Tools for Intel® Server Platform Services Firmware 5.0

User Guide

May 2019

Revision: 1.1

Intel Confidential

Contents

Figures

Tables

Revision History

Revision Number	Description	Revision Date
0.1	Update for sps 5.0	November 2017
0.2	Compliance tests update	January 2018
1.0	Remove MEU tool	July 2018
1.1	Decomposition	May 2019

1 Introduction

The purpose of this document is to describe the tools that are used in the platform design, manufacturing, testing, and validation process.

1.1 Terminology

Acronym/Term	Definition
3PDS	3rd Party Data Storage
AC	Alternating Current
Agent	Software that runs on a client PC with OS running
API	Application Programming Interface
ASCII	American Standard Code for Information Interchange
BBBS	BIOS Boot Block Size
BIN	Binary file
BIOS	Basic Input Output System
BIOS-FW	Basic Input Output System Firmware
CLI	Command Line Interface
CPU	Central Processing Unit
CRB	Customer Reference Board
DHCP	Dynamic Host Configuration Protocol
DID	Device ID
DIMM	Dual In-line Memory Module
DLL	Dynamic Link Library
DNS	Domain Naming System
EC	Embedded Controller
EEPROM	Electrically Erasable Programmable Read Only Memory
EHCI	Enhanced Host Controller Interface
EID	Endpoint ID
End User	The person who uses the computer (either Desktop or Mobile). In corporate, the user usually does not have administrator privileges. The end user may not be aware to the fact that the platform is managed by Intel® AMT.
EOP	End Of Post
FCIM	Full Clock Integrated Mode

Acronym/Term	Definition
FCSS	Flex Clock Source Select
FDI	Flexible Display Interface
FITc	Flash Image Tool
FLOCKDN	Flash Configuration Lock-Down
FMBA	Flash Master Base Address
FPSBA	Flash PCH Strap Base Address
FPT	Flash Programming Tool
FPTW	Flash Programming Tool Window
FQDN	Fully Qualified Domain Name
FRBA	Flash Region Base Address
FW	Firmware
FWUpdate	Firmware Update
G3	A system state of Mechanical Off where all power is disconnected from the system. A G3 power state does not necessarily indicate that RTC power is removed.
GbE	Gigabit Ethernet
GMCH	Graphics and Memory Controller Hub
GPIO	General Purpose Input/Output
GUI	Graphical User Interface
GUID	Globally Unique Identifier
HECI (deprecated)	Host Embedded Controller Interface
Host or Host CPU	The processor running the operating system. This is different than the management processor running the Intel® ME FW.
Host Service/ Application	An application running on the host CPU
HostIF	Host Interface
HTTP	HyperText Transfer Protocol
HW	Hardware
IBEN	Input Buffer Enable
IBV	Independent BIOS Vendor
ICC	Integrated Clock Configuration
ID	Identification
IDER	Integrated Drive Electronics Redirection
INF	An information file (.inf) used by Microsoft operating systems that support the Plug & Play feature. When installing a driver, this file provides the OS with the necessary information about driver filenames, driver components, and supported hardware.
Intel® ME	Intel® Management Engine. The embedded processor residing in the chipset GMCH.

Acronym/Term	Definition
Intel® MEI	Intel® Management Engine Interface (renamed from HECI). The interface between the Intel® Management Engine and the Host system.
Intel® NM	Intel® Node Manager
spsINFO	Intel® ME information tool
spsInfoWin	Windows* version of MEINFO
spsManuf	spsManuf validates Intel® ME functionality on the manufacturing line
spsManufWin	Windows version of spsManuf
ISV	Independent Software Vendor
IT User	Information Technology User. Typically very technical and uses a management console to ensure multiple PCs on a network function.
JEDECID	Joint Electronic Device Engineering Councils ID. Standard Manufacturer's Identification Code that is assigned, maintained and updated by the JEDEC office
JTAG	Joint Test Action Group
KVM	Keyboard, Video, Mouse
LAN	Local Area Network
LED	Light Emitting Diode
LPC	Low Pin Count Bus
M0	Intel® ME power state where all HW power planes are activated. Host power state is S0.
M3	Intel® ME power state where all HW power planes are activated but the host power state is different than S0. (Some host power planes are not activated.) The Host PCIe* interface is unavailable to the host SW. The main memory is not available for Intel® ME use.
M-Off	No power is applied to the management processor subsystem. Intel® ME is shut down.
MAC address	Media Access Control address
NVM	Non-Volatile Memory
NVRAM	Non-Volatile Random Access Memory
OCKEN	Output Clock Enable
ODM	Original Device Manufacturer
OEM	Original Equipment Manufacturer
OEM ID	Original Equipment Manufacturer Identification
OOB	Out Of Band
OOB interface.	Out Of Band interface. An SOAP/XML interface over secure or non-secure TCP protocol.
OS	Operating System
OS Hibernate	OS state where the OS state is saved on the hard drive.

Acronym/Term	Definition
OS not Functional	The Host OS is considered non-functional in Sx power state in any one of the following cases when the system is in S0 power state: OS is hung After PCI reset OS watch dog expires OS is not present
OVR	Override
PC	Personal Computer
PCH	Platform Controller Hub
PCI	Peripheral Component Interconnect
PCIe*	Peripheral Component Interconnect Express
PDR	Platform Descriptor Region
PHY	Physical Layer
PID	Provisioning ID
PKI	Public Key Infrastructure
PM	Power Management
PRTC	Protected Real Time Clock
PSK	Pre-Shared Key
PSL	PCH Strap Length
RNG	Random Number Generator
ROM	Read Only Memory
RSA	A public key encryption method
RTC	Real Time Clock
S0	A system state where power is applied to all HW devices and the system is running normally.
S1, S2, S3	A system state where the host CPU is not running but power is connected to the memory system (memory is in self refresh).
S4	A system state where the host CPU and memory are not active.
S5	A system state where all power to the host system is off but the power cord is still connected.
SDK	Software Development Kit
SHA	Secure Hash Algorithm
SMBus	System Management Bus
SOL	Serial over LAN
SPI	Serial Peripheral Interface
SPI Flash	Serial Peripheral Interface Flash

Acronym/Term	Definition
Standby	OS state where the OS state is saved in memory and resumed from the memory when the mouse/keyboard is clicked.
Sx	All S states which are different than S0
SW	Software
System States	Operating System power states such as S0, S1, S2, S3, S4, and S5.
TCP/IP	Transmission Control Protocol/Internet Protocol
TLS	Transport Layer Security
UI	User Interface
UMA	Unified Memory Access
Un-configured state	The state of the Intel® ME FW when it leaves the OEM factory. At this stage the Intel® ME FW is not functional and must be configured.
UNS	User Notification Services
USB	Universal Serial Bus
USBr	Universal Serial Bus Redirection
VLAN	Virtual Local Area Network
VSCC	Vendor Specific Component Capabilities
Windows* PE	Windows* Preinstallation Environment
WIP	Work in Progress
XML	Extensible Markup Language. Intel® AMT's XML-based protocol has three parts: An envelope that defines a framework for describing what is in a message and how to process it A set of encoding rules for expressing instances of application-defined data types A convention for representing remote procedure calls and responses

1.2 Reference Documents

Document	Document No./Location
Intel® Server Platform Services 4.0 Firmware Integration Guide	550581/CDI
SPI Programming Guide	552296/CDI

2 Preface

2.1 Overview

This document covers the system tools used for creating, modifying, and writing binary image files, manufacturing testing, Intel® Management Engine (Intel® ME) setting information gathering, and debugging. The tools are located in **Kit directory\Tools**.

2.2 Operating System Support

Table 2-1. OS Support for Tools

Intel® ME and Manufacturing Tools	UEFI Shell 2.0	Windows 7 x64	Windows 8.1 x86/x64	Windows 10 x64	Windows Server 2012 R2 SP1 x64	Windows Server 2016 x64	Windows PE x64 based on Windows Server 2012	Linux RHEL 7.2 x64
spsFITc		X	X	X	X	X		
MESDC		X	X	X	X	X		
spsMANUF	X				X	X	X	X
spsINFO	X				X	X	X	X
spsFPT	X				X	X		
IBST		X	X	X	X	X		X

2.3 Error Return

Tools always return 0/1 for the error level (0 = success, 1= error). A detail error code is displayed on the screen and stored on an error.log file in the same directory as the tools. (See Appendix A for a list of these error codes.)

2.4 Running application under EFI

There is limitation on EFI application with assertion linked with empty current working directory. To avoid this kind of EFI assertion, user should either make sure already mounted device is used as current working directory (usually it's fs0, fs1,…), i.e.:

```
> fsx:
fsx:\> cd directory_with_sps_tools_package
…
```

or mount a block device first like below:

```
> mount blkx drive_name
> drive_name:
drive_name:\> cd dir_with_sps_tools_package
…
```

also without naming the mounted file system:

```
> mount blkx
> blkx:
blkx:\> cd dir_with_sps_tools_package
...
```

where:

x – storage device id number as mapped by EFI

to see all mapped devices *map* command should be used.

2.5 Usage of the Double-Quote Character (")

The command shell used to invoke tools in both DOS and Windows* has a built-in CLI.

The command shell is intended to be used for invoking applications as well as running in batch mode and performing basic system and file operations. For this reason, the CLI has special characters that perform additional processing upon command.

The double-quote is the only character which needs special consideration as input. The various quoting mechanisms are the backslash escape character (/), single-quotes ('), and double-quotes ("). A common issue encountered with this is the need to have a double-quote as part of the input string rather than using a double-quote to define the beginning and end of a string with spaces.

For example, you may want these words – one two – to be entered as a single string for a vector instead of dividing it into two strings ("one", "two"). In that case, the entry – including the space between the words – must begin and end with double-quotes ("one two") in order to define this as a single string.

When double-quotes are used in this way in the CLI, they define the string to be passed to a vector, but are NOT included as part of the vector. The issue encountered with this is how to have the double-quote character included as part of the vector as well as bypassed during the initial processing of the string by the CLI. This can be resolved by preceding the double-quote character with a backslash (\").

For example, if you want these words to be input – input"string – the command line is: input\"string.

2.6 PMX Driver Limitation

Several tools (spsINFO, spsMANUF, and spsFPT) use the PMX library to get access to the PCI device. Only one tool can get access to the PMX library at a time because of library limitation. Therefore, running multiple tools to get access to PMX library will result in an error (failure to load driver).

The PMX driver is not designed to work with the latest Windows driver model (it does not conform to the new driver's API architecture).

In Windows* 7, the verifier sits in kernel mode, performing continual checks or making calls to selected driver APIs with simulations of well-known driver related issues.

Running the PMX driver with the Windows 7 driver verifier turned on causes the OS to crash. Do not include PMX as part of the verifier driver list if you are running Windows 7 with the driver verifier turned on.

2.7 HECI Limitation

Tools spsINFO, spsMANUF, spsFPT nad MESDC over HECI(including MESDC Agent) will need to use HECI-1 interface to communicate with ME. It is required to have HECI-1 enabled in BIOS. If BIOS disable HECI-1 interface, the tools mentioned before will not work.

3 Flash Image Tool

The Flash Image Tool (**spsFITc.exe**) creates and configures a complete SPI image file for platforms in the following way:

spsFITc creates and allows configuration of the Flash Descriptor Region, which contains configuration information for platform hardware and FW.

spsFITc assembles the binary files into a single SPI flash image. Following regions can be assembled:

- BIOS
- Intel integrated LAN (GbE)
- Intel ME
- Platform Descriptor Region
- Device Extension Region
- The Flash Descriptor Region created by spsFITc

You can manipulate the completed SPI image via a GUI and change the various chipset parameters to match the target hardware. Various configurations can be saved to independent files, so you don't have to recreate a new image each time. Use of GUI is strongly advised while changing the settings as it uses internal logic to validate and cross-reference the new settings.

spsFITc supports a set of command line parameters that can be used to build an image from the CLI or from a batch file. When a previously stored configuration is used to define the image layout, you don't have to interact with the GUI.

spsFITc only generates a complete SPI image file; it does not program the flash device. This complete SPI image must be programmed into the flash with spsFPT, any third-party flash burning tool, or some other flash programmer device.

3.1 System Requirements

spsFITc runs on Windows 10x64, Windows 7 x86/x64, Windows 8 x64, and Windows 2008 R2 SP1 x64. The tool does not have to run on an Intel ME-enabled system.

3.2 Flash Image Details

A flash image can be composed of six regions. The locations of these regions are referred to in terms of where they can be found within the total memory of the flash.

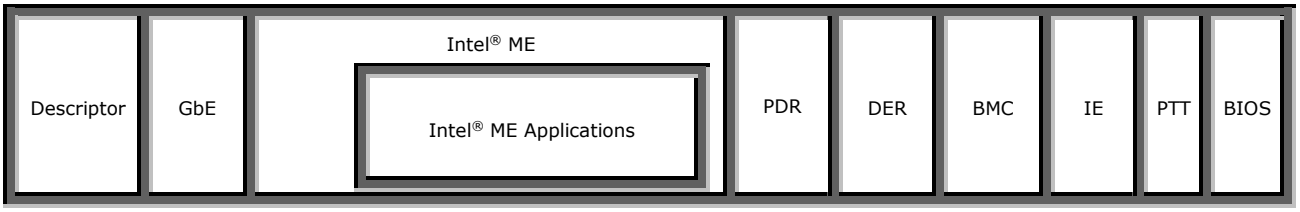


Figure 3-1. SPI Flash Image Regions

Table 3-1. Flash Image Regions – Description

Region	Description
Descriptor	This region contains information such as the space allocated for each region of the flash image, read-write permissions for each region, and a space which can be used for vendor-specific data. It takes up a fixed amount of space at the beginning of the flash memory. **Note:** This region MUST be locked before the serial flash device is shipped to end users. Please see section 0.0.0 for more information. Failure to lock the Descriptor Region leaves the Intel® ME device vulnerable to security attacks. This region is mandatory and enabled by default.
GbE	This region contains code and configuration data for an Intel Integrated LAN (Gigabit Ethernet). This region is not mandatory and disabled by default.
Intel® ME	This region contains code and configuration data for Intel® ME applications. It takes up a variable amount of space up to the BIOS region. This region is mandatory and enabled by default.
PDR	This region lets system manufacturers describe custom features for the platform. This region is not mandatory and disabled by default.
DER	Device Extension Region used by Intel Node Manager-PTU This region is not mandatory and disabled by default.
BMC	Embedded Controller/Baseboard Management Controller This region is not mandatory and disabled by default.
IE	Innovation Engine This region is not mandatory and disabled by default.
PTT	Platform Trusted Technology This region is mandatory and enabled by default.
BIOS	This region contains code and configuration data for the entire computer. This region is not mandatory and disabled by default.

3.2.1 Flash Space Allocation

Space allocation for each region is determined as follows:

Each region can be assigned a fixed amount of space. If a region is not assigned a fixed amount of space, it occupies only as much space as it requires.

If there is still space left in the flash after allocating space to all of the regions, the Intel ME region expands to fill the remaining space.

3.3 Required Files

The spsFITc main executable is **spsFITc.exe**. All files from the **FlashImageTool** directory should be copied along with the executable.

Additional files that can be passed as an input to spsFITc:

configuration XMLs

region specific binaries

3.3.1 Configuration Files

The flash image can be configured in many different ways, depending on the target hardware and the required FW options. spsFITc lets you change this configuration in a graphical manner (via the GUI). Each configuration can be saved to an XML file. These XML files can be loaded at a later time and used to build subsequent flash images.

Creating a New Configuration

spsFITc provides default configuration files that you can use to build a new image. You should open one of the xml configurations files from the package to create a new configuration.

Opening an Existing Configuration

To open an existing configuration file:

Choose **File > Open** or click on **Open** icon on toolbar; the Open File dialog appears

Select the XML file you want to load

Click **Open**.

You can also open a file by dragging and dropping a configuration file into the main window of the application.

SpsFITc starts with default settings loaded which provides configuration for valid image build. This default setting can be overridden by providing configuration xml (default configuration xml can be found in the package), this can be done by:

Loading configuration xml through spsFITc GUI (described at the beginning of this section)

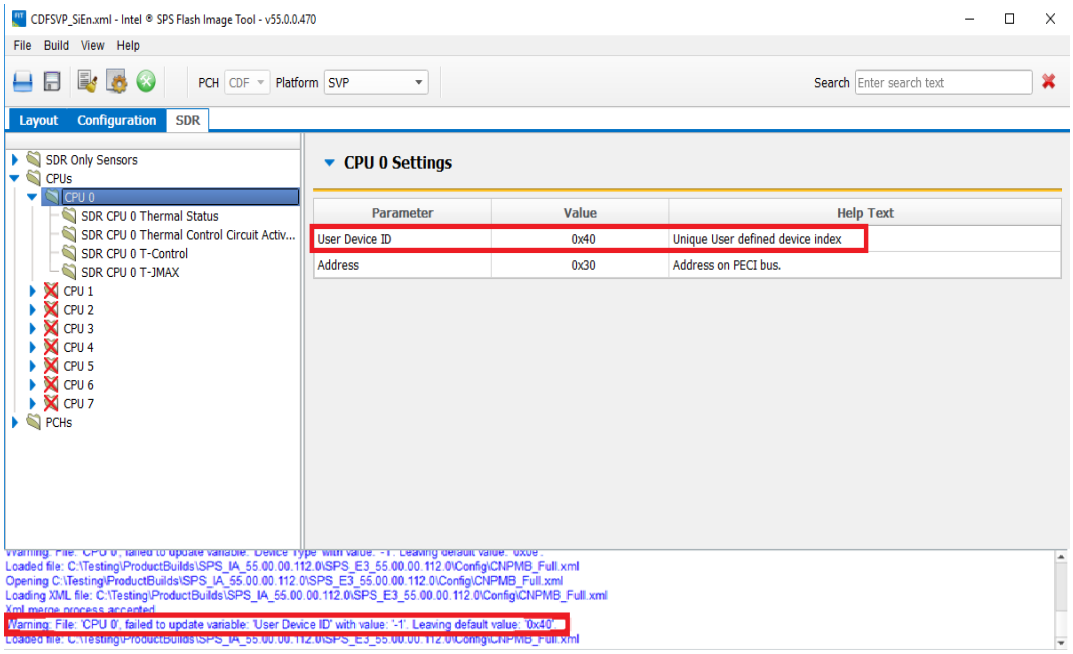
Providing parameter "`-f <filename>`" in the spsFITc CLI (see section Command Line Interface)

Configuration xml is loaded in delta mode what means that:

Settings not provided in the xml will keep their default values.

Settings that could not be recognized will be skipped and proper information will be displayed in CLI or GUI log window.

While loading particular setting basic validation (not dependent from other settings) will be performed. If failed, default value will not be overwritten and proper information will be displayed in CLI or GUI log window (see picture bellow).



After loading all settings additional validation is performed. In this step dependencies between settings are validated. If failed, appropriate settings default values will be restored and proper information will be displayed in CLI or GUI log window.

Delta mode supports loading configuration xmls from previous spsFITc versions.

Saving a Configuration

To save the current configuration in an XML file:

Choose **File > Save**, click on **Save** icon on toolbar or **File > Save As**; the Save File dialog appears if the configuration has not been given a name or if **File > Save As** was chosen.

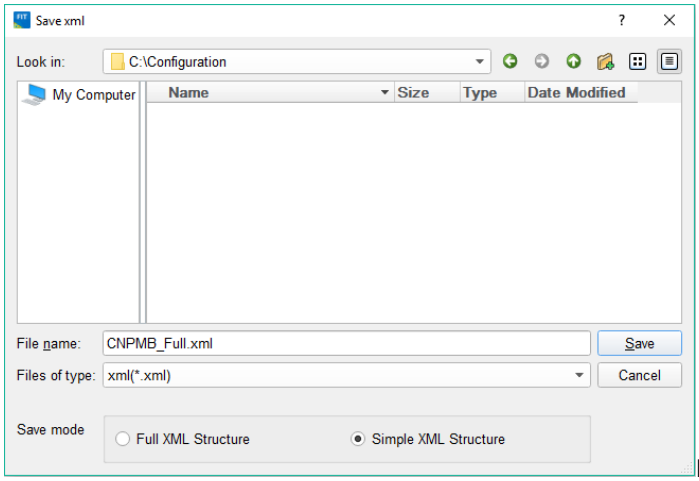
Select the path and enter the file name for the configuration.

Click **Save**.

Configuration file can be saved in two different modes by checking proper radio button:

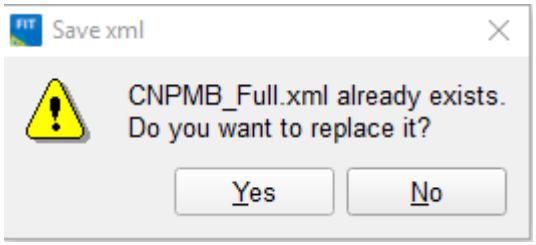
Full XML Structure- save complete configuration with all settings

Simple XML Structure - save only differences between current and default configuration

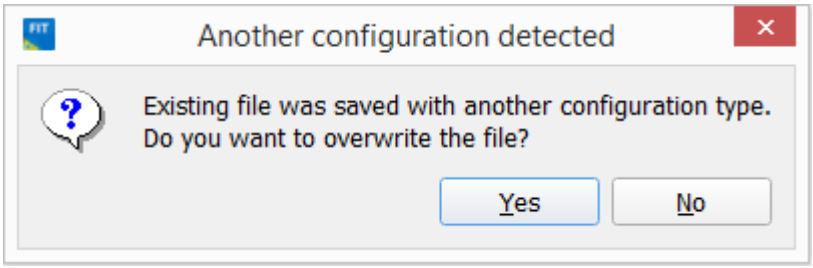


By default full mode is selected.

If you choose existing file when saving, a following question will appear:



If you choose "yes" option, the application will check whether the selected file is saved with the same configuration type (full or simple). When existing file was saved with different configuration, then another question will appear:



You can overwrite file by clicking “yes” or you can click “no” to select different file name.

3.4 Environment Variables

A set of environment variables is provided to make the image configuration files more portable. The configuration is not tied to a particular root directory structure because all of the paths in the configuration are relative to environment variables. You can set the environment variables appropriate for your computer, or override the variables with command line options.

It is recommended that the environment variables be the first thing you set when working with a new configuration. This ensures that spsFITc can properly substitute environment variables into paths to keep them relative. Doing this also speeds up configuration because many of the Open File dialogs default to particular environment variable paths.

To modify the environment variables:

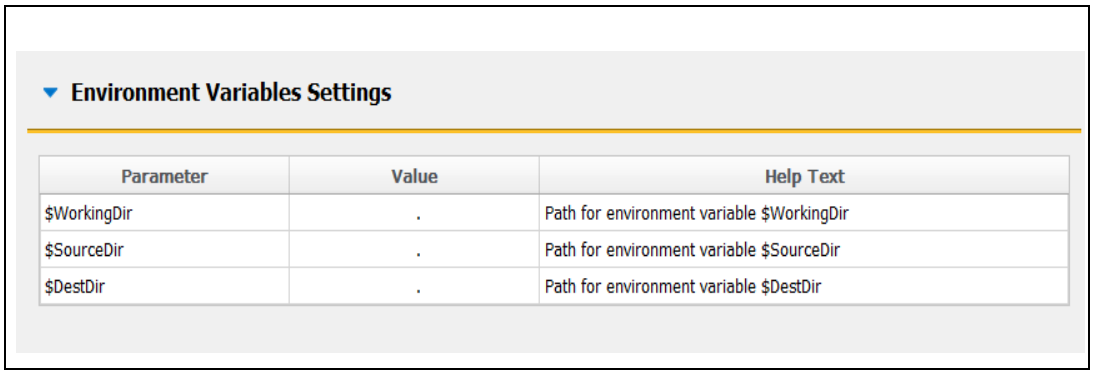
Choose **Build > Build Settings** or click **Build Settings** icon on toolbar, a dialog appears; or go to main window, **Layout** tab, **Build Settings** node in the tree view. Go to the **Environment Variables** section containing current working directory on top, followed by the current values of all the environment variables:

$WorkingDir – the directory where the log file is kept and where the components of an image are stored when an image is decomposed.

$SourceDir – the directory that contains the base image binary files from which a complete flash image is prepared. Usually these base image binary files are obtained from Intel® VIP on the Web, a BIOS programming resource, or another source.

$DestDir – the directory in which the final combined image is saved, as well as all intermediate files generated during the build.

Figure 3-2. Environment Variables Dialog



Edit value field (use double click) and specify the directory where that variable's files will be stored; the name and relative path of that directory appears in the field next to the variable's name.

Repeat Step 2 until the directories of all relevant environment variables have been defined.

Click **OK**.

The environment variables are saved in the application's INI file, not the XML configuration file. This allows the configuration files to be portable across different computers and directory structures.

3.5 Build Settings

spsFITc lets you set several options that control how the image is built. The options that can be modified are described in

Table 3-2.

To modify the build setting:

Choose **Build** > **Build Settings**; a dialog appears showing the current build settings.

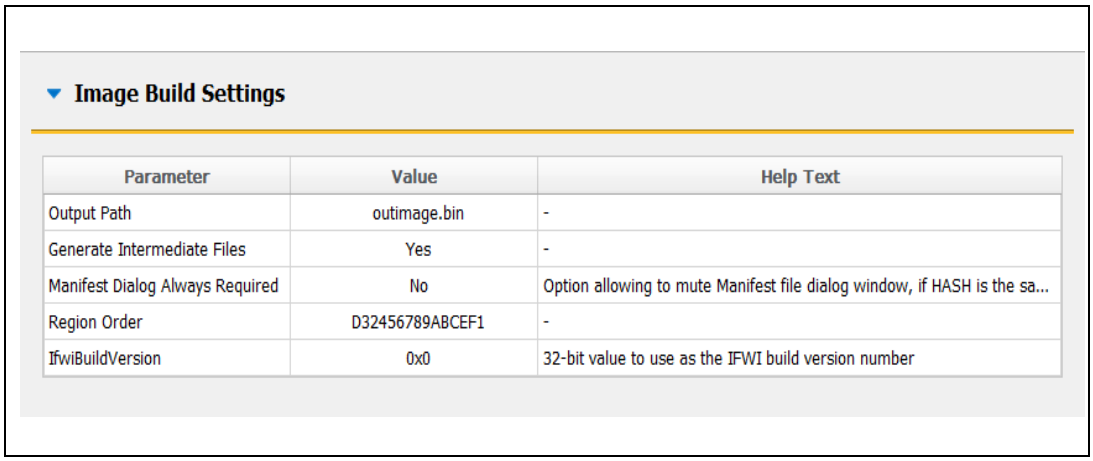
Modify the relevant settings in the **Build Settings** dialog.

Click **OK**; the modified build settings are saved in the XML configuration file.

Table 3-2. Build Settings Dialog Options

Option	Description
Output path	The path and filename where the final image should be saved after it is built. (**Note**: Using the $DestDir environment variable makes the configuration more portable.)
Generate intermediate build files	Causes the application to generate separate (intermediate) binary files for each region, in addition to the final image file (see Figure 3-3). These files are located in the specified output folder's INT subfolder. These image files can be programmed individually with the spsFPT.

Figure 3-3. Build Settings Dialog



3.5.1 Modifying the Flash Settings

The FDR contains information about the flash image and the target hardware. This region contains the read/write values. It is important for this region to be configured correctly or the target platform may not function as expected. This region also needs to be configured correctly in order to ensure that the system is secure.

Descriptor Region Length

The Descriptor Region Length parameter sets the size of the Descriptor region.

To set the value of the Descriptor Region Length parameter:

Select **Flash Layout\Regions\Descriptor Region** in the left panel the **Descriptor Region Length** parameter appears in the right panel.

Enter any non-zero value into the dialog to set the length of the region and click **OK**.

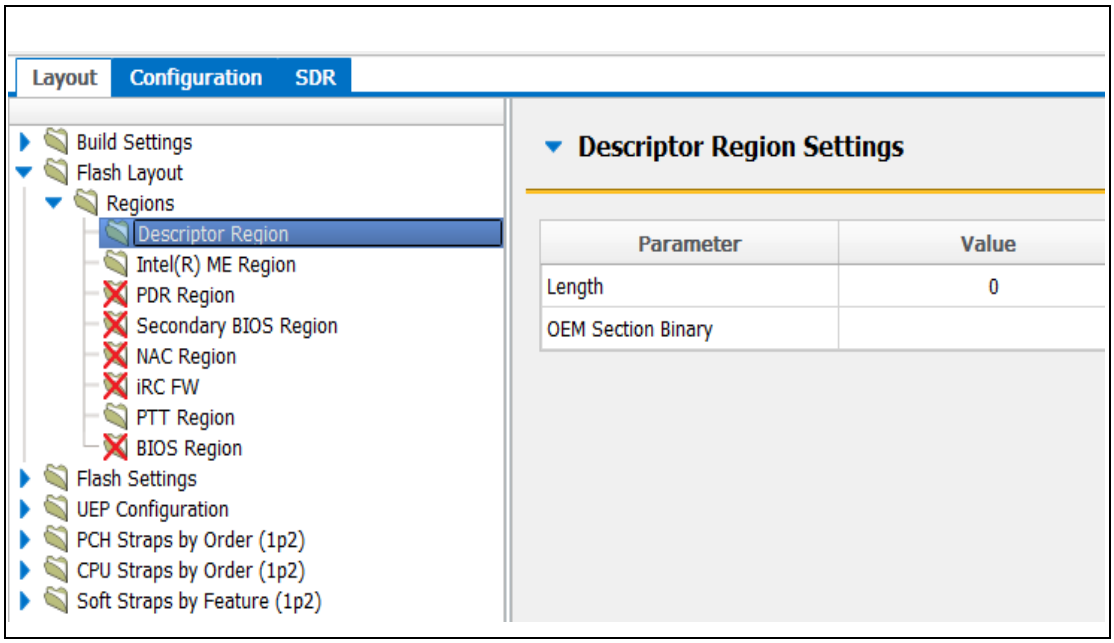


Figure 3-4. Descriptor Region Length Parameter

Setting the Number and Size of the Flash Components

To set the number of flash components:

Expand the **Flash Settings** node of the tree in the left panel.

Select **Flash Components** (see Figure 3-5) all the parameters in the Descriptor Map section are listed in the right panel.

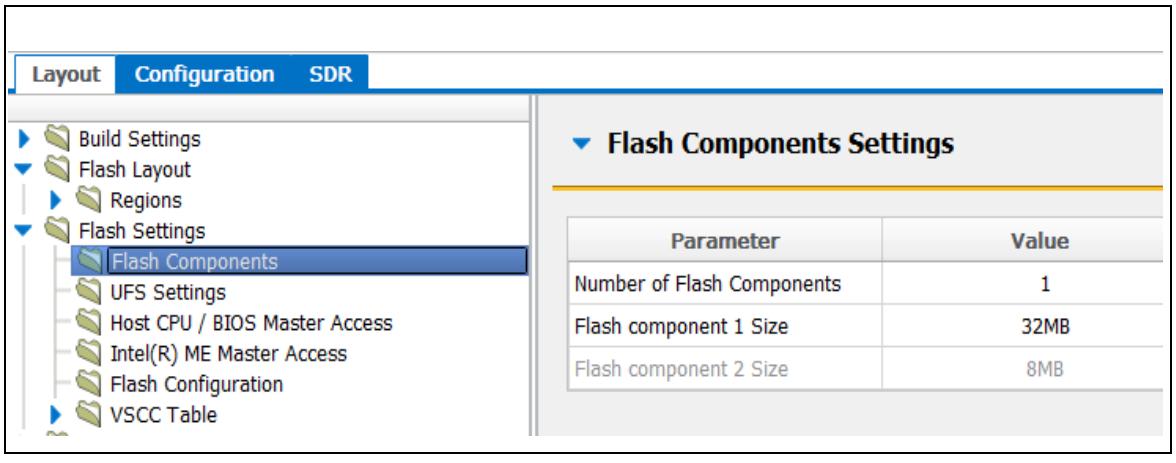


Figure 3-5. Flash Settings > Flash Components

Enter the number of flash components (valid values are 0, 1 or 2).

Click **OK**; the parameter is updated.

To set the size of each flash component:

Expand **Flash Settings** node in the left panel and select **Flash Components**; the Component Section parameters appear in the right panel. The **Flash component 1 Size** and **Flash component 2 Size** parameters specify the size of each flash component.

Select the correct component size from the drop-down list and click **OK**; that parameter is updated.

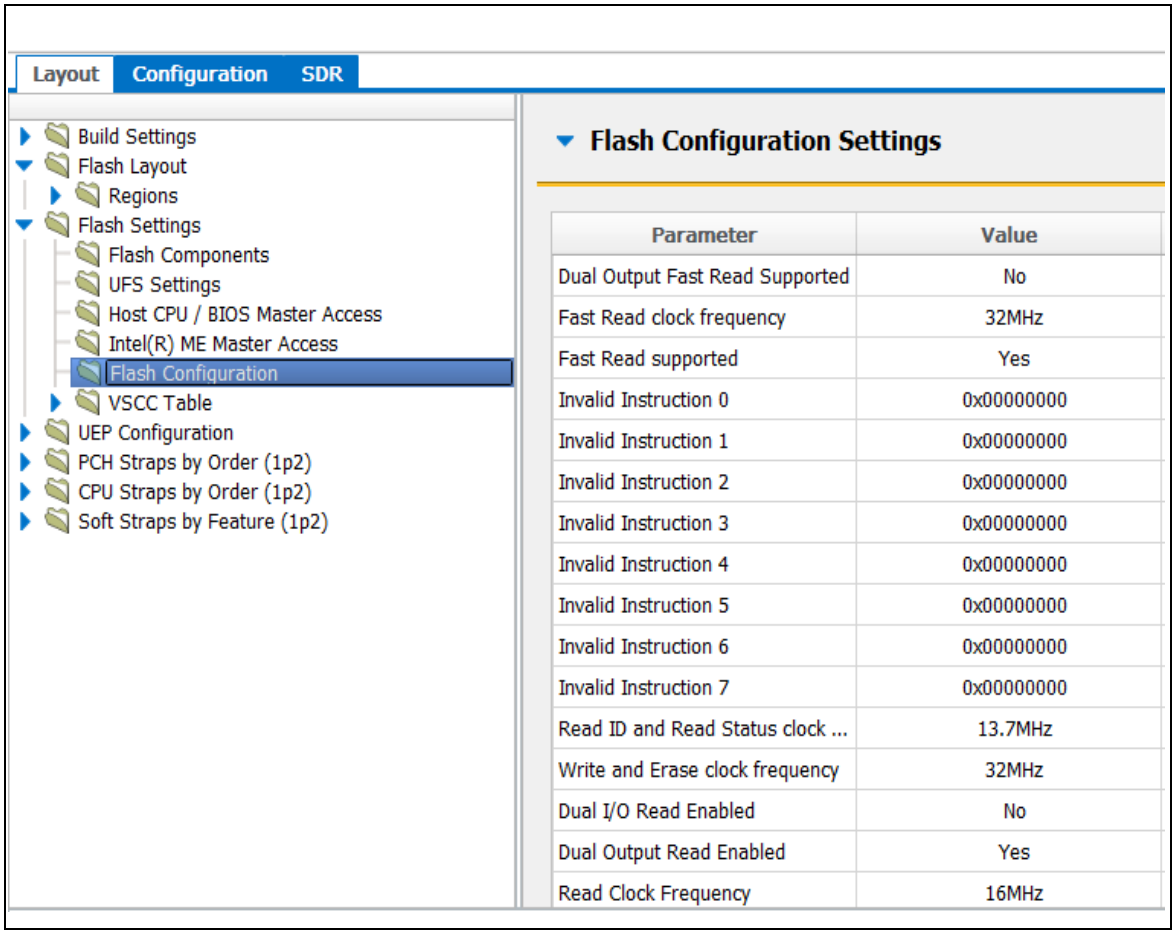
Repeat steps 2-3 for the other parameter.

The size of the second flash component is only editable if the number of flash components is set to 2.

Other Flash settings

All other flash settings are available in **Flash Settings\Flash Configuration.**

Figure 3-6. Flash Settings > Flash Configuration



Region Access Control

Regions of the flash can be protected from read or write access by setting a protection parameter in the Descriptor Region. The Descriptor Region must be locked before Intel ME devices are shipped. If the Descriptor Region is not locked, the Intel ME device is vulnerable to security attacks. The level of read/write access provided is at the discretion of the OEM/ODM. A cross-reference of access settings is shown below.

Table 3-3. Region Access Control Table

Regions that can be accessed					
Region to Grant Access	**PDR**	**Intel® ME**	**GbE**	**BIOS**	**Descriptor**
Intel® ME	None/Read/Write	Intel® ME can always read from and write to Intel® ME Region	None/Read/Write	None/Read/Write	None/Read/Write
GbE	None/Read/Write	None/Read/Write	GbE can always read from and write to GbE Region	None/Read/Write	None/Read/Write
BIOS	None/Read/Write	None/Read/Write	None/Read/Write	Write only. BIOS can always read from and write to BIOS Region	None/Read/Write

There are three parameters in the Descriptor that specify access for each chipset. The bit structure of these parameters is shown below.

Please consult platforms "SPI Programming Guide" for the latest values.

Key:

0 – denied access
1 – allowed access
NC – bit may be either 0 or 1 since it is unused.

Table 3-4. CPU/BIOS Access

Read Access								
	Unused			**PDR**	**GbE**	**Intel ME**	**BIOS**	**Desc**
Bit Number	7	6	5	4	3	2	1	0
Bit Value	X	X	X	0/1	0/1	0/1	NC	0/1

Write Access								
	Unused			PDR	GbE	Intel ME	BIOS	Desc
Bit Number	7	6	5	4	3	2	1	0
Bit Value	X	X	X	0/1	0/1	0/1	NC	0/1

Example:

If the CPU/BIOS needs read access to the GbE and Intel ME and write access to Intel ME, then the bits are set to:

Read Access – 0b 0000 1110 (0x0E in hexadecimal)

Write Access – 0b 0000 0110 (0x06 in hexadecimal)

To set these access values in spsFITc:

Select **Flash Settings\Host CPU / BIOS Master Access** or **GBE Master Access** or **Intel(R) ME Master Access** in the left panel the access parameters are listed in the right panel.

Click on each parameter and set its access value in one of the following ways:

To generate an image for debug purposes or to leave the SPI region open: select 0xFF for both read and write access in all three sections.

To lock the SPI in the image creation phase: select the recommended setting for production (e.g., select 0x0D for Intel ME read access and 0x0C for Intel ME write access).

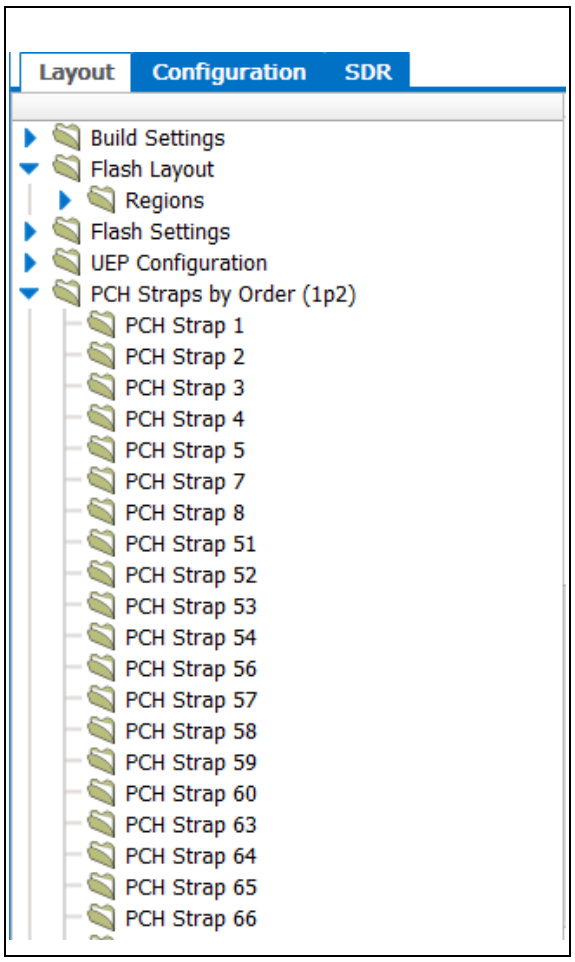
If for the **Host CPU / BIOS Master Access** write access property is set to 0xFFF, user is notified with a proper warning message that the region write protect should be handled by BIOS, otherwise the region is left open. As a precaution, this message is also displayed each time the image is built.

Please consult platforms "SPI Programming Guide" for the latest values.

3.5.2 PCH Soft Straps

These sections contain configuration options for the PCH. The number of Soft Strap sections and their functionality differ based on the target PCH. **Improper settings could lead to undesirable behavior from the target platform.** For more information on how to set them correctly, see the SPI programming guide, Appendix A.

Soft Straps can be loaded from a binary file. This capability is assigned to a node in the left panel (PCH straps by Order / Soft Straps by Feature). To load starps right-click on a node and choose "Update from binary file" option from a context menu.



3.5.3 VSCC Table

This section is used to store information to setup flash access for Intel ME. This does not have any effect on the usage of the spsFPT. **If the information in this section is incorrect, Intel ME FW may not communicate with the flash device.** The information provided is dependent on the flash device used on the system. (For more information, see the PCH SPI programming guide, Section 6.4.)

Adding a New Table Entry

To add a new table entry:

Select **Flash Settings\VSCC table**

Click **Add VSCC Entry** button in the right panel

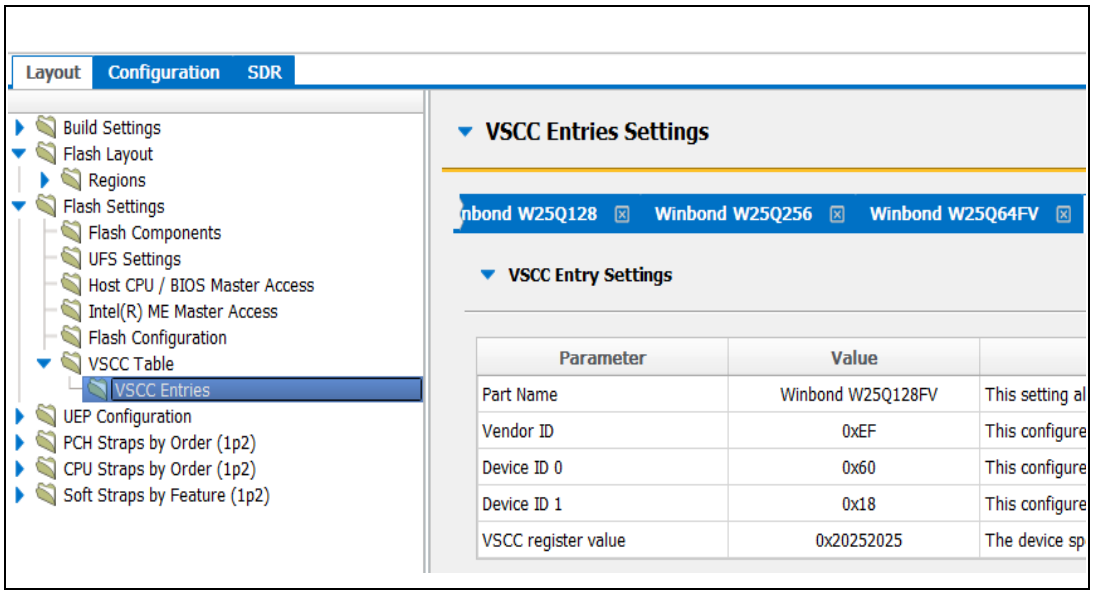


Figure 3-7. VSCC Table

Enter a name into the **Entry Name** field. (**Note**: To avoid confusion it is recommended that each table entry name be unique. There is no checking mechanism in spsFITc to prevent table entries that have the same name and no error message is displayed in such cases.)

The new table is listed in the left panel under **VSCC Table** and you can enter into it the values for the flash device. (Figure 3-8, which shows the parameters of a new VSCC table.)

The values in the VSCC table can be found in the serial flash data sheet. You should use the SPT SPI Programming Guide to calculate the VSSC values.

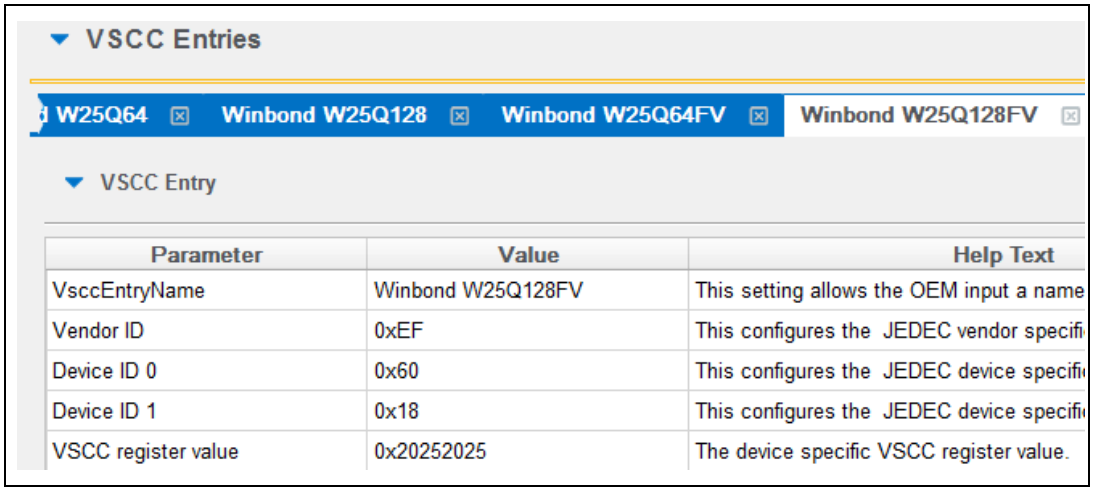


Figure 3-8. Sample VSCC Table Entry

Removing an Existing VSCC Table

To remove an existing table:

Click on the after the name of the VSCC entry you want to remove.

3.5.4 Modifying the Intel® Management Engine (Intel® ME) Region

The Intel ME Region contains all of the FW data for the Intel ME.

Setting the Intel ME Region Binary File.

To select the Intel ME region binary file:

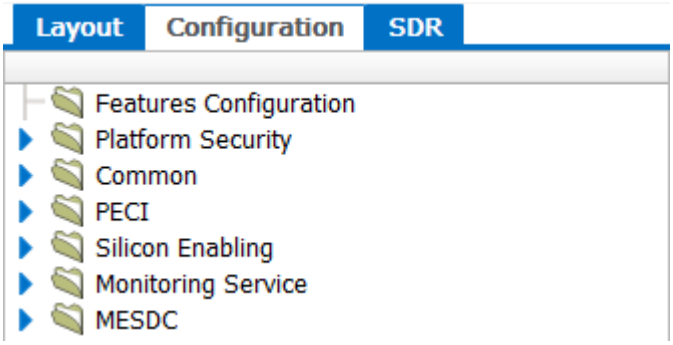
Select the **Flash Layout\Regions\Intel(R) Me Region** tree node.

Update the parameter; when the flash image is built, the contents of this file is copied into the Intel ME Region.

3.5.5 Intel ME FW Configurations

This section contains Intel ME related configurations.

