ipmicooling}: 9a.0/0/af: sp=85.0 y=84.5 xd=0.04
xdi=-484.7 xdp=0.008 u=14.3

04:14:21.733 <debug> {ipmicooling}: New fan level 14 driven by 9a.0/0/af

04:14:26.698 <debug> {ipmicooling}: 9a.0/0/af: sp=85.0 y=84.6 xd=0.00
xdi=-484.6 xdp=-0.008 u=14.7

04:14:26.699 <debug> {ipmicooling}: New fan level 15 driven by 9a.0/0/af

04:14:31.699 <debug> {ipmicooling}: 9a.0/0/af: sp=85.0 y=84.5 xd=0.05
xdi=-484.5 xdp=0.009 u=14.2

04:14:21.733 <debug> {ipmicooling}: New fan level 14 driven by 9a.0/0/af
```

Following table describes the way to interpret these log messages.

Table 4-43 HPI-B daemon Log Messages

Code in log message	Name, Description
`sp`	Set Point for the control
`y`	Filtered value of read sensor
`u`	Calculated speed in %
`xd`	Difference of `sp - y` - tolerance
`xdi`	Integral
`xdp`	Differential

The y values are filtered from the read sensor data. In this example the values vary in decimals, whereas the real read sensors may change in the ±1 range.

User can interpret the following message to know the whereabouts of the sensor driving the new fan speeds. For e.g., as per the following log message the fan level 14 is driven by the sensor AF on IPMP address 9a, FRU ID 0, LUN Number 0.

```
04:14:21.733 <debug> {ipmicooling}: New fan level 14 driven by 9a.0/0/af
```

(9a.0.0/af) means (IPMBAddress(9a).FruID(0).LUN(0)/SensorNumber(af))

NOTICE

Damage of Blades and Shelf

The default configuration used for the cooling management provides optimized cooling of the shelf and all blades/FRUs in it. Changing the configuration may lead to poor cooling and thus damage of the shelf and blades/FRUs in it. Therefore we strongly recommend not to change the configuration.

If you decide to change the standard configuration, after all, then this should be done in close coordination with your local SMART EC representative.

4.6.2 External Cooling Management

If you decide not to rely on the ATCA-MF106/SAM1411 cooling management, you can switch the ATCA-MF106/SAM1411 cooling management off. Then you have the following two options to control the fan speed and thus perform cooling management:

- Use a Linux example program
- Access HPI controls

NOTICE

Damage of Shelf and Blades/FRUs

The cooling management as performed by the ATCA-MF106/SAM1411 provides optimized cooling for the shelf and all blades/FRUs in it. It has been adapted and optimized for the shelf. Performing cooling management manually or via (inadequate) external shelf management software may lead to overheating and thus damage of the shelf and blades/FRUs in it.

Perform cooling management only via the ATCA-MF106/SAM1411 software.

4.6.2.1 Toggling Between External and Internal Cooling Management

Although we strongly recommend to rely on the ATCA-MF106/SAM1411 shelf management software for cooling management, in some situations you may want to control the fan speed manually or you may want to use external shelf management software. For this purpose, the internal cooling management can be switched off.

If the internal cooling management is switched off, the ATCA-MF106/SAM1411 cooling management software still monitors temperature sensors and handles all events, but no set operations are actually executed. If the internal cooling management is switched on again, the cooling management software executes all necessary cooling actions according to the current state of the system.

In order to switch on/off the internal cooling management, use the Linux example program `hpicooling` which is delivered with the ATCA-MF106/SAM1411 software. The usage of the example program is as follows.

Description

This service allows you to switch on/off the internal cooling management.

Synopsis

```
hpicooling [-h][-d domain_id] [-m cooling_mode]
```

Parameters

`-h`	help
`-d domain_id`	Use domain `domain_id`
`-m mode`	Set cooling mode 1: Internal cooling management enabled (default) 2: Internal cooling management disabled (cooling must be done manually via HPI controls in this case)

4.6.2.2 Controlling Fan Speed via Linux Example Program

If the internal cooling management is switched off, you can control the fan speed via the Linux example tool `hpifan`. Its usage is described in the following.

Description

Sets the fan speed for all fans of a specified domain.

Synopsis

```
hpifan [-h] [-d domain_id] [-s fan_speed_level]
```

Parameters

`-h`	help
`-d domain_id`	Use domain `domain_id`
`-s speed`	Set fan speed for all fans of the domain.

4.6.2.3 Cooling Management via HPI Controls

If you decide to use your own software for cooling management, then your software can use a virtual cooling resource and fan resources which are defined in each domain. The virtual cooling resource can be used to read the fan speed which the automatic internal cooling algorithm would have computed, whereas the fan resources can be used to control the fan speeds of individual fans in a shelf. There is one fan resource for each fan in a shelf.

The entity path and the ID string of the virtual cooling resource are defined as follows.

Table 4-44 Virtual Cooling Resource - Entity Path and ID String

Property	Value
Entity Path	{ADVANCEDTCA_CHASSIS,<SGA>}{COOLING_UNIT, 0}{COOLING_UNIT,0}
ID String	Cooling Zone

The virtual cooling resource contains one HPI control which reflects the fan speed which the internal cooling algorithm computes for the shelf. The definition of the control is given in the following two tables.

Table 4-45 Virtual Cooling Resource - Fan Control RDR

SaHpiRdrT	Value
RecordId	Assigned by HPI
RdrType	SAHPI_CTRL_RDR
Entity	Cooling resource entity
IsFru	Unused
RdrTypeUnion	CtrlRec See *Table 4-46 on page 85*
IdString	Fan Control

Table 4-46 Virtual Cooling Resource - Fan Control

SaHpiCtrlRecT	Value
Num	MOTHPI_CTRL_NUM_COOLING
OutputType	SAHPI_CTRL_GENERIC
Type	SAHPI_CTRL_TYPE_DISCRETE
TypeUnion.Analog.Min	Minimum blower speed level
TypeUnion.Analog.Max	Maximum blower speed level
TypeUnion.Analog.Default	Default blower speed level
DefaultMode.Mode	SAHPI_MODE_AUTO
DefaultMode.ReadOnly	SAHPI_FALSE
WriteOnly	SAHPI_FALSE

Table 4-47 Virtual Cooling Resource - Fan Control States

SaHpiCtrlStateT	Value
Type	SAHPI_CTRL_TYPE_ANALOG
StateUnion.Analog	Current fan level calculated by cooling algorithm. This can always be read, independent of the cooling mode (internal/external).

As mentioned above, each domain furthermore contains HPI resources which represent fans in a shelf. There is one fan resource for each fan. The fan resources allow you to control the fan speeds of individual fans. The entity paths and ID strings of these resources are given in the following table. The control mode of an individual fan cannot be changed, it will always be the same mode as the mode of the virtual cooling control.

Table 4-48 Fan Resource - Entity Path and ID String

Property	Entity Path
Entity Path	{ADVANCEDTCA_CHASSIS, <SGA>}{FAN_TRAY_SLOT, <fan num.>}{COOLING_UNIT,0}
ID String	FAN MODULE

The RDR definitions are given in the following two tables.

Table 4-49 Fan Resource - RDR

SaHpiRdrT	Value
RecordId	Assigned by HPI
RdrType	SAHPI_CTRL_RDR
Entity	Fan resource entity
IsFru	Unused
RdrTypeUnion	CtrlRec See *Table 4-50 on page 86*
IdString	Fan 0 Control

Table 4-50 Fan Resource - Control

SaHPICtrlRecT	Value
Num	MOTHPI_CTRL_NUM_FAN
OutputType	SAHPI_CTRL_FAN_SPEED
Type	SAHPI_CTRL_TYPE_ANALOG
TypeUnion.Analog.Min	Minimum blower speed level
TypeUnion.Analog.Max	Maximum blower speed level
TypeUnion.Analog.Default	Default blower speed level
DefaultMode.Mode	SAHPI_CTRL_MODE_AUTO
DefaultMode.ReadOnly	SAHPI_TRUE
WriteOnly	SAHPI_FALSE

Table 4-51 Fan Resource - Control State

SaHpiCtrlStateT	Value
Type	SAHPI_CTRL_TYPE_ANALOG
StateUnion.Analog	Current blower speed level

4.7 Serial Over LAN Configuration

The ATCA-MF106 and SAM-1411 has the capability to automatically configure the Serial Over LAN (SOL) feature as specified for IPMI V2.0. The SOL feature, for example, allows you to view the boot process and the BIOS settings of a blade and to change BIOS settings if necessary.

Whenever a front blade is discovered in the shelf, the ATCA-MF106/SAM-1411 checks whether the blade supports SOL. If it does support SOL, the ATCA-MF106/SAM-1411 performs all configuration steps necessary to use the SOL feature.

This includes the following:

1. Configure an IPMI user
2. Configure user privileges
3. Configure LAN channel parameters
4. Configure SOL parameters

The most significant parameters to configure are the IP addresses of the LAN channels. These are calculated according to the following formula:

LAN channel IP address = <shelf IP address> +<LAN channel> * <configurable multiplier> + 0.0.0.<10*logical slot> + <configurable offset>

Example:

