



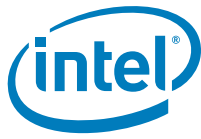
System Tools - Intel® Management Engine Firmware 12.0

User Guide

May 2018

Revision 1.3

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 © 2018, Intel Corporation. All Rights Reserved.



Contents

1	Introduction	8
1.1	Terminology	8
1.2	Reference Documents	15
2	Preface	16
2.1	Overview	16
2.2	Image Editing Tools	16
2.3	Manufacturing Line Validation Tool	17
2.4	Intel® Management Engine Setting Checker Tool	17
2.5	Operating System Support	18
2.6	Generic System Requirements	19
2.7	Error Return	19
2.8	Usage of Double-Quote Character (")	19
2.9	PMX Driver Limitation	20
2.10	Control Handler Support	20
3	Intel® Flash Image Tool	21
3.1	System Requirements	21
3.2	Flash Image Details	21
3.2.1	Flash Space Allocation	22
3.3	Required Files	23
3.4	Intel® Flash Image Tool	23
3.4.1	Configuration Files	23
3.4.2	Creating New Configuration	23
3.4.3	Opening Existing Configuration	23
3.4.4	Saving Configuration	23
3.4.5	Environment Variables	24
3.4.6	Modifying the Flash Descriptor Region	27
3.4.7	Descriptor Region Length	27
3.4.8	Setting the Number and Size of the Flash Components	28
3.4.9	SPI Software Binding (PCH Replacement)	29
3.4.10	Region Access Control	29
3.4.11	VSCC Table	33
3.4.12	Adding New Table	33
3.4.13	Removing Existing VSCC Table	34
3.4.14	FPF Configuration	35
3.4.15	Modifying the Intel® Management Engine Region	35
3.4.16	Setting the Intel® Management Engine Region Binary File	35
3.4.17	Setting the Intel® PMC Binary File	36
3.4.18	Intel® Management Engine Section	36
3.4.19	Power	37
3.4.20	Manageability Application Section	38
3.4.21	Platform Protection	39
3.4.22	Provisioning Section	40
3.4.23	Gbe (LAN) Region Settings	42
3.4.24	Setting Gbe Region Length Option	42
3.4.25	Setting Gbe Region Binary File	42
3.4.26	Enabling/Disabling GbE Region	42
3.4.27	Modifying PDR Region	43
3.4.28	Setting PDR Region Length Option	43
3.4.29	Setting PDR Region Binary File	43
3.4.30	Enabling/Disabling PDR Region	43



3.4.31	Modifying BIOS Region	44
3.4.32	Setting BIOS Region Length Parameter	44
3.4.33	Setting the BIOS Region Binary File	44
3.4.34	Enabling/Disabling the BIOS Region	44
3.4.35	Building Flash Image	44
3.4.36	Decomposing Existing Flash Image	45
3.4.37	Command Line Interface	45
3.4.38	Example – Decomposing Image and Extracting Parameters	47
3.4.39	More Examples of FIT CLI	48
4	Flash Programming Tool	49
4.1	System Requirements	49
4.2	Flash Image Details	50
4.3	Microsoft Windows® Required Files	50
4.4	EFI Required Files	50
4.5	DOS Required Files	51
4.6	Programming Flash Device	51
4.6.1	Stopping Intel® ME SPI Operations	51
4.7	Programming NVARs	51
4.7.1	Programming GPIO NVAR	52
4.8	Usage	53
4.9	Updating Hash Certificate through NVAR	58
4.10	Fparts.txt File	60
4.11	Examples	60
4.11.1	Complete SPI Flash Device with Binary File	60
4.11.2	Program Specific Region	61
4.11.3	Program SPI Flash from Specific Address	62
4.11.4	Dump Full Image	62
4.11.5	Dump Specific Region	62
4.11.6	Display SPI Information	63
4.11.7	Verify Image with Errors	63
4.11.8	Verify Image Successfully	64
4.11.9	Get Intel® ME settings	64
4.11.10	CVAR Configuration File Generation (-cfggen)	65
5	Intel® MEmanuf and MEmanufWin	69
5.1	Windows® PE Requirements	69
5.2	How to Use Intel® MEmanuf	69
5.3	Usage	70
5.3.1	Host based Tests	74
5.4	Intel® MEmanuf –EOL Check	75
5.4.1	MEmanuf.xml File	75
5.4.2	MEmanuf –EOL Variable Check	102
5.4.3	MEmanuf –EOL Config Check	103
5.4.4	Output/Result	103
5.5	Examples	103
5.5.1	Example for Consumer Intel® ME FW SKU	103
5.5.2	Example for Corporate Intel® ME FW SKU	105
6	Intel® MEInfo	108
6.1	Windows® PE Requirements	108
6.2	Usage	108
6.3	Examples	121
6.3.1	Consumer Intel® ME FW SKU Sample Output	121
6.3.2	Corporate Intel® ME FW SKU Sample Output	124
6.3.3	Retrieve Current Value of Flash Version	126



6.3.4	Checks Whether Computer Has Completed Set-up and Configuration Process	126
7	Intel® ME Firmware Update	127
7.1	Requirements	127
7.2	Windows® PE Requirements	128
7.3	Enabling and Disabling Intel® FWUpdate	128
7.4	FWUpdate Flows	128
7.4.1	Full FWUpdate	128
7.4.2	Partial FWUpdate	129
7.5	Usage	129
7.6	Examples	130
7.6.1	Updates Intel® ME with Firmware Binary File	130
7.6.2	Partial Firmware Update	131
7.6.3	Display Supported Commands	131
7.6.4	Language Codes	132
8	UEFI Sample Application Leveraging FWUpdate API Library	134
8.1	Getting Started - FWUpdate Library	134
8.1.1	Introduction	134
8.1.2	Environment	134
8.1.3	Setup	134
8.1.4	Sample App	134
8.2	Function Description	145
8.2.1	Get Interfaces	145
8.2.2	Get Last Status	145
8.2.3	Get Last Update Reset Type	145
8.2.4	Check Policy	146
8.2.5	Check Policy Buffer	146
8.2.6	Verify OEM Id	147
8.2.7	Get Ipu Partition Attributes	147
8.2.8	Get FW Update Info Status	148
8.2.9	FW Update Query Status Get Response	148
8.2.10	FW Update Full – Using Buffer	149
8.3	FW Update Partial Buffer	150
8.3.1	PDT Data (Sensor Calibration Data) Update	151
8.3.2	ISH Firmware Version	151
9	Intel® Manifest Extension Utility (Intel® MEU)	152
9.1	Usage	152

Figures

3-1	SPI Flash Image Regions	22
3-2	Environment Variables Dialog	25
3-3	Build Settings Dialog	27
3-4	Descriptor Region Length Parameter	28
3-5	Flash Settings > Flash Components	28
3-6	Flash Settings > Flash Configuration	29
3-7	Descriptor Region Master Access Section	33
3-8	Add VSCC Table Entry Dialog	34
3-9	Deleting VSCC Table Entry Dialog	35
3-10	Intel® ME Kernel	37
3-11	Power	38
3-12	Manageability Application Section	39
3-13	Platform Protection Section	40
3-14	Provisioning Configuration Section	41



3-15	Provisioning Configuration Section (Cont...)	42
3-16	GbE Region Options	42
3-17	PDR Region Options	43
3-18	BIOS Region Parameters	44
4-1	Raw Hash Values from Certificate File	59
4-2	Sample Hash.txt File	59

Tables

2-1	OS Support for Tools	18
2-2	Tools Summary	19
3-1	Flash Image Regions – Description	22
3-2	Build Settings Dialog Options	26
3-3	Region Access Control Table	30
3-4	CPU/BIOS Access	31
3-5	FIT Command Line Options	46
4-1	FPT OS Requirements	50
4-2	Named Variables Options	52
4-3	Command Line Options for fpt.efi, fpt.exe and fptw.exe	53
4-4	FPT-closemfn Behavior	58
4-5	Intel-Recommend Access Settings	58
5-1	Options for MEManuf	70
5-2	Intel® MEManuf Test Matrix	74
5-3	MEManuf - EOL Config Tests	103
6-1	Intel® MEInfo Command Line Options	108
6-2	List of Components that Intel® MEINFO Displays	110
7-1	Image File Update Options	129



Revision History

Document Number	Revision Number	Description	Date
	0.5	<ul style="list-style-type: none">Initial release	April 2016
	0.6	<ul style="list-style-type: none">Updated MEInfo output examples	October 2016
	0.61	<ul style="list-style-type: none">Updated FPT command line option information.	November 2016
	0.7	<ul style="list-style-type: none">Removed ISH Functionalities from MEInfo and MEManufRemoved NFC References	March 2017
	0.8	<ul style="list-style-type: none">Updated Build Settings ImageReplaced MEU tool usage with reference to Manifesting and Signing GuideSmall FixesUpdates to FIT sectionUpdates to MEInfo section	May 2017
	0.81	<ul style="list-style-type: none">Updates to Supported OS in various sectionsUpdates to FPT sectionMeManuf – BIST runs regardless of power sourceMeInfo –feat supports column nameUpdate to FWUpdLcl.exe requirements	June 2017
	0.9	<ul style="list-style-type: none">Updated OS TableRemoved redundant tool usage information	June 2017
	0.92	<ul style="list-style-type: none">Improved documentation for FPT -IN and -MASTERACCESSGENUpdated example for FPT -cfggenAdded details for SPI software binding (PCH replacement)	August 2017
	0.93	<ul style="list-style-type: none">Updated source for LZMA	September 2017
	1.0	<ul style="list-style-type: none">Updated error codes appendixUpdated MeInfo section	December 2017
	1.1	<ul style="list-style-type: none">Added -ALL command to MEManuf/MEManuf win tableAdded a note for PKI DNS Suffix to indicate dots location within the sting along with an example	January 2018
	1.2	<ul style="list-style-type: none">Add details for -ALL command under Chapter 5.3, "Usage"Added a note clarifying Privacy/Security Level Default Setting under Appendix A, "Intel® ME NVARs"Update Table 6-2, "List of Components that Intel® MEINFO Displays" with Touch relevant informationAdded a new section under Chapter 3, "Setting the Intel® PMC Binary File" with information about adding the Intel® PMC binary file.Added details about SVN ARB in relevant MEInfo, MEManuf, and FPT tools' sections	March 2018
	1.3	<ul style="list-style-type: none">Add new tool Chapter 8, "UEFI Sample Application Leveraging FWUpdate API Library"Updated Appendix with Appendix B.3 "FWUpdate API Library Errors"	May 2018