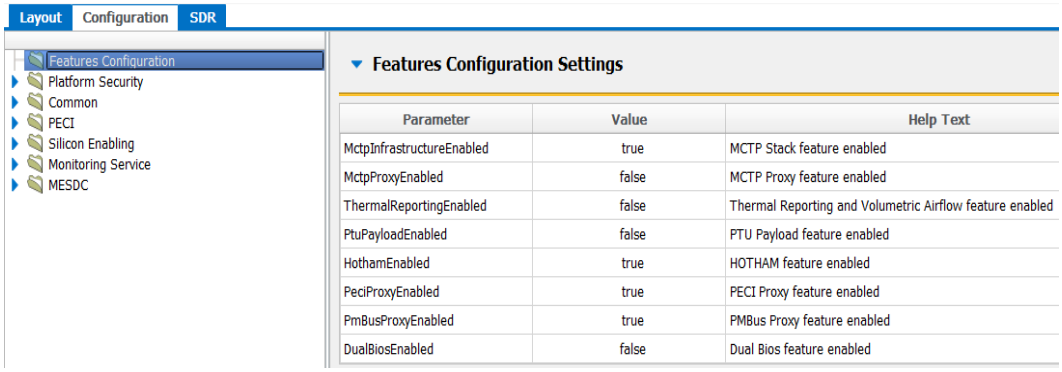
Figure 3-9. Configuration Tab—Intel ME Configuration



This section contains the setting for all Intel ME related feature configurations that apply to all FW configurations including SiEn.

General Intel® Presents

Figure 3-10. Configuration ->Features Configuration



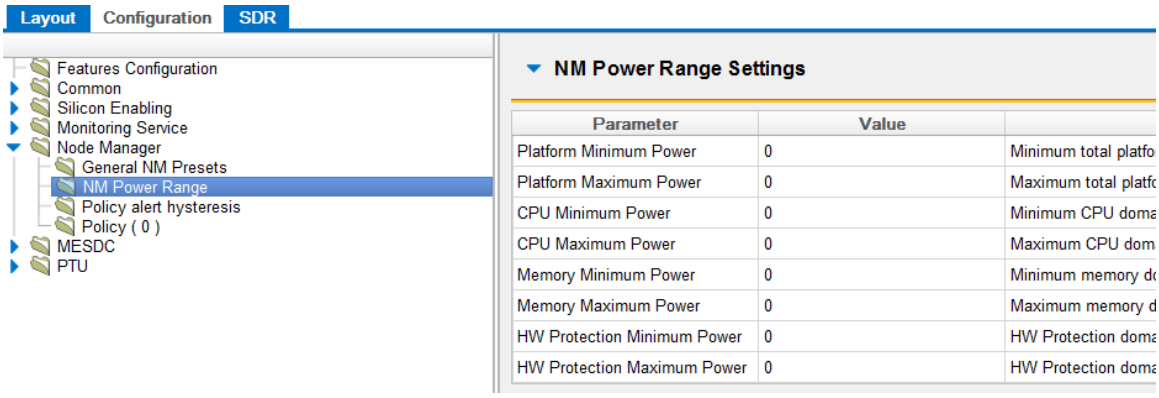
Parameter	Value	Help Text
MctpInfrastructureEnabled	true	MCTP Stack feature enabled
MctpProxyEnabled	false	MCTP Proxy feature enabled
ThermalReportingEnabled	false	Thermal Reporting and Volumetric Airflow feature enabled
PtuPayloadEnabled	false	PTU Payload feature enabled
HothamEnabled	true	HOTHAM feature enabled
PeciProxyEnabled	true	PECI Proxy feature enabled
PmBusProxyEnabled	true	PMBus Proxy feature enabled
DualBiosEnabled	false	Dual Bios feature enabled

This section contains the general Intel Node Manager configuration including policy control, domains state, pre-configured policies and SmaRT&CLST configurations. This only applies to Intel Node Manager SKU.

Any settings which are different than default value are highlighted.

Power Range

Figure 3-11. Configurations-> NM Power Range

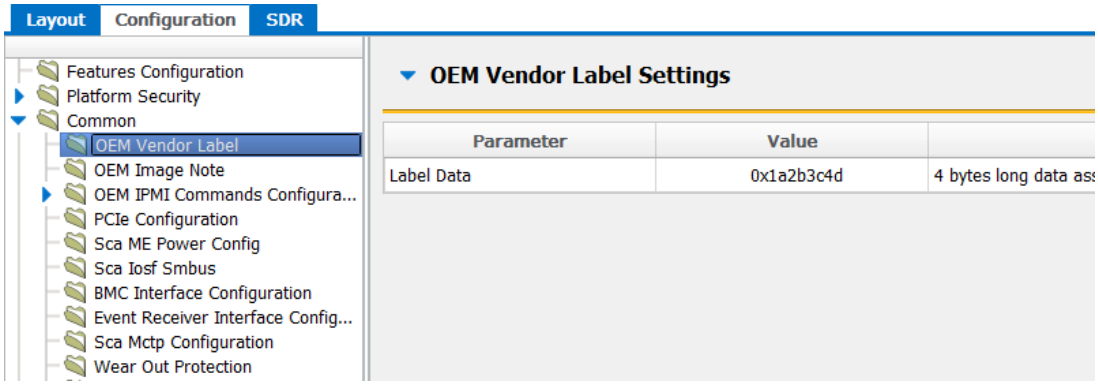


Parameter	Value	
Platform Minimum Power	0	Minimum total platfo
Platform Maximum Power	0	Maximum total platfo
CPU Minimum Power	0	Minimum CPU doma
CPU Maximum Power	0	Maximum CPU dom
Memory Minimum Power	0	Minimum memory do
Memory Maximum Power	0	Maximum memory d
HW Protection Minimum Power	0	HW Protection doma
HW Protection Maximum Power	0	HW Protection doma

This section can set pre-configured power budget applied when policy control is disabled. This is only relevant to Intel Node Manager Configuration.

Vendor Label

Figure 3-12. Configuration -> OEM Vendor Label

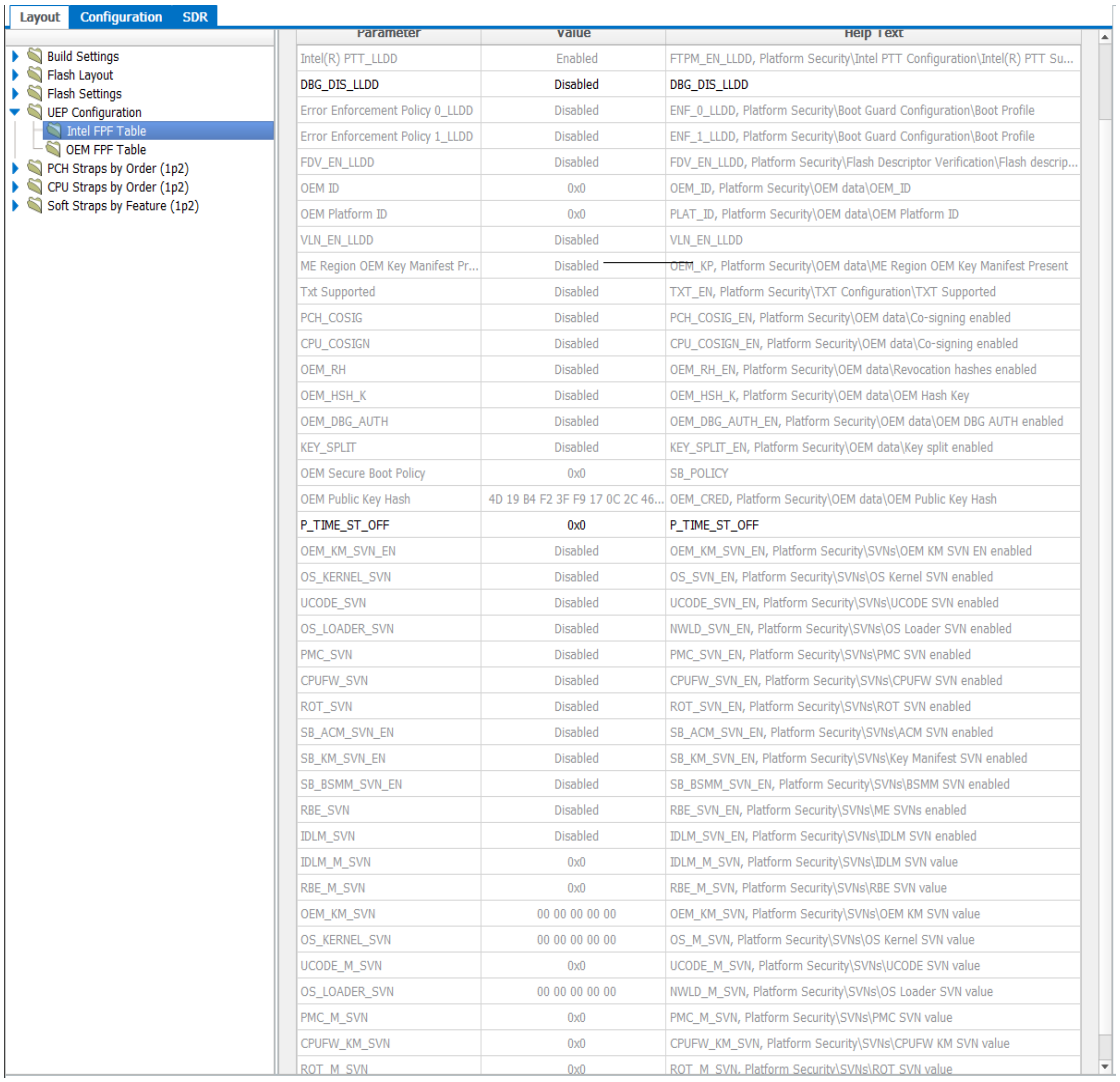


This section shows the users how to configure a 4-byte label that can be used by platform owner.

3.5.6 Fuse Configuration files

This section shows the users how to configure FPF.

Table 3-5. Fuse Configuration files



Parameter	Value	Help Text
Intel(R) PTT_LLDD	Enabled	FTPM_EN_LLDD, Platform Security\Intel PTT Configuration\Intel(R) PTT Su...
DBG_DIS_LLDD	Disabled	DBG_DIS_LLDD
Error Enforcement Policy 0_LLDD	Disabled	ENF_0_LLDD, Platform Security\Boot Guard Configuration\Boot Profile
Error Enforcement Policy 1_LLDD	Disabled	ENF_1_LLDD, Platform Security\Boot Guard Configuration\Boot Profile
FDV_EN_LLDD	Disabled	FDV_EN_LLDD, Platform Security\Flash Descriptor Verification\Flash descrip...
OEM ID	0x0	OEM_ID, Platform Security\OEM data\OEM_ID
OEM Platform ID	0x0	PLAT_ID, Platform Security\OEM data\OEM Platform ID
VLN_EN_LLDD	Disabled	VLN_EN_LLDD
ME Region OEM Key Manifest Pr...	Disabled	OEM_KP, Platform Security\OEM data\ME Region OEM Key Manifest Present
Txt Supported	Disabled	TXT_EN, Platform Security\TXT Configuration\TXT Supported
PCH_COSIG	Disabled	PCH_COSIG_EN, Platform Security\OEM data\Co-signing enabled
CPU_COSIGN	Disabled	CPU_COSIGN_EN, Platform Security\OEM data\Co-signing enabled
OEM_RH	Disabled	OEM_RH_EN, Platform Security\OEM data\Revocation hashes enabled
OEM_HSH_K	Disabled	OEM_HSH_K, Platform Security\OEM data\OEM Hash Key
OEM_DBG_AUTH	Disabled	OEM_DBG_AUTH_EN, Platform Security\OEM data\OEM DBG AUTH enabled
KEY_SPLIT	Disabled	KEY_SPLIT_EN, Platform Security\OEM data\Key split enabled
OEM Secure Boot Policy	0x0	SB_POLICY
OEM Public Key Hash	4D 19 B4 F2 3F F9 17 0C 2C 46...	OEM_CRED, Platform Security\OEM data\OEM Public Key Hash
P_TIME_ST_OFF	0x0	P_TIME_ST_OFF
OEM_KM_SVN_EN	Disabled	OEM_KM_SVN_EN, Platform Security\SVNs\OEM KM SVN EN enabled
OS_KERNEL_SVN	Disabled	OS_SVN_EN, Platform Security\SVNs\OS Kernel SVN enabled
UCODE_SVN	Disabled	UCODE_SVN_EN, Platform Security\SVNs\UCODE SVN enabled
OS_LOADER_SVN	Disabled	NWLD_SVN_EN, Platform Security\SVNs\OS Loader SVN enabled
PMC_SVN	Disabled	PMC_SVN_EN, Platform Security\SVNs\PMC SVN enabled
CPUFW_SVN	Disabled	CPUFW_SVN_EN, Platform Security\SVNs\CPUFW SVN enabled
ROT_SVN	Disabled	ROT_SVN_EN, Platform Security\SVNs\ROT SVN enabled
SB_ACM_SVN_EN	Disabled	SB_ACM_SVN_EN, Platform Security\SVNs\ACM SVN enabled
SB_KM_SVN_EN	Disabled	SB_KM_SVN_EN, Platform Security\SVNs\Key Manifest SVN enabled
SB_BSMM_SVN_EN	Disabled	SB_BSMM_SVN_EN, Platform Security\SVNs\BSMM SVN enabled
RBE_SVN	Disabled	RBE_SVN_EN, Platform Security\SVNs\ME SVNs enabled
IDLM_SVN	Disabled	IDLM_SVN_EN, Platform Security\SVNs\IDLM SVN enabled
IDLM_M_SVN	0x0	IDLM_M_SVN, Platform Security\SVNs\IDLM SVN value
RBE_M_SVN	0x0	RBE_M_SVN, Platform Security\SVNs\RBE SVN value
OEM_KM_SVN	00 00 00 00 00	OEM_KM_SVN, Platform Security\SVNs\OEM KM SVN value
OS_KERNEL_SVN	00 00 00 00 00	OS_M_SVN, Platform Security\SVNs\OS Kernel SVN value
UCODE_M_SVN	0x0	UCODE_M_SVN, Platform Security\SVNs\UCODE SVN value
OS_LOADER_SVN	00 00 00 00 00	NWLD_M_SVN, Platform Security\SVNs\OS Loader SVN value
PMC_M_SVN	0x0	PMC_M_SVN, Platform Security\SVNs\PMC SVN value
CPUFW_KM_SVN	0x0	CPUFW_KM_SVN, Platform Security\SVNs\CPUFW KM SVN value
ROT_M_SVN	0x0	ROT_M_SVN, Platform Security\SVNs\ROT SVN value

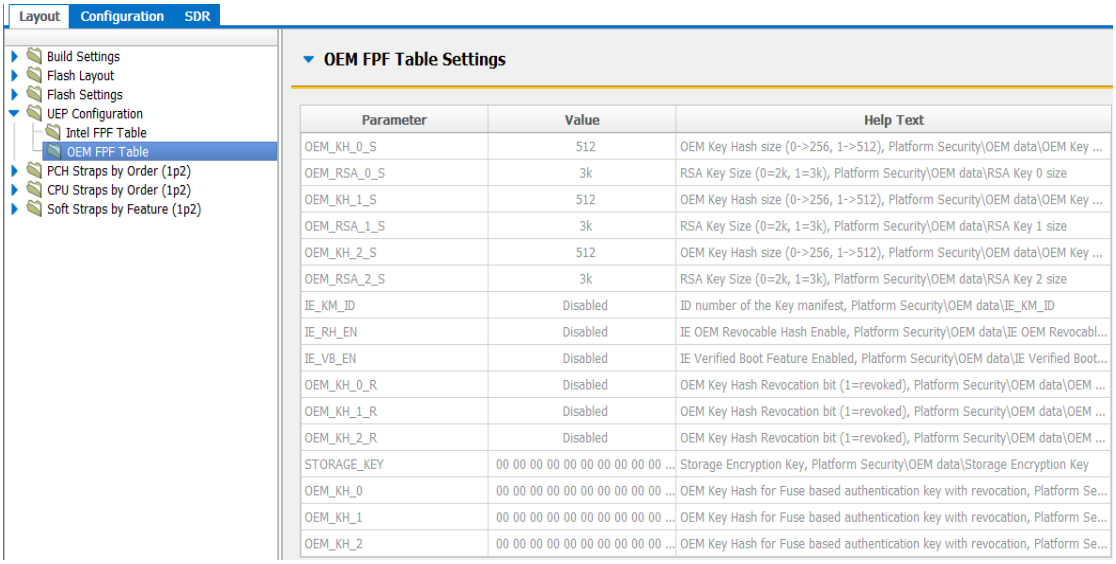
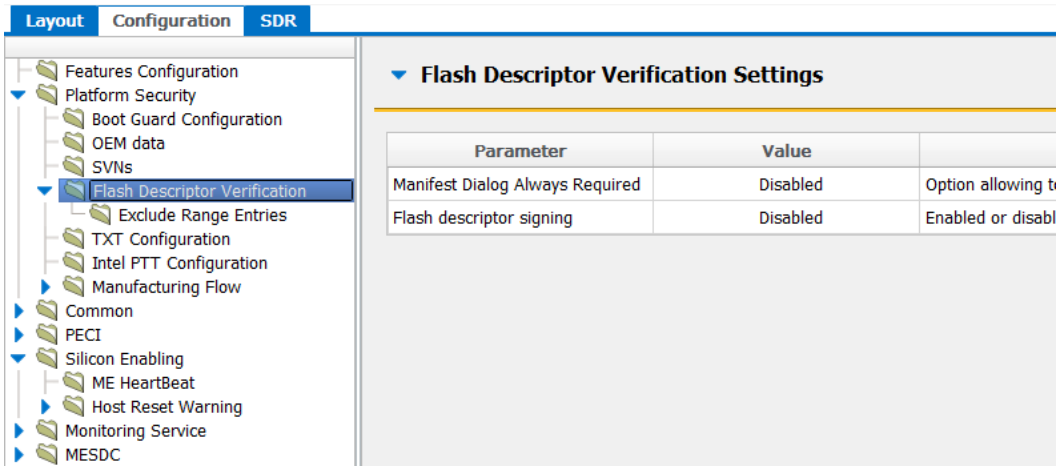


Table 3-6. Boot Guard Profile Configuration

FITc setting	Boot Guard Policy Restricions		Boot Guard Policy Type		Error Enforcement Policy		Notes
Boot Guard Profile Configuration	[0] Force Anchor Cove Boot	[3] Protect BIOS Environ-ment	Verified	Measured	Error Enforcement Policy 0	Error Enforcement Policy 1	
0 (No_FVME)	0	0	0	0	0	0	
3 (VM)	0	1	1	1	0	0	Profile does not support End Of Manufacturing.
4 (FVE)	1	1	1	0	1	1	
5 (FVME)	1	1	1	1	1	1	
6 (FV)	1	1	1	0	0	0	Profile does not support End Of Manufacturing. Debug profile for Intel use only.

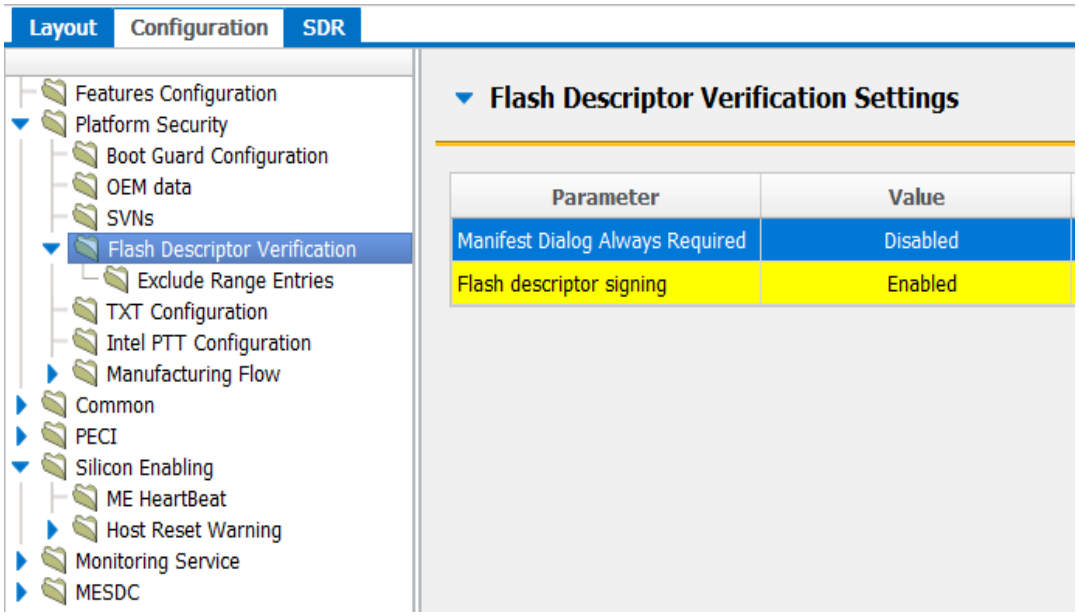
3.5.7 Flash Descriptor Verification

Figure 3-12. Platform Security -> Flash Descriptor Verification



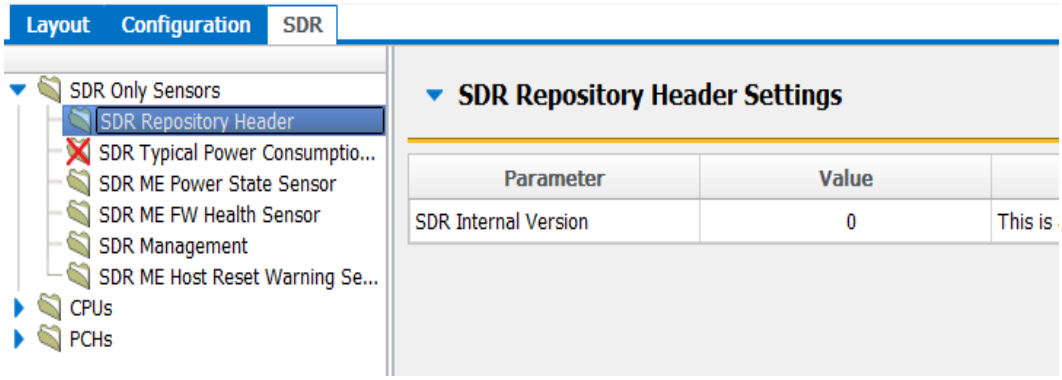
This section allows user to configure Flash Decriptor Verification settings. User can enable it by changing value of Flash descriptor signing from “disabled” to “enabled”. (Figure 3-15).

Figure 3-13. Flash Descriptor Verification



3.5.8 SDR Configuration

Figure 3-146. SDR Configuration



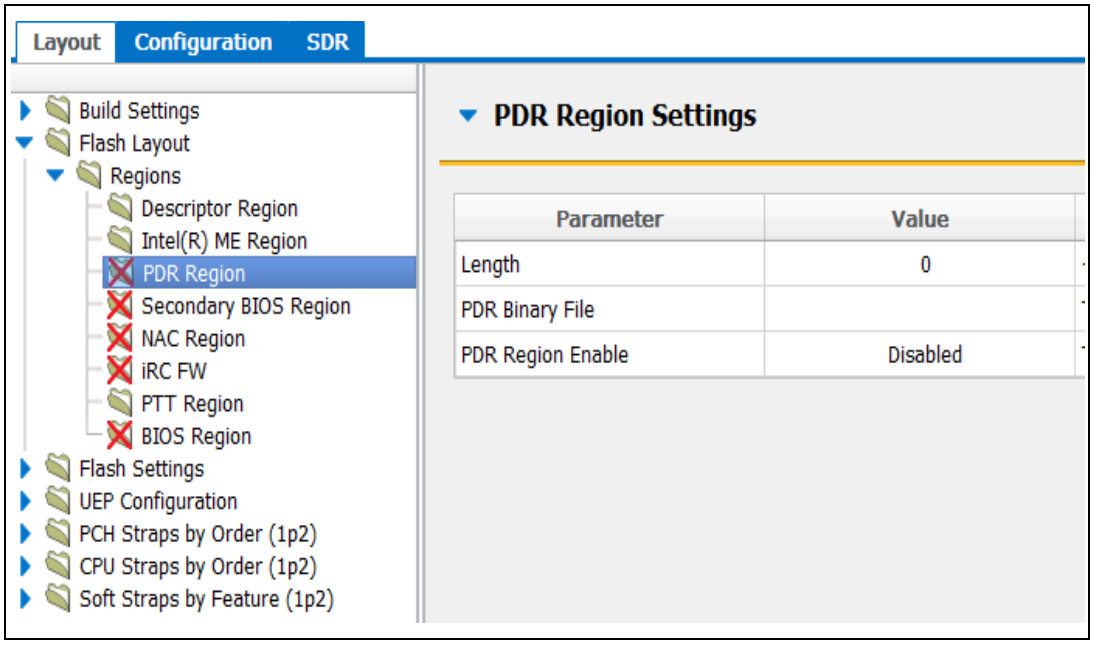
This section shows the configuration for sensors data, HW descriptions (PIA).

Any settings which are different than default value are highlighted.

3.5.9 Modifying the PDR Region

The PDR Region contains various configuration parameters that let you customize the computer's behavior.

Figure 3-158. PDR Region Options



This setting can be also overwritten from the CLI invocation (it applies to GUI settings also) by calling /PDR flag with the valid path to the bios file with the filename. For example: "*spsFITc.exe /pdr pdr.bin*" will show the GUI interface with the PDR region enabled and binary input file set to pdr.bin. More examples included.

Setting the PDR Region Length Parameter

The PDR Region length option should not be altered. A value of 0x00000000 indicates that the PDR Region will be auto-sized as described in Section 3.2.1.

Setting the PDR Region Binary File

To select the PDR region binary file:

Select **Flash Layout\Regions\PDR Region** in the left panel; the PDR Region parameters are listed in the right panel.

Double-click the **PDR Binary File** parameter; a dialog appears that lets you specify which PDR file to use.

Click **OK** to update the parameter; when the flash image is built, the contents of this file is copied into the PDR region.

Enabling/Disabling the PDR Region

The PDR Region can be excluded from the flash image by disabling it in the spsFITc.

To disable/enable the PDR Region:

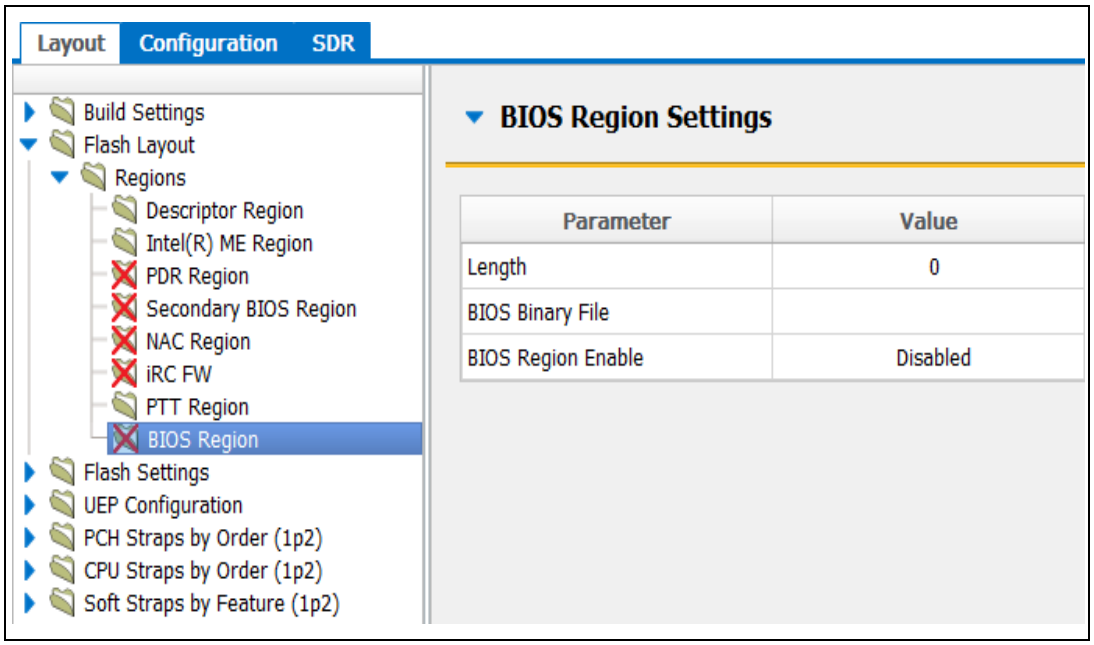
Select **Flash Layout\Regions\PDR Region** in the left panel; the PDR Region parameters are listed in the right panel.

Double-click on the **PDR Region enable** parameter; Choose appropriate option from the dropdown menu.

3.5.10 Modifying the BIOS Region

The BIOS Region contains the BIOS code run by the host processor. spsFITc always aligns this region with the end of the flash image. This is done so that if the flash descriptor becomes corrupt for any reason, the PCH defaults to legacy mode and looks for the reset at the end of the flash memory. By placing the BIOS Region at the end there is a chance the system will still boot. It is also important to note that the BIOS binary file is aligned with the end of the BIOS Region so that the reset vector is in the correct place. This means that if the binary file is smaller than the BIOS Region, the region is padded at the beginning instead of at the end.

Figure 3-169. BIOS Region Parameters



This setting can be also overwritten from the CLI invocation (it applies to GUI settings also) by calling /bios flag with the valid path to the bios file with the filename. For example: "*spsFITc.exe /bios bios.bin*" shows the GUI interface with the bios region enabled and binary input file set to BIOS.bin. More examples included.

Setting the BIOS Region Length Parameter

The value of the BIOS Region length parameter should not be altered. A value of 0x00000000 indicates that the BIOS Region will be auto-sized to fit the least possible space aligned to 4 kb of the binary image file set by the user.

Setting the BIOS Region Binary File

To select the BIOS region binary file:

Select **Flash Layout\Regions\BIOS Region** in the left panel; the BIOS Region parameters are listed in the right panel.

Double-click the **BIOS Binary File** parameter; a dialog appears that lets you specify which BIOS file to use.

Click **OK** to update the parameter; when the flash image is built, the contents of this file are copied into the BIOS region.

Enabling/Disabling the BIOS Region

The BIOS Region can be excluded from the flash image by disabling it in spsFITc.

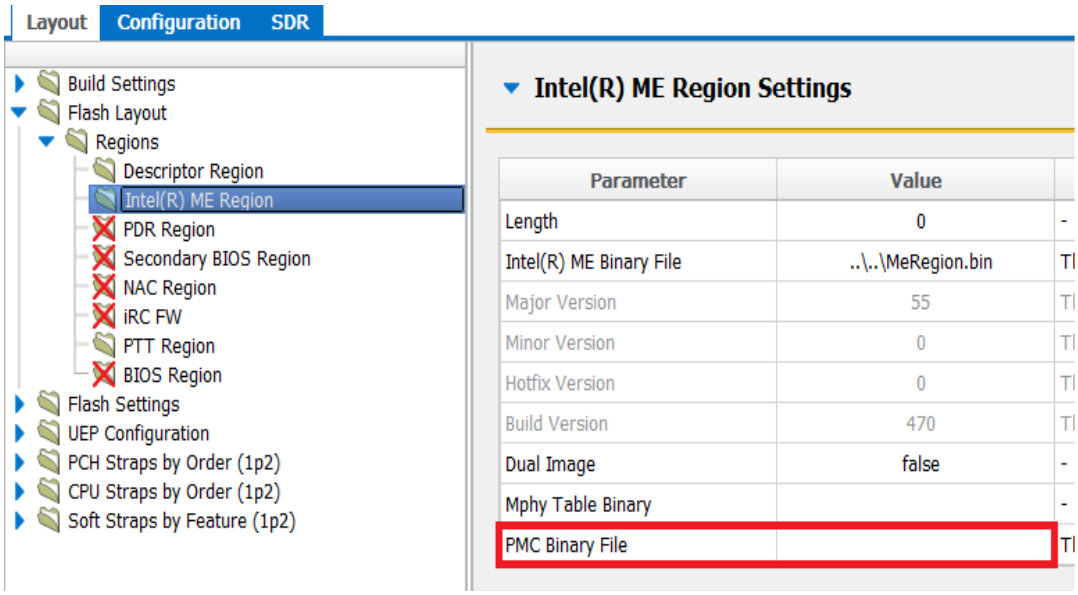
To disable/enable the BIOS Region:

Select **Flash Layout\Regions\BIOS Region** in the left panel; the BIOS Region parameters are listed in the right panel.

Double-click on the **BIOS Region enable** parameter; Choose appropriate option from the dropdown menu.

3.5.11 Modyfying the PMC Binary File

Figure 3-170 PMC Binary File parameter



Setting the PMC Binary File

To select the PMC binary file:

Select **Flash Layout\Regions\Intel® ME Region** in the left panel; the PMC Binary FIle parameter are listed in the right panel.

Double-click the **PMC Binary File** parameter; a dialog appears that lets you specify which PMC file to use.

Click **OK** to update the parameter; when the flash image is built, the contents of this file are copied into the BIOS region.

This setting can be also overwritten from the CLI invocation (it applies to GUI settings also) by calling /pmcp flag with the valid path to the PMC file with the filename. For example: "*spsFITc.exe /pmcp pmc.bin*" will show the GUI interface with the PMC Sub-

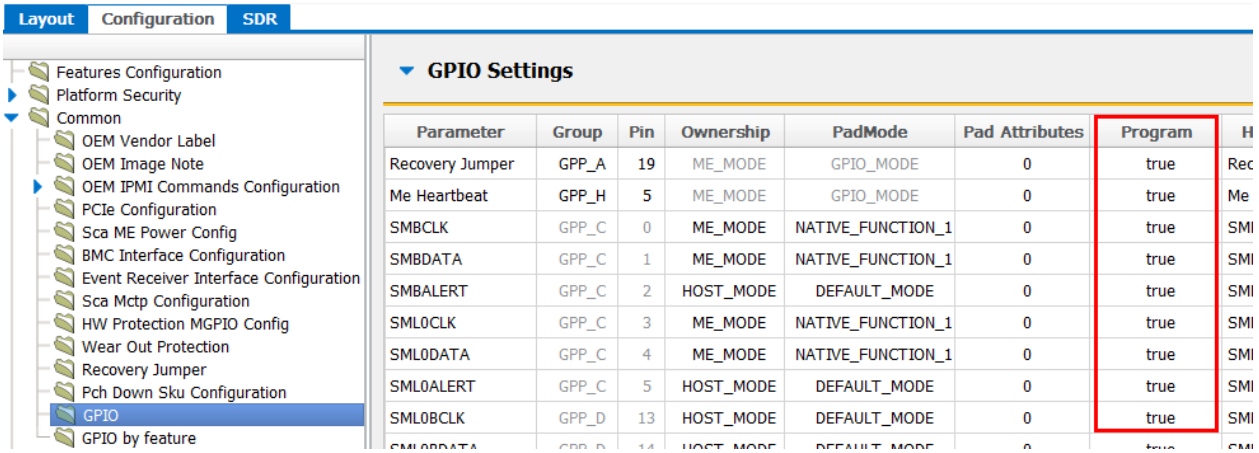
Partiton region enabled and binary input file set to pmc.bin. More examples included below.

3.5.12 Policies

Adding and removing policies is done the same way as for VSCC Table described in section 3.5.3.

3.5.13 GPIO

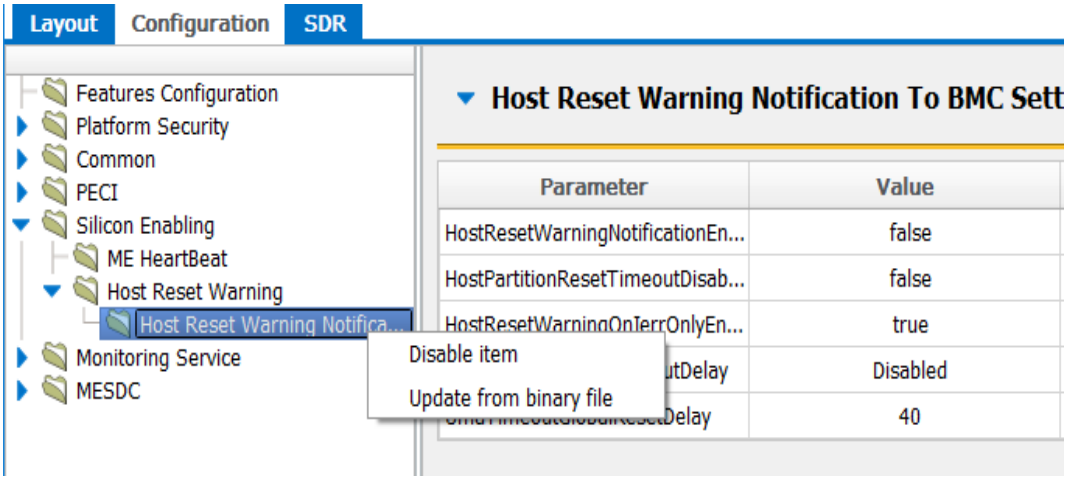
To configure GPIOs go to **Configuration \ Common \ GPIO**. Each parameter (e.g. Recovery Jumper, ME Heartbeat) has a '**Program**' property which must be set to **true** if it should be programmed into the binary image. Otherwise configuration for given GPIO is not put into the image.



Parameter	Group	Pin	Ownership	PadMode	Pad Attributes	Program	H
Recovery Jumper	GPP_A	19	ME_MODE	GPIO_MODE	0	true	Rec
Me Heartbeat	GPP_H	5	ME_MODE	GPIO_MODE	0	true	Me
SMBCLK	GPP_C	0	ME_MODE	NATIVE_FUNCTION_1	0	true	SMI
SMBDATA	GPP_C	1	ME_MODE	NATIVE_FUNCTION_1	0	true	SMI
SMBALERT	GPP_C	2	HOST_MODE	DEFAULT_MODE	0	true	SMI
SML0CLK	GPP_C	3	ME_MODE	NATIVE_FUNCTION_1	0	true	SMI
SML0DATA	GPP_C	4	ME_MODE	NATIVE_FUNCTION_1	0	true	SMI
SML0ALERT	GPP_C	5	HOST_MODE	DEFAULT_MODE	0	true	SMI
SML0BCLK	GPP_D	13	HOST_MODE	DEFAULT_MODE	0	true	SMI

3.5.14 Loading settings for Configuration file from a binary file

Some settings can be loaded from a binary file. This capability is assigned to a node in the left panel (not to a single parameter in the right panel). To load settings right-click on a node and choose "Update from binary file" option from a context menu.



Binary file must be the appropriate size (number of bytes in the file must be equal or less than number of bytes in settings).

3.5.15 Building a Flash Image

The flash image can be built with the spsFITc GUI interface.

To build a flash image with the currently loaded configuration:

Choose **Build > Build Image**

- OR -

Click on **Build** icon ( ) on toolbar

- OR -

Run in CLI mode with /b option, for example: "*spsFITc.exe /b /f mycfg.xml*".

Notice the additional „–f" or „/f" switch needed to load the xml file

spsFITc uses an XML configuration file and the corresponding binary files to build the SPI flash image. The following is produced when an image is built:

Binary file representing the image

Text file detailing the various regions and partitions in the image

Optional set of intermediate files

Multiple binary files containing the image broken up according to the flash component sizes (**Note**: These files are only created if two flash components are specified.)

Text file containing a log of validated dependency expressions (*fit_dependencies.log*)

The individual binary files can be used to manually program independent flash devices using a flash programmer. However, you should select the single larger binary file when using spsFPT.

3.5.16 Decomposing a Flash Image

spsFITc provides a flash image decomposing capability.

To decompose flash image with GUI interface go to **File > Open** and choose the binary image file.

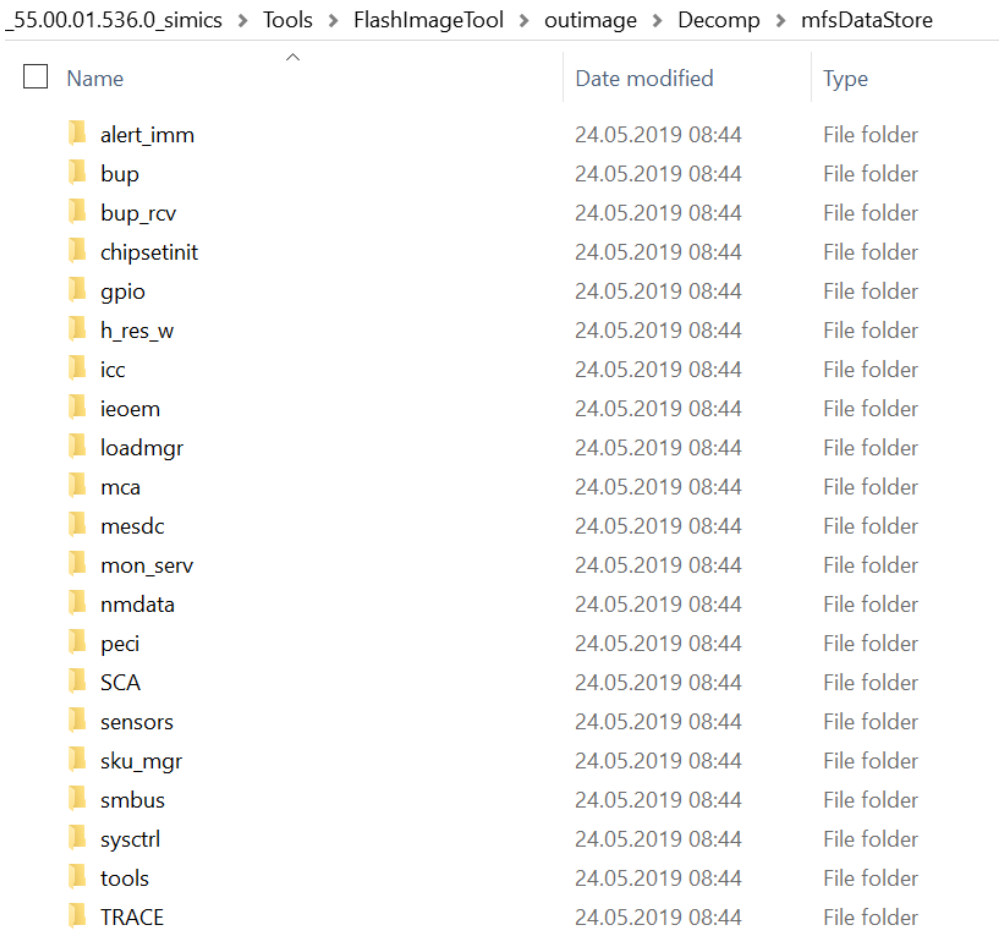
To decompose flash image in CLI mode, run spsFITc with the additional „–f" or „/f" switch to load binary image file.

spsFITc will decompose the binaries based on descriptor region, each region has one binary associated to corresponding folder. Decomposed regions are located in FlashImageTool\outimage\Decomp directory.

Name	Type	Size
mfsDataStore	File folder	
Descriptor Region.bin	BIN File	4 KB
ME Region.bin	BIN File	3 232 KB
OEM KM Binary.bin	BIN File	1 KB
oem.key.bin	BIN File	1 KB
PMC Patch.bin	BIN File	56 KB
PTT Region.bin	BIN File	96 KB
PTT Region_1fe8000h_18000h.bin	BIN File	96 KB

FlashImageTool\outimage\Decomp directory after decomposition.

MFS files are decomposed in mfsDataStore directory.

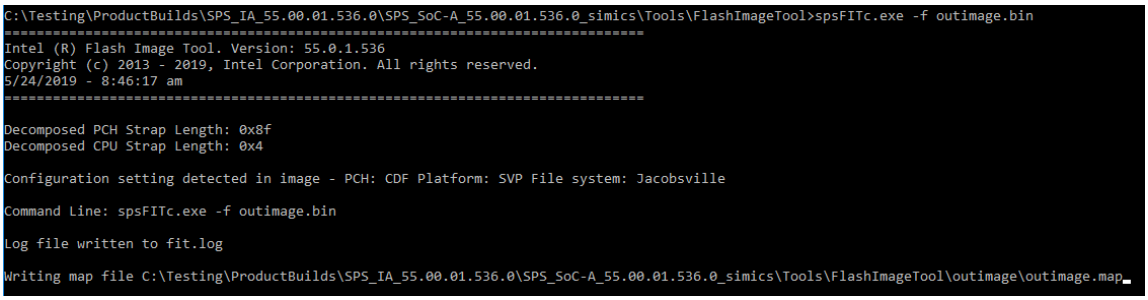


FlashImageTool\outimage\Decomp\mfsDataStore directory after decomposition.

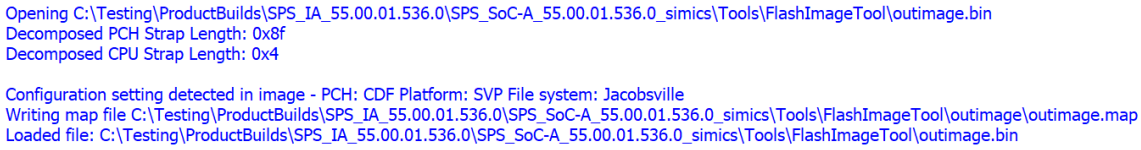
spsFITc will generate an XML which is decomposed from the binary to capture majority of the settings from the SPI image.

Decomposed XML will not be 100% accurate to the original image, some security related settings will be reset to Intel default instead of the value from the image. It is recommended to use original XML to build SPI image.

spsFITc generates a message about configuration setting detected in image and decomposed straps length.



Log after decomposing image in CLI mode.



Log after decomposing image in GUI mode.

3.6 GUI features

3.6.1 Non-default values highlighting

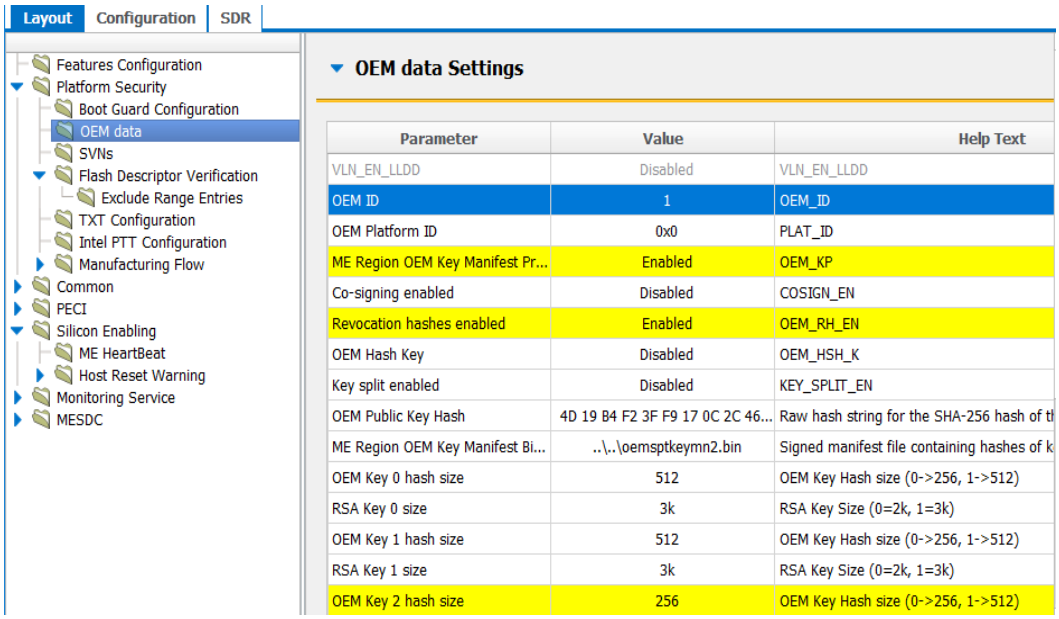
In the right panel of spsFITc GUI interface, each parameter from table view, if different then default value, is highlighted in yellow.

