

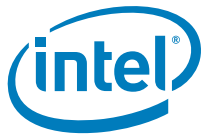
System Tools - Intel® Converged Security Engine Firmware 15.40

User Guide

August 2021

Revision 1.12

Intel Confidential



You may not use or facilitate the use of this document in connection with any infringement or other legal analysis concerning Intel products described herein. You agree to grant Intel a non-exclusive, royalty-free license to any patent claim thereafter drafted which includes subject matter disclosed herein.

No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document.

Intel technologies' features and benefits depend on system configuration and may require enabled hardware, software or service activation. Performance varies depending on system configuration. **No computer system can be absolutely secure.** Check with your system manufacturer or retailer or learn more at intel.com.

Intel technologies may require enabled hardware, specific software, or services activation. Check with your system manufacturer or retailer.

The products described may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Intel disclaims all express and implied warranties, including without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement, as well as any warranty arising from course of performance, course of dealing, or usage in trade.

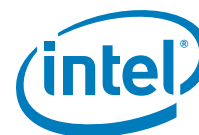
All information provided here is subject to change without notice. Contact your Intel representative to obtain the latest Intel product specifications and roadmaps.

Copies of documents which have an order number and are referenced in this document may be obtained by calling 1-800-548-4725 or visit www.intel.com/design/literature.htm.

Intel, Thunderbolt and the Intel logo are trademarks of Intel Corporation in the U.S. and/or other countries.

*Other names and brands may be claimed as the property of others.

Copyright © 2021, Intel Corporation. All Rights Reserved.



Contents

1	Introduction	8
1.1	Terminology	8
1.2	Reference Documents	14
2	Preface	15
2.1	Overview	15
2.2	Image Editing Tools	15
2.3	Manufacturing Line Validation Tool	15
2.4	Intel® Converged Security Engine Setting Checker Tool	16
2.5	Operating System Support	16
2.6	Generic System Requirements	17
2.7	Error Return	17
2.8	Usage of Double-Quote Character (")	17
2.9	Control Handler Support	18
3	Intel® Flash Image Tool	19
3.1	System Requirements	19
3.2	Flash Image Details	19
3.2.1	Flash Space Allocation	20
3.3	Required Files	20
3.4	Intel® Flash Image Tool	21
3.4.1	Configuration Files	21
3.4.2	Creating New Configuration	21
3.4.3	Opening Existing Configuration	21
3.4.4	Saving Configuration	21
3.4.5	Environment Variables	21
3.4.6	Modifying the Flash Descriptor Region	24
3.4.7	Descriptor Region Length	24
3.4.8	Setting the Number and Size of the Flash Components	25
3.4.9	SPI Software Binding (PCH Replacement)	26
3.4.10	Region Access Control	26
3.4.11	VSCC Table	30
3.4.12	Adding New Table	30
3.4.13	Removing Existing VSCC Table	31
3.4.14	FPF Configuration	32
3.4.15	Modifying the Intel® Converged Security Engine Region	32
3.4.16	Setting the Intel® Converged Security Engine Region Binary File	32
3.4.17	Intel® Converged Security Engine Section	33
3.4.18	Power	33
3.4.19	Platform Protection	34
3.4.20	Modifying PDR Region	35
3.4.21	Setting PDR Region Length Option	35
3.4.22	Setting PDR Region Binary File	35
3.4.23	Enabling/Disabling PDR Region	35
3.4.24	Modifying BIOS Region	36
3.4.25	Setting BIOS Region Length Parameter	36
3.4.26	Setting the BIOS Region Binary File	36
3.4.27	Enabling/Disabling the BIOS Region	36
3.4.28	Building Flash Image	37
3.4.29	Decomposing Existing Flash Image	37
3.4.30	Command Line Interface	38
3.4.31	Example – Decomposing Image and Extracting Parameters	41



	3.4.32 More Examples of FIT CLI	41
4	Flash Programming Tool	43
4.1	System Requirements	43
4.2	Flash Image Details	44
4.3	Microsoft Windows® Required Files	44
4.4	EFI Required Files	44
4.5	Programming Flash Device	45
4.5.1	Stopping Intel® CSE SPI Operations	45
4.6	Programming NVARs	45
4.7	Usage	46
4.8	Fparts.txt File	52
4.9	Examples	53
4.9.1	Complete SPI Flash Device with Binary File	53
4.9.2	Program Specific Region	53
4.9.3	Program SPI Flash from Specific Address	55
4.9.4	Dump Full Image	55
4.9.5	Dump Specific Region	55
4.9.6	Display SPI Information	56
4.9.7	Verify Image with Errors	57
4.9.8	Verify Image Successfully	57
4.9.9	Get Intel® CSE settings	58
4.9.10	CVAR Configuration File Generation (-cfggen)	58
5	Intel® ME Manuf and MEmanufWin	63
5.1	Windows® PE Requirements	63
5.2	How to Use Intel® ME Manuf	63
5.3	Usage	63
5.3.1	Host based Tests	67
5.4	Intel® MEmanuf -EOL Check	68
5.4.1	ErrorAction Field	68
5.4.2	MEmanuf.xml File	68
5.4.3	MEmanuf -EOL Variable Check	145
5.4.4	MEmanuf -EOL Config Check	145
5.4.5	Output/Result	146
5.5	Examples	146
6	Intel® ME Info	149
6.1	Windows® PE Requirements	149
6.2	Usage	149
6.3	Examples	160
6.3.1	ME Info Sample Output	161
6.3.2	Retrieve Current Value of Flash Version	163
6.3.3	Checks Whether Computer Has Completed Set-up and Configuration Process	163
7	Intel® CSE Firmware Update	164
7.1	Requirements	164
7.2	Enabling and Disabling Intel® FW Update	165
7.3	FW Update Flows	165
7.3.1	Full FW Update	165
7.3.2	Partial FW Update	165
7.4	Usage	165
7.5	Examples	167
7.5.1	Updates Intel® CSE with Firmware Binary File	167
7.5.2	Partial Firmware Update	167
7.5.3	Display Supported Commands	168



7.5.4	Language Codes.....	169
8	UEFI Sample Application Leveraging FW Update API Library	170
8.1	Getting Started - FW UpdateFW Update Library	170
8.1.1	Introduction	170
8.1.2	Environment.....	170
8.1.3	Setup	170
8.1.4	Sample App.....	170
8.2	Function Description	181
8.2.1	Get Interfaces.....	181
8.2.2	Get Last Status	181
8.2.3	Get Last Update Reset Type.....	181
8.2.4	Check Policy	182
8.2.5	Check Policy Buffer.....	182
8.2.6	Verify OEM Id	183
8.2.7	Get Ipu Partition Attributes.....	183
8.2.8	Get FW Update Info Status	184
8.2.9	FW Update Query Status Get Response	184
8.2.10	FW Update Full – Using Buffer.....	185
8.3	FW Update Partial Buffer	186
8.3.1	PDT Data (Sensor Calibration Data) Update	187
8.3.2	ISH Firmware Version	187
9	Intel® Manifest Extension Utility (Intel® MEU)	188
9.1	Usage	188

Figures

3-1	SPI Flash Image Regions.....	20
3-2	Environment Variables Dialog	22
3-3	Build Settings Dialog	24
3-4	Descriptor Region Length Parameter.....	25
3-5	Flash Settings > Flash Components	25
3-6	Flash Settings > Flash Configuration.....	26
3-7	Descriptor Region Master Access Section.....	30
3-8	Add VSCC Table Entry Dialog	31
3-9	Deleting VSCC Table Entry Dialog.....	32
3-10	Intel® CSE Kernel.....	33
3-11	Power.....	34
3-12	Platform Protection Section	34
3-13	PDR Region Options	35
3-14	BIOS Region Parameters	36

Tables

2-1	OS Support for Tools	16
2-2	Tools Summary.....	17
3-1	Flash Image Regions – Description	20
3-2	Build Settings Dialog Options	23
3-3	Region Access Control Table	26
3-4	CPU/BIOS Access	28
3-5	FIT Command Line Options.....	38
4-1	FPT OS Requirements	44
4-2	Named Variables Options	46
4-3	Command Line Options for fpt.efi, fpt.exe and fptw.exe.....	47
4-4	FPT–closemn Behavior	52



4-5	Intel-Recommend Access Settings.....	52
5-1	Options for ME Manuf	64
5-2	Intel® ME Manuf Test Matrix	67
5-3	MEManuf - EOL Config Tests	145
6-1	Intel® ME Info Command Line Options	149
6-2	List of Components that Intel® ME Info Displays.....	151
7-1	Image File Update Options	166



Revision History

0.6	<ul style="list-style-type: none">Initial release	January 2019
1.0	<ul style="list-style-type: none">Beta release<ul style="list-style-type: none">Updated tool outputs and syntax	April 2020
1.1	<ul style="list-style-type: none">Added a new section 2.4 Generating Config files or log file using toolsAdded details about FW Update flow under 7.3.1 Full FW UpdateAdded important note under Section 4.5 Programming Flash Device	July 2020
1.12	<ul style="list-style-type: none">Removed Erase and Blank commands from FPTAdded details about configuring master access permissions in FITAdded details about Linux requirements for FPT tool	August 2021

§ §



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
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
BIST	Built In Self-Test
CCM	Client Control Mode (Host Based Setup and Configuration)
CLI	Command Line Interface
CM0	Intel® CSE power state where all HW power planes are activated. Host power state is S0.
CM1	Intel® CSE 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 PCI-E* interface is unavailable to the host SW. This power state is not available in Cougar Point.
CM3	Intel® CSE 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 PCI-E* interface is unavailable to the host SW. The main memory is not available for Intel® CSE use.
CM-Off	No power is applied to the management processor subsystem. Intel® CSE is shut down.
CRB	Customer Reference Board
DHCP	Dynamic Host Configuration Protocol
DIMM	Dual In-line Memory Module
DLL	Dynamic Link Library
DNS	Domain Naming System



Acronym/Term	Definition
EC	Embedded Controller
EEPROM	Electrically Erasable Programmable Read Only Memory
EFI	Extensible Firmware Interface
EHCI	Enhanced Host Controller Interface
EID	Endpoint ID
End User	The person who uses the computer (either Desktop or Mobile).
EOP	End Of Post
FCIM	Full Clock Integrated Mode
FCSS	Flex Clock Source Select
FDI	Flexible Display Interface
FLOCKDN	Flash Configuration Lock-Down
FMBA	Flash Master Base Address
FOV	Fixed Offset Variable
FPSBA	Flash PCH Strap Base Address
FQDN	Fully Qualified Domain Name
FRBA	Flash Region Base Address
FW	Firmware
FW Update	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
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® CSE FW.
Host Service/ Application	An application running on the host CPU
HostIF	Host Interface
HTTP	Hyper Text Transfer Protocol
HW	Hardware
IBEN	Input Buffer Enable



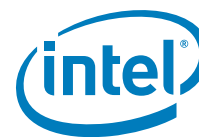
Acronym/Term	Definition
IBV	Independent BIOS Vendor
ICC	Integrated Clock Configuration
ID	Identification
INF	An information file (.inf) used by Microsoft operating systems that support the Plug and Play feature. When installing a driver, this file provides the OS with the necessary information about driver filenames, driver components, and supported hardware.
Intel® DAL	Intel® Dynamic Application Loader (Intel® DAL)
Intel® FIT	Intel® Flash Image Tool
Intel® FPT	Intel® Flash Programming Tool
Intel® CSE	Intel® Converged Security Engine. The embedded processor residing in the chipset PCH.
Intel®MEI driver	Intel® AMT host driver that runs on the host and interfaces between ISV Agent and the Intel® AMT HW.
Intel®ME Info	Intel® Manageability Engine Information Tool to check whether CSE is alive or not.
Intel®MEInfoWin	Windows* version of Intel® Manageability Engine Information Tool
Intel®ME Manuf	Intel® Manageability Engine Manufacturing Tool validates Intel® CSE functionality on the manufacturing line
Intel®MEManufWin	Windows* version of Intel® Manageability Engine Manufacturing Tool
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
LAN	Local Area Network
LED	Light Emitting Diode
LPC	Low Pin Count Bus
MAC address	Media Access Control address
MCP	Multi-Chip Package (Central Processing Unit / Platform Controller Hub)
NM	Number of Masters
NVAR	Named Variable
NVM	Non-Volatile Memory



Acronym/Term	Definition
NVRAM	Non-Volatile Random Access Memory
OCKEN	Output Clock Enable
ODM	Original Device Manufacturer
OEM	Original Equipment Manufacturer
OEM ID	Original Equipment Manufacturer Identification
OS	Operating System
OS Hibernate	OS state where the OS state is saved on the hard drive.
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
PAVP	Protected Video and Audio Path
PC	Personal Computer
PCH	Peripheral 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
RCFG	Remote Configuration
RCS	Remote Connectivity Service
RNG	Random Number Generator
ROM	Read Only Memory
RPAS	Remote Connectivity Service
RSA	A public key encryption method
RTC	Real Time Clock



Acronym/Term	Definition
S0	A system state where power is applied to all HW devices and the system is running normally.
S0ix	Connected Standby
S5	A system state where all power to the host system is off but the power cord is still connected.
SDK	Software Development Kit.
SEBP	Single Ended Buffer Parameters
SHA	Secure Hash Algorithm
SMB	Small Medium Business mode
SMBus	System Management Bus
Snooze mode	Intel® CSE activities are mostly suspended to save power. Intel® CSE monitors HW activities and can restore its activities depending on the HW event.
SPI	Serial Peripheral Interface
SPI Flash	Serial Peripheral Interface Flash
Standby	OS state where the OS state is saved in memory and resumed from the memory when the mouse/keyboard is clicked.
SW	Software
Sx	All S states which are different than S0
System States	Operating System power states such as S0, S0ix, and S5.
TCP/IP	Transmission Control Protocol/Internet Protocol.
TLS	Transport Layer Security
UEP	Unified Emulation Partition
UI	User Interface
UIM	User Identifiable Mark
UMA	Unified Memory Access
Un-configured state	The state of the Intel® CSE FW when it leaves the OEM factory. At this stage the Intel® CSE FW is not functional and must be configured.
UPDPARAM	Update Parameter Tool
USB	Universal Serial Bus
UUID	Universally Unique Identifier
VLAN	Virtual Local Area Network
VSCC	Vendor Specific Component Capabilities
Windows* PE	Windows* Pre installation Environment
WIP	Work in Progress



Acronym/Term	Definition
WLAN	Wireless Local Area Network
XML	Extensible Markup Language.



1.2 Reference Documents

Document	Document Location
FW Bring Up Guide	Release kit
Firmware Variable Structures for Intel® Converged Security Engine	CCL document
PCH EDS	CCL
Elkhart Lake SPI Programming Guide	Release kit
ISS Firmware Bring Up Guide	CCL

§ §



2 Preface

2.1 Overview

This document covers the system tools used for creating, modifying, and writing binary image files, manufacturing testing, Intel® CSE setting information gathering, and Intel® CSE FW updating. The tools are located in

KIT\External_15.40.x.xxxx\Tools\System tools. For information about other tools, refer Tool's user guides in the other directories in the FW release.

The system tools described in this document are platform specific in the following ways:

- Elkhart Lake Converged Mobility platform – All of the tools in the PCH Elkhart Lake FW release kit are designed for 1st Generation Intel® Converged Mobility Processors and Elkhart Lake Converged Mobility platforms only. These tools do not work properly on any other legacy platforms (prior Generations of Intel® Processors). Tools designed for other platforms also do not work properly on the 1st Generation Intel® Converged Mobility Processors or the Elkhart Lake Converged Mobility platform.

2.2 Image Editing Tools

The following tools create and write flash images:

- **Intel® FIT**
Combines the Descriptor, BIOS, PDR and Intel® CSE FW binaries into one image. Configures soft straps and NVARs for Intel® CSE settings and another for outputs that can be programmed by a flash programming device or the FPT Tool.
- **Intel® FPT:**
Programs the SPI flash memory of individual regions or the entire flash device. Modifies some Intel® CSE settings (NVAR), FPFs after Intel® CSE is flashed on the SPI/UFS part.

Note: To flash the firmware image on the UFS device you must use the Intel® Platform Flash Tool. See the Platform Flash Tool User Guide for details.

- **FW Update** updates the Intel® CSE FW code region on a flash device that has already been programmed with a complete image.

Note: The firmware update tool provided by Intel only works on the platforms that support the FW Update feature.

2.3 Manufacturing Line Validation Tool

The manufacturing line validation tool (Intel® ME Manuf) allows the Intel® CSE functionality to be tested immediately after the PCH chipset is generated. This tool is designed to be able to run quickly and is generally run on the manufacturing line to do manufacturing testing.



Note: Programming the flash or running any of the FPT commands has a limitation on the path length to 128 Characters.

2.4 Generating Config files or log file using tools

To ensure no impact to OS system directories or files, Intel Tools prevents generating or creating config files and log files into OS System directories. Files generated from tools can be Intel® FPT & Intel® ME Manuf Configurations files as well as log files created by running -verbose command in Intel® ME Info, Intel® ME Manuf, and Intel® FPT.

The below table displays the directories that tools will not permit specifying them as path targets for generating any sort of file:

Table 2-1. OS directories unsupported by tools

Windows® OS	Linux
C:\Program Files	"/sbin"
C:\Program Files (x86)	"/bin"
C:\Windows	"/etc"
C:\ProgramData	"/boot"
	"/lib"
	"/srv"
	"/sys"
	"/usr"
	"/var" (except "/var/tmp")

2.5 Intel® Converged Security Engine Setting Checker Tool

The Intel® CSE setting checker tool (Intel®ME Info) retrieves and displays information about some of the Intel® CSE settings, the Intel® CSE FW version, and the FW capability on the platform.



2.6 Operating System Support

Table 2-1. OS Support for Tools

Intel® CSE and Manufacturing Tools	UEFI (64 bit)	Windows* 10 DT 64 bit	Linux* (64 bit)	Windows PE for Windows 10
Intel® Flash Programing Tool	X	X		x
Intel® ME Manuf Tool	X	X		x
Intel® ME Info Tool	X	X		x
Intel® Firmware Update Tool	X	X		x
Intel® Manifest Extension Utility Tool		X	x	
Intel® Flash Image Tool		X	x	

Notes:

1. 64 bit support may NOT mean that a tool is compiled as a 64 bit application – but that it can run as a 32 bit application on a 64 bit platform.
2. The Windows* 64 bit tools will not function when the OS is configured to use EFI / GPT boot capabilities.
3. Currently the System Tools use the EDK II Development Kit exclusively.

2.7 Generic System Requirements

The installation of the following services is required by integration validation tools that run locally on the system under test with the Intel® Converged Security Engine:

- Intel® MEI driver.
- Intel® SPD driver

Refer the description of each tool for its exact requirements.

Table 2-2. Tools Summary

ToolName	Feature Tested	Runs on Intel® CSE device
Intel® ME Manuf and Intel® MEManufWin	Connectivity between Intel® CSE Devices	X
Intel® ME Info and Intel® MEInfoWin	Firmware Aliveness – outputs certain Intel® CSE parameters	X
Intel® FPT	Programs the image onto the flash memory and Programming NVARs / FPPFs	X
Intel® FW Update	Updates the FW code while maintaining the previously set values	X

2.8 Error Return

Tools will return errors differently depending on the Operating System the tools run on:

- For Linux*:
 - o tools return the error category only.
- For Windows*:
 - o The first 8 bits will be the error category, and the rest of the bits will represent the error code.

For example, an error with *error code* = 29 (*i.e.* 0001 1101), and *error category* = 11 (*i.e.* 0000 1011). In Linux, tools will return the Error Category only which is 11. In Windows*, tools will return the combination of ErrorCode and the ErrorCategory (0001 1101 0000 1011) which is 7435.

2.9 Usage of Double-Quote Character (")

The EFI version of the tools handle multi-word argument differently than the Windows* version. If there is a single argument that consists of multiple words delimited by spaces, the argument needs to be entered as following:

FPT.efi -f "" well power config"".

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

The command shell was 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, the user 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 the user wants these words to be input – input"string – the command line is: input\"string.

2.10 Control Handler Support

Intel®ME Info and Intel® FPT and Intel®ME Manuf support control handlers (Ctrl + C, Ctrl + Break, Ctrl + Close, etc.) for supported Microsoft Windows versions. When the control handlers are invoked, upon the following execution of the tools (after the 1st execution was aborted by the above control handlers), the tools will execute their regular flows.



3 Intel® Flash Image Tool

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

1. For SPI images, FIT creates and allows configuration of the Flash Descriptor Region, which contains configuration information for platform hardware and FW.
2. FIT assembles the following into a single image:

Binary files of the following regions:

- BIOS
- Descriptor Region
- SubPartitions
- IFWI: Intel®ME and PMC
- Platform Descriptor Region
- PDR Region

The Flash Descriptor Region created by FIT (For SPI images)

3. The user can manipulate the completed image via a GUI and change the various chipset parameters to match the target hardware. Various configurations can be saved to independent files, so the user does not have to recreate a new image each time. FIT supports a set of command line parameters that can be used to build an image from the CLI or from a makefile. When a previously stored configuration is used to define the image layout, the user does not have to interact with the GUI.

Note: FIT just generates a complete image file; it does not program the flash device. This complete image must be programmed into the flash with FPT (SPI Only) Platform Flash Tool (UFS) or any third-party flash burning tool, or some other flash burner device.

Note: Disable Intel®DnX features while boot media is UFS will disable the option to do provisioning or recovery to IFWI in close chassis

Note: For FIT to generate a complete image file; Dekel PHY sub partition binary should be provided for FIT to include in the final image file. Failure to do so would result in FIT tool returning an error and failing to build the final image.

3.1 System Requirements

Intel® FIT runs on Microsoft Windows* 10. The tool does not have to run on an Intel® CSE-enabled system.

3.2 Flash Image Details

A flash image is composed of four 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 3-1. SPI Flash Image Regions

Descriptor	IFWI: Intel® CSE amd PMC Intel® CSE Applications	Sub - part itio ns	GbE	PDR	BIOS
------------	---	--------------------------------	-----	-----	------

Table 3-1. Flash Image Regions – Description

Region	Description
Descriptor	<p>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.</p> <p>Note: This region MUST be locked before the serial flash device is shipped to end users. Refer to Section 3.4.10 below for more information. Failure to lock the Descriptor Region leaves the Intel® CSE device vulnerable to security attacks.</p>
Ifwi: Intel® CSE and PMC	This region contains code and configuration data for Intel® CSE applications. It takes up a variable amount of space at the end of the Descriptor.
GBE	This region contains code and configuration data for an Intel Integrated LAN (Gigabit Ethernet). It takes up a variable amount of space at the end of the Intel® CSE region.
BIOS	This region contains code and configuration data for the entire computer.
PDR	This region lets system manufacturers describe custom features for the platform.
Sub-Partitions	This region contains IUnit and PCH configuration binaries that will be merged into the output image generated by IntelFIT tool

3.2.1 Flash Space Allocation

Space allocation for each region is determined as follows:

1. 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.
2. If there is still space left in the flash after allocating space to all of the regions, the Intel® CSE region expands to fill the remaining space.



3.3 Required Files

The FIT main executable is **FIT.exe**. The following files must be in the same directory as **FIT.exe**:

- vsccommn.bin
- .xml file

3.4 Intel® Flash Image Tool

Refer following for further information:

- General configuration information – Refer FW Bring Up Guide from the appropriate Intel® CSE FW kit.
- Detailed information on how to configure PCH Soft Straps and VSCC information – Refer to the Elkhart Lake PCH SPI Programming Guide within the kit.

3.4.1 Configuration Files

The flash image can be configured in many different ways, depending on the target hardware and the required FW options. FIT lets the user 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.

3.4.2 Creating New Configuration

FIT provides a XML configuration file template that will help the user create their own configuration XML. This template configuration XML file can be created by clicking **File > New and then save**. It can also be created from the command line using **-save** option.

3.4.3 Opening Existing Configuration

To open an existing configuration file:

1. Choose File → **Open**; **Open File** dialog appears.
2. Select the XML file to load.
3. Click Open.

Note: The user can also open a file by dragging and dropping a configuration file into the main window of the application.

3.4.4 Saving Configuration

To save the current configuration in an XML file:

Choose File → **Save** 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.

1. Select the path and enter the file name for the configuration.
2. Click Save.



3.4.5 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. The user can set the environment variables appropriate for the platform being used, or override the variables with command line options.


It is recommended that the environment variables be the first thing that the user sets when working with a new configuration. This ensures that FIT 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:

1. Choose Build → **Build Settings**; a dialog appears displaying the current working directory on top, followed by the current values of all the environment variables:
 - \$WorkingDir – the directory functions as a basic path variable when modified in the GUI. If \$WorkingDir CLI flag is used when launching FIT GUI, then the fit.log will be created in \$WorkingDir directory.
 - \$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 intermediate files generated during the build. Also the directory where the components of an image are stored when an image is decomposed.
 - \$UserVar1-3 – used when the above variables are not populated.

Figure 3-2. Environment Variables Dialog

▼ Environment Variables		
Parameter	Value	
\$WorkingDir	.	Path for environment variable
\$SourceDir	.	Path for environment variable
\$DestDir	.	Path for environment variable
\$UserVar1	.	Path for environment variable
\$UserVar2	.	Path for environment variable
\$UserVar3	.	Path for environment variable

2. Press the  button next to an environment variable and select 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.
3. Repeat Step 2 until the directories of all relevant environment variables have been defined.
4. Click
5. **OK**.

Note: The environment variables are saved in the XML file. They can be overridden on the command line if using the XML file on multiple systems.

Note: Build Settings
FIT lets the user set several options that control how the image is built. The options that can be modified are described in Build Settings Dialog Options.

To modify the build setting:

1. Choose **Build** → **Build Settings**; a dialog appears showing the current build settings.
2. Modify the relevant settings in the **Build Settings** dialog.
3. 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.
FW Update Output Path	The path and filename where the final firmware should be saved after it is Updated
Generate intermediate build files.	Causes the application to generate separate (intermediate) binary files for each region, in addition to the final image file (Refer Figure 3). These files are located in the specified output folder's INT subfolder. These image files can be programmed individually with the FPT.
Enable Boot Guard Warning message at build time.	Allows to enable boot guard warning messages at the build time.
Enable Intel® Platform Trust Technology messages at build time.	Allows to enable Intel® Platform Trust Technology warning messages at the build time
Region Order	Default value is 5241 when 1=BIOS , 2=ME/IFWI , 3=GbE , 4=PDR
IfwiBuildVersion	32-bit value to use as the IFWI build version number

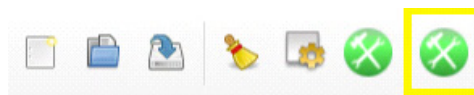


Redundancy Enabled	Enables redundancy support for critical layout components default value is false
Intel® ManifestExtension Utility	The path and filename where the final Manifest Extension Utility should be saved
Signing Tool Path	The path and filename where the final Signing Tool should be saved
Signing Tool	Signing tools used to build the image
Default Data Partition Enabled	Enable Intel® CSE default data partition , the default value is false
Build FW Update With Full Image	Allows for the FW Update image to be built along with the final full image. If set to no, then a full image will be created only.

Figure 3-3. Build Settings Dialog

▼ Image Build Settings		
Parameter	Value	
Output Path	\$DestDir\outimage.bin	-
FWUpdate Output Path	\$DestDir\FWUpdate.bin	-
Build FWUpdate With Full Image	No	-
Generate Intermediate Files	Yes	-
Enable Boot Guard warning me...	Yes	-
Enable Intel (R) Platform Trust ...	Yes	-
Region Order	53241	1=BIOS, 2=...
Target Type	SPI	Select target
IfwiBuildVersion	0x0	32-bit value
Redundancy Enabled	false	Enable Redu
Default Data Partition Enabled	false	Enable CSE I
Intel(R) Manifest Extension Utili...		-
Signing Tool Path		-
Signing Tool	OpenSSL	-

Note: Intel® FIT tool has the ability to build images meant for FW Update purposes. To do so, click on the build icon as marked below. This action would build a FW Update image only and save it in the earlier defined path in the Build Settings Dialog.





3.4.6 Modifying the Flash Descriptor Region

The Flash Descriptor Region 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 computer may not function as expected. This region also needs to be configured correctly in order to ensure that the system is secure.

3.4.7 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:

1. Select **Flash Layout** in the left pane; the **Length** parameter appears in the right pane.
2. 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

The screenshot shows a software window titled "Descriptor Region" with a dropdown arrow. Below the title is a table with two columns: "Parameter" and "Value". The table contains one row with the parameter "OEM Section Binary" and its value "This loads the OEM Section binary that will be merged into th".

Parameter	Value
OEM Section Binary	This loads the OEM Section binary that will be merged into th

3.4.8 Setting the Number and Size of the Flash Components

To set the number of flash components:

1. Select **Flash Settings** in the left pane; expand the Flash Components node in the right pane.

Refer to [Figure 3-5](#), the parameters in the Flash Component section are listed in the right pane.

Figure 3-5. Flash Settings > Flash Components

Flash Components	
Parameter	Value
Number of Flash Components	1
Flash component 1 Size	16MB
Flash component 2 Size	8MB
SPI Global Protected Range	0x0
SPI Idle to Deep Power Down T...	0x5
SPI Out of Order operation Ena...	Yes
SPI Resume Hold-off Delay	8us
SPI Max write / erase Resume ...	No Ceiling
SPI Suspend / Resume Enabled	Yes
Software Re-Binding Enabled	No

- Double-click the value of **Number of Flash Components** in the right pane (Figure 3-5)
- Select the number of flash components (valid values are 0,1 or 2) from the dropdown.

To set the size of each flash component:

- Double-click on the value of one of these parameters Flash Component 0/1 Size
Flash Component 2 Size.
- Select the correct component size from the drop-down list; that parameter is updated.
- Repeat steps 2-3 for the other parameter.

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

3.4.9 SPI Software Binding (PCH Replacement)

When enabled, the Flash Component's "SPI Software Binding Enabled" parameter will allow for SPI re-binding to a new PCH during manufacturing and remanufacturing flows prior to platform EOM.

Note: Note: Re-binding to a replacement PCH can only be done a maximum of 5 times before the SPI part needs to be re-flashed. The replacement counter is exposed in the PCH section of ME Info.

Figure 3-6. Flash Settings > Flash Configuration

▼ Flash Configuration		
Parameter	Value	
Dual I/O Read Enable	No	This soft-strap only has effect if I
Dual Output Read Enable	No	This soft-strap only has effect if I
Fast Read Clock Frequency	48MHz	This setting allows customers to c
Fast Read Supported	Yes	This setting allows customers to c
Invalid Instruction 0	0x21	This setting allows customers to c
Invalid Instruction 1	0x42	This setting allows customers to c
Invalid Instruction 2	0x60	This setting allows customers to c
Invalid Instruction 3	0xAD	This setting allows customers to c
Invalid Instruction 4	0xB7	This setting allows customers to c
Invalid Instruction 5	0xB9	This setting allows customers to c
Invalid Instruction 6	0xC4	This setting allows customers to c
Invalid Instruction 7	0xC7	This setting allows customers to c
Quad I/O Read Enable	No	This soft-strap only has effect if C
Quad Output Read Enable	No	This soft-strap only has effect if C
Read ID and Read Status Clock ...	48MHz	This setting allows customers to c
Write and Erase Clock Frequency	48MHz	This setting allows customers to c

3.4.10 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® CSE devices are shipped. If the Descriptor Region is not locked, the Intel® CSE 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.

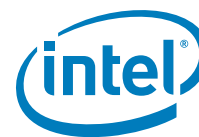
The Master access permission configurations in FIT have different behavior between pre-lock and post-lock:

1. Pre-lock enabled: Master Access Permissions will be set according the configurations set in FIT
2. Post-lock enabled: Master Access Permissions set in FIT will be discarded when EOM is triggered. To configure Master Access permissions with post-lock, users must use the configuration file provided by FPT.