```
Shelf IP address    = 172.16.3.0
logical slot        = 14
LAN channel         = 1
configurable offset = 0.0.100.0

                          172.16.  3.  0

            +               0. 1.  0.  0

            +               0. 0.  0.140

            +               0. 0.100.  0

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

LAN channel IP address = 172.17.103.140
```

Most of the other parameters used during SOL configuration can be configured in the HPI configuration file `/etc/bbs-hpib/bbs-hpibd.conf` on the ATCA-MF106/SAM-1411. Entries to the configuration file are optional. If for a parameter there is no entry in the configuration file, a built-in default value is used. The following table gives details on all configurable parameters:

Table 4-52 Configurable SOL Parameters

Keyword	Description	Allowed Values	Default in Config. File	Built-in Default
Sol_MgmtEnable	1: enables SOL configuration 0: disables SOL configuration	0, 1	1	enabled
Sol_UserId	User ID used for SOL	2-4		2
Sol_UserName	User name used for SOL	String of length 1-16	soluser name	Administrator
Sol_Password	Password used for the SOL user	String of length 1-20	soluserpassword	Administrator
Sol_UserPrivilegeLimit	Privilege limit for the user created for SOL	CALLBACK, USER		ADMINISTRATOR
Sol_ForcePayloadEncryption	1: configures SOL to enforce payload encryption 0: lets the remote software control encryption	0, 1		Do not enforce payload encryption
Sol_ForcePayloadAuthentication	1: configures SOL to enforce payload authentication 0: lets the remote software control authentication	0, 1		Do not enforce payload authentication
Sol_SolPrivilegeLevel	Privilege level that is required to activate SOL	USER, OPERATOR, ADMINISTRATOR		OPERATOR
Sol_CharAccumulate-Interval	Character accumulate interval [ms]	5-1275		40

Table 4-52 Configurable SOL Parameters (continued)

Keyword	Description	Allowed Values	Default in Config. File	Built-in Default
Sol_CharSend-Threshold	Character send threshold	1-255		50
Sol_Retry	Retry count	0-7		7
Sol_RetryInterval	Retry interval [ms]	0-2550		100
Sol_VolatileBitRate	Volatile serial bit rate [baud]	9600, 19200, 38400, 57600, 115200		9600
Sol_NonVolatileBit-Rate	Non-volatile serial bit rate [baud]	9600, 19200, 38400, 57600, 115200		9600
Sol_AddressMultiplier	IP address multiplier	any valid IP address		0.1.0.0
Sol_AddressOffset	IP address offset	Any valid IP address	0.0.100.0	0.0.100.0
Sol_GratuitousArp-Interval	Gratuitous ARP send Interval [s]. If '0' is specified, regular ARP responses are sent instead on gratuitous ARPs	0-127	3	3
Sol_Ch1_GatewayIp	IP address of gateway for LAN channel 1	Any valid IP address	172.17.166.55	0.0.0.0
Sol_Ch1_Gateway-Mac	MAC address of gateway for LAN channel 1	Any valid MAC address	00:0A:5E:1E:92:7C	00:00:00:00:00:00
Sol_Ch1_Backup-GatewayIp	IP address of backup gateway for LAN channel 1	Any valid IP address		0.0.0.0
Sol_Ch1_Backup-GatewayMac	MAC address of backup gateway for LAM channel 1	Any valid MAC address		00:00:00:00:00:00

Table 4-52 Configurable SOL Parameters (continued)

Keyword	Description	Allowed Values	Default in Config. File	Built-in Default
Sol_Ch2_GatewayIp	IP address of gateway for LAN channel 2	Any valid IP address	172.18.166.55	0.0.0.0
Sol_Ch2_Gateway-Mac	MAC address of gateway for LAN channel 2	Any valid MAC address	00:0A:5E:1E:92:7C	00:00:00:00:00:00
Sol_Ch2_Backup-GatewayIp	IP address of backup gateway for LAN channel 2	Any valid IP address		0.0.0.0
Sol_Ch2_Backup-GatewayMac	MAC address of backup gateway for LAM channel 2	Any valid MAC address		00:00:00:00:00:00
Sol_Config	0: Default IP address assignment calculation (as described in Section 4.6 of this guide.) 1:IP Address assignment as per the user configured Base IP address, subnet mask, and with a multiplication factor to derive the fourth octet of the IP Address. It is calculated as: Sol_Ch<1|2>_Base_Address + (slotnumber * Sol_AddressMultiplier)	0, 1	0	0
Sol_Ch1_Base_Address	Base Address for the first LAN Channel	Any Valid IP Address	192.91.191.66	0.0.0.0
Sol_Ch2_Base_Address	Base Address for the second LAN Channel	Any Valid IP Address	192.91.191.98	0.0.0.0
Sol_Ch1_Subnet_mask	Subnet mask for the first LAN Channel	Any Valid IP Address	192.91.191.95	0.0.0.0

Table 4-52 Configurable SOL Parameters (continued)

Keyword	Description	Allowed Values	Default in Config. File	Built-in Default
Sol_Ch2_Subnet_mask	Subnet mask for the second LAN Channel	Any Valid IP Address	192.91.191.127	0.0.0.0
Sol_Multiplier	Multiplier by which the slot number is multiplied and added to the fourth octet of the Base_Address	Any positive integer	2	0
Sol_Slot_Address	0 Use Logical Slot numbers 1 Use Physical Slot numbers for SOL configuration	0, 1	1	1

The open source tool `ipmitool` can be used as SOL client. Example usage:

```
ipmitool -I lanplus -H 172.17.103.140 -L OPERATOR -U solusername -P
soluserpassword sol activate
```

4.8 Using the Firmware Update Management Instrument

The SMART EC HPI-B distribution supports the Firmware Update Management Instrument (FUMI) feature as defined in the SAI-HPI-B-02.01 Service Availability Forum Hardware Platform Interface specification.

FUMI allows to programmatically update the following software types on the following FRUs within the Centellis 2000/C2100/4411 environment.

Table 4-53 Firmware Update Management Instrument-Supported Image Types and Blades/FRUs

Image Type	Upgradeable by FUMI on the following Blades/FRUs
IPMC/MMC firmware	ATCA-7221 ATCA-MF106 ATCA-7107 SAM1411 All ATCA boards and AMCs/RTMs supporting HPM.1 upgrade via IPMI

A detailed description of FUMI is beyond the scope of this manual. Refer to the FUMI section in SAI-HPI-B-02.01 specification for detailed description. For release-specific restrictions of the FUMI functionality, refer to the system release notes, which are delivered together with your system.

When upgrading blades or other FRUs, make sure to use only images provided by SMART EC and make sure that the image types and versions are suitable for the blade or FRU which you want to upgrade. You should use the images in tar format which contains the upgrade image itself and some metainfo.

When upgrading IPMC firmware images, OEM FRU data records are by default preserved during the upgrade. Whether OEM FRU data records are preserved or not, can be configured in the configuration file `/etc/bbs-hpib/bbs-hpib.conf` in the section ipmidirect. The entry is: `FumiPreserveOemrecords = yes/no (yes = default).`

In order to get started with FUMI, you may want to use the `hpifumi` example tool, which is delivered as part of the HPI-B client RPMs. Just like the other example applications (see *Appendix A, Example Applications*), `hpifumi` is invoked via the command line and is controlled via command line parameters. It internally invokes FUMI function calls. A detailed description of `hpifumi` is given below.

Description

`hpifumi` allows to invoke underlying FUMI function calls via the command line interface, and thus update software images via the command line interface.

Synopsis

```
hpifumi [-d DOMAIN_ID] [-r RESOURCE_ID] [-f FUMI_NUM] [-b BANK_NUM]
[-n COMPONENT_ID ][-Z Rollback_disable ][-s URI ] [-u] [-v] [-i]
[-t] [-a] [-g] [-y] [-c] [-V] [-I] [-D] [-C] [-T] [-P] [-L] [-M]
[-G] [-R] [-A] [-U] [-h]
```

Parameters

`-d DOMAIN_ID`	HPI-B domain ID of the FRU whose firmware you want to update.
`-r RESOURCE_ID`	Resource ID of the FRU whose firmware you want to update.
`-f FUMI_NUMBER`	FUMI number which identifies the memory resource (for example boot flash or user flash) which you want to update.
`-b BANK_NUM`	Bank number. This identifies which memory device, out of several redundant memory devices you want to update. It allows to select for example the stand-by memory device or the currently active. (not supported by SMART EC FRUs/blades released in the Centellis 2000/C2100/4411 context)

-s URI	Invokes the underlying FUMI call `saHpiFumiSourceSet()`. Supply the URI of the source image file. See the FUMI specification for details. **NOTE: This parameter supports only the file protocol. This means URIs must be given in the following form: file://<absolute path of file>** **Any other protocols, such as FTP for example, are not supported.**
`-u`	Invokes the FUMI `saHpiFumiUpgradeStart()` call. This starts the update process.
`-v`	Invokes the FUMI `saHpiFumiSourceInfoValidateStart()` call. This checks the validity of the new firmware image.
`-i`	Invokes the FUMI `saHpiFumiSourceInfoGet()` call. This command is invoked after the source image has been set and validated. It can be used to retrieve information such as version number, build date and similar information. Refer to the FUMI specification for further details.
`-t`	Invokes the FUMI call `saHpiFumiTargetInfoGet()`. This function can be used to retrieve information about the target image file. This information includes version number, build date etc.
`-a`	This invokes the FUMI call `saHpiFumiActivate()`. This starts the execution of the active image on the selected FUMI. **NOTE: If you are upgrading IPMC firmware images on the currently supported blades via FUMI, then the IPMC firmware is implicitly activated and the -a command is used to trigger a FRU state change from M1 (Inactive) to M2 (Insertion pending).**
`-g`	This invokes the FUMI call `saHpiFumiUpgradeStatusGet()`. This should be called after the update process has been started. It returns the update status. Refer to the FUMI specification for possible states.
`-y`	This invokes the FUMI call `saHpiFumiTargetVerify()`. This call is intended to be invoked after the upgrade has finished. It verifies the new, updated image at the target memory matches with the source image. Refer to the FUMI specification for details.

`-c`	This invokes the FUMI call `saHpiFumiUpgradeCancel()`. This command is currently not supported.
`-V`	This invokes the version number of the hpifumi command itself.
`-h`	This displays help information, which includes parameter description and overall functionality description.
`-n COMPONENT_ID`	The component identifier of the component for which the information is required.
`-Z Rollback_disable`	A Boolean value representing the value you want to set for the auto rollback status.
`-I`	Spec information for the underlying IPMC. Invokes the API `saHpiSpecInfoGet()`.
`-D`	This provides service impact information while in case an IPMC is upgraded, there will be impact on the Service provided by IPMC or not. Invokes the API `saHpiFumiServiceImpactGet()`.
`-C`	Source component Information, after the upgrade image has been validated you can view the source component information provided by image. Invokes the API `saHpiFumiSourceComponentInfoGet()`.
`-T`	Target IPMC component information, component ID option can be used to request the specific component information. Invokes the API `saHpiFumiTargetComponentInfoGet()`.
`-P`	Logical target information provides the rollback and pending version information of the targeted IPMC. Invokes the API `saHpiFumiLogicalTargetInfoGet()`.
`-L`	Logical target component information provides the rollback and pending version information of the various components present in the target. Invokes the API `saHpiFumiLogicalTargetComponentInfoGet()`.
`-M`	Verify target main instance, checks the upgrade image with the main instance present on the target. Source set and source validation steps need to be done before target verify main. Invokes the API `saHpiFumiTargetVerifyMainStart()`.
`-G`	Auto rollback disable status get. Invokes the API `saHpiFumiAutoRollbackDisableGet()`.

`-R`	Auto rollback disable status set. Invokes the API `saHpiFumiAutoRollbackDisableSet()`.
`-A`	Activate start the new installed image. Invokes the API `saHpiFumiActivateStart()`.
`-U`	Clean up and reset the upgrade activity steps performed. The upgrade status will be set the no operation and source image will be deleted. Invokes the API `saHpiFumiCleanup()`.

Example

Procedure for Updating the IPMC Firmware of HPM.1 Compliant Blade via the hpifumi Example Tool

In order to upgrade the IPMC firmware of HPM.1 compliant blade, proceed as follows.

1. Copy the new firmware image file to a directory on the blade where the `hpifumi` example application runs.
2. Identify the FUMI object which represents the IPMC of the blade. This can be achieved by entering **hpifumi** without any parameters. All available FUMI objects will then be displayed in a way similar to the following example output.