This feature applies only to the **Configuration** and **SDR** tabs.

List of all non-default values can be displayed by selecting:

View -> Show Non Default Settings.

Figure 3-20. Highlights example

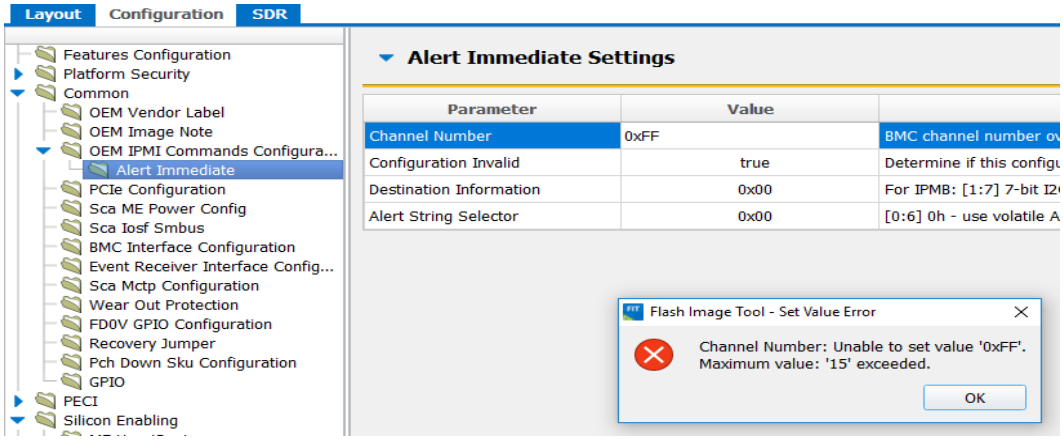


3.6.2 GUI input validation

In the right panel of spsFITc GUI interface, each parameter from table view, is validated after changing its value. In case of numeric values ranges are checked, basing on internally defined ranges and/or parameter data type. In case of string types, their length is checked in order to not exceed the maximum length.

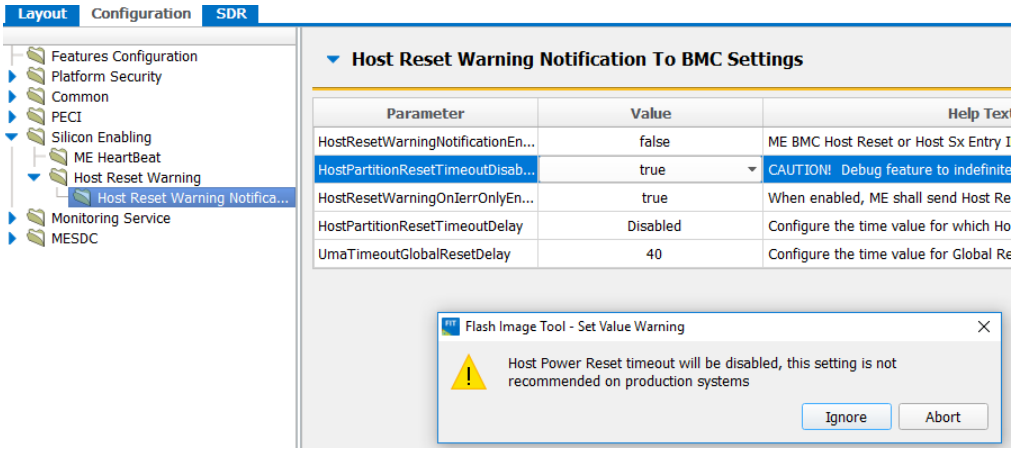
If validation fails error dialog box appears informing which constraint is exceeded.

Figure 3-23-18. GUI Input validation example



Additionally, parameters are validated against a set or range of recommended values. Setting parameter to an unrecommended value results in a warning notification. Still the desired value can be set by selecting ‘Ignore’ button. Otherwise ‘Abort’ button can be selected to return the value to its previous state.

Figure 3-192 GUI Input validation warning example



This feature applies only to the **Configuration** and **SDR** tabs.

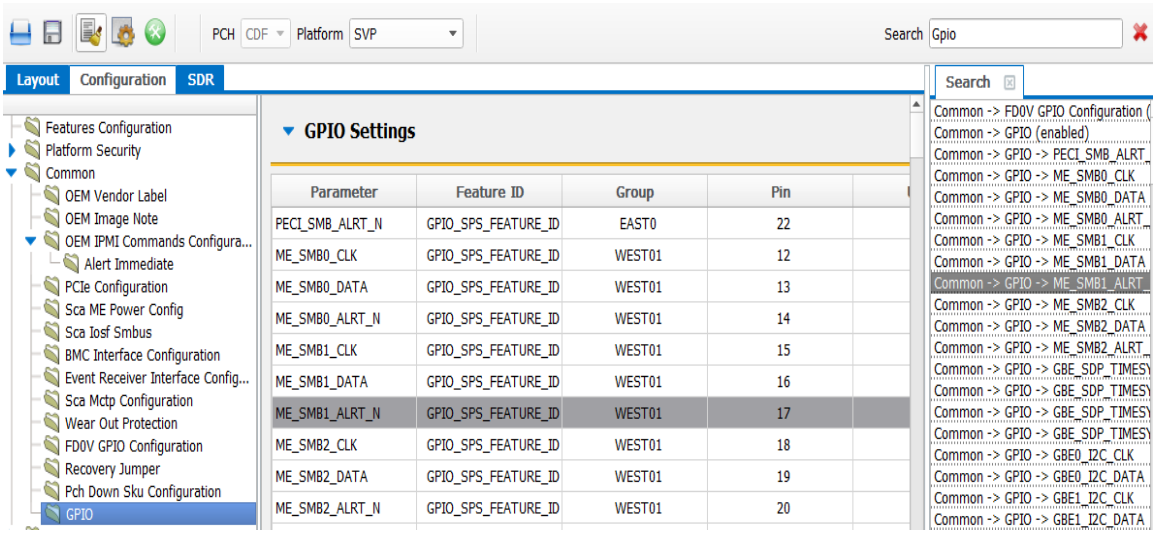
3.6.3 Search box functionality

spsFITc allows to search through all tree nodes as well as table view parameters. To start searching type a word in the search box lying in the upper-right corner of the window and approve it by clicking “enter”. Search result will appear in separate window as a list of clickable string expressions. There are two types of expressions:

string phrase containing just tree node name, clicked will lead to node expanding

string phrase containing tree node name followed by “->” and parameter name, clicked will lead to node expanding and highlighting searched parameter

Figure 3-203. Search box example



3.6.4 User notes

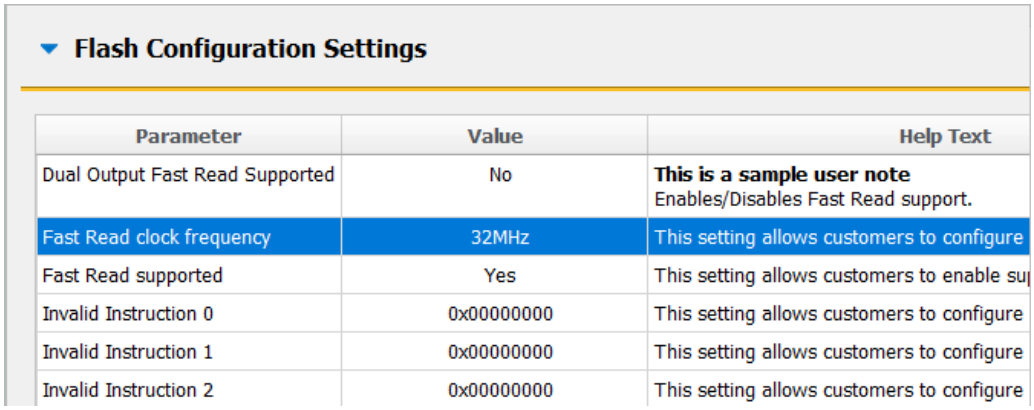
Each configuration option, presented in the right side of the window, is associated with appropriate uneditable help text which describes it. However, it is also possible to add a custom text to the field which is presented above the help text with an additional decoration to make it more visible.

If the entered note length exceeds available space in GUI, it is shortened and dotted out. The full text can be previewed in the form of a tooltip after hovering over the field.

All of the added notes are stored along within the saved configuration xml which allows them to be preserved between working sessions.

List of all defined notes can be displayed by selecting View -> Show Annotations.

Figure 3-214. Setting user note example



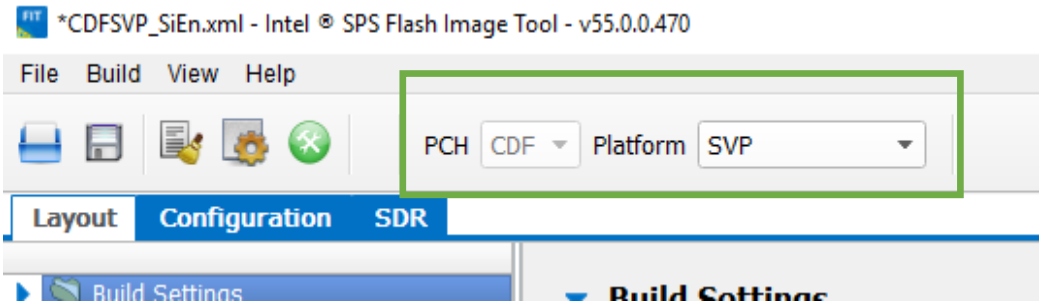
3.6.5 PCH and platform switch

FIT can support multiple PCH and platform settings. In case the package contains a number of configurations the desired one can be selected from the used interface by selecting a proper option in PCH and/or platform comboboxes located in the application toolbar area.

In case only one PCH setting is available for the tool, PCH combobox will appear disabled. However, if only one Platform option is available the corresponding combobox does not appear in the interface.

Note: It needs to be remembered that switching from one pch or platform to another discards all unsaved changes.

Figure 3-225 Platform switch location



3.7 Command Line Interface

spsFITc supports command line options.

To view all of the supported options: Run the application with the `-?` option.

The command line syntax for spsFITc is:

```
spsFITc.exe [-exp] [-h|?] [-version|ver] [-b] [-o] [-f] [-sku]
            [-me] [-bios] [-gbe] [-pdr] [-idlm] [-pchstraps]
            [-unlockToken] [-unlockTokenKey] [-der1] [-ec] [-der2] [-dualimage]
            [-rcv] [-opr] [-fd0vkey] [-sien] [-eom] [-dbgen] [-dbgenfs]
            [-validate] [-forcemerge] [-simple_xml] [-w] [-s] [-d] [-u1] [-u2]
            [-u3] [-i] [-flashcount] [-flashsize1] [-flashsize2] [-save]
            [-layout] [-afs_template] [-afs_files] [-pch] [-platform] [-cfg]
            [-bios2] [-ucode] [-ie] [-gbea] [-gbeb] [-spare1] [-spare2]
            [-new] [-oemKey] [-keyManifest] [-bootProfile]
            [-cpuDebugging] [-bspInitialization] [-pmcp] [-oemkeymn]
            [-cosignEnabled] [-oemRhEn] [-scaMePowerMode]
```

Table 3-7. spsFITc Command Line Options

Option	Description
exp	Displays example usage of this tool.
-h|?	Displays help screen.
-version|ver	Displays version of the tool.
b	Build the flash image.
-o<filename>	Overrides the output file path.
-f<filename>	Specifies input file. XML, full image binary, or ME only binary.
-sku<value>	Sets the SKU type to use.
-me<file>	Overrides the binary source file for the ME region and skips certain build steps
-bios<file>	Overrides the binary source file for the BIOS region.
-gbe<file>	Overrides the binary source file for the GbE region.
-pdr<file>	Overrides the binary source file for the PDR region.
-idlm<file>	Overrides the binary source file for the IDLM region.

Option	Description
-pchstraps<file>	Overrides the binary source file for the PCH Straps.
-unlockToken<file>	Overrides the Unlock Tocken binary file.
-unlockTokenKey<key hash>	Overrides the Unlock Token Key Hash needed to verify an OEM Unlock Token.
-der1<file>	Overrides the binary source file for the DER #1 region.
-ec<file>	Overrides the binary source file for the Embedded Controller region.
-der2<file>	Overrides the binary source file for the DER #2 region.
dualimage	Enable dual image.
-me_binary<file>	Overrides the binary source file for the me file.
-rcv<file>	Overrides the binary source file for the recovery file.
-opr<file>	Overrides the binary source file for the operational file.
-fd0vman<file>	Specifies the binary source file for the FD0v manifest.
force_signing	Suppress the warning message in case of hash differences and force signing.
sien	Silicon Enabling configuration.
-eom<0-1>	Overrides the EOM setting.
dbgen	Generates an external data file.
dbgenfs	Generates an external data file for fs.
validate	Validates both databases.
forcemerge	Suppress the warning message and force xml merge.
legacymap	Writes image map in old format.
simple_xml	Save simple xml.
-w<path>	Overrides the $WorkingDir environment variable.
-s<path>	Overrides the $SourceDir environment variable.

Option	Description
-d<path>	Overrides the $DestDir environment variable.
-u1<value>	Overrides the $UserVar1 environment variable.
-u2<value>	Overrides the $UserVar2 environment variable.
-u3<value>	Overrides the $UserVar3 environment variable.
-i<enable|disable>	Overrides the intermediate file generation.
-flashcount<0-2>	Overrides the number of flash components.
-flashsize1<0-7>	Overrides the size of the 1st flash component (0=512KB, 1=1MB, 2=2MB, 3=4MB, 4=8MB, 5=16MB, 6=32MB, 7=64MB).
-flashsize2<0-7>	Overrides the size of the 2nd flash component (0=512KB, 1=1MB, 2=2MB, 3=4MB, 4=8MB, 5=16MB, 6=32MB, 7=64MB).
-save<file>	Saves the XML file.
-layout<filename>	Overrides the path to the layout file.
-afs_template<filename>	Overrides the path to the AFS template file.
-storage_table<filename>	Overrides the path to the Storage Table file.
-afs_files<filename>	Overrides the path to the AFS files.
-pch<value>	Overrides pch.
-platform<value>	Overrides platform.
-cfg<filename>	Overrides the path to the Ftool configuration file.
-bios2<file>	Overrides the binary source file for the secondary BIOS region.
-ucode<file>	Overrides the binary source file for the Micro Code Patch region.
-ie<file>	Overrides the binary source file for the IE region.
-gbea<file>	Overrides the binary source file for the 10 GBE A region.
-gbeb<file>	Overrides the binary source file for the 10 GBE B region.

Option	Description
-spare1<file>	Overrides the binary source file for the Spare #1 region.
-spare2<file>	Overrides the binary source file for the Spare #2 region.
-oemKey<hash:hex>	Overrides OEM Public Key hash.
-keyManifest<value>	Overrides Key Manifest ID.
-bootProfile<index>	Overrides Boot Profile configuration.
-cpuDebugging<Enabled / Disabled>	Overrides CPU Debugging.

3.7.1 More Examples of spsFITc CLI

If using paths defined in the KIT, be sure to put "" around the path as the spaces cause issues.

Take an existing image (file.bin) or configuration xml (file.xml) and put in a new BIOS binary:

spsFITc.exe /b /bios "..\..\..\Image Components\BIOS\BIOS.ROM" <file.bin or file.xml>

Take an existing image (file.bin) or configuration xml (file.xml) and put in a different Intel® ME region:

spsFITc.exe /b /me ".\..\..\Image Components\Firmware\PCH_REL_BYP_ME_UPD_PreProduction_0xB0.BIN" <file.bin or file.xml>

Take an existing image (file.bin) or configuration xml (file.xml) and put in a different GbE region:

spsFITc.exe /b /gbe ".\..\..\Image Components\GbE\82577_A2_CPT_A1_VER0PT21_MOBILE.bin" <file.bin or file.xml>

4 IBST

IBST (Image Building & Signing Tool) is a command line tool used for generating binary components that may be included in Intel® Management Engine (Intel® ME).

This document covers only basic information on how to run the tool. For more information on IBST see IBSTtool_UG.docx.

4.1 System requiremenets

IBST runs on any machine with Python 3.4 installed. The tool does not have to be run on an Intel ME-enabled system. Also following modules must be installed:

- cryptography
- lxml

There is a requirements.txt file which lists all required modules among with their versions (the two modules above plus dependent modules that are installed automatically). To install them use following command:

```
pip install –r requirements.txt
```

4.2 Usage

As a mandatory input IBST takes an xml file that describes layout and values of the binary file to be generated. Those files are included in the package. Typically OEM will provide all other input files like Private/Public Key.

4.2.1 Example: creating FD0V Manifest

Use FD0V_Manifest.xml configuration file for FD0V manifest generation. At least two settings must be specified: private key (‘key’ setting in xml file) and extension binary (‘extension_binary’ setting). Use the command below to generate FD0V manifest:

```
python ibst.py FD0V_Manifest.xml -s key=private_key.pem
extension_binary="Descriptor Manifest Extension.bin"
```

5 MESDC Tool

5.1 MESDC Tool Overview

Intel ME Debug and Compliance Console (MESDC) is an application to diagnose Intel ME Firmware. This tool performs run-time tests, receives Intel ME Firmware Status (defined in [ME_BIOS_Interface]) and Trace Logs, processes and presents received data.

MESDC consists of the following elements:

SOFTWARE

- MESDC.exe – Windows (Net Framework 4.0) based application;
- Aadrvark.dll – dynamic library to communicate with Aardvark adapter;
- Common.dll – dynamic library with MESDC and MESDC Agent common functions
- Heci.dll – dynamic library to communicate with ME FW via HECI
- TransportRmcpp.dll – dynamic library to communicate with BMC
- Total Phase USB driver for Windows;

HARDWARE

- USB A(M)-B(M) cable;
- Aardvark* I^2C* host adapter;
- 3-wire SMBus cable;
- MDDD Mobile DIMM Adapter / MDDD Desktop DIMM Adapter;

5.2 Installation and Initialization

This section covers installing software and hardware components described in the following sections.

5.2.1 Software installation – USB Driver

To install the appropriate USB communication driver under Windows*, use the Total Phase USB Driver Installer before plugging in any device. The driver installer can be found either on the CD-ROM (use the HTML based guide that is opened when the CD is first loaded to locate the Windows installer), or in the Downloads section of the Aardvark adapter product page on the Total Phase website.

After the driver has been installed, plugging in an Aardvark adapter for the first time will cause the adapter to be installed and associated with the correct driver.

5.2.2 Software installation – MESDC Application

MESDC application is compatible with Windows 7 x86/x64, Windows 8 x64, Windows Server 2008 R2 SP1 x64, Windows Server 2012 x64 with installed .NET Framework 4.0 or higher.

To communicate with Intel ME FW MESDC uses Aardvark.dll, Heci.dll or TransportRmcpp.dll library which should be located in one of the following places:

The directory from which the application binary was loaded;

The application's directory;

System directory, for example, C:\Windows\System32;

The windows directory, for example, C:\Windows

The directory listed in the PATH environmental variable.

5.2.3 Intel ME Image Preparation

When building the image with Intel® spsFITC, check that the following soft strap is set correctly.

Intel® SMBus Address = 0x38 (within Intel® spsFITC under "Configuration -> MESDC -> SMBus Address" section). If 0x38 conflicts with another device on SMBus, this address can be set to a different value. If set to value other than 0x38, specify it in the "MESDC I2C Address [hex]" field in MESDC console under the SMBus Settings in the Configuration menu.Hardware Configuration

3-wire SMBus Configuration

The following steps must be taken to set proper SMBus configuration:

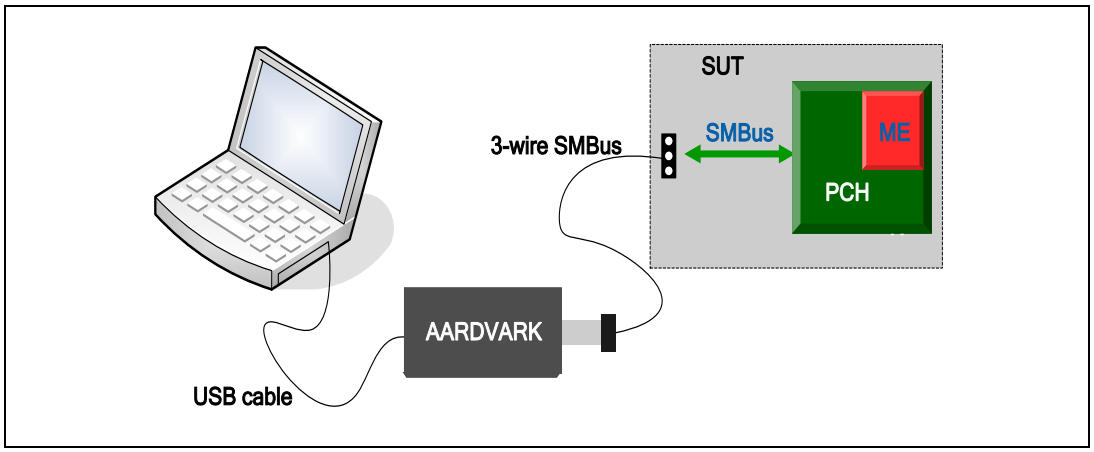
Disconnect AC power,

Connect Aardvark adapter to 3 wire SMBus,

Connect Aardvark adapter to host computer,

Connect AC power.

Figure 5-1. 3-Wire SMBus Configuration



IPMB Configuration

The following steps must be taken to set proper IPMB configuration:

Disconnect AC power,

Connect Aardvark adapter to 3 wire SMLink0,

Connect Aardvark adapter to host computer,

Connect AC power.

RMCP+ Configuration

The following steps must be taken to set proper RMCP+ configuration:

Configure BMC IP address,

Add new BMC user: username and password must be non-empty,

Connect platform to LAN network.

Remote Agent Configuration

The following steps must be taken to set proper Remote Agent configuration:

Install Remote Agent on tested platform and copy required SSL keys. Remote Agent application is compatible with Windows Server 2012 R2 x64, Windows Server 10 x64, Red Hat 7.1 x64

Connect platform to LAN network

HECI Configuration

The following steps must be taken to set proper HECI configuration:

MESDC Application should be run on tested platform.

5.3 MESDC Application

5.3.1 Introduction

Intel ME Debug and Compliance Console (MESDC) is an application to diagnose Intel ME Firmware.

These are the basic steps for start working with MESDC application. For more detailed information, please refer to the specific sections in this manual.

Select Interface (4.4 Initialization of MESDC Application)

For proper MESDC initialization current Intel ME FW Configuration must be read. (more details in section 4.3.3.)

The main application form is divided into five sections:

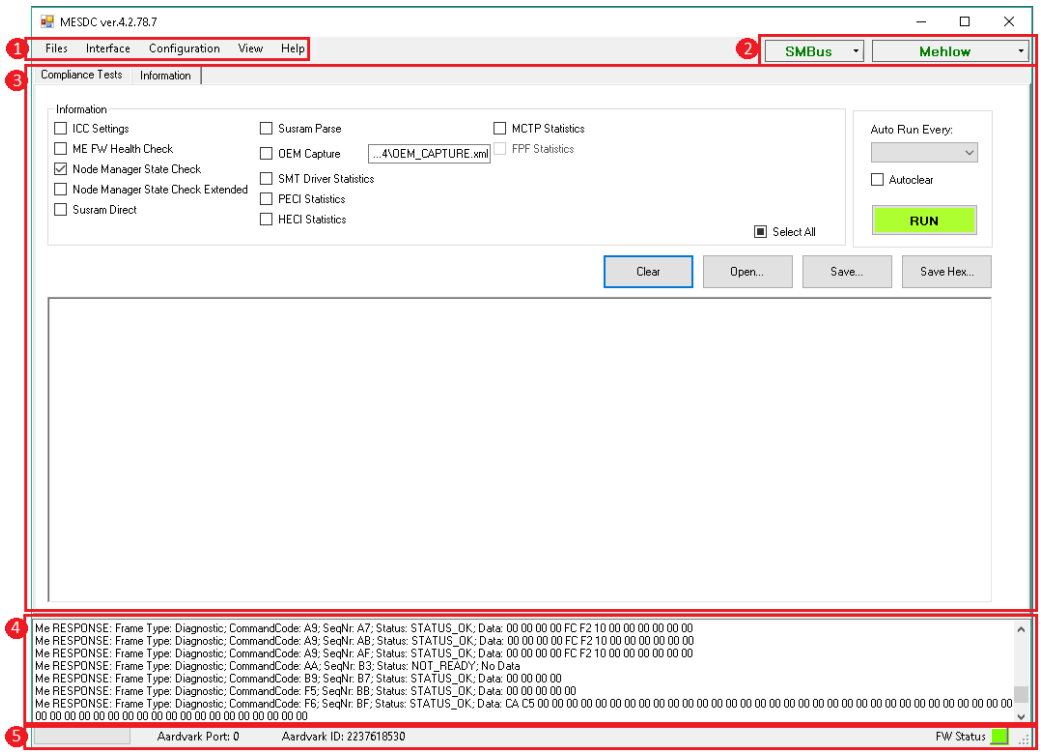
Main menu

Toolbar

Diagnostic modules in tabbed pages

Operation Log, which keeps track of all operations.

Statusbar



Main menu

MESDC main menu contain 5 items

Files

Save

Gather Debug Information: Run all available reports and save their results to zip file, additionally saves operational log.

Save Current Information: Gets logs from Reports, Trace Console and Operational log and save them to zip file.

Save trace logs: save trace logs into a file for future analysis

FW Configuration:

Get FW Configuration: read FW configuration from connected platform

Override All Features to ON: enables all features in MESDC for specific platform type, available only for advanced view

Exit: exit the MESDC application

Interface: MESDC can communicate with Intel ME through one of the following interfaces:

SMBus

MEI (HECI)

IPMI (RMCP+)

IPMB (Aardvark)

Agent (TCP)

Configuration: details configuration settings (see section 4.4 Initialization of MESDC Application):

SMBus Aardvark

MEI (HECI)

IPMI (RMCP+)

IPMB (Aardvark)

Agent (TCP)

AutoConnect: Available only for advanced view

Auto Get FW Configuration: Available only for advanced view

Auto Fetch Report: Available only for advanced view

View

Basic: Basic view will only provide 1 module (Information) for simple usage of the MESDC

Advanced: Advanced view will provide all the stages for advanced user

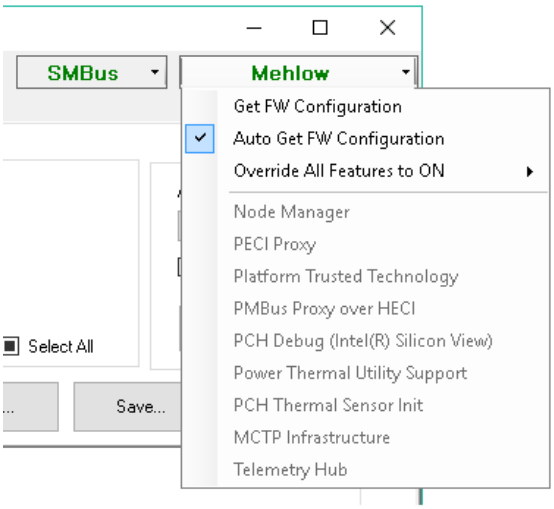
Help

About

Platform Type Information

MESDC Toolbar

MESDC shows state of selected interface and identified platform type with current ME features.



Diagnostic modules

In Basic View MESDC application consists of one stage of operation to diagnose Intel ME FW at run-time. All modules are described in section 5.5 MESDC Application Modules.

MESDC module in Basic View:

Compliance Tests

Information

In Advanced View MESDC application consists of six stages of operation to diagnose Intel ME FW at run-time. All modules are described in section 5.5 MESDC Application Modules.

MESDC modules in Advanced View:

Trace Console

Communication

Information

IDLM

Compliance Tests

Charts

Operation Log

Operation Log keeps track of all received response frames, communication errors and Intel ME Firmware state.

Statusbar

StatusBar contains basic information about working (1), chosen interface (2), read FW Status(3).

FW Status indicator colors:

green – FW active

yellow – recovery, test or init state

red- disabled, transition, wait or reset state

grey – unrecognized state

5.3.2 Autorun features

Autorun features lunch when MESDC starts and provide basic information about current platform. They can be disabled only in advanced view. Received ME features state and platform type allow enabling specified reports, diagnostic commands and compliance tests. When ME configuration status is unavailable there is a possibility to override all features for specified platform.

AutoConnect

Auto Connect: if it is selected when MESDC starts it will try to connect with ME through previous setting of interface automatically.

In basic view there is no possibility to disable this feature.

AutoGet

Auto Get FW configuration: If this is selected, at the time launch of MESDC, MESDC will get the FW configuration automatically and apply that to the compliance. If user wants to override this configuration, user can use Override Features Functionality option in the menu to override the configuration.

In basic view there is no possibility to disable this feature.

Auto Fetch Report

Fetch Report: If this option is selected, MESDC will runreports in information tag automatically when it's launched.

In basic view there is no possibility to disable this feature.

Override Features Functionality

This functionality allows overriding all features specific to selected platform to ON. It can be used when there is no possibility to read ME FW configuration and platform dependant diagnostic commands, compliance tests or information reports are unavailable.

Setting MDES logging interface

Intel ME Firmware is able to send MESDC Traces via Host SMBus or write them to SPI Flash. MDES Logging Interface option can be enabled/disabled by changing proper MFS MDES fields in the XML configuration file or via a diagnostic command. For RMCP+, IPMB, Agent or HECI interfaces Flash Logging enables when 'Write MESDC Traces' checkbox in interface settings is selected. When SMBus Interface is chosen, sending MESDC Traces via Host SMBus are automatically enabled. To apply new MDES Logging Interface settings ME Reset is required.
In basic view there is no possibility to disable this feature.

5.4 Initialization of MESDC Application

There are five communication options between MESDC application and target testing system:

SMBUS with Aardvark

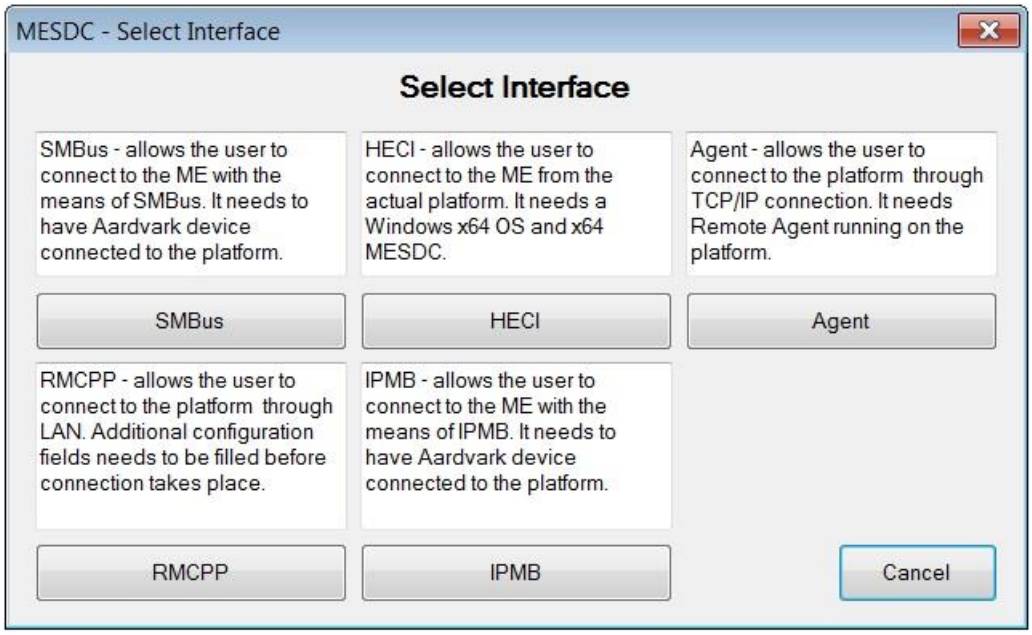
MEI (HECI) on local platform

IPMB with Aardvark

RMCP+

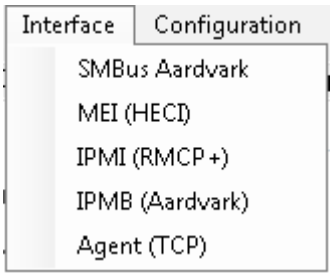
Remote Agent

The first time launch the MESDC tool, following dialog will pop-up for user to choose which interface user wants to connect with Intel ME



When user selects one of the interfaces, some detail level of the setting may be needed for MESDC to work with that interface. The detail configuration setting is available in the configuration menu

Figure 5-2. Communication Configuration for MESDC



5.4.1 SMBus

To communicate with Intel ME FW via SMBUS, a proper Aardvark adapter has to be chosen. After launching the MESDC application ConfigAA form must be opened from menu Interface->SMBus Aardvark. There is a list of all available Aardvark adapters connected to the computer. If there are no available units, then application displays warning message: "Aardvark ERROR: NOT_CONNECTED Aardvark adapter". The list provides the following information:

Port – the port that Aardvark adapter occupies, zero-based number;

FW – Firmware version of Aardvark adapter;

SW – Software version of Aardvark adapter;

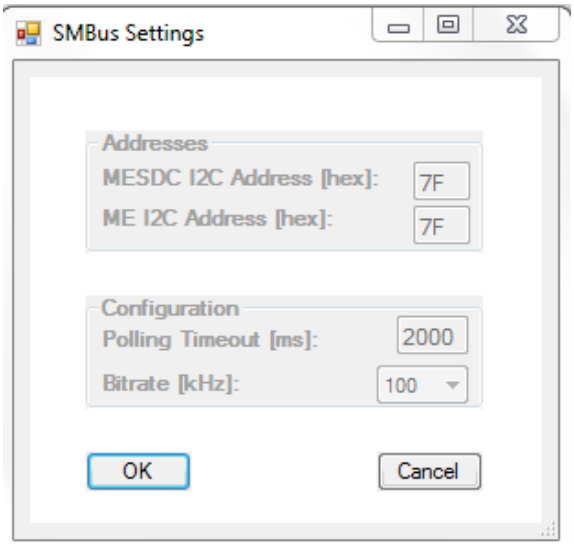
Serial Number of Aardvark adapter.

Figure 5-3. Aardvark Adapter Choosing form of MESDC

It is possible to reinitialize the Aardvark adapter.

After successful connection information about connection parameters are displayed on StatusBar: Aardvark Port, Aardvark ID

User can change some of the SMBus configuration at SMBus Setting dialog. The default setting would work for most of the cases.



5.4.2 IPMB

To communicate with Intel ME FW via IPMB, a proper Aardvark adapter has to be chosen. After launching the MESDC application ConfigAA form must be opened from menu Interface->IPMB Aardvark. There is a list of all available Aardvark adapters connected to the computer. If there are no available units, then application displays

warning message: "Aardvark ERROR: NOT_CONNECTED Aardvark adapter". The list provides the following information:

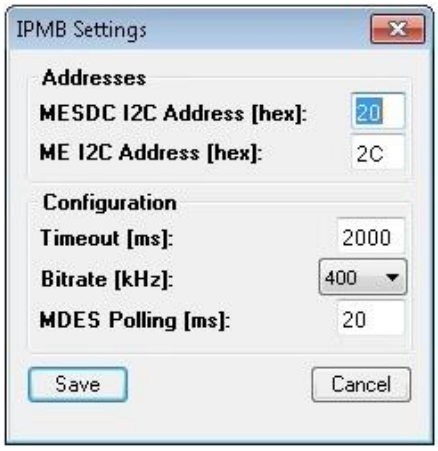
Port – the port that Aardvark adapter occupies, zero-based number;

FW – Firmware version of Aardvark adapter;

SW – Software version of Aardvark adapter;

Serial Number of Aardvark adapter.

User can change some of the IPMB configuration at IPMB setting dialog. The default setting would work for most of the cases.

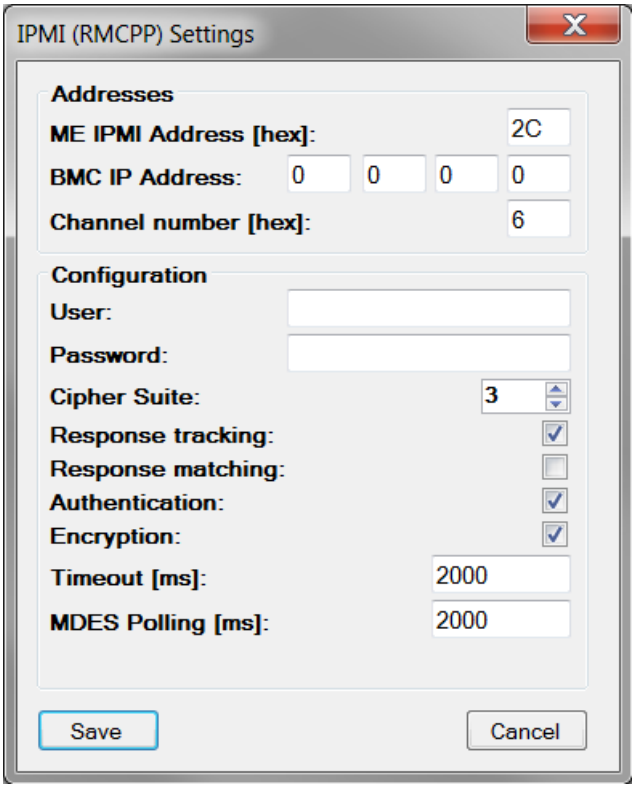


5.4.3 RMCP+

To communicate with Intel ME FW via IPMI proper BMC configuration is needed. User should provide ME IPMI and BMC addresses, account data and encryption settings in the IPMI configuration section to make MESDC work through RMCP+ interface.

Response matching is the only functionality not related directly to IMPI or RMCP+. This option applies only when Response tracking option is selected. For default this option is turned off. When turned off MESDC checks only if received response has the same command code as the sent request. When turned on MESDC additionaly checks if response has valid *netFn*, *rqLUN*, *rqSeq* and *rsLun* fields values.

Figure 5-4. IPMI Setting for MESDC



After successful connection information about connection parameters are displayed on StatusBar: IP and user

5.4.4 Remote Agent

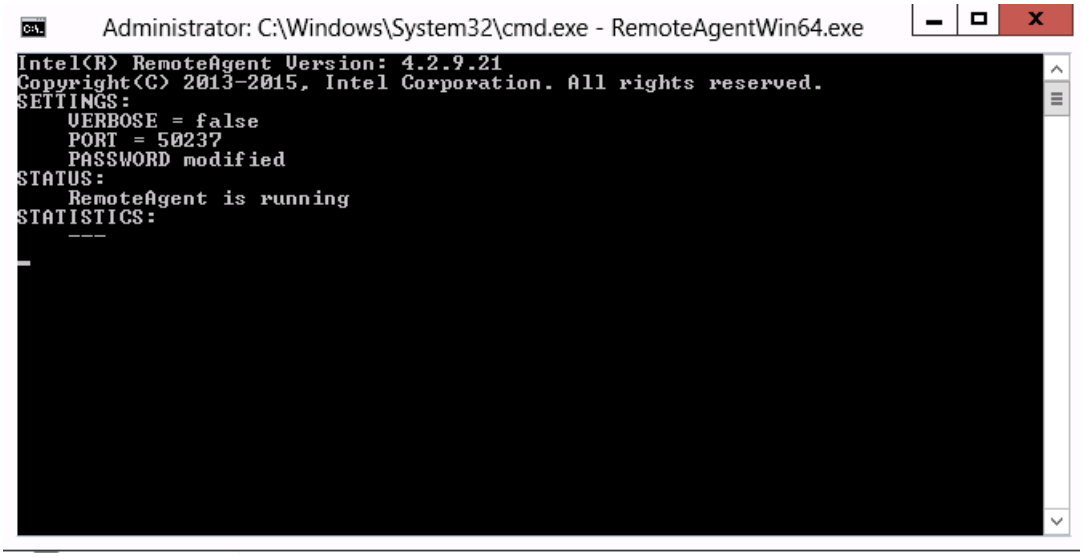
To communicate with Intel ME Firmware via HECI interface using Remote Agent, RemoteAgent.exe application must be running on the platform.

RemoteAgent requires OpenSSL 1.0.2 shared libraries.

On Windows shared libraries libeay32.dll, ssleay32.dll must be present in the same directory as Agent or accessible via environment variable PATH.

In Linux needed libraries are libssl.so.10, libcrypto.so.10. In case used distribution has different names for openssl libraries symbolic links must be created (example: ln -s /lib64/libssl.so.1.0.0 /lib64/libssl.so.10).

Although Remote Agent uses HECI interface only diagnostic commands can be sent. HECI commands are not available.



Remote Agent has the following optional startup parameters:

-?|-H|-HELP
Displays help screen.

-VER|-VERSION
Displays version information.

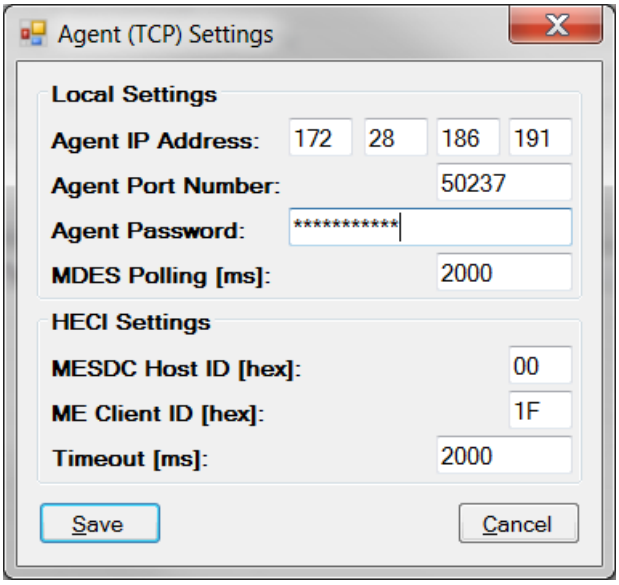
-V|-VERB|-VERBOSE [filename]
Displays the debug information of the tool.
Debug information is shown in the form of ten raw frames lately exchanged with client.
Additionally if filename is specified a full log file from Remote Agent work is created.

-P|-PORT [number]
Sets the port on which Server will listen for incoming requests.
If no port number is specified the default value will be used.

-PASS|-PASSWORD [password]
If password is specified overwrites the default Agent password,
else a prompt for password is shown first.

ARTO [data]
Application running time out. Data format: HH:MM:SS or ‘infinite’

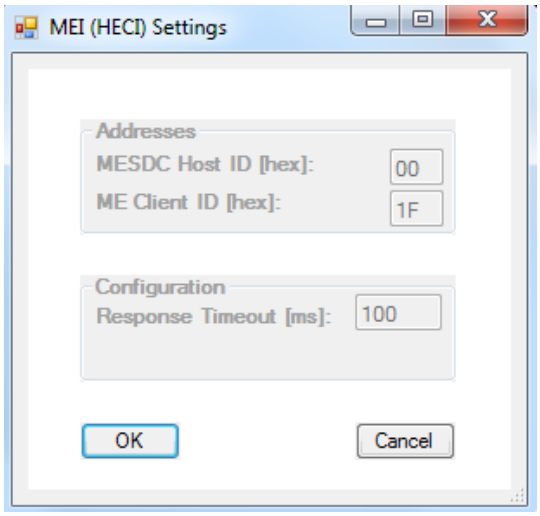
To connect with Remote Agent following TCP configuration has to be done.



After successful connection IP Address and Port Number are displayed on the Status Bar.

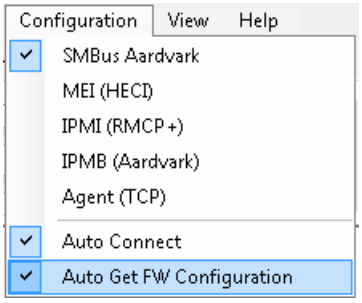
5.4.5 HECI

User can change some of the HECI configuration at HECI setting dialog. The default setting would work for most of the cases.



5.5 MESDC Application Modules

Configuration



MESDC application consists of stages of operation to diagnose Intel ME FW at run-time.

If any of modules below is working, tab page header of this module is highlighted.

5.5.1 Trace Console

In this mode of operation, MESDC receives and parses run-time SMBus messages. Trace Console allows analyzing the firmware flow, in particular the initialization sequence. There are three major types of messages: SVEN String (which is presented as a text message with a timestamp), SVEN Catalog (which is presented like SVEN String or ID number with parameters) and Sven Custom (which is parsed in a more extended way).

MESDC is configured as an SMBus Slave. User can use the Trace Console tab to configure the ME Debug Event Service (MDES) in Intel ME FW so as to report appropriate types of messages for an analysis. Trace Console can be used to:

set dictionary for SVEN Catalog messages. Default dictionary is in SVEN directory, "<…>" options allows to select file outside of SVEN directory.

set logger on/off

set error level to log messages with severity higher or equal to: Critical, High, Low Errors, or Information

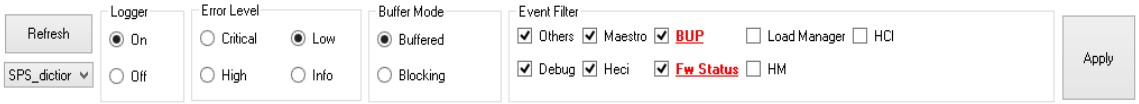
set event filter to log selected events

set buffer mode to determine the way how MDES reports the messages. By default the messages are sent in buffered mode in order not to introduce much load to the Intel ME FW. The blocking mode is designed to help to investigate problems during boot time

set event filters on/off. This controls which groups of messages are sent by ME FW.

To apply changes ME FW needs to be restarted. Settings which are changed but not applied yet (due to lack of ME Reset) are underlined and are in red. User needs to click Apply button to set new logger configuration.

Logger configuration can be received from Intel ME FW by pressing the 'Refresh' button.



To start/stop/clear displaying messages press the 'Capture', 'Stop' or 'Clear' button. User can switch between tabs and perform other operations while the trace is being captured.

Parsed messages can be saved in a file by selecting the 'Files->Save Logs' menu. MESDC will create a file called *log_YYYY_MM_DD_HH_MM_SS.rtf* in the same folder where MESDC tool is located.

Firmware must have proper logging interface set, otherwise it will not send any traces. MESDC checks this every time after pressing "Capture" button. If logging interface must be changed then user is asked if he agrees to perform ME Reset, which is needed to change logging interface. Logging interface can be also checked on MESDC startup. To enable this go to settings of the interface to be used (in 'Configuration' menu) and check 'Write MESDC Traces (Calls ME Reset)' option. This option is visible only in advanced view. On startup, MEDC will NOT ask user for permission to perform ME Reset if logging interface must be changed.

SMBus settings such as MESDC and Intel ME I2C address can be set by selecting the 'Configuration->SMBus Settings' menu. There are following default values of I2C addresses (7-bit format):

MESDC trace address = 0x38;

Intel ME address = 0x48;

The filter settings in the MDES emergency mode are fixed and can't be changed with use of MESDC.