Table 3-3. Region Access Control Table

Master Read/Write Access				
Region (#)	CPU and BIOS	CSE/PCH	EC	Gbe
Descriptor (0)	Not Accessible	Not Accessible	Not Accessible	Not Accessible
BIOS (1)	CPU and BIOS can always read from and write to BIOS region	Read / Write	Read / Write	Read / Write
CSE (2)	Read / Write	ME can always read from and write to CSE region	Read / Write	Read / Write
GbE (3)	Read / Write	Read / Write	Read / Write	GbE software can always read from and write to GbE region
PDR (4)	Not Accessible	Not Accessible	Not Accessible	Not Accessible
EC - Embedded Controller (Optional) (8)	Read / Write	Read / Write	EC can always read from and write to EC region	Read / Write
SubPartitions	Read / Write	PCH Configuration can always read from and write to SubPartition region	Read / Write	Read / Write
NOTES:				
1. Descriptor and PDR region is not a master, so they will not have Master R/W access.				
2. Descriptor should NOT have write access by any master in production systems.				
3. PDR region should only have read and/or write access by CPU/Host. GbE and CSE should NOT have access to PDR region.				



		Regions That Can Be Accessed					
		PDR	Intel® CSE	Gbe	BIOS	IOSF Sideband Privileged Master	Descriptor
Region to Grant Access	Intel® CSE	None/Read/Write	None/Read/Write	Write only. Intel® CSE can always read from and write to Intel® CSE Region	None/Read/Write	None/Read/Write	None/Read/Write
	Gbe	None/Read/Write	Write only. GbE can always read from and write to GbE Region.	None/Read/Write	None/Read/Write	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	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.

Key:

0 – Denied access

1 – Allowed access

NC – Bit may be either 0 or 1 since it is unused.

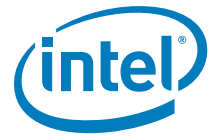
Table 3-4. CPU/BIOS Access

Read Access								
	Unused			PDR	GBE	Intel® CSE	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®CSE	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 PDR and Intel® CSE and write access to Intel®ME, then the bits are set to:

Read Access – 0b 0001 0110 (0x 0E in hexadecimal).

Write Access – 0b 0000 0110 (0x 06 in hexadecimal).

To set these access values in FIT:

1. Select **Flash Settings Tab → Host CPU/BIOS Master Access, Intel ME Master Access, Gbe Master Access and EC Master Access** in the right pane; the access parameters are listed in the right pane.
2. Double-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 0xFFFF for both read and write access in all three sections.

To generate a production image with BIOS access to the PDR region select
read access 0x00F / 0x01F and write access 0x00A / 0x01A.

Note: These settings should only be used if the PDR region is implemented.

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

Figure 3-7. Descriptor Region Master Access Section

▼ Host CPU / BIOS Master Access		
Parameter	Value	
Host CPU / BIOS Write Access I...	0xFFFF	This setting determines write access control for t
Host CPU / BIOS Write Access ...	0x0000	This setting determines write access control for t
Host CPU / BIOS Read Access I...	0xFFFF	This setting determines read access control for th
Host CPU / BIOS Read Access C...	0x0000	This setting determines read access control for th
▼ Intel(R) ME Master Access		
Parameter	Value	
Intel(R) ME Write Access Intel ...	0xFFFF	This setting determines write access control for t
Intel(R) ME Write Access Custom	0x0000	This setting determines read access control for th
Intel(R) ME Read Access Intel R...	0xFFFF	This setting determines read access control for th
Intel(R) ME Read Access Custom	0x0000	This setting determines read access control for th
▼ EC Master Access		
Parameter	Value	
Embedded Controller Read Acc...	0xFFFF	This setting determines read access control for th
Embedded Controller Read Acc...	0x0000	This setting determines read access control for th

3.4.11 VSCC Table

This section is used to store information to setup flash access for Intel®CSE. This does not have any effect on the usage of the FPT. **If the information in this section is incorrect, Intel® CSE FW may not communicate with the flash device.** The information provided is dependent on the flash device used on the system. (For more information, refer to the Elkhart Lake SPI Programming Guide, Section 6.4.)

VSCC Table can be accessed:

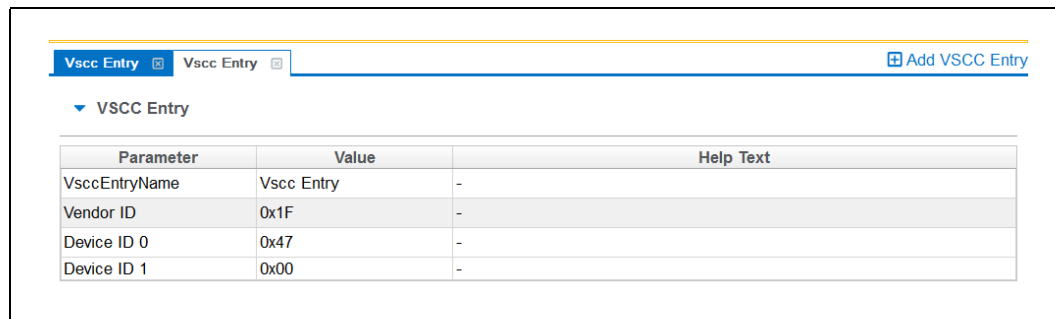
1. Select Flash Settings Tab on the left pan
2. Expand VSCC Entries on the right pan as shown below in [Figure 3-7](#):

3.4.12 Adding New Table

To add a new table:

1. Choose [Add VSCC Entry](#) on top left → **VSCC Entry**.

Figure 3-8. Add VSCC Table Entry Dialog



Parameter	Value	Help Text
VscEntryName	Vsc Entry	-
Vendor ID	0x1F	-
Device ID 0	0x47	-
Device ID 1	0x00	-

1. 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 FIT to prevent table entries that have the same name and no error message is displayed in such cases.

2. User can enter into the values for the flash device. ([Figure 3-7](#)), which shows the parameters of a new VSCC table.)

Note: The VSCC register value will be automatically populated by FIT using the vsccommn.bin file the appropriate information for the Vendor and Device ID.

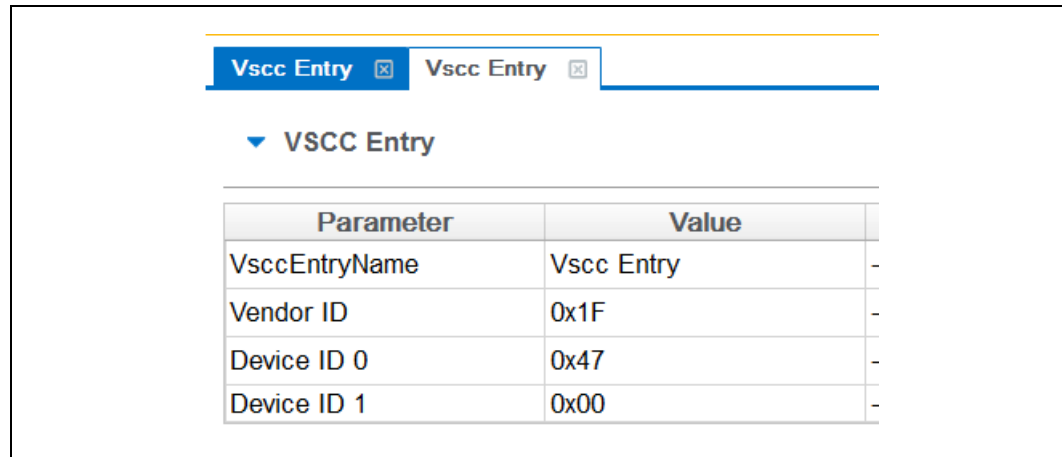
Note: If the descriptor region is being built manually the user will need to reference the VSCC table information for the parts being supported from the manufacturers' serial flash data sheet. The PCH-LP SPI Programming Guide should be used to calculate the VSCC values. For C620 family of workstation systems, use the LBG SPI Programming Guide for further reference concerning the VSCC table definitions.

3.4.13 Removing Existing VSCC Table

To remove an existing table:

1. Click on the name of the table in the top tab that the user wants to remove.

Figure 3-9. Deleting VSCC Table Entry Dialog



2. Click close, the table and all of the information will be removed.

3.4.14 FPF Configuration

The "FPF Hardware Binding Enabled" setting configures the FPF hardware binding behavior for the platform image.

For non-revenue parts:

If the "FPF Hardware Binding Enabled" setting is enabled
Hardware binding will occur when the close manufacturing flow is executed.

If the "FPF Hardware Binding Enabled" setting is disabled
Hardware binding will not occur when the close manufacturing flow is executed.

Note: *For Revenue parts this setting will be ignored and FPF Hardware binding will take place when close manufacturing flow is executed.*

3.4.15 Modifying the Intel® Converged Security Engine Region

The Intel® CSE Region contains all of the FW data for the Intel® CSE (including the Intel® CSE FW Kernel).

Note: Changing the Intel® CSE Region will prompt the user and require the users to reset parameters in Intel® FIT.

3.4.16 Setting the Intel® Converged Security Engine Region Binary File

To select the Intel® CSE region binary file:

1. Select the Intel® CSE Region available under Flash Layout tab on the left pane.

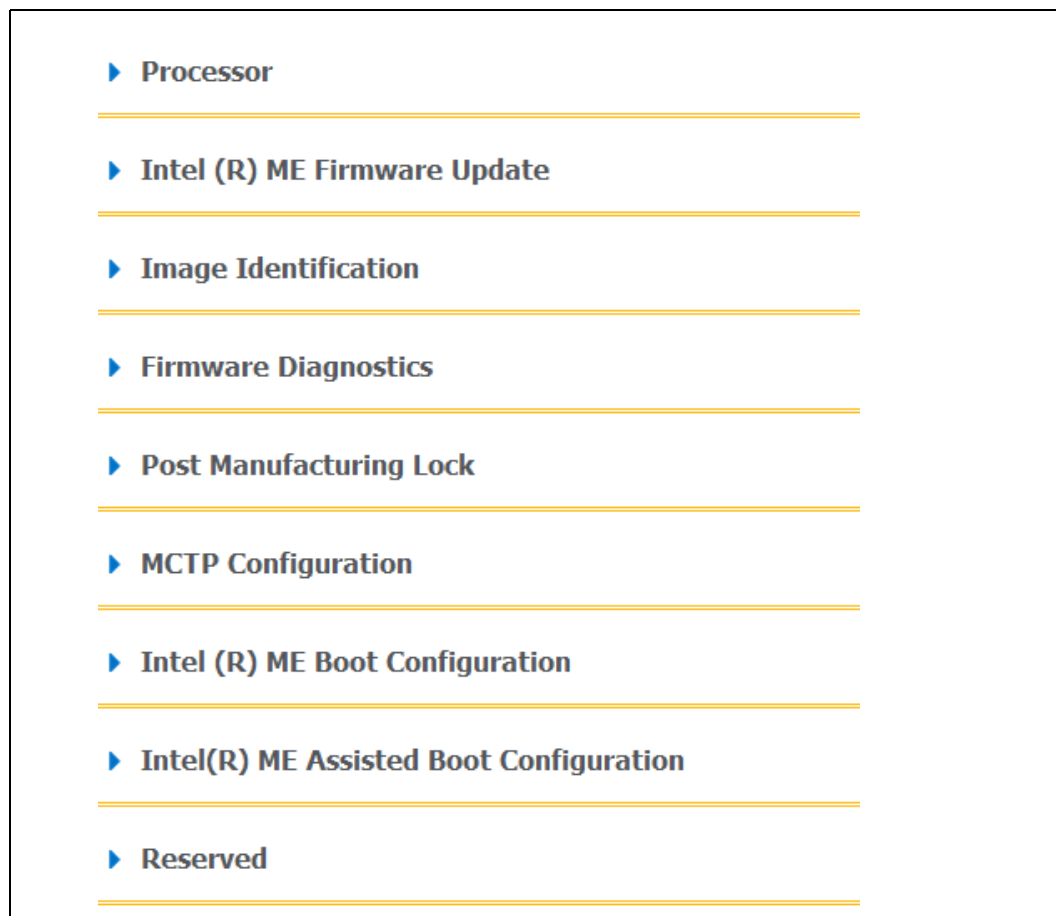
2. Double-click on the **Binary file parameter** in the list; select the Intel® CSE file to be used.
3. Click **OK** to update the parameter; when the flash image is built, the contents of this file is copied into the Intel® CSE Region.

3.4.17 Intel® Converged Security Engine Section

This section describes Intel® CSE FW Kernel parameters. (Refer FW Bringup guide for general information and refer Appendix for more details.)

Click on the Intel® CSE Kernel Tab on the left pane to configure the Intel® CSE parameters.

Figure 3-10. Intel® CSE Kernel



3.4.18 Power

This section describes the platform power configuration settings. Click on the Power tab on the left pane to configure power parameters.

Figure 3-11. Power

▼ Platform Power		
Parameter	Value	
SLP_S5# / GPD10 Signal Config...	Enable as SLP_S5#	This setting allows the user to assign
SLP_S3# / GPD4 Signal Configu...	Enable as SLP_S3#	This setting allows the user to assign
SLP_S4# / GPD5 Signal Configu...	Enable as SLP_S4#	This setting allows the user to assign
SLP_A# / GPD6 Signal Configur...	Enable as SLP_A#	This setting allows the user to assign
SLP_S0# Tunnel	Disabled	This setting Enables / Disables the tun
▼ Deep Sx		
Parameter	Value	
Deep Sx Enabled	Yes	This setting enables / disables suppor
▼ PchThermalReporting		

3.4.19 Platform Protection

The Platform Protection section determines which features are supported by the system. If a system does not meet the minimum hardware requirements, no error message is given when programming the image. (Refer to the FW Bringup guide for general information).

Figure 3-12. Platform Protection Section



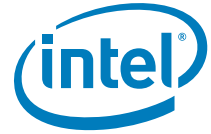
These options control the availability and visibility of FW features.

The ability to change certain options is SKU-dependent and – depending on the SKU selected – some of default values will be disabled and cannot be changed.

Note:

PCH SKU and FW SKU selection is not within the tool. It is based on the PCH SKU part that customer chooses and the FW SKU they load on that platform.

- Intel® Platform Trusted Technology
- Intel® Content Protection



3.4.20 Modifying PDR Region

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

Figure 3-13. PDR Region Options

▼ PDR Region		
Parameter	Value	Help Text
Length	0	-
PDR Binary File		-
PDR Region Enable	Disabled	-

3.4.21 Setting PDR Region Length Option

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](#).

3.4.22 Setting PDR Region Binary File

To select the PDR region binary file:

1. Click on Flash Layout tab on the left pane to load the binary file for PDR region
2. Click **OK** to update the parameter; when the flash image is built, the contents of this file is copied into the BIOS region.

3.4.23 Enabling/Disabling PDR Region

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

To disable the PDR Region:

1. Click Flash Layout tab on the left pane to load the binary file for PDR region.
2. Choose **Disable Region** from the drop down menu; when the flash image is built, there is no PDR Region in it.

Note: This region is disabled by default.

To enable the PDR Region:

1. Click on Flash Layout tab on the left pane to load the binary file for PDR region
2. Choose **Enable Region** from the drop down menu.

3.4.24 Modifying BIOS Region

The BIOS Region contains the BIOS code run by the host processor. 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 3-14. BIOS Region Parameters

▼ BIOS Region		
Parameter	Value	
Length	0	-
BIOS Binary File		This loads the BIOS binary

3.4.25 Setting 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 as described in [Section 3.2.1](#).

3.4.26 Setting the BIOS Region Binary File

To select the BIOS region binary file:

1. Click on Flash Layout tab on the left pane to load the binary file for BIOS region
2. Click **OK** to update the parameter; when the flash image is built, the contents of this file are copied into the BIOS region.

3.4.27 Building Flash Image

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

To build a flash image with the currently loaded configuration:

- Choose **Build > Build Image**.
- OR –
- Specify an XML file with the `/b` option in the command line.

FIT 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 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.)



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

3.4.28 Decomposing Existing Flash Image

FIT is capable of taking an existing flash image and decomposing it in order to create the corresponding configuration. This configuration can be edited in the GUI like any other configuration (refer below). A new image can be built from this configuration that is almost identical to the original, except for the changes made to it.

To decompose an image:

1. Chose **File** → **Open**.
2. Change the file type filter to the appropriate file type.
3. Select the required file and click **Open**; the image is automatically decomposed, the GUI is updated to reflect the new configuration, and a folder is created with each of the regions in a separate binary file.

Note: It is also possible to decompose an image by simply dragging and dropping the file into the main window. When decomposing an image, there are some NVARs will not be able to be decomposed by FIT. FIT will use Intel default value instead. User might want to check the log file to find out which NVARs were not parsed.

Note: The Intel®CSE region binary contained in INT folder after image generation only contains the firmware default base settings for CSE region no FIT customization is applied.

3.4.29 Command Line Interface

FIT supports command line options.

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

The command line syntax for FIT is:

```
FIT [-exp] [-h|?] [-version|ver] [-b] [-bfwu] [-ofwu] [-o] [-f]
    [-me] [-bios] [-pdr] [-sku] [-ec] [-gbe] [-iunit] [-rombypass]
    [-pmcp] [-ish] [-sd_token] [-iom] [-nphy] [-tbt] [-oem_km] [-w] [-s]
    [-d] [-u1] [-u2] [-u3] [-i] [-flashcount] [-flashsize1]
    [-flashsize2] [-save] [-set] [-cloHelp]
```

Following is the output generated when running FIT with the help option (`-h` or `?`):

```
Intel (R) Flash Image Tool. Version: 15.40.0.1039
Copyright (c) 2013 - 2020, Intel Corporation. All rights reserved.
4/27/2020 - 7:51:28 pm
=====
fit.exe [-exp] [-h|?] [-version|ver] [-b] [-bfwu] [-ofwu] [-o] [-f]
    [-me] [-bios] [-pdr] [-sku] [-ec] [-gbe] [-iunit] [-rombypass]
    [-pmcp] [-ish] [-sd_token] [-iom] [-nphy] [-tbt] [-oem_km] [-w] [-s]
```



[-d] [-u1] [-u2] [-u3] [-i] [-flashcount] [-flashsize1]
[-flashsize2] [-save] [-set] [-cloHelp]

-exp	Displays example usage of this tool.
-h ?	Displays help screen.
-version ver	Displays version of the tool.
-b	Build the flash image.
-bfwu	Build FW Update image.
-ofwu <filename>	Overrides the FW Update output file path.
-o <filename>	Overrides the output file path.
-f <filename>	Specifies input file. XML, full image binary, or ME only binary.
-me <file>	Overrides the binary source file for the ME region.
-bios <file>	Overrides the binary source file for the BIOS region.
-pdr <file>	Overrides the binary source file for the PDR region.
-sku <value>	Sets the SKU type to use.
-ec <file>	Overrides the binary source file for the EC region.
-gbe <file>	Overrides the binary source file for the GbE region.
-iunit <file>	Overrides the binary source file for the iUnit region.
-rombypass <true false>	Overrides the ROM bypass setting in the XML file.
-pmcp <file>	Overrides the binary source file for the PMCP region.
-ish <file>	Overrides the binary source file for the ISH region.
-sd_token <file>	Overrides the binary source file for the Secure Debug Token.
-iom <file>	Overrides the binary source file for the IOM region.
-nphy <file>	Overrides the binary source file for the North PHY region.
-tbt <file>	Overrides the binary source file for the TBT region.
-oem_km <file>	Overrides the binary source file for the OEM KM. override from cli enabled only in FW Update build
-w <path>	Overrides the \$WorkingDir environment variable.
-s <path>	Overrides the \$SourceDir environment variable.
-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.
-set <expression:filename>	Command line overridable key value parameter.
-cloHelp	Prints all available Command Line Overridable global paths and with path usage hint.

**Table 3-5. FIT Command Line Options**

Option	Description
<XML_file>	Used when generating a flash image file. A sample xml file is provided along with the FIT. When an xml file is used with the /b option, the flash image file is built automatically.
<Bin File>	Decomposes the BIN file. The individual regions are separated and placed in a folder with the same name as the BIN file.
-H or -?	Displays the command line options.
-version ver	Displays version of the tool.
-B	Automatically builds the flash image. The GUI does not appear if this flag is specified. This option causes the program to run in auto-build mode. If there is an error, a valid message is displayed and the image is not built. If a BIN file is included in the command line, this option decomposes it.
-bfwu	Builds FW Update image
-ofwu <filename>	overrides the FW Update output file path
-O <file>	Path and filename where the image is saved. This command overrides the output file path in the XML file.
-f <filename>	Specifies input file. XML, full image binary, or CSE only binary
-ME <file>	Overrides the binary source file for the Intel® CSE Region with the specified binary file.
-BIOS <file>	Overrides the binary source file for the BIOS Region with the specified binary file.
-PDR <file>	Overrides the binary source file for the PDR Region with the specified binary file.
EC	Overrides the binary source file for the EC region.
IUNIT	Overrides the binary source file for the iUnit region.
IOM	Overrides the binary source file for the IOM region.
-ROMBYPASS <true false>	Overrides rombypass settings in the XML file.
-tbt <file>	Overrides the binary source file for the TBT region.
-SKU <value>	This option is used to change the SKU configuration being built. Use the words Q77, QM77, etc. as a reference to a SKU from the drop-down menu.
-pmcp <file>	overrides the binary source file for the PMCP region
-sd_token <file>	overrides the binary source file for the secure debug token
-oem_km <file>	overrides the binary source file for the OEM KM. override from CLI enabled only in FW Update build.



Option	Description
-nphy <file>	Overrides the binary source file for the Dekel PHY region
-I <enable disable>	Enables or disables intermediate file generation.
-W <path>	Overrides the working directory environment variable \$WorkingDir. It is recommended that the user set these environmental variables first. (Suggested values can be found in the OEM Bringup Guide.)
-S <path>	Overrides the source file directory environment variable \$SourceDir. It is recommended that the user set these environmental variables before starting a project.
-D <path>	Overrides the destination directory environment variable \$DestDir. It is recommended that the user set these environmental variables before starting a project.
-U1 <value>	Overrides the \$UserVar1 environment variable with the value specified. Can be any value required.
-U2 <value>	Overrides the \$UserVar2 environment variable with the value specified. Can be any value required.
-U3 <value>	Overrides the \$UserVar3 environment variable with the value specified. Can be any value required.
-FLASHCOUNT <0-2>	Overrides the number of flash components in the Descriptor Region. If this value is zero, only the Intel® CSE Region is built.
-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.



Option	Description
-set <expression:filename >	Command line overridable key value parameter
-cloHelp	Prints all available Command Line Overridable global paths and with path usage hint.

3.4.30 Example – Decomposing Image and Extracting Parameters

The NVARS variables and the current value parameters of an image can be viewed by dragging and dropping the image into the main window, which then displays the current values of the image's parameters.

An image's parameters can also be extracted by entering the following commands into the command line:

```
FIT.exe /f output.bin /b
```

This command would create a folder named "output". The folder contains the individual region binaries (Descriptor, Intel®ME, and BIOS) and the Map file.

The xml file contains the current Intel® CSE parameters.

The Map file contains the start, end, and length of each region.

3.4.31 More Examples of FIT CLI

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

Take an existing (dt_ori.bin) image and put in a new BIOS binary:
 FIT.exe /b /bios "..\..\..\Image Components\BIOS\BIOS.ROM" <file.bin or
 file.xml>

Take an existing image and put in a different Intel® CSE region:
 FIT.exe /b /me "..\..\..\Image Components\Firmware\CSE15.40_PreProduction.BIN"
 <file.bin or file.xml>

Note: The CSE override option changes the CSE base used on command line but still uses the values from the xml or binary passed in.

Take an existing image and put in a different GbE region:
 FIT.exe /b /gbe "..\..\..\Image



4 Flash Programming Tool

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

Note:

FPT can not be used to program UFS or EMMC flash devices. Please use the Platform Flash Tool to program UFS or EMMC devices. FPT can be used to program Named Variables on all flash devices.

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

- View the contents of the flash on the screen.
- Write the contents of the flash to a log file.
- Perform a binary file to flash comparison.
- Write to a specific address block.
- Program Named variables (SPI and UFS).
- Provision HDCP
- Provided FPF's Access
- Helps doing Closemfnf

4.1 System Requirements

The EFI version of FPT (**fpt.efi**) runs on a 64-bit EFI environment.

The Windows* version (**fptw.exe**) requires administrator privileges to run under Windows* OS. The user needs to use the **Run as Administrator** option to open the CLI in Windows* 10.

The Windows* 64 bit version (fptw64.exe) is designed for running in native 64 bit OS environment which does not have 32 bit compatible mode available for example Windows*PE 64.

Starting Kernel 5.4, driver /dev/mem has been disabled by default to harden the linux environment security, hence blocking FPT execution. Kernel should be recompiled with the /dev/mem set to enabled.

FPT requires that the platform is bootable (i.e. working BIOS) and has an operating system available 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. FPT must be run on the system with the flash memory to be programmed.

One possible workflow for using FPT is:

1. A pre-programmed flash with a bootable BIOS image is plugged into a new computer.
2. The computer boots.



3. FPT is run and a new BIOS/Intel® CSE image is written to flash.
4. The computer powers down.
5. The computer powers up, boots, and is able to access its Intel® CSE capabilities as well as any new custom BIOS features.

4.2 Flash Image Details

See the flash image details as described in the FIT [Chapter 3](#).

4.3 Microsoft Windows* Required Files

The Microsoft Windows* version of the FPT executable is **fptw.exe** (**FPTW64.exe** for 64-bit based Windows* OS). The following files must be in the same directory as **fptw.exe**:

- fptw.exe – the executable used to program the final image file into the flash.
- idrvdll.dll

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

Table 4-1. FPT OS Requirements

FPT Version	Target OS	Support Drivers
FPTw.EXE	Windows* 32 / 64 bit w/WOW64	idrvdll.dll
FPTW64.EXE	Windows* Native 64 bit	idrvdll32e.dll

Note: In the Windows* environment for operations involving global reset you should add a pause or delay when running FPTW using a batch or script file.

4.4 EFI Required Files

The EFI version of the FPT executable is **fpt.efi**. The following files must be placed in **the root directory** as **fpt.efi**:

- fpt.efi – the executable used to program the final image file into the flash. Before running fpt.efi, all the required files must be placed at root directory of the disk otherwise error like "FPT is unable to find FPARTS.TXT "might be displayed.



4.5 Programming Flash Device

Once the Intel® CSE is programmed, it runs at all times. Intel® CSE 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.

Note: Programming the flash or running any of the FPT commands has a limitation on the path length to 128 Characters.

4.5.1 Stopping Intel® CSE SPI Operations

FPT will automatically halt Intel® CSE SPI access prior to erasing or writing data in the CSE region. Customers do not have use either of the following steps listed below when updating platforms unless the descriptor has been locked.

Intel® CSE SPI Operations can be stopped in the following ways:

- Assert HDA_SDO (known as GPIO 33 or Flash descriptor override/Intel® CSE manufacturing jumper) to high while powering on the system. This is not a valid method if the parameters are configured to ignore this jumper.
- Send the HMRFPO ENABLE Intel®MEI command to Intel® CSE (for more information refer PCH Intel® CSE BIOS writer's guide).

Note: Pulling out DIMM from slot 0 or leaving the Intel® CSE region empty to stop Intel® CSE are not valid options for current generation platforms.

4.6 Programming NVARS

FPT can program the NVARS and change the default values of the parameters. The modified parameters are used by the Intel® CSE FW after a global reset (Intel® CSE + HOST reset) or upon returning from a G3 state. NVARS can be programmed using getFileEx/ setFileEx/ CommitFiles APIs.

SetFileEx API will allow for the host to change the values in UEP (Unified Emulation Partition). Note: Intel® CSE Firmware does NOT require commit File after a UEP SetFileEx. Attempting to execute Commit file when not necessary will result in firmware returning an error.

The variables can be modified individually or all at once via a text file.

Note: Files output when using the Intel® FPT -CFGGEN command line option in EFI environments do not contain the Carriage Return code 0x0D ('\r') as part of the EOL (end-of-line) sequence. As a result, when opened in Windows* environments, some applications may show all lines of text on a single line. If the output configuration files are intended to be edited in Windows* environments, it is recommended to use the Windows* version of Intel® FPT accordingly to collect the configuration data. Otherwise, they may be opened using a text editor which can process files which contain only Line Feed 0x0A ('\n') EOL sequences.

Table 4-2. Named Variables Options



Option	Description
fpt.exe -CVARS	Displays a list of the supported manufacturing configurable named variables (NVARs) [Intel(R) AMT HW Status].
fpt.exe -cfggen	Creates a list of blank NVARs in a text file that lets the user update multiple line configurable NVARs. The variables have the following format in the text file: NVAR name = value which will be used by setfileEx.
fpt.exe -U -N <NVAR name> -V <NVAR Value>	Accept for updating UEP values using SetFile API
fpt.exe -U -IN <Text file>	Accepts cfggen file with values set and will use setfile to update

Refer to [Appendix A](#) for a description of all the NVAR parameters.

4.7 Usage

The EFI and Windows* versions of the FPT can run with command line options.

To view all of the supported commands: Run the application with the '-?' Option for windows* OS, and the '-h' option for EFI.

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

```
FPTW64.exe [-H|?] [-VER] [-EXP] [-VERBOSE] [-NORESET] [-Y] [-I]
           [-F] [-VERIFY] [-NOVERIFY] [-D] [-DESC] [-BIOS]
           [-ME] [-GBE] [-PDR] [-EC] [-SAVEMAC] [-SAVESXID] [-E]
           [-REWRITE] [-ADDRESS|A] [-LENGTH|L] [-CVARS] [-MASTERACCESSGEN]
           [-CFGGEN] [-U] [-CLEAR] [-O] [-IN] [-N] [-V] [-CLOSEMNF] [-
GRESET]
           [-PAGE] [-R] [-VARS] [-COMMIT] [-DISABLEME] [-FPFS] [-PROVHDCP]
           [-READHDCP] [-GETPID] [-WRITETOKEN] [-ERASETOKEN] [-PROVKB]
           [-COMMITARBSVN]
```

Table 4-3. Command Line Options for fpt.efi, fpt.exe and fptw.exe

Option	Description
Help (-H, -?)	Displays the list of command line options supported by FPT tool. Note: Use -H for help when running in the EFI Shell.
-CLEAR	Overwrites a pending NVAR value update request with the file system's current.
-VARS	retrieves all the readable vars.
-VER	Shows the version of the tools.



Option	Description
-EXP	Shows examples of how to use the tools.
-VERBOSE [<file>]	Displays the tool's debug information or stores it in a log file.
-Y	Bypasses Prompt. FPT does not prompt user for input. This confirmation will automatically be answered with "y".
-I	Info. Displays information about the image currently used in the flash.
-F <file> [NOVERIFY]	Flash. Programs a binary file into an SPI flash. The user needs to specify the binary file to be flashed. FPT reads the binary, and then programs the binary into the flash. After a successful flash, FPT verifies that the SPI flash matches the provided image. Without specify the length with -L option, FPT will use the total SPI size instead of an image size. The NOVERIFY sub-option <i>*must*</i> follow the file name. This will allow flashing the SPI without verifying the programming was done correctly. The user will be prompted before proceeding unless '-y' is used.
-VERIFY <file>	Verify. Compares a binary to the SPI flash. The image file name has to be passed as a command line argument if this flag is specified.
-NOVERIFY	Suboption of -F, see -F for details.
-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 4KB sections. The total size of the flash device must also be in increments of 4KB.
-DESC	Read/Write 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 BIOS region. Specifies that the BIOS region is to be read, written, or verified. Start address is the beginning of the region.
-ME	Read/Write Intel® CSE region. Specifies that the Intel® CSE region is to be read, written, or verified. The start address is the beginning of the region.
-PDR	Read/Write PDR region. Specifies that the PDR region is to be read, written, or verified. The start address is the beginning of the region.
-E	Skip Erase. Does not erase blocks before writing. This option skips the erase operation before writing and should be used if the part being flashed is a blank SPI flash device.