```
resource: 65 ATCA-MF106:
{ADVANCEDTCA_CHASSIS,6}{SHELF_MANAGER_SLOT,1}{SHELF_MANAGER,1}
  Fumi 0
    IdString   : IPMC
    AccessProt : SAHPI_FUMI_PROT_LOCAL
    Capability :
    NumBanks   : 1
    Oem        : 0x00100000

resource: 70 ATCA-MF106:
{ADVANCEDTCA_CHASSIS,6}{SHELF_MANAGER_SLOT,2}{SHELF_MANAGER,2}
  Fumi 0
    IdString   : IPMC
    AccessProt : SAHPI_FUMI_PROT_LOCAL
    Capability :
    NumBanks   : 1
    Oem        : 0x00120000
```

```
resource: 77 ATCA-9305:
{ADVANCEDTCA_CHASSIS,6}{PHYSICAL_SLOT,1}{PICMG_FRONT_BLADE,1}
  Fumi 0
    IdString   : IPMC
    AccessProt : SAHPI_FUMI_PROT_LOCAL
    Capability :  SAHPI_FUMI_CAP_ROLLBACK
SAHPI_FUMI_CAP_COMPONENTS  SAHPI_FUMI_CAP_AUTOROLLBACK
SAHPI_FUMI_CAP_AUTOROLLBACK_CAN_BE_DISABLED
    NumBanks   : 0
    Oem        : 0x009a0000
```

3. In the FUMI object which represents the blade IPMC firmware, identify the FUMI number (this is the number after the string "`Fumi`") and the resource number (this is the number that follows after the string "`resource:`").
4. Using the previously obtained resource and FUMI number, enter the following command:
   `hpifumi -r<resource number> -f<FUMI number> -s file:<path of new firmware image>`
5. Validate the source image by entering: **`hpifumi -v -r <res-id> -f <fumi-num>`**
6. Enter the following command to display version information about the new root file system image file
   `hpifumi -i -r <res-id> -f <fumi-num>`
7. Enter the following command to start the update:
   `hpifumi -u -r <res-id> -f <fumi-num>`
8. In order to check if the upgrade was successful, invoke the following command:
   `hpifumi -g -r <res-id> -f <fumi-num>`

   A message is displayed which describes the upgrade status.
9. If the update was successful, you may want to display version information of the newly installed root file system image file. To do so, enter the following command:
   `hpifumi -t -r <res-id> -f <fumi-num>`
10. Activate the new, updated image by entering any one of the following:
    `hpifumi -a -r <res-id> -f <fumi-num>`

    or

    `hpifumi -A -r <res-id> -f <fumi-num>`

Example Applications

The HPI-B client base package contains precompiled example applications. They are invoked via the command line and can be configured via command line parameters. Each example application illustrates a certain feature of HPI-B and makes use of the respective HPI-B function calls.

After extracting the HPI-B client RPMs, the example applications can be found in the following directory: `/opt/bladeservices/bin`. You can obtain information about the command usage by invoking the application from the command line and providing `-h` as parameter.

The HPI-B example programs are provided "as is" without any warranty of any kind, either express or implied. The entire risk as to the quality, operability and execution of the programs is with you. Should the programs prove to be faulty or incorrect, you assume the cost of all necessary servicing, repair or correction. In no event SMART EC will be liable to you for any damages, any lost profits or other special, incidental or consequential damages arising out of the use or inability to use the programs.

SMART EC reserves the right to revise or remove the programs in subsequent releases without obligation of SMART EC to notify any person of such revision or changes.

A.1 Example Application Source Files

In order to ease application development and help you to get familiar with the HPI-B API usage, SMART EC provides the source files of the example applications and an example make file. These files are available as different RPM files, depending on the operating system and CPU architecture. The naming scheme used for the RPMs is: `bbs-hpib-clientsrc-<version>-1.<CPU architecture>-<distribution>-<os>.rpm`

After installing the RPMs, the source files and the example make file are located in the following directory: `/opt/bladeservices/src/bbs-hpib/clients`.

A.2 List of Supported Example Applications

The following is an automatically generated output (based on the files contained in `/opt/bladeservices/bin` and the output obtained via the `-h` parameter) that describes all HPI example applications which were available when this manual was written and the usage of these applications.

Note that in the meantime further example applications may have been added or the functionality of existing HPI example applications may have been changed slightly. The current usage and functionality can always be obtained by invoking the example application with the `-h` parameter.