Trace Logger Mode of Data Reception

Majority of the logs will be sent during boot time. It is allowed to catch trace logs via Ipmb, Rmcp+, Heci interface. In this case special settings in MDES are needed (see section 0.0.0 Setting MDES logging interface

In order to receive the log at boot time via SMBus:

Once all the hardware is set up and Intel ME image is prepared to enable diagnostic service, start the MESDC on a Host Platform while DUT is off (G3 or S5).

Press the 'Capture' button to enable displaying logs.

Power on the DUT and you will see the boot time logs displayed on MESDC.

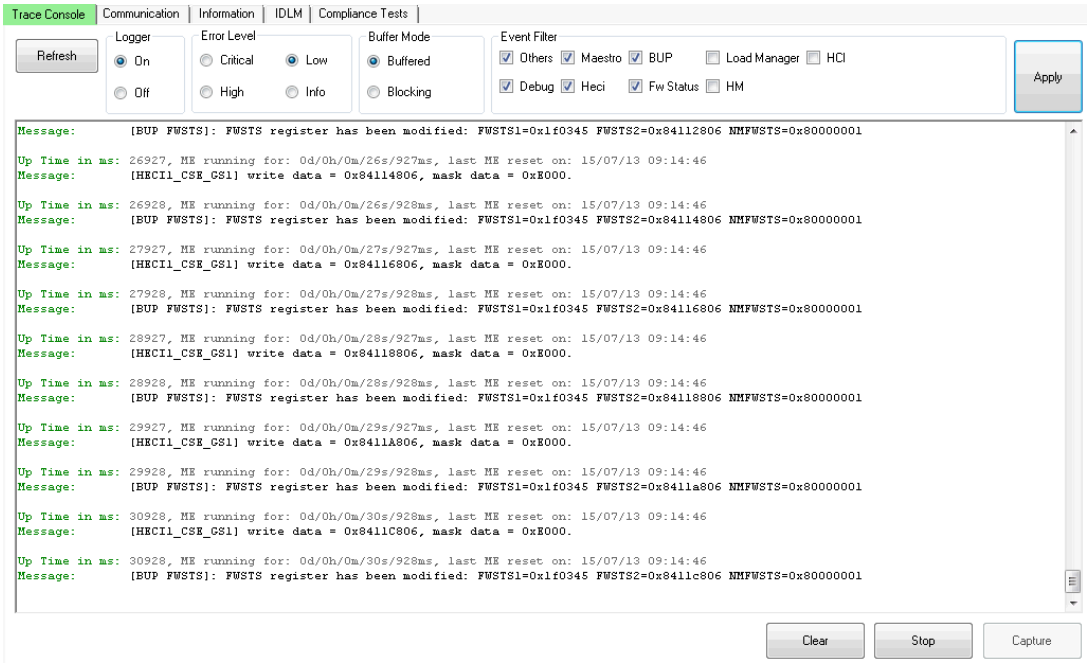
In order to receive the log at boot time via RMCP+/HECI/Agent:

Press the 'Capture' button to enable displaying logs.

If all logs are read MESDC automatically STOP.

Changes in FwStatus Events (Firmware Status frame)are highlighted.

Figure 5-5.MESDC GUI – Trace Console Tabbed Page



5.5.2 Communication

Diagnostics tab

In this mode MESDC start a conversation with Intel ME FW to execute diagnostics by Intel ME FW and receives a response with results.

Diagnostic tab page consists of the following elements:

Element	Description
Command Group	Command Group is used to select a group with the command to be executed. User can select one of the following groups: - Diagnostics - System - Intel Node Manager - Tests - Statistics - MDES
Request	Request group is used to configure SMBus request. User can choose a proper request (like AUX or Memory read) from Command drop down menu and specify the value in hex. Then a request is sent to Intel ME by clicking RUN button
Request Frame	Displays the entire Request: Addr - SMBus slave address of Intel ME Type - type of SMBus message Len - number of data bytes in SMBus frame Comm - command ID Seq.Nr - sequence number of request Data Bytes - data in SMBus frame
Response	Displays the result of executed diagnostics in a decoded format
Intel ME Response Frame	Displays the entire Intel ME Response: Addr - SMBus slave address of receiver Type - type of SMBus message Len - number of data bytes in SMBus frame Comm - command ID Seq.Nr - sequence number of response Status - generic status code Data Bytes - data in SMBus frame
Auto Refresh	User can keep sending MESDC command by checking Auto Refresh. If Auto Refresh is enabled Communication indicator on statusbar blinks green every iteration.
Operation Log	Displays communication errors

More detail MESDC supported commands are available in Appendix B.

0x3CC3A55A is the magic number sending to Intel ME for Intel ME reset command.

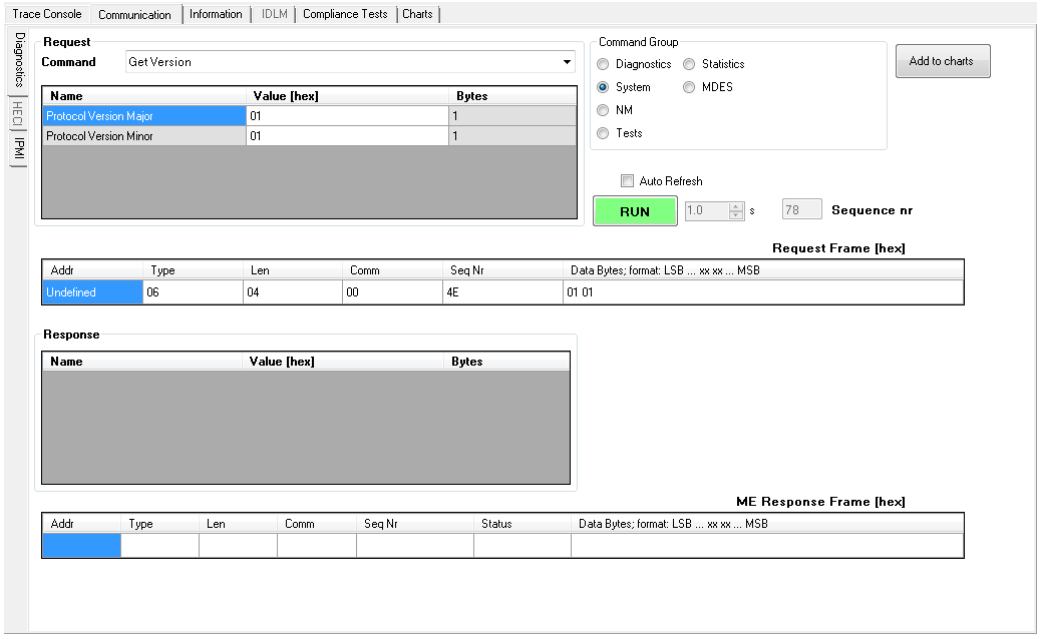


Figure 5-6. MESDC GUI–Communication Page

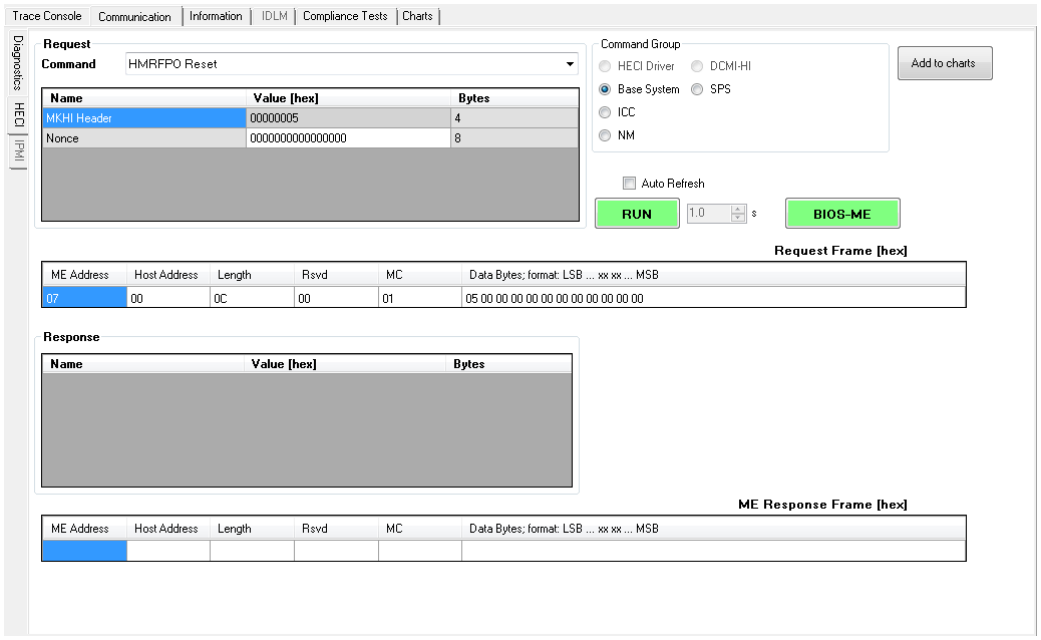
HECI tab

HECI tab allows user to send HECI command to Intel ME FW.

HECI tab page consists of the following elements:

Element	Description
Command Group	Command Group is used to select a group with the command to be executed. User can select one of the following groups: - Base System - ICC - Intel Server Platform Services
Request	Request group is used to configure HECI request. User can choose a proper request (like ICC Set Clock Enables) from Command drop down menu and specify the value in hex. Then a request is sent to Intel ME by clicking RUN button
Request Frame	Displays the entire Request: Me Address – Intel ME client ID, part of HECI Header Host Address – Host Client ID, part of HECI Header Length - number of data bytes in HECI frame, part of HECI Header Rsvd – reserved bits MC – Message Complete, part of HECI Header Data Bytes - data in HECI frame
Response	Displays the result of executed command in a decoded format
Intel ME Response Frame	Displays the entire Intel ME Response:

Element	Description
Command Group	Command Group is used to select a group with the command to be executed. User can select one of the following groups: - Base System - ICC - Intel Server Platform Services
	Me Address – ME client ID, part of HECI Header Host Address – Host Client ID, part of HECI Header Length - number of data bytes in HECI frame, part of HECI Header Rsvd – reserved bits MC – Message Complete, part of HECI Header Data Bytes - data in HECI frame
ME-BIOS	User can simulate Intel ME-BIOS communication by invoking 'ME-BIOS' button. MESDC application sends commands: MKHI Get FW Version Get Intel ME-BIOS Interface HMRFPO Lock End Of Post
Auto Refresh	Checking "Auto Refresh" checkbox will make the MESDC automatically send the command every n seconds, where n is a specified time interval left to the Run button. If Auto Refresh is enabled Communication indicator on statusbar blinks green every iteration.
Operation Log	Displays communication errors

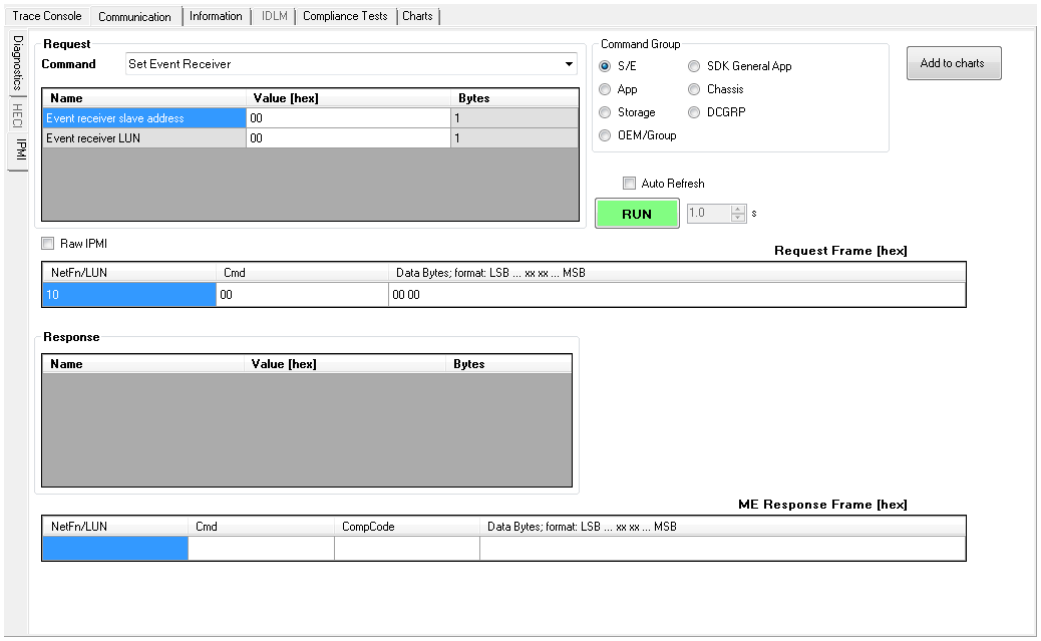


IPMI tab

IPMI tab allows user to send IPMI command to Intel ME FW.

IPMI tab page consists of the following elements:

Element	Description
Command Group	Command Group is used to select a group with the command to be executed. User can select one of the following groups: - S/E - App - Storage - OEM/Group - SDK General App - Chassis
Request	Request group is used to configure IPMI request. User can choose a proper request (like Get Device ID) from Command drop down menu and specify the value in hex. Then a request is sent to Intel ME by clicking RUN button
Request Frame	Displays the entire Request: NetFn/LUN Cmd - command ID Data Bytes - data in IPMI frame
Response	Displays the result of executed command in a decoded format
Intel ME Response Frame	Displays the entire Intel ME Response: NetFn/LUN Cmd - command ID Data Bytes - data in IPMI frame
Raw IPMI	User can send raw IPMI frame by checking "Raw IPMI" checkbox. Request Frame fields will be editable and can be overwritten by user.
Auto Refresh	Checking "Auto Refresh" checkbox will make the MESDC automatically send the command every n seconds, where n is a specified time interval left to the Run button. If Auto Refresh is enabled Communication indicator on statusbar blinks green every iteration.
Operation Log	Displays communication errors



In order to swiftly find desired command in all currently available commands on each tab, user may want to try search and autocomplete functionality, simply by typing characters in editable portion of the combo box. List with commands' names containing entered sequence will be displayed.

Bit fields

Name	Value [hex]	Bytes
CPU Index	00	1
-Reserved	00	[7:6]
-PECI Client Address	00	[5:0]
PECI Interface Selection	00	1
-PECI Interface Selection	00	[7:5]
-Reserved	00	[4:0]

Some of byte fields in command requests and responses are divided into bit fields, which are situated below corresponding byte field. Name of bit field is indented and is preceded by dash. "Bytes" column consist range of each bit field starting from MSB to LSB ([MSB:LSB]). Bit notation starts from zero. Editing value of bit field will cause value update of corresponding byte field and conversely, changing value of byte field will affect bit field values.

Conditional fields

In specific commands description of data is conditional. Any change of condition value will cause change of data grid structure. For example:

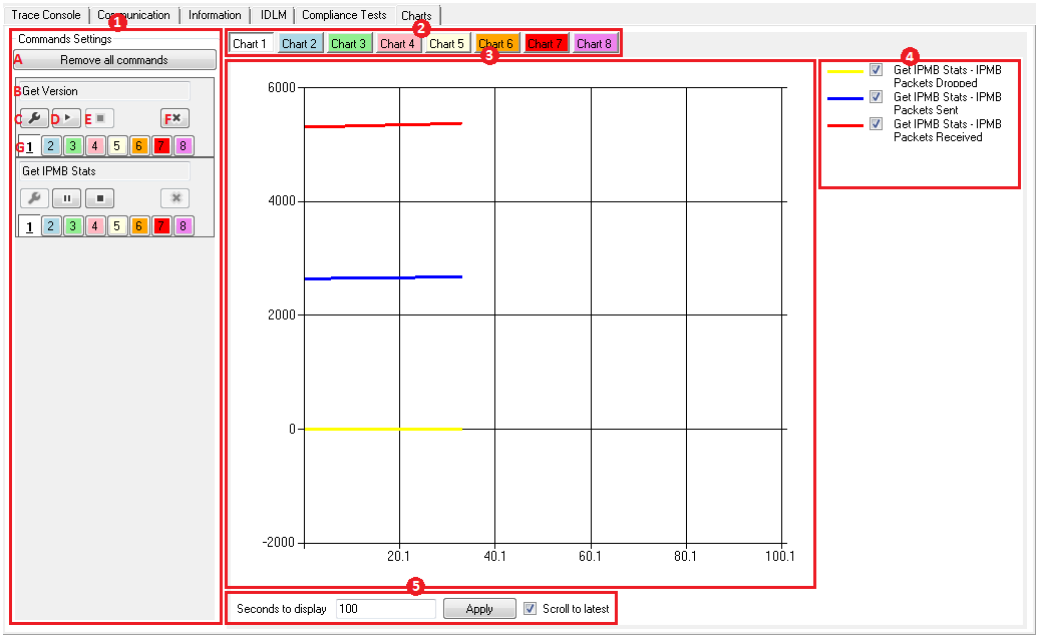
Name	Value [hex]	Bytes
Domain ID	00	1
Control Knob	02	1
Threads Enabled	0000	2

Name	Value [hex]	Bytes
Domain ID	00	1
Control Knob	1	1
P-State	00	1
T-State	00	1

Data grid will refresh dynamicly during value typing.

5.5.3 Charts

In this module user can see how request commands' response values are changing with time.



MESDC charts tab contains 5 main areas:

Commands Settings - added commands from communication tab are visible here

Remove all commands - removes every command from 1. area, including started ones

Added command name - appends with appropriate ordinal number when same command is already added

Command Settings - opens window for customizing desired command drawing

Start Drawing/Pause Sending Command - when command drawing is started, buttons C and F are nonclickable. When paused, button C is nonclickable. To be able to use C button again, user has to use button E before.

Clear Command Data - command had to be started or paused before being able to use this button. Clears command drawings from every chart

Remove Command - removes command from 1. area. When command was stared then E button has to be used before removing

Charts Numbers - select on which chart command drawing will be visible. More than one chart can be selected

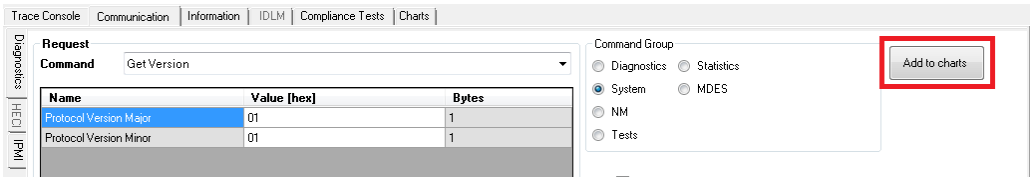
Charts Drawing Selection - select which chart will be visible in 3. area

Selected Chart Drawing - all commands drawings, if same number was selected from 1.G, are visible here

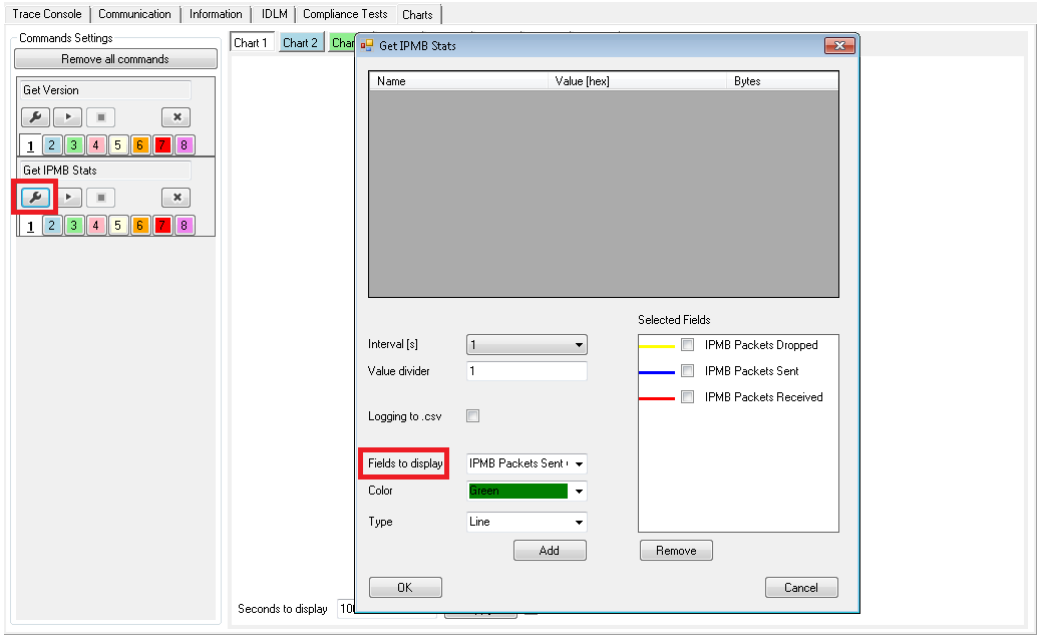
Legend - each command's fields to display chosen from 1.C window are visible here. Unchecking checkbox hides drawing of field with no command data clearing

Chart Scale - changes chart's scale, gives possibility to look in drawing history after unchecking Scroll to latest button

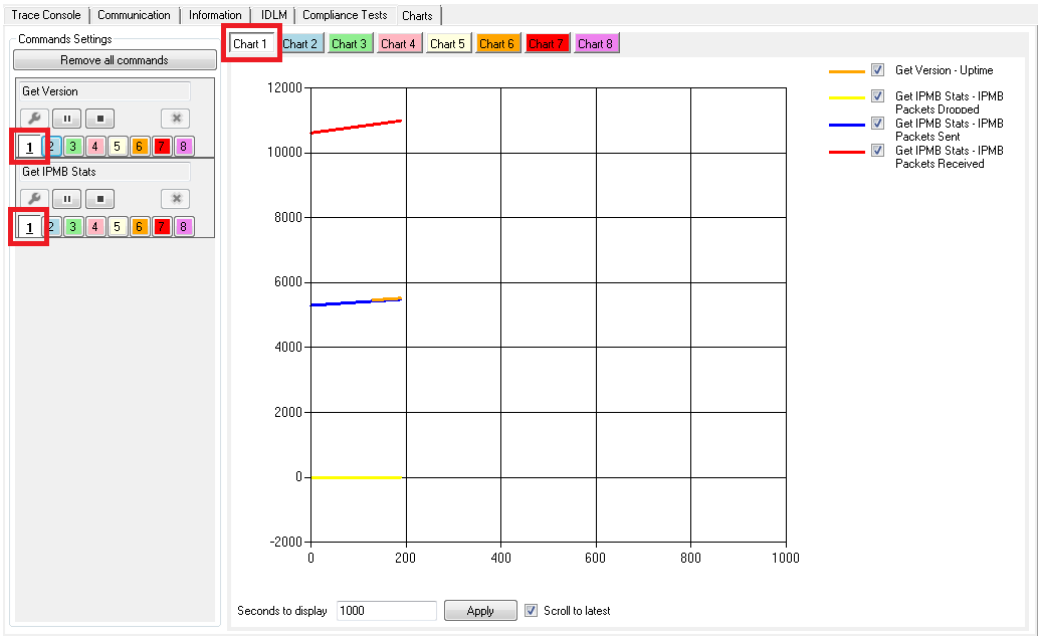
User has to use communication page in order to select and add commands to charts. Chosen request commands must have respone data fields which will be presented on charts tab.



After adding commands to charts, they are shown on left side in commands settings area. To be able to run drawing, user needs to edit each command settings to choose desired response data fields to display. Drawing time interval, value divider, color, type and logging to .csv can also be changed from that window. When logging is active, a file with time and value pairs will be created for each selected field to display.



Last thing to do is to choose desired chart numbers on which informations will be presented, and press start drawing button for each command. On the right side of chart a legend area shows up. It gives possibility to hide/unhide drawing of fields for each chart. Under the chart user can change amount of seconds to display and after that time a scrool bar appears which gives user insight in chart history after unchecking scroll to latest checkbox.



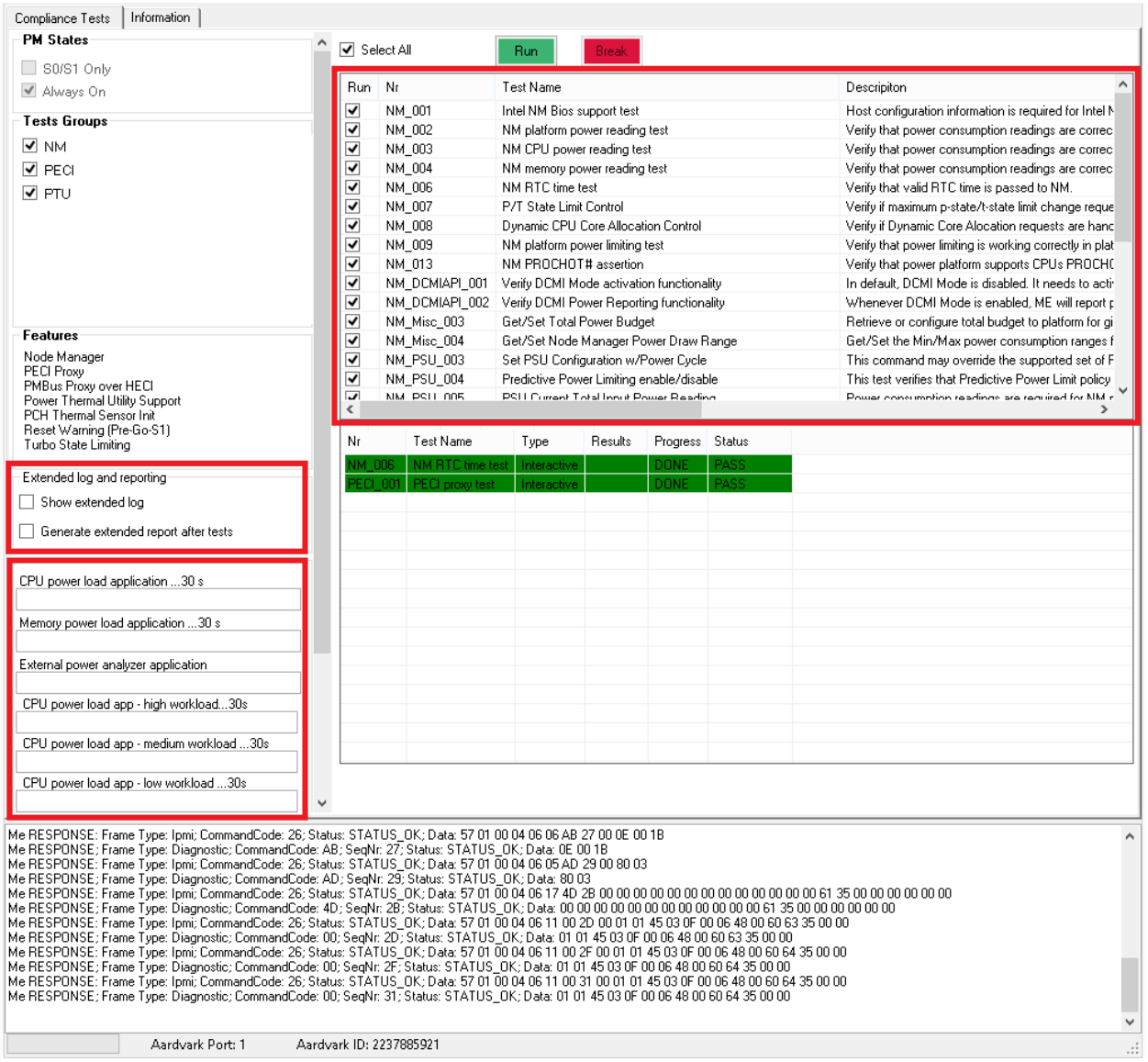
There is a possibility to add as many and also same commands to charts as needed. In that case their names would be appended with appropriate ordinal number.

5.5.4 Compliance Tests

This tab is design for customer to perform compliance tests. List of available compliance tests depends on Platform Type and enabled FW features. It is for user convenience to make some of the tests into tests Groups as BIOS, FW status and power States.

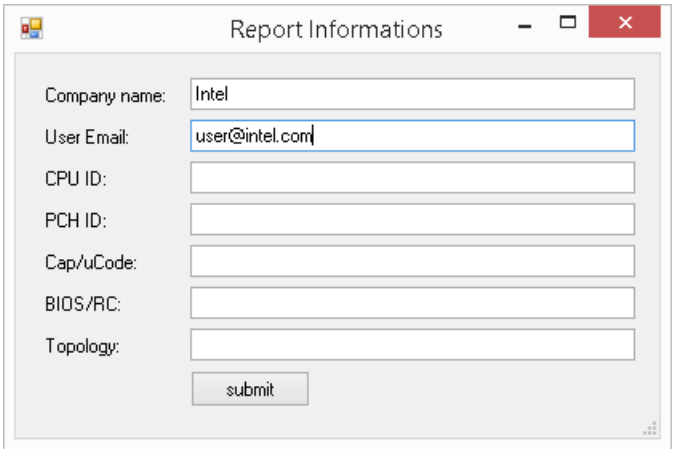
The test result and detail log is available in the same directory as MESDC named as report-XXXX and log-XXXX.

List of available tests is presented on the right side of the window. Results are listed beneath the test list after running and completing all selected tests. For more detailed information about every compliance test, please refer to the specific sections in this manual (Intel ME FW Compliance Tests).



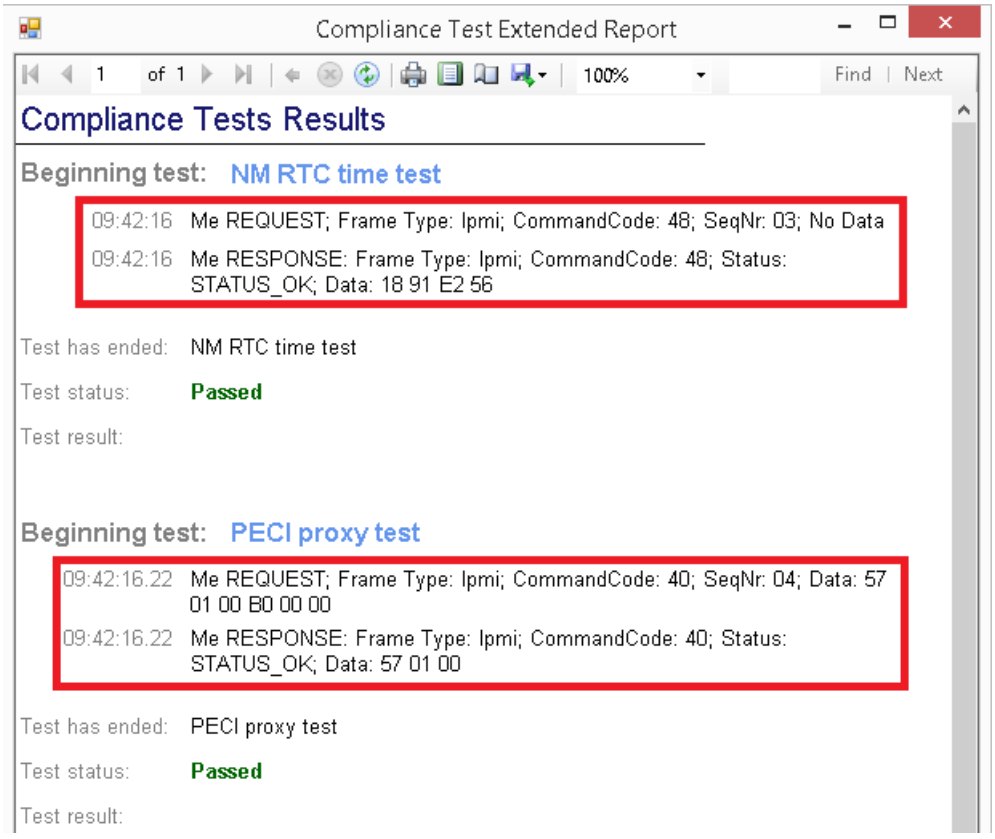
Extended log reporting

After completing all selected tests MESDC can generate a report that can be printed or exported to PDF, Microsoft Word or Microsoft Excel. To enable this feature select "Generate extended report after tests" option. The report contains status of each executed test, list of features that are enabled on the server platform and some additional information provided by user. This information is collected after test execution.



All fields are optional.

The compliance test report can additionaly contain list of sent frames and responses. To enable this feature select “Show extended log” option.



External applications

Some compliance tests require additional actions that cannot be performed by MESDC. This includes:

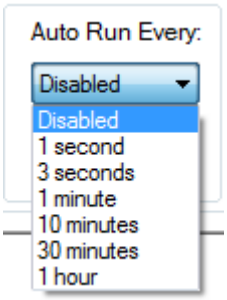
Executing various types of power load on tested system

Reading power load from external power analyzer

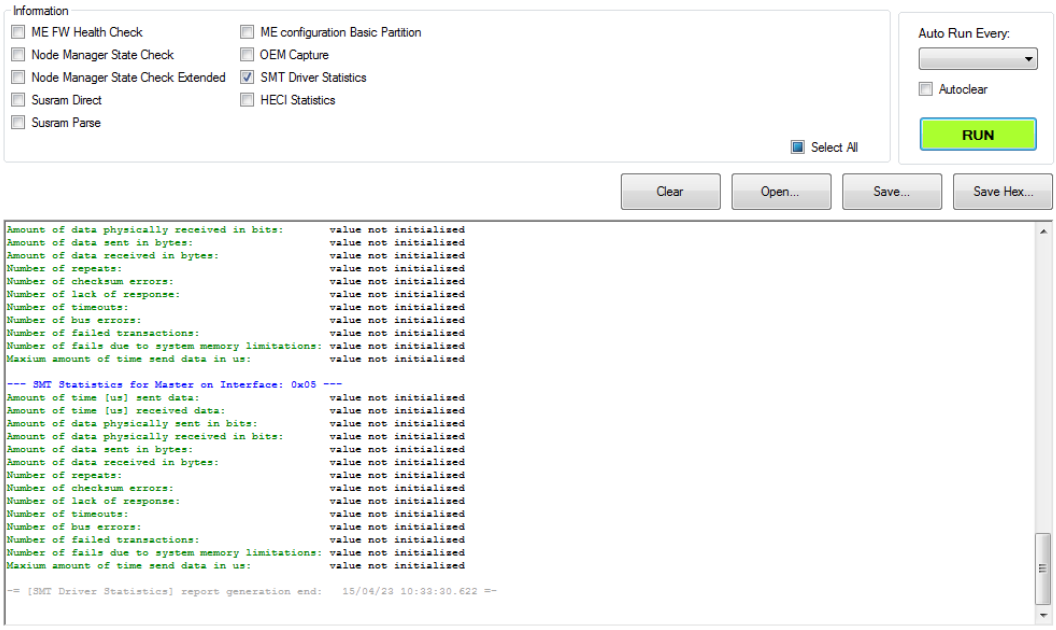
These tasks can be accomplished in two ways: manual or automatic. If the path to external application is not specified MESDC will ask user to perform specific action or provide necessary information. If the path is given MESDC will start the application and optionaly parse the application output. Manually steps with external applications will be marked special TAG in *Procedure* section – [EXTERNAL APPLICATION]

5.5.5 Information

This tab is design for customer to capture useful information from the platform. All/any of the option can be run once by clicking RUN button or run multiple times by change the Auto Run configuration. If Auto Run is enabled Information indicator on statusbar blinks green every iteration.



The result will be show in log frame with highlighted changes and can be saved by click Save button. Each section of the information will have a time stamp attached in the log.



The different lines will be highlighted for Intel ME FW Health Check, Intel Node Manager State Check, Intel Node ManagerStateCheck Extended, Susram Direct, Susram Parse, ME configuration Basic Partition, OEM Capture, SMT Driver Statistics, HECI Statistics.

Autoclear option (above the Run button) causes old reports to be deleted before new ones are shown. If autoclear option is selected then from each raport only the newest data is displayed.

ICC setting

MESDC will retrieve ICC setting exposed by Intel ME FW.

Detail reference for ICC register, please refer to Luisburg EDS. The ICC information retrieve from Pre-production silicon might be different than the ones from post production silicon.

Intel ME FW Health Check

This will retrieve Intel ME FW version, operational mode, FW self-test result, last Global Reset Cause.

Intel Node Manager State Check

This will retrieve Intel Node Manager Features information: e.g. Ptam State, Total Power Budget, Intel NM Statistics. The output of report is displayed in human-readable form.

Example of Intel Node Manager State Check report is available in Appendix C.

Intel Node Manager State Check Extended

This is the reserved data for Intel to analysis. If needed, user should capture this and send Intel for next level of analysis based on Intel guidance.

Susram Direct

MESDC will retrieve Intel ME related information which is stored in SUSRAM directory in File System and show it in HEX form.

Susram Parse

MESDC will retrieve one of SUSRAM files which contains information related to exceptions. It will also parse the data that it contains (if it is not empty) and show it in a human readable form.

OEM Capture

OEM capture will make MESDC run several MESDC commands based on OEM capture .xml file. MESDC will run the MESDC command one by one and capture the result in log. User can also run the OEM capture file multiple times with selection autorun feature. An example of xml file is as following

```
<?xml version="1.0" encoding="utf-8"?>

<OEMCapture>

  <Command Interface="SMBus" Group="System" Name="Get Version"
Arguments="0x0101" Info="Get Version Command in System group." />

</OEMCapture>
```

The interface attribute determines which set of commands is taken into consideration and must be one of: “SMBus” (commands from Diagnostic tab), “IPMI” (commands from IPMI tab), “HECI” (commands from HECI tab) or it can be not defined at all. If it is not defined then default value (“SMBus”) is used. The Command Group and Name should align with MESDC command in GUI interface. The Arguments attribute must contain all bytes for each field in the frame, e.g. if command has two fields: Manufacturer ID (0x000157) on 3 bytes and Mode (0x01) on one byte then the Arguments attribute must have value: 0x00015701.

Interface Statistics

SMT Driver Statistics

SMT driver is the driver for additional SMBUS available in Wellsburg PCH. This function will give you the statistics for SMT driver

HECI Statistics

MESDC supports collecting communication statistics for HECI interfaces (HECI-1 and HECI-2).

PECI Wire Statistics

MESDC will retrieve communication statistics for PECI

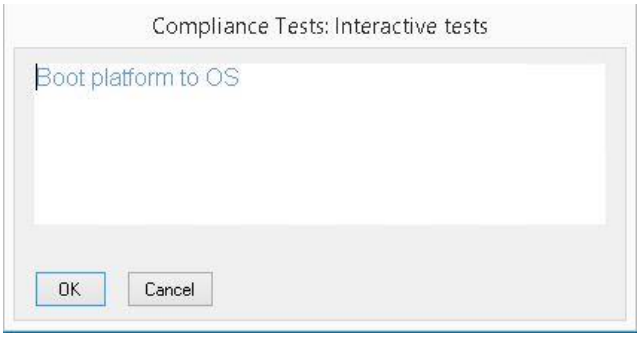
MCTP Statistics

MESDC will retrieve statistics for MCTP

5.6 Intel ME FW Compliance Tests

In most part each compliance test is automatically invoked, but some compliance tests require additional actions that cannot be performed by MESDC. Manually steps will be marked special TAG in *Procedure* section – [MANUAL]. Example communique:

> Please restart your platform
>
> Make sure that power load has ended and press OK
>
> Shutdown system and remove AC power cord on DUT
>
> Boot platform to UEFI Shell
>
> Put platform into G3 power state
>
> Boot platform to OS



5.6.1 NM_001: Verify that BIOS provides ME with host configuration information.

This set of tests is applicable to the following Intel ME firmware variants: Intel Node Manager.

Test needs RMCP+ or IPMB interface.

This test verify that BIOS provides ME with host configuration information. This information is required for NM power limiting functionality. The information is also useful for SiEn firmware in order to make it aware of the allowed CPU turbo limits

Configuration:

Test requires enable P/T-State in BIOS (CRB):

EDKII Menu -> Socket Configuration -> Advanced Power Manament Configuration -> CPU Thermal Management -> CPU T-State Control -> Software Controlled T-State (select to “enabled”)

Procedure:

[MANUAL] Boot the SUT

Verify that Host Configuration Information message is sent from BIOS

Verify that Host Configuration Information message format is compliant with ME-BIOS Specification

- Check that Host Configuration Information message length is 56 Bytes

- Check that Host Configuration Information message HECI Header (Bytes 0:3) equals 0x80340007

- Check that Host Configuration Information message MKHI Header (Bytes 4:7) equals 0x00000011

Verify that Host Configuration Information message data is correct

- P-States Number (Byte 10) is greater than 1

- T-States Number (Byte 11) is greater than 1

- Proc Cores Number (Byte 17) is greater than 2

Send IPMI command Get Number of P/T-States and verify that reported number of P/T-States and number of logical processors on the platform are correct. Number of P/T-States should match the number of P/T-States reported by HECI Host Configuration Information message

Success Criteria:

Host Configuration Information message is sent from BIOS

- Pass if HECI Host Configuration Information message is received on ME FW side

- Fail otherwise

Host Configuration Information message format is compliant with ME-BIOS Specification

- Pass if:

Host Configuration Information message length is 56 Bytes

Host Configuration Information message HECI Header (Bytes 0:3) equals 0x80340007

Host Configuration Information message MKHI Header (Bytes 4:7) equals 0x00000011

- Fail otherwise

Host Configuration Information message data is correct

- Pass if:

P-States Number (Byte 10) is greater than 1

T-States Number (Byte 11) is greater than 1

Proc Cores Number (Byte 17) is greater than 2

Number of P/T-States and number of logical processors on the platform reported with IPMI command “Get Number of P/T-States” are correct.

- Pass if

P-States Number reported by Get Number of P/T-States command (Byte 5) equals to P-States Number reported by HECI Host Configuration Information message (Byte 10)

T-States Number reported by Get Number of P/T-States command (Byte 6) equals to T-States Number reported by HECI Host Configuration Information message (Byte 11)

Number of logical processors reported by Get Number of P/T-States command (Bytes 5:6) equals to Proc Cores Number reported by HECI Host Configuration Information message (Byte 17)

- Fail otherwise

5.6.2 NM_002: NM platform power reading test

This set of tests is applicable to the following Intel ME firmware variants: Intel Node Manager.

Test needs RMCP+ or IPMB interface.

This test verify that power consumption readings are correct in platform power domain. Correct power consumption power readings need to be provided for each defined power domain.

Test also requires an application (.exe) or script (.bat) that will start memory and CPU load on DUT on 100% (load must last for at least 2 min) and another one to read from external power meter. In case when external applications were not provided, popup window will be displayed with information that user have to run them manually.

Procedure:

[MANUAL] Boot the SUT

With IPMI command "Get NM Statistics" get global power statistics for platform power domain.

Check with external power meter platform power consumption is matching the current value (Byte 5:6) reported by global power statistics.

[EXTERNAL APPLICATION] Run load on host system

With IPMI command "Get NM Statistics" get global power statistics for platform power domain.

Check with external power meter platform power consumption is matching the current value (Byte 5:6) reported by global power statistics.

Success Criteria:

Check that platform power consumption readings reported with global power statistics current value (Byte 5:6) are matching platform power consumption measured with external power meter on idle and loaded host system.

Pass if:

Global power statistics current value (Byte 5:6) reported for platform domain are matching platform power consumption measured with external power meter on idle and loaded host system with 5% tolerance

Fail otherwise

5.6.3 NM_003: NM CPU power reading test

This set of tests is applicable to the following Intel ME firmware variants: Intel Node Manager.

Test needs RMCP+ or IPMB interface.

This test verify that power consumption readings are correct in CPU power domain.

Test also requires an application (.exe) or script (.bat) that will start CPU load on DUT on 100% (load must last for at least 2 min). In case when external application was not provided, popup window will be displayed with information that user have to run them manually.

Procedure:

[MANUAL] Boot the SUT

With IPMI command "Get NM Capabilities" get CPU domain power range Min Power (Byte 8:9) and Max Power (Byte 6:7).

With IPMI command "Get NM Statistics" get global power statistics for CPU power domain -current value (Byte 5:6).

[EXTERNAL APPLICATION] Run load CPU on host system

With IPMI command "Get NM Statistics" get global power statistics for CPU power domain - current value (Byte 5:6).

Success Criteria:

Check that power statistics reported for CPU power domain - current value (Byte 5:6) are correct on idle and loaded host system:

Pass if:

IPMI command "Get NM Statistics" global power statistics for CPU power domain - current value (Byte 5:6) is greater than zero and lower than Max Power (Byte 6:7) reported by IPMI command "Get NM Capabilities" for CPU domain.

Fail otherwise

5.6.4 NM_004: NM memory power reading test

This set of tests is applicable to the following Intel ME firmware variants: Intel Node manager.

Test needs RMCP+ or IPMB interface.

This test verify that power consumption readings are correct in Memory power domain.

Test also requires an application (.exe) or script (.bat) that will start memory load on DUT on 100% (load must last for at least 2 min). In case when external application was not provided, popup window will be displayed with information that user have to run them manually.

Procedure:

[MANUAL] Boot the SUT

With IPMI command "Get NM Capabilities" get Memory domain power range Min Power (Byte 8:9) and Max Power (Byte 6:7).

With IPMI command "Get NM Statistics" get global power statistics for Memory power domain - current value (Byte 5:6).

[EXTERNAL APPLICATION] Run load memory on host system

With IPMI command "Get NM Statistics" get global power statistics for Memory power domain - current value (Byte 5:6).

Success Criteria:

Check that power statistics reported for Memory power domain - current value (Byte 5:6) are correct on idle and loaded host system:

Pass if:

IPMI command "Get NM Statistics" global power statistics for Memory power domain - current value (Byte 5:6) is greater than zero and lower than Max Power (Byte 6:7) reported by IPMI command "Get NM Capabilities" for Memory domain.

Fail otherwise

5.6.5 NM_006: NM RTC time test

This set of tests is applicable to the following Intel ME firmware variants: Intel Node manager.

Test needs RMCP+ or IPMB interface.

This test verify that valid RTC time is passed to NM. NM suspend periods are activated only if NM receives valid RTC time.

Procedure:

[MANUAL] Boot the SUT

Get internal Intel NM clock value with IPMI command “Get SEL Time”.

Check that reported time value is valid

Success Criteria:

NM reports valid time value

Pass if:

IPMI command “Get SEL Time” response Present Timestamp value (Bytes 2:5) is different from 0xFFFFFFFF.

Fail otherwise

5.6.6 NM_007: P/T State Limit Control

This set of tests is applicable to the following Intel ME firmware variants: Intel Node manager.

This test verifies BIOS support for setting P-state and T-state limit. It is done automatically by MESDC console without any user interaction.

Note that this test does not check whether the operating system respects the P-state and T-state limit set by ME. This test only checks whether
NM notifications reach OSPM and are properly handled in ACPI tables provided by BIOS.

Configuration:

Test requires enable P/T-State in BIOS:

EDKII Menu -> Socket Configuration -> Advanced Power Manament Configuration -> CPU Thermal Management -> CPU T-State Control -> Software Controlled T-State (switch to “enabled”)

Procedure:

[MANUAL] Boot the SUT

Set P-State/T-State limit with IPMI command "Set Max Allowed CPU P-State/T-State".

Verify that P-State/T-State limit change request is correctly handled by OSPM.

Check that set P-State/T-State limit is passed to CPU with IPMI command "Get Max Allowed CPU P-State/T-State".

Success Criteria:

P-State/T-State limit change requests are correctly handled by OSPM.

Pass if:

P-State/T-State limit change requests is correctly acknowledged by OSPM on HECI interface

Fail otherwise:

P-State/T-State limit is correctly passed to CPU.