§ §



1 Introduction

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

1.1 Terminology

Acronym/Term	Definition
3PDS	3rd Party Data Storage
AC	Alternating Current
Agent	Software that runs on a client PC with OS running
AMT	Intel® AMT
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® ME power state where all HW power planes are activated. Host power state is S0.
CM1	Intel® ME power state where all HW power planes are activated but the host power state is different than S0. (Some host power planes are not activated.) The Host PCI-E* interface is unavailable to the host SW. This power state is not available in Cougar Point.
CM3	Intel® ME power state where all HW power planes are activated but the host power state is different than S0. (Some host power planes are not activated.) The Host PCI-E* interface is unavailable to the host SW. The main memory is not available for Intel® ME use.
CM-Off	No power is applied to the management processor subsystem. Intel® ME is shut down.
CRB	Customer Reference Board
DHCP	Dynamic Host Configuration Protocol



Acronym/Term	Definition
DIMM	Dual In-line Memory Module
DLL	Dynamic Link Library
DNS	Domain Naming System
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). In corporate, the user usually does not have administrator privileges. The end user may not be aware to the fact that the platform is managed by Intel® AMT.
EOP	End Of Post
FCIM	Full Clock Integrated Mode
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
FPT	Flash Programming Table
FQDN	Fully Qualified Domain Name
FRBA	Flash Region Base Address
FTP	Fault Tolerant Partition
Full Image	A full image starts with an FPT and contains FTP and NFTP partitions
Full Update	Updates all the regions
FW	Firmware
FWUpdate	Firmware Update
FWUpdateLib	Firmware Update Library
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



Acronym/Term	Definition
GUID	Globally Unique Identifier
HECI (deprecated)	Host Embedded Controller Interface
Host or Host CPU	The processor running the operating system. This is different than the management processor running the Intel® ME FW.
Host Service/ Application	An application running on the host CPU
HostIF	Host Interface
HTTP	Hyper Text Transfer Protocol
HW	Hardware
IBEN	Input Buffer Enable
IBV	Independent BIOS Vendor
ICC	Integrated Clock Configuration
ID	Identification
IDER	Integrated Drive Electronics Redirection
INF	An information file (.inf) used by Microsoft operating systems that support the Plug 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® AMT	The Intel® AMT Firmware running on the embedded processor
Intel® DAL	Intel® Dynamic Application Loader (Intel® DAL)
Intel® FIT	Intel® Flash Image Tool
Intel® FPT	Intel® Flash Programming Tool
Intel® ME	Intel® Management Engine. The embedded processor residing in the chipset PCH.
Intel® MEBx	Intel® Management Engine BIOS Extensions
Intel® MEI driver	Intel® AMT host driver that runs on the host and interfaces between ISV Agent and the Intel® AMT HW.
Intel® MEINFO	Intel® Manageability Engine Information Tool to check whether ME is alive or not.
Intel® MEInfoWin	Windows® version of Intel® Manageability Engine Information Tool
Intel® MEManuf	Intel® Manageability Engine Manufacturing Tool validates Intel® ME functionality on the manufacturing line
Intel® MEManufWin	Windows® version of Intel® Manageability Engine Manufacturing Tool
ISV	Independent Software Vendor



Acronym/Term	Definition
IT User	Information Technology User. Typically very technical and uses a management console to ensure multiple PCs on a network function.
JEDECID	Joint Electronic Device Engineering Councils ID. Standard Manufacturer's Identification Code that is assigned, maintained and updated by the JEDEC office
JTAG	Joint Test Action Group
KVM	Keyboard, Video, Mouse
LAN	Local Area Network
LED	Light Emitting Diode
LOCL	Localization Language
LMS	Local Management Service. An SW application which runs on the host machine and provides a secured communication between the ISV agent and the Intel® Management Engine Firmware.
LPC	Low Pin Count Bus
MAC address	Media Access Control address
MCP	Multi-Chip Package (Central Processing Unit / Platform Controller Hub)
NFTP	Non-Fault Tolerant Partition
NM	Number of Masters
NVAR	Named Variable
NVM	Non-Volatile Memory
NVRAM	Non-Volatile Random Access Memory
OCKEN	Output Clock Enable
ODM	Original Device Manufacturer
OEM	Original Equipment Manufacturer
OEM ID	Original Equipment Manufacturer Identification
OOB	Out Of Band
OOB interface	Out Of Band interface. An SOAP/XML interface over secure or non-secure TCP protocol.
OS	Operating System
OS Hibernate	OS state where the OS state is saved on the hard drive.
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.



Acronym/Term	Definition
OVR	Override
PAVP	Protected Video and Audio Path
Partial Image	A partial image starts with either WCOD or LOCL partitions. No FPT, FTO, and NFTP in the file
Partial Update	Only updates regions that require an Update such as WCOD or LOCL
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
S0	A system state where power is applied to all HW devices and the system is running normally.
S1, S2, S3	A system state where the host CPU is not running but power is connected to the memory system (memory is in self refresh).
S4	A system states where the host CPU and memory are not active.
S5	A system state where all power to the host system is off but the power cord is still connected.
SDK	Software Development Kit.
SEBP	Single Ended Buffer Parameters
SHA	Secure Hash Algorithm
SMB	Small Medium Business mode



Acronym/Term	Definition
SMBus	System Management Bus
Snooze mode	Intel® ME activities are mostly suspended to save power. Intel® ME monitors HW activities and can restore its activities depending on the HW event.
SOAP	Simple Object Access Protocol
SOL	Serial over LAN
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, S1, S2, S3, S4, 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® ME FW when it leaves the OEM factory. At this stage the Intel® ME FW is not functional and must be configured.
UNS	User Notification Services
UPDPARAM	Update Parameter Tool
USB	Universal Serial Bus
USBr	Universal Serial Bus Redirection
UUID	Universally Unique Identifier
VLAN	Virtual Local Area Network
VSCC	Vendor Specific Component Capabilities
WCOD	Wireless Card Device
Windows® PE	Windows® Pre installation Environment
WIP	Work in Progress
WLAN	Wireless Local Area Network



Acronym/Term	Definition
XML	<p>Extensible Markup Language. Intel® AMT's XML-based protocol has 3 parts:</p> <p>An envelope that defines a framework for describing what is in a message and how to process it.</p> <p>A set of encoding rules for expressing instances of application-defined data types.</p> <p>A convention for representing remote procedure calls and responses.</p>
ZTC	Zero Touch Configuration
ARB SVN	Anti Rollback Security Version Number



1.2 Reference Documents

Document	Document No./Location
FW Bring Up Guide	Included in released Kits
Firmware Variable Structures for Intel® Management Engine and Intel® Active Management Technology 12.0	CDI document
Cannon Lake PCH External Design Specification - EDS	CNL-H Volume 1: CDI# 571182 CNL-H Volume 2: CDI# 572235 CNL-LP Volume 1: CDI# 566439 CNL-LP Volume 2: CDI# 565870
Cannon Lake PCH-LP SPI Programming Guide	Included in released Kits

§ §

2 Preface

2.1 Overview

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

- Cannon LakePCH platform – All of the tools in the Cannon Lake PCH FW release kit are designed for 8th Generation Intel® Core™ Processors and Cannon Lake PCH platforms only. These tools do not work properly on any other legacy platforms (prior Generations of Intel® Core™ Processors). Tools designed for other platforms also do not work properly on the 8th Generation Intel® Core™ Processors or the Cannon Lake PCH platform.
- Intel® vPro™ platform – All features listed in this document are available for Intel® vPro™ platforms with Intel® ME FW 12.0. There are some features that are specifically designed for the Intel® vPro™ platform and only work on it.
- Intel® ME Firmware 12.0 SKU – A common set of tools are provided for the following Intel® ME FW 12.0 SKUs: Consumer Intel® ME FW SKU and Corporate Intel® ME FW SKU. The following features are only available for Corporate Intel® ME FW SKUs and Consumer Intel® ME FW SKU users should generally ignore them:
 - Intel® AMT
 - Intel® ME BIOS Extension (Intel® MEBx)

The description of each tool command or option that is not available for Consumer Intel® ME FW SKU contains a note indicating this.

- Note: For LBG, Non-POR features are WLAN and PTT.

2.2 Image Editing Tools

The following tools create and write flash images:

- Intel® FIT
Combines the Descriptor, GbE, BIOS, PDR, ISH and Intel® ME FW binaries into one image.
Configures soft straps and NVARs for Intel® ME settings and another for outputs



that can be programmed by a flash programming device or the FPT Tool.