```
#
# Lists the usage of all supported HPI-B example programs.
#
# Copyright (c) 2007 by Motorola GmbH
# Copyright (c) 2008, Emerson Network Power - Embedded Computing GmbH
#
#------------------------------#
# hpiautotimer
#------------------------------#
Usage: hpiautotimer [OPTION]...
HPI example application to manage the timeout values of the auto insert
timer and auto extract timer.
Options:
  -d DOMAIN_ID        use domain with id DOMAIN_ID
  -D                  walk recursively through DRT
  -r RESOURCE_ID      use resource with id RESOURCE_ID
  -i INSERT_TIMEOUT   set auto insert timeout value in msec to
INSERT_TIMEOUT
  -e EXTRACT_TIMEOUT  set auto extract timeout value in msec to
EXTRACT_TIMEOUT
  -V                  print version information and exit
  -h                  display this help and exit
#------------------------------#
# hpibootbanks
#------------------------------#
Usage: hpibootbanks [OPTION]...
HPI example application to switch the boot bank of Motorola/Emerson
specific dual-flash-bank boards.
Options:
  -d DOMAIN_ID    use domain with id DOMAIN_ID
```

```
  -r RESOURCE_ID  use resource with id RESOURCE_ID
  -b BANK_NUM     set boot bank number to BANK_NUM [0..<bankNumber>]
  -p PROCESSOR_ID set boot bank of procesor with this ID
  -V              print version information and exit
  -h              display this help and exit
#------------------------------#
# hpibootoptions
#------------------------------#
Usage: hpibootoptions [OPTION]... [PARAM# PARAMS]
HPI example application to manage the system boot options defined in IPMI
v2.0 .
Options:
  -d DOMAIN_ID    use domain with id DOMAIN_ID
  -r RESOURCE_ID  use resource with id RESOURCE_ID
  -c              turn on console redirection
                  (Not applicable on some boards)
  -o              turn off console redirection
                  (Not applicable on some boards)
  -x              perform FRU cold-reset
  -V              print version information and exit
  -h              display this help and exit
Examples:
Set the system boot option 1 (service partition) to value 2 for resource
4 on domain 0:
  hpibootoptions -d 0 -r 4 1 2
Turn on console redirection and perform a cold-reset for resource 34 on
domain 0:
  hpibootoptions -d 0 -r 34 -c -x
#------------------------------#
```

```
# hpibootparameter
#-----------------------------#
Usage: hpibootparameter [OPTION]... [PARAM# PARAMS]
HPI example application to manage IPMI boot parameter.
Options:
  -d DOMAIN_ID    use domain with id DOMAIN_ID
  -r RESOURCE_ID  use resource with id RESOURCE_ID
  -g              get IPMI boot parameter USER area
  -b              get IPMI boot parameter DEFAULT area
  -s filename     set IPMI Boot parameter, read from file
  -c              clear IPMI Boot parameter stored in USER area
  -x              additionally output the IPMI boot parameter as hexdump
  -V              print version information and exit
  -h              display this help and exit
#-----------------------------#
# hpichassisstatus
#-----------------------------#
Usage: hpichassisstatus [OPTION]...
HPI example application to display the control state of the chassis status
control.
Options:
  -d DOMAIN_ID        use domain with id DOMAIN_ID
  -D                  walk recursively through DRT
  -r RESOURCE_ID      use resource with id RESOURCE_ID
  -V                  print version information and exit
  -h                  display this help and exit
#-----------------------------#
# hpicooling
#-----------------------------#
```

```
Usage: hpicooling [OPTION]...
HPI example application to control the cooling mode.
Note: Only applicable if the HPI daemon runs in Shelf Manager mode.
Options:
  -d DOMAIN_ID    use domain with id DOMAIN_ID
  -D              walk recursively through DRT
  -r RESOURCE_ID  use resource with id RESOURCE_ID
  -m mode         set cooling mode [1=AUTO, 2=MANUAL]
  -V              print version information and exit
  -h              display this help and exit
#------------------------------#
# hpidomain
#------------------------------#
Usage: hpidomain [OPTION]...
HPI example application to display all domains found.
Options:
  -V   print version information and exit
  -h   display this help and exit
#------------------------------#
# hpidomainel
#------------------------------#
Usage: hpidomainel [OPTION]...
HPI example application to display the domain event log.
Options:
  -d DOMAIN_ID  use domain with id DOMAIN_ID
  -c            clear the event log
  -A            display everything
  -t            display RDR with the event log
```

```
  -p            display RPT with the event log
  -x            display debug messages
  -V            print version information and exit
  -h            display this help and exit
#------------------------------#
# hpidomainself
#------------------------------#
Usage: hpidomainself [OPTION]...
HPI example application to print the domain ID where this program is
running on.
Note: Requires multishelf library.
Options:
  -V   print version information and exit
  -h   display this help and exit
#------------------------------#
# hpifailedextract
#------------------------------#
Usage: hpifailedextract [OPTION]...
HPI example to remove a failed resource using the failed extract control.
Options:
  -d DOMAIN_ID    use domain with id DOMAIN_ID
  -r RESOURCE_ID  use resource with id RESOURCE_ID to extract
  -a           display all resources (default: list only failed resources)
  -V              print version information and exit
  -h              display this help and exit
#------------------------------#
# hpifan
#------------------------------#
Usage: hpifan [OPTION]...
```

```
HPI example application to control the cooling mode.
Note: Only applicable if the HPI daemon runs in Shelf Manager mode.
Options:
  -d DOMAIN_ID    use domain with id DOMAIN_ID
  -D              walk recursively through DRT
  -r RESOURCE_ID  use resource with id RESOURCE_ID
  -s level        set cooling level (only in MANUAL mode)
  -V              print version information and exit
  -h              display this help and exit
#------------------------------#
# hpifruactivation
#------------------------------#
Usage: hpifruactivation [OPTION]...
HPI example application to manage the FRU activation mode.
Options:
  -d DOMAIN_ID     use domain with id DOMAIN_ID
  -D               walk recursively through DRT
  -r RESOURCE_ID   use resource with id RESOURCE_ID
  -a {0|1}         disable/enable ShM activation
  -w DELAY         set delay before next power on to DELAY * 1/10 sec
  -V               print version information and exit
  -h               display this help and exit
#------------------------------#
# hpifrudeactivation
#------------------------------#
Usage: hpifrudeactivation [OPTION]...
HPI example application to manage the FRU deactivation mode.
Options:
```

```
  -d DOMAIN_ID     use domain with id DOMAIN_ID
  -D               walk recursively through DRT
  -r RESOURCE_ID   use resource with id RESOURCE_ID
  -a {0|1}         enable/disable ShM deactivation
  -V               print version information and exit
  -h               display this help and exit
#-----------------------------#
# hpifruipmcreset
#-----------------------------#
Usage: hpifruipmcreset [OPTION]...
HPI example application to set FRU IPMC Reset Control actions.
Options:
  -d DOMAIN_ID    use domain with id DOMAIN_ID
  -r RESOURCE_ID  use resource with id RESOURCE_ID
  -c              IPMC cold reset
  -w              IPMC warm reset
  -V              print version information and exit
  -h              display this help and exit
#-----------------------------#
# hpifruresetdiag
#-----------------------------#
Usage: hpifruresetdiag [OPTION]...
HPI example application to set FRU Reset and Diagnostic Control actions.
Options:
  -d DOMAIN_ID    use domain with id DOMAIN_ID
  -r RESOURCE_ID  use resource with id RESOURCE_ID
  -s VALUE        set control value VALUE
                  [1=Graceful Reboot, 2=Diagnostic Interrupt]
```