Option	Description
-A <value>, -ADDRESS <value>	Write/Read Address. Specifies the start address at which a read, verify, or write operation must be performed. The user needs to provide an address. This option is not used when providing a region since the region dictates the start address.
-L <value>, -LENGTH <value>	Write/Read Length. Specifies the length of data to be read, written, or verified. The user needs to provide the length. This option is not used when providing a region since the region/file length determines this.
-CVARS	Lists all the current manufacturing line configurable variables.
-MASTERACCESSGEN	Generates a Manufacturing Line Configurable Master Access Input File.
-CFGGEN	NVAR Input file generation option. This creates a file which can be used to update the line configurable NVARS. The Config file can handle up to 36 NVARS.
-U	Update. Updates variables (NVARS and FPFs) in the UEP. The user can update by specifying their names and values in the parameter file. The parameter file must be in an INI file format (the same format generated by the <code>-cfggen</code> command). The <code>-in <file></code> option is used to specify the input file.
-CLEAR	Using the <code>-CLEAR</code> flag will clear the variable in the UEP. This flag is only supported for a single variable.
-O <file>	Output File. The file used by FPT to output NVAR information.
-IN <file>	Input File. This option flag must be followed by a text file The text file may be either: <div style="margin-left: 40px;">A parameter file such as the one generated with the <code>-cfggen</code> option (used with the <code>-u</code> option)</div> <div style="margin-left: 40px;">or:</div> <div style="margin-left: 40px;">A Configurable Master Access file such as the one generated with the <code>-masteraccessgen</code> option (used with the <code>-closemanuf</code> option)</div> <div style="margin-left: 40px;">The provided config file can handle up to 36 NVARS</div>
-N <value>	Name. Specifies the name of the NVAR that the user wants to update in the image file or flash. The name flag must be used with Value (<code>-v</code>).
-V <value>	Value. Specifies the value for the NVAR variable. The name of variable is specified in the Name flag. The Value flag must follow the Name flag.



Option	Description
-CLOSEMNF [NO]	<p>End of Manufacturing. This option is executed at the end of manufacturing phase. This option does the following:</p> <p>Sets the Intel® CSE manufacturing mode done bit (Global Locked bit).</p> <p>Verifies that the Intel® CSE manufacturing mode done bit (Global Locked) is set.</p> <p>Sets the master region access permission in the Descriptor region to its Intel-recommended value (see the -MASTERACCESSGEN and -IN options)</p> <p>Verifies that flash regions are locked.</p> <p>If the image was properly set before running this option, FPT skips all of the above and reports PASS. If anything was changed, FPT automatically forces a global reset through the CF9GR mechanism. The user can use the no reset option to bypass the reset. If nothing was changed, based on the current setting, the tool reports PASS without any reset.</p> <p>The "NO" addition will prevent the system from doing a global reset following a successful update of the CSE Manufacturing Mode Done, the Region Access permissions, or both.</p> <p>Note: Running <code>FPT-closemnf</code> also sets the default value for any unprovisioning process. Run <code>FPT -closemnf</code> first if the user wants to test any unprovisioning related process. In order to allow FPT to perform a global reset, BIOS should not lock CF9GR when Intel® CSE is in manufacturing mode. This step is highly recommended to the manufacturing process. Without doing proper end of manufacturing process would lead to ship platform with potential security/privacy risk.</p> <p>Important:</p> <p>Before using this option with Production MCP / FW verify that the values for the PTT and Anchor Cove are correct in your image. Once this setting is used it will permanently commit values into the Field Programmable Fuses and cannot be undone.</p>
-GRESET	Global Reset. FPT performs a global reset.
-PAGE	Pauses the screen when a page of text has been reached. Hit any key to continue.
-R <name>	NVAR and PPFs Read. FPT uses this option to retrieve NVAR value for a specific NVAR file name. The value of the variable is displayed. By default, all non- secure variables are displayed in clear-text and secure NVAR will be displayed in HASH. The -hashed option can be used to display the hash of a value instead of the clear-text value.



Option	Description
-VARS	Display Supported Variables. FPT uses this option to display all variables supported for the -R and -COMPARE commands. Note: This will no longer display UEP based values which are tied to configuring iFPF's.
-COMMIT	Commit. FPT uses this option to commit all setfile commands NVARs changes to NVAR and cause relevant reset accordingly. If no pending variable changes are present, Intel® CSE does not reset and the tool displays the status of the commit operation.
-DISABLEME	Disable the Management Engine.
-COMPAREFPF<name>	Compare the FPF with a value passed in by the user.
-FPFS	Displays a list of the FPFs.
-COMMITFPF <name>	Commits NVAR values to FPF via firmware and prevents further modification of FPFs.
-PROVHDCP <file> <file>	Provision platform with the key and cert provided.
-READHDCP	Displays the HDCP Rx provisioning status.
-GETPID <file>	Retrieve the part id.
-REWRITE	Allows to rewrite the SPI with file data even if flash is identical.
-WRITETOKEN <file>	Write the token where the file name is the token name.
-ERASETOKEN	Delete the token.
-PROVKB <iv_and_keybox.bin>	Provision Widevine using IV (Initialization Vector) and encrypted KeyBox file.

Table 4-4. FPT-closemfnf Behavior

Condition before FPT -closemfnf			Condition after FPT -closemfnf			Other FPT Action	
Intel CSE Mfg Done bit set	Flash Access set to Intel rec values	Intel CSE Mfg Mode	Intel CSE Mfg Done bit set	Flash Access set to Intel rec values?	Intel CSE Mfg Mode	FPT return value **	Global Reset
No	No	Enabled	Yes	Yes	Disabled	0	Yes
No	Yes	Enabled	No	Yes	Enabled	1	No
Yes	No	Enabled	Yes	Yes	Disabled	0	Yes
Yes	Yes	Disabled	Yes	Yes	Disabled	0	No



** Return value 0 indicates successful completion. In the second case, FPT -closemnf returns 1 (= error) because it is unable to set the Intel CSE Mfg Done bit, because flash permissions are already set to Intel recommended values (host cannot access Intel CSE Region).

Table 4-5. Intel-Recommend Access Settings

	CSE	GBE	BIOS	EC
Read	0b 0000 0000 1101 = 0x00d	0b 0000 0000 1000 = 0x009	0b 0000 000† 000‡ 1011 = 0x0†‡F	0b 0000 0001 0000 00*1 = 0x0101 or 0x0103
Write	0b 0000 0000 1100 = 0x004	0b 0000 0000 1000 = 0x008	0b 000† 000‡ 1010 = 0x†‡A	0b 0000 0001 0000 0x100
Note: <ol style="list-style-type: none"> ‡ = Value dependent on if PDR is implemented and if Host access is desired. † = Optional BIOS access to the EC region. * = Optional EC Read access to BIOS. 				

Notes:

1. Case **A** depends on platform design if optional BIOS access to PDR, add PDR parameter after -closemnf; BIOS Read = 0x1F, BIOS Write = 0x1A.
2. Case **B** depends on platform design if optional BIOS access to the EC region, add EC parameter after -closemnf; BIOS Read = 0x10F, BIOS Write = 0x10A.
3. Case **C** depends on platform design if optional enable EC read access to BIOS, add BIOS parameter after -closemnf; EC Read = 0x103

4.8 Examples

The following examples illustrate the usage of the EFI versions of the tool (fpt.efi and fpt.exe respectively). The Windows* version of the tool (Fptw.exe) behaves in the same manner apart from running in a Windows* environment.

4.8.1 Complete SPI Flash Device with Binary File

In order to use FPT Tool for Flashing the SPI Image the following BIOS settings need to be done manually otherwise errors may be seen related to BIOS Region Protected while executing fpt.exe -f spi.bin.

1. BIOS MENU INTEL ADVANCED Menu → CPU CONFIGURATION → BIOS GUARD: Disabled
2. BIOS MENU → INTEL ADVANCED Menu → PCH I/O CONFIGURATION → SECURITY CONFIGURATION → BIOS LOCK: Disabled
3. BIOS MENU -> INTEL ADVANCED Menu -> PCH I/O CONFIGURATION -> Flash Protection Range: Disabled.

```
C:\> fpt.exe -f spi.bin
```

EFI:

```
>fpt.efi -f spi.bin or fs0:\>fpt.efi -f spi.bin
```



This command writes the data in the **spi.bin** file into a whole SPI flash from address 0x0.

4.8.2 Program Specific Region

```
fpt.exe -f bios.rom -BIOS
```

```
EFI:
```

```
fpt.efi -f bios.rom -BIOS
```

```
Intel © Flash Programming Tool Version: xx.x.x.xxxx
```

```
Copyright (C) 2005 - 2019, Intel Corporation. All rights reserved.
```

```
Reading HSFSTS register... Flash Descriptor: Valid
```

```
Processing Flash memory block 950 from 2559.
```

```
- Erasing Flash Block [0x9B7000] - 100 percent complete.
```

```
- Programming Flash [0x09B7000] 2332KB of 2332KB - 100 percent complete.
```

```
Processing Flash memory block 1550 from 2559.
```

```
- Erasing Flash Block [0xC0F000] - 100 percent complete.
```

```
- Programming Flash [0x0C0F000] 1916KB of 1916KB - 100 percent complete.
```

```
Processing Flash memory block 1591 from 2559.
```

```
- Erasing Flash Block [0xC38000] - 100 percent complete.
```

```
- Programming Flash [0x0C38000] 160KB of 160KB - 100 percent complete.
```

```
Processing Flash memory block 1748 from 2559.
```

```
- Erasing Flash Block [0xCD5000] - 100 percent complete.
```

```
- Programming Flash [0x0CD5000] 532KB of 532KB - 100 percent complete.
```

```
Processing Flash memory block 1805 from 2559.
```

```
- Erasing Flash Block [0xD0E000] - 100 percent complete.
```

```
- Programming Flash [0x0D0E000] 188KB of 188KB - 100 percent complete.
```

```
Processing Flash memory block 1816 from 2559.
```

```
- Erasing Flash Block [0xD19000] - 100 percent complete.
```

```
- Programming Flash [0x0D19000] 36KB of 36KB - 100 percent complete.
```

```
Processing Flash memory block 1908 from 2559.
```

```
- Erasing Flash Block [0xD75000] - 100 percent complete.
```

```
- Programming Flash [0x0D75000] 344KB of 344KB - 100 percent complete.
```

```
Processing Flash memory block 2042 from 2559.
```

```
- Erasing Flash Block [0xDFB000] - 100 percent complete.
```

```
- Programming Flash [0x0DFB000] 364KB of 364KB - 100 percent complete.
```

```
Processing Flash memory block 2324 from 2559.
```

```
- Erasing Flash Block [0xF15000] - 100 percent complete.
```

```
- Programming Flash [0x0F15000] 596KB of 596KB - 100 percent complete.
```

```
Processing Flash memory block 2540 from 2559.
```

```
- Erasing Flash Block [0xFED000] - 100 percent complete.
```

```
- Programming Flash [0x0FED000] 52KB of 52KB - 100 percent complete.
```

```
Processing Flash memory block 2559 from 2559.
```

```
- Erasing Flash Block [0x1000000] - 100 percent complete.
```

```
- Programming Flash [0x1000000] 20KB of 20KB - 100 percent complete.
```

```
RESULT: The data is identical.10240KB of 10240KB - 100 percent complete.
```



FPT Operation Successful.

This command writes the data in **bios.bin** into the BIOS region of the SPI flash and verifies that the operation ran successfully.

4.8.3 Program SPI Flash from Specific Address

```
fpt.exe -F image.bin -A 0x100 -L 0x800
```

EFI:

```
fpt.efi -F image.bin -A 0x100 -L 0x800
```

This command loads 0x800 of the binary file **image.bin** starting at address 0x0100. The starting address and the length needs to be a multiple of 4KB.

4.8.4 Dump Full Image

```
fpt.exe -d imagedump.bin
```

EFI:

```
fpt.efi -d imagedump.bin
```

```
-----  
Intel ® Flash Programming Tool. Version:  x.x.x.xxxx  
Copyright (c) 2005-2019, Intel Corporation. All rights reserved.
```

```
Platform: Intel® Qxx Express Chipset  
Reading HSFSTS register... Flash Descriptor: Valid
```

```
- Reading Flash [0x1000000] 16384KB of 16384KB - 100% complete.  
Writing flash contents to file "imagedump.bin"...  
Memory Dump Complete
```

```
Warning: There are some addresses that are not defined in any regions.  
Read/Write/Erase operations are not possible on those addresses.
```

```
FPT Operation Successful
```

4.8.5 Dump Specific Region

```
fpt.exe -d descdump.bin -desc
```

EFI:

```
fpt.efi -d descdump.bin -desc
```

```
-----  
Intel ® Flash Programming Tool. Version:  x.x.x.xxxx
```




```
Copyright (c) 2005-2019, Intel Corporation. All rights reserved.

Reading HSFSTS register... Flash Descriptor: Valid

- Reading Flash [0x0001000]... 4KB of 4KB - 100% complete.

Writing flash contents to file "descdump.bin"...

Memory Dump Complete

Warning: There are some addresses that are not defined in any regions.

Read/Write/Erase operations are not possible on those addresses.

FPT Operation Successful
```

This command writes the contents of the Descriptor region to the file **descdump.bin**.

4.8.6 Display SPI Information

```
fptw.exe -I
-----
Intel ® Flash Programming Tool Version: 15.40.x.xxxx
Copyright (C) 2005 - 2019, Intel Corporation. All rights reserved.
```

```
Reading HSFSTS register... Flash Descriptor: Valid

--- Flash Devices Found ---
ID:0xEF4019    Size: 32768KB (262144Kb)
ID:0xEF4017    Size: 8192KB (65536Kb)

--- Flash Image Information --
Signature: VALID
Number of Flash Components: 1
  Component 1 - 32768KB (262144Kb)
Regions:
  DESC      - Base: 0x00000000, Limit: 0x00000FFF
  BIOS      - Base: 0x01300000, Limit: 0x01FFFFFF
  CSME      - Base: 0x00001000, Limit: 0x01000FFF
  GbE       - NOT PRESENT
  PDR       - NOT PRESENT
  EC        - NOT PRESENT
Master Region Access:
  BIOS      - ID: Read: 0xFFFF, Write: 0xFFFF
  CSME      - ID: Read: 0xFFFF, Write: 0xFFFF
  GbE       - ID: NOT PRESENT
  EC        - ID: NOT PRESENT
```

Total Accessible SPI Memory: 32768KB, Total Installed SPI Memory: 40960KB

Warning: There are some addresses that are not defined in any regions. Read/Write/ Erase operations are not possible on those addresses.

FPT Operation Successful.



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**, FPT returns the error message "There is no supported SPI flash device installed".

4.8.7 Verify Image with Errors

```
fpt.exe -verify outimage.bin
```

EFI:

```
fpt.efi -verify outimage.bin
```

Intel® Flash Programming Tool. Version: x.x.x.xxxx

Copyright (c) 2005-2019, Intel Corporation. All rights reserved.

Reading HSFSTS register... Flash Descriptor: Valid

-Verifying Flash [0x0000000] 4KB of 16384KB - 0 percent complete

Error 207: Data verify mismatch found.

Warning: There are some addresses that are not defined in any regions. Read/Write/Erase operations are not possible on those addresses.

This command compares the Intel® CSE region programmed on the flash with the specified FW image file **outimage.bin**. If the **-y** option is not used; the user is notified that the file is smaller than the binary image. This is due to extra padding that is added during the program process. The padding can be ignored when performing a comparison. The **-y** option proceeds with the comparison without warning.

4.8.8 Verify Image Successfully

```
fpt.exe -verify outimage.bin
```

EFI:

```
fpt.efi -verify outimage.bin
```

Intel ® Flash Programming Tool. Version: x.x.x.xxxx

Copyright (c) 2005-2019, Intel Corporation. All rights reserved.

Platform: Intel® Qxx Express Chipset

Reading HSFSTS register... Flash Descriptor: Valid

-Verifying Flash [0x800000] 32768KB of 32768KB - 100% complete.



RESULT: The data is identical.

FPT Operation Successful

This command compares **image.bin** with the contents of the flash. Comparing an image should be done immediately after programming the flash device. Verifying the contents of the flash device after a system reset results in a mismatch because Intel® CSE changes some data in the flash after a reset.

4.8.9 CVAR Configuration File Generation (-cfggen)

It creates an input file which can be used to update CVARs and FPFs. The file includes all the current CVAR or FPFs. When creating the file, it extracts the fixed offset variables from flash. Note, the file generated will change every time the list of CVAR changes.

```
fpt.exe -cfggen [ -o <Output Text File> ][ options ]
```

-o <Output File Name>	The desired name of the file generated. If none is provided the default, fpt.cfg, will be used.
-p < file name >	Alternate SPI Flash Parts list file.
-page	Pauses at screen / page / window boundaries. Hit any key to continue.
-Verbose [<file name>]	Displays more information.
-y	Will not pause to user input to continue

Example FPT.CFG output:

```
Intel (R) Flash Programming Tool Version: 15.40.0.1029
Copyright (C) 2005 - 2020, Intel Corporation. All rights reserved.
```

```
LPC Device Id: 4B00.
Platform: ElkhartLake Platform
General FW Information
FW Status Register1 0x90000255
FW Status Register2 0x62000506
FW Status Register3 0x00000020
FW Status Register4 0x00004000
FW Status Register5 0x02001F03
FW Status Register6 0x044003C9
```

```
Current FW State Normal
Flash Partition Table Valid
FW Memory State CM0 with UMA
FW Initialization Complete
```



```
BUP Loading state Success
FW Error Code No Error
FW Mode Of Operation Normal
SPI Flash Log Not Present
FW Loading Phase HOSTCOMM Module
FW Loading Phase Status UNKNOWN
ME File System Corrupted No
RPMC status OK
Creating FOV base input file [ FPT.cfg ] ...
FPT Operation Successful.
```

§ §



5 Intel®ME Manuf and MEManufWin

Intel®ME Manuf validates Intel® CSE functionality on the manufacturing line. It does not check for LAN functionality as it assumes that all Intel® CSE components on the test board have been validated by their respective vendors. It does verify that these components have been assembled together correctly.

The Windows* version of Intel®MEManufWin (Intel®MEManufWin) requires administrator privileges to run under Windows* OS. The user needs to use the **Run as Administrator** option to open the CLI in Windows* 10.

Intel®ME Manuf validates all components and flows that need to be tested according to the FW installed on the platform in order to ensure the functionality of Intel® CSE applications: BIOS-FW, Flash, SMBus, M-Link, KVM, etc. This tool is meant to be run on the manufacturing line.

5.1 Windows* PE Requirements

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

5.2 How to Use Intel®ME Manuf

Intel®ME Manuf checks the FW SKU and runs the proper tests accordingly unless an option to select tests is specified.

Intel®ME Manuf is intelligent enough to know if it should run the test or report a result. If there is no test result available for an Intel® CSE enabled platform, ME Manuf calls the test. Otherwise, it reports the result or the failure message from the previous test.

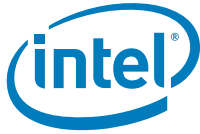
Intel®ME Manuf tools report the result or cause a reboot. If there is a reboot, Intel®ME Manuf should be run again.

5.3 Usage

The Windows* version of the tool can be executed by:

```
MEManufWin64.exe [-EXP] [-H|?] [-VER] [-BLOCKNET] [-ALLOWNET]
                 [-TEST] [-S0] [-BISTRESULT] [-NEXTREBOOT] [-EOL]
                 [-CFGGEN] [-F] [-VERBOSE] [-PAGE] [-ALL] [-LEVEL]
```

Following is the output generated when the tool is run with the help option (**-h** or **?**):



Intel (R) MEmanuf Version: 15.40.0.1039

Copyright (C) 2005 - 2020, Intel Corporation. All rights reserved.

```
MEmanufWin64.exe [-EXP] [-H|?] [-VER] [-BLOCKNET] [-ALLOWNET]
                  [-TEST] [-S0] [-BISTRESULT] [-NEXTREBOOT] [-EOL]
                  [-CFGGEN] [-F] [-VERBOSE] [-PAGE] [-ALL] [-LEVEL]
```

-EXP [arg_name]	Display example usage of this tool
-H ?	Display help screen
-VER	Display version information
-BLOCKNET	Block network traffic from GbE and Wireless LAN
-ALLOWNET	Allow network traffic from GbE and Wireless LAN
-TEST	Rerun BIST test
-S0	Rerun BIST test that does not require power cycle
-BISTRESULT	Return last BIST results
-NEXTREBOOT	Remove Live HW BIST data results
-EOL [var config]	Do end of line check
-CFGGEN <filename>	Generate test configuration file
-F <filename>	Load config file
-VERBOSE [filename]	Display the debug information of the tool
-PAGE	Pause after each screenful of information
-ALL	Generate all possible tests for config file
-LEVEL [level between 1 and 3]	Selective eol tests level

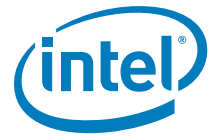


Table 5-1. Options for ME Manuf

Option	Description
No option	<p>There are differences depending on the firmware SKU type the system is running on:</p> <p>If BIST is disabled in the Intel® CSE Boot: The first time running Intel®ME Manuf, since there is no CM3 test result stored in SPI, the tool will request the FW to run a complete BIST which includes a Hibernation for the Windows* version. This power reset is only host side power cycle that triggered by Intel®ME. When host resets, Intel® CSE FW will transition from CM0 to CM3, and then attempt automatically transition back from CM3 to CM0 along bringing host back to S0. Once host is booted back into OS, user needs to run the tool again in order to run runtime BIST and retrieve the test result.</p> <p>If BIST is enabled in the Intel® CSE Boot: If there is no CM3 test result, the tool will report error and request user to use -test to run a full BIST. If there is CM3 test result, the tool will execute the runtime BIST and report the result.</p> <p>If BIST test result is not displayed after BIST test is done, the tool needs to be run again (with or without any BIST related argument combinations) to retrieve the result, once test result is displayed, it will be cleared.</p> <p>Tool is capable of remembering whether/what tests (including host based tests) have been run from previous invocation. Host based tests will be run for all cases (whether it's retrieving test result or run the actual BIST). Currently there are two host based tests; they are VSCC Table validation check and ICC data check.</p>
-EXP	Shows examples of how to use the tools.
-H or -?	Displays the help screen.
	Note: Use -H for help when running in the EFI Shell.
-VER	Shows the version of the tools.
-F <filename>	Load customer defined .cfg file
-TEST	Run full test
-BISTRESULT	Returns last BIST results.



Option	Description
-EOL <Var Config> - F <filename>	<p>This option runs several checks for the use of OEMs to ensure that all settings and configurations have been made according to Intel requirements before the system leaves the manufacturing process. The check can be configured by the customer to select which test items to run and their expected value (only applicable for Variable Values, FW Version and BIOS Version). The sub option <code>config</code> or <code>var</code> is optional. Using <code>-EOL</code> without a sub option is equivalent to the <code>-EOL config</code>. ICC data check is performed for all options.</p> <p>The Full BIST test for is a combination of M0_HW, Live_HW and M0_Config. The Runtime BIST is a combination of M0_HW and M0_Config.</p> <p>Intel®ME Manuf Sx test will require system is capable to enter sleep state, keep pinging the platform with network package and keep the system up will make the test failed.</p> <p>Host based Tests</p> <p>ME/BIOS VSCC validation, Intel®ME Manuf verifies that flash SPI ID on the system is described in VSCC table. If found, VSCC entry for relevant SPI part should match the known good values that pre-populated in the file.</p> <p>Intel® CSE state check, Intel®ME Manuf verifies Intel® CSE is in normal state. This is done by checking the value of 4 fields (initialization state, mode of operation, current operation state, and error state) in FW status register1. If any of these fields indicates Intel® CSE is in abnormal state, Intel®ME Manuf will report error without running BIST test.</p> <p>When <code>-f</code> flag is used along with a file name (<filename>), the tool will load the file as the configuration file, instead of using MEmanuf.xml.</p>
-NEXTREBOOT	<p>Upon successful platform reboot CM3 Autotest will be performed.</p> <p>Note: This is a standalone command and will only work if CM3 Autotest has been enabled in the firmware image. CM3 Autotest will be executed on the next CMoff – CM0 transition (example: Cold Reset), Global Reset or G3. The option itself will not trigger any platform reboots.</p>
-CFGGEN <filename>	<p>Use this option along with a filename to generate a default configuration file. This file (with or without modification) can be used for the <code>-EOL</code> option. Rename it MEmanuf.xml before using it. It is highly recommended to use this option to generate a new MEmanuf.xml with an up-to-date variable names list before using the Intel®MEmanuf End-Of-Line check feature.</p>
-VERBOSE <file>	Displays the debug information of the tool or stores it in a log file.
-ALLOWNET	Allow network traffic from GbE and Wireless LAN
-BLOCKNET	Block network traffic from GbE and Wireless LAN
-LEVEL [level between 1 and 3]	Selective eol tests level
-GFX	This option will force KVM related test.
-DRVLESS	Driverless mode

**Table 5-2. Intel®ME Manuf Test Matrix**

		CM3 Supported SKU	Consumer SKU
BIST Disabled in the CSE BOOT	No option	-1st time: Run full BIST test (with CSE triggered reset under DOS, host triggered hibernation under Windows*), and save the CM3 test result in SPI - After: Run Runtime BIST and query CM3 test result from SPI without reset.	Run runtime BIST test
	-Test	-Run full BIST test with Intel CSE triggered reset in DOS and host triggered hibernation in Windows* - Save the CM3 test result in SPI.	Run runtime BIST test (with no reset)
	-S0	Run runtime BIST test (with no reset).	Same as CM3 Supported SKU
BIST Enabled in the CSE BOOT	No option	Run the Runtime BIST and query M3 test result from SPI without reset, if not CM3 test result retrieved, return error.	Run runtime BIST test (with no reset)
	-Test	-Run full BIST test with Intel CSE triggered reset in DOS and host triggered hibernation in Windows* - Save the CM3 test result in SPI .	Run runtime BIST test (with no reset)
	-S0	Run runtime BIST test (with no reset)	Same as CM3 Supported SKU

Note: The Full BIST test for is a combination of M0_HW, Live_HW and M0_Config. The Runtime BIST is a combination of M0_HW and M0_Config.

Intel®ME Manuf Sx test will require system is capable to enter sleep state, keep pinging the platform with network package and keep the system up will make the test failed.

5.3.1 Host based Tests

1. CSE/BIOS VSCC validation, Intel®ME Manuf verifies that flash SPI ID on the system is described in VSCC table. If found, VSCC entry for relevant SPI part should match the known good values that pre-populated in the file.
2. Intel® CSE state check, Intel®ME Manuf verifies Intel® CSE is in normal state. This is done by checking the value of 4 fields (initialization state, mode of operation, current operation state, and error state) in FW status register1. If



any of these fields indicates Intel® CSE is in abnormal state, Intel®ME Manuf will report error without running BIST test.

3. ICC data check, Intel®ME Manuf verifies that valid OEM ICC data is present and programmed accordingly. This is done by checking FW status register2 ICC bits (which are bit 1 and 2 equal to 3).

5.4 Intel®MEManuf –EOL Check

MEManuf `-EOL VAR` check is used to give customers the ability to check Intel®ME-related configuration before shipping. There are two sets of tests that can be run: variable check and configuration check. Variable check is very similar as FPT `-compare` option. Refer that section.

5.4.1 ErrorAction Field

The `end_of_line` (-EOL) check is split into two categories; *Variable Check*, and *Configuration Check*. If any of these checks fails, by default Intel®ME Manuf will report the error and continue to the next check.

If it is desired to change this default behavior, 'ErrorAction' field can be used. In other words, ErrorAction is used to define the importance of a test. It can be defined with one of the following values:

- **ErrorContinue:** this is the default value, it reports the error and continue to the next check.
- **ErrorStop:** When an error is encountered, it's reported and the testing process stops.
- **WarnContinue:** reports a warning regarding the error and continues to the next check.

5.4.2 MEManuf.xml File

The **MEManuf.xml** file includes all the test configurations for MEManuf `-EOL` check. It needs to be at the same folder that ME Manuf is run. If there is no **MEManuf.xml** file on that folder, MEManuf `-EOL config` runs the Intel recommended default check only.