- FPT:
Programs the SPI flash memory of individual regions or the entire flash device.
Modifies some Intel® ME settings (NVAR), FPFs after Intel® ME is flashed on the SPI part.
- FWUpdate – updates the Intel® ME 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 FWUpdate feature.

2.3 Manufacturing Line Validation Tool

The manufacturing line validation tool (Intel® MEmanuf) allows the Intel® ME and Intel® AMT 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.

2.4 Intel® Management Engine Setting Checker Tool

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



2.5 Operating System Support

Table 2-1. OS Support for Tools

Intel® ME and Manufacturing Tools	Free DOS	UEFI (64 bit)	Windows® 10 DT 64 bit	OSX® (El Capitan / Yosemite)	Windows PE for Windows 10	Ubuntu 16.04.3 LTS (64 Bit)
Intel® Flash Programing Tool	X	X	X		x	X
Intel® MEManuf Tool	X	X	X		x	X
Intel® ME Info Tool	X	X	X		x	X
Intel® Firmware Update Tool	X	X	X		x	X
Intel® Manifest Extension Utility Tool			X	x		
Intel® Flash Image Tool			X	x		
ICC CCT Tool	X	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. ISH is not supported on MEInfo/ MEManuf for Linux or UEFI. Also, a separate ISH tool must be used where functionalities are ported from MEInfo and MEManuf tool.
4. Currently the System Tools use the EDK II Development Kit exclusively.



2.6 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® Manageability Engine:

- Intel® MEI driver.
- Intel® AMT LMS – not applicable to Consumer Intel® ME FW SKU.

Refer the description of each tool for its exact requirements.

Table 2-2. Tools Summary

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

2.7 Error Return

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

For Intel® MEManuf tool, there is error level 2 which indicates Success with Warnings.

2.8 Usage of Double-Quote Character (")

The EFI version of the tools handle multi-word argument differently than the DOS/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 "" Wlan well power config "".

The command shell used to invoke the tools in EFI, DOS 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.

2.9 PMX Driver Limitation

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

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

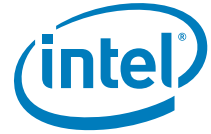
In Windows® 7 (and higher), the verifier sits in kernel mode, performing continual checks or making calls to selected driver APIs with simulations of well-known driver related issues.

Warning: Running the PMX driver with the Windows® 7 (and higher) driver verifier turned on causes the OS to crash. Do not include PMX as part of the verifier driver list if the user is running Windows® 7 (and higher) with the driver verifier turned on.

2.10 Control Handler Support

Intel® MEInfo and Intel® FPT and Intel® MEManuf 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 image file for Cannon Lake platforms in the following way:

1. 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
- Intel integrated LAN (GbE)
- IFWI: Intel® ME and PMC
- EC
- Platform Descriptor Region
- ISH

The Flash Descriptor Region created by FIT

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 any third-party flash burning tool, or some other flash burner device.

3.1 System Requirements

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

3.2 Flash Image Details

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



Figure 3-1. SPI Flash Image Regions

Descriptor	IFWI: Intel® ME and PMC Intel® ME Applications	EC	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® ME device vulnerable to security attacks.</p>
Ifwi: Intel® ME and PMC	This region contains code and configuration data for Intel® ME applications, such as Intel® AMT technology. It takes up a variable amount of space at the end of the Descriptor.
EC	This contains the Embedded Controller binary used for eSPI.
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® ME 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.

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® ME 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® ME FW kit.
- Detailed information on how to configure PCH Soft Straps and VSCC information – Refer to the Cannon Lake PCH SPI Programming Guide and to the C620 Lewisburg platforms refer LBG 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

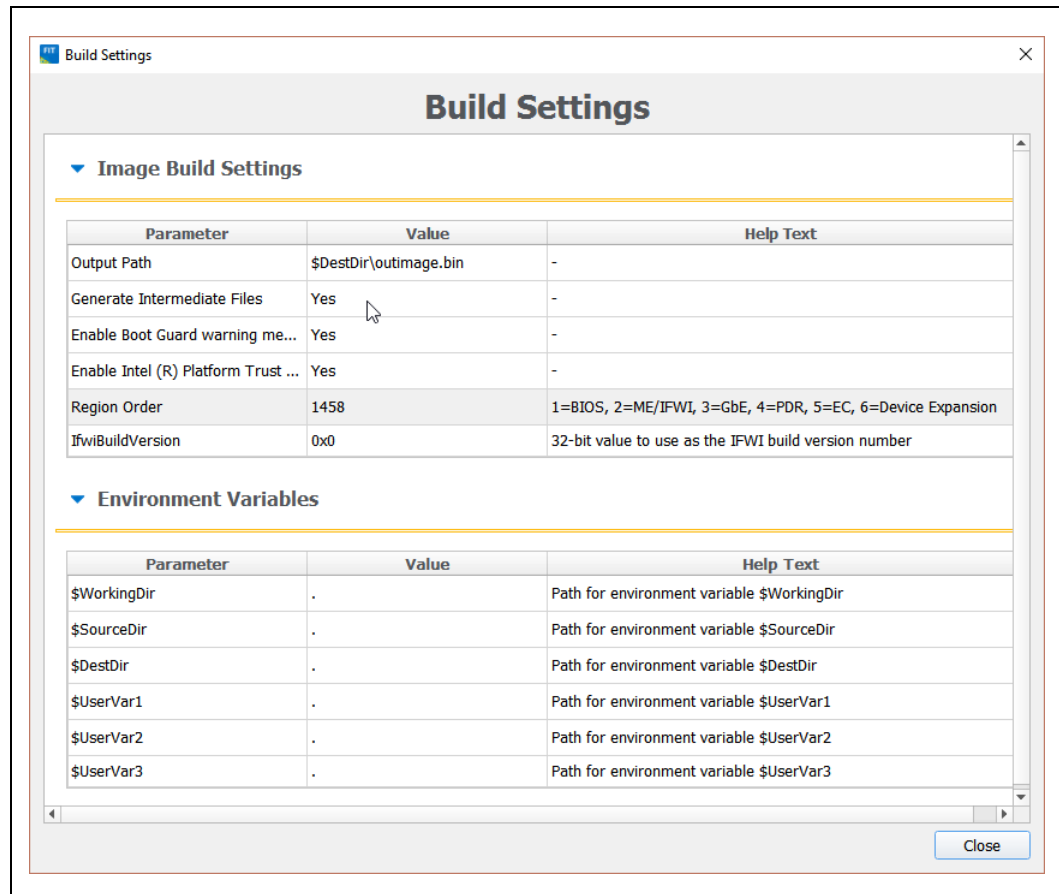
A set of environment variables is provided to make the image configuration files more portable. The configuration is not tied to a particular root directory structure because all of the paths in the configuration are relative to environment variables. The user can set the environment variables appropriate 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



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:

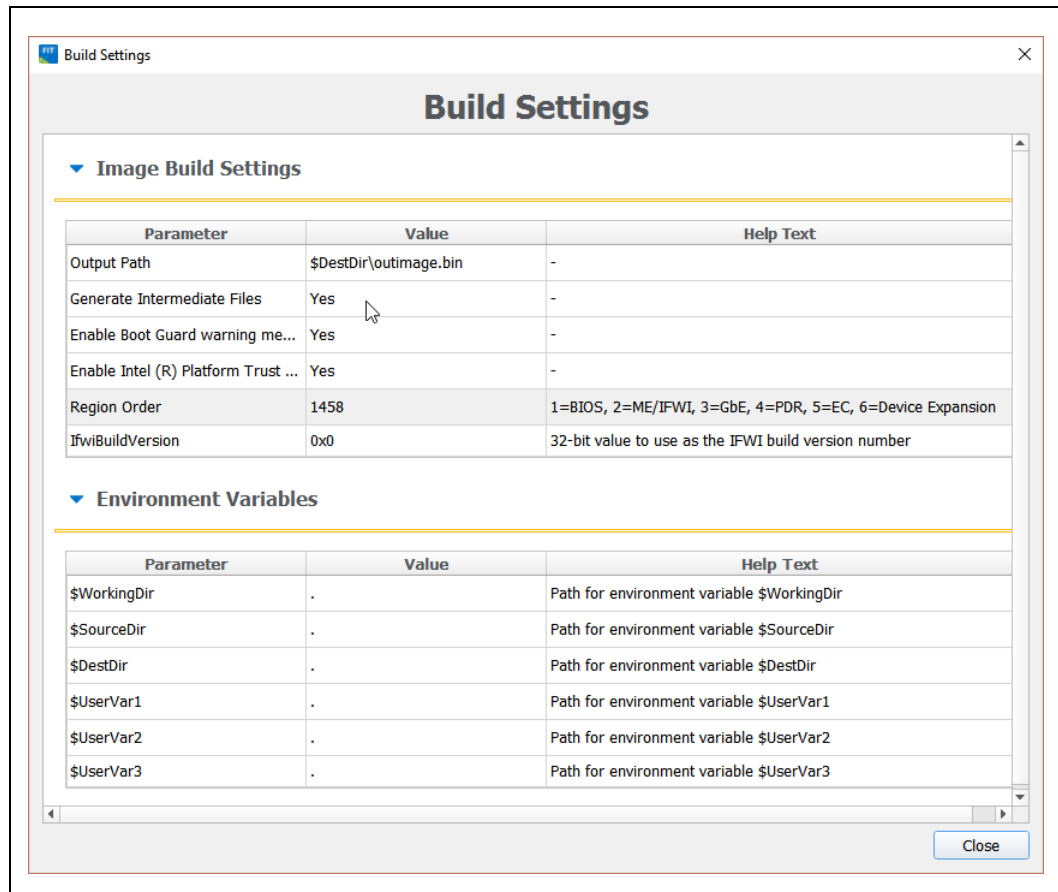


1. Choose **Build** → **Build Settings**; a dialog appears showing the current build settings.
2. Modify the relevant settings in the **Build Settings** dialog.
3. Click **OK**; the modified build settings are saved in the XML configuration file.