Pass if:

IPMI command "Get Max Allowed CPU P-State/T-State" reports P-State (Byte 5) and T-State (Byte 6) values the same as set with IPMI command "Set Max Allowed CPU P-State/T-State" P-State (Byte 5) and T-State (Byte 6).

Fail otherwise

5.6.7 NM_008: Dynamic CPU Core Allocation Control

This set of tests is applicable to the following Intel ME firmware variants: Intel Node manager.

This test verifies if OSPM is responsible for CPU core allocation control. It is done automatically by MESDC console without any user interaction.This test verify that power limiting is working correctly in platform power domain.

Note that this test does not check whether the operating system supports dynamic changes of the number of CPU cores running. Additional OSPM is not required to fulfill the request for number of CPU cores to be allocated. This test only checks whether NM notifications reach OSPM via ACPI tables provided by BIOS.

Procedure:

[MANUAL] Boot the SUT

Set number of CPU cores to be allocated with IPMI command "Set Max Allowed CPU P-State/T-State" (Byte 5:6).

Verify that CPU core allocation change request is correctly handled by OSPM.

Success Criteria:

CPU core allocation change requests are correctly handled by OSPM.

Pass if:

CPU core allocation change requests is correctly acknowledged by OSPM on HECI interface

Fail otherwise

5.6.8 NM_009: NM platform power limiting test

This set of tests is applicable to the following Intel ME firmware variants: Intel Node Manager.

Test needs RMCP+ or IPMB interface.

This test verify that power limiting is working correctly in platform power domain.

Test also requires an application (.exe) or script (.bat) that will start memory and CPU load on DUT on 100% (load must last for at least 2 min). In case when external application was not provided, popup window will be displayed with information that user have to run them manually.

Procedure:

[MANUAL] Boot the SUT

[EXTERNAL APPLICATION] Run load on host system

With IPMI command "Get NM Statistics" get global power statistics for platform power domain - current value (Byte 5:6).

With IPMI command "Set NM Policy" set NM policy for platform power domain with power limit set to 80% of the power statistics current value collected in previous step.

Wait for set NM policy correction time.

With IPMI command "Get NM Statistics" get global power statistics for platform power domain - current value (Byte 5:6) and verify if it is matching set NM policy power limit with 5% tolerance.

Success Criteria:

NM power limiting is working correctly in platform power domain.

Pass if:

After setting NM policy power consumption in platform power domain is equal to set NM policy power limit with 5% tolerance.

Fail otherwise

5.6.9 NM_014: NM fast limiting test

This set of tests is applicable to the following Intel ME firmware variants: Intel Node Manager.

Test needs RMCP+ or IPMB interface.

This test verify that Fast NM Limiting is working correctly.

Procedure:

[MANUAL] Boot the SUT

[EXTERNAL APPLICATION] Run load on host system with OS PTU

With IPMI command "Get NM Statistics" get global power statistics for platform power domain - current value (Byte 5:6).

With IPMI command "Set Node Manager Power Draw range" set power limit for HW protection domain in range from 1W to 80% of the power statistics current value collected in previous step.

Wait for 1 sec.

With IPMI command "Get NM Statistics" get global per policy power statistics for HW protection domain and policy ID 00h - current value (Byte 5:6) and verify if it is matching set NM policy power limit with 5% tolerance. Check if policy activation state bit is set. Check if policy is actively limiting bit is set.

Success Criteria:

NM power limiting is working correctly in HW protection domain.

Pass if:

After setting NM policy power consumption in platform power domain is equal to set NM policy power limit with 5% tolerance.

Fail otherwise

5.6.10 PECI_001: Verify PECI connectivity

Test needs RMCP+ or IPMB interface.

Test also requires DMI interface connected to PCH or/and PECI wire connected to PCH.

This test verify that PECI interface is supported by the SPS FW. It could be PECI wire.

Test needs PECI Proxy feature enabled.

Procedure:

[MANUAL] Boot the SUT

Run Get CPU and Memory Temperature (4Bh) IPMI command to check if the PECI interface between CPUs and PCH works fine.

0x4B 0x57 0x01 0x00 0x01 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00

Expected response is: 0x00 0x57 0x01 0x00 0xtt,
where tt is temperature reading for CPU0

Success Criteria:

Pass – response byte 1=0x00

Fail - otherwise

5.6.11 PECI_003: Verify PECI proxy through wire functionality

Test needs RMCP+ or IPMB interface.

Test also requires PECI wire connected to PCH.

This test verify that PECI proxy interface is provided by SPS FW when PECI wire is connected to PCH.

Test needs PECI Proxy feature enabled.

Procedure:

[MANUAL] Boot the SUT

Run PECI Ping command using IPMI cmd to verify if PECI Proxy communication is available.

0x40 0x57 0x01 0x00 0xB0 0x00 0x00

Success Criteria:

Pass – response byte 1=0x00

Fail - otherwise

5.6.12 NM_PTU_001: NM PTU Manufacturer and BIOS Opt-in test

This set of tests is applicable to the following Intel ME firmware variants: Intel Node manager.

Test needs RMCP+ or IPMB or HECI interface.

Test verifies NM PTU is manufacturer opted-in in the flash image built by FITC and BIOS opted-in for BIOS support.

Test needs Power Thermal Utility Support feature enabled.

Configuration:

Test requires changes in BIOS (CRB):

EDKII Menu -> Platform Configuration Server ME Debug Configuration -> NM Configuration ->

- Power Measurement Ovveride (selected)

- Power Measurment (enabled)

- Hardware Change Override (selected)

- Hardware Changed (Yes)

- PTU Load Override (selected)

Test requires additional settings via Intel® spsFITC which are listed in Table 4-4.

Table 4-4. FITc settings PTU

FITc tree view nodes localization	FITc parameter	Value
under Configuration -> PTU -> PTU Manuf Optin	**Optin**	Opt In – Option ROM always enabled

Procedure:

[MANUAL] Boot the SUT

Send NM PTU Launchability State command

Success Criteria:

In Response (MESDC), “PTU State” value

bit[0] = 1 Manufacturer Opt-in

bit[1] = 1 BIOS Opt-in

If bit[0] and/or bit[1] is 0, this test has failed.

5.6.13 NM_PTU_002: NM PTU Launchability test (BIOS initiated)

This set of tests is applicable to the following Intel ME firmware variants: Intel Node manager.

Test needs RMCP+ or IPMB interface.

This test is to monitor NM PTU launchability state at different phases during BIOS POST when NM PTU characterization is initiated/requested by BIOS.

Test needs Power Thermal Utility Support feature enabled.

Configuration:

Test requires changes in BIOS (CRB):

EDKII Menu -> Platform Configuration Server ME Debug Configuration -> NM Configuration ->

- Power Measurement Ovveride (selected)

- Power Measurment (enabled)

- Hardware Change Override (selected)

- Hardware Changed (Yes)

- PTU Load Override (selected)

Test requires additional settings via Intel® spsFITC which are listed in Table 4-4.

Procedure:

Send "NM PTU Launchability State" every 0.5 sec

On SUT

[MANUAL] Boot/Reboot the SUT

NM PTU characterization will be initiated by BIOS upon the very first-time platform boot or detected hardware change

Success Criteria:

In Response (MESDC), "PTU State" value will go through the following sequence: 01 -> 2B (or 2F) -> 23 ->03

If sequence "2B -> 23" or "2F -> 23" is not seen, NM PTU is not launched by BIOS and this test has failed.

Note: sequence "2F -> 23" will be seen if BIOS forces NM PTU characterization at every reboot or BIOS detected hardware change and required NM PTU characterization at reboot after BMC sends 0x60 command.

5.6.14 NM_PTU_003: NM PTU Launchability test (BMC initiated)

This set of tests is applicable to the following Intel ME firmware variants: Intel Node manager.

Test needs RMCP+ or IPMB interface.

This test is to monitor NM PTU launchability state at different phases during BIOS POST when NM PTU characterization is initiated/requested by BMC.

Test needs Power Thermal Utility Support feature enabled.

Configuration:

Test requires changes in BIOS (CRB):

EDKII Menu -> Platform Configuration Server ME Debug Configuration -> NM Configuration ->

- Power Measurement Ovveride (selected)

- Power Measurment (enabled)

- Hardware Change Override (selected)

- Hardware Changed (Yes)

- PTU Load Override (selected)

Test requires additional settings via Intel® spsFITC which are listed in Table 4-4.

Procedure:

Issue IPMI command to Intel ME (either via ipmitool or MESDC, on SUT or debug host)

NetFn: 0x2e

Command: 0x60

Data: 0x57 0x01 0x00 0x01

Send "NM PTU Launchability State" every 0.5 sec

On SUT

[MANUAL] Boot/Reboot the SUT

NM PTU characterization will be initiated by BIOS upon the very first-time platform boot or detected hardware change

Success Criteria:

In Response (MESDC), "PTU State" value will go through the following sequence: 01 -> 27 (or 2F) -> 23 ->03

If sequence "27 -> 23" or "2F -> 23" is not seen, NM PTU is not launched by BIOS and this test has failed.

Note: sequence "2F -> 23" will be seen if BMC happened to send 0x60 command as well before SUT reboot.

5.6.15 NM_PTU_004: NM PTU reporting platform domain characterization test

This set of tests is applicable to the following Intel ME firmware variants: Intel Node manager.

This test is to verify NM PTU reporting functionality is operational

Test needs RMCP+ or IPMB interface.

Test needs Power Thermal Utility Support feature enabled.

Configuration:

Test requires changes in BIOS (CRB):

EDKII Menu -> Platform Configuration Server ME Debug Configuration -> NM Configuration ->

- Power Measurement Ovveride (selected)

- Power Measurment (enabled)

- Hardware Change Override (selected)

- Hardware Changed (Yes)

- PTU Load Override (selected)

Test requires additional settings via Intel® spsFITC which are listed in Table 4-4.

Procedure:

[MANUAL] Boot the SUT

Issue IPMI command to Intel ME

NetFn: 0x2e

Command: 0x61

Data: 0x57 0x01 0x00 0x00

Success Criteria:

Examine response bytes

Byte	Description
1	Completion Code =0x00
2:4	Intel IANA =0x57 0x01 0x00
5:8	IPMI Specification-based timestamp when the characterization data was collected
9:10	Platform Maximum Power Draw in Watts (unsigned integer)

11:12	Platform Minimum Power Draw in Watts (unsigned integer)
13:14	Platform Efficient Power Draw in Watts (unsigned integer)

Additional check on power draw values: CPU Maximum > CPU Efficient >= CPU Minimum, and all values are non-zeros

5.6.16 NM_PTU_005: NM PTU reporting CPU domain characterization test

This set of tests is applicable to the following Intel ME firmware variants: Intel Node manager.

Test needs RMCP+ or IPMB interface

This test is to verify NM PTU reporting functionality is operational.

Test needs Power Thermal Utility Support feature enabled.

Configuration:

Test requires changes in BIOS (CRB):

EDKII Menu -> Platform Configuration Server ME Debug Configuration -> NM Configuration ->

- Power Measurement Ovveride (selected)

- Power Measurment (enabled)

- Hardware Change Override (selected)

- Hardware Changed (Yes)

- PTU Load Override (selected)

Test requires additional settings via Intel® spsFITC which are listed in Table 4-4.

Procedure:

[MANUAL] Boot the SUT

Issue IPMI command to Intel ME

NetFn: 0x2e

Command: 0x61

Data: 0x57 0x01 0x00 0x01

Success Criteria:

Examine response bytes

Byte	Description
1	Completion Code =0x00
2:4	Intel IANA =0x57 0x01 0x00
5:8	IPMI Specification-based timestamp when the characterization data was collected
9:10	CPU Maximum Power Draw in Watts (unsigned integer)
11:12	CPU Minimum Power Draw in Watts (unsigned integer)
13:14	CPU Efficient Power Draw in Watts (unsigned integer)

Additional check on power draw values: CPU Maximum > CPU Efficient >= CPU Minimum, and all values are non-zeros

5.6.17 NM_PTU_006: NM PTU reporting memory domain characterization test

This set of tests is applicable to the following Intel ME firmware variants: Intel Node manager.

Test needs RMCP+ or IPMB interface.

This test is to verify NM PTU reporting functionality is operational.

Test needs Power Thermal Utility Support feature enabled.

Configuration:

Test requires changes in BIOS (CRB):

EDKII Menu -> Platform Configuration Server ME Debug Configuration -> NM Configuration ->

- Power Measurement Ovveride (selected)

- Power Measurment (enabled)

- Hardware Change Override (selected)

- Hardware Changed (Yes)

- PTU Load Override (selected)

Test requires additional settings via Intel® spsFITC which are listed in Table 4-4.

Procedure:

[MANUAL] Boot the SUT

Issue IPMI command to Intel ME

NetFn: 0x2e

Command: 0x61

Data: 0x57 0x01 0x00 0x02

Success Criteria:

Examine response bytes

Byte	Description
1	Completion Code =0x00
2:4	Intel IANA =0x57 0x01 0x00
5:8	IPMI Specification-based timestamp when the characterization data was collected
9:10	Memory Maximum Power Draw in Watts (unsigned integer)
11:12	Memory Minimum Power Draw in Watts (unsigned integer)
13:14	0x00

Additional check on power draw values: Memory Maximum > Memory Minimum

5.6.18 NM_PTU_007: NM PTU start on reset test

This set of tests is applicable to the following Intel ME firmware variants: Intel Node manager.

Test needs RMCP+ or IPMB interface

This test is to verify NM PTU is successfully launched on platform reset.

Test needs Power Thermal Utility Support feature enabled.

Configuration:

Test requires changes in BIOS (CRB):

EDKII Menu -> Platform Configuration Server ME Debug Configuration -> NM Configuration ->

- Power Measurement Ovveride (selected)

- Power Measurment (enabled)

- Hardware Change Override (selected)

- Hardware Changed (Yes)

- PTU Load Override (selected)

Test requires additional settings via Intel® spsFITC which are listed in Table 4-4.

Procedure:

[MANUAL] Boot the SUT

Target system running in S0 with active OS

Issue IPMI command to Intel ME

NetFn: 0x2e

Command: 0x60

Data: 0x57 0x01 0x00 0x01

Examine response bytes: 0x00 0x57 0x01 0x00

[MANUAL] Shutdown and restart OS on target system

Wait until target system is running in S0 and OS is active

Issue IPMI command to Intel ME

NetFn: 0x2e

Command: 0x61

Data: 0x57 0x01 0x00 0x00

Success Criteria:

Examine response bytes 5:8 reflects proper time when target system was restarted

Byte	Description
1	Completion Code =0x00
2:4	Intel IANA =0x57 0x01 0x00
5:8	IPMI Specification-based timestamp when the characterization data was collected
9:10	Platform Maximum Power Draw in Watts (unsigned integer)
11:12	Platform Minimum Power Draw in Watts (unsigned integer)
13:14	Platform Efficient Power Draw in Watts (unsigned integer)

Additional check on power draw values: Platform Maximum > Platform Efficient >= Platform Minimum, and all values are non-zeros

5.6.19 ME_Power_States_001: ME power state after shutdown

Test needs RMCP+ or IPMB interface, OS (to initiate S5 state – shutdown from OS).

Test will report results depending on ME Power state configuration (Powered in all SX states or S0 Only.

Procedure:

Use MESDC tool to check communication with ME and ME state.

Send GetDeviceID (IPMB over SMLink0) - IPMICmd <bridging info> 0x2e 0x00 0xe1 0x02 0x18 0x01,

Send GetVersion (Diagnostic over SMBus) –

[MANUAL] Issue shutdown from OS (S5 state).

Wait until platform will finish the transition from S0 to S5 state.

Use Aardvark and MESDC tool to check communication with ME.

Send GetDeviceID (IPMB over SMLink0) - IPMICmd <bridging info> 0x2e 0x00 0xe1 0x02 0x18 0x01,

Send GetVersion (Diagnostic over SMBus),

[MANUAL] Wake up the platform using power button.

Wait until platform will finish the transition from S5 to S0 (boot to OS).

Use Aardvark and MESDC tool to check communication with ME.

Send GetDeviceID (IPMB over SMLink0) - IPMICmd <bridging info> 0x2e 0x00 0xe1 0x02 0x18 0x01,

Send GetVersion (Diagnostic over SMBus).

Success Criteria:

ME behavior in Powered in all SX states mode will be as follows:

In step 1) and 7),

ME for GetDevice ID should respond in byte 16, 01b or 10b (operational state).

ME for GetVersion should respond in byte 9, 45h and byte 10, 02h (M0 with UMA state)

In step 4),

ME for GetDevice ID should respond in byte 16, 01b or 10b (operational state).

ME for GetVersion should respond in byte 9, 05h and byte 10, 03h (M3 without UMA).

ME behavior in S0 Only power mode:

In step 1) and 7), ME should behave in the same way is in Powered in all SX state power mode.

In step 4), ME should not respond at all

5.6.20 PMBUS_Proxy_001: PSUs compliance with the PMBus specification.

Test needs RMCP+ or IPMB interface, PSUs supporting PMBus specification.

Test verifies whether PSUs and ME are supporting PMBus specification.

Test needs PMBus Proxy over HECI feature enabled.

Procedure:

[MANUAL] Boot the SUT

SendRAW PMBUS command (D9h). See details of this command in the SPS 5.0 External Interface Specification.

Success Criteria:

PSUs should respond for Send RAW PMBus command with completion code 00h.

5.6.21 UMA_001: Loading UMA after Power ON

Test needs SMBus or RMCP+ or IPMB interface, DDR4 installed on board, capability to AC cycle (ON/OFF) platform, capability to monitor DID message from the BIOS to ME .

Test verifies UMA state at each stage of platform boot up process.

Procedure:

[MANUAL] Put platform into G3 power state (AC OFF).

[MANUAL] Transit platform from G3 to G0 (AC ON).

After 3 sec from AC ON, send GetVersion command. GetVersion should respond with:

in byte 9, 45h

in byte 10, 01h or 03h (01h – M0 without UMA Init not yet done, 03h – M0 without UMA)

[MANUAL] Wait for DID message.

After DID, send GetVersion command. GetVersion should respond with:

in byte 9, 45h

in byte 10, 02h (M0 with UMA state – normal state for Intel Server Platform Services firmware with UMA support for Purley platform).

Success Criteria:

ME should respond as it is described in the test procedure section. Response is described in SPS 5.0 ME-BIOS Interface.

5.6.22 ME_Reset_001: Host Cold Reset

Test needs RMCP+ or IPMB interface, UEFI Shell.

Test verifies that platform and ME FW is capable to perform Host Reset with power cycle without any unexpected behaviors.

Procedure:

[MANUAL] Boot to UEFI shell.

Send GetDeviceID (IPMB over SMLink0) - IPMICmd <bridging info> 0x2e 0x00 0xe1 0x02 0x18 0x01,

ME should respond in byte 16, 01b or 10b (operational state). Completion Code should be 00h.

[MANUAL] Issue Cold Reset of HOST (HOST reset with power cycle) by sending "reset".

Check/observe whether platform actually performs Cold Reset.

Boot to EFI.

Send GetDeviceID (IPMB over SMLink0) - IPMICmd <bridging info> 0x2e 0x00 0xe1 0x02 0x18 0x01,

ME should respond in byte 16, 01b or 10b (operational state). Completion Code should be 00h.

Success Criteria:

ME behavior in Powered in all SX states mode should be as follows:

Platform should successfully perform reset. ME should not reset during this flow (should not send any OEM power state change notifications on IPMB).

ME behavior in S0 only power mode should be as follows:

Platform should successfully perform reset. ME should reset during this flow (should send OEM power state change notification to BMC twice. First will be "e3h 02h" and while booting "e3h 00h".

5.6.23 ME_Reset_002: Host Warm Reset

Test needs RMCP+ or IPMB interface, UEFI Shell.

Test verifies that platform and ME FW is capable to perform Host Reset without power cycle without any unexpected behaviors.

Procedure:

[MANUAL] Boot to UEFI shell.

Send GetDeviceID (IPMB over SMLink0) - IPMICmd <bridging info> 0x2e 0x00 0xe1 0x02 0x18 0x01,

ME should respond in byte 16, 01b or 10b (operational state). Completion Code should be 00h.

[MANUAL] Issue Cold Reset of HOST (HOST reset with power cycle) by sending "reset -w".

Check/observe whether platform actually performs Cold Reset.

Boot to EFI.

Send GetDeviceID (IPMB over SMLink0) - IPMICmd <bridging info> 0x2e 0x00 0xe1 0x02 0x18 0x01,

ME should respond in byte 16, 01b or 10b (operational state). Completion Code should be 00h.

Success Criteria:

ME behavior in Powered in all SX states and S0 only state should be as follows:

Platform should successfully perform reset. ME should not reset during this flow (should not send any OEM power state change notifications on IPMB).

5.6.24 ME_Reset_003: ME Cold Reset

Test needs RMCP+ or IPMB interface.

Test initiates ME Cold Reset by IPMI command and is checking the results.

Procedure:

[MANUAL] Boot platform to S0 state.

Send GetDeviceID (IPMB over SMLink0) - IPMICmd <bridging info> 0x2e 0x00 0xe1 0x02 0x18 0x01,

ME should respond in byte 16, 01b or 10b (operational state). Completion Code should be 00h.

Issue ME Cold Reset IPMI command (02h).

Wait for 5 seconds.

Send GetDeviceID (IPMB over SMLink0) - IPMICmd <bridging info> 0x2e 0x00 0xe1 0x02 0x18 0x01,

ME should respond in byte 16, 01b or 10b (operational state). Completion Code should be 00h.

Success Criteria:

ME behavior in Powered in all SX states and S0 only state should be as follows:

ME FW should successfully perform reset. ME should reset during this flow (should send OEM power state change notification to BMC twice. First will be "e3h 02h" and while booting "e3h 00h".

Only ME FW should reset during ME Cold Reset, Host (platform) should not be affected by ME FW Cold Reset.

5.6.25 IPMI_001: IPMI communication verification using the simple IPMI command.

Test needs RMCP+ or IPMB interface.

Test verifies whether Intel ME FW and BMC can successfully communicate.

Procedure:

[MANUAL] Boot platform to S0 power state.

Send GetDeviceID. Send GetDeviceID (IPMB over SMLink0) - IPMICmd <bridging info> 0x2e 0x00 0xe1 0x02 0x18 0x01

Success Criteria:

ME should respond with Completion Code 00h.

ME for GetDeviceID should respond in byte 16, 01b or 10b (operational state).

5.6.26 PTT_001: UMA Verification Test

Test needs SMBus or HECI interface.

Bios allocates UMA region during boot process and this test is to ensure that UMA is set up correctly and Node Manager FW can access UMA for PTT functionality.

Test needs Platform Trusted Technology (PTT) feature enabled.

Configuration:

Test requires additional settings via Intel® spsFITC which are listed in Table 4-5.

Table 4-5. FITc settings PTT

FITc tree view nodes localization	FITc parameter	Value
under Configuration -> Platform Security -> Intel Ptt Configuration	**Intel(R) PTT initial power-up state** **Intel(R) PTT Supported** **PTT Secure Keys Management**	All to Enabled

Procedure:

Build and burn image with PTT Enabled.

[MANUAL] Power on the platform and boot to OS.

Send Diagnostic command GetVersion

Verify that command response is successfull

From command response get FWStatus

Verify that bits 8:6 of FW Status are equal 001b – which corresponds to M0 with UMA – normal state for SPS ME FW on Purley platform.

Success Criteria:

Response is successful

FW Status Bits[8:6] == 001b

5.6.27 PTT_002: OS/PTT Communication Test

Test needs RMCP+ or IPMB interface.

This test verifies that the communication channel between PTT and OS is set up correctly.

Test needs Platform Trusted Technology (PTT) feature enabled and Microsoft Diagnostic Console (tpm.msc).

Configuration:

Test requires additional settings via Intel® spsFITC which are listed in Table 4-5.

Procedure:

Build and burn image with PTT Enabled.

[MANUAL] Power on the platform boot to OS.

Run "Get Device ID' IPMI cmd to read the SPS FW version

[MANUAL] Run Microsoft diagnostic console 'tpm.msc'

[MANUAL] Verify that TPM manufacturer is INTC

[MANUAL] Verify that TPM version equals SPS FW version.

[MANUAL] Verify that TPM status is Ready (May required 'Preparing the TPM' from TPM console. Windows need to take ownership over TPM to state that it's ready for use.).

Success Criteria:

SPS FW version from the Get Device Id equals the PTT version from tpm.msc

TPM status from tpm.msc == ready

5.6.28 PTT_003: OOB PTT Communication Test – verify if PTT is enabled on the platform

Test needs RMCP+ or IPMB interface.

Test verifies that the SPS FW integration with PTT is successful in order to communicate OOB with PTT and that PTT is enabled on the platform.

Test needs Platform Trusted Technology (PTT) feature enabled.

Configuration:

Test requires additional settings via Intel® spsFITC which are listed in Table 4-5.

Procedure:

Build and burn image with PTT Enabled.

[MANUAL] Power on the platform boot to OS.

Run'Get PTT Capabilities'

Request – ipmitool <bridging info> raw 0x2e 0x71 0x57 0x01 0x00

Verify PTT is enabled.

Success Criteria:

Response from Get PTT Capabilities shows that PTT is enabled. Response Byte5 [7:6] equals 11

Fail otherwise

5.6.29 PTT_004: OOB PTT Communication Test – verify if PTT version is valid

Test needs RMCP+ or IPMB interface.

Test verifies that the SPS FW integration with PTT is successful in order to communicate OOB with PTT and that PTT version is the valid version.

Test needs Platform Trusted Technology (PTT) feature enabled.

Configuration:

Test requires additional settings via Intel® spsFITC which are listed in Table 4-5.

Procedure:

Build and burn image with PTT Enabled.

[MANUAL] Power on the platform boot to OS.

Run'Get PTT Version'

Request – ipmitool <bridging info> raw 0x2e 0x71 0x57 0x01 0x00

Verify that the PTT version is correct.

Success Criteria:

PTT version read back from Get PTT Version cmd is valid. Byte 5 equals 01

Fail otherwise

5.6.30 BTG_001: BTG Enable and Initialization Test

Test needs SMBus or RMCP+ or IPMB or HECI interface.

By default Boot Guard is disabled on every platform. It's up to the OEM to enable Boot Guard and program the right Boot profile. In some cases boot guard may encounter problems during boot so this test verifies if boot guard initialization was successful or not.

Test needs Boot Guard feature enabled.

Configuration:

Test requires additional settings via Intel® spsFITC which are listed in Table 4-6.

Table 4-6. FITc settings BTG

FITc tree view nodes localization	FITc parameter	Value
under Configuration -> Platform Security -> Boot Guard Configuration	**OEM Public Key Hash**	AF69F62499DF4234265A43 E9FABE6A34A3034DA4CB3F 9FADF95CA685BFE7683C
under Configuration -> Platform Security -> Boot Guard Configuration	**Key Manifest ID**	1
	Boot Guard Profile Configuration	Boot Guard Profile 4 – FVE
	CPU Debugging	False
	BSP Initialization	False

Procedure:

Build and burn image with BtG Enabled.

[MANUAL] Power on the platform boot to OS.

Read FWSTS5 and FWSTS6

Success Criteria:

FWSTS 6 bit [28] is set to 0

FWSTS 5 bit [31] is set to 0

FWSTS 5 bit [8] is set to 1

FWSTS 5 bits [7:6] are set to 0

FWSTS 5 bit [0] is set to 0

Fail otherwise

5.6.31 BTG_002: Boot Profile Verification

Test needs RMCP+ or IPMB or HECI interface

Test verifies if the Profile selected matches with the profile selected during build.

Test needs Boot Guard feature enabled.

Configuration:

Test requires additional settings via Intel® spsFITC which are listed in Table 4-6.

Procedure:

Build and burn image with BtG Enabled and profile 4 selected.

[MANUAL] Power on the platform boot to OS.

Read FWSTS5 and FWSTS6

Success Criteria:

FWSTS 6 bit [0] is set to 1

FWSTS 6 bit [3] is set to 1

FWSTS 6 bit [8] is set to 0

FWSTS 6 bits [7:6] are set to 0x03

FWSTS 1 bit [9] is set to 1

Fail otherwise

5.6.32 BTG_003: OOB BTG Communication Test

Test needs RMCP+ or IPMB interface

Test verifies the SPS FW integration with BTG is successful in order to communicate OOB with BTG and that BTG is enabled on the platform with profile 4

Test needs Boot Guard feature enabled.

Configuration:

Test requires additional settings via Intel® spsFITC which are listed in Table 4-6.

Procedure:

Build and burn image with BtG Enabled and profile 4 selected.

[MANUAL] Power on the platform boot to OS.

Run 'Get Boot Guard health'

Request – ipmitool <bridging info> raw 0x2e 0x82 0x57 0x01 0x00

Success Criteria:

Byte 12 [6] == 1

Byte 10 [0] == 0

Byte 10[1] == 1

Byte 8[5] == 1

Byte 9[5] ==1

Byte 9[7:6] == 0x3

Fail otherwise

5.6.33 FD0V_001: FD0V Enable Test

Test needs RMCP+ or IPMB or HECI interface

Test verifies that FD0V has been enabled on the platform successfully and SPS boots to operational mode after power on with FD0V enabled.

FD0V is disabled default in XML. This feature can be enabled by providing the right Public key and Manifest. Once these values are used to build an image, this test verifies if the platform boots to operational and FD0V PASSED.

GPIO can be used to verify the PASS/FAIL results on a production/non-production platform.

Test needs Flash Descriptor Region Verification (FD0V) feature enabled.

Procedure:

Build an image with proper values (OEM public key, hash of Flash Descriptor, hash of OEM public key) and FD0V Enabled

Implement/configure a GPIO for FD0V.

[MANUAL] Power on the platform and boot to OS.

Send Diagnostic command GetVersion

From command response get FWStatus

Verify the GPIO status

Success Criteria:

Response is successful

FW is in operational

FWSTS1 Bit[24] == 1

FWSTS1 Bit[3:0] == 0101b

Allocated GPIO is asserted, set to 0

Fail otherwise

5.6.34 SmaRT_001: Verify SmaRT&CLST functionality

Test needs RMCP+ or IPMB interface

CPU0 present on the platform.

#SMBAlert signal assertion method: changing PSU temperature or current thresholds using Send RAW PMBus command to the level when normal PSU operation triggers the alert.

SmaRT&CLST feature serves as a protection of PSUs against over temperature and overcurrent events and against platform shutdown during undervoltage events.

Configuration:

Test requires connected power supply to PSU (address of device must be set to 0x58).

Procedure:

[MANUAL] Power up the platform.

Read actual setting of the PSU temperature threshold that is used to generate #SMBAlert, for reference:

0xD9 0x57 0x01 0x00 0x86 0xB0 0x00 0x00 0x01 0x02 0x51
Response will look like this:
0x00 0x57 0x01 0x00 **0xA9 0xEB**, where two last bytes represent the actual temperature setting

Change PSU temperature threshold to the value below the ambient temperature (7 ^{0}C in this example) – this will cause immediate #SMBAlert assertion:
0xD9 0x57 0x01 0x00 0x88 0xB0 0x00 0x00 0x03 0x00 0x51 0x1C 0xF0
Response: 0x00 0x57 0x01 0x00

Read actual PL2 limit sent to CPU0 by ME to check if the #SMBAlert assertion caused ME set limit of 0 W to force maximum CPU throttling:
0x40 0x57 0x01 0x00 0x30 0x05 0x05 0xA1 0x00 0x1B 0x00 0x00
Expected response is as follows:
0x00 0x57 0x01 0x00 0x40 0x00 **0x80** 0x47 0x00

0x80 means limit is set to 0 W and is active

Restore the original PSU temperature threshold read in stage 2:
0xD9 0x57 0x01 0x00 0x88 0xB0 0x00 0x00 0x03 0x00 0x51 **0xA9 0xEB**.

Success Criteria:

PSU temperature threshold was successfully changed and #SMBAlert assertion occured.

Original PSU temperature threshold restored

Fail otherwise

5.6.35 MCTP_001: MCTP BO HECI Message test.

Test needs SMBus interface.

Test verifies that MCTP BO HECI Message was sent.

In MCTP Bus Owner Proxy mode, MCTP BO HECI Message should be sent to make the communication possible. In such case, MCTP BO Proxy statistics from 1 to 6 will be greater than 0.

If statistics are equal to 0, than HECI Message probably was not send.

Possible root cause of this situation is not enabled setting MCTP Bus Owner in BIOS.

HECI msg should be sent before EOP.

Test needs MCTP Infrastructure feature enabled.

Procedure:

Use Aardvark and MESDC tool to check MCTP statistics.

[MANUAL] Plug MCTP PCIe Device (endpoint)

[MANUAL] Power on the platform.

Verify statistics from 1st to 6th. If all stats are equal to 0, than MCTP BO HECI Message was not send.

Wait for EOP.

Success Criteria:

If configuration is set correctly, statistics should be greater than 0. If MCTP is set incorrectly, all statistics 1-6 should be equal to 0.

If the HECI msg will not be sent before EOP, ME will generate Health Event 0a e0 02.

Please note that this is valid for Intel ME as MCTP Bus Owner Proxy mode. MCTP Infrastructure (Stack mode) has different statistics and conditions.

5.6.36 MCTP_002: MCTP communication test.

Test needs SMBus interface.

Test will verify whether MCTP Bus Owner is initialized in Intel ME as Bus Owner Proxy mode. Will confirm that BO sends messages to Endpoints through Intel ME Proxy. In other words, full communication between BO and EP was successful.

Test needs MCTP Infrastructure feature enabled.

Procedure:

Use Aardvark and MESDC tool to check MCTP statistics.

[MANUAL] Plug MCTP PCIe Device (endpoint)

[MANUAL] Power on the platform.

Check 4th statistic.

Success Criteria:

If statistics are greater than 0, than MCTP Bus Owner Proxy mode of Intel ME works correctly.

All other settings within FITc and BIOS are set properly.

What is more, it is a confirmation that communication between BO and Endpoint is possible.

Communication Point-to-Point works.

Fail otherwise

Please note that this is valid for Intel ME as MCTP Bus Owner Proxy mode. MCTP Infrastructure (Stack mode) has different statistics and conditions.

5.6.37 MCTP_003: MCTP Bus Owner/Intel ME communication test

Test needs SMBus interface.

Test will verify whether MCTP Bus Owner is initialized in Intel ME as Bus Owner Proxy mode. Will confirm that BO sends messages. Intel ME Proxy received those messages but for some reason, messages were incorrect and were not send to MCTP Endpoint.

Test needs MCTP Infrastructure feature enabled.

Procedure:

Use Aardvark and MESDC tool to check MCTP statistics.

[MANUAL] Plug MCTP PCIe Device (endpoint)

[MANUAL] Power on the platform.

Check 5th statistic.

Success Criteria:

If statistics are greater than 0, than MCTP Bus Owner Proxy mode of Intel ME works correctly.

What is more, it is a confirmation that all other settings within FITc and BIOS are set properly.

However, messages were incorrect an Proxy were unable to pass them to MCTP Endpoints. It may be caused by many issues. For example, Endpoint is not compatible with specification or Bus Owner sent message on wrong Ednpoint ID.

Fail otherwise

Please note that this is valid for Intel ME as MCTP Bus Owner Proxy mode. MCTP Infrastructure (Stack mode) has different statistics and conditions.

5.6.38 MCTP_004: MCTP End point/Intel ME communication test

Test needs SMBus interface.

Test will verify whether MCTP Endpoint support MCTP and was initialized correctly. Test will check if Endpoint sends requests to Bus Owner.

Test needs MCTP Infrastructure feature enabled.

Procedure:

Use Aardvark and MESDC tool to check MCTP statistics.

[MANUAL] Plug MCTP PCIe Device (endpoint)

[MANUAL] Power on the platform.

Check 7th statistic.

Success Criteria:

If statistic is greater than 0, than MCTP Endpoint supports MCTP Protocol. Endpoint send messages to Bus Owner.

Fail otherwise

Please note that this is valid for Intel ME as MCTP Bus Owner Proxy mode. MCTP Infrastructure (Stack mode) has different statistics and conditions.

5.6.39 MCTP_005: MCTP Endpoint/Intel ME MCTP Proxy/Bus owner communication test

Test needs SMBus interface.

Test verifies the communication between MCTP Endpoint - Intel ME MCTP Proxy – MCTP Bus Owner

Test will verify whether MCTP Endpoint support MCTP and was initialized correctly.

Test will verify that Bus Owner was initialized/registered correctly.

Test will confirm that all settings within FITc and BIOS are correct.

Test will check if Endpoint sends requests to Bus Owner.

Test needs MCTP Infrastructure feature enabled.

Procedure:

Use Aardvark and MESDC tool to check MCTP statistics.

[MANUAL] Plug MCTP PCIe Device (endpoint)

[MANUAL] Power on the platform.

Check 9th statistic.

Success Criteria:

If statistic is greater than 0, than MCTP Endpoint supports MCTP Protocol. Endpoint send messages to Bus Owner. Intel ME received passed those messages to Bus Owner. All settings in FITc and BIOS are correct. MCTP endpoint supports MCTP.

Fail otherwise

5.6.40 MCTP_006: MCTP infrastructure basic test

Test needs SMBus interface.

Test will verify whether MCTP Infrastructure is enabled in the ME FW.

Test needs MCTP Infrastructure feature enabled.

Procedure:

Use Aardvark and MESDC tool to check MCTP statistics.

[MANUAL] Plug MCTP PCIe Device (endpoint)

[MANUAL] Power on the platform.

Check „Prepare for endpoint discovery” requests statistic.

Success Criteria:

If statistic are greater than 0, MCTP Infrastructure is initiated in ME FW.

Fail otherwise

Note that this statistic will increment differently in early and late boot up stage.

Before EOP – ME is sending Prepare for EP Discovery every second.

After EOP – ME is sending Prepare for EP discovery once per 10s until ME will detect and successfully discover first valid endpoint which supports MCTP

5.6.41 MCTP_007: MCTP infrastructure communication with the endpoint

Test needs SMBus interface.

Test will verify whether MCTP is enabled in the ME FW. Test is checking MCTP statistics if they increment as it is specified in the MCTP specification.

Test needs MCTP Infrastructure feature enabled.

Procedure:

Use Aardvark and MESDC tool to check MCTP statistics.

[MANUAL] Plug MCTP PCIe Device (endpoint)

[MANUAL] Power on the platform.

Check number of „Endpoint discovery” responses statistic.

Success Criteria:

If statistic are greater than 0, MCTP Infrastructure is initiated in ME FW. Valid endpoint is installed and is responding for MCTP commands from ME.

Fail otherwise

Note that statistic will increment when endpoint will respond to ME request.

After successful discovery process, statistics “Endpoint Discovery” should be at least equal to the statistic “Endpoints Discovered”. It may be bigger and continue to increment in case when more valid endpoints are present on the platform, then max number of endpoints is configured within spsFITC (or above max).

5.6.42 MCTP_008: MCTP infrastructure advanced test (full communication).

Test needs SMBus interface.

Test will verify whether MCTP is enabled in the ME FW. Test is checking MCTP statistics if they increment as it is specified in the MCTP specification. Test verifies the communication on each step.

Test needs MCTP Infrastructure feature enabled.

Procedure:

Use Aardvark and MESDC tool to check MCTP statistics.

[MANUAL] Plug MCTP PCIe Device (endpoint)

[MANUAL] Power on the platform.

Check statistics:

Number of „Prepare for endpoint discovery” requests,

Number of „Endpoint discovery” requests and responses,

Number of Get EID requests and responses,

Number of Set EID requests and responses,

Number of endpoints discovered.

Success Criteria:

If statistic should increment as follows:

Number of „Prepare for endpoint discovery” requests are incrementing,

Number of „Endpoint discovery” requests and responses are non-zero,

Number of Get EID requests and responses are non-zero,

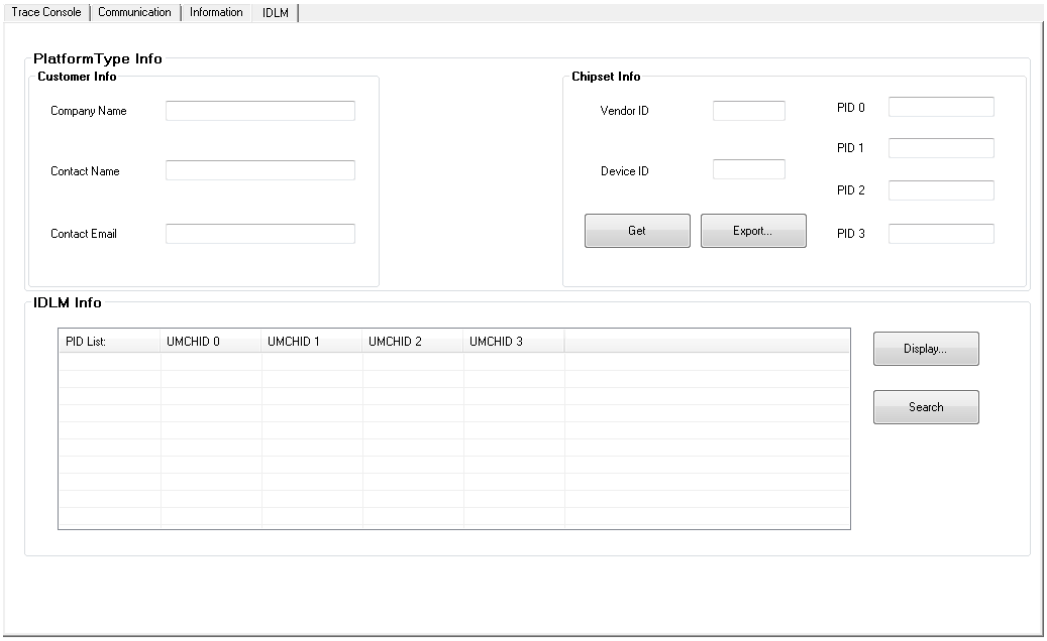
Number of Set EID requests and responses are non-zero,

Number of endpoints discovered are non-zero and are equal at least to the value in the Endpoint discovery statistic

Fail otherwise

5.7 IDLM Module

The IDLM (Intermediate Debug Load Module) functionality is used by Intel support team to enable additional debug capabilities in a particular system.



5.8 Command Line Mode Support

MESDC also has Command Line Interface to support test and reports to be run automatically. The command line will only support limited feature of MESDC. Here is the options supported in MESDC command line interface.

MESDC.exe [-ver] [-h|-?|-help] [-testsCmdLine <path> [-powerReadAppPath <path>] [-cpuLoadAppPath <path>] [-memLoadAppPath <path>] [-interface <interface>] [-aardvarkPort <port>]] [-stateReport <report> [-logFile <file>] [-interface <interface>] [-aardvarkPort <port>] [-oemCaptureConfig <file>]]

Table 5-7.MESDC Command Line Options

Option	Description
-h|-?|-help	Display help screen
-testsCmdLine <path>	Run Compliance Tests from command line with configuration specified in the given file
-powerReadAppPath <path>	Path to Power Read Application for compliance test
-cpuLoadAppPath <path>	Path to CPU Load Application for compliance test
-memLoadAppPath <path>	Path to Memory Load Application for compliance test
-interface <interface>	Select interface to use. For compliance test default interface is SMBus, for reports default is last used interface in GUI <SMBus> <IPMB> <RMCPP>
-aardvarkPort <port>	Aardvark port number. Default: 0

Option	Description
-stateReport <report>	Run report(s); <Oem> - OEM Capture; <MeConfBasic> - ME configuration Basic Partition; <SusramDirect> - Susram Direct; <SusramParse> - Susram Parse; <MeFwHealth> - ME FW Health Check; <Nm> - Node Manager State Check; <Smt> - SMT Driver Statistics; <Heci> - Heci Statistics; <NmExt> - Node Manager State Check Extended; <Icc> - ICC settings;
-logFile <file>	Set file log name. Default: log-<date>-<time>
-oemCaptureConfig <file>	Set configuration file name for Oem Capture report. Default: OEM_CAPTURE.xml

5.8.1 Compliance tests

Here is an example on how to run compliance test(s) over CLI

MESDC.exe –testsCmdLine test.xml –interface SMBus *–aardvarkPort 0*

Switches "– interface" and "*– aardvarkPort" are optional. MESDC.exe by default tries to connect by SMBus on port 0.*

If an incorrect or non-existing xml file is specified, pre-defined file will be created automatically after user's confirmation.

Following is an example of test item in xml file:

<Test id="NM_001" name="Intel NM Bios support test">

<Params>enabled</Params>

</Test>

Test ID should align with the test ID shown in the GUI interface.

Name is optional and for notes only

There are two valid values for Params field, "enabled" and "disabled".

All tests with enabled value in the pre-defined xml file will be run in the command line.

Test result/log will be stored at MESDC directory, same as tests run from GUI.

5.8.2 Reports

Here is an example on how to run report over CLI

MESDC.exe –stateReport SysInfo –logFile log.txt –interface SMBus *–aardvarkPort 0*

Switches "– interface" and "*– aardvarkPort" are optional. MESDC.exe by default tries to connect by interface used in GUI mode last time.*

Switch "–logFile" is optional. By default "ReportsLog.txt" name is used.

Argument for "–stateReport" switch is also optional. By default basic set of reports will be run ("Sys Info" and "Node Manager State Check" if Intel Node Manager feature enabled*).*

6 Flash Programming Tool

The spsFPT is used to program a complete SPI image into the SPI flash device(s).

spsFPT can program each region individually or it can program all of the regions with a single command. You can also use FPT to perform various functions such as:

View the list of regions in the flash on the screen.

Dump the contents of the flash to a file.

Perform a binary file to flash comparison.

Write to a specific address block.

6.1 System Requirements

The Windows version (**spsFPTW.exe**) requires administrator privileges to run under Windows OS. You must use the **Run as Administrator** option to open the CLI in Windows* Vista 64/32-bit and Windows* 7 64/32-bit.

The Windows 64-bit version (spsFPTW64.exe) is designed for running in a 64-bit OS environment which does not have 32-bit compatible mode available, for example WinPE 64.

spsFPT requires an operating system to run on. It is designed to deliver a custom image to a computer that is already able to boot and is not a means to get a blank system up and running. spsFPT must be run on the system with the flash memory that you are programming.

One possible workflow for using spsFPT is:

A pre-programmed flash with a legacy or generic BIOS image is plugged into a new computer.

The computer boots.

spsFPT is run and a custom BIOS/Intel ME/GbE/PDR/DER (optional) image is written to flash.

The computer powers down.

The computer powers up, boots, and is able to access its Intel ME/GbE capabilities as well as any new custom BIOS features.

6.2 Flash Image Details

A flash image is composed of six regions. The locations of these regions are referred to in terms of where they can be found within the overall layout of the flash memory.