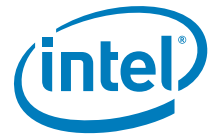
Here is an example of the new xml configuration file:

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

<!-- This is the configuration file for the csmemanuf test tool. -->

<!-- This file is divided into the different test types (csmebist, eolconfig, eolvar). -->

<!-- Any line in this file that is marked with "<!--" to start with is NOT editable by the
user and is strictly informational. Any changes to these lines will be ignored -->
```



```
<!-- Generally the user may change enabled(true/false), errorlevel(error,warning), and
in some cases required value -->
```

```
<!-- It is recommended that you edit this document with an XML specific/capable editor
-->
```

```
<!-- A missing field or bad value will fail validation and result in an error -->
```

```
<!-- State PossibleValues="Enabled/Disabled" -->
```

```
<!-- ErrAction PossibleValues="ErrorContinue/ErrorStop/WarningContinue" -->
```

```
<memanuf_config>
```

```
    <!-- CSME BIST TESTS -->
```

```
    <csmebist name="Policy Kernel - Boot Guard : Self Test">
```

```
        <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->
```

```
        <!-- Description>Get test result from NVAR
SECURE_BOOT_SELF_TEST_RESULT.</Description -->
```

```
        <!-- IntelRequired>True</IntelRequired -->
```

```
        <!-- TestType>M0_HW</TestType -->
```

```
        <!-- End of uneditable fields -->
```

```
        <!-- Please edit the fields below ONLY with the State or ErrAction -->
```

```
        <State>Enabled</State>
```

```
        <ErrAction>ErrorContinue</ErrAction>
```

```
    </csmebist>
```

```
    <csmebist name="Policy Kernel - ME Configuration : PROC_MISSING">
```

```
        <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->
```

```
        <!-- Description>Only on mobile. Test fails if rule is not set to
MEFWCAPS_NO_ONBOARD_GLUE_LOGIC.</Description -->
```

```
        <!-- IntelRequired>True</IntelRequired -->
```

```
        <!-- TestType>M0_CONFIG</TestType -->
```

```
        <!-- End of uneditable fields -->
```



```
<!-- Please edit the fields below ONLY with the State or ErrAction -->

<State>Enabled</State>

<ErrAction>ErrorContinue</ErrAction>

</csmebist>

<csmebist name="VDM - General : VDM engine">

    <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

    <!-- Description>Test VDM.</Description -->

    <!-- IntelRequired>True</IntelRequired -->

    <!-- TestType>M0_HW</TestType -->

    <!-- End of uneditable fields -->

    <!-- Please edit the fields below ONLY with the State or ErrAction -->

    <State>Enabled</State>

    <ErrAction>ErrorContinue</ErrAction>

</csmebist>

<csmebist name="PAVP - General : Verify Edp and Lspcon Configurations">

    <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

    <!-- Description>Check if LSPCON and 5K ports are overlapped</
Description -->

    <!-- IntelRequired>True</IntelRequired -->

    <!-- TestType>M0_HW</TestType -->

    <!-- End of uneditable fields -->

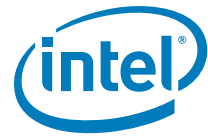
    <!-- Please edit the fields below ONLY with the State or ErrAction -->

    <State>Enabled</State>

    <ErrAction>ErrorContinue</ErrAction>

</csmebist>

<csmebist name="PAVP - General : Set Lspcon Port">
```



<!-- The commented fields below CANNOT be edited. Any edits will be ignored by the tool -->

<!-- Description>Test the validity of the 5K port configuration</Description -->

<!-- IntelRequired>True</IntelRequired -->

<!-- TestType>M0_HW</TestType -->

<!-- End of uneditable fields -->

<!-- Please edit the fields below ONLY with the State or ErrAction -->

<State>Enabled</State>

<ErrAction>ErrorContinue</ErrAction>

</csmebist>

<csmebist name="PAVP - General : Set Edp Port">

<!-- The commented fields below CANNOT be edited. Any edits will be ignored by the tool -->

<!-- Description>Test the validity of the LSPCON port configuration</Description -->

<!-- IntelRequired>True</IntelRequired -->

<!-- TestType>M0_HW</TestType -->

<!-- End of uneditable fields -->

<!-- Please edit the fields below ONLY with the State or ErrAction -->

<State>Enabled</State>

<ErrAction>ErrorContinue</ErrAction>

</csmebist>

<!-- END OF CSME BIST TESTS -->

<!-- EOL CONFIG TESTS -->

<eolconfig name="Secure boot KM SVN">

<!-- The commented fields below CANNOT be edited. Any edits will be ignored by the tool -->

<!-- Description>Check fpf Secure boot KM SVN against expected value</Description -->



```
<!-- IntelRequired>False</IntelRequired -->

<!-- Dependencies></Dependencies -->

<!-- Level>1</Level -->

<!-- End of uneditable fields -->

<!-- Please edit the fields below ONLY with the State or ErrAction -->

<State>Enabled</State>

<ErrAction>ErrorContinue</ErrAction>

<RequiredValue format="Hex number with 0x prefix."
example="0x00"> </RequiredValue>

</eolconfig>

<eolconfig name="Secure boot BSMM SVN">

    <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

    <!-- Description>Check fpf Secure boot BSMM SVN against expected
value</Description -->

    <!-- IntelRequired>False</IntelRequired -->

    <!-- Dependencies></Dependencies -->

    <!-- Level>1</Level -->

    <!-- End of uneditable fields -->

    <!-- Please edit the fields below ONLY with the State or ErrAction -->

    <State>Enabled</State>

    <ErrAction>ErrorContinue</ErrAction>

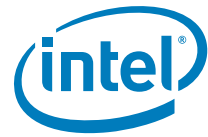
    <RequiredValue format="Hex number with 0x prefix."
example="0x00"> </RequiredValue>

</eolconfig>

<eolconfig name="Secure boot ACM SVN">

    <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

    <!-- Description>Check fpf Secure boot ACM SVN against expected
value</Description -->
```



```

        <!-- IntelRequired>False</IntelRequired -->

        <!-- Dependencies></Dependencies -->

        <!-- Level>1</Level -->

        <!-- End of uneditable fields -->

        <!-- Please edit the fields below ONLY with the State or ErrAction -->

        <State>Enabled</State>

        <ErrAction>ErrorContinue</ErrAction>

        <RequiredValue format="Hex number with 0x prefix."
example="0x00"> </RequiredValue>

    </eolconfig>

    <eolconfig name="ROT KM SVN">

        <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

        <!-- Description>Check fpf ROT KM SVN against expected value</
Description -->

        <!-- IntelRequired>False</IntelRequired -->

        <!-- Dependencies></Dependencies -->

        <!-- Level>1</Level -->

        <!-- End of uneditable fields -->

        <!-- Please edit the fields below ONLY with the State or ErrAction -->

        <State>Enabled</State>

        <ErrAction>ErrorContinue</ErrAction>

        <RequiredValue format="Hex number with 0x prefix."
example="0x00"> </RequiredValue>

    </eolconfig>

    <eolconfig name="PMC SVN">

        <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

        <!-- Description>Check fpf PMC SVN against expected value</
Description -->

```



```
<!-- IntelRequired>False</IntelRequired -->

<!-- Dependencies></Dependencies -->

<!-- Level>1</Level -->

<!-- End of uneditable fields -->

<!-- Please edit the fields below ONLY with the State or ErrAction -->

<State>Enabled</State>

<ErrAction>ErrorContinue</ErrAction>

<RequiredValue format="Hex number with 0x prefix."
example="0x00"> </RequiredValue>

</eolconfig>

<eolconfig name="OEM KM SVN">

    <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

    <!-- Description>Check fpf OEM KM SVN against expected value</
Description -->

    <!-- IntelRequired>False</IntelRequired -->

    <!-- Dependencies></Dependencies -->

    <!-- Level>1</Level -->

    <!-- End of uneditable fields -->

    <!-- Please edit the fields below ONLY with the State or ErrAction -->

    <State>Enabled</State>

    <ErrAction>ErrorContinue</ErrAction>

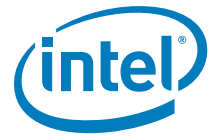
    <RequiredValue format="Hex number with 0x prefix."
example="0x00"> </RequiredValue>

    </eolconfig>

    <eolconfig name="IDLM SVN">

        <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

        <!-- Description>Check fpf IDLM SVN against expected value</
Description -->
```

```

        <!-- IntelRequired>False</IntelRequired -->

        <!-- Dependencies></Dependencies -->

        <!-- Level>1</Level -->

        <!-- End of uneditable fields -->

        <!-- Please edit the fields below ONLY with the State or ErrAction -->

        <State>Enabled</State>

        <ErrAction>ErrorContinue</ErrAction>

        <RequiredValue format="Hex number with 0x prefix."
example="0x00"> </RequiredValue>

    </eolconfig>

    <eolconfig name="DNX SVN">

        <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

        <!-- Description>Check fpf DNX SVN against expected value</
Description -->

        <!-- IntelRequired>False</IntelRequired -->

        <!-- Dependencies></Dependencies -->

        <!-- Level>1</Level -->

        <!-- End of uneditable fields -->

        <!-- Please edit the fields below ONLY with the State or ErrAction -->

        <State>Enabled</State>

        <ErrAction>ErrorContinue</ErrAction>

        <RequiredValue format="Hex number with 0x prefix."
example="0x00"> </RequiredValue>

    </eolconfig>

    <eolconfig name="USB Port ID">

        <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

        <!-- Description>Check fpf USB Port ID against expected value</
Description -->

```



```
<!-- IntelRequired>False</IntelRequired -->

<!-- Dependencies></Dependencies -->

<!-- Level>1</Level -->

<!-- End of uneditable fields -->

<!-- Please edit the fields below ONLY with the State or ErrAction -->

<State>Enabled</State>

<ErrAction>ErrorContinue</ErrAction>

<RequiredValue format="Hex number with 0x prefix."
example="0x00000000"> </RequiredValue>

</eolconfig>

<eolconfig name="UFS Boot Source">

    <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

    <!-- Description>Check fpf UFS Boot Source against expected value</
Description -->

    <!-- IntelRequired>False</IntelRequired -->

    <!-- Dependencies></Dependencies -->

    <!-- Level>1</Level -->

    <!-- End of uneditable fields -->

    <!-- Please edit the fields below ONLY with the State or ErrAction -->

    <State>Enabled</State>

    <ErrAction>ErrorContinue</ErrAction>

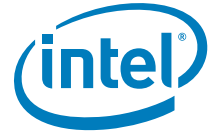
    <RequiredValue format="Enabled/00/Disabled/01"
example="Enabled"> </RequiredValue>

</eolconfig>

<eolconfig name="TXT Supported">

    <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

    <!-- Description>Check fpf TXT Supported against expected value</
Description -->
```



```

        <!-- IntelRequired>False</IntelRequired -->

        <!-- Dependencies></Dependencies -->

        <!-- Level>1</Level -->

        <!-- End of uneditable fields -->

        <!-- Please edit the fields below ONLY with the State or ErrAction -->

        <State>Enabled</State>

        <ErrAction>ErrorContinue</ErrAction>

        <RequiredValue format="Disabled/00/Enabled/01"
example="Disabled"> </RequiredValue>

    </eolconfig>

    <eolconfig name="Secure boot KM Anti Rollback">

        <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

        <!-- Description>Check fpf Secure boot KM Anti Rollback against
expected value</Description -->

        <!-- IntelRequired>False</IntelRequired -->

        <!-- Dependencies></Dependencies -->

        <!-- Level>1</Level -->

        <!-- End of uneditable fields -->

        <!-- Please edit the fields below ONLY with the State or ErrAction -->

        <State>Enabled</State>

        <ErrAction>ErrorContinue</ErrAction>

        <RequiredValue format="Disabled/00/Enabled/01"
example="Disabled"> </RequiredValue>

    </eolconfig>

    <eolconfig name="Safety Island Config">

        <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

        <!-- Description>Check fpf Safety Island Config against expected
value</Description -->

```



```
<!-- IntelRequired>False</IntelRequired -->

<!-- Dependencies></Dependencies -->

<!-- Level>1</Level -->

<!-- End of uneditable fields -->

<!-- Please edit the fields below ONLY with the State or ErrAction -->

<State>Enabled</State>

<ErrAction>ErrorContinue</ErrAction>

<RequiredValue format="config is signed/00/config is not signed/01/
config is not present/02/isi is not present/03" example="config is signed"> </
RequiredValue>

</eolconfig>

<eolconfig name="SPI Boot Source">

    <!-- The commented fields below CANNOT be edited. Any edits will be
    ignored by the tool -->

    <!-- Description>Check fpf SPI Boot Source against expected value</
    Description -->

    <!-- IntelRequired>False</IntelRequired -->

    <!-- Dependencies></Dependencies -->

    <!-- Level>1</Level -->

    <!-- End of uneditable fields -->

    <!-- Please edit the fields below ONLY with the State or ErrAction -->

    <State>Enabled</State>

    <ErrAction>ErrorContinue</ErrAction>

    <RequiredValue format="Enabled/00/Disabled/01"
    example="Enabled"> </RequiredValue>

    </eolconfig>

    <eolconfig name="SOC Config Lock State">

        <!-- The commented fields below CANNOT be edited. Any edits will be
        ignored by the tool -->

        <!-- Description>Check fpf SOC Config Lock State against expected
        value</Description -->
```



```

        <!-- IntelRequired>False</IntelRequired -->

        <!-- Dependencies></Dependencies -->

        <!-- Level>1</Level -->

        <!-- End of uneditable fields -->

        <!-- Please edit the fields below ONLY with the State or ErrAction -->

        <State>Enabled</State>

        <ErrAction>ErrorContinue</ErrAction>

        <RequiredValue format="Disabled/00/Enabled/01"
example="Disabled"> </RequiredValue>

    </eolconfig>

    <eolconfig name="RPMB Migration Done">

        <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

        <!-- Description>Check fpf RPMB Migration Done against expected
value</Description -->

        <!-- IntelRequired>False</IntelRequired -->

        <!-- Dependencies></Dependencies -->

        <!-- Level>1</Level -->

        <!-- End of uneditable fields -->

        <!-- Please edit the fields below ONLY with the State or ErrAction -->

        <State>Enabled</State>

        <ErrAction>ErrorContinue</ErrAction>

        <RequiredValue format="Disabled/00/Enabled/01"
example="Disabled"> </RequiredValue>

    </eolconfig>

    <eolconfig name="ROT Anti Rollback">

        <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

        <!-- Description>Check fpf ROT Anti Rollback against expected value</
Description -->

```



```
<!-- IntelRequired>False</IntelRequired -->

<!-- Dependencies></Dependencies -->

<!-- Level>1</Level -->

<!-- End of uneditable fields -->

<!-- Please edit the fields below ONLY with the State or ErrAction -->

<State>Enabled</State>

<ErrAction>ErrorContinue</ErrAction>

<RequiredValue format="Disabled/00/Enabled/01"
example="Disabled"> </RequiredValue>

</eolconfig>

<eolconfig name="RBE Anti Rollback">

    <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

    <!-- Description>Check fpf RBE Anti Rollback against expected value</
Description -->

    <!-- IntelRequired>False</IntelRequired -->

    <!-- Dependencies></Dependencies -->

    <!-- Level>1</Level -->

    <!-- End of uneditable fields -->

    <!-- Please edit the fields below ONLY with the State or ErrAction -->

    <State>Enabled</State>

    <ErrAction>ErrorContinue</ErrAction>

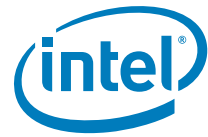
    <RequiredValue format="Disabled/00/Enabled/01"
example="Disabled"> </RequiredValue>

</eolconfig>

<eolconfig name="Persistent PRTC Backup Power">

    <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

    <!-- Description>Check fpf Persistent PRTC Backup Power against
expected value</Description -->
```



```

        <!-- IntelRequired>False</IntelRequired -->

        <!-- Dependencies></Dependencies -->

        <!-- Level>1</Level -->

        <!-- End of uneditable fields -->

        <!-- Please edit the fields below ONLY with the State or ErrAction -->

        <State>Enabled</State>

        <ErrAction>ErrorContinue</ErrAction>

        <RequiredValue format="Enabled/00/Disabled/01"
example="Enabled"> </RequiredValue>

    </eolconfig>

    <eolconfig name="PMC Anti Rollback">

        <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

        <!-- Description>Check fpf PMC Anti Rollback against expected value</
Description -->

        <!-- IntelRequired>False</IntelRequired -->

        <!-- Dependencies></Dependencies -->

        <!-- Level>1</Level -->

        <!-- End of uneditable fields -->

        <!-- Please edit the fields below ONLY with the State or ErrAction -->

        <State>Enabled</State>

        <ErrAction>ErrorContinue</ErrAction>

        <RequiredValue format="Disabled/00/Enabled/01"
example="Disabled"> </RequiredValue>

    </eolconfig>

    <eolconfig name="PCH_COSIG">

        <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

        <!-- Description>Check fpf PCH_COSIG against expected value</
Description -->

```



```
<!-- IntelRequired>False</IntelRequired -->

<!-- Dependencies></Dependencies -->

<!-- Level>1</Level -->

<!-- End of uneditable fields -->

<!-- Please edit the fields below ONLY with the State or ErrAction -->

<State>Enabled</State>

<ErrAction>ErrorContinue</ErrAction>

<RequiredValue format="Disabled/00/Enabled/01"
example="Disabled"> </RequiredValue>

</eolconfig>

<eolconfig name="Force Boot Guard ACM">

    <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

    <!-- Description>Check fpf Force Boot Guard ACM against expected
value</Description -->

    <!-- IntelRequired>False</IntelRequired -->

    <!-- Dependencies></Dependencies -->

    <!-- Level>1</Level -->

    <!-- End of uneditable fields -->

    <!-- Please edit the fields below ONLY with the State or ErrAction -->

    <State>Enabled</State>

    <ErrAction>ErrorContinue</ErrAction>

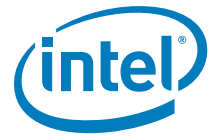
    <RequiredValue format="Disabled/00/Enabled/01"
example="Disabled"> </RequiredValue>

</eolconfig>

<eolconfig name="Key Manifest ID">

    <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

    <!-- Description>Check fpf Key Manifest ID against expected value</
Description -->
```

```

        <!-- IntelRequired>False</IntelRequired -->

        <!-- Dependencies></Dependencies -->

        <!-- Level>1</Level -->

        <!-- End of uneditable fields -->

        <!-- Please edit the fields below ONLY with the State or ErrAction -->

        <State>Enabled</State>

        <ErrAction>ErrorContinue</ErrAction>

        <RequiredValue format="Hex number with 0x prefix."
example="0x00"> </RequiredValue>

    </eolconfig>

    <eolconfig name="OEM Secure Boot Policy">

        <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

        <!-- Description>Check fpf OEM Secure Boot Policy against expected
value</Description -->

        <!-- IntelRequired>False</IntelRequired -->

        <!-- Dependencies></Dependencies -->

        <!-- Level>1</Level -->

        <!-- End of uneditable fields -->

        <!-- Please edit the fields below ONLY with the State or ErrAction -->

        <State>Enabled</State>

        <ErrAction>ErrorContinue</ErrAction>

        <RequiredValue format="Hex number with 0x prefix."
example="0x0000"> </RequiredValue>

    </eolconfig>

    <eolconfig name="OEM Platform ID">

        <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

        <!-- Description>Check fpf OEM Platform ID against expected value</
Description -->

```



```
<!-- IntelRequired>False</IntelRequired -->

<!-- Dependencies></Dependencies -->

<!-- Level>1</Level -->

<!-- End of uneditable fields -->

<!-- Please edit the fields below ONLY with the State or ErrAction -->

<State>Enabled</State>

<ErrAction>ErrorContinue</ErrAction>

<RequiredValue format="Hex number with 0x prefix."
example="0x0000"> </RequiredValue>

</eolconfig>

<eolconfig name="OEM Key Revocation State">

    <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

    <!-- Description>Check fpf OEM Key Revocation State against
expected value</Description -->

    <!-- IntelRequired>False</IntelRequired -->

    <!-- Dependencies></Dependencies -->

    <!-- Level>1</Level -->

    <!-- End of uneditable fields -->

    <!-- Please edit the fields below ONLY with the State or ErrAction -->

    <State>Enabled</State>

    <ErrAction>ErrorContinue</ErrAction>

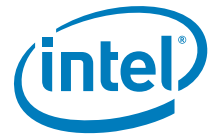
    <RequiredValue format="Disabled/00/Enabled/01"
example="Disabled"> </RequiredValue>

</eolconfig>

<eolconfig name="OEM Key Manifest">

    <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

    <!-- Description>Check fpf OEM Key Manifest against expected
value</Description -->
```



```

        <!-- IntelRequired>False</IntelRequired -->

        <!-- Dependencies></Dependencies -->

        <!-- Level>1</Level -->

        <!-- End of uneditable fields -->

        <!-- Please edit the fields below ONLY with the State or ErrAction -->

        <State>Enabled</State>

        <ErrAction>ErrorContinue</ErrAction>

        <RequiredValue format="Disabled/00/Enabled/01"
example="Disabled"> </RequiredValue>

    </eolconfig>

    <eolconfig name="OEM KM Anti Rollback">

        <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

        <!-- Description>Check fpf OEM KM Anti Rollback against expected
value</Description -->

        <!-- IntelRequired>False</IntelRequired -->

        <!-- Dependencies></Dependencies -->

        <!-- Level>1</Level -->

        <!-- End of uneditable fields -->

        <!-- Please edit the fields below ONLY with the State or ErrAction -->

        <State>Enabled</State>

        <ErrAction>ErrorContinue</ErrAction>

        <RequiredValue format="Disabled/00/Enabled/01"
example="Disabled"> </RequiredValue>

    </eolconfig>

    <eolconfig name="OEM ID">

        <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

        <!-- Description>Check fpf OEM ID against expected value</
Description -->

```



```
<!-- IntelRequired>False</IntelRequired -->

<!-- Dependencies></Dependencies -->

<!-- Level>1</Level -->

<!-- End of uneditable fields -->

<!-- Please edit the fields below ONLY with the State or ErrAction -->

<State>Enabled</State>

<ErrAction>ErrorContinue</ErrAction>

<RequiredValue format="Hex number with 0x prefix."
example="0x0000"> </RequiredValue>

</eolconfig>

<eolconfig name="Intel(R) PTT">

    <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

    <!-- Description>Check fpf Intel(R) PTT against expected value</
Description -->

    <!-- IntelRequired>False</IntelRequired -->

    <!-- Dependencies></Dependencies -->

    <!-- Level>1</Level -->

    <!-- End of uneditable fields -->

    <!-- Please edit the fields below ONLY with the State or ErrAction -->

    <State>Enabled</State>

    <ErrAction>ErrorContinue</ErrAction>

    <RequiredValue format="Disabled/00/Enabled/01"
example="Disabled"> </RequiredValue>

</eolconfig>

<eolconfig name="Intel(R) AMT HW Status">

    <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

    <!-- Description>Check fpf Intel(R) AMT HW Status against expected
value</Description -->
```



```

        <!-- IntelRequired>False</IntelRequired -->

        <!-- Dependencies></Dependencies -->

        <!-- Level>1</Level -->

        <!-- End of uneditable fields -->

        <!-- Please edit the fields below ONLY with the State or ErrAction -->

        <State>Enabled</State>

        <ErrAction>ErrorContinue</ErrAction>

        <RequiredValue format="Enabled/00/Disabled/01"
example="Enabled"> </RequiredValue>

    </eolconfig>

    <eolconfig name="IDLM Anti Rollback">

        <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

        <!-- Description>Check fpf IDLM Anti Rollback against expected
value</Description -->

        <!-- IntelRequired>False</IntelRequired -->

        <!-- Dependencies></Dependencies -->

        <!-- Level>1</Level -->

        <!-- End of uneditable fields -->

        <!-- Please edit the fields below ONLY with the State or ErrAction -->

        <State>Enabled</State>

        <ErrAction>ErrorContinue</ErrAction>

        <RequiredValue format="Disabled/00/Enabled/01"
example="Disabled"> </RequiredValue>

    </eolconfig>

    <eolconfig name="Glitch Detection Enabled">

        <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

        <!-- Description>Check fpf Glitch Detection Enabled against expected
value</Description -->

```



```
<!-- IntelRequired>False</IntelRequired -->

<!-- Dependencies></Dependencies -->

<!-- Level>1</Level -->

<!-- End of uneditable fields -->

<!-- Please edit the fields below ONLY with the State or ErrAction -->

<State>Enabled</State>

<ErrAction>ErrorContinue</ErrAction>

<RequiredValue format="Disabled/00/Enabled/01"
example="Disabled"> </RequiredValue>

</eolconfig>

<eolconfig name="Glitch Detection Disabled">

    <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

    <!-- Description>Check fpf Glitch Detection Disabled against expected
value</Description -->

    <!-- IntelRequired>False</IntelRequired -->

    <!-- Dependencies></Dependencies -->

    <!-- Level>1</Level -->

    <!-- End of uneditable fields -->

    <!-- Please edit the fields below ONLY with the State or ErrAction -->

    <State>Enabled</State>

    <ErrAction>ErrorContinue</ErrAction>

    <RequiredValue format="Enabled/00/Disabled/01"
example="Enabled"> </RequiredValue>

</eolconfig>

<eolconfig name="Glitch Detection Cfg Done">

    <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

    <!-- Description>Check fpf Glitch Detection Cfg Done against expected
value</Description -->
```



```

        <!-- IntelRequired>False</IntelRequired -->

        <!-- Dependencies></Dependencies -->

        <!-- Level>1</Level -->

        <!-- End of uneditable fields -->

        <!-- Please edit the fields below ONLY with the State or ErrAction -->

        <State>Enabled</State>

        <ErrAction>ErrorContinue</ErrAction>

        <RequiredValue format="Disabled/00/Enabled/01"
example="Disabled"> </RequiredValue>

    </eolconfig>

    <eolconfig name="Flash Descriptor Verification">

        <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

        <!-- Description>Check fpf Flash Descriptor Verification against
expected value</Description -->

        <!-- IntelRequired>False</IntelRequired -->

        <!-- Dependencies></Dependencies -->

        <!-- Level>1</Level -->

        <!-- End of uneditable fields -->

        <!-- Please edit the fields below ONLY with the State or ErrAction -->

        <State>Enabled</State>

        <ErrAction>ErrorContinue</ErrAction>

        <RequiredValue format="Disabled/00/Enabled/01"
example="Disabled"> </RequiredValue>

    </eolconfig>

    <eolconfig name="Error Enforcement Policy 1">

        <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

        <!-- Description>Check fpf Error Enforcement Policy 1 against
expected value</Description -->

```



```
<!-- IntelRequired>False</IntelRequired -->

<!-- Dependencies></Dependencies -->

<!-- Level>1</Level -->

<!-- End of uneditable fields -->

<!-- Please edit the fields below ONLY with the State or ErrAction -->

<State>Enabled</State>

<ErrAction>ErrorContinue</ErrAction>

<RequiredValue format="Disabled/00/Enabled/01"
example="Disabled"> </RequiredValue>

</eolconfig>

<eolconfig name="Error Enforcement Policy 0">

    <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

    <!-- Description>Check fpf Error Enforcement Policy 0 against
expected value</Description -->

    <!-- IntelRequired>False</IntelRequired -->

    <!-- Dependencies></Dependencies -->

    <!-- Level>1</Level -->

    <!-- End of uneditable fields -->

    <!-- Please edit the fields below ONLY with the State or ErrAction -->

    <State>Enabled</State>

    <ErrAction>ErrorContinue</ErrAction>

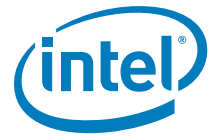
    <RequiredValue format="Disabled/00/Enabled/01"
example="Disabled"> </RequiredValue>

</eolconfig>

<eolconfig name="EMMC Boot Source">

    <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

    <!-- Description>Check fpf EMMC Boot Source against expected
value</Description -->
```

```

        <!-- IntelRequired>False</IntelRequired -->

        <!-- Dependencies></Dependencies -->

        <!-- Level>1</Level -->

        <!-- End of uneditable fields -->

        <!-- Please edit the fields below ONLY with the State or ErrAction -->

        <State>Enabled</State>

        <ErrAction>ErrorContinue</ErrAction>

        <RequiredValue format="Enabled/00/Disabled/01"
example="Enabled"> </RequiredValue>

    </eolconfig>

    <eolconfig name="DNX Anti Rollback">

        <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

        <!-- Description>Check fpf DNX Anti Rollback against expected
value</Description -->

        <!-- IntelRequired>False</IntelRequired -->

        <!-- Dependencies></Dependencies -->

        <!-- Level>1</Level -->

        <!-- End of uneditable fields -->

        <!-- Please edit the fields below ONLY with the State or ErrAction -->

        <State>Enabled</State>

        <ErrAction>ErrorContinue</ErrAction>

        <RequiredValue format="Disabled/00/Enabled/01"
example="Disabled"> </RequiredValue>

    </eolconfig>

    <eolconfig name="DAL OEM Signing">

        <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

        <!-- Description>Check fpf DAL OEM Signing against expected value</
Description -->

```



```
<!-- IntelRequired>False</IntelRequired -->

<!-- Dependencies></Dependencies -->

<!-- Level>1</Level -->

<!-- End of uneditable fields -->

<!-- Please edit the fields below ONLY with the State or ErrAction -->

<State>Enabled</State>

<ErrAction>ErrorContinue</ErrAction>

<RequiredValue format="Disabled/00/Enabled/01"
example="Disabled"> </RequiredValue>

</eolconfig>

<eolconfig name="CPU Co-signing">

    <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

    <!-- Description>Check fpf CPU Co-signing against expected value</
Description -->

    <!-- IntelRequired>False</IntelRequired -->

    <!-- Dependencies></Dependencies -->

    <!-- Level>1</Level -->

    <!-- End of uneditable fields -->

    <!-- Please edit the fields below ONLY with the State or ErrAction -->

    <State>Enabled</State>

    <ErrAction>ErrorContinue</ErrAction>

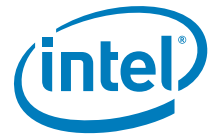
    <RequiredValue format="Disabled/00/Enabled/01"
example="Disabled"> </RequiredValue>

</eolconfig>

<eolconfig name="BSMM Anti Rollback">

    <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

    <!-- Description>Check fpf BSMM Anti Rollback against expected
value</Description -->
```



```

        <!-- IntelRequired>False</IntelRequired -->

        <!-- Dependencies></Dependencies -->

        <!-- Level>1</Level -->

        <!-- End of uneditable fields -->

        <!-- Please edit the fields below ONLY with the State or ErrAction -->

        <State>Enabled</State>

        <ErrAction>ErrorContinue</ErrAction>

        <RequiredValue format="Disabled/00/Enabled/01"
example="Disabled"> </RequiredValue>

    </eolconfig>

    <eolconfig name="2nd OEM RSA Key size">

        <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

        <!-- Description>Check fpf 2nd OEM RSA Key size against expected
value</Description -->

        <!-- IntelRequired>False</IntelRequired -->

        <!-- Dependencies></Dependencies -->

        <!-- Level>1</Level -->

        <!-- End of uneditable fields -->

        <!-- Please edit the fields below ONLY with the State or ErrAction -->

        <State>Enabled</State>

        <ErrAction>ErrorContinue</ErrAction>

        <RequiredValue format="Disabled/00/Enabled/01"
example="Disabled"> </RequiredValue>

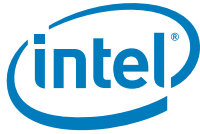
    </eolconfig>

    <eolconfig name="2nd OEM Public Key Hash">

        <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

        <!-- Description>Check fpf 2nd OEM Public Key Hash against expected
value</Description -->

```



```
<!-- IntelRequired>False</IntelRequired -->

<!-- Dependencies></Dependencies -->

<!-- Level>1</Level -->

<!-- End of uneditable fields -->

<!-- Please edit the fields below ONLY with the State or ErrAction -->

<State>Enabled</State>

<ErrAction>ErrorContinue</ErrAction>

<RequiredValue format="48 hex pairs with space between pairs"
example="04 AB F3 45 03 1D EF A2 B7 E8 98 79 10 45 AB DE F2 35 49 A0 01 35 78 29
37 AB DE EF FA 10 EF 33 04 AB F3 45 03 1D EF A2 B7 E8 98 79 10 45 AB DE"> </
RequiredValue>

</eolconfig>

<eolconfig name="2nd OEM Key Hash valid">

    <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

    <!-- Description>Check fpf 2nd OEM Key Hash valid against expected
value</Description -->

    <!-- IntelRequired>False</IntelRequired -->

    <!-- Dependencies></Dependencies -->

    <!-- Level>1</Level -->

    <!-- End of uneditable fields -->

    <!-- Please edit the fields below ONLY with the State or ErrAction -->

    <State>Enabled</State>

    <ErrAction>ErrorContinue</ErrAction>

    <RequiredValue format="Disabled/00/Enabled/01"
example="Disabled"> </RequiredValue>

</eolconfig>

<eolconfig name="1st OEM Public Key Hash">

    <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->
```



```
<!-- Description>Check fpf 1st OEM Public Key Hash against expected
value</Description -->
```

```
<!-- IntelRequired>False</IntelRequired -->
```

```
<!-- Dependencies></Dependencies -->
```

```
<!-- Level>1</Level -->
```

```
<!-- End of uneditable fields -->
```

```
<!-- Please edit the fields below ONLY with the State or ErrAction -->
```

```
<State>Enabled</State>
```

```
<ErrAction>ErrorContinue</ErrAction>
```

```
<RequiredValue format="48 hex pairs with space between pairs"
example="04 AB F3 45 03 1D EF A2 B7 E8 98 79 10 45 AB DE F2 35 49 A0 01 35 78 29
37 AB DE EF FA 10 EF 33 04 AB F3 45 03 1D EF A2 B7 E8 98 79 10 45 AB DE"> </
RequiredValue>
```

```
</eolconfig>
```

```
<eolconfig name="">
```

```
<!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->
```

```
<!-- Description>Check fpf against expected value</Description -->
```

```
<!-- IntelRequired>False</IntelRequired -->
```

```
<!-- Dependencies></Dependencies -->
```

```
<!-- Level>1</Level -->
```

```
<!-- End of uneditable fields -->
```

```
<!-- Please edit the fields below ONLY with the State or ErrAction -->
```

```
<State>Enabled</State>
```

```
<ErrAction>ErrorContinue</ErrAction>
```

```
<RequiredValue format="Hex number with 0x prefix."
example="0x00"> </RequiredValue>
```

```
</eolconfig>
```

```
<eolconfig name="Attestation KeyBox test">
```

```
<!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->
```



```
-->      <!-- Description>Check Attestation KeyBox data validity</Description>