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.
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
CPU Stepping	Which CPU stepping to use.
Environment Variables	

Figure 3-3. 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

▼ Descriptor Region		
Parameter	Value	Help Text
Length	0	-

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 Component 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	1	Help Text
Number of Flash Components	2	Specifies the number of Flash components that will be installed on the target machine if usi...
Flash component 1 Size	16MB	This field identifies the size of the 1st Flash component.
Flash component 2 Size	8MB	This field identifies the size of the 2nd Flash component.
SPI Voltage Select	3.3 Volts	This strap sets the internal control signal on the pad for either 1.8 or 3.3 V operation. See ...
SPI Global Protected Range	0x0	Sets the default value of the Global Protected Range register in the SPI Flash Controller.
SPI Idle to Deep Power Down T...	0x5	SPI Idle to Deep Power Down Timeout Default Specifies the time in microseconds that the FL...
SPI Out of Order operation Ena...	Yes	When this setting is enabled priority operations may be issued while waiting for write / eras...
SPI Resume Hold-off Delay	4us	Specifies the time after the completion of a pri_op before the flash controller sends the resu...
SPI Max write / erase Resume ...	No Ceiling	This setting specifies the maximum value for the write and erase Resume to Suspend interv...
SPI Suspend / Resume Enabled	Yes	When this setting is enabled writes and erases may be suspended to allow a read to be issu...

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

To set the size of each flash component:

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

Figure 3-6. Flash Settings > Flash Configuration

▼ Flash Configuration		
Parameter	Value	
Dual I/O Read Enabled	No	-
Dual Output Fast Read Suppo...	No	Enables/Disables Fast Read support.
Dual Output Read Enabled	No	-
Fast Read clock frequency	17MHz	This field is undefined if the Fast Read Support is set to false.
Fast Read supported	No	false: Not Supported. true: Dual Output Fast Read instruction is is
Invalid Instruction 0	0x00000000	Op-code for an invalid instruction that the Flash Controller should
Invalid Instruction 1	0x00000000	Op-code for an invalid instruction that the Flash Controller should
Invalid Instruction 2	0x00000000	Op-code for an invalid instruction that the Flash Controller should
Invalid Instruction 3	0x00000000	Op-code for an invalid instruction that the Flash Controller should
Invalid Instruction 4	0x00000000	Op-code for an invalid instruction that the Flash Controller should
Invalid Instruction 5	0x00000000	Op-code for an invalid instruction that the Flash Controller should
Invalid Instruction 6	0x00000000	Op-code for an invalid instruction that the Flash Controller should
Invalid Instruction 7	0x00000000	Op-code for an invalid instruction that the Flash Controller should
Quad I/O Read Enabled	No	-
Quad Output Read Enabled	No	-
Read ID and Read Status clo...	17MHz	If more that one Flash component exists, this field must be the low
Write and Erase clock freque...	17MHz	If more that one Flash component exists, this field must be the low

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® ME devices are shipped. If the Descriptor Region is not locked, the Intel® ME device is vulnerable to security attacks. The level of read/write access provided is at the discretion of the OEM/ODM. A cross-reference of access settings is shown below.



Table 3-3. Region Access Control Table

Master Read/Write Access				
Region (#)	CPU and BIOS	ME/PCH	GbE Controller	EC
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
ME (2)	Read / Write	ME can always read from and write to ME region	Read / Write	Read / Write
GbE (3)	Read / Write	Read / Write	GbE software can always read from and write to GbE region	Read / Write
PDR (4)	Not Accessible	Not Accessible	Not Accessible	Not Accessible
EC - Embedded Controller (Optional) (8)	Read / Write	Read / Write	Read / Write	EC can always read from and write to EC region
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 ME should NOT have access to PDR region.				



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

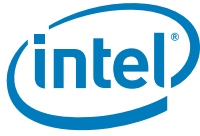
1 – Allowed access

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

Table 3-4. CPU/BIOS Access

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

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



Example:

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

Read Access – 0b 0000 1110 (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 0xFF for both read and write access in all the sections.

To generate a production image with BIOS access to the PDR region select
read access 0x00B / 0x01B 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® ME read access and 0x0D for Intel® ME write access).

Figure 3-7. Descriptor Region Master Access Section

▼ Host CPU / BIOS Master Access		
Parameter	Value	
Host CPU / BIOS Write ...	0xFFF	-
Host CPU / BIOS Read ...	0xFFF	-
▼ Intel(R) ME Master Access		
Parameter	Value	
Intel(R) ME Write Access	0xFFF	-
Intel(R) ME Read Access	0xFFF	-
▼ GbE Master Access		
Parameter	Value	
GbE Write Access	0xFFF	-
GbE Read Access	0xFFF	-

3.4.11 VSCC Table

This section is used to store information to setup flash access for Intel® ME. This does not have any effect on the usage of the FPT. **If the information in this section is incorrect, Intel® ME FW may not communicate with the flash device.** The information provided is dependent on the flash device used on the system. (For more information, refer to the Cannon Lake PCH-LPSPI Programming Guide, Section 6.4.) and For Lewisburg C620 family platform, refer LBG SPI Programming Guide, Section 4.4.)

VSCC Table can be accessed:

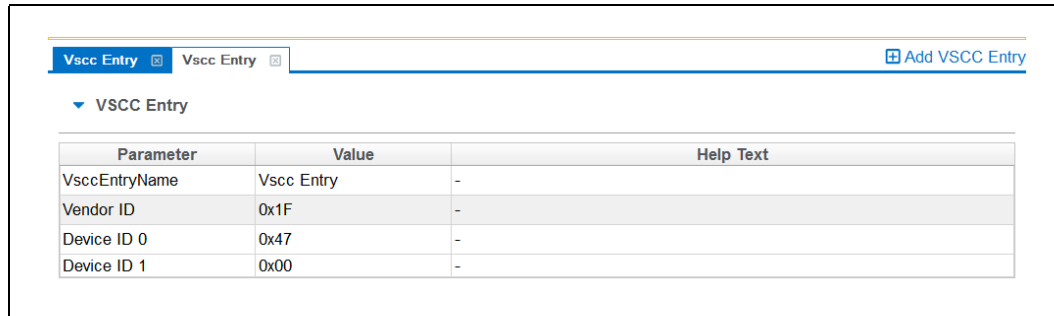
1. Select Flash Settings Tab on the left pan
2. Expand VSCC Entries on the right pan as shown below in [Figure 3-7](#):

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.

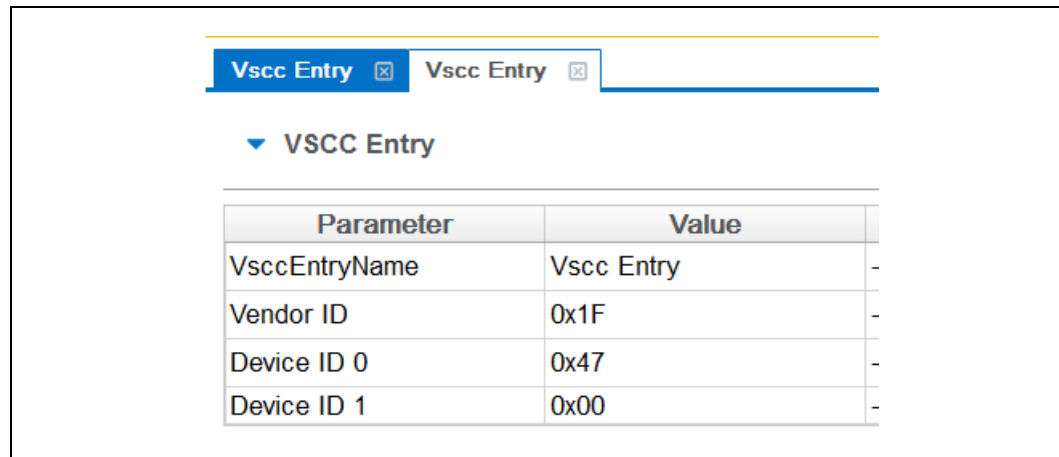
Note: If the descriptor region is being built manually the user will need to reference the VSCC table information for the parts being supported from the manufacturers' serial flash data sheet. The Cannon Lake PCH-LP SPI Programming Guide should be used to calculate the VSSC values. For C620 family of 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® Management Engine Region

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

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

3.4.16 Setting the Intel® Management Engine Region Binary File

To select the Intel® ME region binary file:

1. Select the Intel® ME and PMC Region available under Flash Layout tab on the left pane.



2. Double-click on the **Intel® ME Binary file parameter** in the list; select the Intel® ME 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® ME Region.

3.4.17 Setting the Intel® PMC Binary File

To select the Intel® PMC binary file:

1. Select the Intel® ME & PMC Region available under Flash Layout tab on the left pane.
2. Double-click on the **PMC Binary file parameter** in the list; select the Intel® PMC file to be used.
3. Click **OK** to update the parameter; when the flash image is built, the contents of this file will be merged into the output image generate by the Intel® FIT tool.

Note: Intel FIT tool would return a build error in case wrong PMC binary is selected for stitching.