Figure 6-1. Flash Image Regions

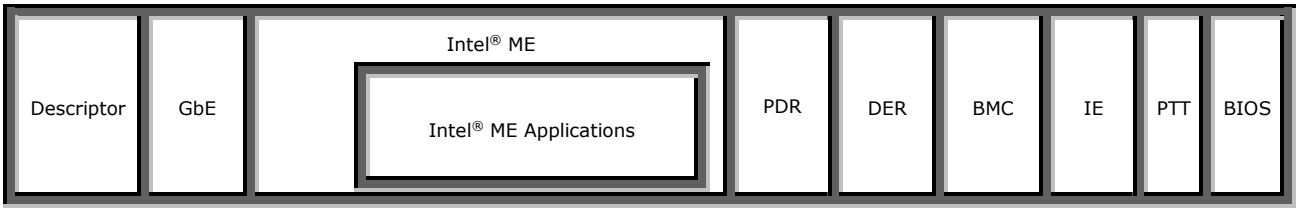


Table 6-1. Flash Image Regions–Description

Region	Description
Descriptor	This region contains information such as the space allocated for each region of the flash image, read-write permissions for each region, and a space which can be used for vendor-specific data. It takes up a fixed amount of space at the beginning of the flash memory. **Note:** This region MUST be locked before the serial flash device is shipped to end users. Please see section 0.0.0 for more information. Failure to lock the Descriptor Region leaves the Intel® ME device vulnerable to security attacks. This region is mandatory and enabled by default.
GbE	This region contains code and configuration data for an Intel Integrated LAN (Gigabit Ethernet). This region is not mandatory and disabled by default.
Intel® ME	This region contains code and configuration data for Intel® ME applications. It takes up a variable amount of space up to the BIOS region. This region is mandatory and enabled by default.
PDR	This region lets system manufacturers describe custom features for the platform. This region is not mandatory and disabled by default.
DER	Device Extension Region used by Intel Node Manager-PTU This region is not mandatory and disabled by default.
BMC	Embedded Controller/Baseboard Management Controller This region is not mandatory and disabled by default.
IE	Innovation Engine This region is not mandatory and disabled by default.
PTT	Platform Trusted Technology This region is mandatory and enabled by default.

BIOS	This region contains code and configuration data for the entire computer. This region is not mandatory and disabled by default.

6.3 Microsoft Windows* Required Files

The Microsoft Windows version of the spsFPT executable is spsFPTW**.exe**. The following files must be in the same directory:

fparts.txt – contains a comma-separated list of attributes for supported flash devices. The text in the file explains each field. An additional entry may be required in this file to describe the flash part which is on the target system. Examine the target board before adding the appropriate attribute values. The supplied file is already populated with default values for SPI devices used with Intel CRBs.

spsFPTW.exe – the executable used to program the final image file into the flash.

In order for tools to work under the Windows* PE environment, you must manually load the driver with the .inf file in the Intel® ME interface driver installation files. Once you locate the .inf file you must use the Windows* PE cmd drvload *.inf to load it into the running system each time Windows* PE reboots. Failure to do so causes errors for some features.

6.4 EFI Required Files

We only support UEFI Shell 2.0.

The EFI version of the spsFPT main executable is **spsFPT.efi**. The following files must be in the same directory:

spsFPT.efi – the executable used to program the final image file into the flash.

fparts.txt – contains a comma-separated list of attributes for supported flash devices. The text in the file explains each field. An additional entry may be required in this file to describe the flash part which is on the target system. Examine the target board before adding in the appropriate attribute values. The supplied file is already populated with default values for SPI devices used with CRBs.

6.5 Programming the Flash Device

Once the Intel ME is programmed, it runs at all times. Intel ME is capable of writing to the flash device at any time, even when the management mode is set to none and it may appear that no writing would occur.

Programming the flash device while Intel ME is running may cause the flash device to become corrupted. Intel ME SPI accessing should be stopped for any flash accessing before programming the full flash device. This should be done to force Intel ME into recovery mode.

6.6 Usage

In UEFI option -? not work (rest options of help work correctly)

Windows and EFI versions of the spsFPT can run with command line options.

To view all of the supported commands: Run the application with the `-?` option.

The commands in Windows and EFI versions have the same syntax. The command line syntax for **spsFPTW.exe and spsFPT.efi** is:

```
spsFPT.exe [-H|?] [-VER] [-EXP] [-VERBOSE] [-Y] [-P] [-LIST]
[-I] [-F] [-ERASE] [-VERIFY] [-D] [-DESC] [-BIOS] [-ME]
[-GBE] [-PDR] [-DER] [-BIOS2] [-BMC] [-DER2] [-IE] [-10GBEA]
[-10GBEB] [-PTT] [-SAVEMAC] [-NOLAN] [-C] [-B] [-E] [-REWRITE]
[-HARDERASE] [-ADDRESS|A] [-LENGTH|L] [-PAGE]
```

Table 6-2. Command Line Options for spsFPT.exe and spsFPTW.exe

Option	Description
-H|?:	Displays help screen.
-VER:	Shows the version of the tools.
-EXP:	Displays example usage of the tool.
-VERBOSE <file>	Displays the tool's debug information or stores it in a log file.
-Y:	Prevents the tool from prompting when a warning occurs and assumes YES as the default answer.
-P:	Specifies a flash part definition file to use.
-LIST:	Supported Flash Parts. Displays all supported flash parts. This option reads the contents of the flash parts definition file and displays the contents on the screen.
-I:	Info. Displays information about the image currently used in the flash.
-F <file> [NoVerify]:	Flash. Programs a binary file into an SPI flash. You must specify the binary file to be flashed. spsFPT reads the binary, erases the flash, and then programs the binary into the flash. After a successful flash, spsFPT verifies that the SPI flash matches the provided image. Without specify the length with –L option, spsFPT will use the total SPI size instead of an image size. NoVerify flag prevents the tool from verifying the flash content after programming it.
-ERASE:	Block Erase. Erases all the blocks in a flash. If a block is already empty then the tool skips it. This option does not use the chip erase command but instead erases the SPI flash block by block. This option can be used with a specific region argument to erase that region. This option cannot be used with the `-f`, `-b`, `-c`, `-d` or `-verify` options.
-VERIFY <file>:	Compare a content of a binary file with the content of the flash.
-D <file> :	Dump. Reads the SPI flash and dumps the flash contents to a file or to the screen using the STDOUT option. The flash device must be written in 4 KB sections. The total size of the flash device must also be in increments of 4 KB.

Option	Description
-DESC:	Read/Write/Verify Descriptor region. Specifies that the Descriptor region is to be read, written, or verified. The start address is the beginning of the region.
-BIOS:	Read/Write/Verify BIOS region. Specifies that the BIOS region is to be read, written, or verified. The start address is the beginning of the region.
-ME:	Read/Write/Verify Intel ME region. Specifies that the Intel ME region is to be read, written, or verified. The start address is the beginning of the region.
-GBE:	Read/Write/Verify GbE region. Specifies that the GbE region is to be read, written, or verified. The start address is the beginning of the region.
-PDR:	Read/Write/Verify PDR region. Specifies that the PDR region is to be read, written, or verified. The start address is the beginning of the region.
-DER:	Read/Write/Verify DER region. Specifies that the DER region is to be read, written, or verified. The start address is the beginning of the region.
-BIOS2:	Read/Write/Verify BIOS2 region. Specifies that the BIOS2 region is to be read, written, or verified. The start address is the beginning of the region.
-BMC:	Read/Write/Verify EC/BMC region. Specifies that the EC/BMC region is to be read, written, or verified. The start address is the beginning of the region.
-DER2:	Read/Write/Verify DER 2 region. Specifies that the DER 2 region is to be read, written, or verified. The start address is the beginning of the region.
-IE:	Read/Write/Verify Innovation Engine region. Specifies that the Innovation Engine region is to be read, written, or verified. The start address is the beginning of the region.
-10GBEA:	Read/Write/Verify 10 GbE A region. Specifies that the 10 GbE A region is to be read, written, or verified. The start address is the beginning of the region.
-10GBEB:	Read/Write/Verify 10 GbE B region. Specifies that the 10 GbE B region is to be read, written, or verified. The start address is the beginning of the region.
-PTT:	Read/Write/Verify PTT region. Specifies that the PTT region is to be read, written, or verified. The start address is the beginning of the region.
-SAVEMAC:	Saves the GbE MAC when GbE is being reflashed - Region 3(GBE) only.
-NOLAN:	Skips 10 Gbe A and 10 Gbe B regions during reflash.
-C:	Chip erase. Erases the contents of SPI flash device(s). This function does NOT erase block by block.
-B:	Blank Check. Checks whether the SPI flash is erased. If the SPI flash is not empty, the application halts as soon as contents are detected. The tool reports the address at which data was found.
-E:	Do not erase area before writing to flash
-REWRITE:	Rewrites the SPI flash with data from a file even if the content of the file is identical to the content of the flash.
-HARDERSE:	Block Erase. Erases all the blocks in a flash without checking firstly if each block is empty. This option does not use the chip erase command but instead erases the SPI flash block by block. This option can be used with a specific region argument to erase that region. This option cannot be used with the `-f`, `-b`, `-c`, `-d` or `-verify` options.
-ADDRESS | A <address>:	Specifies the address from which spsFPT will start reading/writing/verifying.

Option	Description
-LENGTH | L <length>	Specify the length of data which will be read/written/verified.
-page	Pauses at screen / page / window boundaries. Hit any key to continue.

Please be aware that –rewrite option used without any region option will first try to erase entire flash from the beginning.

Table 6-3. Intel Recommended Access Settings

	Intel® ME	GbE	BIOS
Read	0b 0000 1101 = 0x0d	0b 0000 1000 = 0x08	0b 0000 0011 = 0x0B
Write	0b 0000 1100 = 0x0c	0b 0000 1000 = 0x08	0b 0000 0010 = 0x0A

6.7 fparts.txt File

The **fparts.txt** file contains a list of all flash devices that are supported by spsFPT. The flash devices listed in this file must contain a 4 KB erase block size. If the flash device is not listed, you receive the following error:

```
Intel (R) Flash Programming Tool for Server Platform Services.
Version:  X.X.XX.XX
Copyright (c) 2007 - 2015, Intel Corporation. All rights reserved.

Initializing SPI utilities
Reading HSFSTS register... Flash Descriptor: Valid

Region Limits as programmed into the SPI Registers
    FREG0 -  DESC Region:Base Address: 0x000000   Limit :  0x000FFF
    FREG1 -  BIOS Region:Base Address: 0x800000   Limit :  0xFFFFFF
    FREG2 -  ME   Region:Base Address: 0x013000   Limit :  0x7FFFFF
    FREG3 -  GbE  Region:Base Address: 0x001000   Limit :  0x002FFF
    FREG4 -  PDR  Region:Base Address: 0x003000   Limit :  0x012FFF
    FREG5 -  DER  Region:Base Address: 0x1FFF000   Limit :
0x000FFF
Address Limit 0x1000000    Maximum Memory 16384kB


    --- Flash Devices Found ---
```

Error 103: There are no supported SPI flash devices installed. Please check connectivity and orientation of SPI flash device. If the device is not located in **fparts.txt**, you are expected to provide information about the device, inserting the values into **fparts.txt** in same format as is used for the rest of the devices. Detailed information on how to derive the values in **fparts.txt** is found in the Intel® 6 Series Chipset SPI Programming Guide. The device must have a 4 KB erase sector and the total size of the SPI Flash device must be a multiple of
4 KB. The values are listed in columns in the following order:

- Display name
- Device ID (2 or 3 bytes)
- Device Size (in bits)
- Block Erase Size (in bytes - 256, 4K, 64K)
- Block Erase Command
- Write Granularity (1 or 64)
- Enable Write Status Register Command (1- True, 0- False) Chip Erase Command.

Chip Erase Timeout (in milliseconds)

6.8 Examples

The following examples illustrate the usage of the DOS version of the tool (**spsFPT.exe)**. The Windows version of the tool (**spsFPTW.exe**) and EFI version of the tool (**spsFPT.efi**) behave in the same manner apart from running in a Windows/EFI environment.

6.8.1 Example 1 – Flash SPI Flash Device with Binary File

```
C:\ spsFPTW.exe –f spi.bin
```

This command writes the data in the **spi.bin** file into a whole SPI flash from address 0x00.

6.8.2 Example 2 – Program a Specific Region

```
spsFPTW.exe –f ME.rom –ME
-------------------------------------------
Intel (R) Flash Programming Tool for Server Platform Services.
Version:  4.2.12.3
Copyright (c) 2007 - 2015, Intel Corporation. All rights reserved.

Reading HSFSTS register... Flash Descriptor: Valid

    --- Flash Devices Found ---
    W25Q128BV    ID:0xEF4018    Size: 16384KB (131072Kb)


- Reading Flash [0x800000] 8116KB of 8116KB - 100% complete.
- Erasing Flash Block [0x019000] - 100% complete.
- Programming Flash [0x019000]   24KB of   24KB - 100% complete.
- Erasing Flash Block [0x025000] - 100% complete.
- Programming Flash [0x025000]    8KB of    8KB - 100% complete.
- Erasing Flash Block [0x04D000] - 100% complete.
- Programming Flash [0x04D000]  100KB of  100KB - 100% complete.
- Erasing Flash Block [0x055000] - 100% complete.
- Programming Flash [0x055000]    4KB of    4KB - 100% complete.
- Erasing Flash Block [0x062000] - 100% complete.
- Programming Flash [0x062000]   40KB of   40KB - 100% complete.
```

```
- Erasing Flash Block [0x065000] - 100% complete.
- Programming Flash [0x065000]    4KB of    4KB - 100% complete.
- Erasing Flash Block [0x069000] - 100% complete.
- Programming Flash [0x069000]    4KB of    4KB - 100% complete.
- Erasing Flash Block [0x06D000] - 100% complete.
- Programming Flash [0x06D000]    4KB of    4KB - 100% complete.
- Erasing Flash Block [0x071000] - 100% complete.
- Programming Flash [0x071000]    4KB of    4KB - 100% complete.
- Erasing Flash Block [0x075000] - 100% complete.
- Programming Flash [0x075000]    4KB of    4KB - 100% complete.
- Erasing Flash Block [0x079000] - 100% complete.
- Programming Flash [0x079000]    4KB of    4KB - 100% complete.
- Erasing Flash Block [0x07D000] - 100% complete.
- Programming Flash [0x07D000]    4KB of    4KB - 100% complete.
- Erasing Flash Block [0x081000] - 100% complete.
- Programming Flash [0x081000]    4KB of    4KB - 100% complete.
- Erasing Flash Block [0x085000] - 100% complete.
- Programming Flash [0x085000]    4KB of    4KB - 100% complete.
- Erasing Flash Block [0x089000] - 100% complete.
- Programming Flash [0x089000]    4KB of    4KB - 100% complete.
- Erasing Flash Block [0x08D000] - 100% complete.
- Programming Flash [0x08D000]    4KB of    4KB - 100% complete.
- Erasing Flash Block [0x091000] - 100% complete.
- Programming Flash [0x091000]    4KB of    4KB - 100% complete.
- Erasing Flash Block [0x1EB000] - 100% complete.
- Programming Flash [0x1EB000] 1372KB of 1372KB - 100% complete.
- Erasing Flash Block [0x3EB000] - 100% complete.
- Programming Flash [0x3EB000] 1372KB of 1372KB - 100% complete.
- Erasing Flash Block [0x497000] - 100% complete.
- Programming Flash [0x497000]   12KB of   12KB - 100% complete.
- Verifying Flash [0x800000] 8116KB of 8116KB - 100% complete.
RESULT: The data is identical.
```

spsFPT Operation Passed - This command writes the data in **ME.bin** into the Intel ME region of the SPI flash and verifies that the operation ran successfully.

6.8.3 Example 3 – Display SPI Information

```
spsFPTW.exe –I
-----------------------------------------
Intel (R) Flash Programming Tool for Server Platform Services.
Version:  X.X.XX.XX
Copyright (c) 2007 - 2014, Intel Corporation. All rights reserved.

Reading HSFSTS register... Flash Descriptor: Valid

    --- Flash Devices Found ---
    W25Q128BV    ID:0xEF4018    Size: 16384KB (131072Kb)

    --- Flash Image Information --
    Signature: VALID
    Number of Flash Components: 1
        Component 1 - 16384KB (131072Kb)
    Regions:
        Descriptor - Base: 0x000000, Limit: 0x000FFF
        BIOS       - Base: 0x800000, Limit: 0xFFFFFF
        ME         - Base: 0x013000, Limit: 0x7FFFFF
        GbE        - Base: 0x001000, Limit: 0x002FFF
        PDR        - Base: 0x003000, Limit: 0x012FFF
        DER        - Not present
    Master Region Access:
        CPU/BIOS - ID: 0x0000, Read: 0x1B, Write: 0x3A
        ME       - ID: 0x0000, Read: 0x25, Write: 0x04
        GbE      - ID: 0x0118, Read: 0x09, Write: 0x08

Total Accessible SPI Memory: 16384KB, Total Installed SPI Memory :
16384KB
```

spsFPT Operation Passed - This command displays information about the flash devices present in the computer. The base address refers to the start location of that region and the limit address refers to the end of the region. If the flash device is not specified in **fparts.txt**, spsFPT returns the error message "There is no supported SPI flash device installed."

7 spsManuf and spsManufWin

spsManuf validates Intel ME functionality (verifies that all its components have been assembled together correctly) on the manufacturing line.

The Windows version of spsManuf requires administrator privilege to run under windows OS. You need to explicitly click on the context menu in Windows "Run as Administrator" under Windows Server 2008 R2 64 bit.

We only support UEFI Shell 2.0.

spsManuf does not check for LAN functionality. The tool assumes that all Intel ME components on the test board have been validated by their respective vendors. The tool verifies that these components have been assembled together correctly.

7.1 How to use spsManuf

Functionality of spsManuf consist of two test groups:

- default tests
- optional tests

Default tests are run every time user starts spsManuf and there is no possibility to turn it off.

By configuration file user can specify optional tests which should be executed. It is necessary because spsManuf have to know what value is correct for particular test in user opinion.

VSCCOMMN.bin file is required to verify the VSCC entry on the platform. You need to have this file at the location you run spsManuf, otherwise spsManuf will report error.

7.2 Tests Description

Table 7-1 List of Default and Optional tests

Test Group	Subtest	Runs when	Purpose	Not applicable for
Default	ME Hardware and Firmware Status	Always	Confirms that Intel ME HW and FW are alive and operating in Normal Mode.	
	ME VSCC		Confirms that VSCC in Intel ME include the Intel-recommended value for the installed SPI device(s).	
Optional	Runtime Image FW Version	If subtest is not commented out in config file and spsManuf.cfg exists or –F<file> option was set.	Compares FW version of the running OpImage.	
	Backup Image FW Version		Compares FW version of the backup OpImage.	
	Recovery Image FW Version		Compares FW version of the Recovery Image.	
	Factory Default Configuration		Confirms that Factory Default Configuration matches the intended design.	
	ME Integrity Check			
	Node Manager		Checks if Intel Node Manager is enabled or disabled	Harrisonville, Purley-EPO
	PECI Proxy		Checks if PECI Proxy is enabled or disabled	Purley-EPO
	ICC		Checks if ICC is enabled or disabled	Harrisonville
	ME Storage Services		Checks if ME Storage Services are enabled or disabled	Greenlow, Purley-EPO, Harrisonville, Ridgeport
	Boot Guard		Checks if Boot Guard is enabled or disabled	Greenlow, Purley-EPO, Harrisonville, Ridgeport
	Platform Trusted Technology (PTT)		Checks if PTT is enabled or disabled	Greenlow, Harrisonville, Ridgeport
	OEM Defined CPU Debug Policy		Checks if OEM Defined CPU Debug Policy is enabled or disabled	Greenlow, Purley-EPO, Purley, Ridgeport
	Reset Suppression (Pre-Go-S1)		Checks if Reset Suppression is enabled or disabled	Purley-EPO, Ridgeport
	PMBus Proxy over HECI		Checks if PMBus Proxy over HECI is enabled or disabled	Purley-EPO

Test Group	Subtest	Runs when	Purpose	Not applicable for
	CPU Hot Plug/Remove		Checks if CPU Hot Plug/Remove over HECI is enabled or disabled	Greenlow, Purley-EPO, Harrisonville, Bakerville
	MIC Proxy (aka IPMB Proxy)		Checks if MIC Proxy (aka IPMB Proxy) is enabled or disabled	Greenlow, Purley-EPO, Harrisonville, Bakerville, Ridgeport
	MCTP Proxy		Checks if MCTP Proxy is enabled or disabled	Harrisonville, Purley-EPO
	Thermal Reporting and Volumetric Airflow		Checks if Thermal Reporting and Volumetric Airflow is enabled or disabled	Greenlow, Purley-EPO, Harrisonville, Bakerville, Ridgeport
	SoC Thermal Reporting		Checks if SoC Thermal Reporting is enabled or disabled	Greenlow, Purley-EPO, Purley, Harrisonville, Ridgeport
	Dual BIOS Support		Checks if Dual BIOS is supported or unsupported	Greenlow, Purley-EPO, Purley, Harrisonville, Bakerville
	MPHY Survivability Programming		Checks if MPHY Survivability Programming is enabled or disabled	Greenlow, Purley-EPO, Purley, Harrisonville, Ridgeport
	InBand PECI		Checks if InBand PECI is enabled or disabled	Greenlow, Purley-EPO
	PCH Debug (Intel(R) Silicon View)		Checks if PCH Debug (Intel(R) Silicon View) is enabled or disabled	Greenlow
	Power Thermal Utility Support		Checks if Power Thermal Utility is supported or unsupported	Harrisonville, Purley-EPO
	FIA MUX Configuration		Checks if FIA MUX Configuration is enabled or disabled	Greenlow, Purley-EPO, Purley, Ridgeport, Bakerville
	PCH Thermal Sensor Init		Checks if PCH Thermal Sensor Init is enabled or disabled	
	DeepSx (EU Lot6) support		Checks if DeepSx (EU Lot6) is supported or unsupported	Greenlow, Purley-EPO
	Dual Intel(R) ME FW Image		Checks if Dual Intel(R) ME FW Image is supported or unsupported	
	Direct FW Update (DFU)		Checks if Direct FW Update (DFU) is enabled or disabled	Harrisonville, Purley-EPO
	MCTP Infrastructure		Checks if MCTP Infrastructure is enabled or disabled	Harrisonville, Purley-EPO
	CUPS		Checks if CUPS is enabled or disabled	Greenlow, Purley-EPO, Harrisonville, Ridgeport
	Flash Descriptor Region Verification		Checks if Flash Descriptor Region Verification is enabled or disabled	Greenlow, Ridgeport
	Turbo State Limiting		Checks if Turbo State Limiting is enabled or disabled	Purley-EPO, Harrisonville, Bakerville, Ridgeport

Test Group	Subtest	Runs when	Purpose	Not applicable for
	Telemetry Hub		Checks if Telemetry Hub is enabled or disabled	Greenlow, Purley-EPO, Harrisonville
	Intel(R) ME Shutdown on EOP		Checks if Intel(R) ME Shutdown on EOP is supported or unsupported	Greenlow, Purley-EPO, Purley, Bakerville, Ridgeport
	ASA		Checks if ASA is enabled or disabled	Greenlow, Purley-EPO, Harrisonville, Bakerville, Ridgeport
	Warm Reset Notification Sub-Flow		Checks if Warm Reset Notification Sub-Flow is enabled or disabled	Greenlow, Purley-EPO, Harrisonville
	PTU Option ROM Version Check		Performs comparison between expected and obtained PTU ROM version	
	EOP Status		Checks End-Of-Post reception by Intel ME Firmware.	
	Fd Public Key Hash		Performs comparison between expected and obtained truncated Flash Descriptor Public Key Hash	
	Flash Descriptor Verification Enabled		Performs comparison between expected and obtained Flash Descriptor Verification Enabled	
	Intel(R) PTT Supported [FPF]		Performs comparison between expected and obtained Intel(R) PTT Supported [FPF]	
	OEM Secure Boot Policy		Performs comparison between expected and obtained OEM Secure Boot Policy value	
	OEM ID		Performs comparison between expected and obtained OEM ID value	Products below SPS 5.0
	OEM Platform ID		Performs comparison between expected and obtained OEM Platform ID value	Products below SPS 5.0
	ACM SVN		Performs comparison between expected and obtained ACM SVN value	Products below SPS 5.0
	KM SVN		Performs comparison between expected and obtained KM SVN value	Products below SPS 5.0
	BSMM SVN		Performs comparison between expected and obtained BSMM SVN value	Products below SPS 5.0

Test Group	Subtest	Runs when	Purpose	Not applicable for
	Anti Rollback SVN		Performs comparison between expected and obtained Anti Rollback SVN value	Products below SPS 5.0
	Cpu Debug Policy Enabled		Performs comparison between expected and obtained Cpu Debug Policy Enabled	
	Boot Guard FPFs Enabled Note: This subtest will only work after End Of Manufacturing		Performs comparison between expected and obtained Boot Guard FPFs Enabled	
	Error Enforcement Policy 0		Performs comparison between expected and obtained Error Enforcement Policy 0	
	Error Enforcement Policy 1		Performs comparison between expected and obtained Error Enforcement Policy 1	
	Boot Guard Policy Restrictions		Performs comparison between expected and obtained Boot Guard Policy Restrictions	
	Boot Guard Policy Type		Performs comparison between expected and obtained Boot Guard Policy Type	
	Key Manifest ID		Performs comparison between expected and obtained Key Manifest ID	
	IE Verified Boot Enabled		Checks if IE Verified Boot is enabled or disabled	
	OEM Public Key Hash		Performs comparison between expected and obtained OEM Public Key Hash	
	IE Verified Boot Hash		Performs comparison between expected and obtained IE Verified Boot Hash	
	IE OEM Key Hash		Performs comparison between expected and obtained IE OEM Key Hash	

Test Group	Subtest	Runs when	Purpose	Not applicable for
	Chipset Real Fusing		Checks if Chipset Real Fusing is supported or unsupported	Greenlow, Purley-EPO, Harrisonville, Ridgeport, Bakerville
	State of End Of Manufacturing		Checks if End Of Manufacturing is enabled or disabled	Greenlow
	TXT Supported		Checks if TXT is supported (FPF check)	Products below SPS 5.0
	OEM Key Manifest Present		Checks presence of OEM Key Manifest (FPF check)	Products below SPS 5.0
	FPF consistency check		Checks if co-signing is properly configured	Idaville, Jacobsville

7.3 Usage

In UEFI option -? not work (rest options of help work correctly)

The DOS, EFI and Linux version of the tool can be operated using the same syntax as the Windows version. The Windows version of the tool can be executed by:

```
spsManufWin64.exe [-EXP] [-H|?] [-VER] [-F] [-CFGGEN] [-VERBOSE] [-PAGE]
[-SETEOM] [-FPFDISABLE][-PCHBUSID]
```

It is possible to use "/" instead of "-" in command line.

Table 7-2. Command Line Options for spsManuf

Option	Description
No option	Runs all hardcoded default subtests. In addition, if a file named spsManuf.cfg exists in the spsManuf directory, all optional subtests found in it will run.
–F <file>	Runs all hardcoded default subtests. In addition, this option will run several checks according to configuration file. The checks can be configured by customer to select which test items he is expecting to run and what is the proper value. Sub option "file" is mandatory.
–CFGGEN <-F [file]>	This option generates default spsManuf.cfg configuration file with complete help and comments included. User can specify name of generated file by <– F[file]> sub option.
–VERBOSE <file>	Displays the tool's debug information or stores it in a log file.
–PAGE	When more than one full screen (80 x 25 under DOS, various under Windows depending on console windows setting for the visible windows size) of information is displayed, this option allows user to pause the output and press any key before continuing on to the next screen.

Option	Description
–VER	Show the version of the tool.
–H or –?	Display help screen.
–EXP	Show the examples on how to use the tool.
–SETEOM	Send "End of manufacturing" and disable "Manufacturing Mode", operation is permanent and irreversable
-FPFDISABLE <feature>	Disable FPF security features defined by subaparams (can be used more than one). Available subparams: BtG – disables Boot Guard PTT – disables Platform Trusted Technology FDV – disables Flash Descriptor Verification CPUDEBUG – OEM Defined CPU Debug IE – Innovative Engine Disabling security features is available only during "Manufacturing Mode". Changes will become effective only after setting "End of manufacturing" (-SETEOM option)
–PCHBUSID <pchBusId>	Select PCH by PCI Bus Id Note: This option applies only for multi-PCH system. Without this option by default PCI Bus Id is 0. To select PCH connected to another PCI bus you need to know to which PCI bus Id the PCH is attached. ***Note***: Tool doesn't provide scan functionality.

7.4 spsManuf.cfg File

Configuration file (by default: spsManuf.cfg) includes all the test's configurations for spsManuf –F check. It needs to be at the same folder as you run spsManuf from. If there is no configuration file existing in that folder you can generate it by –CFGGEN <-F[file]> command.

Here is an example of configuration file:

```
// If one of these check fails, by default spsManuf will report error and
// continue on to the next check. If a user doesn't wish to continue
// when an error is found, ErrAction field can be used. Please see
// the examples here for detailed explanation:
//
//  SubTestName="Runtime Image FW Version", ReqVal="1.2.3.4", ErrAction="ErrorStop"
//
// If the above test fails, spsManuf will report error and stop. There
// are total of three different error actions user can choose from:
//
// ErrorContinue - report error and continue on to the next check
// ErrorStop - report error and stop any check after the current one
// WarnContinue - report warning and continue on to the next check
//
// To add comment or take out a specific test, leave // at the start
// of a line. This file is processed by spsManuf line by line as text
// file. Duplication of the same sub-tests are allowed, but spsManuf
```

```
// will always perform the first test to the last test from the file.

// All string comparisons given in this file are case insensitive
// compare. There might be multiple field name/value pairs in one
// entry, but each field needs to be specified in the following
// format where <field name> can be replaced by SubTestName, ReqVal
// or ErrAction, <field value> can be replaced by any string including
// dash and/or spaces surrounded by double quotation marks.
// No line Wrapping is supported:
//
//    <field name>="<field value>", such as ReqVal="<value>"

// Some default checks run every time user starts spsManuf
// There is no possibility to turn it off
//
// SubTestName="ME Hardware and Firmware Status"
// SubTestName="ME VSCC"

////////////////////////////////////////////////////////////////////////////////
// The following Configuration Check requires a user to enter an expected
// value after ReqVal=
////////////////////////////////////////////////////////////////////////////////

////////////////////////////////////////////////////////////////////////////////
// Runtime/Backup/Recovery Image FW version is a string as
//"<major ver>.<minor ver>.<hotfix ver>.<build num>"
////////////////////////////////////////////////////////////////////////////////

// SubTestName="Runtime Image FW version", ReqVal=
// SubTestName="Backup Image FW version", ReqVal=
// SubTestName="Recovery Image FW version", ReqVal=

////////////////////////////////////////////////////////////////////////////////
// Factory Default Configuration is a string as
//"XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX
XX XX XX XX XX"
// where X is a hex value
////////////////////////////////////////////////////////////////////////////////

// SubTestName="Factory Default Configuration", ReqVal=

////////////////////////////////////////////////////////////////////////////////
// ME Integrity Check
////////////////////////////////////////////////////////////////////////////////

// SubTestName="ME Integrity Check"

////////////////////////////////////////////////////////////////////////////////
// ReqVal for each feature below is a string as:
//"enabled", "disabled", "supported" or "unsupported"
////////////////////////////////////////////////////////////////////////////////

// Not applicable for: Harrisonville, Purley-EPO, Jacobsville, Tinsley
```

```
// SubTestName="Node Manager", ReqVal=

// Not applicable for: Purley-EPO, Tinsley
// SubTestName="PECI Proxy", ReqVal=

// Not applicable for: Harrisonville
// SubTestName="ICC", ReqVal=

// Not applicable for: Greenlow, Purley-EPO, Harrisonville, Ridgeport, Tinsley
// SubTestName="ME Storage Services", ReqVal=

// Not applicable for: Greenlow, Purley-EPO, Harrisonville, Ridgeport, Tinsley
// SubTestName="Boot Guard", ReqVal=

// Not applicable for: Greenlow, Harrisonville, Ridgeport, Tinsley
// SubTestName="Platform Trusted Technology (PTT)", ReqVal=

// Not applicable for: Greenlow, Purley, Purley-EPO, Ridgeport, Whitley, Mehlow,
Tinsley
// SubTestName="OEM Defined CPU Debug Policy", ReqVal=

// Not applicable for: Purley-EPO, Ridgeport, Mehlow, Tinsley
// SubTestName="Reset Suppression (Pre-Go-S1)", ReqVal=

// Not applicable for: Purley-EPO, Tinsley
// SubTestName="PMBus Proxy over HECI", ReqVal=

// Not applicable for: Greenlow, Purley-EPO, Harrisonville, Bakerville, Idaville,
Jacobsville, Mehlow, Tinsley
// SubTestName="CPU Hot Plug/Remove", ReqVal=

// Not applicable for: Greenlow, Purley-EPO, Harrisonville, Ridgeport, Bakerville,
Idaville, Jacobsville, Mehlow, Tinsley
// SubTestName="MIC Proxy (aka IPMB Proxy)", ReqVal=

// Not applicable for: Purley-EPO, Harrisonville, Idaville, Jacobsville, Mehlow,
Tinsley
// SubTestName="MCTP Proxy", ReqVal=

// Not applicable for: Greenlow, Purley-EPO, Harrisonville, Ridgeport, Bakerville,
Idaville, Jacobsville, Mehlow, Tinsley
// SubTestName="Thermal Reporting and Volumetric Airflow", ReqVal=

// Not applicable for: Greenlow, Purley, Purley-EPO, Harrisonville, Ridgeport,
Whitley, Idaville, Jacobsville, Mehlow, Tinsley
// SubTestName="SoC Thermal Reporting", ReqVal=

// Not applicable for: Greenlow, Purley, Purley-EPO, Harrisonville, Bakerville,
Whitley, Mehlow, Tinsley
// SubTestName="Dual BIOS Support", ReqVal=

// Not applicable for: Greenlow, Purley, Purley-EPO, Harrisonville, Ridgeport,
Tinsley
```

```
// SubTestName="MPHY Survivability Programming", ReqVal=

// Not applicable for: Greenlow, Purley-EPO, Mehlow, Tinsley
// SubTestName="InBand PECI", ReqVal=

// Not applicable for: Greenlow, Tinsley
// SubTestName="PCH Debug (Intel(R) Silicon View)", ReqVal=

// Not applicable for: Purley-EPO, Harrisonville, Jacobsville, Tinsley
// SubTestName="Power Thermal Utility Support", ReqVal=

// Not applicable for: Greenlow, Purley, Purley-EPO, Ridgeport, Bakerville, Whitley,
Mehlow, Tinsley
// SubTestName="FIA MUX Configuration", ReqVal=

// Not applicable for: Tinsley
// SubTestName="PCH Thermal Sensor Init", ReqVal=

// Not applicable for: Greenlow, Purley-EPO, Idaville, Jacobsville, Tinsley
// SubTestName="DeepSx (EU Lot6) support", ReqVal=

// Not applicable for: Tinsley
// SubTestName="Dual Intel(R) ME FW Image", ReqVal=

// Not applicable for: Purley-EPO, Tinsley
// SubTestName="Direct FW Update (DFU)", ReqVal=

// Not applicable for: Purley-EPO, Harrisonville, Tinsley
// SubTestName="MCTP Infrastructure", ReqVal=

// Not applicable for: Greenlow, Purley-EPO, Harrisonville, Ridgeport, Jacobsville,
Mehlow, Tinsley
// SubTestName="CUPS", ReqVal=

// Not applicable for: Greenlow, Ridgeport, Bakerville
// SubTestName="Flash Descriptor Region Verification", ReqVal=

// Not applicable for: Purley-EPO, Harrisonville, Ridgeport, Bakerville, Idaville,
Jacobsville, Mehlow, Tinsley
// SubTestName="Turbo State Limiting", ReqVal=

// Not applicable for: Greenlow, Purley-EPO, Harrisonville, Jacobsville, Tinsley
// SubTestName="Telemetry Hub", ReqVal=

// Not applicable for: Greenlow, Purley, Purley-EPO, Ridgeport, Bakerville, Whitley,
Idaville, Jacobsville, Mehlow, Tinsley
// SubTestName="Intel(R) ME Shutdown on EOP", ReqVal=

// Not applicable for: Greenlow, Purley-EPO, Harrisonville, Ridgeport, Bakerville,
Whitley, Idaville, Jacobsville, Mehlow, Tinsley
// SubTestName="ASA", ReqVal=
```

```
// Not applicable for: Greenlow, Purley-EPO, Harrisonville, Idaville, Jacobsville,
Mehlow, Tinsley
// SubTestName="Warm Reset Notification Sub-Flow", ReqVal=


////////////////////////////////////////////////////////////////////////////////
// PTU Option ROM version is a string of format X.Y
// where X is major and Y is minor version decimal value
////////////////////////////////////////////////////////////////////////////////

// SubTestName="PTU Option ROM Version Check", ReqVal=

////////////////////////////////////////////////////////////////////////////////
// Tests without ReqVal needed
////////////////////////////////////////////////////////////////////////////////

// SubTestName="EOP Status"

////////////////////////////////////////////////////////////////////////////////
// Fd Public Key Hash, ReqVal is string as
//"XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX"
////////////////////////////////////////////////////////////////////////////////

// SubTestName="Fd Public Key Hash", ReqVal=

////////////////////////////////////////////////////////////////////////////////
// Flash Descriptor Verification Enabled, ReqVal is string as
//"XX"
////////////////////////////////////////////////////////////////////////////////

// SubTestName="Flash Descriptor Verification Enabled", ReqVal=

////////////////////////////////////////////////////////////////////////////////
// Intel(R) PTT Supported [FPF], ReqVal is string as
//"XX"
////////////////////////////////////////////////////////////////////////////////

// SubTestName="Intel(R) PTT Supported [FPF]", ReqVal=

////////////////////////////////////////////////////////////////////////////////
// OEM Secure Boot Policy, ReqVal is string as
//"XX XX"
////////////////////////////////////////////////////////////////////////////////

// SubTestName="OEM Secure Boot Policy", ReqVal=

////////////////////////////////////////////////////////////////////////////////
// OEM ID, ReqVal is string as
//"XX XX"
////////////////////////////////////////////////////////////////////////////////

// Not applicable before SPS 5.0
// SubTestName="OEM ID", ReqVal=
```

```
////////////////////////////////////////////////////////////////////////////////
// OEM Platform ID, ReqVal is string as
//"XX XX"
////////////////////////////////////////////////////////////////////////////////

// Not applicable before SPS 5.0
// SubTestName="OEM Platform ID", ReqVal=

////////////////////////////////////////////////////////////////////////////////
// ACM SVN, ReqVal is string as
//"XX"
////////////////////////////////////////////////////////////////////////////////

// Not applicable before SPS 5.0
// SubTestName="ACM SVN", ReqVal=

////////////////////////////////////////////////////////////////////////////////
// KM SVN, ReqVal is string as
//"XX"
////////////////////////////////////////////////////////////////////////////////

// Not applicable before SPS 5.0
// SubTestName="KM SVN", ReqVal=

////////////////////////////////////////////////////////////////////////////////
// BSMM SVN, ReqVal is string as
//"XX"
////////////////////////////////////////////////////////////////////////////////

// Not applicable before SPS 5.0
// SubTestName="BSMM SVN", ReqVal=

////////////////////////////////////////////////////////////////////////////////
// Anti Rollback SVN, ReqVal is string as
//"XX"
////////////////////////////////////////////////////////////////////////////////

// Not applicable before SPS 5.0
// SubTestName="Anti Rollback SVN", ReqVal=

////////////////////////////////////////////////////////////////////////////////
// Cpu Debug Policy Enabled, ReqVal is string as
//"XX"
////////////////////////////////////////////////////////////////////////////////

// SubTestName="Cpu Debug Policy Enabled", ReqVal=

////////////////////////////////////////////////////////////////////////////////
// Boot Guard FPFs Enabled, ReqVal is string as
//"XX"
// This test will only work in End Of Manufacturing mode
////////////////////////////////////////////////////////////////////////////////
```

```
// SubTestName="Boot Guard FPFs Enabled", ReqVal=

////////////////////////////////////////////////////////////////////////////////
// Error Enforcement Policy 0, ReqVal is string as
//"XX"
////////////////////////////////////////////////////////////////////////////////

// SubTestName="Error Enforcement Policy 0", ReqVal=

////////////////////////////////////////////////////////////////////////////////
// Error Enforcement Policy 1, ReqVal is string as
//"XX"
////////////////////////////////////////////////////////////////////////////////

// SubTestName="Error Enforcement Policy 1", ReqVal=

////////////////////////////////////////////////////////////////////////////////
// Boot Guard Policy Restrictions, ReqVal is string as
//"XX"
////////////////////////////////////////////////////////////////////////////////

// SubTestName="Boot Guard Policy Restrictions", ReqVal=

////////////////////////////////////////////////////////////////////////////////
// FPGA Platform Anchoring, ReqVal is string as
//"XX"
////////////////////////////////////////////////////////////////////////////////

// SubTestName="FPGA Platform Anchoring", ReqVal=

////////////////////////////////////////////////////////////////////////////////
// Boot Guard Policy Type, ReqVal is string as
//"XX"
////////////////////////////////////////////////////////////////////////////////

// SubTestName="Boot Guard Policy Type", ReqVal=

////////////////////////////////////////////////////////////////////////////////
// Key Manifest ID, ReqVal is string as
//"XX"
////////////////////////////////////////////////////////////////////////////////

// SubTestName="Key Manifest ID", ReqVal=

////////////////////////////////////////////////////////////////////////////////
// OEM Public Key Hash, ReqVal is string as
//"XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX
XX XX XX XX XX"
////////////////////////////////////////////////////////////////////////////////

// SubTestName="OEM Public Key Hash", ReqVal=
```

```
////////////////////////////////////////////////////////////////////////////////
// IE Verified Boot Enabled, ReqVal is string as
//"XX"
////////////////////////////////////////////////////////////////////////////////

// SubTestName="IE Verified Boot Enabled", ReqVal=

////////////////////////////////////////////////////////////////////////////////
// IE Verified Boot Hash, ReqVal is string as
//"XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX
XX XX XX XX XX"
////////////////////////////////////////////////////////////////////////////////

// SubTestName="IE Verified Boot Hash", ReqVal=

////////////////////////////////////////////////////////////////////////////////
// IE OEM Key Hash, ReqVal is string as
//"XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX
XX XX XX XX XX"
////////////////////////////////////////////////////////////////////////////////

// SubTestName="IE OEM Key Hash", ReqVal=

////////////////////////////////////////////////////////////////////////////////
// SubTestName="Chipset Real Fusing", ReqVal is string as "true" or "false"
////////////////////////////////////////////////////////////////////////////////

// Not applicable for: Greenlow, Purley-EPO, Harrisonville, Ridgeport, Bakerville
// SubTestName="Chipset Real Fusing", ReqVal=

////////////////////////////////////////////////////////////////////////////////
// SubTestName="State of End Of Manufacturing", ReqVal is string as "Pre-EOM" or
"EOM"
////////////////////////////////////////////////////////////////////////////////

// Not applicable for: Greenlow
// SubTestName="State of End Of Manufacturing", ReqVal=

////////////////////////////////////////////////////////////////////////////////
// SubTestName="TXT Supported", ReqVal is string as "true" or "false"
////////////////////////////////////////////////////////////////////////////////

// Not applicable before SPS 5.0
// SubTestName="TXT Supported", ReqVal=

////////////////////////////////////////////////////////////////////////////////
// SubTestName="OEM Key Manifest Present", ReqVal is string as "true" or "false"
////////////////////////////////////////////////////////////////////////////////

// Not applicable before SPS 5.0
// SubTestName="OEM Key Manifest Present", ReqVal=

////////////////////////////////////////////////////////////////////////////////
```

```
// SubTestName="FPF consistency check"
////////////////////////////////////////////////////////////////////////////////

// Applicable only for: Idaville, Jacobsville
// SubTestName="FPF consistency check"
```

Please note that lines start with // are for comment, and they also used for the purpose to inform users the available test group names and specific checks names included in each test that spsManuf recognizes. To select which test items to run, user can create a line begins with `SubTestName=` with a specific sub test name. Here are some additional examples that explain how to use this feature:

User wants to run Intel ME FW version check and a valid Intel ME FW version should be equal to string 1.2.3.4:

```
SubTestName="Runtime Image FW version", ReqVal="1.2.3.4"
```

7.5 Output/Result

There are 3 possible results displayed in verbose mode at the optional tests checking:

Pass – meaning all tests passed

Pass with warning – meaning only tests with error action set as `"WarnContinue"` failed.

Fail - meaning any error occurs in the test as customer defined at error items.

7.6 Examples

>spsManufWin64.exe -f spsManufAllPositive.cfg

```
spsManuf Test Passed
```

>spsManufWin64.exe -f spsManufAllPositive.cfg -verbose

```
Intel(R) spsManuf Version: 3.0.3.7013
Copyright(C) 2005 - 2015, Intel Corporation. All rights reserved.

Number of LPC Devices supported: 65
LPC Device ID: A148.
Platform: Intel(R) SPT_H
Checking ME Hardware and Firmware Status...passed
vsccommn.bin was created on 18:47:14 06/03/2013 GMT
SPI Flash ID #1 ME VSCC value is 0x2025, device supports SFDP capability
SPI Flash ID #1 (ID: 0xEF4018) SFDP BES is 4kB, ME VSCC comparison to
Intel recommended value not needed.
Checking ME VSCC status...passed
Checking default spsManuf tests...passed

spsManuf.cfg is found with 7 valid test entries
```

```
Checking Node Manager status...passed

Checking PECI Proxy status...passed

Checking ICC status...passed

Checking MPHY Survivability Programming status...passed

Checking Power Thermal Utility Support status...passed

Checking MCTP Infrastructure status...passed

Checking Turbo State Limiting status...passed


Number of optional tests executed: 7
Passed: 7
Failed: 0
Checking optional spsManuf tests...passed

spsManuf Test Passed
```

7.6.1 Examples for manufacturing flow SpsManuf options

In order to use manufacturing flow SpsManuf options platform has to operate in manufacturing mode.

`-fpfdisable` option disables one of the security features on the platform: Boot Guard (btg), Platform Trusted Technology (ptt), Flash Descriptor Region Verification (fdv), OEM Defined CPU Debug Policy (cpudebug), and Innovative Engine (ie).

`-seteom` option switches the platform to End Of Manufacturing state. No more changes in FPF fuses are possible then.