      <!-- IntelRequired>False</IntelRequired -->

      <!-- Dependencies></Dependencies -->

      <!-- Level>1</Level -->

      <!-- End of uneditable fields -->

      <!-- Please edit the fields below ONLY with the State or ErrAction -->

      <State>Enabled</State>

      <ErrAction>ErrorContinue</ErrAction>

</eolconfig>

<eolconfig name="HW Glitch Detection State">

      <!-- The commented fields below CANNOT be edited. Any edits will be
      ignored by the tool -->

      <!-- Description>Check HW Glitch Detection state</Description -->

      <!-- IntelRequired>False</IntelRequired -->

      <!-- Dependencies></Dependencies -->

      <!-- Level>3</Level -->

      <!-- End of uneditable fields -->

      <!-- Please edit the fields below ONLY with the State or ErrAction -->

      <State>Enabled</State>

      <ErrAction>ErrorContinue</ErrAction>

      <RequiredValue format="Disabled/Enabled" example="Disabled"> </
RequiredValue>

</eolconfig>

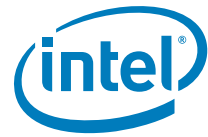
<eolconfig name="Enforce RPMC Support">

      <!-- The commented fields below CANNOT be edited. Any edits will be
      ignored by the tool -->

      <!-- Description>Check if RPMC configuration is enabled</Description>

-->

      <!-- IntelRequired>False</IntelRequired -->
```



```

        <!-- Dependencies>SPI_DEP</Dependencies -->

        <!-- Level>1</Level -->

        <!-- End of uneditable fields -->

        <!-- Please edit the fields below ONLY with the State or ErrAction -->

        <State>Enabled</State>

        <ErrAction>ErrorContinue</ErrAction>

    </eolconfig>

    <eolconfig name="PCHC FW version">

        <!-- The commented fields below CANNOT be edited. Any edits will be
        ignored by the tool -->

        <!-- Description>Check PCHC FW version against expected value</
        Description -->

        <!-- IntelRequired>False</IntelRequired -->

        <!-- Dependencies>PCHC_PARTITION</Dependencies -->

        <!-- Level>1</Level -->

        <!-- End of uneditable fields -->

        <!-- Please edit the fields below ONLY with the State or ErrAction -->

        <State>Enabled</State>

        <ErrAction>ErrorContinue</ErrAction>

        <RequiredValue format="major_ver.minor_ver.hotfix_ver.build_num"
        example="1.2.3.0004"> </RequiredValue>

    </eolconfig>

    <eolconfig name="Boot Guard status">

        <!-- The commented fields below CANNOT be edited. Any edits will be
        ignored by the tool -->

        <!-- Description>Verifies validity of Boot Guard FW status. As a
        RequiredValue provide Profile Level for profile dependent checks</Description -->

        <!-- IntelRequired>True</IntelRequired -->

        <!-- Dependencies></Dependencies -->

        <!-- Level>1</Level -->

```



```
<!-- End of uneditable fields -->

<!-- Please edit the fields below ONLY with the State or ErrAction -->

<State>Enabled</State>

<ErrAction>ErrorContinue</ErrAction>

<RequiredValue format="Hex number with 0x prefix."
example="0x00"> </RequiredValue>

</eolconfig>

<eolconfig name="FW status">

    <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

    <!-- Description>Verifies validity of FW status</Description -->

    <!-- IntelRequired>True</IntelRequired -->

    <!-- Dependencies></Dependencies -->

    <!-- Level>1</Level -->

    <!-- End of uneditable fields -->

    <!-- Please edit the fields below ONLY with the State or ErrAction -->

    <State>Enabled</State>

    <ErrAction>ErrorContinue</ErrAction>

</eolconfig>

<eolconfig name="Checking NVM for fatal flash logs">

    <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

    <!-- Description>Inspection of NVM found fatal flash logs</Description
-->

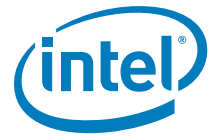
    <!-- IntelRequired>False</IntelRequired -->

    <!-- Dependencies></Dependencies -->

    <!-- Level>1</Level -->

    <!-- End of uneditable fields -->

    <!-- Please edit the fields below ONLY with the State or ErrAction -->
```

```

        <State>Enabled</State>

        <ErrAction>ErrorContinue</ErrAction>

    </eolconfig>

    <eolconfig name="Confirm ARB SVN value">

        <!-- The commented fields below CANNOT be edited. Any edits will be
        ignored by the tool -->

        <!-- Description>Confirms that the minimum ARB SVN saved in the
        PCH fuses matches the ARB SVN of the FW image</Description -->

        <!-- IntelRequired>False</IntelRequired -->

        <!-- Dependencies></Dependencies -->

        <!-- Level>1</Level -->

        <!-- End of uneditable fields -->

        <!-- Please edit the fields below ONLY with the State or ErrAction -->

        <State>Enabled</State>

        <ErrAction>ErrorContinue</ErrAction>

    </eolconfig>

    <eolconfig name="PCH Unlocked state">

        <!-- The commented fields below CANNOT be edited. Any edits will be
        ignored by the tool -->

        <!-- Description>Verifies that PCH is locked</Description -->

        <!-- IntelRequired>True</IntelRequired -->

        <!-- Dependencies></Dependencies -->

        <!-- Level>1</Level -->

        <!-- End of uneditable fields -->

        <!-- Please edit the fields below ONLY with the State or ErrAction -->

        <State>Enabled</State>

        <ErrAction>ErrorContinue</ErrAction>

    </eolconfig>

    <eolconfig name="HW Binding Disabled">

```



```
<!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

<!-- Description>Verifies that HW binding is disabled</Description -->

<!-- IntelRequired>True</IntelRequired -->

<!-- Dependencies></Dependencies -->

<!-- Level>1</Level -->

<!-- End of uneditable fields -->

<!-- Please edit the fields below ONLY with the State or ErrAction -->

<State>Enabled</State>

<ErrAction>ErrorContinue</ErrAction>

</eolconfig>

<eolconfig name="SOC Config Lock">

    <!-- The commented fields below CANNOT be edited. Any edits will be
    ignored by the tool -->

    <!-- Description>Check that SOC Config Lock FPF is set.</Description -
->

    <!-- IntelRequired>True</IntelRequired -->

    <!-- Dependencies></Dependencies -->

    <!-- Level>1</Level -->

    <!-- End of uneditable fields -->

    <!-- Please edit the fields below ONLY with the State or ErrAction -->

    <State>Enabled</State>

    <ErrAction>ErrorContinue</ErrAction>

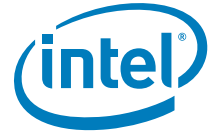
</eolconfig>

<eolconfig name="FPFs in UEP Committed">

    <!-- The commented fields below CANNOT be edited. Any edits will be
    ignored by the tool -->

    <!-- Description>Check that FPFs in UEP are committed to
    Hardware.</Description -->

    <!-- IntelRequired>True</IntelRequired -->
```



```

        <!-- Dependencies></Dependencies -->

        <!-- Level>1</Level -->

        <!-- End of uneditable fields -->

        <!-- Please edit the fields below ONLY with the State or ErrAction -->

        <State>Enabled</State>

        <ErrAction>ErrorContinue</ErrAction>

    </eolconfig>

    <eolconfig name="Validate Keybox Provisioning">

        <!-- The commented fields below CANNOT be edited. Any edits will be
        ignored by the tool -->

        <!-- Description>Check to see if Keybox is provisioned</Description --
    >

        <!-- IntelRequired>False</IntelRequired -->

        <!-- Dependencies></Dependencies -->

        <!-- Level>3</Level -->

        <!-- End of uneditable fields -->

        <!-- Please edit the fields below ONLY with the State or ErrAction -->

        <State>Enabled</State>

        <ErrAction>ErrorContinue</ErrAction>

    </eolconfig>

    <eolconfig name="Firmware Update OEM ID">

        <!-- The commented fields below CANNOT be edited. Any edits will be
        ignored by the tool -->

        <!-- Description>Check Firmware Update OEM ID value</Description -
    ->

        <!-- IntelRequired>False</IntelRequired -->

        <!-- Dependencies></Dependencies -->

        <!-- Level>3</Level -->

        <!-- End of uneditable fields -->

```



```
<!-- Please edit the fields below ONLY with the State or ErrAction -->

<State>Enabled</State>

<ErrAction>ErrorContinue</ErrAction>

<RequiredValue format="UUID" example="00000000-0000-0000-
0000-000000000000"> </RequiredValue>

</eolconfig>

<eolconfig name="GBE version">

    <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

    <!-- Description>Check Gbe Version against expected value</
Description -->

    <!-- IntelRequired>False</IntelRequired -->

    <!-- Dependencies>SPI_DEP</Dependencies -->

    <!-- Level>1</Level -->

    <!-- End of uneditable fields -->

    <!-- Please edit the fields below ONLY with the State or ErrAction -->

    <State>Enabled</State>

    <ErrAction>ErrorContinue</ErrAction>

    <RequiredValue format="major_ver.minor_ver" example="0.6"> </
RequiredValue>

</eolconfig>

<eolconfig name="BIOS version">

    <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

    <!-- Description>Check BIOS Version against expected value</
Description -->

    <!-- IntelRequired>False</IntelRequired -->

    <!-- Dependencies></Dependencies -->

    <!-- Level>1</Level -->

    <!-- End of uneditable fields -->
```



```

        <!-- Please edit the fields below ONLY with the State or ErrAction -->

        <State>Enabled</State>

        <ErrAction>ErrorContinue</ErrAction>

        <RequiredValue format="Customer specific"
example="HSLPTU1.86C.0117.R00.1303102001"> </RequiredValue>

        </eolconfig>

        <eolconfig name="ME FW version">

                <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

                <!-- Description>Check Firmware Version against expected value</
Description -->

                <!-- IntelRequired>False</IntelRequired -->

                <!-- Dependencies></Dependencies -->

                <!-- Level>1</Level -->

                <!-- End of uneditable fields -->

                <!-- Please edit the fields below ONLY with the State or ErrAction -->

                <State>Enabled</State>

                <ErrAction>ErrorContinue</ErrAction>

                <RequiredValue format="major_ver.minor_ver.hotfix_ver.build_num
H|LP|ULT Corporate|Consumer|Slim" example="12.0.0.1040 LP Consumer"> </
RequiredValue>

        </eolconfig>

        <eolconfig name="Security Descriptor Override (SDO) check">

                <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

                <!-- Description>Check SDO pin</Description -->

                <!-- IntelRequired>True</IntelRequired -->

                <!-- Dependencies>SPI_DEP</Dependencies -->

                <!-- Level>1</Level -->

                <!-- End of uneditable fields -->

```



```
<!-- Please edit the fields below ONLY with the State or ErrAction -->

<State>Enabled</State>

<ErrAction>ErrorContinue</ErrAction>

</eolconfig>

<eolconfig name="RPMC Configuration">

    <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

    <!-- Description>Check RPMC configuration</Description -->

    <!-- IntelRequired>True</IntelRequired -->

    <!-- Dependencies>SPI_DEP</Dependencies -->

    <!-- Level>1</Level -->

    <!-- End of uneditable fields -->

    <!-- Please edit the fields below ONLY with the State or ErrAction -->

    <State>Enabled</State>

    <ErrAction>ErrorContinue</ErrAction>

</eolconfig>

<eolconfig name="EC Write Access Permissions">

    <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

    <!-- Description>Check EC write access</Description -->

    <!-- IntelRequired>True</IntelRequired -->

    <!-- Dependencies>SPI_DEP</Dependencies -->

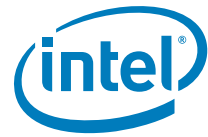
    <!-- Level>2</Level -->

    <!-- End of uneditable fields -->

    <!-- Please edit the fields below ONLY with the State or ErrAction -->

    <State>Enabled</State>

    <ErrAction>ErrorContinue</ErrAction>
```



<RequiredValue format="Hex number with 0x prefix."
example="0x0101. Value left empty will result in checking against Intel recommended values."> </RequiredValue>

</eolconfig>

<eolconfig name="EC Read Access Permissions">

<!-- The commented fields below CANNOT be edited. Any edits will be ignored by the tool -->

<!-- Description>Check EC read access</Description -->

<!-- IntelRequired>True</IntelRequired -->

<!-- Dependencies>SPI_DEP</Dependencies -->

<!-- Level>2</Level -->

<!-- End of uneditable fields -->

<!-- Please edit the fields below ONLY with the State or ErrAction -->

<State>Enabled</State>

<ErrAction>ErrorContinue</ErrAction>

<RequiredValue format="Hex number with 0x prefix."
example="0x0101. Value left empty will result in checking against Intel recommended values."> </RequiredValue>

</eolconfig>

<eolconfig name="BIOS Write Access Permissions">

<!-- The commented fields below CANNOT be edited. Any edits will be ignored by the tool -->

<!-- Description>Check BIOS write access</Description -->

<!-- IntelRequired>True</IntelRequired -->

<!-- Dependencies>SPI_DEP</Dependencies -->

<!-- Level>2</Level -->

<!-- End of uneditable fields -->

<!-- Please edit the fields below ONLY with the State or ErrAction -->

<State>Enabled</State>

<ErrAction>ErrorContinue</ErrAction>



<RequiredValue format="Hex number with 0x prefix."
example="0x0101. Value left empty will result in checking against Intel recommended
values."> </RequiredValue>

</eolconfig>

<eolconfig name="BIOS Read Access Permissions">

<!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

<!-- Description>Check BIOS read access</Description -->

<!-- IntelRequired>True</IntelRequired -->

<!-- Dependencies>SPI_DEP</Dependencies -->

<!-- Level>2</Level -->

<!-- End of uneditable fields -->

<!-- Please edit the fields below ONLY with the State or ErrAction -->

<State>Enabled</State>

<ErrAction>ErrorContinue</ErrAction>

<RequiredValue format="Hex number with 0x prefix."
example="0x0101. Value left empty will result in checking against Intel recommended
values."> </RequiredValue>

</eolconfig>

<eolconfig name="GBE Write Access Permissions">

<!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

<!-- Description>Check GBE write access</Description -->

<!-- IntelRequired>True</IntelRequired -->

<!-- Dependencies>SPI_DEP</Dependencies -->

<!-- Level>2</Level -->

<!-- End of uneditable fields -->

<!-- Please edit the fields below ONLY with the State or ErrAction -->

<State>Enabled</State>

<ErrAction>ErrorContinue</ErrAction>



<RequiredValue format="Hex number with 0x prefix."
example="0x0101. Value left empty will result in checking against Intel recommended values."> </RequiredValue>

</eolconfig>

<eolconfig name="GBE Read Access Permissions">

<!-- The commented fields below CANNOT be edited. Any edits will be ignored by the tool -->

<!-- Description>Check GBE read access</Description -->

<!-- IntelRequired>True</IntelRequired -->

<!-- Dependencies>SPI_DEP</Dependencies -->

<!-- Level>2</Level -->

<!-- End of uneditable fields -->

<!-- Please edit the fields below ONLY with the State or ErrAction -->

<State>Enabled</State>

<ErrAction>ErrorContinue</ErrAction>

<RequiredValue format="Hex number with 0x prefix."
example="0x0101. Value left empty will result in checking against Intel recommended values."> </RequiredValue>

</eolconfig>

<eolconfig name="ME Write Access Permissions">

<!-- The commented fields below CANNOT be edited. Any edits will be ignored by the tool -->

<!-- Description>Check ME write access</Description -->

<!-- IntelRequired>True</IntelRequired -->

<!-- Dependencies>SPI_DEP</Dependencies -->

<!-- Level>2</Level -->

<!-- End of uneditable fields -->

<!-- Please edit the fields below ONLY with the State or ErrAction -->

<State>Enabled</State>

<ErrAction>ErrorContinue</ErrAction>



<RequiredValue format="Hex number with 0x prefix."
example="0x0101. Value left empty will result in checking against Intel recommended
values."> </RequiredValue>

</eolconfig>

<eolconfig name="ME Read Access Permissions">

<!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

<!-- Description>Check ME read access</Description -->

<!-- IntelRequired>True</IntelRequired -->

<!-- Dependencies>SPI_DEP</Dependencies -->

<!-- Level>2</Level -->

<!-- End of uneditable fields -->

<!-- Please edit the fields below ONLY with the State or ErrAction -->

<State>Enabled</State>

<ErrAction>ErrorContinue</ErrAction>

<RequiredValue format="Hex number with 0x prefix."
example="0x0101. Value left empty will result in checking against Intel recommended
values."> </RequiredValue>

</eolconfig>

<eolconfig name="ME Manufacturing Mode status">

<!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

<!-- Description>Check End of Manufacturing Mode against Intel
recommended value</Description -->

<!-- IntelRequired>True</IntelRequired -->

<!-- Dependencies></Dependencies -->

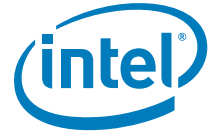
<!-- Level>1</Level -->

<!-- End of uneditable fields -->

<!-- Please edit the fields below ONLY with the State or ErrAction -->

<State>Enabled</State>

<ErrAction>ErrorContinue</ErrAction>



```

</eolconfig>

<eolconfig name="EOP status check">

    <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

    <!-- Description>Check that EOP was sent/recieved</Description -->

    <!-- IntelRequired>True</IntelRequired -->

    <!-- Dependencies></Dependencies -->

    <!-- Level>1</Level -->

    <!-- End of uneditable fields -->

    <!-- Please edit the fields below ONLY with the State or ErrAction -->

    <State>Enabled</State>

    <ErrAction>ErrorContinue</ErrAction>

</eolconfig>

<!-- END OF EOL CONFIG TESTS -->

<!-- EOL VAR TESTS -->

<eolvar name="EOM Settings">

    <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

    <!-- Description>Test variable against expected value</Description --
>

    <!-- IntelRequired>False</IntelRequired -->

    <!-- Dependencies></Dependencies -->

    <!-- Level>3</Level -->

    <!-- End of uneditable fields -->

    <!-- Please edit the fields below ONLY with the State or ErrAction -->

    <State>Disabled</State>

    <ErrAction>ErrorContinue</ErrAction>

    <RequiredValue format="Lock(Flash,Config)/00/Lock(Flash,Config) on
1st Boot/01/Lock(Config)/02/Lock(Config) on 1st Boot/03/Lock(Flash)/04/Lock(Flash)

```



on 1st Boot/05/Lock(none)/06/Lock(none) on 1st Boot/07"
example="Lock(Flash,Config)"> </RequiredValue>

</eolvar>

<eolvar name="SMx state">

<!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

<!-- Description>Test variable against expected value</Description -->
>

<!-- IntelRequired>False</IntelRequired -->

<!-- Dependencies></Dependencies -->

<!-- Level>3</Level -->

<!-- End of uneditable fields -->

<!-- Please edit the fields below ONLY with the State or ErrAction -->

<State>Disabled</State>

<ErrAction>ErrorContinue</ErrAction>

<RequiredValue format="Disabled/01/Enabled/00"
example="Disabled"> </RequiredValue>

</eolvar>

<eolvar name="CSME Measured Boot to TPM">

<!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

<!-- Description>Test variable against expected value</Description -->
>

<!-- IntelRequired>False</IntelRequired -->

<!-- Dependencies></Dependencies -->

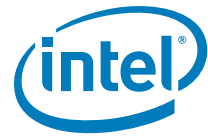
<!-- Level>3</Level -->

<!-- End of uneditable fields -->

<!-- Please edit the fields below ONLY with the State or ErrAction -->

<State>Disabled</State>

<ErrAction>ErrorContinue</ErrAction>



```

        <RequiredValue format="Disabled/00/Enabled/01"
example="Disabled"> </RequiredValue>

    </eolvar>

    <eolvar name="Intel(R) ME Region Flash Protection Override">

        <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

        <!-- Description>Test variable against expected value</Description --
>

        <!-- IntelRequired>False</IntelRequired -->

        <!-- Dependencies></Dependencies -->

        <!-- Level>3</Level -->

        <!-- End of uneditable fields -->

        <!-- Please edit the fields below ONLY with the State or ErrAction -->

        <State>Disabled</State>

        <ErrAction>ErrorContinue</ErrAction>

        <RequiredValue format="False/00/True/01" example="False"> </
RequiredValue>

    </eolvar>

    <eolvar name="FW Update State">

        <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

        <!-- Description>Test variable against expected value</Description --
>

        <!-- IntelRequired>False</IntelRequired -->

        <!-- Dependencies></Dependencies -->

        <!-- Level>3</Level -->

        <!-- End of uneditable fields -->

        <!-- Please edit the fields below ONLY with the State or ErrAction -->

        <State>Disabled</State>

        <ErrAction>ErrorContinue</ErrAction>

```



```
<RequiredValue format="Disabled/00/Enabled/01/Full and Partial
disabled/03" example="Disabled"> </RequiredValue>

</eolvar>

<eolvar name="Intel(R) ME CLINK Signal">

    <!-- The commented fields below CANNOT be edited. Any edits will be
    ignored by the tool -->

    <!-- Description>Test variable against expected value</Description --
>

    <!-- IntelRequired>False</IntelRequired -->

    <!-- Dependencies></Dependencies -->

    <!-- Level>3</Level -->

    <!-- End of uneditable fields -->

    <!-- Please edit the fields below ONLY with the State or ErrAction -->

    <State>Disabled</State>

    <ErrAction>ErrorContinue</ErrAction>

    <RequiredValue format="Disabled/00/Enabled/01"
example="Disabled"> </RequiredValue>

</eolvar>

<eolvar name="OEM Tag">

    <!-- The commented fields below CANNOT be edited. Any edits will be
    ignored by the tool -->

    <!-- Description>Test variable against expected value</Description --
>

    <!-- IntelRequired>False</IntelRequired -->

    <!-- Dependencies></Dependencies -->

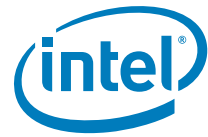
    <!-- Level>3</Level -->

    <!-- End of uneditable fields -->

    <!-- Please edit the fields below ONLY with the State or ErrAction -->

    <State>Disabled</State>

    <ErrAction>ErrorContinue</ErrAction>
```



```

        <RequiredValue format="Hex number with 0x prefix."
example="0x00000000"> </RequiredValue>

    </eolvar>

    <eolvar name="Processor Emulation">

        <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

        <!-- Description>Test variable against expected value</Description --
>

        <!-- IntelRequired>False</IntelRequired -->

        <!-- Dependencies></Dependencies -->

        <!-- Level>3</Level -->

        <!-- End of uneditable fields -->

        <!-- Please edit the fields below ONLY with the State or ErrAction -->

        <State>Disabled</State>

        <ErrAction>ErrorContinue</ErrAction>

        <RequiredValue format="No Emulation/00/vPro/01/Core/02/Celerno/
03/Pentium/04/Xeon/05/Xeon Manageability Capable/06" example="No Emulation">
</RequiredValue>

    </eolvar>

    <eolvar name="PROC_MISSING">

        <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

        <!-- Description>Test variable against expected value</Description --
>

        <!-- IntelRequired>False</IntelRequired -->

        <!-- Dependencies>MOBILE</Dependencies -->

        <!-- Level>3</Level -->

        <!-- End of uneditable fields -->

        <!-- Please edit the fields below ONLY with the State or ErrAction -->

        <State>Disabled</State>

        <ErrAction>ErrorContinue</ErrAction>

```



```
<RequiredValue format="No onboard glue logic/ff" example="No
onboard glue logic"> </RequiredValue>

</eolvar>

<eolvar name="LAN Power Well">

    <!-- The commented fields below CANNOT be edited. Any edits will be
    ignored by the tool -->

    <!-- Description>Test variable against expected value</Description --
>

    <!-- IntelRequired>False</IntelRequired -->

    <!-- Dependencies></Dependencies -->

    <!-- Level>3</Level -->

    <!-- End of uneditable fields -->

    <!-- Please edit the fields below ONLY with the State or ErrAction -->

    <State>Disabled</State>

    <ErrAction>ErrorContinue</ErrAction>

    <RequiredValue format="CoreWell/00/SusWell/01/MEWell/02/
SLP_LAN#(MGPIO3)/03" example="CoreWell"> </RequiredValue>

</eolvar>

<eolvar name="Unconfigure On RTC">

    <!-- The commented fields below CANNOT be edited. Any edits will be
    ignored by the tool -->

    <!-- Description>Test variable against expected value</Description --
>

    <!-- IntelRequired>False</IntelRequired -->

    <!-- Dependencies></Dependencies -->

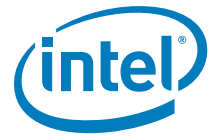
    <!-- Level>3</Level -->

    <!-- End of uneditable fields -->

    <!-- Please edit the fields below ONLY with the State or ErrAction -->

    <State>Disabled</State>

    <ErrAction>ErrorContinue</ErrAction>
```

```

        <RequiredValue format="Enabled/00/Disabled/01"
example="Enabled"> </RequiredValue>

    </eolvar>

    <eolvar name="Intel(R) PTT initial power-up state">

        <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

        <!-- Description>Test variable against expected value</Description --
>

        <!-- IntelRequired>False</IntelRequired -->

        <!-- Dependencies></Dependencies -->

        <!-- Level>3</Level -->

        <!-- End of uneditable fields -->

        <!-- Please edit the fields below ONLY with the State or ErrAction -->

        <State>Disabled</State>

        <ErrAction>ErrorContinue</ErrAction>

        <RequiredValue format="Disabled/00/Enabled/01"
example="Disabled"> </RequiredValue>

    </eolvar>

    <eolvar name="FeatureShipState">

        <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

        <!-- Description>Test variable against expected value</Description --
>

        <!-- IntelRequired>False</IntelRequired -->

        <!-- Dependencies></Dependencies -->

        <!-- Level>3</Level -->

        <!-- End of uneditable fields -->

        <!-- Please edit the fields below ONLY with the State or ErrAction -->

        <State>Disabled</State>

        <ErrAction>ErrorContinue</ErrAction>

```



```
<RequiredValue format="Hex number with 0x prefix."
example="0x00000000"> </RequiredValue>

</eolvar>

<eolvar name="Integrated Sensor Hub Supported">

    <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

    <!-- Description>Test variable against expected value</Description --
>

    <!-- IntelRequired>False</IntelRequired -->

    <!-- Dependencies></Dependencies -->

    <!-- Level>3</Level -->

    <!-- End of uneditable fields -->

    <!-- Please edit the fields below ONLY with the State or ErrAction -->

    <State>Disabled</State>

    <ErrAction>ErrorContinue</ErrAction>

    <RequiredValue format="Disabled/00/Enabled/01"
example="Disabled"> </RequiredValue>

</eolvar>

<eolvar name="Intel(R) PTT Supported">

    <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

    <!-- Description>Test variable against expected value</Description --
>

    <!-- IntelRequired>False</IntelRequired -->

    <!-- Dependencies></Dependencies -->

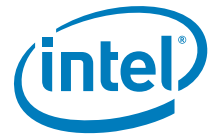
    <!-- Level>3</Level -->

    <!-- End of uneditable fields -->

    <!-- Please edit the fields below ONLY with the State or ErrAction -->

    <State>Disabled</State>

    <ErrAction>ErrorContinue</ErrAction>
```



```

        <RequiredValue format="Disabled/00/Enabled/01"
example="Disabled"> </RequiredValue>

    </eolvar>

    <eolvar name="Firmware Dynamic Application Loader Supported">

        <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

        <!-- Description>Test variable against expected value</Description --
>

        <!-- IntelRequired>False</IntelRequired -->

        <!-- Dependencies></Dependencies -->

        <!-- Level>3</Level -->

        <!-- End of uneditable fields -->

        <!-- Please edit the fields below ONLY with the State or ErrAction -->

        <State>Disabled</State>

        <ErrAction>ErrorContinue</ErrAction>

        <RequiredValue format="No/00/Yes/01" example="No"> </
RequiredValue>

    </eolvar>

    <eolvar name="Intel(R) ME Network Services Supported">

        <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

        <!-- Description>Test variable against expected value</Description --
>

        <!-- IntelRequired>False</IntelRequired -->

        <!-- Dependencies></Dependencies -->

        <!-- Level>3</Level -->

        <!-- End of uneditable fields -->

        <!-- Please edit the fields below ONLY with the State or ErrAction -->

        <State>Disabled</State>

        <ErrAction>ErrorContinue</ErrAction>

```



```
<RequiredValue format="Yes/00/No/01" example="Yes"> </
RequiredValue>

</eolvar>

<eolvar name="TLS Supported">

    <!-- The commented fields below CANNOT be edited. Any edits will be
    ignored by the tool -->

    <!-- Description>Test variable against expected value</Description --
>

    <!-- IntelRequired>False</IntelRequired -->

    <!-- Dependencies></Dependencies -->

    <!-- Level>3</Level -->

    <!-- End of uneditable fields -->

    <!-- Please edit the fields below ONLY with the State or ErrAction -->

    <State>Disabled</State>

    <ErrAction>ErrorContinue</ErrAction>

    <RequiredValue format="No/00/Yes/01" example="No"> </
RequiredValue>

</eolvar>

<eolvar name="PAVP Supported">

    <!-- The commented fields below CANNOT be edited. Any edits will be
    ignored by the tool -->

    <!-- Description>Test variable against expected value</Description --
>

    <!-- IntelRequired>False</IntelRequired -->

    <!-- Dependencies></Dependencies -->

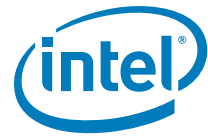
    <!-- Level>3</Level -->

    <!-- End of uneditable fields -->

    <!-- Please edit the fields below ONLY with the State or ErrAction -->

    <State>Disabled</State>

    <ErrAction>ErrorContinue</ErrAction>
```



```

        <RequiredValue format="No/00/Yes/01" example="No"> </
RequiredValue>

    </eolvar>

    <eolvar name="OEMSKURule">

        <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

        <!-- Description>Test variable against expected value</Description --
>

        <!-- IntelRequired>False</IntelRequired -->

        <!-- Dependencies></Dependencies -->

        <!-- Level>3</Level -->

        <!-- End of uneditable fields -->

        <!-- Please edit the fields below ONLY with the State or ErrAction -->

        <State>Disabled</State>

        <ErrAction>ErrorContinue</ErrAction>

        <RequiredValue format="Hex number with 0x prefix."
example="0x00000000"> </RequiredValue>

    </eolvar>

    <eolvar name="Automatic Built in Self Test">

        <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

        <!-- Description>Test variable against expected value</Description --
>

        <!-- IntelRequired>False</IntelRequired -->

        <!-- Dependencies></Dependencies -->

        <!-- Level>3</Level -->

        <!-- End of uneditable fields -->

        <!-- Please edit the fields below ONLY with the State or ErrAction -->

        <State>Disabled</State>

        <ErrAction>ErrorContinue</ErrAction>

```



```
<RequiredValue format="Disabled/00/Enabled/01"
example="Disabled"> </RequiredValue>

</eolvar>

<eolvar name="LSPCON Port Config">

    <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

    <!-- Description>Test variable against expected value</Description --
>

    <!-- IntelRequired>False</IntelRequired -->

    <!-- Dependencies></Dependencies -->

    <!-- Level>3</Level -->

    <!-- End of uneditable fields -->

    <!-- Please edit the fields below ONLY with the State or ErrAction -->

    <State>Disabled</State>

    <ErrAction>ErrorContinue</ErrAction>

    <RequiredValue format="0x0/00/0x2/02/0x4/04/0x8/08"
example="0x0"> </RequiredValue>