3.4.18 Intel® Management Engine Section

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

Click on the Intel® ME Kernel Tab on the left pane to configure the Intel® ME parameters. The parameter values can be found in the Help Text next to the parameter value as shown in [Figure 3-9](#).

Figure 3-10. Intel® ME Kernel

▼ Processor

Parameter	Value	Help Text
Processor Emulation	No Emulation	-
ProcMissing	No onboard glue logic	-

▼ Intel (R) ME Firmware Update

Parameter	Value	Help Text
Firmware Update OEM ID	00000000-0000-0000-0...	-
Hide MEBx Firmware Update Control	No	-
Intel(R) ME Region Flash Protection Override	Yes	-

▼ Intel (R) Services Configuration

Parameter	Value	Help Text
ODM ID used by Intel(R) Services	0x00000000	-
System Integrator ID used by Intel(R) Services	0x00000000	-
Reserved ID used by Intel(R) Services	0x00000000	-

▼ Image Identification

Parameter	Value	Help Text
OEM Tag	0x00000000	-

▼ MCTP Configuration

Parameter	Value	Help Text
MCTP Stack Configurat...	0x920030	Defines the ME's 8-bits MCTP Endpoint IDs for each SMBus physical interface (...)

▼ Reserved

Parameter	Value	Help Text
Reserved	No	-

3.4.19 Power

This section describes the platform power configuration settings.

Click on the Power tab on the left pane to configure power parameters.
(Figure 3-10)

Figure 3-11. Power

▼ Platform Power		
Parameter	Value	Help Text
SLP_A# / GPD6 Signal ...	SLP_A#	-
SLP_S3# / GPD4 Signa...	SLP_S3#	-
SLP_S4# / GPD5 Signa...	SLP_S4#	-
SLP_S5# / GPD10 Sign...	SLP_S5#	-
USB_Wakeout# / GPD7...	USB_WAKEOUT#	-
APWROK Timing	2 ms	-
▼ Intel(R) ME Power Configuration		
Parameter	Value	Help Text
M3 Power Rail Available	No	-
▼ Deep Sx		
Parameter	Value	Help Text
Deep Sx Enabled	Yes	This requires the target platform to support Deep SX state

3.4.20 Manageability Application Section

Note: This section is not applicable to Consumer Intel® ME FW SKU.

This section describes the Manageability Application parameters. (Refer FW Bring up guide for general information.)

The Manageability section lets the user define the default Intel® AMT parameters. The values specified in this section are used after the Intel® AMT device is un-provisioned (full or partial). Click Intel® AMT Tab on the left tab to configure Intel® AMT parameters.



Figure 3-12. Manageability Application Section

Intel (R) AMT Configuration

Parameter	Value	Help Text
Intel(R) AMT initial power-up state	Enabled	-
Intel(R) AMT Supported	Yes	-
Intel(R) ME Network Services Supported	No	-
Intel(R) AMT Idle Timeout	0xFFFF	-
ManageAppPerm	No	-
DynAppLoad	No	-

KVM Configuration

Parameter	Value	Help Text
KVM Redirection Suppo...	Yes	-

Provisioning Configuration

Parameter	Value	Help Text
Embedded Host Based ...	No	-
PKI Domain Name Suffix		-

OEM Customizable Certificate 1

OEM Customizable Certificate 2

OEM Customizable Certificate 3

OEM Default Certificate 1

OEM Default Certificate 2

OEM Default Certificate 3

OEM Default Certificate 4

OEM Default Certificate 5

Redirection Configuration

Parameter	Value	Help Text
Redirection Privacy / S...	Default	-

TLS Configuration

Parameter	Value	Help Text
Transport Layer Securit...	Yes	-

3.4.21 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 3-13. Platform Protection Section

▶ Hash Key Configuration for Bootguard / ISH

▶ Boot Guard Configuration

▼ Intel (R) PTT Configuration

Parameter	Value	Help Text
Intel(R) PTT initial power-up st...	Enabled	This setting determines if Intel(R) PTT is enabled on platform power-up.
Intel(R) PTT Supported	Yes	This setting permanently disables Intel(R) PTT in the firmware image.
Intel(R) PTT Supported [FPF]	Yes	This setting will permanently disable Intel(R) PTT through platform FPFs. Caution: Using this opti...
Intel(R) PTT RTC Clear Detection	Enabled	This setting determines how the Intel(R) PTT will behave when RTC (CMOS) clear is triggered on ...

▼ TPM Over SPI Bus Configuration

Parameter	Value	Help Text
TPM Clock Frequency	17MHz	This setting determines the clock frequency setting to be used for the TPM over SPI bus.
TPM Over SPI Bus Enabled	No	This setting determines if TPM over SPI bus is enabled on the platform.

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

3.4.22 Provisioning Section

The Provisioning section allows the end user to specify the configuration settings, Intel® Upgrade Service, and Intel® DAL. (See the FW Bring up guide for general information).

Click the Intel® AMT tab on the left pane to specify the OEM settings.

Figure 3-14. Provisioning Configuration Section

▼ Provisioning Configuration

Parameter	Value	Help Text
Embedded Host Based Configuration Enabled	No	-
PKI Domain Name Suffix		-

▼ OEM Customizable Certificate 1

Parameter	Value	Help Text
Certificate Enabled	No	-
Certificate Friendly Name		Enter Hash Name. Maximum of 32 characters.
Certificate Stream		Enter raw hash string or certificate file.

▼ OEM Customizable Certificate 2

Parameter	Value	Help Text
Certificate Enabled	No	-
Certificate Friendly Name		Enter Hash Name. Maximum of 32 characters.
Certificate Stream		Enter raw hash string or certificate file.

▼ OEM Customizable Certificate 3

Parameter	Value	Help Text
Certificate Enabled	No	-
Certificate Friendly Name		Enter Hash Name. Maximum of 32 characters.
Certificate Stream		Enter raw hash string or certificate file.

▼ OEM Default Certificate 1

Parameter	Value	Help Text
Certificate Enabled	No	-
Certificate Friendly Name		Enter Hash Name. Maximum of 32 characters.
Certificate Stream		Enter raw hash string or certificate file.

▼ OEM Default Certificate 2

Parameter	Value	Help Text
Certificate Enabled	No	-
Certificate Friendly Name		Enter Hash Name. Maximum of 32 characters.
Certificate Stream		Enter raw hash string or certificate file.

▼ OEM Default Certificate 3

Parameter	Value	Help Text
Certificate Enabled	No	-
Certificate Friendly Name		Enter Hash Name. Maximum of 32 characters.
Certificate Stream		Enter raw hash string or certificate file.

▼ OEM Default Certificate 4

Parameter	Value	Help Text
Certificate Enabled	No	-
Certificate Friendly Name		Enter Hash Name. Maximum of 32 characters.
Certificate Stream		Enter raw hash string or certificate file.

Figure 3-15. Provisioning Configuration Section (Cont...)

▼ OEM Default Certificate 5

Parameter	Value	Help Text
Certificate Enabled	No	-
Certificate Friendly Name		Enter Hash Name. Maximum of 32 characters.
Certificate Stream		Enter raw hash string or certificate file.

3.4.23 Gbe (LAN) Region Settings

The Gbe Region contains various configuration parameters (e.g., the MAC address) for the embedded Ethernet controller.

Figure 3-16. GbE Region Options

▼ GbE Region

Parameter	Value	Help Text
Length	0	-
GbE Binary File	C:/Users/ratnameh/Downloads/...	-
GbE Region Enable	Disabled	-

3.4.24 Setting Gbe Region Length Option

The Gbe Region length option should not be altered. A value of 0x00000000 indicates that the Gbe Region will be auto-sized as described in [Section 3.2.1](#).

3.4.25 Setting Gbe Region Binary File

To select the Gbe Region binary file:

1. Click on Flash Layout tab on the left pane to load the binary file for Gbe region.
2. Select a file. When the flash image is built, the contents of this file are copied into the Gbe Region.

3.4.26 Enabling/Disabling GbE Region

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

To disable the GbE Region:

1. Click on Flash Layout tab on the left pane to load the binary file for Gbe region.
5. Choose **Disable Region** from the drop down. When the flash image is built it will not contain a GbE Region.

To enable the GbE Region:



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

3.4.27 Modifying PDR Region

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

Figure 3-17. PDR Region Options

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

3.4.28 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](#).

3.4.29 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.

3.4.30 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 Gbe 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 Gbe region
2. Choose **Enable Region** from the drop down menu.

3.4.31 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 3-18. BIOS Region Parameters

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

3.4.32 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](#).

3.4.33 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.

3.4.34 Enabling/Disabling the BIOS Region

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

To disable the BIOS Region:

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

To enable the BIOS Region:

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

3.4.35 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.

3.4.36 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 ME region binary contained in INT folder after image generation only contains the firmware default base settings for ME region no FIT customization is applied.