```
  -V              print version information and exit
  -h              display this help and exit
#-----------------------------#
# hpifumi
#-----------------------------#
Usage: hpifumi [OPTION]...
HPI example application to manage the firmware upgrade of FRUs using FUMI.
Options:
  -d DOMAIN_ID    use domain with id DOMAIN_ID
  -r RESOURCE_ID  use resource with id RESOURCE_ID
  -f FUMI_NUM     use fumi with number FUMI_NUM
  -b BANK_NUM     use bank with number BANK_NUM
  -n COMPONENT_ID use Component with number COMPONENT_ID
  -Z Rollback_disable use value for rollback disable 0/1
  -s URI          calls saHpiFumiSourceSet URI
  -u              calls saHpiFumiInstallStart
  -v              calls saHpiFumiSourceInfoValidateStart
  -i              calls saHpiFumiSourceInfoGet
  -t              calls saHpiFumiTargetInfoGet
  -a              calls saHpiFumiActivate
  -g              calls saHpiFumiUpgradeStatusGet
  -y              calls saHpiFumiTargetVerifyStart
  -c              calls saHpiFumiUpgradeCancel
  -I              calls saHpiFumiSpecInfoGet
  -D              calls saHpiFumiServiceImpactGet
  -C              calls saHpiFumiSourceComponentInfoGet
  -T              calls saHpiFumiTargetComponentInfoGet
  -P              calls saHpiFumiLogicalTargetInfoGet
```

```
  -L              calls saHpiFumiLogicalTargetComponentInfoGet
  -M              calls saHpiFumiTargetVerifyMainStart
  -G              calls saHpiFumiAutoRollbackDisableGet
  -R              calls saHpiFumiAutoRollbackDisableSet
  -A              calls saHpiFumiActivateStart
  -U              calls saHpiFumiCleanUp
  -V              print version information and exit
  -h              display this help and exit
#------------------------------#
# hpiha
#------------------------------#
Usage: hpiha [OPTION]...
HPI example application to show the HA state and to initiate a switch-over.
Options:
  -d <domain ID>  use domain with ID <domain ID>
  -m              initiate a Shelf Manager switch-over
  -s              initiate a HPI Daemon switch-over
  -V              print version information and exit
  -h              display this help and exit
#------------------------------#
# hpihotswap
#------------------------------#
Usage: hpihotswap [OPTION]...
HPI example application to invoke hotswap actions.
Note: If calling without options the application switches to interactive
mode.
Options:
  -d DOMAIN_ID    use domain with id DOMAIN_ID
  -s              show hotswap resources
```

```
  -a RESOURCE_ID  activate resource
  -t RESOURCE_ID  deactivate resource
  -i RESOURCE_ID  insert resource
  -e RESOURCE_ID  extract resource
  -p RESOURCE_ID  get powerstate of resource
  -u RESOURCE_ID  power up resource
  -o RESOURCE_ID  power down resource
  -y RESOURCE_ID  power cycle resource
  -c RESOURCE_ID  cold-reset resource
  -V              print version information and exit
  -h              display this help and exit
#------------------------------#
# hpiidh
#------------------------------#
Usage: hpiidh [OPTION]...
HPI example application to manage the Inventory Data
Options:
  -d DOMAIN_ID    use domain with id DOMAIN_ID
  -r RESOURCE_ID  use resource with id RESOURCE_ID
  -t TARGET       repository target: [[idr][:area][:field]]
  -a              add new custom Area
  -u              add new FORCE User Info Area
  -s STRING       set a Field specified by -t with STRING
  -f FILE         set a Field specified by -t from FILE
  -w FILE         write Field specified by -t to FILE
  -e              erase Area specified by -t
  -x              display binary fields in HEX format
  -V              print version information and exit
```

```
  -h              display this help and exit
#-----------------------------#
# hpiipmb0
#-----------------------------#
Usage: hpiipmb0 [OPTION]...
HPI example application to set the IPMB-A or IPMB-B state control.
Options:
  -d DOMAIN_ID    use domain with id DOMAIN_ID
  -D              walk recursively through the DRT
  -r RESOURCE_ID  use resource with id RESOURCE_ID
  -a {0|1}        isolate/join IPMB-A
  -b {0|1}        isolate/join IPMB-B
  -l LINK_NUM     use link number LINK_NUM when isolating the IPMB
  -V              print version information and exit
  -h              print this help and exit
#-----------------------------#
# hpiipmi
#-----------------------------#
Usage: hpiipmi [OPTION]... LUN NETFN CMD [DATA]
HPI example application to send native IPMI commands using the
Motorola/Emerson specific IPMI control.
LUN, NETFN, CMD and DATA will be interpreted as hexadecimal values
Options:
  -d DOMAIN_ID    use domain with id DOMAIN_ID
  -r RESOURCE_ID  use resource with id RESOURCE_ID
  -V              print version information and exit
  -h              display this help and exit
Example:
```

```
Send GetDeviceId command (Lun=0, Netfn=6, Cmd=1) to resource 43 on domain
1:

  hpiipmi -d 1 -r 43 0 6 1

#------------------------------#

# hpiled

#------------------------------#

Usage: hpiled [OPTION]...

HPI example application to control the LEDs.

Options:

  -d DOMAIN_ID    use domain with id DOMAIN_ID

  -r RESOURCE_ID  use resource with id RESOURCE_ID

  -n CONTROL_ID   use control with id CONTROL_ID for set operations

  -a              set control mode to auto

  -1 RATE         set led on duration rate in 1/100sec [0-255]

  -0 RATE         set led off duration rate in 1/100sec [0-255]

  -t RATE         led test

  -V              print version information and exit

  -h              display this help and exit

#------------------------------#

# hpilink

#------------------------------#

Usage: hpilink [OPTION]...

HPI example application to display E-Keying link states and optionally
listen for link state events.

Options:

  -d DOMAIN_ID    use domain with id DOMAIN_ID

  -r RESOURCE_ID  use resource with id RESOURCE_ID

  -e              listen for E-Keying link state events

  -V              print version information and exit
```

```
  -h              display this help and exit
#-----------------------------#
# hpilist
#-----------------------------#
Usage: hpilist [OPTION]...
HPI example application to list all RPT/RDR entries and to get all events.
Options:
  -d DOMAIN_ID    use domain with id DOMAIN_ID
  -D              walk recursively through DRT
  -r RESOURCE_ID  use resource with id RESOURCE_ID
  -s              output short information (don't display RDR data)
  -y SENSOR_NUM   return value of sensor number SENSOR_NUM of resource
with id RESOURCE_ID
  -z [1/0]        enable/disable events from sensor number SENSOR_NUM of
resource with id RESOURCE_ID
  -t              don't display Timestamps
  -p CSIWDF      display only Control,Sensor,Watchdog,Inventory,Dimi,Fumi
                  (ignored if no valid parameter is given)
  -e              listen for events
  -n              no discover (useful for just getting events)
  -i              interactive mode (for development use only)
  -V              print version information and exit
  -h              display this help and exit
#-----------------------------#
# hpilog
#-----------------------------#
Usage: hpilog [OPTION]...
HPI example application to control the logger.
Options:
```