</eolvar>

<eolvar name="eDP Port Config">

    <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

    <!-- Description>Test variable against expected value</Description --
>

    <!-- IntelRequired>False</IntelRequired -->

    <!-- Dependencies></Dependencies -->

    <!-- Level>3</Level -->

    <!-- End of uneditable fields -->

    <!-- Please edit the fields below ONLY with the State or ErrAction -->

    <State>Disabled</State>

    <ErrAction>ErrorContinue</ErrAction>
```



```

        <RequiredValue format="None-None/00/A-None/01/B-None/02/A-B/
03/C-None/04/C-A/05/C-B/06" example="None-None"> </RequiredValue>

    </eolvar>

    <eolvar name="MCTP Device Ports">

        <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

        <!-- Description>Test variable against expected value</Description --
>

        <!-- IntelRequired>False</IntelRequired -->

        <!-- Dependencies></Dependencies -->

        <!-- Level>3</Level -->

        <!-- End of uneditable fields -->

        <!-- Please edit the fields below ONLY with the State or ErrAction -->

        <State>Disabled</State>

        <ErrAction>ErrorContinue</ErrAction>

        <RequiredValue format="Hex number with 0x prefix."
example="0x00000000"> </RequiredValue>

    </eolvar>

    <eolvar name="End of Manufacturing Enable">

        <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

        <!-- Description>Test variable against expected value</Description --
>

        <!-- IntelRequired>False</IntelRequired -->

        <!-- Dependencies></Dependencies -->

        <!-- Level>3</Level -->

        <!-- End of uneditable fields -->

        <!-- Please edit the fields below ONLY with the State or ErrAction -->

        <State>Disabled</State>

        <ErrAction>ErrorContinue</ErrAction>

```



```
<RequiredValue format="No/00/Yes/01" example="No"> </
RequiredValue>

</eolvar>

<eolvar name="Delayed Authentication Mode Config">

    <!-- The commented fields below CANNOT be edited. Any edits will be
    ignored by the tool -->

    <!-- Description>Test variable against expected value</Description --
>

    <!-- IntelRequired>False</IntelRequired -->

    <!-- Dependencies></Dependencies -->

    <!-- Level>3</Level -->

    <!-- End of uneditable fields -->

    <!-- Please edit the fields below ONLY with the State or ErrAction -->

    <State>Disabled</State>

    <ErrAction>ErrorContinue</ErrAction>

    <RequiredValue format="Disabled/00/Enabled/01"
example="Disabled"> </RequiredValue>

</eolvar>

<eolvar name="Firmware Update OEM ID">

    <!-- The commented fields below CANNOT be edited. Any edits will be
    ignored by the tool -->

    <!-- Description>Test variable against expected value</Description --
>

    <!-- IntelRequired>False</IntelRequired -->

    <!-- Dependencies></Dependencies -->

    <!-- Level>3</Level -->

    <!-- End of uneditable fields -->

    <!-- Please edit the fields below ONLY with the State or ErrAction -->

    <State>Disabled</State>

    <ErrAction>ErrorContinue</ErrAction>
```




```

        <RequiredValue format="Hex" example="00000000-0000-0000-0000-
000000000000"> </RequiredValue>

    </eolvar>

    <eolvar name="Debug Override Production Silicon">

        <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

        <!-- Description>Test variable against expected value</Description --
>

        <!-- IntelRequired>False</IntelRequired -->

        <!-- Dependencies></Dependencies -->

        <!-- Level>3</Level -->

        <!-- End of uneditable fields -->

        <!-- Please edit the fields below ONLY with the State or ErrAction -->

        <State>Disabled</State>

        <ErrAction>ErrorContinue</ErrAction>

        <RequiredValue format="Hex number with 0x prefix."
example="0x00000000"> </RequiredValue>

    </eolvar>

    <eolvar name="Debug Override Pre-Production Silicon">

        <!-- The commented fields below CANNOT be edited. Any edits will be
ignored by the tool -->

        <!-- Description>Test variable against expected value</Description --
>

        <!-- IntelRequired>False</IntelRequired -->

        <!-- Dependencies></Dependencies -->

        <!-- Level>3</Level -->

        <!-- End of uneditable fields -->

        <!-- Please edit the fields below ONLY with the State or ErrAction -->

        <State>Disabled</State>

        <ErrAction>ErrorContinue</ErrAction>

```



```
<RequiredValue format="Hex number with 0x prefix."
example="0x00000000"> </RequiredValue>
```

```
</eolvar>
```

```
<!-- END OF EOL VAR TESTS -->
```

```
</memanuf_config>
```

Lines which start with `<!-- -- -->` are comments. They are also used to inform users of the available test group names and the names of specific checks that are included in each test that Intel®ME Manuf recognizes.

To select which test items to run: Modify the State item as `<State> Enabled </State>` to enable the subtest

Wherever there is a section for Required Value, Example: `<RequiredValue format="major_ver.minor_ver" example="0.6"> </RequiredValue>`, Please enter the required values in the xml file which will be used by ME Manuf for testing.

Here is the example that explain how to use this feature:

```
<eolconfig name="PTT FPF">
  <!-- The commented fields bellow CANNOT be edited. Any edits will be
ignored by the tool -->
  <!-- Description>Check ptt against expected value</Description -->
  <!-- IntelRequired>False</IntelRequired -->
  <!-- Dependencies>PlatformTrust</Dependencies -->
  <!-- End of uneditable fields -->
  <!-- Please edit the fields below ONLY with the State or ErrAction -->
  <State>Enabled</State>
  <ErrAction>ErrorContinue</ErrAction>
  <RequiredValue format="Not set/Enabled/Disabled" example="Not
set"> </RequiredValue>
</eolconfig>
```

5.4.3 MEManuf –EOL Variable Check

MEManuf -EOL variable check is designed to check the Intel® CSE settings on the platform before shipping. To minimize the security risk in exposing this in an end-user environment, this test is only available in Intel® CSE manufacturing mode or No EOP Message Sent.

Note: -EOL Variable check. The system must be in Intel® CSE manufacturing mode when -EOL Variable check is run or No EOP Message Sent.

5.4.4 MEManuf –EOL Config Check

MEManuf -EOL Config check is designed to check the Intel®ME-related configuration before shipping. Running Intel-recommended tests before shipping is highly recommended.

**Table 5-3. MEmanuf - EOL Config Tests**

Test	Expected Configuration
Ucode SVN">	Image SVN
USB Port ID	User expected value
UFS Boot Source	User expected value
TXT Supported	Disabled
Secure boot KM SVN	User value
Secure boot KM Anti Rollback	Enabled
Secure boot BSMM SVN	User expected value
Secure boot ACM SVN	User expected value
Safety Island Config	Config is present
SPI Boot Source	Enabled
SOC Config Lock State	Disabled
RPMB Migration Done	Enabled
ROT KM SVN	User value
ROT Anti Rollback	Enabled
RBE Anti Rollback	Enabled
Persistent PRTC Backup Power	Enabled
PMC SVN	User value
PMC Anti Rollback	Disabled
PCH_COSIG	Disabled
OS SVN	User value
OS Anti Rollback	Disabled
Force Boot Guard ACM	Disabled
Key Manifest ID	User expected value
OEM Secure Boot Policy	User value
OEM Platform ID	User expected value
OEM Key Revocation State	Disabled
OEM Key Manifest	Disabled
OEM KM SVN	User expected value



OEM KM Anti Rollback	Disabled
OEM ID	User expected value
NWLD SVN	User expected value
NWLD Anti Rollback	Disabled
Intel® PTT	Disabled
Intel AMT HW Status	Complicated check
IDLM SVN	User expected value
IDLM Anti Rollback	Disabled
Glitch Detection Enabled	Disabled
Glitch Detection Disabled	Disabled
Glitch Detection Cfg Done	Disabled
Error Enforcement Policy 1	Disabled
Error Enforcement Policy 0	Disabled
EMMC Boot Source	Enabled
DNX SVN	User expected value
DNX Anti Rollback	Enabled
CPU Key Manifest SVN	User expected value
CPU FW Anti Rollback	Disabled
CPU Co-signing	User expected value
BSMM Anti Rollback	Disabled
2nd OEM RSA Key size	User expected value
2nd OEM Public Key Hash	User expected value
2nd OEM Key Hash valid	Disabled
1st OEM Public Key Hash	User expected value
Attestation KeyBox test	Enabled
Enforce RPMC Support	User expected version.
PCHC FW version	User expected version.
Boot Guard status	Complicated check – BootGuard files in FWStatus are ok.
FW status	Complicated check – FWStatus is ok.



Checking NVM for fatal flash logs	No errors in log.
Confirm ARB SVN value	Image SVN = Minimum ARB SVN.
PCH Unlocked state	Unlocked.
HW Binding Disabled	True.
SOC Config Lock	1=Enabled.
FPFs in UEP Committed	All UEPs committed and equal to the FPFs' values.
Validate Keybox Provisioning	
Firmware Update OEM ID	User expected value (Format: "00000000-0000-0000-0000-000000000000").
GBE version	User expected version.
BIOS version	User expected version.
CSE FW version	User expected version.
Security Descriptor Override (SDO) check	
RPMC Configuration	Complicated check.
EC Write Access Permissions	Disabled..
EC Read Access Permissions	User value (0x001 for example)User value (0x001 for example).
BIOS Write Access Permissions	User value (0x001 for example).
BIOS Read Access Permissions	User value (0x001 for example).
GBE Write Access Permissions	User value (0x001 for example).
GBE Read Access Permissions	User value (0x001 for example).
ME Write Access Permissions	User value (0x001 for example).
ME Read Access Permissions	User value (0x001 for example).
ME Manufacturing Mode status	Disabled.
EOP status check	Enabled.
Touch – Vendor ID	Expected vendor ID.
ME Manufacturing Mode status	Disabled.
ME Read Access Permissions	User value (0x001 for example).
ME Write Access Permissions	User value (0x001 for example).
GBE Read Access Permissions	User value (0x001 for example).
GBE Write Access Permissions	User value (0x001 for example).
BIOS Read Access Permissions	User value (0x001 for example).



BIOS Write Access Permissions	User value (0x001 for example).
-------------------------------	---------------------------------

Note: -EOL Config check. If the system is in Intel® CSE manufacturing mode when -EOL Config check is run there will be an error report or No EOP Message Sent.

5.4.5 Output/Result

The following test results can be displayed at the end-of-line checking:

- Pass – all tests passed.
- Pass with warning – all tests passed except the tests that were modified by the customer to give a warning on failure. (This modification does not apply to Intel-recommended tests.
- Fail with warning - all tests passed except some Intel-recommended tests that were modified by the customer to give a warning on failure.
- Fail - any customer-defined error occurred in the test.

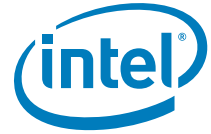
5.5 Examples

```
MEManuf -verbose
Intel (R) MEManuf Version: 15.40.0.1038
Copyright (C) 2005 - 2020, Intel Corporation. All rights reserved.
```

```
LPC Device Id: 4B00.
Platform: ElkhartLake Platform
General FW Information
  FW Status Register1      0x90000255
  FW Status Register2      0x60000516
  FW Status Register3      0x00000020
  FW Status Register4      0x00004000
  FW Status Register5      0x02001F03
  FW Status Register6      0x044003C9

  Current FW State          Normal
  Flash Partition Table     Valid
  FW Memory State           CM0 with UMA
  FW Initialization         Complete
  BUP Loading state         Success
  FW Error Code             No Error
  FW Mode Of Operation      Normal
  SPI Flash Log             Not Present
  FW Loading Phase          HOSTCOMM Module
  FW Loading Phase Status   UNKNOWN
  ME File System Corrupted  No
  RPMC status               OK

Feature enablement is 0x31309240
ME initialization state valid
ME operation mode valid
Current operation state valid
ME error state valid
```



MFS is not corrupted
PCH SKU Emulation is correct

Request Intel(R) ME BIST status command... done

Get Intel(R) ME test data command... done

Get Intel(R) ME test data command... done
Total of 6 Intel(R) ME test result retrieved

Policy Kernel - Boot Guard : Self Test - Passed

VDM - General : VDM engine - Passed

PAVP - General : Verify Edp and Lspcon Configurations - Passed

PAVP - General : Set Lspcon Port - Passed

PAVP - General : Set Edp Port - Passed

Policy Kernel - ME Configuration : PROC_MISSING - Passed

Clear Intel(R) ME test data command... done

MEManuf Operation Passed

IS §



6 Intel®ME Info

MEInfoWin and Intel®ME Info provide a simple test to check whether the Intel® CSE FW is alive. Both tools perform the same test; query the Intel® CSE FW and retrieve data.

Table 18 contains a list of the data that each tool returns.

The Windows* version of ME Info (MEInfoWin) requires administrator privileges to run under Windows* OS. The user needs to use the Run as Administrator option to open the CLI in Windows* 10.

6.1 Windows* PE Requirements

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

ME Info reports an LMS error. This behavior is expected as the LMS driver cannot be installed on Windows* PE.

6.2 Usage

The executable can be invoked by:

```
MEInfoWin64.exe [-EXP] [-H|?] [-VER] [-FITVER] [-FEAT]
                [-VALUE] [-FWSTS] [-VERBOSE] [-PAGE]
```

Following is the output generated when the tool is run with the help option (**-h** or **?**):

```
Intel (R) MEInfo Version: 15.40.0.1039
```

```
Copyright (C) 2005 - 2020, Intel Corporation. All rights reserved.
```

```
MEInfoWin64.exe [-EXP] [-H|?] [-VER] [-FITVER] [-FEAT]
                [-VALUE] [-FWSTS] [-VERBOSE] [-PAGE]
```

```
-EXP [arg_name]      Display example usage of this tool
-H|?                Display help screen
-VER                Display version information
```




-FITVER Display Intel(R) FIT version

-FEAT <name> Retrieve a related platform setting

-VALUE <value> An expected platform setting value

-FWSTS Retrieve/decode ME Firmware status register

-VERBOSE [filename] Display the debug information of the tool

-PAGE Pause after each screenful of information

Note: Name/value more than one word has to be between quotations.

Table 6-1. Intel®ME Info Command Line Options

Option	Description
-FEAT <name> <column> -VALUE <value>	Compares the value of the given feature name (and optional column name for features displayed in a table) with the value in the command line. If the feature name or value is more than one word, the entire name or value must be enclosed in quotation marks (together with the optional column name). For example -feat "PTT FPF". If the values are identical, a message indicating success appears. If the values are not identical, the actual value of the feature is returned. Only one feature may be requested in a command line.
-FITVER	Displays FIT version information
-FEAT <name> <column>	Retrieves the current value for the specified feature (and optional column name for features displayed in a table). If the feature name is more than one word, the entire feature name (and optional column name) must be enclosed in quotation marks. For example -feat "PTT FPF". The feature name entered must be the same as the feature name displayed by Intel®ME Info. Intel®ME Info can retrieve all of the information detailed below. However, depending on the SKU selected, some information may not appear. Note: For the EFI shell version you need to add additional "^" to enclose the text string in order for it to be properly parsed. Example: MEINFO.efi -feat "^"BIOS boot state"^"



Option	Description																										
-FWSTS	<p>Decodes the Intel® CSE FW status register value field and breaks it down into the following bit definitions for easy readability:</p> <p>FW Status Register1: 0x90000255 FW Status Register2: 0x00F10506 FW Status Register3: 0x00000020 FW Status Register4: 0x00004004 FW Status Register5: 0x00000000 FW Status Register6: 0x00400000</p> <table><tr><td>CurrentState:</td><td>Normal</td></tr><tr><td>ManufacturingMode:</td><td>Enabled</td></tr><tr><td>FlashPartition:</td><td>Valid</td></tr><tr><td>OperationalState:</td><td>CM0 with UMA</td></tr><tr><td>InitComplete:</td><td>Complete</td></tr><tr><td>BUPLoadState:</td><td>Success</td></tr><tr><td>ErrorCode:</td><td>No Error</td></tr><tr><td>ModeOfOperation:</td><td>Normal</td></tr><tr><td>SPI Flash Log:</td><td>Present</td></tr><tr><td>Phase:</td><td>ROM/Preboot</td></tr><tr><td>CSE File System Corrupted:</td><td>No</td></tr><tr><td>PhaseStatus:</td><td>PROTECTED_START</td></tr><tr><td>FPF and CSE Config Status:</td><td>Not committed</td></tr></table>	CurrentState:	Normal	ManufacturingMode:	Enabled	FlashPartition:	Valid	OperationalState:	CM0 with UMA	InitComplete:	Complete	BUPLoadState:	Success	ErrorCode:	No Error	ModeOfOperation:	Normal	SPI Flash Log:	Present	Phase:	ROM/Preboot	CSE File System Corrupted:	No	PhaseStatus:	PROTECTED_START	FPF and CSE Config Status:	Not committed
CurrentState:	Normal																										
ManufacturingMode:	Enabled																										
FlashPartition:	Valid																										
OperationalState:	CM0 with UMA																										
InitComplete:	Complete																										
BUPLoadState:	Success																										
ErrorCode:	No Error																										
ModeOfOperation:	Normal																										
SPI Flash Log:	Present																										
Phase:	ROM/Preboot																										
CSE File System Corrupted:	No																										
PhaseStatus:	PROTECTED_START																										
FPF and CSE Config Status:	Not committed																										
-VERBOSE <filename>	<p>Turns on additional information about the operation for debugging purposes. This option has to be used together with the above mentioned option(s). Failure to do so generates the error: "Error 9254: Invalid command line option".</p> <p>This option works with no option and -feat.</p>																										
-H or -?:	<p>Displays the list of command line options supported by the Intel®ME Info tool.</p> <p>Note: Use -H for help when running in the EFI Shell.</p>																										
-VER	Shows the version of the tools.																										
- PAGE	When it takes more than one screen to display all the information, this option lets the user pause the display and then press any key to continue on to the next screen.																										
-EXP	Shows examples about how to use the tools.																										
No option:	If the tool is invoked without parameters, it reports information for all components listed in Table 6-2 below for full SKU FW.																										
-DRVLESS	Driverless mode																										

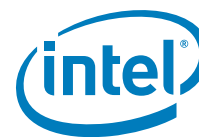
Note: Name/value more than one word has to be between quotations.

**Table 6-2. List of Components that Intel®ME Info Displays**

Feature Name	Feature Data Source (Intel® CSE Kernel/ SW/ Other)	Consumer SKU	Specific Feature Dependency	Field Value
Tools Version	SW (Intel®ME Info)	X	N/A	Version string Example: 14.x.y.ZZZZ; where x=minor, y = HF/MR, ZZZZ = Build Number.
Platform Type	SW (Intel® ME Info)	X	Indicating the type of the platform	Indicating the type of the platform (Mobile, desktop, etc.)
FW Image Type	Intel® CSE Kernel	X	N/A	Indicating the type of the FW image (Pre-Production, Production, etc.)
Last CSE reset reason	Intel® CSE Kernel	X	N/A	Power up/ Firmware reset/ Global system reset/ Unknown
BIOS Boot State	Intel® CSE Kernel	X	N/A	Pre Boot/ In Boot/ Post Boot
Boot Critical Code Redundancy	Intel® CSE Kernel	X	BIOS	Enabled/Disabled
Current Boot Partition	Intel® CSE Kernel	X	n/a	String
CSE Measured Boot to TPM	Intel® CSE Kernel	X	N/A	Enabled/Disabled/Unknown
Capability Licensing Service State	Intel® CSE Kernel	X	Not available on Corporate Sku. Not shown unless Fw feature capability supports it	Enabled/Disabled
Crypto HW Support	Intel® CSE Kernel	X	BIOS	Enabled/Disabled
FW Update State	Intel® CSE Kernel	X	n/a	Enabled/Disabled/ Password Protected



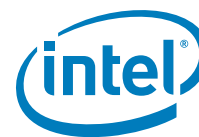
Feature Name	Feature Data Source (Intel® CSE Kernel/ SW/ Other)	Consumer SKU	Specific Feature Dependency	Field Value
Firmware Update OEM ID	Intel® CSE Kernel	X	Only if fw image supports OEM Id	UUID for OEM to check during FW Update
Intel® PTT State	Intel® CSE Kernel	X	FIT PTT is set to 'Enable'	Enabled/Disabled
Intel® PTT initial power-up state	Intel® CSE Kernel	X	N/A	Enabled/Disabled
OEM Tag	Intel® CSE Kernel	X	N/A	A 32bit Hexadecimal number
TLS State	Intel® CSE Kernel	X	N/A	Enabled/Disabled
FW Version	Intel® CSE Kernel	X	N/A	Version string 15.x.y.ZZZZ A B; where x=minor, y = HF/MR, ZZZZ = Build Number, A=LP/H, B=SKU type [Consumer]
FW Type	Intel® CSE Kernel	X	N/A	Pre-Production/Production
LMS Version	Intel® CSE Kernel	X	Only when Windows* LMS driver is installed	A version string
MEI Driver Version	Other (Reading Windows® registry entries)	X	Only when Windows® Intel® MEI driver is installed	A version string
PMC FW Version	Intel® CSE Kernel	X	PMC Region to be present in the image	A version string
OEM FW Version	Intel® CSE Kernel	X	N/A	A version string



Feature Name	Feature Data Source (Intel® CSE Kernel/SW/Other)	Consumer SKU	Specific Feature Dependency	Field Value
PCHC FW Version	Intel® CSE Kernel	X	PMC Region to be present in the image	A version string
PCH Name	Intel® CSE Kernel	X	N/A	String (e.g. EHL)
PCH Device ID	Intel® CSE Kernel	X	N/A	String
PCH Revision ID	Intel® CSE Kernel	X	N/A	PCH Stepping string
PCH SKU Type	Intel® CSE Kernel	X	N/A	String (e.g. Pre-Production ES)
PCH Replacement State	Intel® CSE Kernel	X	Should be enabled in Intel® FIT	Disabled/Enabled
PCH Replaceable Counter	Intel® CSE Kernel	X	PCH Replacement should be enabled	Counter indicating the number that PCH has been replaced
PCH Unlocked State	Intel® CSE Kernel	X	N/A	Enabled/Disabled
Storage Device Type	Intel® CSE Kernel	X	Only when there are flash parts HW installed	SPI/UFS
SPI Flash ID 1	Other (Directly reading from SPI)	X	Only when there are flash parts HW installed	A JEDEC ID number (in Hex)
RPMC	Other (Directly reading from SPI)	X	FIT PTT RPMC Supported feature set to 'Yes'	Supported/Not Supported
RPMC Bind Counter	Intel® CSE Kernel	X	N/A	Counter indicating the number that SPI flash has been rebound
RPMC Bind Status	Intel® CSE Kernel	X	N/A	Pre-Bind/Bound



Feature Name	Feature Data Source (Intel® CSE Kernel/ SW/ Other)	Consumer SKU	Specific Feature Dependency	Field Value
RPMC Rebind	Intel® CSE Kernel	X	FIT PTT RPMC Rebinding Enabled feature set to 'Yes'	Supported/Not Supported
RPMC Replay Protection Max Rebind	Intel® CSE Kernel	X	N/A	Counter indicating the maximum number of rebinds
BIOS Read Access	Other (Directly reading from SPI)	X	N/A	32 bits controlling read access permission of the CPU/BIOS
BIOS Write Access	Other (Directly reading from SPI)	X	N/A	32 bits controlling read write access permission of the CPU/BIOS
ME Read Access	Other (Directly reading from SPI)	X	N/A	32 bits controlling read access permission of CSE
ME Write Access	Other (Directly reading from SPI)	X	N/A	32 bits controlling write access permission of CSE
EC Read Access	Other (Directly reading from SPI)	x	N/A	32 bits controlling read access permission of EC
EC Write Access	Other (Directly reading from SPI)	x	N/A	32 bits controlling read access permission of EC



Feature Name	Feature Data Source (Intel® CSE Kernel/ SW/ Other)	Consumer SKU	Specific Feature Dependency	Field Value
FW Capabilities	Intel® CSE Kernel	x	N/A	Combination of feature name list breakdown (with a Hexadecimal value) *This is a display of the Feature State for the Intel® ME. Is enabled / disabled on the system. Each bit in the value represents a feature state. Intel® ME features including Full manageability, standard manageability, Anti-theft technology etc.
Intel® Protected Audio Video Path	Intel® CSE Kernel	x	PAVP status in the image	Present/Enabled Present/Disabled Not Present/Disabled
Intel® Dynamic Application Loader	Intel® CSE Kernel	x	DAL status in the image	Present/Enabled Present/Disabled Not Present/Disabled
Intel® Platform Trust Technology	Intel® CSE Kernel	x	PTT status in the image	Present/Enabled Present/Disabled Not Present/Disabled
EOM Settings	Intel® CSE Kernel	x	Set in FIT or FPT. Defines the behavior of EOM triggering	Lock(Flash, Config) Lock(Flash, Config) on 1st Boot Lock(Config) Lock(Config) on 1st Boot Lock(Flash) Lock(Flash) on 1st Boot Lock(none) Lock(none) on 1st Boot
NVAR Configuration State	Intel® CSE Kernel	x	Changes after triggering EOM	Unlocked/ Locked
HW Binding State	Intel® CSE Kernel	x	Changes after triggering EOM	Enabled / Disabled
Flash Protection Mode	Intel® CSE Kernel	x	Changes after triggering EOM	Unprotected/Protected



Feature Name	Feature Data Source (Intel® CSE Kernel/ SW/ Other)	Consumer SKU	Specific Feature Dependency	Field Value
FPF Committed	Intel® CSE Kernel	x	Changes after triggering EOM	Yes/No
Widevine provisioning state	Intel® CSE Kernel	n/a	Only shown when AMT is enabled but since there is no AMT it's always Not provisioned	Provisioned/Not Provisioned
Attestation KeyBox	Intel® CSE Kernel	N/A	Only shown when AMT is enabled but since there is no AMT it's always Not provisioned	Provisioned/Not Provisioned
EPID Group ID	Intel® CSE Kernel	n/a	n/a	A 32bit Hexadecimal number
EPID Re-key needed	Intel® CSE Kernel	n/a	N/A	False/True
PAVP State	Intel® CSE Kernel	n/a	N/A	Yes /No
Minimum Allowed Anti Rollback SVN	Intel® CSE Kernel	x	BIOS	Counter indicating the minimum allowed ARB SVN value
Image Anti Rollback SVN	Intel® CSE Kernel	x	BIOS	Counter indicating the ARB SVN existing in the FW Image
Trusted Computing Base SVN	Intel® CSE Kernel	x	BIOS	Counter indicating TCB SVN
1st OEM Key Hash size	Intel® CSE Kernel	x	BIOS	Enabled/ Disabled ,Hash of Public Key to verify Boot Policy Manifest



Feature Name	Feature Data Source (Intel® CSE Kernel/ SW/ Other)	Consumer SKU	Specific Feature Dependency	Field Value
1st OEM Key hash valid	Intel® CSE Kernel	x	BIOS	Enabled/ Disabled
1st OEM RSA Key size	Intel® CSE Kernel	x	BIOS	Enabled/ Disabled
2nd OEM Key Hash size	Intel® CSE Kernel	x	BIOS	Enabled/ Disabled
2nd OEM Key Hash valid	Intel® CSE Kernel	x	BIOS	Enabled/ Disabled
2nd OEM RSA Key size	Intel® CSE Kernel	x	BIOS	Enabled/ Disabled
BSMM Anti Rollback	Intel® CSE Kernel	x	BIOS	Enabled/ Disabled
CPU Co-signing	Intel® CSE Kernel	x	BIOS	Enabled/ Disabled
CPU FW Anti Rollback	Intel® CSE Kernel	x	BIOS	Enabled/ Disabled
CPU Key Manifest SVN	Intel® CSE Kernel	x	BIOS	Hexadecimal Number
DNX Anti Rollback	Intel® CSE Kernel	x	BIOS	Enabled/ Disabled
DNX SVN	Intel® CSE Kernel	X	BIOS	Hash of Public Key to verify Boot Policy Manifest
Error Enforcement Policy 0	Intel® CSE Kernel	X	BIOS	Enabled/ Disabled
Error Enforcement Policy 1	Intel® CSE Kernel	X	BIOS	Enabled/ Disabled
Glitch Detection Disabled	Intel® CSE Kernel	x	BIOS	Enabled/ Disabled/ Unknown



Feature Name	Feature Data Source (Intel® CSE Kernel/ SW/ Other)	Consumer SKU	Specific Feature Dependency	Field Value
Glitch Detection Enabled	Intel® CSE Kernel	x	BIOS	Enabled/ Disabled/ Unknown
IDLM Anti Rollback	Intel® CSE Kernel	x	BIOS	Enabled/ Disabled
IDLM SVN	Intel® CSE Kernel	x	BIOS	Hash of Public Key to verify Boot Policy Manifest
Intel PTT Anti Hammering	Intel® CSE Kernel	x	BIOS	Hash of Public Key to verify Boot Policy Manifest
Intel PTT Encryption Key	Intel® CSE Kernel	x	BIOS	Enabled/ Disabled
Intel® PTT	Intel® CSE Kernel	x	BIOS	Enabled / Disabled
NWLD Anti Rollback	Intel® CSE Kernel	x	BIOS	Enabled / Disabled
NWLD SVN	Intel® CSE Kernel	x	BIOS	A 32bit Hexadecimal number
OEM ID	Intel® CSE Kernel	x	BIOS	Hash of Public Key to verify Boot Policy Manifest
OEM KM Anti Rollback	Intel® CSE Kernel	x	BIOS	Enabled / Disabled
OEM KM SVN	Intel® CSE Kernel	x	BIOS	Hash of Public Key to verify Boot Policy Manifest
OEM Key Hash 0 Revocation	Intel® CSE Kernel	x	BIOS	Enabled / Disabled
OEM Key Hash 1 Revocation	Intel® CSE Kernel	x	BIOS	Enabled / Disabled
OEM Key Hash 2 Revocation	Intel® CSE Kernel	x	BIOS	Enabled / Disabled



Feature Name	Feature Data Source (Intel® CSE Kernel/ SW/ Other)	Consumer SKU	Specific Feature Dependency	Field Value
OEM Key Manifest	Intel® CSE Kernel	x	BIOS	Enabled / Disabled
OEM Key Revocation State	Intel® CSE Kernel	x	BIOS	Enabled / Disabled
OEM Platform ID	Intel® CSE Kernel	x	BIOS	Hash of Public Key to verify Boot Policy Manifest
OEM Secure Boot Policy	Intel® CSE Kernel	x	BIOS	Hash of Public Key to verify Boot Policy Manifest
CPU Debugging	Intel® CSE Kernel	x	BIOS	Enabled / Disabled
BSP Initialization	Intel® CSE Kernel	x	BIOS	Enabled /Disabled
Protect BIOS Environment	Intel® CSE Kernel	x	BIOS	Enabled /Disabled
Measured Boot	Intel® CSE Kernel	x	BIOS	Enabled /Disabled
Verified Boot	Intel® CSE Kernel	x	BIOS	Enabled /Disabled
Key Manifest ID	Intel® CSE Kernel	x	BIOS	Hash of Public Key to verify Boot Policy Manifest
Force Boot Guard ACM	Intel® CSE Kernel	x	BIOS	Enabled /Disabled
OEM key Hash RSA key size	Intel® CSE Kernel	x	BIOS	Enabled /Disabled
OEM key Hash size	Intel® CSE Kernel	x	BIOS	Enabled /Disabled
OS Anti Rollback	Intel® CSE Kernel	x	BIOS	Enabled /Disabled
PCH_COSIG	Intel® CSE Kernel	X	BIOS	Enabled /Disabled



Feature Name	Feature Data Source (Intel® CSE Kernel/ SW/ Other)	Consumer SKU	Specific Feature Dependency	Field Value
PMC Anti Rollback	Intel® CSE Kernel	X	BIOS	Enabled /Disabled
PMC SVN	Intel® CSE Kernel	X	BIOS	Hash of Public Key to verify Boot Policy Manifest
Persistent PRTC Backup Power	Intel® CSE Kernel	X	BIOS	Enabled /Disabled
RBE Anti Rollback	Intel® CSE Kernel	X	BIOS	Enabled /Disabled
ROT Anti Rollback	Intel® CSE Kernel	X	BIOS	Enabled /Disabled
ROT KM SVN	Intel® CSE Kernel	X	BIOS	Hash of Public Key to verify Boot Policy Manifest
RPMB Migration Done	Intel® CSE Kernel	X	BIOS	Enabled /Disabled
RPMB Monotonic Counters	Intel® CSE Kernel	X	BIOS	Hash of Public Key to verify Boot Policy Manifest
SOC Config Lock State	Intel® CSE Kernel	X	BIOS	Enabled /Disabled
SPI Boot Source	Intel® CSE Kernel	X	BIOS	Enabled /Disabled
Safety Island Config	Intel® CSE Kernel	X	BIOS	Confi Log is present / Config Log is not present
Secure boot ACM SVN	Intel® CSE Kernel	X	BIOS	Hash of Public Key to verify Boot Policy Manifest
Secure boot BSMM SVN	Intel® CSE Kernel	X	BIOS	Hash of Public Key to verify Boot Policy Manifest
Secure boot KM Anti Rollback	Intel® CSE Kernel	X	BIOS	Enabled /Disabled



Feature Name	Feature Data Source (Intel® CSE Kernel/ SW/ Other)	Consumer SKU	Specific Feature Dependency	Field Value
Secure boot KM SVN	Intel® CSE Kernel	X	BIOS	Hash of Public Key to verify Boot Policy Manifest
TXT Supported	Intel® CSE Kernel	X	BIOS	Enabled /Disabled
UFS Boot Source	Intel® CSE Kernel	X	BIOS	Enabled /Disabled
USB Port ID	Intel® CSE Kernel	X	BIOS	Hash of Public Key to verify Boot Policy Manifest
Ucode SVN	Intel® CSE Kernel	X	BIOS	Enabled /Disabled
1st OEM Public Key Hash FPF	Intel® ME Kernel	X	BIOS	Set /Not Set
1st OEM Public Key Hash UEP	Intel® ME Kernel	X	BIOS	SHA-256bit Hash entry
2nd OEM Public Key Hash FPF	Intel® ME Kernel	X	BIOS	Set /Not Set
2nd OEM Public Key Hash UEP	Intel® ME Kernel	X	BIOS	SHA-256bit Hash entry
OS SVN	Intel® CSE Kernel	x	BIOS	Hash of Public Key to verify Boot Policy Manifest

6.3 Examples

This is a simple test that indicates whether the FW is alive. If the FW is alive, the test returns device-specific parameters. The output is from the Windows* version.

Note: **If EOM is set, for FPF's the FPF and CSE column values both will be displayed**

Intel (R) MEInfo Version: 15.40.0.1039

Copyright (C) 2005 - 2020, Intel Corporation. All rights reserved.



General FW Information

FW Status Register1	0x90000255
FW Status Register2	0x62000506
FW Status Register3	0x00000020
FW Status Register4	0x00004000
FW Status Register5	0x02001F03
FW Status Register6	0x044003C9

Current FW State	Normal
Flash Partition Table	Valid
FW Memory State	CM0 with UMA
FW Initialization	Complete
BUP Loading state	Success
FW Error Code	No Error
FW Mode Of Operation	Normal
SPI Flash Log	Not Present
FW Loading Phase	HOSTCOMM Module
FW Loading Phase Status	UNKNOWN
ME File System Corrupted	No
RPMC status	OK

Platform Type	Mobile
FW Image Type	Pre-Production
Last ME Reset Reason	Global system reset
BIOS Boot State	Post Boot
Boot Critical Code Redundancy	Disabled
Current Boot Partition	1
Firmware Update OEM ID	00000000-0000-0000-0000-000000000000
Crypto HW Support	Enabled
Intel(R) ISH Power State	Disabled
OEM Tag	0x73c8176
FW Update State	Enabled
Capability Licensing Service State	Enabled
TLS State	Enabled
CSME Measured Boot to TPM	Disabled

Intel(R) ME Code Versions

BIOS Version	EHLFWI1.R00.2115.A00.2003130949
MEI Driver Version	2016.15.40.1190
FW Version	15.40.0.1039 LP Consumer
LMS Version	Not Installed

IUPs Information

PMC FW Version	154.1.1.1014
OEM FW Version	15.40.0.7036
PCHC FW Version	15.0.0.7006
GBST FW Version	0.0.0.0000

PCH Information

PCH Name	EHL
PCH Device ID	4B00
PCH Revision ID	A0
PCH SKU Type	Pre-Production ES
PCH Replacement State	Disabled



PCH Replaceable Counter	0
PCH Unlocked State	Disabled
Flash Information	
Storage Device Type	SPI
SPI Flash ID 1	EF4019
RPMC	Unsupported
RPMC Bind Counter	0
RPMC Bind Status	Pre-bind
RPMC Rebind	Unsupported
RPMC Replay Protection Max Rebind	1
BIOS Read Access	0xFFFF
BIOS Write Access	0xFFFF
GBE Read Access	0xFFFF
GBE Write Access	0xFFFF
ME Read Access	0xFFFF
ME Write Access	0xFFFF
EC Read Access	0xFFFF
EC Write Access	0xFFFF
FW Capabilities	
Intel(R) Protected Audio Video Path	0x31309240
Intel(R) Dynamic Application Loader	Present/Enabled
Intel(R) Platform Trust Technology	Present/Enabled
Persistent RTC and Memory	Present/Enabled
End Of Manufacturing	
NVAR Configuration State	Unlocked
EOM Settings	Lock(Flash,Config)
HW Binding State	Disabled
Flash Protection Mode	Unprotected
FPF Committed	No
Intel(R) Protected Audio Video Path	
Widevine provisioning state	Not Provisioned
Attestation KeyBox	Not Provisioned
EPID Group ID	0x4DC
EPID Re-key needed	False
PAVP State	Yes
Security Version Numbers	
Trusted Computing Base SVN	1
Anti Rollback SVNs	
PMC	1 [minimum allowed: 1]
CSE	1 [minimum allowed: 1]
ROT KM	1 [minimum allowed: 1]
SECURE BOOT BSMM	1 [minimum allowed: 1]
SECURE BOOT KM	1 [minimum allowed: 1]
SECURE BOOT ACM	1 [minimum allowed: 1]
HW Glitch Detection	
TRC Polarity	0x08
TRC Mode	Rising Trans
TRC State	Full-cycle polarity trans
	Disabled



Intel(R) Platform Trust Technology
 Intel(R) PTT initial power-up state
 Intel(R) PTT State
 SMx State

Enabled
 Enabled
 Enabled

FW Supported FPFs

FPF UEP
 *In Use

---	---	
1st OEM Key Hash size Disabled=0, Enabled=1	Not set	Enabled #
1st OEM Key hash valid Disabled=0, Enabled=1	Not set	Enabled #
1st OEM RSA Key size Disabled=0, Enabled=1	Not set	Enabled #
2nd OEM Key Hash size Disabled=0, Enabled=1	Not set	Enabled #
2nd OEM Key Hash valid Disabled=0, Enabled=1	Not set	Enabled #
2nd OEM RSA Key size Disabled=0, Enabled=1	Not set	Disabled #
BSMM Anti Rollback Disabled=0, Enabled=1	Not set	Enabled #
CPU Co-signing Disabled=0, Enabled=1	Not set	Disabled #
DAL OEM Signing Disabled=0, Enabled=1	Not set	Disabled #
DNX Anti Rollback Disabled=0, Enabled=1	Not set	Enabled #
Error Enforcement Policy 0 Disabled=0, Enabled=1	Not set	Enabled #
Error Enforcement Policy 1 Disabled=0, Enabled=1	Not set	Enabled #
Flash Descriptor Verification Disabled=0, Enabled=1	Not set	Disabled #
Glitch Detection Disabled Enabled=0, Disabled=1	Not set	Enabled #
Glitch Detection Enabled Disabled=0, Enabled=1	Not set	Disabled #
IDLM Anti Rollback Disabled=0, Enabled=1	Not set	Enabled #
Intel PTT Anti Hammering	Not set	0x00
Intel PTT Encryption Key Revoked=0, Revoked=1	Not set	Not Revoked # Not
Intel(R) AMT HW Status Enabled=0, Disabled=1	Not set	Enabled #
Intel(R) PTT Disabled=0, Enabled=1	Not set	Enabled #
OEM ID	Not set	0x00
OEM KM Anti Rollback Disabled=0, Enabled=1	Not set	Enabled #
OEM Key Hash 0 Revocation Disabled=0, Enabled=1	Not set	Disabled #
OEM Key Hash 1 Revocation Disabled=0, Enabled=1	Not set	Disabled #



OEM Key Hash 2 Revocation Disabled=0, Enabled=1	Not set	Enabled	#
OEM Key Manifest Disabled=0, Enabled=1	Not set	Enabled	#
OEM Key Revocation State Disabled=0, Enabled=1	Not set	Disabled	#
OEM Platform ID	Not set	0x00	
OEM Secure Boot Policy	Not set	0x79	
CPU Debugging Enabled=0, Disabled=1	Not set	Enabled	#
BSP Initialization Enabled=0, Disabled=1	Not set	Enabled	#
Protect BIOS Environment Disabled=0, Enabled=1	Not set	Enabled	#
Measured Boot Disabled=0, Enabled=1	Not set	Enabled	#
Verified Boot Disabled=0, Enabled=1	Not set	Enabled	#
Key Manifest ID	Not set	0x01	
Force Boot Guard ACM Disabled=0, Enabled=1	Not set	Enabled	#
OEM key Hash RSA key size Disabled=0, Enabled=1	Not set	Enabled	#
OEM key Hash size Disabled=0, Enabled=1	Not set	Enabled	#
PCH_COSIG Disabled=0, Enabled=1	Not set	Disabled	#
PMC Anti Rollback Disabled=0, Enabled=1	Not set	Enabled	#
Persistent PRTC Backup Power Enabled=0, Disabled=1	Not set	Enabled	#
RBE Anti Rollback Disabled=0, Enabled=1	Not set	Enabled	#
ROT Anti Rollback Disabled=0, Enabled=1	Not set	Enabled	#
RPMB Migration Done Disabled=0, Enabled=1	Not set	Disabled	#
RPMB Monotonic Counters	Not set	0x00	
SOC Config Lock State Disabled=0, Enabled=1	Not set	Disabled	#
SPI Boot Source Enabled=0, Disabled=1	Not set	Enabled	#
Safety Island Config # config is signed=0, config is not signed=1, config is not present=2, isi is not present=3	Not set	config is not present	
Secure boot KM Anti Rollback Disabled=0, Enabled=1	Not set	Enabled	#
TXT Supported Disabled=0, Enabled=1	Not set	Disabled	#
UFS Boot Source Enabled=0, Disabled=1	Not set	Disabled	#
USB Port ID	Not set	0x00	
DNX SVN	Not set	0x00	
IDLM SVN	Not set	0x00	



OEM KM SVN	Not set	0x00
PMC SVN	Not set	0x00
ROT KM SVN	Not set	0x00
Secure boot ACM SVN	Not set	0x00
Secure boot BSMM SVN	Not set	0x00
Secure boot KM SVN	Not set	0x00

1st OEM Public Key Hash FPF	Not set
1st OEM Public Key Hash UEP	
F8F0E369158176990A549ED4C36D1A8639D8873DEFF7ED2DE34CB41BCCB30476CE0AA063BC5B7AACFFD	
9509E9640C699	
2nd OEM Public Key Hash FPF	Not set
2nd OEM Public Key Hash UEP	
00	
00000000000000	

6.3.1 ME Info Sample Output

6.3.2 Retrieve Current Value of Flash Version

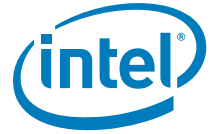
```
C:\ MEINFO.exe -feat "BIOS boot state"
Intel® MEINFO Version: XX.XX.XX.xxxx
Copyright(C) 2005 - 2019, Intel Corporation. All rights reserved.
```

BIOS boot State: Post Boot