3.4.37 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 [/h] [/?][/b] [/o <file>] [/rombypass <true|false>] [/sku <value>]  
    [/me <file>] [/gbe <file>] [/bios <file>] [/pdr <file>] [/w <path>]  
    [/s <path>] [/d <path>] [/u1 <value>] [/u2 <value>] [/u3 <value>]  
    [/i <enable|disable>] [/flashcount <1|2>] [/flashsize1 <size>]  
    [/flashsize2 <size>] [/save <file>] [XML or BIN file]
```

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.
-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.
-O <file>	Path and filename where the image is saved. This command overrides the output file path in the XML file.
-ROMBYPASS	Overrides rombypass settings in the XML file.
-ME <file>	Overrides the binary source file for the Intel® ME Region with the specified binary file.
-GBE <file>	Overrides the binary source file for the GbE 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.
-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.



Option	Description
-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, 1 or 2>	Overrides the number of flash components in the Descriptor Region. If this value is zero, only the Intel® ME Region is built.
-FLASHSIZE1 <0, 1, 2, 3, 4 or 5>	Overrides the size of the first flash component with the size of the option selected as follows: 0 = 512KB 1 = 1MB 2 = 2MB 3 = 4MB 4 = 8MB 5 = 16MB.
-FLASHSIZE2 <0, 1, 2, 3, 4 or 5>	Overrides the size of the first flash component with the size of the option selected as follows: 0 = 512KB 1 = 1MB 2 = 2MB 3 = 4MB 4 = 8MB 5 = 16MB.
-Save <file>	Saves the XML file.
-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.

3.4.38 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, GBE, Intel® ME, and BIOS) and the Map file.

The xml file contains the current Intel® ME parameters.

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

3.4.39 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.

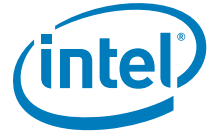
Take an existing (dt_ori.bin) image and put in a new BIOS binary:
FIT.exe /b /bios "..\..\..\Image Components\BIOS\BIOS.ROM" <file.bin or file.xml>

Take an existing image and put in a different Intel® ME region:
FIT.exe /b /me "..\..\..\Image Components\Firmware\ME12.0_5M_PreProduction.BIN" <file.bin or file.xml>

Note: The ME override option changes the ME 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 Components\GbE\NAHUM6_CLARKSVILLE_DESKTOP_11.bin" <file.bin or file.xml>





4 Flash Programming Tool

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

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.
- Provision HDCP
- Provided FPF's Access
- Helps doing Closemfn

Note: For proper function in a Multi-SPI configuration the Block Erase, Block Erase Command and Chip Erase must all match.

4.1 System Requirements

The DOS version of FPT (**fpt.exe**) runs on FreeDOS.

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.

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® ME/GbE image is written to flash.
4. The computer powers down.



5. The computer powers up, boots, and is able to access its Intel® ME/GbE 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**. The following files must be in the same directory as **fptw.exe**:

- fparts.txt – contains a comma-separated list of attributes for supported flash devices. The text in the file explains each field. An additional entry may be required in this file to describe the flash part which is on the target system. Examine the target board before adding the appropriate attribute values. The supplied file is already populated with default values for SPI devices used with Intel CRBs.
- fptw.exe – the executable used to program the final image file into the flash.
- pmxdll.dll
- 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
FPT.EXE	DOS	None
FPTw.EXE	Windows® 32 / 64 bit w/WOW64	idrvdll.dll, pmxdll.dll
FPTW64.EXE	Windows® Native 64 bit	idrvdll32e.dll, pmxdll32e.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**:

- fparts.txt – contains a comma-separated list of attributes for supported flash devices. The text in the file explains each field. An additional entry may be required



in this file to describe the flash part which is on the target system. Examine the target board before adding the appropriate attribute values. The supplied file is already populated with default values for SPI devices used with Intel CRBs.

- 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 DOS Required Files

The DOS version of the FPT main executable is **fpt.exe**. The following files must be in the same directory as **fpt.exe**:

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

4.6 Programming Flash Device

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

4.6.1 Stopping Intel® ME SPI Operations

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

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

- Assert HDA_SDO (known as GPIO 33 or Flash descriptor override/Intel® ME 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 HMRFP0 ENABLE Intel® MEI command to Intel® ME (for more information refer PCH Intel® ME BIOS writer's guide).

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

4.7 Programming NVARS

FPT can program the NVARS and change the default values of the parameters. The modified parameters are used by the Intel® ME FW after a global reset (Intel® ME +



HOST reset) or upon returning from a G3 state. NVARS can be programmed using getfile/setfile/CommitFiles APIs.

SetFile API will allow for the host to change the values in UEP (Unified Emulation Partition). Note: Intel® ME Firmware does NOT require commit File after a UEP SetFile. 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® or DOS 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® or DOS environments, it is recommended to use the Windows® or DOS 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).
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 setfile.
fpt.exe -U -N <NVAR name>	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.

4.7.1 Programming GPIO NVAR

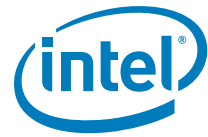
FPT tool will support configuring the GPIO via string inputted by the user on command line. The string inputted should be in defined format which FPT tool will parse and turn into a binary.

In this method, customer will specify the string which includes configuration data required by the GPIO NVAR (Feature ID, Usage, Owner and Attributes).

Format of command line will look like:

FPT -u CSE_GPIO GPIO [(FID, Usage, Owner, Attributes),...].

Each GPIO entry will include the FID, Usage, Owner, Attributes



4.8 Usage

The EFI, DOS and Windows® versions of the FPT can run with command line options.

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

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

```
FPT.exe [-H|?] [-VER] [-EXP] [-VERBOSE] [-Y] [-P] [-LIST] [-I]
        [-F] [-ERASE] [-VERIFY] [-NOVERIFY] [-D] [-DESC] [-BIOS]
        [-ME] [-GBE] [-PDR] [-EC] [-SAVEMAC] [-SAVESXID] [-B] [-E]
        [-REWRITE] [-ADDRESS|A] [-LENGTH|L] [-CVARS] [-MASTERACCESSGEN]
        [-CFGGEN] [-U] [-CLEAR] [-O] [-IN] [-N] [-V] [-CLOSEMNF] [-GRESET]
        [-PAGE] [-SPIBAR] [-R] [-VARS] [-COMMIT] [-HASHED] [-DISABLEME]
        [-COMPAREFPF] [-FPFS] [-COMMITFPF] [-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.
-VER	Shows the version of the tools.
-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".
-P <file>	Flash parts file. Specifies the alternate flash definition file which contains the flash parts description that FPT has to read. By default, FPT reads the flash parts definitions from fparts.txt.
-LIST	Supported Flash Parts. Displays all supported flash parts. This option reads the contents of the flash parts definition file and displays the contents on the screen.
-I	Info. Displays information about the image currently used in the flash.



Option	Description
-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 specifying 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.
-ERASE	Block Erase. Erases all the blocks in a flash. This option does not use the chip erase command but instead erases the SPI flash block by block. This option can be used with a specific region argument to erase that region. This option cannot be used with the -f, -b, -c, -d or -verify options.
-VERIFY <file>	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® ME region. Specifies that the Intel® ME region is to be read, written, or verified. The start address is the beginning of the region.
-EC	Read/Write EC region. Specifies that the EC region is to be read, written, or verified. The start address is the beginning of the region.
-GBE	Read/Write GbE region. Specifies that the GbE region is to be read, written, or verified. The start address is the beginning of the region.
-PDR	Read/Write PDR region. Specifies that the PDR region is to be read, written, or verified. The start address is the beginning of the region.



Option	Description
-SAVEMAC	This is used to save the GbE MAC Address. It is appropriate only when GbE Firmware is being over written. It also saves the GbE SSID and SVID.
-SAVESXID	Saves the GbE SSID and SVID when GbE is being reflashed.
-B	Blank Check. Checks whether the SPI flash is erased. If the SPI flash is not empty, the application halts as soon as contents are detected. The tool reports the address at which data was found.
-E	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.
-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.
-U	Update. Updates variables in the UEP. The user can update the multiple FOVs 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>



Option	Description
-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 (-v).
-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.
-CLOSEMNF <NO> <PDR>	<p>End of Manufacturing. This option is executed at the end of manufacturing phase. This option does the following:</p> <p>Sets the Intel® ME manufacturing mode done bit (Global Locked bit).</p> <p>Verifies that the Intel® ME 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 ME Manufacturing Mode Done, the Region Access permissions, or both.</p> <p>The "PDR" addition will allow CPU\BIOS Read and Write access to the PDR region of flash.</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® ME 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 <NO>	Global Reset. FPT performs a global reset.



Option	Description
-PAGE	Pauses the screen when a page of text has been reached. Hit any key to continue.
-SPIBAR	Display SPI BAR. FPT uses this option to display the SPI Base Address Register.
-R <name>	NVAR 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 <code>-hashed</code> option can be used to display the hash of a value instead of the clear-text value.
-VARS	Display Supported Variables. FPT uses this option to display all variables supported for the <code>-R</code> and <code>-COMPARE</code> commands. Note: This will no longer display UEP based values which are tied to configuring FPFs.
-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® ME does not reset and the tool displays the status of the commit operation.
-HASHED	Hash Variable Output. FPT uses this option to distinguish whether the displayed output is hashed by the FW. For variables that can only be returned in hashed form (such as the Intel® MEBx password), this option has no effect – the data displayed is hashed regardless.
-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.
-COMMITARBSVN	Commits ARB SVN to FPFs. This would commit the Anti Rollback SVN to the FPFs

Table 4-4. FPT-closemfnf Behavior

Condition before FPT - closemfnf			Condition after FPT -closemfnf			Other FPT Action	
Intel ME Mfg Done bit set	Flash Access set to Intel rec values	Intel ME Mfg Mode	Intel ME Mfg Done bit set	Flash Access set to Intel rec values?	Intel ME 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 -closemfnf returns 1 (= error) because it is unable to set the Intel ME Mfg Done bit, because flash permissions are already set to Intel recommended values (host cannot access Intel ME Region).