```
  -d <domain ID>  use domain with ID <domain ID>
  -c <ctrl num>   use control with number <ctrl num>
  -p <properties> set log properties to <properties>
  -q <properties> reset log properties <properties>
  -r <severities> <facilities>
                  reset log <severities> for <facilities>
  -s <severities> <facilities>
                  set log <severities> for <facilities>
  -V              print version information and exit
  -h              display this help and exit
Supported properties:
stdout|stderr|file|simplefile|syslog|prefix|threadid|
Supported severities: debug|info|warning|error|all
Supported facilities:
other|connection|transport|session|plugin|remote|daemon|client|core|ha|h
picall|resource|sensor|control|inventory|watchdog|dimi|fumi|hotswap|sel|
ipmi|ipmicon|ipmidump|ipmimcthread|ipmidiscover|ipmimc|ipmisdr|ipmiservi
ce|ipmicooling|ipmipower|ipmiptpekeying|ipmisub|softwareupgrade|redundan
cy|script|shfruvalidation|sdrrepository|rmcp|deassert|solmgmt|marshal|al
l
#------------------------------#
# hpiposttype
#------------------------------#
Usage: hpiposttype [OPTION]...
HPI example application to control POST type.
Options:
  -d DOMAIN_ID     use domain with id DOMAIN_ID
  -D               walk recursively through DRT
  -r RESOURCE_ID   use resource with id RESOURCE_ID
  -t POSTTYPE      set POST Type value
  -c CPU_NUM       set CPU number [default=0]
```

```
  -V              print version information and exit
  -h              display this help and exit
#-----------------------------#
# hpipoweronsequence
#-----------------------------#
Usage: hpipoweronsequence [OPTION]...
HPI example application to manage the power on sequence of FRUs during
initial startup.
Options:
  -d DOMAIN_ID    use domain with id DOMAIN_ID
  -D              walk recursively through DRT
  -r RESOURCE_ID  use resource with id RESOURCE_ID
  -p POSITION     use POSITION as power on position
  -s SLOT_RES_ID  set SLOT_RES_ID for specific position
                  (Requires '-r' and '-p' option)
  -c              commit power on sequence to FRU Info
  -V              print version information and exit
  -h              display this help and exit
#-----------------------------#
# hpireset
#-----------------------------#
Usage: hpireset [OPTION]...
HPI example application to reset a FRU resource.
Options:
  -d DOMAIN_ID    use domain with id DOMAIN_ID
  -r RESOURCE_ID  use resource with id RESOURCE_ID
  -c              cold-reset resource (Requires '-r' option)
  -w              warm-reset resource (Requires '-r' option)
  -V              print version information and exit
```

```
  -h              display this help and exit
#-----------------------------#
# hpiresourceself
#-----------------------------#
Usage: hpiresourceself [OPTION]...
HPI example application to print the resource ID  and entity path where
this program is running on.
Options:
  -d DOMAIN_ID  use domain with id DOMAIN_ID
  -V            print version information and exit
  -h            display this help and exit
#-----------------------------#
# hpishaddr
#-----------------------------#
Usage: hpishaddr [OPTION]...
HPI example application to display and set the shelf address.
Options:
  -d DOMAIN_ID    use domain with id DOMAIN_ID
  -b HEX_STRING   set shelf address using binary hex string HEX_STRING
  -V              print version information and exit
  -h              display this help and exit
#-----------------------------#
# hpishelf
#-----------------------------#
Usage: hpishelf [OPTION]...
HPI example application to manage connections to domains.
Note: Requires multishelf library.
Options:
  -c DOMAIN_NAME  create domain with name DOMAIN_NAME
```

```
  -l DOMAIN_NAME  delete domain with name DOMAIN_NAME
  -i IP_ADDR      use IP address IP_ADDR to connect to domain
  -p PORT         use port PORT to connect to domain
  -V              print version information and exit
  -h              display this help and exit
Examples:
Add domain "Gandalf" with IP address 192.168.111.86:
  hpishelf -c Gandalf -i 192.168.111.86
Delete domain "Gandalf":
  hpishelf -l Gandalf
#------------------------------#
# hpiship
#------------------------------#
Usage: hpiship [OPTION]...
HPI example application to display and set the Shelf Manager IP address.
Options:
  -d DOMAIN_ID  use domain with id DOMAIN_ID
  -n CTRL_NUM   use control with number CTRL_NUM
  -i IP_ADDR    set IP address IP_ADDR
  -m NETMASK    set netmask NETMASK
  -g GW_ADDR    set default gateway address GW_ADDR
  -V            print version information and exit
  -h            display this help and exit
#------------------------------#
# hpislotrestore
#------------------------------#
Usage: hpislotrestore [OPTION]...
HPI example application to initiate slot re-discovery
```

```
Options:
  -d DOMAIN_ID     use domain with id DOMAIN_ID
  -D               walk recursively through DRT
  -r RESOURCE_ID   use resource with id RESOURCE_ID
  -R               restore failed slot
  -V               print version information and exit
  -h               display this help and exit
#------------------------------#
# hpisol
#------------------------------#
Usage: hpisol [OPTION]... [PARAM# PARAMS]
HPI example application to manage IPMI v2.0 SOL (Serial over LAN)
settings.
Options:
  -d DOMAIN_ID    use domain with id DOMAIN_ID
  -r RESOURCE_ID  use resource with id RESOURCE_ID
  -V              print version information and exit
  -h              display this help and exit
Examples:
Show SOL configuration for domain 1:
  hpisol -d 1
Set SOL param 1 (SOL Enable) to value 1 for resource 4 on domain 1:
  hpisol -d 1 -r 4 1 1
#------------------------------#
# hpitelcoalarm
#------------------------------#
Usage: hpitelcoalarm [OPTION]...
HPI example application to control telco alarms.
Options:
```

```
  -d DOMAIN_ID    use domain with id DOMAIN_ID
  -r RESOURCE_ID  use resource with id RESOURCE_ID
  -i VALUE        set minor alarm value
  -a VALUE        set major alarm value
  -c VALUE        set critical alarm value
  -V              print version information and exit
  -h              display this help and exit
#------------------------------#
# hpitop
#------------------------------#
Usage: hpitop [OPTION]...
HPI example application to display system topology.
Options:
  -d DOMAIN_ID    use domain with id DOMAIN_ID
  -r RESOURCE_ID  use resource with id RESOURCE_ID
  -A              display everything
  -p              display RPTs
  -s              display sensors
  -c              display controls
  -w              display watchdogs
  -i              display inventories
  -a              display annunciators
  -x              display debug messages
  -V              print version information and exit
  -h              display this help and exit
#------------------------------#
# hpiversion
#------------------------------#
```

```
Usage: hpiversion [OPTION]...