```
> MEINFO.efi -feat “^"BIOS boot state"^”
Intel® MEINFO Version: XX.XX.XX.xxxx
Copyright(C) 2005 - 2017, Intel Corporation. All rights reserved.
```

BIOS boot State: Post Boot

§ §



7 Intel® CSE Firmware Update

FW Update allows an end user, such as an IT administrator, to update Intel® CSE FW without having to reprogram the entire flash device. It then verifies that the update was successful.

FW Update does not update the BIOS or Descriptor Regions. It updates the FW code portion along with the WCOD, LOCL and IUNP . Intel® FW Update updates the entire Intel® CSE code area. In addition FW Update local can perform a partial update to change / update the WCOD, LOCL and IUNP .

The image file that the FW Update tool uses is one of the image files that are generated by the FIT tool. Two images are created automatically by the FIT tool, *_base*.bin and *_full*.bin.

- The *_base*.bin file contains the CSE firmware stitched together with the PMC binary only.
- The *_full*.bin file contains the CSE firmware stitched together with the PMC binary as well as any IUPs and the OEM Key Manifest (when provided).
- It is important to note that WCOD & LOCL are part of Intel® CSME and therefore included in the *_base*.bin file.

FW Update takes approximately 1-4 minutes to complete depending on the flash device on the system.

After FW Update a host reset is needed to complete FW update. The user can also use the `-FORCERESET` option to do this automatically.

Note: In previous generations there were two tools: Intel® CSE Local Firmware Update and Intel® CSE Remote Firmware Update. Now there is just a local firmware update tool that is called Intel® CSE Firmware Update (FW Update).

7.1 Requirements

FWUpdLcl.exe is a command line executable that can be run on an Intel®ME-enabled system that needs updated FW.

FW can only be updated when the system is in an S0 state. FW updates are NOT supported in the S3/S4/S5 state.

Intel® CSE FW Update must be enabled through BIOS.

The Intel® CSE Interface driver must be installed for running this tool in a Windows* environment.

Note: FWUpdLcl.exe must be run with Administrator privilege for access to the Intel®MEI driver



7.2 Enabling and Disabling Intel® FW Update

In BIOS, there is an option to enable/disable local firmware update.

This option supports three value, enabled, disabled and Password protected.

Disabled – does not allow FW to be updated

Enabled – allows FW to be updated

7.3 FW Update Flows

7.3.1 Full FW Update

This will help allow to update Intel® CSE Firmware. If IUP's are present in the payload image along with Intel® CSE Firmware, IUP's will also be updated along with Intel® CSE as part of the Full FW Update.

Global Reset will be required to complete the FW Update operation.

PMC Firmware Update: This will be handled as part of the Full FW Update flow and cannot be updated on its own. PMC Firmware needs to be stitched with Intel® CSE Firmware using Intel® FIT Tool and that image will be used as the payload to Full FW Update Flow for updating PMC Firmware.

Intel® CSE Firmware Update: This will be handled as part of the Full FW Update Flow. Requirement: Only CSE Image won't be allowed as the payload to execute update. Pre-Stitched CSE + PMC binary needs to be used as the payload to execute CSE update.

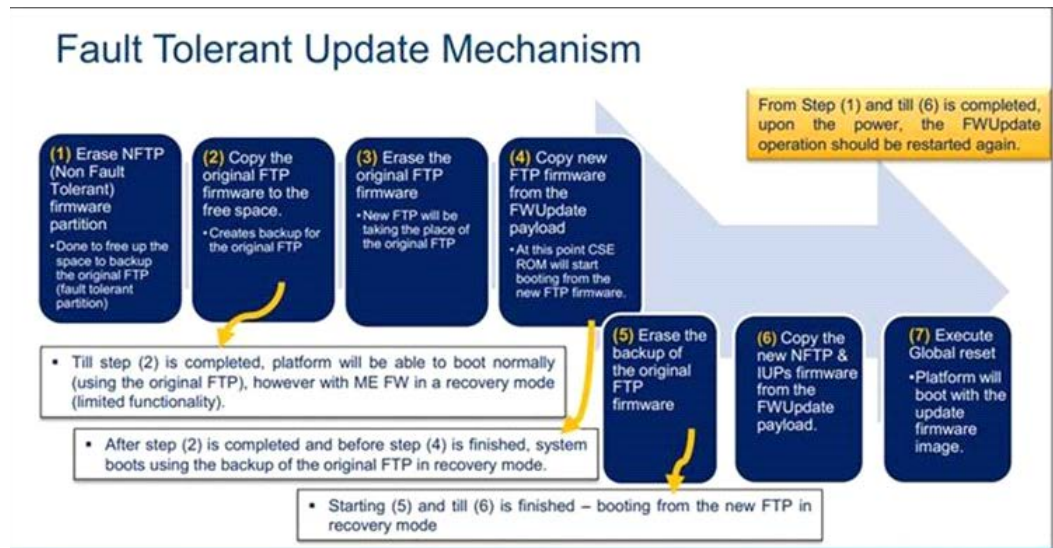
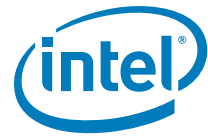
The Update Flow explained: A full update flow is a fault tolerant one and consists of the following steps:

7.3.2 Partial FW Update

This will help allow to update IUP's (Independent Updatable Partitions) only i.e. WLAN micro-code, Localization, IUnit Loader etc.

For optional IUP's Firmware Update only. No stitching with Intel® CSE Firmware required.

Note: As an IUP, Dekel PHY can be updated using partial FW Update without resorting to full update.



7.4 Usage

Note: In this section, <Image File> refers to an Intel-provided image file of the section of the FW to be updated, not the image file used in FIT to program the entire flash memory.

```
FWUpLcl64.exe [-H|?] [-VER] [-EXP] [-VERBOSE] [-F] [-Y]
               [-ALLOWSV] [-SILENT] [-FWVER] [-SAVE] [-PARTID] [-INSTID]
               [-FORCERESET] [-OEMID] [-PARTVER] [-PARTVENDOR]
```

Following is the output generated when the tool is run with the help option (-h or ?)

```
Intel (R) Firmware Update Utility Version: 15.40.0.1039
Copyright (C) 2005 - 2020, Intel Corporation. All rights reserved.
```

```
FWUpLcl64.exe [-H|?] [-VER] [-EXP] [-VERBOSE] [-F] [-Y]
               [-ALLOWSV] [-SILENT] [-FWVER] [-SAVE] [-PARTID] [-INSTID]
               [-FORCERESET] [-OEMID] [-PARTVER] [-PARTVENDOR]
```

```
-H|?           Display help screen
-VER           Displays version information.
```



-EXP	Displays example usage of this tool.
-VERBOSE <file>	Display the debug information of the tool.
-F <file>	File used for updating the FW.
-Y	Automatically answer Yes to prompts.
-ALLOWSV	Allows same version firmware updates.
-SILENT prompts.	Update without display and without user prompts.
-FWVER <file>	Display the FW Version of current FW or update image.
-SAVE <file>	Save the current FW to an update image.
-PARTID <Partition ID>	Provide specific Partition ID to perform partial update.
-INSTID <Instance ID>	Provide specific Instance ID of a partition to perform partial update.
-FORCERESET	Automatically Reboots system after update (if needed) .
-OEMID <UUID>	OEM ID needed to perform firmware update.
-PARTVER <Partition ID>	Display the Version of specific partition.
-PARTVENDOR <Partition ID>	Display the Vendor ID of specific partition.

Table 7-1. Image File Update Options

Option	Description
-VERBOSE [<FILE>]	Verbose. Enables additional information about the tool's operation to be displayed for debugging purposes.
-Y	Ignore warning. If the warning asks for input "Y/N", this flag makes the tool automatically take "y" as the input.
-F <FILE>	File. Specifies the FW Update image file to be used for performing an update.
-SAVE <file>	Restore Point. Retrieves an update image from the FW based on the currently running FW. The update image is saved to the user-specified file.
-ALLOWSV	Allow Same Version. Allows the version of the input FW (based on the file input) to be the same as the version of the FW currently on the platform. Without this option, an attempt to perform an update on the same version will not proceed.



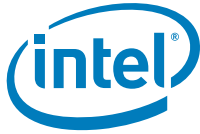
Option	Description
-FORCERESET	Force Reset. The tool automatically reboots the system after the update process with FW is complete. The system reboot is necessary for the new FW to take effect. An attempt to update the FW without this option will end with a message telling the user to reset the platform for the changes to take effect.
-OEMID <UUID>	OEM ID. The tool uses the specified OEM ID during the transaction of the new FW image with the Manageability Engine. The purpose of the OEM ID is for manufacturers to have an identifier for their system. Using any other OEM ID value other than what is on the FW running on the target platform results in a failure of the FW Update process. The full image (including all necessary flash partitions) flashed to the system can be configured with the Flash Image Tool to specify the OEM ID (this tool specifies a default of zeros for the OEM ID.) If this command line option is not used, the default OEM ID used for the update is zeros. The OEM ID is configured in the existing FW image running on the platform. The OEM ID value is specified in the UUID format (8-4-4-4-12).
-PARTID <wcod, locl, ishc, iunp>	<p>This option is always used along with the -F option.</p> <p>The partition ID is requested using the "partid" option, which must be one of the following strings: wcod, locl, iunp or ishc. If the requested partition is expected by the Firmware the tool will search for the expected partition in the image provided, extract it and send it to the FW to perform the update. If the expected partition is not found in the image an invalid file error will be returned by the tool. Also, if the requested partition is not expected by the firmware an error will be returned to the user.</p> <p>Note: For partial fw update the image provided must either be a Full or Partial image. A full image starts with a FPT and contains FTP and NFTP partitions. A partial image starts with either WCOD or LOCL partitions.</p>
-FWVER	Display FW version
-H or -?	Displays the list of command line options supported by the Intel®ME Info tool. Note: Use -H for help when running in the EFI Shell.
-EXP	Shows examples about how to use the tools.
-INSTID <Instance ID>	provide specific instance id of a partition to perform partial update.
-VER	Shows the version of the tools.
-PARTVER	Display flashed ISH FW Version
-silent	Update without display and without user prompts
-Partvendor <Partition ID>	display the vendor ID of the specific region
-pass <pass>	MEBx password. Optional with the -f option

7.5 Examples

7.5.1 Updates Intel® CSE with Firmware Binary File

Note: In order to execute FWUpdLcl in EFI, make sure all the payload files and FW Update executable are located in the root folder.

This command updates Intel® CSE with FW.BIN file. If the firmware on current



platform is newer than then version in FW.BIN file, tools will promote a warning to let user know there will be a firmware downgrade (rollback) event and let user choose Y/N to continue. User can always use -y to skip this warning automatically. If the firmware on the platform is the same as the version in FW.BIN, tools will return an error. User can use -allowsv to allow same version update.

```
FWUpdLcl.exe -f FW.BIN
```

```
EFI:
```

```
FWUpdLcl.efi -f FW.BIN
```

7.5.2 Partial Firmware Update

This command will perform a partial update of the FW via Intel®MEI for either the wcod, locl and iunp..

```
FWUpdLcl.exe -f FW.bin -partid <wcod, locl, iunp >
```

```
EFI:
```

```
FWUpdLcl.efi -f upd.bin -partid <wcod, locl, iunp >
```

Non-Verbos Mode

```
C:\> FWUpdLcl.exe -f FW.BIN.bin -partid WCOD
```

```
Intel ® Firmware Update Utility version xx.xx.xx.xxxx  
Copyright (C) 2007-2019, Intel Corporation. All rights reserved.
```

```
Communication Mode: MEI
```

```
Sending the update image to FW for verification: [ COMPLETE ]
```

```
FW Update: [ 100% (Stage: 31 of 19) (|)]
```

```
FW Update is completed successfully.
```

Verbose Mode

```
C:\> FWUpdLcl.exe -f FW.BIN.bin -partid WCOD -verbose
```

```
Intel ® Firmware Update Utility version xx.xx.xx.xxxx  
Copyright (C) 2007-2019, Intel Corporation. All rights reserved.
```

```
Communication Mode: MEI
```

```
Sending the update image to FW for verification: [ COMPLETE ]
```

```
Firmware last update status = Firmware update success
```

```
Firmware last update reset type = 2
```

```
FW Update is completed successfully.
```

7.5.3 Display Supported Commands

Display a list of supported command line sequences based on the arguments provided.



The arguments relevant for this usage are any of the command line options with the prefix '-' removed. The tool will display all valid command sequences based on the options provided. Below is an example which displays valid command sequences with the -ipu option

```
C:\> FWUpdLcl.exe -exp partid
```

```
Intel ® Firmware Update Utility version xx.xx.xx.xxxx
Copyright (C) 2007-2019, Intel Corporation. All rights reserved.
```

The parameters provided are supported in the following command-line sequences:

1. F<file> PARTID[<Partition ID>] [FORCERESET] [VERBOSE[<file>]] [Y] [PASS<pass>]
2. F<file> PARTID[<Partition ID>] INSTID[<Instance ID>] [FORCERESET] [VERBOSE[<file>]] [Y] [PASS<pass>]

Using -EXP without any additional input will display examples of common command-line input.

```
EFI:
> FWUpdLcl.efi -exp partid
```

```
Intel ® Firmware Update Utility version xx.xx.xx.xxxx
Copyright (C) 2007-2017, Intel Corporation. All rights reserved.
```

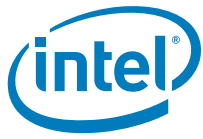
The parameters provided are supported in the following command-line sequences:

1. F<file> PARTID[<Partition ID>] [FORCERESET] [VERBOSE[<file>]] [Y] [PASS<pass>]
2. F<file> PARTID[<Partition ID>] INSTID[<Instance ID>] [FORCERESET] [VERBOSE[<file>]] [Y] [PASS<pass>]

Using -EXP without any additional input will display examples of common command-line input.

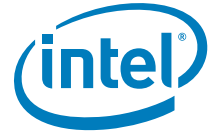
7.5.4 Language Codes

Language	Language Code
English	0x01
French	0x02
German	0x03
Chinese Traditional	0x04
Japanese	0x05
Russian	0x06
Italian	0x07
Spanish	0x08
Brazilian Portuguese	0x09
Korean	0x0A



Chinese Simplified	0x0B
Arabic	0x0C
Czech	0x0D
Danish	0x0E
Greek	0x0F
Finnish	0x10
Hebrew	0x11
Hungarian	0x12
Dutch	0x13
Norwegian	0x14
Polish	0x15
Portuguese-Portugal	0x16
Slovak	0x17
Slovenian	0x18
Swedish	0x19
Thai	0x1A
Turkish	0x1B

§ §



8 UEFI Sample Application Leveraging FW Update API Library

8.1 Getting Started - FW Update Library

8.1.1 Introduction

This chapter will describe the Firmware Update Libraries that will be used for Intel® CSE FW update. It contains a description of the various APIs to be used.

The Firmware Update process es essential for updating WCOD and LOCL regions by utilizing the APIs provided in the Firmware Update Library.

8.1.2 Environment

The FW Update Library provided is compiled using the EFI toolkit V2.0 and MSDK.

8.1.3 Setup

Follow the setting of the references below to get started with using the Firmware Update (FW Update) library and compiling it correctly.

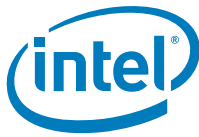
1. You will need to include/reference the "FWUpdateLib.h" file in your program.
2. A make file referencing the FW Update Library. Libraries to Reference:

```
LIBS = $(LIBS) \  
$(SDK_BUILD_DIR)\lib\libc\libc.lib \  
$(SDK_BUILD_DIR)\lib\libefi\libefi.lib \  
$(SDK_BUILD_DIR)\lib\libsmbios\libsmbios.lib \  
$(SDK_BUILD_DIR)\lib\libefishell\libefishell.lib \  
$(SDK_BUILD_DIR)\lib\FwUpdateEfiLib\FwUpdateEfiLib.lib
```

8.1.4 Sample App

The sample code provides you with an example of how to integrate the UEFI FW Update lib into your BIOS or UEFI application. Error handling, command line processing and loading the update image into memory is left to the customers.

Example – Developing FW Update Sample App



**Note: Please Refer to the Actual Sample App Source Code under EFI/
SampleSource/ Provided in the Kit for Proper Details.**

/*++

Copyright (c) 2014-2019 Intel Corporation

Module Name:

FwUpdLcl.c

Abstract:

Sample application demonstrating the usage of the FWU Client UEFI interface

Revision History

--*/

```
#include "efi.h"
#include "efilib.h"
#include "Fwu_Common.h"
#include "Common.h"
#include "me_status.h"
#include "FWUpdateLib.h"
#include "cse_basic_types.h"
#include "typedef.h"
```

// This function handles the callback from the FWU library for displaying

// the percentage of completeness of the FW update

void DisplaySendStatus(float BytesSent, float BytestobeSent)

```
{
    float Value = BytesSent/BytestobeSent * 100;

    UINT32 pValue = (UINT32)Value;

    if (pValue != 100)
    {
        Print (L"Sending the update image to FW for verification: [ %d%% ]\r",pValue);
    }else
    {
        Print (L"Sending the update image to FW for verification: [ COMPLETE ] \n");
    }
}
```

// This is the main entry point for the FW Update application.

// It handles the initialization of the required libraries and

// interfaces to the FW Update Library.

EFI_DRIVER_ENTRY_POINT (InitializeFwUpdLclApplication)

EFI_STATUS

InitializeFwUpdLclApplication (

IN EFI_HANDLE ImageHandle,

IN EFI_SYSTEM_TABLE *SystemTable



```

    )
{
    EFI_STATUS      Status;
    CHAR16          ImageName[256];
    UINTN           ImageLength = 0;
    UINT8           *ImageBuffer = NULL;
    BOOLEAN         bAllowSV;
    BOOLEAN         bUsePassword;
    BOOLEAN         bPid;
    BOOLEAN         bF;
    BOOLEAN         bPdt;
    BOOLEAN         bIshVer;
    //CHAR          Password[9];
    char            *Password = NULL;
    UINT32          FWUpdateStatus;
    DWORD           loops = 500;
    BOOLEAN         done = FALSE;
    UINT32          lastStatus = 0;
    UINT32          platCheck = 0;
    FWVersion       fwVersion;
    INT32           platCheckReturn = 0;
    UINT32          CheckPolicyStatus = 0;
    UPDATE_TYPE     Upd_Type;
    VersionLib      ver;
    UINT32          index = 0;
    UINT32          status;
    UINT32          UpdateStatus = 0;
    UINT32          TotalStages = 0;
    UINT32          PercentWritten = 0;
    CHAR8           symbol;
    UINT32          lastResetType;
    UPDATE_FLAGS_LIB update_flags;
    UINT16          interfaces;
    int             timer30s = 0;
    unsigned int    indexMod;
    int             percentage0s = 0;
    int             percentdiff = 0;
    UINT32          ComparePartID = 0;
    UINT32          hexValueInstId = 0;
    IPU_UPDATED_INFO IpuUpdatedInfo;
    UINT32          PartId = 0;
    UINT32          fwuError;
    FWU_GET_IPU_PT_ATTRB_MSG_REPLY FwuGetIpuAttrbMsgInfo;
    bool            found = false;
    UINT32          i, j = 0;
    UINT16 major = 0;
    UINT16 minor = 0;
    UINT16 hotfix = 0;
    UINT16 build = 0;
    UINT32 itr = 0;

    // Zero out the update flag structure

    ZeroMem(&update_flags, sizeof(UPDATE_FLAGS_LIB));
    ZeroMem((char*)&IpuUpdatedInfo, sizeof(IPU_UPDATED_INFO));

```



```
// Initialize the EFI Toolkit Library. Set BS, RT, &ST globals
// BS = Boot Services RT = RunTime Services
// ST = System Table

InitializeLib (ImageHandle, SystemTable);