Table 4-5. Intel-Recommend Access Settings

	Intel® ME	GbE	BIOS
Read	0b 0000 1101 = 0x0d	0b 0000 1000 = 0x08	0b 0000 0011 = 0x0B 0b 0001 1011 = 0x1B – BIOS access to PDR
Write	0b 0000 1100 = 0x0c	0b 0000 1000 = 0x08	0b 0000 0010 = 0x0A 0b 0001 1010 = 0x1A – BIOS access to PDR

4.9 Updating Hash Certificate through NVAR

Note: This section is not applicable for Consumer Intel® ME FW SKU.

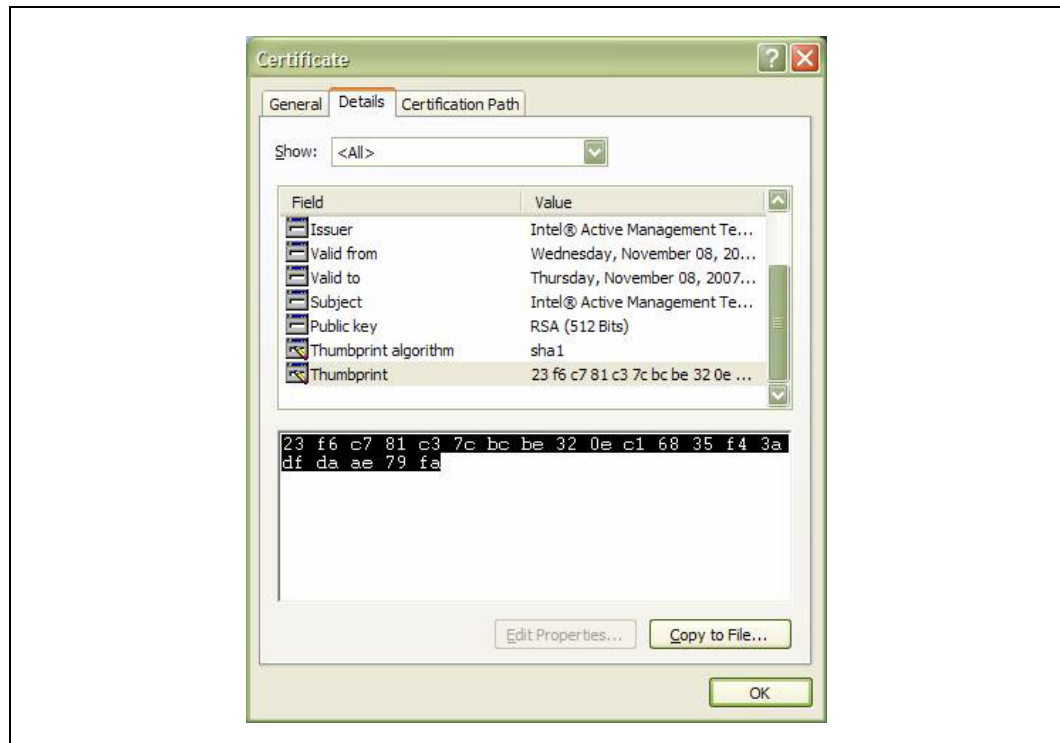
There are 3 OEM Customizable certificate hash values that can be stored in the Intel® ME region:

- The OEM Customizable Certificates 1-3 are not default certificates and are deleted after a full un-provisioning.
- The OEM Customizable Certificates 1-3 are configurable by NVAR (with FPT or other flash programming methods) or FIT.

To store certificate hash values in the Intel® ME region:

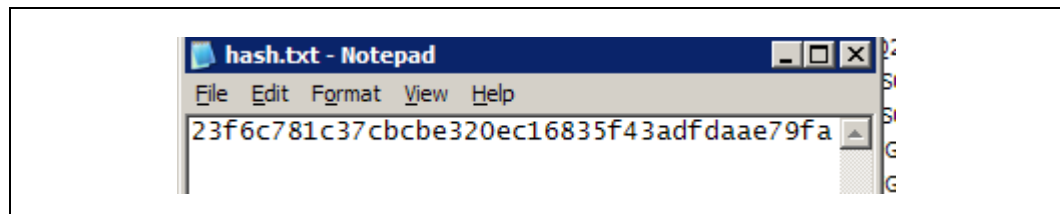
1. Copy the raw hash values from a valid certificate file.

Figure 4-1. Raw Hash Values from Certificate File



2. Paste the raw hash values into a text file
3. Remove all the spaces from the text file.

Figure 4-2. Sample Hash.txt File



4. Save the text file as **hash.txt**.
5. Copy and paste the text saved from hash.txt and add it to **FPT.CFG file** in order to update the NVAR:

EXAMPLE:

```
; OEMCustomCert1 Certificate
; All data is required to update the certificate.
; See the Tools Users Guide for detailed explanation
; of required data and format.
OEMCustomCert1 IsActive      = 0x01
OEMCustomCert1 FriendlyName  = MyCert
OEMCustomCert1 RawHashFile   = 23f6c781c37cbcb320ec16835f43adfdade79fa
```



6. Flash Hash NVAR with FPT's `-u -in` option (e.g., `fpt -u -in fpt.cfg`).

Note: **FTP.CFG** is the file that is used to update NVAR

4.10 Fparts.txt File

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

```
Intel (R) Flash Programming Tool. Version:  x.x.x.xxxx
Copyright (c) 2007-2014, Intel Corporation. All rights reserved.
Platform: Intel(R) Qxx Express Chipset
Error 75: "fparts.txt" file not found.
```

If the device is not located in **fparts.txt**, the user is expected to provide information about the device, inserting the values into **fparts.txt** in same format as is used for the rest of the devices. Detailed information on how to derive the values in **fparts.txt** is found in the Cannon LakePCH-LP SPI Programming Guide. The device must have a **4KB erase sector** and the total size of the SPI Flash device must be a multiple of 4KB. The values are listed in columns in the following order:

- Display name
- Device ID (2 or 3 bytes)
- Device Size (in bits)
- Block Erase Size (in bytes - 256, 4K, 64K)
- Block Erase Command
- Write Granularity (1 or 64)
- Unused

4.11 Examples

The following examples illustrate the usage of the EFI and DOS 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.

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



4. BIOS MENU -> INTEL ADVANCED -> PCH I/O CONFIGURATION -> Flash Protection Range: Disabled..

In order to use FPT Tool with Lewisburg C620 series, the following BIOS settings are recommended (to avoid errors when running fpt.exe -f spi.bin):

1. EDKII Menu → Platform Configuration → PCH Configuration → Security Configuration → SMM BIOS Write Protect = Disabled
2. EDKII Menu
3. → Platform Configuration → PCH Configuration → PCH DFX Configuration → Show SPI device = Enable
4. → Platform Configuration → PCH Configuration → PCH DFX Configuration → BIOS Lock = Disable
5. EDKII Menu
6. → Platform Configuration → Miscellaneous Configuration → BIOS Guard = unchecked
7. EDKII Menu
8. → Platform Configuration → Server ME Configuration → Manageability Application Configuration → Manageability State = Enable
9. EDKII Menu
10. → Platform Configuration → PCH Configuration → PCH Devices → Dirty Warm Reset = Disable

```
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.

4.11.2 Program Specific Region

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

EFI:

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

```
-----
Intel (R) Flash Programming Tool. Version:  x.x.x.xxxx
Copyright (c) 2007-2014, Intel Corporation. All rights reserved.
Platform: Intel(R) Qxx Express Chipset
Reading HSFSTS register... Flash Descriptor: Valid
--- Flash Devices Found ---
   W25Q64BV   ID:0xEF4017   Size: 8192KB (65536Kb)
- Erasing Flash Block [0x800000]... - 100% complete.
- Programming Flash [0x800000]2560KB or 2560KB - 100% complete.
- Verifying Flash [0x800000]2560KB or 2560KB - 100% complete.
```



RESULT: The Data is identical.
FPT Operation Passed

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

4.11.3 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.

4.11.4 Dump Full Image

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

EFI:

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

```
-----  
Intel (R) Flash Programming Tool. Version:  x.x.x.xxxx  
Copyright (c) 2007-2014, Intel Corporation. All rights reserved.  
Platform: Intel(R) Qxx Express Chipset  
Reading HSFSTS register... Flash Descriptor: Valid  
--- Flash Devices Found ---  
    W25Q64BV    ID:0xEF4017    Size: 8192KB (65536Kb)  
- Reading Flash [0x00800000]... 8192KB of 8192KB - 100% complete.  
Writing flash contents to file "imagedump.bin"...  
Memory Dump Complete  
FPT Operation Passed
```

4.11.5 Dump Specific Region

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

EFI:

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

```
-----  
Intel (R) Flash Programming Tool. Version:  x.x.x.xxxx  
Copyright (c) 2007-2014, Intel Corporation. All rights reserved.  
Platform: Intel(R) Qxx Express Chipset  
Reading HSFSTS register... Flash Descriptor: Valid  
--- Flash Devices Found ---  
    W25Q64BV    ID:0xEF4017    Size: 8192KB (65536Kb)  
- Reading Flash [0x000040]... 4KB of 4KB - 100% complete.  
Writing flash contents to file "descdump.bin"...  
Memory Dump Complete
```



FPT Operation Passed

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

4.11.6 Display SPI Information

```
fptw.exe -I
```

```
-----
Intel (R) Flash Programming Tool. Version: XX.X.X.XXXX
Copyright (c) 2007 - 2017, Intel Corporation. All rights reserved.
```

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

```
--- Flash Devices Found ---
W25Q256FVID:0xEF4019Size: 32768KB (262144Kb)
```

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