Disabling Boot Guard is specific because it requires platform global reset to take effect:

> run: `spsManufWin64.exe -fpfdisable btg`

```
Warning: To apply changes global reset is required
```

> perform platform global reset

For the other security features (ptt, fdv, ie), e.g.:

> run: `spsManufWin64.exe -fpfdisable ptt`

```
Security features successfully disabled
```

Finalizing all the settings of the platform:

> run: `spsManufWin64.exe -seteom`

```
Sending Intel(R) ME Set End Of Manufacturing: Successful
```

7.7 Mapping Fuse Configuration FITc settings to spsManuf subTest

Table 7-3. Fuse Configuration FITc settings / spsManuf subTest table

FPF			ME File System			FITc parameter name	spsManuf subTest
File name	offset [bit]	length [bit]	File name	offset [bit]	length [bit]		
/fpf/SBValid	*0*	*1*	*/home/SCA/CpuDbg*	*0*	*1*	Cpu Debug Policy Enabled	Cpu Debug Policy Enabled
/fpf/SBValid	*0*	*1*	*N/A; When BtG FPFs burned correctly, set to 1*	*N/A*		No FITc parameter	Boot Guard FPFs Enabled
/fpf/PttEnabled	*0*	*1*	*/home/policy/skumgr/ftpm_enable*	*0*	*1*	Intel(R) PTT Supported [FPF]	Intel(R) PTT Supported [FPF]
/fpf/Fd0vEn	*0*	*1*	*/home/SCA/Fd0vDRA*	*0*	*1*	Node: Flash Descriptor Verification	Flash Descriptor Verification Enabled
/fpf/Enf0	*0*	*1*	*/home/secureboot/enfpolicy*	*0*	*1*	Part of Boot Guard Profile Configuration	Error Enforcement Policy 0
/fpf/Enf1	*0*	*1*		*1*	*1*		Error Enforcement Policy 1
/fpf/SbPolicies	*0*	*4*	*/home/secureboot/bootpolres*	*0*	*4*	Part of Boot Guard Profile Configuration	*Boot Guard Policy Restrictions -> [0] - Force Boot Guard ACM Boot Policy*
						CPU debugging	*Boot Guard Policy Restrictions -> [1] - CPU debugging capability probe mode*
						BSP Initialization	*Boot Guard Policy Restrictions -> [2] - Determines BSP behavior*
						Part of Boot Guard Profile Configuration	*Boot Guard Policy Restrictions -> [3] - Protect BIOS Environment*
	4	*2*	*/home/secureboot/bootpoltype*	*0*	*2*	Part of Boot Guard Profile Configuration	Boot Guard Policy Type
	6	*4*	*/home/secureboo*	*2*	*4*	Key Manifest ID	Key Manifest ID

			ot/kmid				
/fpf/OemCred	*0*	*256*	*/home/secureboot/pubkeyhash*	*0*	*256*	OEM Public Key Hash	OEM Public Key Hash
/fpf/Fd0vHash	*0*	*128*	*/home/SCA/Fd0vHPK*	*0*	*128*	Fd Public Key Hash	Fd Public Key Hash
/fpf/VBEn */fpf/VBEnNT* */fpf/VBEnCp*	*0*	*1*	*N/A: FPF is set to 1 if IEVBKey is non zero and 0 otherwise*	*N/A*		Node: IE Verified Boot	IE Verified Boot Enabled*
/fpf/VBHsh */fpf/VBHshNT* */fpf/VBHshCp*	*0*	*256*	*/home/ieoem/IEVBKey*	*0*	*256*	IE -> Verified Boot Key Hash.	IE Verified Boot Hash*
/fpf/Hsh */fpf/HshNT* */fpf/HshCp*	*0*	*256*	*/home/ieoem/IEOEMKey*	*0*	*256*	IE -> IEOEMKeyHash	IE OEM Key Hash*

*There is only test, spsMANUF will read all three of FPFs

7.8 BootGuard Profile

In order to read the Boot Guard Profile value from fuse using spsManuf utility, 3 fuses need to be read back:

Boot Guard Policy Restrictions,

Boot Guard Policy Type,

Error Enforcement Policy.

The correlation of these variables in determining the Boot Guard Profile is as shown in Table 6-3.

Table 7-4. Correlations in determining the Boot Guard Profile

FITc setting	Boot Guard Policy Restricions*		Boot Guard Policy Type**		Error Enforcement Policy***	
BootGuard Profile	[0] Force Anchor Cove Boot	[3] - Protect BIOS Environment	Verified	Measured	Error Enforcement Policy 0	Error Enforcement Policy 1
0 (No_FVME)	0	0	0	0	0	0
3 (VM)	0	1	1	1	0	0
4 (FVE)	1	1	1	0	1	1

5 (FVME)	1	1	1	1	1	1
6 (FV)	1	1	1	0	0	0

* Force Anchor Cove Boot is bit 0 of Boot Guard Policy Restrictions

** Boot Guard Policy Type is a 2 bit field

*** Error Enforcement Policy is a 2 bit field

Please note that spsManuf doesn't return Boot Guard Profile set in spsFITc, but it's possible to retrieve all 4 FPFs needed to determine mentioned profile by running 4 spsManuf subtests (Table 6-4).

Table 7-5. SpsManuf tests which are required to retrieve 4 FPFs

	spsManuf subtest	FPF file path	Offset*	Length*
1	Boot Guard Policy Restrictions	/fpf/SbPolicies	0	4
2	Boot Guard Policy Type	/fpf/SbPolicies	4	2
3	Error Enforcement Policy (bit 0)	/fpf/Enf0	0	1
4	Error Enforcement Policy (bit 1)	/fpf/Enf1	0	1

* Offset (from start of FPF file) and length are in bits (no**t** bytes).

8 spsInfo and spsInfoWin

spsInfoWin and spsInfo provide a simple test to check whether the Intel ME FW is alive or not. Both tools perform the same test, query the Intel ME FW

The Windows version of spsInfo (spsInfoWin) requires administrator privileges to run under Windows OS. You must use the Run as Administrator option to open the CLI in Windows* Vista 64/32 bit and Windows* 7 64/32 bit.

8.1 Usage

In UEFI option -? not work (rest options of help work correctly)

The EFI and Linux version of the tool can be operated using the same syntax as the Windows version. The Windows version of the tool can be executed by:

```
spsInfoWin64.exe [-EXP] [-H|?] [-FWSTS] [-VER] [-VERBOSE] [-PAGE] [-
PCHBUSID]
```

It is possible to use "/" instead of "-" in command line.

Table 8-1. Command Line Options for spsInfo

Option	Description
No option	Display all information about Intel ME FW.
–VERBOSE <file>	Display the debug information of the tool or store it in a log file.
–PAGE	When more than one full screen (80 x 25 under DOS, various under Windows depending on console windows setting for the visible windows size) of information is displayed, this option allows user to pause the output and press any key before continuing on to the next screen.
–VER	Show the version of the tool.
–H or –?	Display help screen.
–EXP	Show the examples on how to use the tool.
–PCHBUSID <pchBusId>	Select PCH by PCI Bus Id Note: This option applies only for multi-PCH system. Without this option by default PCI Bus Id is 0. To select PCH connected to another PCI bus you need to know to which PCI bus Id the PCH is attached. ***Note***: Tool doesn't provide scan functionality.
–FWSTS 0x... [0x...]	Decode given hex like ME Firmware status register. It is acceptable to type only first register to decode, second one is optional.

8.2 Examples

>spsInfoWin64.exe

```
Intel(R) spsInfo Version: 4.2.97.135
Copyright(C) 2005 - 2018, Intel Corporation. All rights reserved.

FW Status Register 1: 0x00000345
  CurrentState (3:0):                       Normal (5)
  ManufacturingMode (4):                    Disabled (0)
  FlashPartition (5):                       Valid (0)
  OperationalState (8:6):                   M0 with no UMA (5)
  InitComplete (9):                         Complete (1)
  BUPLoadState (10):                        Success (0)
  FwUpdateInProgress (11):                  No (0)
  ErrorCode (15:12):                        No Error (0)
  ModeOfOperation (19:16):                  Normal (0)
  MeResetCount (23:20):                     0
  FlashDescriptorVerificationStatus (24): Verification failed (0)
  OEMDefinedCPUDebugPolicyStatus (25):    CPU Debug capability enabled (0)
  FIASKULimitViolationStatus (26):        SKU Limit not Violated (0)
  CurrentBIOSRegion (27):                   Primary BIOS region (0)
  D0I3_SUPPORT (31):                        No (0)

FW Status Register 2: 0x308A4000
  CPU Replacement Valid (0):                No (0)
  ICC programmed successfully (1):          No (0)
  ICC: valid data read from SPI (2):        No (0)
  Restricted Mode (3):                      Disabled (0)
  CPU Replacement (4):                      Not detected (0)
  Chipset Hard Fused (5):                   False (0)
  MfsFailure (6):                           No Mfs failure (0)
  WarmReset (7):                            No warm reset request (0)
  EndOfPOST (11):                           Not Received (0)
  TargetImageBoot (12):                     Success (0)
  Heartbeat (15:13):                        2
  ExtendedStatusData (23:16):               8Ah
  PM Event (27:24):                         Clean CMoff->CMx wake (0h)
  Phase (31:28):                            BUP (3)

FW Status Register 4: 0x00004000
  Flash Log Exists (2):                     No (0)
  dTPM 1.2 Deactivated (8):                 No (0)
  Enforcement Flow (9):                     ME is not in BtG enforcement
flow (0)
  Sx Resume Type (10):                      S5 or G3 (0)
  S3 Optimization Disabled (11):            No (0)
  All TPMs Disconnected (12):               Disconnection flow not executed
(0)
  HAP Type (13):                            0
  Boot Guard FWSTS Valid (14):              Valid (1)
  Boot Guard Self-Test Failed (15):         No (0)
  Boot Guard FPF error (17):                Success (0)
  Token Applied (18):                       No (0)
  CPU co-signing (21):                      Disabled (0)
  CPU co-signing error encountered (22):  No (0)

FW Status Register 6: 0x00400000
  Force BtG ACM Boot Policy (0):            Legacy BIOS boot vector (0)
  CPU Debug Disabled (1):                   False (0)
  BSP Initialization Disabled (2):          False (0)
  Protect BIOS Environment Policy (3):      Take no action (0)
  Bypass Boot Policy (4):                   No (0)
```

```
  Boot Policy Invalid (5):                 No (0)
  Error Enforcement Policy (7:6):          Not shut down (0)
  Measured Boot Policy (8):                Disable (0)
  Verified Boot Policy (9):                Disable (0)
  Boot Guard ACM SVN (13:10):              0
  Key Manifest SVN (17:14):                0
  Boot Policy Manifest SVN (21:18):        0
  Key Manifest ID (25:22):                 1
  BSP Boot Policy Manifest Exec Sts (26): Execution on BSP not completed
(0)
  Error (27):                              No (0)
  Boot Guard (28):                         FW Supported (0)
  Flash Programmable Fuses (29):           Enabled (0)
  Config lock (30):                        No (0)
  TXT Support (31):                        No (0)

No PTU Option ROM detected in DER region.
* CPU replacement is displayed only on Mehlow-Refresh/Jacobsville/Idaville
  CPU co-signing is displayed only on Jacobsville/Idaville.
```

>spsInfoWin64.exe -FWSTS 0x001F0347 0xB9006101

```
Intel(R) spsInfo Version: 4.2.97.135
Copyright(C) 2005 - 2015, Intel Corporation. All rights reserved.

FW Status Register 1: 0x001F0347
  CurrentState (3:0):                      State transition (7)
  ManufacturingMode (4):                   Disabled (0)
  FlashPartition (5):                      Valid (0)
  OperationalState (8:6):                  M0 with no UMA (5)
  InitComplete (9):                        Complete (1)
  BUPLoadState (10):                       Success (0)
  FwUpdateInProgress (11):                 No (0)
  ErrorCode (15:12):                       No Error (0)
  ModeOfOperation (19:16):                 Unknown (15)
  MeResetCount (23:20):                    1
  FlashDescriptorVerificationStatus (24): Verification failed (0)
  OEMDefinedCPUDebugPolicyStatus (25):     CPU Debug capability enabled (0)
  FIASKULimitViolationStatus (26):         SKU Limit not Violated (0)
  D0I3_SUPPORT (31):                       No (0)

FW Status Register 2: 0xB9006101
  ICC programmed successfully (1):         No (0)
  ICC: valid data read from SPI (2):       No (0)
  Restricted Mode (3):                     Disabled (0)
  Chipset Hard Fused (5):                  False (0)
  MfsFailure (6):                          No Mfs failure (0)
  WarmReset (7):                           No warm reset request (0)
  EndOfPOST (11):                          Not Received (0)
  TargetImageBoot (12):                    Success (0)
  Heartbeat (15:13):                       3
  ExtendedStatusData (23:16):              0h
  PM Event (27:24):                        Non-power cycle reset (9h)
  Phase (31:28):                           Unknown (11)
```

>spsInfoWin64.exe -FWSTS 0x001F0345

```
Intel(R) spsInfo Version: 4.2.97.135
Copyright(C) 2005 - 2018, Intel Corporation. All rights reserved.

FW Status Register 1: 0x001F0345
  CurrentState (3:0):                       Normal (5)
  ManufacturingMode (4):                    Disabled (0)
  FlashPartition (5):                       Valid (0)
  OperationalState (8:6):                   M0 with no UMA (5)
  InitComplete (9):                         Complete (1)
  BUPLoadState (10):                        Success (0)
  FwUpdateInProgress (11):                  No (0)
  ErrorCode (15:12):                        No Error (0)
  ModeOfOperation (19:16):                  Unknown (15)
  MeResetCount (23:20):                     1
  FlashDescriptorVerificationStatus (24): Verification failed (0)
  OEMDefinedCPUDebugPolicyStatus (25):    CPU Debug capability enabled (0)
  FIASKULimitViolationStatus (26):        SKU Limit not Violated (0)
  D0I3_SUPPORT (31):                        No (0)
```

Tool Detail Error Code

Common Error Code for all Tools

Error Code	Error Message	Response
0	Success	
1	Memory allocation error occurred	Make sure there is enough memory in the system
2	Invalid descriptor region	Check descriptor region
3	Region does not exist	Check region to be programmed
4	Failure. Unexpected error occurred	Contact Intel
5	Invalid data for Read ID command	Contact Intel
6	Error occurred while communicating with SPI device	Check SPI device
7	Hardware sequencing failed. Make sure that you have access to target flash area	Check descriptor region access settings
8	Software sequencing failed. Make sure that you have access to target flash area	Check descriptor region access settings
9	Unrecognized value in the HSFSTS register	Unrecognized value in the HSFSTS register
10	Hardware Timeout occurred in SPI device	Hardware Timeout occurred in SPI device
11	AEL is not equal to zero	AEL is not equal to zero
12	FCERR is not equal to zero	FCERR is not equal to zero
25	The host CPU does not have write access to the target flash area. To enable write access for this operation you must modify the descriptor settings to give host access to this region.	Check descriptor region access settings
26	The host CPU does not have read access to the target flash area. To enable read access for this operation you must modify the descriptor settings to give host access to this region.	Check descriptor region access settings
27	The host CPU does not have erase access to the target flash area. To enable erase access for this operation you must modify the descriptor settings to give host access to this region.	Check descriptor region access settings
28	Protected Range Registers are currently set by BIOS, preventing flash access. Contact the target system BIOS vendor for an option to disable Protected Range Registers.	Assert Flash Descriptor Override Strap (GPIO33) to Low, Power Cycle, and Retry. If Protected Range Registers (memory location: SPIBAR + 74h -> 8Fh) are still set, contact the target BIOS vendor.
50	General Erase failure	Attempt the command again. If it fails again, contact Intel.

Error Code	Error Message	Response
51	An attempt was made to read beyond the end of flash memory	Check address
52	An attempt was made to write beyond the end of flash memory	Check address
53	An attempt was made to erase beyond the end of flash memory	Check address
54	The address <address> of the block to erase is not aligned correctly	Check address
55	Internal Error	Contact Intel
56	The supplied zero-based index of the SPI Device is out of range.	The supplied zero-based index of the SPI Device is out of range.
57	AEL or FCERR is not equal to zero for Software Sequencing	AEL or FCERR is not equal to zero for Software Sequencing
75	Common VSCC file not found	Check file location
76	Access was denied opening the file	Check file location
77	An unknown error occurred while opening the file	Verify the file is not corrupt
78	Failed to allocate memory for the flash part definition file	Check system memory Verify the file is not corrupt
79	Failed to read the entire file into memory	Check system memory Verify the file is not corrupt
80	Parsing of file failed	Check system memory Verify the file is not corrupt
100	The SPI Flash configuration registers are write protected by the Flash Configuration Lock-Down bit (FLOCKDN). Cannot access the SPI flash. Contact your BIOS vendor to unlock this bit or enable hardware sequencing in descriptor mode.	Check with BIOS vendor or SPI programming Guide
101	No SPI flash device could be identified. Please verify if Fparts.txt has support for this part	Verify **Fparts.txt** contains device supported.
102	Failed to read the device ID from the SPI flash part	Verify **Fparts.txt** has correct values
103	There are no supported SPI flash devices installed. Check connectivity and orientation of SPI flash device	Verify **Fparts.txt** has correct values. Check SPI Device
104	The two SPI flash devices do not have compatible command sets	Verify both SPI devices on the system are compatible
105	An error occurred while writing to the write status register of the SPI flash device. This program will not be able to modify the SPI flash	Check SPI Device
8196	HECI message receive buffer memory allocation failed	
8193	Intel® ME Interface: Cannot locate Intel® ME device driver	

Error Code	Error Message	Response
8199	Could not issue %s command message Where %s can be the following: Get FWU Version Get FWU Info Get FWU Feature State Intel ME Kernel Test	Contact Intel
8203	Unexpected result in %s command response Where %s can be the following: Get FWU Version Get FWU Info Get FWU Feature State Intel ME Kernel Test	Contact Intel
8204	Intel ME Interface: Unsupported message type	
8213	Requesting HECI receive buffer size is too small	
9489	Couldn't receive Intel(R) MEI get FW features message response	
9507	Fail to load driver (PCI access for Windows).Tool needs to run with an administrator privilege account	

spsManuf Errors

Error Codes	Error Messages
9458	Communication error between application and Intel(R) ME module (FW Update client)
9459	Internal error (Could not determine FW features information)
9487	Couldn't issue Intel(R) MEI get FW version message (0x%X)
9488	Couldn't receive Intel(R) MEI get FW version message response (0x%X)
9489	Couldn't receive Intel(R) MEI get FW features message response
9500	spsManuf Test Failed
9501	Unsupported command line option(s)
9502	Unknown or unsupported hardware platform
9503	Configuration file %s is missing
9504	spsManuf config file generation failed
9505	Intel(R) Fail to read FW Status Register value 0x%X
9506	Fail to create verbose log file %s
9507	Fail to load driver (PCI access for Windows).Tool needs to run with an administrator priviledge account.
9508	Configuration file syntax corrupted

Error Codes	Error Messages
9510	Intel(R) ME FW invalid status
9511	Intel(R) Bad checksum of Flash Partition Table or broken factory defaults
9512	Intel(R) Failure in starting desired ME FW image
9520	Failure getting SPI address and/or loading VSCC file
9521	Single flash part found, Flash Partition Boundary Address must be zero
9522	Flash Partition Boundary Address should be on the boundary between flash parts
9523	The two flash parts on this platform require different BIOS VSCC values
9524	Access flash device failure
9525	Fail to establish a communication with SPI flash interface
9526	Fail to load vsccommn.bin
9527	Flash ID 0x%06X Intel(R) %s VSCC value mismatch
9528	No recommended %s VSSCC value found for Flash ID 0x%06X
9530	ME FW version is incorrect
9531	ME recovery version is incorrect
9532	Backup ME FW version is incorrect
9533	No backup image or single image configuration
9540	Intel(R) ME-BIOS Interface Version mismatch
9541	Intel(R) %s error
9542	Intel(R) %s status mismatch
9550	Intel(R) ME internal communication error (FW)
9551	Error: %s Factory Default Configuration status failed
9553	PTU OROM version mismatch, actual value is - %d.%d
9554	No PTU Option ROM detected in DER region.
9555	Invalid PTU OROM version length, actual value is - %d.%d
9556	FW not in SPS mode of operation
9557	Intel(R) ME Integrity Check reading status failed
9558	Intel(R) ME Integrity Check calculation status failed
9559	Intel(R) ME Integrity Check mismatch
9560	Error: %s BMC Connection status failed
9561	Intel(R) Incorrect ME Address for BMC Connection test
9562	Intel(R) BMC Connection test can not be run under this configuration
9570	Intel(R) Read flash master region permission failure
9571	Intel(R) Incorrect format of expected access permission

Error Codes	Error Messages
9572	Intel(R) Incorrect Access Rights
9573	Intel(R) Correct vsccommn.bin file was not found
9580	Intel(R) Not existing or invalid region
9581	Intel(R) ME Region Definition address mismatch
9582	Intel(R) ME Region Definition length mismatch
9590	Error: %s End-Of-Post status failed
9591	Error: %s BIOS VSCC failed
9628	Memory allocation error occurred.
9629	The host CPU does not have read access to the target flash area. To enable read access for this operation you must modify the descriptor settings to give host access to this region.
9630	Protected Range Registers are currently set by BIOS, preventing flash access. Please contact the target system BIOS vendor for an option to disable Protected Range Registers.
9631	Hardware timeout occurred in SPI device.
9632	An attempt was made to read beyond the end of flash memory
9633	General Read failure.
9657	Command rejected because Intel(R) ME FW is already in End of Manufacturing state
9674	Sending Intel(R) ME Set End Of Manufacturing failed, response status: 0x%X
9675	Intel(R) ME Disable security features unknown error
9676	Sending Intel(R) ME Disable security features failed, response status: 0x%X
9677	Sending Intel(R) ME Read File failed
9678	File "%s" inconsistent with provided data
9679	%s is not supported on current platform
9680	Platform's Server Segment unknown
9685	Chipset fusing check error
9686	End Of Manufacturing state check error
9687	All IE OEM key hash files empty or with null content
9688	No matching IE OEM key hash found
9689	All IE Verified Boot key hash files empty or with null content
9690	No matching IE Verified Boot key hash found
9691	All IE Verified Boot Enabled files empty or with null content
9692	No matching IE Verified Boot Enabled found
9693	Co-signing FPF consitency check failed
9694	Invalid format of expected data string

Error Codes	Error Messages
9695	Failed to set End of Manufacturing mode

spsInfo Errors

Error Code	Error Messages
9253	Firmware did not return a valid value for iTPM full self-test
9258	TPM parsing response problem, response is less than minimum required
9259	TPM parsing response problem, bad tag value
9260	TPM parsing response problem, bad param size
9269	Zero flash device found for VSCC check
9271	Incorrect VSCC table entry mismatch
9272	No VSCC table entry found
9279	SPI flash Intel(R) ME region is not locked
9280	Intel(R) Gbe/ME has read or write access to BIOS region
9281	SPI flash descriptor region is not locked
9282	BIOS has granted Intel(R) Gbe and/or ME access to its region
9283	Region access permissions don't match Intel recommended values
9284	Tool fails to retrieve setting information
9289	Couldn't issue Intel(R) MEI get event log message
9290	Couldn't receive Intel(R) MEI get event log message response
9293	Create Context in Vista OS failed
9299	Single flash part found, Flash Partition Boundary Address isn't zero
9300	Flash Partition Boundary Address should be in between flash parts
9301	Two flash parts require different BIOS VSCC values
9303	Checking variable "%s" memory allocation failed
9304	Getting variable "%s" failed or not found
9306	System UUID status failed
9307	MAC address status failed
9308	Security Descriptor Override status failed
9310	ME Manufacturing Mode status failed
9311	CF9GR locking status failed
9451	Communication error between application and Intel(R) AMT module (PTHI client)
9452	Communication error between application and Intel(R) ME module (iCLS client)

Error Code	Error Messages
9455	Failed to read FW Status Register value 0x%X
9457	Failed to create verbose log file %s: Where %s is the log file name user specified
9458	Communication error between application and Intel® ME module (FW Update client)
9459	Internal error (Could not determine FW features information)
9460	Cannot locate hardware platform identification This program cannot be run on the current platform. Unknown or unsupported hardware platform Or A %s hardware platform is detected This program cannot be run on the current platform. Unknown or unsupported hardware platform Where %s is the official name of the hardware platform
9467	Cannot use zero as SPI Flash ID index number
9468	Couldn't find a matching SPI Flash ID
9469	Access to SPI Flash device(s) failed
9471	%s feature was not found
9472	Parameter invalid
9473	Parameter not equal
9474	Internal error
9475	Version feature was not available
9476	Feature was not available
9487	Couldn't issue Intel(R) MEI get FW version message (0x%X)
9488	Couldn't receive Intel(R) MEI get FW version message response (0x%X)
9489	Couldn't receive Intel(R) MEI get FW features message response
9502	Unknown or unsupported hardware platform
9505	Intel(R) Fail to read FW Status Register value 0x%X
9506	Fail to create verbose log file %s

spsFPT Errors

Error Code	Error	Response
1	Memory allocation error occurred	Make sure there is enough memory in the system
106	Flash component does not support SFDP capability.	

Error Code	Error	Response
200	Invalid parameter value specified by the user. Use -? Option to see help.	Check the command line arguments supported by using the "-?"
201	spsFPT.exe cannot be run on the current platform. Please contact your vendor.	Contact your vendor.
202	Confirmation is not received from the user who performed the operation.	User input required
203	Flash is not blank. Data <data> found at address <address>.	Attempt to erase the device again
204	Data verify mismatch found at address <address>.	Reprogram the device
205	Failure. Unexpected error occurred	File a sighting
206		PDR region exists
207	Invalid parameter value specified by user. The option specified cannot be run on a platform with Intel (R) ME Ignition FW.	
210	The Intel ME Failed to reset.	
211	There was a communications error between spsFPT and the Intel ME	
212	The request to disable the Intel ME failed.	
215	The attempt to commit the FOVs has failed.	
216	The Close Manufacturing process failed.	
217	Setting Global Reset Failed	
218	Selected region is not supported on current PCH	
219	NOLAN switch cannot be used because of map layout mismatch with the new flash image. Remove this flag from command line if you want to execute a full flash update.	
240	Access was denied while opening the file <file>	Check the permissions for the file
241	Access was denied while creating the file <file>	Check the permissions for the file
242	An unknown error occurred while opening the file <file>	Verify the file is not corrupt
243	An unknown error occurred while creating <file>	Verify the file is not corrupt
244	<name> is not a valid file name.	Check the filename
245	<file> file not found	Check file location
246	Failed to read the entire file into memory. File: <file>	Check system memory. Verify the file is not corrupt
247	Failed to write the entire flash contents to file	Check system memory
248	<file> file already exists	Delete the file that already exists
249	The file is longer than the flash area to write	Check file size

Error Code	Error	Response
250	The file is smaller than the flash area to write	Check file size
251	Length of image file extends past the flash area	Check file size
252	Image file <file> not found	Check filename
253	<file> file does not exist	Check filename
254	Not able to open the file <file>	Check filename
255	Error occurred while reading the file <file>.	Check filename
256	Error occurred while writing to the file <file>	Check filename
280	Failed to disable write protection for the BIOS space!	Verify BIOS does not have write protection enabled
281	Device cannot respond to Memory Space accesses."	
282	Failed to get information about the installed flash devices	Check descriptor region access settings
283	Unable to write data to flash. Address <address>.	Check descriptor region access settings
284	Failed to load driver (PCI access for Windows). Tool needs to run with an administrator privilege account.	
320	General Read failure	Attempt the command again. If symptom persists file a sighting
321	The address <address> is outside the boundaries of flash area	Check address
360	Invalid Block Erase Size value in <file>.	Check **fparts.txt** or its equivalent file
361	Invalid Write Granularity value in <file>	Check **fparts.txt** or its equivalent file
362	Invalid Enable Write Status Register Command value in <file>	Check fparts.txt or its equivalent file
363	Invalid Chip Erase Timeout value in <file>	Check fparts.txt or its equivalent file
400	Flash descriptor does not have correct signature	Verify file is not corrupt
401	An error occurred reading the flash mapping data	Check SPI device
402	An error occurred while reading the flash components data	Check SPI device
403	An error occurred while reading the flash region base/limit data	Check SPI device
404	An error occurred while reading the flash master access data	Check SPI device
405	An error occurred while reading the flash descriptor signature	Check SPI device
406	System booted in Non-Descriptor mode, but the flash appears to contain a valid signature	Check SPI device
407	User-provided Chip Erase Timeout has been reached. If the timeout value was set incorrectly the chip erase may still occur.	Check fparts.txt or its equivalent file

Error Code	Error	Response
440	Invalid Fixed Offset variable name	
441	Invalid Fixed Offset variable Id	
442	Param file is already opened.	
444	Invalid name or Id of FOV.	
445	Invalid length of FOV value. Check FOV configuration file for correct length.	
446	Password does not match the criteria.	
447	Error occurred while reading FOV configuration file	
448	Invalid hash certificate file	
449	Valid PID/PPS/Password records are not found	
450	Invalid ME Manufacturing Mode Done value entered.	
451	Unable to get master base address from the descriptor	Check file integrity
452	Verification of End Of Manufacturing settings failed	
453	End Of Manufacturing Operation failure - Verification failure on ME Manufacturing Mode Done settings.	
454	The Global Lock Bit has already been set.	
455	End Of Manufacturing Operation failure - Verification failure on Intel ME Manuf counter.	
456	End Of Manufacturing Operation failure - Verification failure on Descriptor Lock set	
457	Parsing of file <file> failed	
459	There is a problem with the GbE binary which prevents saving the data	
480	The setup file header has an illegal UUID	
481	The setup file version is unsupported	Check setup file integrity
482	A record encountered that does not contain an entry with the Current MEBx password	
483	The given buffer length is invalid	Check buffer length value
484	The record chunk count cannot contain all of the setup file record data	Setup file number exceeded
485	The setup file header indicates that there are no valid records	Setup file has no valid records. Check setup file integrity
486	The given buffer is invalid	Check buffer value
487	A record entry with an invalid Module ID was encountered	Check record values. Check Setup file integrity
488	A record was encountered with an invalid record number	Check record values. Check Setup file integrity

Error Code	Error	Response
489	The setup file header contains an invalid module ID list	Check record values. Check Setup file integrity
490	The setup file header contains an invalid byte count	Check record values. Check Setup file integrity
491	The setup file record ID is not RECORD_IDENTIFIER_DATA_RECORD	Check record values. Check Setup file integrity
492	The list of data record entries is invalid	Check record values. Check Setup file integrity
493	The CurrentMEBx password is invalid	
494	The NewMEBx password is invalid	
495	The PID is invalid	
496	The PPS is invalid	
497	The PID checksum failed	
498	The PPS checksum failed	
499	The data record is missing a CurrentMEBx password entry	
500	The data record is missing a NewMEBx password entry	
501	The data record is missing a PID entry.	
502	The data record is missing a PPS entry.	
503	The file <file> has an invalid entry	
504	The requested index is invalid	
505	Failed to write to the given file	
506	Failed to read from the given file	
507	Failed to create random numbers	
508	The data record is missing a PKI DNS Suffix entry	
509	The data record is missing a Config Server FQDN entry	
510	The data record is missing a ZTC entry	
511	The data record is missing a Pre-Installed Certificate enabled entry	
512	The data record is missing a User defined certificate config entry	
513	The data record is missing a User defined certificate Add entry	
514	The data record is missing a SOL/IDER enable entry	
515	OEM Firmware Update Qualifier data missing in USB file	
516	The file "%s" has an invalid entry	
517	User selected to cancel the operation	

Error Code	Error	Response
522	Failed getting variable "%s" value	
523	Failed comparing variable "%s" value	
525	Failed to perform ME Reset	
1000	Invalid command line option(s)	
1001	Unsupported OS	
1002	Failed to retrieve Intel (R) ME FW Version	

MESDC Commands

Command	Name	Fields	Length [bytes]	Value	Recovery	SiEn	Full	SMBHost	HECI	IPMB
00h	Get Version	**Request:**			Y	Y	Y	Y	Y	Y
		Protocol Version Major	1	Major Version of protocol used by Console Application. Currently there is only one version supported – 1. Versions with different major version are not compatible.						
		Protocol Version Minor	1	Minor Version of protocol used by Console Application. If versions are different it is assumed that only a subset of commands recognized by party with lower version minor can be used.						
		Response:								
		Protocol Version Major	1	Major Version of protocol used by Console Application. Currently there is only one version supported – 1. Versions with different major version are not compatible.						
		Protocol Version Minor	1	Minor Version of protocol used by Console Application. If versions are different it is assumed that only a subset of commands recognized by party with lower version minor can be used.						
		Firmware Status	4	Current FW status						
		Extended Firmware Status	4	Additional information about FW Status						
		Uptime	4	System uptime						
		Command Status		STATUS_SUCCESS. This command should always return status STATUS_SUCCESS						
01h	Send Raw IPMI	**Request:**			N	Y	Y	N	N	N
		Net Function/Lun	1	[7:6] - Network Function code [1:0] - LUN						
		IPMI Command	1	Command byte						
		Payload Length	1	IPMI payload length						
		Payload	N	IPMI data						
		Response:								
		IPMI Response Data	1	IPMI Completion code						
			M - 1	IPMI response data						
		Command Status		=STATUS_SUCCESS - Diagnostic test has						

				been processed successfuly =STATUS_SIZE_ERROR - received IPMI message is longer than 80B =STATUS_NO_EVENTS - IPMI response didn't arrive within specified time (2000 ms)" =STATUS_RESOURCE_BUSY - there is no resorces for a new command, previous command is still processed or other error codes						
03h	**Access SMBus**	**Request:**			N	Y	Y	Y	N	Y
		SMB Command	1	Smbus Command						
		Flags	1	Smbus Command Flags						
		Slave Address	4	Logical address in Grantley, In Denlow 1 byte slave address						
		Data	N	Writing data.						
		Response:								
		Data	N	Reading data.						
		Command Status		STATUS_SUCCESS, STATUS_NOT_FOUND, STATUS_FAILURE or other error code						
1Fh	**GetManufacturingInfo (GetManufacturingStatus)**	**Request:**								
		RequestCode	1	00h - BasicStatus , 01h…0FFh - for future purposes						
		Respond:								
			4	LSB first (little endian) format. BIT[0]: ReadUnlock status; 0 - not active, 1 - active (set) BIT[1] - EOM (EndOfManufacturing) state; 0 - Manufacturing State is active (Platform is in Manufacturing Mode), 1 - Manufacturing State is inactive (End Of Manufacturing Mode) BIT[2]...BIT[31] should be set to 0 (for future purposes). In current implementation these bits could no be validated by the tools.						
		Command Status		00h - Status OK, FFh - Staus Failure						
30h	**Send Raw PECI**	**Request:**			N	Y	Y	Y	N	Y
		CPU Index	1	[7:6] - reserved [5:0] - PECI Client Address (values shall be in the range from 0x30 through 0x37).						

		PECI Interface Selection	1	[7:5] – PECI Interface selection: 000b – ME will send the PECI request using in-band PECI interface. If in-band PECI is not functional (not configured by BIOS or not working due to failures), ME will use serial PECI interface when connected directly to chipset. 001b – ME will send the PECI request using in-band PECI. ME will not try the serial PECI interface 010b – ME will send the PECI request serial PECI interface, if the interface is connected directly to the chipset. 100b – ME will send the PECI request using in-band PECI interface. If in-band PECI is not functional (not configured by BIOS or not working due to failures), ME will use serial PECI interface when connected to BMC a.k.a. Reverse PECI Proxy Path. 110b – ME will send the PECI request serial PECI interface, if the interface is connected to BMC - a.k.a Reverse PECI Proxy path 101b - Reserved [4:0] – Reserved – Not used **NOTE: This byte is not present in Denlow**
		Write Length	1	Write Length (part of PECI standard header); this field shall be set to the proper value for this PECI command as if there was AWFCS byte provided
		Read Length	1	Read Length (part of PECI standard header); this field shall be set to the proper value for this PECI command
		PECI Write Data	N	The remaining part of PECI command following the Read Length field (if any – this field does not exist for PECI Ping command); onlywrite data bytes shall be put here, excluding AWFCS bytes (AWFCS will be added by ME FW); note that the retry bit shall normally be set to zero and the command code byte shall be one of the codes understood by ME FW (0x01, 0xF7, 0xA1, 0xA5, 0xB1, 0xB5, 0xC1, 0xC5, 0xE1, 0xE5; note that only Domain 0 codes are supported)
		Response:		
		PECI Read Data	N	PECI response data (if any – no data is returned for Ping command or for Completion Code in Byte#1 other than 00h); data following the Write FSC field are put here exactly as received from PECI client during Read transaction phase, excluding the Write FCS and Read FCS bytes

		Command Status		= 00h – PECI response successfully retuned (see PECI response completion code for detailed response from PECI client) = A4h – Bad Read FSC in the response (even after the retry) = A5h – Bad Write FCS field in the response (even after the retry); this error code is also returned in case of Abort FSC in the response (as defined in PECI spec) and no response from PECI client (client device is not responding at all) = A6h – bad Write Length in the request = A7h – bad Read Length in the request = A8h - Selected PECI interface not available = ABh – command code in the request not understood by ME FW						
33h	**Get Image Version**	**Request:**			Y	Y	Y	Y	Y	Y
		Image	1							
		Response:								
		Major Version	2							
		Minor Version	2							
		Hotfix	2							
		Build	2							
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
34h	**Switch to Image**	**Request:**			Y	Y	Y	Y	N	Y
		Image	1							
		Response:								
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
35h	**Get Current Image**	**Request:**			Y	Y	Y	Y	Y	Y
		Response:								
		Image Number	1							
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
37h	**File Directory Get First**	**Request:**			N	Y	Y	Y	Y	Y
		Path	32	Example: "/home", "/home/mesdc"						
		Response:								
		Name	13	Null terminated file name string						
		Attributes	4	See Parameter definition: Ref R2 at bottom of this document						
		Enumeration Context	1	Next enumeration to get (0..n) and is set by FW and referenced by the interface						
		Length	2	Size of file						
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						

38h	**File Directory Get Next**	**Request:**			N	Y	Y	Y	Y	Y
		Path	32	Path and name of file: "/home/mesdc/heci_cb_en"						
		Enumeration context	1							
		Response:								
		Name	13	Null terminated file name string						
		Attributes	4	See Parameter definition: Ref R2 at bottom of this document						
		Enumeration Context	1	Next enumeration to get (0..n) and is set by FW and referenced by the interface						
		Length	2	Size of file						
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
39h	**File contents Get**	**Request:**			N	Y	Y	Y	Y	Y
		File path	32	Path and name of file: "/home/mesdc/heci_cb_en"						
		File offset	2	Byte offset into file						
		Read length	2	Amount of data to read						
		Response:								
		File path	32	Path and name of file: "/home/mesdc/heci_cb_en"						
		File attributes	4	See Parameter definition: Ref R2 at bottom of this document						
		Actual bytes returned of file contents	2	Bytes read						
		File contents	n	File data						
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
3Dh	**Get Encrypted File Contents**	**Request:**						N	N	N
		See command 0x39, File Content Get								
		Response:								
40h	**Get MCTP Statistics**	**Request:**			N	Y	Y	Y	Y	Y
		MCTP port number	1	0x00 - PCIe - currently only available, 0x01-0xFF - reserved for future use						
		Command types to be reported	1	0x00 - control commands, 0x01-0xFF - reserved for future use; this field is ignored for MCTP proxy!						
		Response:								
		Common fields:								
		MCTP mode	1	0x00 - MCTP normal stack (ME as bus owner), 0x01 - Bus owner proxy mode, 0x02-0xFF - reserved for future use						
		Fields available for MCTP stack (MCTP mode == 0):								
		ME/Bus owner EID	4	Bus owner EID: 0x01-0xFE, 0x00 and 0xFF are illegal, 0x100 - 0xFFFFFFFF - reserved for future use						

		Number of endpoints supported	4	Number of endpoints supported by ME (in config XML), legal values: 0x00-0x10							
		Number of endpoints discovered	4	Number of endpoints discovered by bus owner - legal values <= Number of endpoint supported							
		Number of Prepare for Endpoint Discovery requests sent	4								
		Number of Prepare for Endpoint Discovery responses received	4								
		Number of Endpoint Discovery requests sent	4								
		Number of Endpoint Discovery responses received	4								
		Number of Discovery Notify requests received	4								
		Number of Discovery Notify responses sent	4								
		Number of Set EID requests sent	4								
		Number of Set EID responses received	4								
		Number of Get EID requests sent	4								
		Number of Get EID responses received	4								
		Number of Resolve EID requests received	4								
		Number of Allocate Endpoint IDs requests received	4								
		Number of control messages rejected	4								
		Fields available for BO proxy mode (MCTP mode == 1):									
		Number of BOProxy protocol control messages received	4								
		No of BOProxy protocol control msgs discarded on errors	4								
		Number of BOProxy protocol control msg responses sent	4								
		Number of BOProxy protocol encapsulated msgs received	4								
		No of BOProxy proto encaps. msgs rcvd discarded on errors	4								
		No of BOProxy proto encaps. msgs rcvd discarded no buffers	4								

		Number of messages received from endpoints	4							
		No of msgs received from endpoints discarded no buffers	4							
		Number of BOProxy protocol encapsulated messages sent	4							
		Completion codes:								
			1	STATUS_SUCCESS = 0x00 STATUS_INVALID_PARAMS = 0x85 - for Command types to be processed in reserved range STATUS_INVALID_COMMAND = 0x8D - if MCTP functionality disabled in ME or MCTP service not initialized (diagnostic command not registered) STATUS_SIZE_ERROR = 0x05 - for wrong length of diagnostic command frame STATUS_NOT_READY = 0x88 - MCTP port not initialized STATUS_GENERAL_ERROR = 0xA1 - other internal error occured							
41h	**Reset MCTP Statistics**	**Request:**			N	Y	Y	Y	N	Y
		MCTP port number	1	0x00 - PCIe - currently only available, 0x01-0xFF - reserved for future use						
		Command types to be reset	1	If MCTP runs in non-proxy mode: 0x00 - control commands, 0x01-0xFF - reserved for future use						
		Counters to be reset	4	Bit masked field - masking allows resetting single and many counter(s) in single command call						
				Bit interpretation for MCTP stack:						
				Bit 0 (0x01) - reset number of Prepare for Endpoint Discovery requests sent						
				Bit 1 (0x02) - reset number of Prepare for Endpoint Discovery responses sent						
				Bit 2 (0x04) - reset number of Endpoint Discovery requests sent						
				Bit 3 (0x08) - reset number of Endpoint Discovery responses received						
				Bit 4 (0x10) - reset number of Discovery Notify requests received						
				Bit 5 (0x20) - reset number of Discovery Notify responses sent						
				Bit 6 (0x40) - reset number of Set EID requests sent						
				Bit 7 (0x80) - reset number of Set EID responses received						
				Bit 8 (0x100) - reset number of Get EID requests sent						
				Bit 9 (0x200) - reset number of Get EID responses received						
				Bit 10 (0x400) - reset number of Resolve EID requests received						
				Bit 11 (0x800) - reset number of Allocate Endpoint IDs requests received						
				Bit 12 (0x1000) - reset number of control messages rejected						
				Bits 13-31 - reserved for future use (should be set to zeros)						

				Bit interpretation for MCTP Bus owner proxy:						
				Bit 0 (0x01) - reset number of BOProxy protocol control messages received						
				Bit 1 (0x02) - reset number of BOProxy protocol control messages discarded on errors						
				Bit 2 (0x04) - reset number of BOProxy protocol control message responses sent						
				Bit 3 (0x08) - reset number of BOProxy protocol encapsulated messages received						
				Bit 4 (0x10) - reset number of BOProxy protocol encapsulated messages rcvd discarded on errors						
				Bit 5 (0x20) - reset number of BOProxy protocol encapsulated messages rcvd discarded no buffers						
				Bit 6 (0x40) - reset number of messages received from endpoints						
				Bit 7 (0x80) - reset number of messages received from endpoints discarded no buffers						
				Bit 8 (0x100) - reset number of BOProxy protocol encapsulated messages sent						
				Bits 9-31 - reserved for future use (should be set to zeros)						
		Response:								
		Completion codes:								
			1	STATUS_SUCCESS = 0x00 STATUS_INVALID_PARAMS = 0x85 - for Command types to be processed in reserved range STATUS_INVALID_COMMAND = 0x8D - if MCTP functionality disabled in ME or MCTP service not initialized (diagnostic command not registered) STATUS_SIZE_ERROR = 0x05 - for wrong length of diagnostic command frame STATUS_GENERAL_ERROR = 0xA1 - other internal error occured STATUS_NOT_READY = 0x88 - MCTP port not initialized						
4Ch	**NM Reset Stats**	**Request:**			N	N	Y	Y	Y	Y
		Mode	1	As defined in IPMI command **"Reset Node Manager Statistics"**						
		Domain ID	1							
		Policy ID	1							
		Response:								
			0							
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						

4Dh	**NM Get Stats**	**Request:**			N	N	Y	Y	Y	Y
		Mode	1	As defined in IPMI command **"Get Node Manager Statistics"**						
		Domain ID	1							
		Policy ID	1							
		Response:								
		Current Reading	2							
		Minimum	2	Min reading value						
		Maximum	2	Max reading value						
		Average	2	Average reading value						
		Timestamp	4	As defined in IPMI command **"Get Node Manager Statistics"**						
		Statistics Reporting Period	4	As defined in IPMI command **"Get Node Manager Statistics"**						
		Domain ID / Policy State	1	As defined in IPMI command **"Get Node Manager Statistics"**						
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
50h	**NM Set Power/Throttling Target**	**Request:**			N	N	Y	Y	N	Y
		Domain ID	1	0x00 - platform; 0x01 - CPU; 0x02 - Memory						
		Action Type	1	0x00 - non-aggressive power limit; 0x01 - aggressive power limit; 0x02 - throttling level						
		Limit	2	power limit in Watts; throttling level in percentage						
		Response:								
			0							
		Command Status		= 00h – P/T Limit was set successfully = 81h – Invalid Domain ID = 83h – Invalid Action Type = 84h – Invalid P/T Limit						
51h	**Get Current PMC Patch Info**	**Request:**			Y	Y	Y	Y	Y	Y
			0							
		Response:								
		PMC Patch Product ID	1	ID of the matherboard for which patch was prepared.						
		PMC Patch Rev ID Min	1	Minimum version of the PMC revision for which patch was prepared.						
		PMC Patch Rev ID Max	1	Maximum version of the PMC revision for which patch was prepared.						
		PMC Patch ROM ID Min	1	Minimum version of the ROM for which patch was prepared.						
		PMC Patch ROM ID Max	1	Maximum version of the ROM for which patch was prepared.						
		PMC Patch Release Date - Month	3	PMC Patch Release date in ASCII code, Month e.g."May"						
		PMC Patch Release Date - Day	2	PMC Patch Release date in ASCII code, Day e.g."09"						
		PMC Patch Release Date - Year	4	PMC Patch Release date in ASCII code, Year e.g."2012"						

		PMC Patch Release Time - Hour	2	PMC Patch Release time in ASCII code, Hour e.g "16"						
		PMC Patch Release Time - Minutes	2	PMC Patch Release time in ASCII code, Minute e.g "06"						
		PMC Patch Release Time - Seconds	2	PMC Patch Release time in ASCII code, Second e.g "38"						
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
52h	**Get Current PMC Patch State**	**Request:**			Y	Y	Y	Y	Y	Y
			0							
		Response:								
		PMC Patch Timeout Counter	2	Incremented when procedure of PMC Patching were waiting for PMC Wait For Ready flag						
		PMC Patch Error Counter	2	Incremented when PMC Patching encountered problems and was not done						
		PMC Patch Success	2	Incremented when PMC Patching ended with success						
		PMC Patch Time	4	Time in HPT ticks when the patching ocured counting from ME start						
		PMC Patch Status	4	Status of the PMC Patching procedure {PMCPATCH_STATUS {PMC_SUCCESS = 0, PMC_NOT_PRESENT, PMC_NOT_NEEDED, PMC_NO_PMC_PATCH_FOR_ID, PMC_TIMEOUT, PMC_INCORRECT_DATA_VER, PMC_NO_DATA_FOR_PATCH, PMC_AUTHENTICATE_ERROR, PMC_GET_ID_ERROR, PMC_SET_PATCH_NOT_PRESENT_BIT_ERROR, PMC_SIZE_ERROR, PMC_COPY_FAILED, PMC_LOCK_ERROR, PMC_DIAG_ERROR, PMC_FUSE_DISABLED}						
		PMC Patch Index	1	Index of the current PMC Patch in the table of Patches - valid indexes: [0..15]						
		Prod ID	1	Actual ID of the matherboard of the DUT.						
		Rev ID	1	Actual PMC Revision of the DUT.						
		ROM ID	1	Actual ROM of the DUT.						
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
53h	**Get Next PMC Patch Info**	**Request:**			Y	Y	Y	Y	Y	Y
		Enumeration Context	1	From the response to Get First PMC Patch Info/Get Next PMC Patch Info						
		Response:								
		PMC Patch Existance	1	If patch at this enumeration exists.						
		PMC Patch Product ID	1	ID of the matherboard for which patch was prepared.						
		PMC Patch Rev ID Min	1	Minimum version of the PMC revision for which patch was prepared.						
		PMC Patch Rev ID Max	1	Maximum version of the PMC revision for which patch was prepared.						
		PMC Patch ROM ID Min	1	Minimum version of the ROM for which patch was prepared.						
		PMC Patch ROM ID Max	1	Maximum version of the ROM for which patch was prepared.						
		PMC Patch Release Date - Month	3	PMC Patch Release date in ASCII code, Month e.g."May"						