HPI example application to display the version of the different HPI
components.

Options:
  -d DOMAIN_ID  use domain with id DOMAIN_ID
  -i ITEM       print one of the following version item:
  -V            print version information and exit
  -h            display this help and exit
                HPI
                HPI-ATCA-MAPPING
                CLIENT
                CLIENT_PROTOCOL
                MULTISHELF
                MULTISHELF-PROTOCOL
                DAEMON
                DAEMON-PROTOCOL
                DAEMON-HA-PROTOCOL
```

OnATCA-MF106 and SAM1411 shelf managers, do not use the `hpirestartdaemon` sample application to restart the HPI-B daemon. Start/Stop the daemon as mentioned in *Section 2.2.1, Installing the HPI-B Daemon on page 19*. The supervisor does not allow to restart the HPI-B daemon fourth time if it happened to be stopped with `hpirestartdaemon` sample application. After three restarts of the HPI-B daemon, supervisor thinks that there is a problem with the HPI-B daemon, and stops spawning it.

A.3 hpi_shell Utility

The `hpi_shell` utility is contained in `/opt/bladeservices/bin` directory. The usage of `hpi_shell` with sample output is shown below. At any stage, type **`help`** for getting information about the `hpi_shell` usage.

```
-bash-3.00$ hpi_shell

Discovery done

Enter a command or "help" for list of commands
```

```
Available commands are:
addcfg       ann          clearevtlog     ctrl
dat          debug        dimi            domain
domaininfo   dscv         echo            event
evtlogtime   evtlogreset  evtlogstate     exec
fumi         help         history         hs
inv          lsres        lsensor         more
parmctrl     power        quit            rdr
reopen       reset        rpt             run
sen          settag       setsever        settimeevtlog
showevtlog   showinv      showrdr         showrpt
ver          wtdget       wtdreset        wtdset
?            failedrm
Domain list:
    ID: 0   SessionId: 1    Tag: Management
OpenHPI>
```

The following table lists the `hpi_shell` commands and its syntax.

Table A-1 hpi_shell Commands

Command	Description	Syntax
`addcfg`	Add plugins, domains, and handlers from config file.	`addcfg <config file>`
`dat`	Domain Alarm Table list	`dat`
`ann*`	Annunciator command block	`ann <resourceId> <num>`
`clearevtlog`	Clear system event logs	`clearevtlog [<resource id>]`
`ctrl`	Control command block	`ctrl [<ctrlId>]` `ctrlId:: <resourceId> <num>`
`debug`	Set or unset `OPENHPI_ERROR` environment	`debug [ on | off ]`

Table A-1 hpi_shell Commands (continued)

Command	Description	Syntax
`dimi *`	DIMI command block	`dimi [<DimiId>]` `DimiId:: <resourceId> <DimiNum>`
`domain`	Show domain list and set current domain	`domain [<domain id>]`
`domaininfo`	Show current domain info	`domaininfo`
`dscv`	Discover resources	`dscv`
`echo`	Pass string to the stdout	`echo <string>`
`event`	Enable or disable event display on screen	`event [enable|disable|short|full]`
`evtlogtime`	Show the event log's clock	`evtlogtime [<resource id>]`
`evtlogstate`	Show and set the event log state	`evtlogstate [<resource id>] [enable|disable]`
`exec`	Execute external program	`exec <filename> [parameters]`
`fumi`	FUMI command block	`fumi [<FumiId>]` `FumiId:: <resourceId> <FumiNum>`
`help`	Help information for OpenHPI commands	`help [optional commands]`
`history`	Show input commands history	`history`
`hs`	Hot swap command block	`hs <resourceId>`
`inv`	Inventory command block	`inv [<InvId>]` `InvId:: <resourceId> <IdrId>`
`lsres`	List resources	`lsres [stat] [path]`
`lsensor`	List sensors	`lsensor`
`more`	Set or unset more enable	`more [ on | off ]`
`parmctrl`	Save and restore parameters for a resource	`parmctrl <resource id> <action>` `action - default | save | restore`

Table A-1 hpi_shell Commands (continued)

Command	Description	Syntax
power	Power the resource on, or off, or cycle	power <resource id> [on|off|cycle]
quit	Close session and quit console	quit
rdr	Show resource data record	showrdr [<resource id> [type [<rdr num>]]] or rdr [<resource id> [type [<rdr num>]]] type = c - control rdr, s - sensor, i - inventory rdr, w - watchdog, a - annunciator
reopen	Reopen session	reopen [force]
reset	Perform specified reset on the entity	reset <resource id> [cold|warm|assert|deassert]
rpt	Show resource information	showrpt [<resource id>] or rpt [<resource id>]
run	Execute command file	run <file name>
sen	Sensor command block	sen [<sensorId>] sensorId:: <resourceId> <num>
settag	Set tag for a particular resource	settag [<resource id>]
setsever	Set severity for a resource	setsever [<resource id>]
settimeevtlog	Sets the event log's clock	settimeevtlog
showevtlog	Show system event logs	showevtlog [<resource id>]
showinv	Show inventory data of a resource	showinv [<resource id>]

Table A-1 hpi_shell Commands (continued)

Command	Description	Syntax
showrdr	Show resource data record	showrdr [<resource id> [type [<rdr num>]]] or rdr [<resource id> [type [<rdr num>]]] type = c - control rdr, s - sensor, i - inventory rdr w - watchdog, a - annunciator
showrpt	Show resource information	showrpt [<resource id>] or rpt [<resource id>]
ver	Show HPI specification and package version	ver
wtdget	Show watchdog timer	wtdget <resource id> <watchdogNum>
wtdreset	Reset watchdog timer	wtdreset <resource id>
wtdset	Set watchdog timer	wtdset <resource id> <watchdogNum> <values>
?	Help information for OpenHPI commands	?
failedrm	Remove failed resources from RPT	failedrm <resource Id>

Related Documentation

B.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 B-1 SMART EC Documentation

Document Title	Publication Number
Centellis 2000 Release 3.0 Installation and Use	6806800L99
Centellis 2100 Shelf Release 3.9 Installation and Use	6806800T37
SAM1411 Installation and Use	6806800M91
ATCA-MF106 Installation and Use	6806800M62

B.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 B-2 Related Specifications

Organization	Document Title
Intel developer.intel.com/design/servers/ipmi	Platform Management FRU Information Storage Definition v1.0 IPMI Specification v1.5/2.0
PICMG picmg.org/specifications.stm	PICMG 3.0 Revision 2.0 Advanced TCA Base Specification
Service Availability Forum saforum.org	SAI-HPI-B02.01 Service Availability Forum Hardware Platform Interface specification SAIM-HPI-B-01.01-ATCA Service Availability Forum HPI-to-AdvancedTCA Mapping specification

Related Documentation