Print (L"\n Intel ® Firmware Update Utility Sample Application \n");
Print (L"\n Intel ® Firmware Update Utility Version: %d.%d.%d.", VER_MAJOR,
VER_MINOR, VER_HOTFIX);
Print (L"%d\n", VER_BUILD);

Print (L"\n");

Print (ID_INFO_1);

// Determine the command line arguments

Status = ParseCommandLine (ImageHandle, ImageName, &bAllowSV,
&bUsePassword, &bPid, &bF, &bPdt, &bIshVer);
if (EFI_ERROR (Status))
{
    DEBUG ((D_ERROR, "Unable to process command line - %r\n", Status));
    return Status;
}

// Display ISH FW Version with '/g' option

if (bIshVer){
    Status = GetPartVersion(FPT_PARTITION_NAME_ISHC, &major, &minor, &hotfix,
&build);

    if (EFI_ERROR (Status))
    {
        DEBUG ((D_ERROR, "GetPartVersion Error %r\n", Status));
        return Status;
    }

    Print(L"ISH FW Version: %d.%d.%d.%d \n\n", major, minor, hotfix, build);
}

//
// Load image into memory buffer
//
Print (L"\n Loading image into memory : ...\n");

Status = GetUpdateImage (ImageHandle, ImageName, &ImageLength,
&ImageBuffer);
if (EFI_ERROR (Status)) {
    Print(L" %r ",Status);
    return Status;
}
```



```

if (bPdt)
{
    Print(L"Sending Image for Executing PDT Update. \n");

    Status = HeciPdt((char *)ImageBuffer, (unsigned int)ImageLength);
    if (EFI_ERROR(Status)) {
        Print(L"Send Failed. \n");
    }
    else {
        Print(L"Send Succeeded. \n");
    }
}
else
{
    //
    // Get the current status of the CSE FWUpdate Client - verifies if the
client is
    // installed
    //

    if (GetLastStatus(&lastStatus))
    {
        Print (ID_ERROR_19, FWU_LAST_STATUS);
        return EFI_SUCCESS;
    }
    //
    // Is there a pending reset?
    //

    if (GetLastUpdateResetType (&lastResetType))
    {
        Print (ID_ERROR_19, FWU_LAST_STATUS);
        return EFI_SUCCESS;
    }
    if (STATUS_UPDATE_HOST_RESET_REQUIRED == lastStatus)
    {
        Print (ID_ERROR_51, FWU_REBOOT_NEEDED);
        return EFI_SUCCESS;
    }

    if (IsUpdateStatusPending (lastStatus))
    {
        Print (ID_ERROR_20, FWU_UPD_PROCESS);
        return EFI_SUCCESS;
    }

    switch (lastResetType)
    {
    case MFT_PART_INFO_EXT_UPDATE_ACTION_HOST_RESET:

    case MFT_PART_INFO_EXT_UPDATE_ACTION_GLOBAL_RESET:
        Print (ID_ERROR_51, FWU_REBOOT_NEEDED);
        return EFI_SUCCESS;
        break;
    default:

```



```
        break;
    }

    Print (ID_INFO_3);

    //
    // Is update supported?
    //
    if (GetInterfaces (&interfaces))
    {
        Print (ID_ERROR_19, FWU_LAST_STATUS);
        return EFI_SUCCESS;
    }

    switch (interfaces)
    {
    case FW_UPDATE_DISABLED:
        Print (L"Local FWUpdate is Disabled\n");
        return EFI_SUCCESS;
    case FW_UPDATE_PASSWORD_PROTECTED:
        Print (L"Local FWUpdate is Password Protected\n");
        break;
    case FW_UPDATE_ENABLED:
        break;
    default:
        break;
    }

    Print (L"\n Checking Firmware Parameters ... \n \n");
    CheckPolicyStatus = CheckPolicyBuffer((char *)ImageBuffer,
(INT32)ImageLength, (INT32)bAllowSV, &Upd_Type, &ver);

    switch (Upd_Type)
    {
    case DOWNGRADE_SUCCESS:

    case SAMEVERSION_SUCCESS:

    case UPGRADE_SUCCESS:
        break;

    case DOWNGRADE_FAILURE:
        Print (L"FW Update downgrade not allowed\n");
        return EFI_SUCCESS;
        break;
    case SAMEVERSION_FAILURE:
        Print (L"FW Update same version not allowed, specify /s on
command line\n");
        return EFI_SUCCESS;
        break;
    default:
        break;
    }
}
```




```

        if(bPid)
        {
            Print(L"\n Executing ISH Partial FWUpdate");

            ComparePartID = FPT_PARTITION_NAME_ISHC;
            //Print(L"\n compareID: 0x%x",ComparePartID);

            //Get Partition Attribute from Firmware
            if (FWU_ERROR_SUCCESS != (fwuError =
GetExtendedIpuPartitionAttributes(&FwuGetIpuAttrbMsgInfo,
FWU_IPU_UPDATE_OPERATION)))
            {
                DisplayTextForReturnErrorCode(fwuError);
                return fwuError;
            }

            PartId = 0;
            //Loop through expected partitions from FW to find partition
requested
            for(j=0;j<FwuGetIpuAttrbMsgInfo.NumOfPartition;j++)
            {
                if(ComparePartID ==
FwuGetIpuAttrbMsgInfo.PtAttribute[j].PtNameId)
                {
                    PartId =
FwuGetIpuAttrbMsgInfo.PtAttribute[j].PtNameId;
                    found = true;
                    break;
                }
            }

            if(!found)
            {
                DisplayTextForReturnErrorCode(FWU_PID_NOT_EXPECTED);
                //Print(L"ParID: 0x%x\tInstId: 0x%x
\n",ComparePartID,hexValueInstId);
                return FWU_PID_NOT_EXPECTED;
            }
            Print(L"%s", ID_WARN_0);
            ////Actual Partial FW update
            //
            // Password hack for testing - replace with OEM version if password
required
            //
            if (!bUsePassword)
            {
                ZeroMem (Password, sizeof (Password));
            }
            if (bUsePassword)
            {
                FWUpdateStatus = FwUpdatePartialBuffer ((char
*)ImageBuffer, (unsigned
int)ImageLength,PartId,0,&IpuUpdatedInfo,"P@ssw0rd",FWU_ENV_MANUFACTURING,
mOemId, update_flags, &DisplaySendStatus);

```



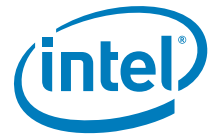
```
    }
    else
    {
        FWUpdateStatus = FwUpdatePartialBuffer ((char
*)ImageBuffer, (unsigned int)ImageLength, PartId, 0, &IpuUpdatedInfo, Password,
FWU_ENV_MANUFACTURING, mOemId, update_flags, &DisplaySendStatus);
    }

    if (FWU_ERROR_SUCCESS != FWUpdateStatus)
    {
        DisplayTextForReturnErrorCode(FWUpdateStatus);
        if (ImageBuffer)
        {
            FreePool (ImageBuffer);
        }
        return EFI_SUCCESS;
    }

    if (ImageBuffer)
    {
        FreePool (ImageBuffer);
    }
}
else{
    //
    // Password hack for testing - replace with OEM version
    if password required
    //
    Print(L"\n");
    Print(L"%s \n", ID_WARN_0);
    /*if (!bUsePassword)
    {
        ZeroMem (Password, sizeof (Password));
    }*/

    //if (bUsePassword)
    //{
        // FWUpdateStatus = FwUpdateFullBuffer
        ((char *)ImageBuffer, (unsigned int)ImageLength, "P@ssw0rd", 0,
        FWU_ENV_MANUFACTURING, mOemId, update_flags, &DisplaySendStatus);
        //}
        //else
        //{
            FWUpdateStatus = FwUpdateFullBuffer ((char
*)ImageBuffer, (unsigned int)ImageLength, Password, 0,
FWU_ENV_MANUFACTURING, mOemId, update_flags, &DisplaySendStatus);
        //}

        if (FWU_ERROR_SUCCESS != FWUpdateStatus)
        {
            DisplayTextForReturnErrorCode(FWUpdateStatus);
            if (ImageBuffer)
            {
                FreePool (ImageBuffer);
            }
        }
    }
}
```



```

        return EFI_SUCCESS;
        //return FWUpdateStatus;
    }

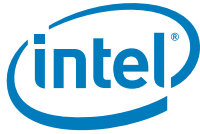
    if (ImageBuffer)
    {
        FreePool (ImageBuffer);
    }
}

//
// Image downloaded to FW Update Client
// Now query the status of the update being applied
//
Print(L"\n FW Update: [ 0 %% ]\r");

index = 0;
//
// Loop through Polling for Fw Update Stages
//
ProgressDot();
do {
    //We mod4 the index to determine which ascii animation frame
    to display for this iteration.
    indexMod = (++index % 4);
    //symbol = (++index % 2 == 0)?'|': '-';
    switch(indexMod)
        //loop through (|) (/) (-) (\) (|) (/) ...
        {
            case CMD_LINE_STATUS_UPDATE_1: symbol = '|'; break;
            case CMD_LINE_STATUS_UPDATE_2: symbol = '/'; break;
            case CMD_LINE_STATUS_UPDATE_3: symbol = '-'; break;
            case CMD_LINE_STATUS_UPDATE_4: symbol = '\\'; break;
        }
    Status =
FWUpdate_QueryStatus_Get_Response(&UpdateStatus, &TotalStages,
&PercentWritten, &lastStatus, &lastResetType);
    if(PercentWritten > 100)
    {
        break;
    }
    if (Status == FWU_ERROR_SUCCESS)
    {
        Print (L"FW Update: [ %d%% (%c)]\r"
,PercentWritten, symbol);
    } else if (lastStatus != STATUS_UPDATE_NOT_READY)
    {
        Print (L"\n");
        break; //break out of the loop
    }

    BS->Stall(100000); // Wait 1 sec before polling again
    if(timer30s >= 30)
    {
        percentdiff = PercentWritten - percentage0s;

```



```
        if(percentdiff < 1)
        {
            //TODO: Add timeout when add cmdline option
            //Status = FWU_UPDATE_TIMEOUT;
        } else
        {
            percentage0s = PercentWritten;
            timer30s = 0;
        }
    } else
    {
        timer30s++;
    }
} while ((PercentWritten != 100) && (Status ==
FWU_ERROR_SUCCESS));

switch (Status)
{
case FWU_NO_MEMORY:

case FWU_IME_NO_DEVICE:
    Print (ID_ERROR_68, FWU_UPDATE_POLLING_FAILED);
    return EFI_SUCCESS;
case FWU_IME_NOT_READY:
    DisplayTextForReturnErrorCode(status);
    return EFI_SUCCESS;
case FWU_ERROR_FW:
    Print (ID_ERROR_69, FWU_ERROR_FW, UpdateStatus);
    return EFI_SUCCESS;
default:
    break;
}

switch (lastStatus)
{
case STATUS_SUCCESS:
    switch (lastResetType)
    {
    case MFT_PART_INFO_EXT_UPDATE_ACTION_NONE:

    case MFT_PART_INFO_EXT_UPDATE_ACTION_CSE_RESET:
        Print (L"\nFW Update is completed successfully.\n");
        break;
    case MFT_PART_INFO_EXT_UPDATE_ACTION_HOST_RESET:

    case MFT_PART_INFO_EXT_UPDATE_ACTION_GLOBAL_RESET:
        Print (L"\nFW Update is complete and a reboot will run
the new FW.\n");
        break;
    default:
        Print (L"\nFW Update is complete and a reboot will run
the new FW.\n");
        break;
    }
    fwuError = FWU_ERROR_SUCCESS;
```



```

        break;
    case STATUS_UPDATE_IMAGE_INVALID:
        DisplayTextForReturnErrorCode(FWU_IMG_HEADER);
        break;
    case STATUS_UPDATE_INTEGRITY_FAILURE:
        DisplayTextForReturnErrorCode(FWU_SGN_MISMATCH);
        break;
    case STATUS_UPDATE_SKU_MISMATCH:
        DisplayTextForReturnErrorCode(FWU_SKU_MISMATCH);
        break;
    case STATUS_UPDATE_FW_VERSION_MISMATCH:
        DisplayTextForReturnErrorCode(FWU_VER_MISMATCH);
        break;
    case STATUS_UPDATE_GENERAL_FAILURE:
        DisplayTextForReturnErrorCode(FWU_GENERAL);
        break;
    case STATUS_UPDATE_OUT_OF_RESOURCES:
        DisplayTextForReturnErrorCode(FWU_NO_MEMORY);
        break;
    case STATUS_UPDATE_AUDIT_POLICY_FAILURE:
        DisplayTextForReturnErrorCode(FWU_AUDIT_POLICY_FAILURE);
        break;
    case STATUS_UPDATE_ERROR_CREATING_FT:
        DisplayTextForReturnErrorCode(FWU_ERROR_CREATING_FT);
        break;
    case STATUS_UPDATE_SAL_NOTIFICATION_ERROR:
        DisplayTextForReturnErrorCode(FWU_SAL_NOTIFICATION_ERROR);
        break;
    case STATUS_INVALID_OEM_ID:
        DisplayTextForReturnErrorCode(FWU_INVALID_OEM_ID);
        break;
    case STATUS_DOWNGRADE_NOT_ALLOWED_VCN_RESTRICTION:
        DisplayTextForReturnErrorCode(FWU_IMAGE_UNDER_VCN);
        break;
    case STATUS_DOWNGRADE_NOT_ALLOWED_SVN_RESTRICTION:
        Print("FW downgrade is not allowed due to SVN
restriction.\n");
        break;
    case STATUS_UPDATE_IMAGE_BLACKLISTED:
        Print("FW update/downgrade is not allowed to the supplied FW
image.\n");
        break;
    default:
        DEBUG ((D_ERROR, "lastStatus = %d\n", lastStatus));
        DisplayTextForReturnErrorCode(FWU_GENERAL);
        break;
    }
    return EFI_SUCCESS;
}
}
}

```



8.2 Function Description

This section describes all the functions listed in FWUpdateLib.h. It explains the purpose, Input arguments and return types.

8.2.1 Get Interfaces

```
unsigned int GetInterfaces(unsigned short *interfaces);
```

Purpose: This function gets the local FW update settings from Intel® Management Engine BIOS Extension (Intel®MEBX) to determine whether or not Firmware can be updated.

Arguments	Interfaces - whether the Local FW Update is disabled (0) or enabled (1) or password protected (2)
Returns	Gets the Interfaces from HECI 0 = Success Non-zero value = Failure

8.2.2 Get Last Status

```
unsigned int GetLastStatus(unsigned int *lastStatus);
```

Purpose: This function will get the previous FW update status to ensure that FW update was successfully executed.

Arguments	Laststatus - Last FW Update process Status (E.g. Success, Invalid OEM ID, FW Version mismatch etc) Refer "me_status.h" for specific values
Returns	Gets the last FW update status from HECI 0 = Success Non-zero value = Failure

8.2.3 Get Last Update Reset Type

```
unsigned int GetLastUpdateResetType(unsigned int *lastResetType);
```

Purpose: This function will get the last Update Reset type to determine what type of system reset is required to load the partition into the memory.



Arguments	LastResetType - The last FW Update reset type No reset – 0 Host reset – 1 ME – 2 Global - 3
Returns	Gets the last FW update status from HECI 0 = Success Non-zero value = Failure

8.2.4 Check Policy

```
unsigned int CheckPolicy(char* ImageFileLib, int AllowSV, UPDATE_TYPE
*Upd_Type, VersionLib *ver);
```

Purpose: This function determines whether it is a FW upgrade/downgrade or same version update using a file.

Arguments	Image File - Binary Image file AllowSV - Allow Same Version flag (Set to 1 to execute same version flow) Update Type - Update Type Output. Can be DOWNGRADE_SUCCESS = 0, DOWNGRADE_FAILURE = 1, SAMEVERSION_SUCCESS = 2, SAMEVERSION_FAILURE = 3, UPGRADE_SUCCESS = 4, UPGRADE_PROMPT = 5, Ver - FW Version (Major, Minor, Hotfix, Build)
Returns	0 = Success Non-zero value = Failure

8.2.5 Check Policy Buffer

```
unsigned int CheckPolicyBuffer(char* buffer, int bufferLength, int AllowSV,
UPDATE_TYPE *Upd_Type, VersionLib *ver);
```

Purpose: This function determines whether it is a FW upgrade/downgrade or same version update using buffer.



Arguments	Buffer - buffer to access BufferLength - Length of buffer AllowSV - Allow Same Version flag Update Type - Update Type Output. Can be DOWNGRADE_SUCCESS = 0, DOWNGRADE_FAILURE=1, SAMEVERSION_SUCCESS=2, SAMEVERSION_FAILURE=3, UPGRADE_SUCCESS=4, UPGRADE_PROMPT=5, Ver - FW Version (Major, Minor, Hotfix, Build)
Returns	0 = Success Non-zero value = Failure

8.2.6 Verify OEM Id

```
bool VerifyOemId(_UUID id);
```

Purpose: This function verifies the OEM ID provided by the user with the one embedded in the FW.

Arguments	Id - OEM id
Returns	True = OEM ID matched False = OEM id mismatch

8.2.7 Get Ipu Partition Attributes

```
unsigned int GetIpuPartitionAttributes(FWU_GET_IPU_PT_ATTRB_MSG_REPLY  
*FwuGetIpuAttrbMsgInfo);
```

Purpose: This function gets the number of Independent partial update partition attributes that is currently present and also the list of expected IPU to be updated.

Arguments	Out parameter: FWU_GET_IPU_PT_ATTRB_MSG_REPLY - is a data structure with IPU related information
Returns	0 = Success 8193 = Heci Device not found 8204 = Heci message has incorrect message type 8728 = Heci Buffer Size is Small Error 8710 = Insufficient memory Error 8776 = Failure to Send or Receive the Get Partition Attribute Command Or even when FW returns an error status after receiving command



8.2.8 Get FW Update Info Status

```
unsigned int GetFwUpdateInfoStatus(FWU_INFO_FLAGS *StatusFlags);
```

Purpose: This function gets the current status of the firmware.

Note: This API is not used by the FW Update tool. It is being used by the UNS services.

Arguments	StatusFlags - BITS 0:1 (2 bits) 0 = No recovery; 1 = Full Recovery Mode; 2 = Partial Recovery Mode (unused at present). BIT2; IPU_NEEDED bit, if set we are in IPU_NEEDED state. BIT3; FW_INIT_STATUS done. BIT4; FWU_IN_PROGRESS
Returns	0 = Success 8193 = Heci Device not found 8204 = Heci message has incorrect message type 8213 = Heci Buffer Size is Small Error 8710 = Insufficient memory Error 8777 = Failure in Send or Receive of the Get Info Status Command. Or even when FW returns an error status after receiving command

8.2.9 FW Update Query Status Get Response

```
unsigned int FWUpdate_QueryStatus_Get_Response(unsigned int* UpdateStatus,
unsigned int *TotalStages, unsigned int* PercentWritten, unsigned int *
LastUpdateStatus, unsigned int * LastResetType );
```

Purpose: This function queries FW to get response regarding the different stages of FW Update process.



Arguments	UpdateStatus - indicates the current FW Update stage being executed. TotalStages - indicates the total number of FW Update stages available. PercentWritten - indicates the percentage complete of the FW Update process LastUpdateStatus - indicates the status of the FW Update process just completed LastResetType - indicates Reset type required for the FW Update process just completed
Returns	0= Success 1 = Invalid Manifest Data in partition 8193 = Heci Device not found 8204 = Heci message has incorrect message type 8213 = Heci Buffer Size is Small Error 8710 = Insufficient memory Error 8724 = Failure to send or receive messages to heci to get Status Info 8741 = FW returns incorrect Message Type

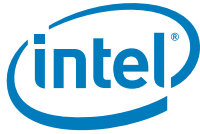
8.2.10 FW Update Full – Using Buffer

```
unsigned int FwUpdateFull (char* buffer, unsigned int bufferLength, char* _pwd,int  
_forceResetLib, unsigned int UpdateEnvironment,_UUID OemID,  
UPDATE_FLAGS_LIB update_flags, void(*func)(float,float));
```

Purpose: This function performs the full FW Update using the Buffer provided by the calling function.



Arguments	<p>Buffer – Buffer with the update image</p> <p>Buffer Length – Length of buffer</p> <p>Password – MEBX Password</p> <p>ForceResetLib – Flag to perform system reset</p> <p>UpdateEnvironment – differentiates various firmware update process environment within the firmware (manufacturing/non-manufacturing)</p> <p>UUID OEMID – OEM ID</p> <p>update_flags – flag to indicate FW of recovery/rollback</p> <p>Func pointer – (bytes of Binary</p>
Returns	<p>0 = Success</p> <p>Non-zero value = Failure</p>



9 Intel® Manifest Extension Utility (Intel®MEU)

The Intel® Manifest Extension Utility (MEU) inputs a firmware binary created by a 3rd party and outputs an independent-updateable partition (IUP) that is compressed and signed. After completing this process the signed binary can be added to the flash image using the Intel® FIT tool.

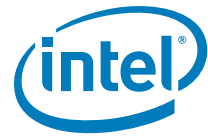
The Intel®MEU tool completes the following steps:

- Creates an Independent Updatable Partition (IUP) by adding manifest and meta-data information to the firmware.
- Calls the OpenSSL tool as the signing infrastructure tool to sign the partition.

9.1 Usage

Refer to the *Signing & Manifesting Guide* in the latest Intel CSE FW kit for details on MEU usages, signing & manifesting flows, etc.





A Intel® CSE NVARs

This appendix only covers fixed offset variables that are directly available to FPT and FPTW. A complete list of NVARs can be found in the *Firmware Variable Structures for Intel® Converged Security and Engine*. All of the fixed offset variables have an ID and a name. The `-CVAR` option displays a list of the IDs and their respective names. The variable name must be entered exactly as displayed below.

This table is for reference use only and will be updated later.

Table A-1. NVARs Descriptions

Fixed Offset Name	Description	Data Length (in Bytes)	Expected Value	Reset Type	Mfg. Post EOM/Pre EOP
Non-Application Specific Fixed Offset Item Descriptions					
MEBxPassword	<p>Overrides the MEBx default password. It must be at least eight characters and not more than 32 characters in length. All characters must meet the following:</p> <p>ASCII(32) <= char <= ASCII(126)</p> <p>Cannot contain these characters: , : "</p> <p>Must contain for complexity:</p> <ul style="list-style-type: none"> a. At least one Digit character (0 - 9) b. At least one 7-bit ASCII non alpha-numeric character above 0x20 (e.g. ! \$;) c. Both lower-case and upper case Latin d. underscore and space are valid characters but are not used in determination of complexity <p>See section 2.7 for format and strong password requirements.</p>	8<=N<=32	Password	CSE	Yes



Fixed Offset Name	Description	Data Length (in Bytes)	Expected Value	Reset Type	Mfg. Post EOM/Pre EOP
OEMSkuRule	<p>UINT32 (little endian) value. This controls what features are permanently disabled by OEM.</p> <p>Notes:</p> <p>There are reserved bits that the must not be changed for proper platform operation. The user should only modify the bit(s) for the feature(s) they wish to change. There is NO ability to change features one at a time. This NVAR sets OEM Permanent Disable for ALL features. In addition prior updating or changing any of available settings it is highly recommended that the user first retrieves the current OEM Sku Rule and toggling only the desired bits, and then resave them.</p> <p>This will not enable functionality that is not capable of working in the target hardware SKU. Please see the respective Firmware Bring-up Guide for a list of what features are capable with what firmware bundle and Hardware SKU of Intel 9 Series Chipset.</p>	4	<p>Feature Capable: 1 Feature Permanently disabled: 0</p> <p>Bit Description Notes</p> <p>31 Reserved</p> <p>30 Reserved</p> <p>29 PTT</p> <p>29:22 Reserved</p> <p>21 TLS</p> <p>20 DAL</p> <p>19 Reserved</p> <p>18 KVM</p> <p>2 Reserved</p> <p>17 Reserved</p> <p>16 ME Network Disable</p> <p>15:13 Reserved</p> <p>12 PAVP</p>	Global	No



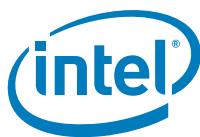
Fixed Offset Name	Description	Data Length (in Bytes)	Expected Value	Reset Type	Mfg. Post EOM/Pre EOP
FeatureShipState	<p>UINT32 (little endian) value. This controls what features are enabled or disabled. These features may be enabled /disabled by mechanisms such as MEBx or provisioning. This setting is only relevant for features NOT permanently disabled by the OEM Permanent Disable. This will not enable functionality that is not capable of working in the target hardware SKU. Please see the respective Firmware Bring-up Guide for a list of what features are capable with what firmware bundle and Hardware SKU of Intel 8 Series Chipset.</p> <p>Notes: There are reserved bits that the must not be changed for proper platform operation. The user should only modify the bit(s) for the feature(s) they wish to change. There is NO ability to change features one at a time. This NVAR sets OEM Permanent Disable for ALL features. In addition prior updating or changing any of available settings it is highly recommended that the user first retrieves the current Feature Shipment Time State and toggling only the desired bits, and then resave them.</p>	4	<p>Feature Enabled: 1 Feature Disabled: 0</p> <p>Bit Description Notes</p> <p>31:30 Reserved</p> <p>29 PTT</p> <p>28:11 Reserved</p> <p>10 ISH</p> <p>3:9 Reserved</p> <p>2 Manageability Full</p> <p>1:0 Reserved</p> <p>Note: When disabling PTT using Feature Shipment Time state NVAR, please execute a reset after executing fpt.efi –commit to ensure PTT is disabled completely.</p>	Global	Yes



Fixed Offset Name	Description	Data Length (in Bytes)	Expected Value	Reset Type	Mfg. Post EOM/Pre EOP
OEM Tag	A human readable 32-bit number to describe the flash image represented by value	4	Readable 32 bit hex value identifying the image. Can be empty (Null).	Global	No



Fixed Offset Name	Description	Data Length (in Bytes)	Expected Value	Reset Type	Mfg. Post EOM/Pre EOP
GPIO	GPIO	60	<p>GPIO groups and pad range for each grp pad#</p> <p>For LP platforms:</p> <p>GPP_A 0-23 GPP_B 0-23 GPP_C 0-23 GPP_D 0-19 GPP_E 0-23 GPP_F 0-19 GPP_G 0-7 GPP_H 0-23 GPD 0-11</p> <p>For H platforms:</p> <p>GPP_A 0-23 GPP_B 0-23 GPP_C 0-23 GPP_D 0-15 GPP_E 0-12 GPP_F 0-23 GPP_G 0-15 GPP_H 0-23 GPP_I 0-14 GPP_J 0-9 GPP_K 0-11 GPD 0-11</p> <p>Example read of GPIO: Variable: "gpio" Value: 0x0000 : 00 00 00 00 04 00 00 00 06 00 00 00 01 00 00 00 0x0010 : 00 00 00 00 01 00 00 00 04 00 00 00 0C 00 00 00 0x0020 : 01 00 00 00 00 00 00 00 08 00 00 00 01 00 00 00 0x0030 : 0F 00 00 00 01 00 00 00 00 00 00 00</p> <p>Note: the only locations that can be modified are underlined above.</p> <p>The format for updating the GPIO is as follows...</p> <p>Gpio = 0x000000000030000000110000000010000 000000000001000000002000000170000 00010000000000000000080000000030000 0013000000010000000000000000</p> <p>RST = GPP_D_17 IRQ = GPP_C_23 DFU = GPP_D_19</p>	CSE	No



Fixed Offset Name	Description	Data Length (in Bytes)	Expected Value	Reset Type	Mfg. Post EOM/Pre EOP
FW Update State	Enabled Firmware Update Capability	1	0 = disabled 1 = enabled	Global	Yes
eDP Port Config	EDP Port Configuration. Up to two ports can be enabled 0x00 - 0x01 - A 0x02 - B 0x04 - C 0x08 - D 0x10 - E 0x20 - F	1	0x0:None/None 0x1:A/None 0x2:B/None 0x3:A/B 0x4:C/None 0x5:C/A 0x6:C/B 0x8:D/None 0x9:D/A 0xa:D/B 0xc:D/C 0x10:E/None 0x11:E/A 0x12:E/B 0x14:E/C 0x18:E/D 0x20:F/None 0x21:F/A 0x22:F/B 0x24:F/C 0x28:F/D 0x30:F/E	Global	No
LSPCON Port Config	LSPCON Port Configuration.	1		Global	No
Attestation keyBox		16384	File		
CSME Measured Boot to TPM		1	0 = Disabled 1 = Enabled		
Discrete vPro NIC on-board State		1	0 = Disabled 1 = Enabled		
EOM Settings	FlexEom NVAR	3	{0, 1, 2, 3, 4, 5, 6, 7}, // PossibleValuesIDs {"Lock(Flash,Config)", "Lock(Flash,Config) on 1st Boot", "Lock(Config)", "Lock(Config) on 1st Boot", "Lock(Flash)", "Lock(Flash) on 1st Boot", "Lock(none)", "Lock(none) on 1st Boot"}, // possibleValuesStrings		
On Board Discrete vPro NIC SMBus address					
Delayed Authentication Mode Config	Enables Delayed Authentication Mode on the platform	1	0 = Disabled 1 = Enabled	CSE	Yes



Fixed Offset Name	Description	Data Length (in Bytes)	Expected Value	Reset Type	Mfg. Post EOM/Pre EOP
Unconfigure On RTC	Enables platform to unconfigure on RTC removal	1	0 = Disabled 1 = Enabled	CSE	Yes
Field Programming Fuses					
Glitch Detection	Enables/Disabled the Glitch detection FPF	1	0 = Disabled 1 = Enabled	CSE	NO
Intel(R) PTT	Enables / Disables the fTPM / PTT FPFs	1	0 = Disabled 1 = Enabled	CSE	No
BSP Initialization	Indicating the BSP initialization on boot	1	0 = Disabled 1 = Enabled	CSE	No
CPU Debugging	Indication CPU debug capabilities	1	0 = Disabled 1 = Enabled	CSE	No
Error Enforcement Policy 0	Error Enforcement Policy 0	1	0 = Disabled 1 = Enabled	CSE	No
Error Enforcement Policy 1	Error Enforcement Policy 1	1	0 = Disabled 1 = Enabled	CSE	No
Force Boot Force Boot Guard ACM	Indicates Boot Guard ACM is enforced or not	1	0 = Disabled 1 = Enabled	CSE	No
Key Manifest ID	Contains key manifest required for authentication	1	0 = Disabled 1 = Enabled	CSE	No
Measured Boot	One of the applicable profiles for Boot Guard	1	0 = Disabled 1 = Enabled	CSE	No
OEM ID	OEM ID	1	0 = Disabled 1 = Enabled	CSE	No
OEM Platform ID	OEM Platform ID	1	0 = Disabled 1 = Enabled	CSE	No
OEM Secure Boot Policy	OEM Secure Boot Policy	1	0 = Disabled 1 = Enabled	CSE	No
Persistent PRTC Backup Power	Persistent PRTC Backup Power	1	0 = Disabled 1 = Enabled	CSE	No
Protect BIOS Environment	Indicated if BIOS environment protection is enforced or not	1	0 = Disabled 1 = Enabled	CSE	No
S3 Optimization	S3 optimization for Boot Guard	1	0 = Disabled 1 = Enabled	CSE	No
Txt Supported	Txt Supported	1	0 = Disabled 1 = Enabled	CSE	No
Verified Boot	One of the applicable profiles for Boot Guard	1	0 = Disabled 1 = Enabled	CSE	No
OEM Public Key Hash	Hash of the provided OEM public key	32	32 Hex Pairs with space between pairs	CSE	No
Second OEM Public Key Hash	Hash of the second provided OEM public key	32	32 Hex Pairs with space between pairs	CSE	No

§ §