		PMC Patch Release Date - Day	2	PMC Patch Release date in ASCII code, Day e.g."09"						
		PMC Patch Release Date - Year	4	PMC Patch Release date in ASCII code, Year e.g."2012"						
		PMC Patch Release Time - Hour	2	PMC Patch Release time in ASCII code, Hour e.g "16"						
		PMC Patch Release Time - Minutes	2	PMC Patch Release time in ASCII code, Minute e.g "06"						
		PMC Patch Release Time - Seconds	2	PMC Patch Release time in ASCII code, Second e.g "38"						
		Enumeration Context	1	Opaque number (handler) that needs to be returned to the ME FW to get a next PMC Patch						
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
54h	**Get PBC statistics**	**Request:**			N	N	Y	Y	Y	Y
		Domain ID	1	0x00 - platform; 0x01 - CPU; 0x02 - Memory; 0x03 - HW protection; 0x04 - HPIO domain						
		Mode Type	1	0x00 - Limiting quality statistics; 0x01 - processing statistics; Other - Reserved						
		Response:								
		Mode	1	0x00 - Limiting quality statistics; 0x01 - processing statistics;						
		For Mode 0x00 or without Mode Type byte:								
		Above Limit Time	1	[0.1s] Time above limit						
		Readings Error Time	1	[0.1s]						
		Statistics Time	1	[0.1s] Total time for which statistics are reported. Max 20s.						
		For Mode 0x01:								
		Reserved	1							
		Max reading processing time in domain	2	[10us] Time of reading processing in domain						
		Max reading processing time in all domains	2	[10us] Time of reading processing in all domains						
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
55h	**Clear PBC statistics**	**Request:**			N	N	Y	Y	Y	Y
		Domain ID	1	0x00 - platform; 0x01 - CPU; 0x02 - Memory						
		Mode	1	0x00 - Limiting quality statistics; 0x01 - processing statistics; Other - Reserved						
		Response:								
			0							
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						

56h	**Set Max Allowed CPU P-State/T-State**	**Request:**			N	N	Y	Y	N	Y
		Domain ID	1	0 – Entire platform – for compatibility with previous NM versions P/T state settings are applied to CPU subsystem, others reserved						
		Control Knob	1	1 – set max allowed CPU P-state and T-state; 2 – set max allowed CPU cores; Others – reserved						
		– For Control Knob set to 1:								
		P-State	1	P-State number to be set						
		T-State	1	T-State number to be set						
		– For Control Knob set to 2:								
		Threads Enabled	2	Number of threads that should be enabled on a system.						
		Response:								
		Knob Sequence No	1	Sequence number of the request sent to host OSPM						
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
57h	**Get Max Allowed CPU P-State/T-State**	**Request:**			N	N	Y	Y	Y	Y
		Domain ID	1	0x00 – Entire platform – for compatibility with previous NM versions P/T state settings are applied to CPU subsystem, others – Reserved						
		Control Knob	1	1 – get max allowed CPU P-state/T-state 2 – get max allowed CPU cores Others – Reserved						
		Response:								
		– For Control Knob set to 1:								
		P-State	1	Current maximum P-State						
		T-State	1	Current maximum T-State						
		– For Control Knob set to 2:								
		Cores	1	Total requested by ME number of allowed cores on a system. This is a number requested by ME and OSPM is not required to fulfill this request						
		Knob Sequence No	1	Sequence number of the request recently confirmed by host OSPM						
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code. In case of failure, ME error code will be returned.						
58h	**Read All Fuses**	**Request:**			N	Y	Y	Y		Y
		Response:								
		HW fuses	4	Settings of all HW fuses of the chipset responsible for disabling/enabling capabilities (aka Hard Staps)						

		FW straps	4	Values of fuse overrides saved in ME FW responsible for disabling/enabling capabilities (aka Soft Staps)						
		Final capability states	4	The result of an AND operation for capabilities enabled/disabled by the HW and FW straps						
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
59h	**Get Me Configuration**	**Request:**			Y	Y	Y	Y	Y	Y
		Response:								
		Platform Type	1	[0] – Bromolow type platform [1] – Romley type platform [2] – Denlow type platform [3] – Grantley type platform [4] - Greenlow type platform [5] - Purley type platform [6] - Harissonville type platform [7] – Reserved (extended platform info to be added in the future)						
		HW Configuration	1	[0] – Recovery Jumper enabled [1] – Always on power mode enabled [2…7] – Reserved						
		SW Configuration	4	Bit coded ME FW feature set as defined in ME-BIOS spec and Castle Crest Configurability.xls "Feature List" tab						
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
5Ah	**Set PTAM State**	**Request:**			N	N	Y	Y	N	Y
		PTAM State	1	0 – Disable PTAM; 1 – Enable PTAM						
		Response:								
		PTAM State	1	PTAM state before this command was executed						
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
5Bh	**Get PTAM State**	**Request:**			N	N	Y	Y	Y	Y
		Response:								
		PTAM State	1	0x00 – PTAM Enabled; 0x01 – PTAM Disabled						
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
5Ch	**Get P/T State Violation Time**	**Request:**			N	N	Y	Y	Y	Y
		Response:								
		ViolationTime / TotalTime [%]	1	Percentage of time when P/T State violation occurred						
		TotalTime [0.1s]	4	Total time in which violation was counted						
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
5Dh	**Reset P/T State Violation Time**	**Request:**			N	N	Y	Y	N	Y
			0							
		Response:								
			0							
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						

60h	**Read ICC Register**	**Request:**			Y	Y	Y	Y	Y	Y
		Endpoint ID	1	One of 0xED, 0xE9,0xEA, 0xDC						
		Register Offset	2	AdrL, AdrH See CSME_ICC_REGS_SPT.xlsx for exact values						
		Response:								
		Register Value	4	Current value of ICC register D0(LSB), D1 ,D2, D3(MSB)						
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
67h	**Get Sensor ID**	**Request:**			N	Y	Y	Y	Y	Y
		ReadingType	2	One of predefined reading types. Use 0 for any ReadingType.						
		DeviceIdx	2	Unique number for devices of this same DeviceType. Use 0 for any.						
		IPMI Sensor Number	1	IPMI Sensor Number (SDR) associated with MS sensor. Setting this to value other than 0 overrides other search filters.						
		Previous Sensor ID	4	Previous sensor ID returned after search. Use 0 at beginning.						
		Response:								
		Sensor ID	4							
		ReadingType	2	One of predefined reading types.						
		DeviceIdx	2	Unique number for devices of this same DeviceType						
		IPMI Sensor Number	1	IPMI Sensor Number (SDR) associated with MS sensor.						
		HW config	1-10	Minimal HW configuration required to identify device. It is 10 first bytes of HW description in PIA file.						
68h	**Get Sensor Discovery Data**	**Request:**			N	Y	Y	Y	Y	Y
		Sensor ID	4	MS sensor identifier.						
		IPMI Sensor Number	1	IPMI Sensor Number (SDR) associated with MS sensor. Setting this to value other than 0 overrides Sensor ID value.						
		Discovery data bytes offset	1	Identify which bytes should be returned. Should be set to 0 if discovery data read is started.						
		Flags	1	[0] - Trigger discovery - Force update of discovery data. Discovery would be triggered for all devices of this same reading type. Data would be available after discovery finished – additional read would be required. [1-7] - reserved						
		Response:								
		ReadingType	2	One of predefined reading types.						
		DeviceIdx	2	Unique number for devices of this same DeviceType						

		Discovery state	1	Defines if device associated with sensor was detected in system 0x00 - MS_DISCOVERY_STATUS_PRESENT - Discovery process has detected the sensor 0x01 - MS_DISCOVERY_STATUS_NOT_PRESENT - Discovery process cannot detect the sensor 0x02 - MS_DISCOVERY_STATUS_NOT_SUPPORTED_IN_SX - Sensor is not supported in Sx states					
		Discovery data size	1	Total discovery data size. If it is greater than number of received bytes next request should be send with proper data bytes offset.					
		Discovery data	1+	Device parameters read in discovery process. Size and format depends on DeviceType. Set to 0 if device isn't present. Array of 1 byte values					
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.					
6Ah	**Get Page Fault Statistics**	**Request**			N	Y	Y	Y	Y
			0						
		Response							
		SyncNvmRateAverage	4	1 per second					
		SyncNvmRateMinimum	4	1 per second					
		SyncNvmRateMaximum	4	1 per second					
		SyncNvmProcessingTimeAvg	4	in microseconds					
		SyncNvmProcessingTimeMin	4	in microseconds					
		SyncNvmProcessingTimeMax	4	in microseconds					
		SyncUmaRateAverage	4	1 per second					
		SyncUmaRateMinimum	4	1 per second					
		SyncUmaRateMaximum	4	1 per second					
		SyncUmaProcessingTimeAvg	4	in microseconds					
		SyncUmaProcessingTimeMin	4	in microseconds					
		SyncUmaProcessingTimeMax	4	in microseconds					
		AsyncNvmRateAverage	4	1 per second					
		AsyncNvmRateMinimum	4	1 per second					
		AsyncNvmRateMaximum	4	1 per second					
		AsyncNvmProcessingTimeAvg	4	in microseconds					
		AsyncNvmProcessingTimeMin	4	in microseconds					
		AsyncNvmProcessingTimeMax	4	in microseconds					
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.					

6Bh	**Get Sensor HW Configuration**	**Request:**			N	Y	Y	Y	Y	Y
		Sensor ID	4B	Monitoring Service sensor identifier						
		IPMI Sensor Number	1B	IPMI Sensor Number (SDR) associated with MS sensor. Setting this to value other than 0 overrides Sensor ID value.						
		Response:								
		ReadingType	2B	One of predefined reading types.						
		DeviceIdx	2B	Unique number for devices of this same DeviceType						
		Discovery state	1B	Defines if device associated with sensor was detected in system						
		HW config	1B-10B	Minimal HW configuration required to identify device. It is 10 first bytes of HW description in PIA file.						
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
6Ch	**Get Sensor Value**	**Request:**			N	Y	Y	Y	Y	Y
		Sensor ID	4	Monitoring Service sensor identifier						
		IPMI Sensor Number	1	IPMI Sensor Number (SDR) associated with MS sensor. Setting this to value other than 0 overrides Sensor ID value.						
		Response:								
		ReadingType	2	One of predefined reading types.						
		DeviceIdx	2	Unique number for devices of this same DeviceType						
		Update timestamp	4	Timestamp of last sensor update (HPT ticks, 10 us resolution)						
		Reading value	4	Current sensor value						
		ReadingFormat	1	Format of returned value: 0 - integer, 1 - FIXED_INT, 2 - bitmask						
		Command Status		= 00h - Command performed successfully = 88h - Sensor not present or last reading not successful = 89h - search by IPMI Sensor Number not supported (should be set to 0) = 05h - invalid command length = 03h - internal error (invalid pointer error) = 01h - generic error (unspecified)						
6Fh	**Aggregated Send Raw PECI**	**Request:**			N	Y	Y	Y	N	N
		PECI Request	1:N	Raw PECI command bytes formatted according to the same rules as bytes 1 to N in Send Raw PECI.						
		Next PECI Request	N+1:M	Next RAW PECI request command (if any).						
				PECI Interface Selection field must be same for all PECI transactions listed in the request						
		Response:								
		PECI Read Data	1	Completion Code related to overall request						

				= 00h – Success; it means that all the PECI raw requests have been processed and the responses fit in the IPMI response message (not necessarily that all the responses have completed with success). The completion codes for the particular PECI raw transactions are included in the appropriate parts of this response frame.						
			2	Completion Code for this particular transaction – same as byte#1 in Send Raw PECI command response.						
			3:3+N	the first PECI response data received from PECI client during Read transaction phase (if any), formatted in the same way as response in Send Raw PECI.						
			N+1:M+1	next PECI transaction: Completion Code byte + response (if any); individual responses are returned in the order they were in the request.						
				Note – It is the BMC's responsibility to ensure the responses can fit in the IPMI response message (knowing the max IPMI response frame length supported by Intel® ME FW). If some responses do not fit into the IPMI response message, they are not returned (but execution of the corresponding PECI commands is attempted).						
				Note – Intel® ME FW interprets each raw PECI request based on the Write Length field in each PECI request. Invalid Write Length in a malformed raw PECI request will likely cause all subsequent requests to be interpreted by Intel ME FW starting from a wrong offset and will likely result in C7h error code for the whole IPMI request.						
		Command Status		= 00h – PECI response successfully retuned (see PECI response completion code for detailed response from PECI client) = A4h – Bad Read FSC in the response (even after the retry) = A5h – Bad Write FCS field in the response (even after the retry); this error code is also returned in case of Abort FSC in the response (as defined in PECI spec) and no response from PECI client (client device is not responding at all) = A6h – bad Write Length in the request = A7h – bad Read Length in the request = A8h - Selected PECI interface not available = ABh – command code in the request not understood by ME FW = CCh - Aggregated PECI requests have selected different PECI interfaces to be used						
7Dh	**Get IPMB Stats**	**Request**			N	Y	Y	Y	Y	Y
		Response								
		IPMB Packets Received	2	Number of IPMB frames received						
		IPMB Packets Sent	2	Number of IPMB frames sent						

		IPMB Packets Sent with Error	2	number of failed attempts to sent a packet through I2C						
		IPMB Packets Received with Error	2	number of recieve errors from I2C						
		IPMB Packets Dropped	2	number of packets received but not processed (dropped)						
		IPMB Incomplete Requests	2	number of requests that did not finish successfully (i.e. no responses, etc.)						
		IPMB Packets Sent Back	2	number of requests received but sent back because of errors						
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
7Eh	**Get PCH GPIO Ownership**	**Request**			Y	N	N	Y	N	Y
		Group	1	GPIO Group: 0: GPP_A 1: GPP_B 2: GPP_C 3: GPP_D 4: GPP_E 5: GPP_F 6: GPP_G 7: GPP_H 8: GPP_I 9: GPD						
		Response								
		Group	1	GPIO Group: 0: GPP_A 1: GPP_B 2: GPP_C 3: GPP_D 4: GPP_E 5: GPP_F 6: GPP_G 7: GPP_H 8: GPP_I 9: GPD						
		Gpio	24	Each byte represents a single pin ownership 0: Host Mode 1: CSME Mode 2: ISH Mode 3: IE Mode						
87h	**Get HECI Stats**	**Request**			N	Y	Y	Y	Y	Y
		Heci Device Index	1	HECI Device Index (0:HECI 1, 1:HECI2)						
		Response								
		Number of Driver Request	2							
		Number of Unrecognized Request	2							
		Buffer Overflow	2							
		Me to Host Counter	2							
		Bad Header Count	2							
		Link status	1							
		Last Bad Header	4							
		Host control status register	4							
		Me control status register	4							

		Host to me Last three heci Header	12							
		Me to Host Last three Heci Header	12							
		Host to Me Counter array	0 - 66	Displayed in [ME_ADDR,COUNT_VALUE] pairs when count > 0						
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
88h	**Get PBC Scalability Factors**	**Request**			N	Y	Y	Y	Y	Y
			0							
		Response								
		Array of scalability factors for each CPU	1-4							
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
89h	**Get GPIO**	**Request:**			Y	N	N	Y	N	Y
		GPIO Group	1	GPIO Group: 0: GPP_A 1: GPP_B 2: GPP_C 3: GPP_D 4: GPP_E 5: GPP_F 6: GPP_G 7: GPP_H 8: GPP_I 9: GPD						
		GPIO Pad	1	Pad Number [0…23]						
		Respond:								
		GPIO Group	1	GPIO Group: 0: GPP_A 1: GPP_B 2: GPP_C 3: GPP_D 4: GPP_E 5: GPP_F 6: GPP_G 7: GPP_H 8: GPP_I 9: GPD						
		GPIO Pad	1	Pad Number [0…23]						
		GPIO Value	1	0 or 1						
		GPIO Register	4	Full bits of the register that describe this GPIO						
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
8Ah	**Set GPIO Output State**	**Request:**			Y	N	N	Y	N	Y
		GPIO Group	1	GPIO Group: 0: GPP_A 1: GPP_B 2: GPP_C 3: GPP_D 4: GPP_E 5: GPP_F 6: GPP_G 7: GPP_H 8: GPP_I 9: GPD						
		GPIO Pad	1	Pad Number [0…23]						

		GPIO Value	1	0 or 1						
		GPIO Direction	1	0 = input, 1 = output						
		Respond:								
		GPIO Value	1	0 or 1						
		GPIO Direction	1	0 = input, 1 = output						
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
8Bh	**Reset HECI Stats**	**Request**			N	Y	Y	Y	Y	Y
		Heci Device Index	1	HECI Device Index (0:HECI 1, 1:HECI2)						
		Response								
		HECI 1 Subsystem Response	1							
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
8Eh	**Get Mic Jitter Per Sensor**	**Request**			N	N	Y	N	N	N
		Sensor ID	4B	MS sensor identifier.						
		IPMI Sensor Number	1B	IPMI Sensor Number (SDR) associated with MS sensor. As the Sensor ID is equal to SDR number in CastleCrest this should always be set to 0.						
		Reset Stats Flag	1B	If equals 1 reset stats						
		Response								
		Jitter Value Avg	4B	Average value of mic jitter in HPTtimer ticks						
		Time for Avg	4B	Time which in average value is calculated						
		Jitter Value Min	4B	Minimal value of mic jitter in HPTtimer ticks						
		Jitter Value Max	4B	Maximum value of mic jitter in HPTtimer ticks						
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
90h	**MDES Set Logger On**	**Request**			N	Y	Y	Y	Y	Y
			0							
		Response								
			0							
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
91h	**MDES Set Logger Off**	**Request**			N	Y	Y	Y	Y	Y
			0							
		Response								
			0							
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
92h	**MDES Get Logger State**	**Request**			N	Y	Y	Y	Y	Y
			0							
		Response								
		Logger State	1	Off = 0, On = 1						

		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
93h	**MDES Set Error Filter**	**Request**			N	Y	Y	Y	Y	Y
		Error Filter	1	All = 0, Low = 1, High = 2, Critical = 3						
		Response								
			0							
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
94h	**MDES Get Error Filter**	**Request**			N	Y	Y	Y	Y	Y
			0							
		Response								
		Error Filter	1	All = 0, Low = 1, High = 2, Critical = 3						
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
95h	**MDES Set Event Filter**	**Request**			N	Y	Y	Y	Y	Y
		Event Group	1	0 ... 127 (CP = 1, LOADMGR = 4, PWRMGMT = 5, FWSTS = 72, TMRALIVE = 73, KERNEL = 82, POLICY = 83, HOSTCOMM = 84)						
		Event Filter	4	0x00000000 ... 0xFFFFFFFF Recommended values: CP = 0x1, LOADMGR = 0x3F6, PWRMGMT = 0x1, FWSTS = 0x1, TMRALIVE = 0x1, KERNEL = 0xFFFFFFFF, POLICY = 0xFFFFFFFF, HOSTCOMM = 0xFFFFFFFF						
		Response								
			0							
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
96h	**MDES Get Event Filter**	**Request**			N	Y	Y	Y	Y	Y
		Event Group	1	0 ... 127 (CP = 1, LOADMGR = 4, PWRMGMT = 5, FWSTS = 72, TMRALIVE = 73, KERNEL = 82, POLICY = 83, HOSTCOMM = 84)						
		Response								
		Event Group	1	0 ... 127 (CP = 1, LOADMGR = 4, PWRMGMT = 5, FWSTS = 72, TMRALIVE = 73, KERNEL = 82, POLICY = 83, HOSTCOMM = 84)						
		Event Filter	4	0x00000000 ... 0xFFFFFFFF						
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
97h	**MDES Set Logging Iface**	**Request**			N	Y	Y	Y	Y	Y
		Logging Interface	1	None = 0, SmBus = 2, Flash = 4						
		Response								
			0							

		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
98h	**MDES Get Logging Iface**	**Request**			N	Y	Y	Y	Y	Y
			0							
		Response								
		Logging Interface	1	None = 0, SmBus = 2						
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
99h	**MDES Set Buffer Mode**	**Request**			N	Y	Y	Y	Y	Y
		Buffer Mode	1	Blocking = 0, Buffered = 1, Delayed Flush = 2						
		Response								
			0							
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
9Ah	**MDES Get Buffer Mode**	**Request**			N	Y	Y	Y	Y	Y
			0							
		Response								
		Buffer Mode	1	Blocking = 0, Buffered = 1, Delayed Flush = 2						
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
9Bh	**MDES Set SmBus Address**	**Request**			N	Y	Y	Y	Y	Y
		SmBus Address	1	7-bit SmBus address [0x00 ... 0x7F]						
		Response								
			0							
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
9Ch	**MDES Get SmBus Address**	**Request**			N	Y	Y	Y	Y	Y
			0							
		Response								
		SmBus Address	1	7-bit SmBus address [0x00 ... 0x7F]						
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
9Dh	**Set GPIO Output State**	**Request**			N	Y	Y	Y	N	Y
		GPIO Group	1	GPIO Group: 0: GPP_A 1: GPP_B 2: GPP_C 3: GPP_D 4: GPP_E 5: GPP_F 6: GPP_G 7: GPP_H 8: GPP_I 9: GPD						
		GPIO Pad	1	Pad Number [0…23]						
		MGPIO State	1	0 - for low state, 1 for high state						
		Response								

			0							
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
9Eh	**Send Raw PMBus**	**Request**			N	Y	Y	Y	N	Y
			1	[7] - Reserved [6:4] - Transaction Type [3:2] - Device Address Format [1] - Ignore PEC Error [0] - Enable PEC						
		SmBus Address	1	[7] - Reserved [6:0] - 7-bit slave address						
			1	[7:2] - Mux State [1:0] - Reserved						
			1	[7:3] - Reserved [2] - TransProtocol [1:0] - Reserved						
		Write Length	1							
		Read Length	1							
		Request Data	1+n	Variable length						
		Response								
		Data returned from PMBus device	1+n							
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
9Fh	**Read Power Data**	**Request:**			Y	Y	Y	Y	Y	Y
		Register Address	2	Address of the register						
		Response								
		Register Value	4	Register Value						
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
A7h	**Get Total Power Budget**	**Request**			N	N	Y	Y	Y	Y
		Domain ID	1							
		Response								
		Total Power Budget	2	If Status Code != STATUS_SUCCESS then no data byte						
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
A8h	**Get Limiting Policy Id**	**Request**			N	N	Y	Y	Y	Y
		Domain ID	1							
		Response								
		Current Limiting Policy ID	2	If Status Code != STATUS_SUCCESS then no data byte						
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						

A9h	**Get SMART CLST Stats**	**Request**			N	N	Y	Y	Y	Y
		Mode	1							
		Response								
		Record Data	13	If Status Code != STATUS_SUCCESS then no data byte						
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
AAh	**Get PECI Plug-in Data**	**Request**			N	Y	Y	Y	Y	Y
		Data ID	1							
		Data Sub ID	1							
		Response								
		Reading Data	1	If Status Code != STATUS_SUCCESS then no data byte						
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
ABh	**Get NM Status Monitor Data**	**Request**			N	N	Y	Y	Y	Y
		Data ID	1							
		Data Sub ID	1							
		Response								
		Capabilities Data	2	If Status Code != STATUS_SUCCESS then no data byte						
		Last Change Source Value	1	If Status Code != STATUS_SUCCESS then no data byte						
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
ACh	**Get Pmbus Plug-in Data**	**Request**			N	Y	Y	Y	Y	Y
		Data ID	1							
		Data Sub ID	1							
		Response								
		Data	4	if Status Code != STATUS_SUCCESS then no data byte. The number of bytes accessed (See FAS) may vary, but it is always returned as a 4 byte value.						
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
AD h	**Get Self-Test Results**	**Request**			N	N	Y	Y	Y	Y
			0							
		Response								
		Test Code	1	If Status Code != STATUS_SUCCESS then no data byte						
		Error Code	1	If Status Code != STATUS_SUCCESS then no data byte						
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
AEh	**Get ME crash dump**	**Request**			N	N	N	N	N	N
		Offset	1	Offset within crash dump data to read. 0 means the beginning of the data.						
		Response								
		Bytes Left	1	The number of bytes left from the last read.						

		Data	1 to 16	Data in the crash dump file started from offset						
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
AFh	**Get FW Data PBC**	**Request**			N	N	Y	Y	Y	Y
		Data ID	1							
		Data Sub ID	1							
		Response								
		Capabilities Data	4	if Status Code != STATUS_SUCCESS then no data byte. The number of bytes accessed (See Remote MESDC FAS) may vary, but it is always returned as a 4 byte value.						
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
B0h	**Get IDLM PID**	**Request**			Y	N	N	Y	N	Y
			0							
		Response								
		DeviceId	2	Device ID of the MBB bridge						
		FuseTestFlags	2	Flags to be passed to Host						
		UMCHID[0]	4	UMCHID value calculated from unique fuses						
		UMCHID[1]	4	UMCHID value calculated from unique fuses						
		UMCHID[2]	4	UMCHID value calculated from unique fuses						
		UMCHID[3]	4	UMCHID value calculated from unique fuses						
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
B1h	**MDES Clear FEL Entries**	**Request**			N	Y	Y	Y	Y	Y
		Response								
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						

B2h	**MDES Read FEL Entry**	**Request**			N	Y	Y	Y	Y	Y
		Command	4	0 - Get first FEL Entry 1 - Get next FEL Entry 2 - Get next block in current entry This command must follow a procedure to ensure all FEL entries are retrieved. 1. This command must always start by sending Command = 0. This will reset the internal FEL entry pointer to the first entry in the ring buffer. 2. Then retrieve the next block of bytes with Command = 2. Continue issuing requests with Command = 2 until all bytes of the current entry have been retrieved. 3. Then issue Command = 1. This retrieves the first block of bytes in the next entry. Expect completion code DIAG_STATUS_NOT_FOUND (0x81) if there are no more entries. 4. Again, issue Command = 2 repeatedly to request all blocks of bytes in the current entry. Expect completion code DIAG_STATUS_NOT_FOUND (0x81) if there are no more blocks of data to read for current entry. 5. Goto step 3 until all FEL entries have been retrieved.						
		Reserved	2	reserved for future use. Just set this to 0 for now.						
		Response								
		Bytes Left	2	The number of bytes left in current entry from the last read.						
		Data	0-14	Data bytes from the current entry request						
		Command Status		STATUS_SUCCESS - OK DIAG_STATUS_GENERAL_FAILURE - General failure completing the request DIAG_STATUS_SIZE_ERROR - Internal problem with message or buffer size DIAG_STATUS_NOT_FOUND - If buffer is empty or no more entries to read						
B3h	**MDES SET FEL Entry**	**Request**			N	Y	Y	Y	Y	Y
		Reserved	1	reserved for future use. Just set this to 0 for now.						
		Data	20	A NULL terminated ascii string (19 chars and 1 NULL terminator). Example: 0x41 0x41 0x41 0x41 0x41 0x41 0x41 0x41 0x41 0x41 0x41 0x41 0x41 0x41 0x41 0x41 0x41 0x41 0x41 0x00 Output: AAAAAAAAAAAAAAAAAA						
		Response								
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
B4h	**FMM Get Statistics**	**Request**			N	Y	Y	N	N	N
		Response								
		CurrentValue	4	Current power value						
		AverageValue	4	Average power value						
		MaxValue	4	Maximum power value						

		MinValue	4	Minimum power vlue						
		FailedReadingsCount	4	Failed readings count - indicates how many readings was not successfully received						
B5h	**FMM Reset Statistics**	**Request**			N	Y	Y	N	N	N
		Response								
B6h	**FMM Get Threshold**	**Request:**			N	Y	Y	N	N	N
		Response								
		CurrentThreshold	4	Current threshold						
B8h	**Get FW Data PSC**	**Request**			N	N	Y	Y	Y	Y
		Data ID	1							
		Data Sub ID	1							
		Response								
		Data	4	if Status Code != STATUS_SUCCESS then no data byte. The number of bytes accessed (See Remote MESDC FAS) may vary, but it is always returned as a 4 byte value.						
		Command Status		STATUS_SUCCESS, STATUS_NOT_SUPPORTED (0x89) – if command filter does not allow to execute this command STATUS_INVALID_PARAMS (0x85) – if one of parameters is out of range						
B9h	**Get FW Data PSU OPTIM**	**Request**			N	N	Y	Y	Y	Y
		Data ID	1							
		Data Sub ID	1							
		Response								
		Data	4	if Status Code != STATUS_SUCCESS then no data byte. The number of bytes accessed (See Remote MESDC FAS) may vary, but it is always returned as a 4 byte value.						
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
BAh	**Get FW Data PTAM**	**Request**			N	N	Y	Y	Y	Y
		Data ID	1							
		Data Sub ID	1							
		Response								
		Data	4	if Status Code != STATUS_SUCCESS then no data byte. The number of bytes accessed (See Remote MESDC FAS) may vary, but it is always returned as a 4 byte value.						
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						

BBh	**Read PMC Register**	**Request:**			Y	N	N	Y	N	N
		Offset	2	PPS - 0x204, PSCTRL- 0x210, MEWS 0x218						
		Response:								
		Register Value	4	PMC Register content						
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
C0h	**Get PECI Statistics**	**Request**						Y	Y	Y
		CpuId	1	CPU Index						
		Response								
		SendTime	4	Amount of time [us] sent data						
		RcvdTime	4	Amount of time [us] received data						
		DataSentPhys	4	Amount of data physically sent in bits						
		DataRcvdPhys	4	Amount of data physically received in bits						
		DataSent	4	Amount of data sent in bytes						
		DataRcvd	4	Amount of data received in bytes						
		NbOfRepeats	4	Number of repeats						
		NbOfChecksumError	4	Number of checksum errors						
		NbOfLackOfResp	4	Number of lack of response						
		NbOfTimeouts	4	Number of timouts						
		NbOfBusErrors	4	Number of bus errors						
		NbOfFailedTrans	4	Number of failed transactions						
		NbOfSystemFails	4	Number of fails due system limitations (not enough memory to process transaction)						
		MaxTransTime	4	Maxium amount of time send data in us						
		DrvStatState	2	Check if above mentioned statistic values are initialized (14 bits from LSB accordingly)						
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
C1h	**Get PECI Extended Statistics**	**Request**						Y	Y	Y
		CpuId	1	CPU Index						
		Response								
		NbOfStartTrans	4	number of start transactions						
		NbOfRecvTrans	4	number of transactions with response						
		NbOfRetrySucTrans	4	number of successful retries						
		NbOfRetryUnsucTrans	4	number of completely unsuccessful transactions						
		NbOfHwWTrue	4	number of start transactions with HW Workaround						
		WriteFcsError	4	number of write FCS errors						
		ReadFcsError	4	number of read FCS errors						
		LostArbitration	4	number of lost arbitrations						

		DrvExtStatState	2	Check if abobe mentioned statistic values are initialized (8 bits from LSB accordingly)						
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
C2h	**Clear PECI Statistics**	**Request**						Y	Y	Y
		CpuId	1	CPU Index						
		Response								
			0							
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
C3h	**Set PECI Statistics Configuration**	**Request**						Y	Y	Y
		CpuId	1	CPU Index						
		EnableStatistics	1	1 = Enable Statistics, 0 = Disable Statistics for given CPU						
		Response								
			0							
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
C4h	**Get PECI Statistics Configuration**	**Request**						Y	Y	Y
		CpuId	1	CPU Index						
		Response								
		StatisticsEnabled	1	1 = Statistics enabled, 0 = Statistics disabled for given CPU						
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
C5h	**Get Inband PECI Statistics**	**Request**						Y	N	N
			0							
		Response								
		TimePeriod	4	time period in mili senonds						
		RequestCounter	4	number of requests sent via Inband PECI						
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
C8h	**Get SMT Statistics per Interface**	**Request**						Y	Y	Y
		Interface Index	1	Interface Index						
		Master/Slave	1	1 = Statistics for Master, 0 = Statistics for Slave						
		Response								
		SendTime	4	Amount of time [us] sent data						
		RcvdTime	4	Amount of time [us] received data						
		DataSentPhys	4	Amount of data physically sent in bits						
		DataRcvdPhys	4	Amount of data physically received in bits						
		DataSent	4	Amount of data sent in bytes						
		DataRcvd	4	Amount of data received in bytes						
		NbOfRepeats	4	Number of repeats						

		NbOfChecksumError	4	Number of checksum errors					
		NbOfLackOfResp	4	Number of lack of response					
		NbOfTimeouts	4	Number of timouts					
		NbOfBusErrors	4	Number of bus errors					
		NbOfFailedTrans	4	Number of failed transactions					
		NbOfSystemFails	4	Number of fails due system limitations (not enough memory to process transaction)					
		MaxTransTime	4	Maxium amount of time send data in us					
		DrvStatState	2	Check if above mentioned statistic values are initialized (14 bits from LSB accordingly)					
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.					
C9h	**Clear SMT Statistics per Interface**	**Request**							
		Interface Index	1	Interface Index					
		Master/Slave	1	1 = Clear Statistics for Master, 0 = Clear Statistics for Slave			Y	Y	Y
		Response							
			0						
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.					
CAh	**Add SMT Statistics per Device**	**Request**							
		Logical Address	4	Need definition from Base System guys					
		TransportChannel	1	Transport Channel - values are checked for valid channels.					
		CollectionLevel	1	Currently not used 0 - means "generic" stats at SMT Level, for technical reasons there are statistics for outgoing (ME is MASTER for a target devivce) transactions only 1 - means "IPMI" stats, there are stats for both directions (ME is Master and ME is SLAVE for a target device) transactions, but for devices using IPMB only (MICs)	N	Y	Y	Y	Y	Y
		Response							
			0						
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.					
CBh	**Remove SMT Statistics per Device**	**Request**							
		Logical Address	4	Need definition from Base System guys					
		TransportChannel	1	Transport Channel - values are checked for valid channels.					
		Response			N	Y	Y	Y	Y	Y
			0						
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.					

CCh	**Clear SMT Statistics per Device**	**Request**			N	Y	Y	Y	Y	Y
		Logical Address	4	Need definition from Base System guys						
		TransportChannel	1	Transport Channel - values are checked for valid channels.						
		Response								
			0							
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
CDh	**Get SMT Statistics per Device**	**Request**			N	Y	Y	Y	Y	Y
		Logical Address	4	Need definition from Base System guys						
		TransportChannel	1	Transport Channel - values are checked for valid channels.						
		Response								
		SendTime	4	Amount of time [us] sent data						
		RcvdTime	4	Amount of time [us] received data						
		DataSentPhys	4	Amount of data physically sent in bits						
		DataRcvdPhys	4	Amount of data physically received in bits						
		DataSent	4	Amount of data sent in bytes						
		DataRcvd	4	Amount of data received in bytes						
		NbOfRepeats	4	Number of repeats						
		NbOfChecksumError	4	Number of checksum errors						
		NbOfLackOfResp	4	Number of lack of response						
		NbOfTimeouts	4	Number of timouts						
		NbOfBusErrors	4	Number of bus errors						
		NbOfFailedTrans	4	Number of failed transactions						
		NbOfSystemFails	4	Number of fails due system limitations (not enough memory to process transaction)						
		MaxTransTime	4	Maxium amount of time send data in us						
		DrvStatState	2	Check if above mentioned statistic values are initialized (14 bits from LSB accordingly)						
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
CEh	**Get list of SMT Device**	**Request**			N	Y	Y	Y	Y	Y
			0							
		Response								
		Number of devices	4	Number of devices that statistics are being collected for						
		Logical Address	4	Need definition from Base System guys						
		TransportChannel	1	Transport Channel						
		Data	??	Total length will be 4 + (5*NumberOfDevices)						
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						

D0h	**Read HECI Circular Buffers**	**Request**		Read HECI Circular Buffers	N	Y	Y	Y	Y	Y
		Device	1	HECI Device Index (0=HECI 1, 1=HECI 2)						
		Function	1	Buffer Index(0:InputBuffer-HostToME, 1:OutputBuffer-METoHost)						
		Register Address	1	Offset within the circular buffer to be read (d-word index, not byte index) "0" is beginning of buffer and creates a snapshot of the active HECI circular buffer that will be copied from until another "0" address is passed in. Since you can only retrive 4 d-words at a time with this command, you really want to be able to snapshot the buffer.						
		Response								
		Register Value	1	Number of d-words left in the buffer from the last read						
			4-N	Circular Buffer Data. Up to 4 d-words (16-bytes) shall be retuned per read (due to transport limit)						
		Command Status		STATUS_SUCCESS, STATUS_FAILURE, STATUS_INVALID_COMMAND,STATUS_SIZE_ERROR,STATUS_INVALID_ACCESS						
D1h	**Get/Clear VDM Statistics**	**Request**			N	Y	Y	Y	Y	Y
		Get or Clear VDM Statistics	1	0:Get statistics, 1:Clear statistics						
		Response								
		FatalErrorsCounter	4							
		NonFatalErrorsCounter	4							
		RxMsgInterruptsCounter	4							
		UpstreamErrorIntCounter	4							
		RingBufWriteIntCounter	4							
		Command Status		STATUS_SUCCESS, STATUS_FAILURE, STATUS_INVALID_COMMAND,STATUS_SIZE_ERROR,STATUS_INVALID_ACCESS						
D2h	**Get/Clear SPI Flash Statistics**	**Request**			N	Y	Y	Y	N	N
		Options	1	0:Get statistics, 1:Clear statistics						
		Response								
		ReadCount	4	Returns the number of bytes readed since last reset or clear stati						
		WriteCount	4	Returns the number of bytes written since last reset or clear stati						
		ReadErrorCount	2							
		EraseErrorCount	2							
		WriteErrorCount	2	Rerurns the number of errors during flash write exluding Flash cycle error and access error						
		FlashCycleErrorOnWrite	2							
		AccessErrorOnWrite	2							
		CurrentHwSeqFlashStatus	2	Returns the status register value when the command is issued						

		Command Status		STATUS_SUCCESS, STATUS_FAILURE, STATUS_INVALID_COMMAND,STATUS_SIZE_ERROR,STATUS_INVALID_ACCESS						
D3h	**Read SUSRAM File**	**Request**			Y			Y		
		Offset	2	Offset within the file. Used when reading files that are 96 bytes or more.						
		ReadSize	1	Define how many bytes you want to read.						
		Filename	1-N	File name listed inside SUSRAM directory + Null terminator (0x00)						
		Response								
		Bytes Read	1 (N)	# of bytes read						
		SUSRAM File Content	1-N	N equals the size of the SUSRAM file.						
		Command Status		STATUS_SUCCESS, STATUS_NOT_FOUND, STATUS_INVALID_PARAMS, STATUS_FAILURE						
D5h	**Get UMA State**	**Request**			Y	Y	Y	Y	N	N
			0							
		Response								
		UMA Size Valid	1	0: Invalid, 1: Valid						
		UMA Size	4	UMA Size in MB (0x10 0x00 0x00 0x00 = 16MB)						
		Reserved	8							
		Command Status		STATUS_SUCCESS, STATUS_FAILURE						
E1h	**SCA Get Info**	**Request**			N	Y	Y	Y	N	N
		Response								
		ScaInfo	24	Sca Info structure as described in SCA FAS chapter 2.5.1						
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
E2h	**SCA Get Record**	**Request**			N	Y	Y	Y	N	N
		RecordID	4	SCA Record Identifier (LSB first)						
		NofDwordToRead	4	Number of data DWORDs to read						
		Response								
		Offset	4	Data Offset						
		NofDwordsLeft	4	Number of Dwords left untill end of record						
		SCA_data	N	SCA Data record (variable lenght)						
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
E3h	**SCA Consistency Check**	**Request**			N	Y	Y	Y	N	N
		Response								
			0							
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						

F0h	**Force ME Reset**	**Request**			N	Y	Y	Y	N	Y
		Magic Number	4	Magic Number = 0x3CC3A55A						
		Response								
			0							
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
F3h	**Get NM PTU Option ROM Version**	**Request:**			N	N	Y	Y	Y	Y
			0	No request data						
		Response:								
		NM PTU Option ROM Version	1	[7:4] = Major ID [3:0] = Minor ID						
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
F5h	**Get NM PTU Launch State**	**Request:**			N	N	Y	Y	Y	Y
				No request data						
		Response:								
		PTU Launch State	1	[0] - Manufacture Opt-in [1] - BIOS Opt-in [2] - BMC Activate (Both NM and BMC phases) [3] - BIOS Activate [4] – OEM Empty Run [5] – ROM Launched [6] – reserved [7] – BMC Activate (BMC phase only)						
		MROMSPIBAR Address	4							
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
F6h	**Get NM PTU Statistics**	**Request:**			N	N	Y	Y	Y	Y
				No request data						
		Response:								
		Last 4 error codes encountered by NM PTU executing	4							
		Running DC Total Platform Maximum	4							
		Running AC Total Platform Maximum	4							
		Running Core Maximum	4							
		Running Memory Maximum	4							
		Current DC Total Platform Reading	4							
		Current AC Total Platform Reading	4							
		Current Core Reading	4							
		Current Memory Reading	4							
		Current State	4							

		Number of Valid States in BMC Phase	4							
		Debug Out 32	4							
		BMC Phase State	1							
		Debug Out 8	1							
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
F7h	**Option ROM Verbose Enable**	**Request:**			N	N	Y	Y	Y	Y
			0	No request data						
		Respond:								
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
FCh	**Get NM PTU Characterization Data**	**Request:**			N	N	Y	Y	Y	Y
			0	No request data						
		Response:								
		Current DC Total Platform Minimum Power	2							
		Current DC Total Platform Maximum Power	2							
		Current DC Total Platform Efficient Power	2							
		Current AC Total Platform Minimum Power	2							
		Current AC Total Platform Maximum Power	2							
		Current AC Total Platform Efficient Power	2							
		Current Core Minimum Power	2							
		Current Core Maximum Power	2							
		Current Core Efficient Power	2							
		Current Memory Minimum Power	2							
		Current Memory Maximum Power	2							
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						
FDh	**Get NM PTU Sample Window**	**Request:**			N	N	Y	Y	Y	Y
		Domain ID	1	0x00 - Total System, 0x01 - CPU, 0x02 - Memory, 0x03 - Total System DC						
		Response:								
		Domain sample window	40	Each sample is 4 bytes large						
		Domain total sample count	2							
		Command Status		STATUS_SUCCESS, STATUS_FAILURE or other error code.						

MESDC Example Reports

OEM Capture

```
-= [OEM Capture] report generation start: 15/02/03 15:23:14.323 =-

Command Info                                        Argument            Response Value

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

Manage Images: Get Image Version Command in System group.0x00                0x0007000100000904

Get Sensor Id Command in NM group.                  0x000000000000000000
0x00000000040000005802000000000000000004F

Get Current PMC Patch State Command in Diagnostics
group.                    0x19000A010000000000092FC4000200000000

Get SmBus Address Command in MDES group.                                     0x38

-= [OEM Capture] report generation end:   15/02/03 15:23:14.494 =-
```

Intel ME FW Health Check

```
 [ME FW Health Check] report generation start: 15/02/03 15:18:06.124 =-
ME FW VER: 4.0.1.9 (Operational)
-= [ME FW Health Check] report generation end:   15/02/03 15:18:06.467 =-
```

Intel Node Manager State Check

```
-= [Node Manager State Check] report generation start: 15/02/03 15:15:12.106 =-
NM Enabled
POWER STATS: Measurements Suspended for Domain 0
POWER STATS: Input Power Domain 1 Minimum = 0
POWER STATS: Input Power Domain 1 Maximum = 13
POWER STATS: Input Power Domain 1 Average = 1
POWER STATS: Input Power Domain 2 Minimum = 0
POWER STATS: Input Power Domain 2 Maximum = 0
POWER STATS: Input Power Domain 2 Average = 0
POWER STATS: Measurements Suspended for Domain 3
POWER STATS: Input Power Domain 4 Minimum = 0
POWER STATS: Input Power Domain 4 Maximum = 0
POWER STATS: Input Power Domain 4 Average = 0
NM Capabilities: Capability Monitoring Not Availiable
Host Communication: Failure Not Detected
SMART/CLST not triggered
CPU 0 Thermal Status: OK
Current Throttling Level: 0.
-= [Node Manager State Check] report generation end:   15/02/03 15:15:12.621 =-
```

Intel ME configuration Basic Partition

```
-= [ME configuration Basic Partition] report generation start: 15/02/03 14:45:25.528 =-
Nr      File Name ASCII                  Attributes      Size
\\0     RTFD                               00041777        48
\\1     alert_imm                          00041555        72
\\2       AlertImm                         00100770         3
\\3     bup                                00041771       144
\\4     fpf                                00041700        72
\\5     gpio                               00041750        72
\\6     hotham                             00041766        72
\\7     icc                                00041771       672
\\8     .wop                               00100400      1040
\\9     manuf                              00041777        48
\\10    mca                                00041775       192
\\11      eom                              00101744         1
\\12      fpf_commit                       00101700         0
\\13      manuf_lock                       00101744         1
\\14      deploy                           00101740        32
\\15      dal_hack1                        00103777         1
\\16      dal_hack2                        00123777         1
\\17    mca_temp                           00041700        48
\\18    mesdc                              00041555       144
\\19      heci_cb_en                       00100555         1
\\20      heci_filter                      00100555        32
\\21      ipmb_filter                      00100555        32
\\22      smbus_filter                     00100555        32
\\23    mon_serv                           00041555       192
\\24      config                           00041555       120
\\25        0000                           00100770        20
\\26        0001                           00100556         9
\\27        0002                           00100770        96
\\28      dev                              00041555      1416
\\29        0000                           00100770        69
\\30        0001                           00100770        69
\\31        0002                           00100770         0
\\32        0003                           00100770         0
\\33        0004                           00100770         0
\\34        0005                           00100770         0
\\35        0006                           00100770         0
\\36        0007                           00100770         0
\\37        0008                           00100555         0
\\38        0009                           00100770         0
\\39        000a                           00100770         0
\\40        000b                           00100770         0
\\41        000c                           00100555        69
\\42        000d                           00100555        69
\\43        000e                           00100555         0
\\44        000f                           00100555        69
\\45        0010                           00100555         0
\\46        0011                           00100555         0
\\47        0012                           00100555        69
\\48        0013                           00100555         0
\\49        0014                           00100555        69
\\50        0015                           00100555         0
\\51        0016                           00100555         0
…
```