```
--- Flash Image Information --
Signature: VALID
Number of Flash Components: 1
  Component 1 - 32768KB (262144Kb)
Regions:
  DESC    - Base: 0x00000000, Limit: 0x00000FFF
  BIOS    - Base: 0x01183000, Limit: 0x01B82FFF
  CSME    - Base: 0x00083000, Limit: 0x01082FFF
  GbE     - Base: 0x00081000, Limit: 0x00082FFF
  PDR     - Not present
  EC      - Base: 0x00001000, Limit: 0x00080FFF
Master Region Access:
  BIOS    - ID: Read: 0xFFFF, Write: 0xFFFF
  CSME    - ID: Read: 0xFFFF, Write: 0xFFFF
  GbE     - ID: Read: 0xFFFF, Write: 0xFFFF
  EC      - ID: Read: 0xFFFF, Write: 0xFFFF
```

Total Accessable SPI Memory: 28172KB, Total Installed SPI Memory : 32768KB

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".

4.11.7 Verify Image with Errors

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

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

```
-----
```



```
Intel(R) Flash Programming Tool. Version: x.x.x.xxxx
Copyright (c) 2007-2014, Intel Corporation. All rights reserved.
Platform: Intel(R) Qxx Express Chipset
Reading HSFSTS register... Flash Descriptor: Valid
--- Flash Devices Found ---
    W25Q64BV    ID:0xEF4017    Size: 8192KB (65536Kb)
RESULT: Data does not match!
[0x00000000] Expected 0x5A, Found: 0x5A
[0x00000001] Expected 0xA5, Found: 0xA5
Total mismatches found in 64 byte block: 2
Error 204: Data verify mismatch found at address 0x000
```

This command compares the Intel® ME 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.

4.11.8 Verify Image Successfully

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

EFI:

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

```
-----
Intel (R) Flash Programming Tool. Version: x.x.x.xxxx
Copyright (c) 2007-2014, Intel Corporation. All rights reserved.
Platform: Intel(R) Qxx Express Chipset
Reading HSFSTS register... Flash Descriptor: Valid
--- Flash Devices Found ---
    W25Q64BV    ID:0xEF4017    Size: 8192KB (65536Kb)
-Verifying Flash [0x800000] 8192KB of 8192KB - 100% complete.
RESULT: The data is identical.
FPT Operation Passed
```

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® ME changes some data in the flash after a reset.

4.11.9 Get Intel® ME settings

```
fpt.exe -r "Privacy/SecurityLevel"
fpt.efi -r "^"Privacy/SecurityLevel"^"
```

```
-----
Intel (R) Flash Programming Tool. Version: x.x.x.xxxx
Copyright (c) 2007-2014, Intel Corporation. All rights reserved.
Platform: Intel(R) Qxx Express Chipset
Reading HSFSTS register... Flash Descriptor: Valid
--- Flash Devices Found ---
    W25Q64BV    ID:0xEF4017    Size: 8192KB (65536Kb)
Variable: "Privacy/SecurityLevel"
```




Value: True / 01

Retrieve Operation: Successful

Note: Only -r (get command) supports the -hashed optional command argument. When -hashed is used, variable value will be returned in hashed format, otherwise it will be returned in clear txt. There are a few exceptions in the case of variables MEBxPassword, PID and PPS, their value will be always returned in hashed format regardless -hashed is used or not. This is primarily because of security concern.

4.11.10 CVAR Configuration File Generation (-cfggen)

It creates an input file which can be used to update CVARs. The file includes all the current CVAR. 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:

```
;
;   Flash Programming Tool FOV Programming File
;
;   Any entry that is not included, or does not have a value
;   following the label will not be updated.
;
;   Comments can be added by using a ';' as the first entry
;   on the line.
;
;   For further explanation of the required inputs see the
;   System Tools User Guide.doc
;
;   Any entries, FOVs, that are displayed with values
;   indicates that the FOV has already been given a value,
;   but has not yet been committed. Entries without values
;   indicates that the FOV has not been written, at least
;   since the system reset or use of the '-commit' command.
```



66



```
CfgSrvAdr =

CfgSrvPort = 0x26F3

Privacy/SecurityLevel = 0x01

IdleTO = 0xFFFF

ScreenBlankingEn = 0x00

AmtWdAutoReset = 0x00

;   PkiDns NVAR value is not displayed because it is stored encrypted.
PkiDns =

EhbcState = 0x00

;   MEBxPassword NVAR value is not displayed because it is stored
encrypted.
MEBxPassword =

;   ODM_ID NVAR value is not displayed because it is stored encrypted.
ODM_ID =

;   SystemIntegratorID NVAR value is not displayed because it is stored
encrypted.
SystemIntegratorID =

;   ReservedID NVAR value is not displayed because it is stored encrypted.
ReservedID =

Intel(R) AMT Supported = 0x01

Manageability Application Supported = 0x01

Transport Layer Security Supported = 0x01

iTouch = 0x00

PTTEnable = 0x00

URTC = 0x00

SetWLANPowerWell = 0x86

OEM_TAG = 0x00000000

FWUpdLcl = 0x01

PTT = 0x01

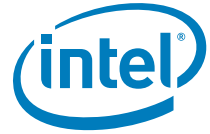
ENF0 = 0x00

ENF1 = 0x00
```



```
OEM_DID =  
  
OEM_PID =  
  
NCC = 0x00  
  
TxtSupp = 0x00  
  
BootGuard = 0x0040  
  
CPU Debugging = 0x00  
  
BSP Initialization = 0x00  
  
Protect BIOS Environment Enabled = 0x00  
  
Measured Boot Enabled = 0x00  
  
Verified Boot Enabled = 0x00  
  
Key Manifest ID = 0x01  
  
Force Boot Guard ACM Enabled = 0x00  
  
S3 Optimization Disabled = 0x00  
  
; OEM_CRD NVAR value is not displayed because it is stored encrypted.  
OEM_CRD =
```

§ §



5 Intel® MEmanuf and MEmanufWin

Intel® MEmanuf validates Intel® ME functionality on the manufacturing line. It does not check for LAN functionality as it assumes that all Intel® ME 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® MEmanuf 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® ME 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® MEmanuf

Intel® MEmanuf checks the FW SKU and runs the proper tests accordingly unless an option to select tests is specified. If Intel® AMT is enabled on the platform; it automatically causes a reboot to test system hardware connections when the system is in sleep state.

Intel® MEmanuf 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® ME enabled platform, MEmanuf calls the test. Otherwise, it reports the result or the failure message from the previous test.

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

VSCCMMN.bin is required to verify the VSCC entry on the platform. This file must be in same folder as the MEmanuf executable or MEmanuf reports an error.



5.3 Usage

The DOS version of the tool can be operated using the same syntax as the Windows® version. The Windows® version of the tool can be executed by:

```
MManuf[-EXP] [-H|?] [-VER] [-BLOCKNET] [-ALLOWNET]
        [-TEST] [-S0] [-BISTRESULT] [-NEXTREBOOT] [-EOL]
        [-CFGGEN] [-F] [-VERBOSE] [-PAGE] [-ERRLIST] [-ALL]
        [-NOWLAN] [-WLAN] [-NOGFX] [-GFX] [-NOLAN] [-LAN]
```

Tool might returning following values for BIST to indicate either SUCCESS/ ERROR/ SUCCESS WITH WARNING.

0 means SUCCESS

1 means ERROR

2 means SUCCESS (With Warnings)

Table 5-1. Options for MEmanuf

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® ME Boot: The first time running Intel® MEmanuf, since there is no CM3 test result stored in SPI, the tool will request the FW to run a complete BIST which includes a power reset at the end of the test for the DOS version and a Hibernation for the Windows® version. This power reset is only host side power cycle that triggered by Intel® ME. When host resets, Intel® ME 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® ME 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 running on a Consumer SKU image, the tool will request the FW to run a complete BIST which does not involve any power transition at the end of the test. Test result will be reported back right after the test is done and cleared.</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.



Option	Description
-H or -?	Displays the help screen. Note: Use -H for help when running in the EFI Shell.
-VER	Shows the version of the tools.
-S0	The same as No option, except that there is no power reset/hibernation performed at the end of the BIST test including Intel® AMT SKU. The test result is reported back right after the test is done and cleared.
-F <filename>	Load customer defined .cfg file
-TEST	Run full test
-NOWLAN	Note: This option is not applicable for Consumer Intel® ME FW SKU. This option only applies to the AMT test so that the user can skip the wireless LAN NIC test if there is no wireless LAN NIC attached to the hardware. When -nowlan switch is not used, Intel® MEmanuf also checks for the HW presence of Intel WLAN card based on a pre-defined list. If Intel® MEmanuf detects an Intel WLAN card present on the platform, Intel® MEmanuf runs the WLAN BIST test and reports pass/fail accordingly. If Intel® MEmanuf cannot find any known WLAN card, Intel® MEmanuf skips the WLAN BIST test and does not report errors. With the -verbose option, it displays "No Intel wireless LAN card detected" Note: -S0 can only be used on the platform which Intel® AMT is present and can be enabled in the field.
-WLAN	Force wireless LAN test
-BLOCKNET	Note: This option is not applicable for Consumer Intel® ME FW SKU. This option blocks any network traffic that goes in/out of the integrated GbE wired/wireless LAN interface. If Intel® AMT is disabled, "Error 9257: Cannot run the command since Intel® AMT is not available" is returned.
-ALLOWNET	Note: This option is not applicable for Consumer Intel® ME FW SKU. This option allows any network traffic that goes in/out of the integrated GbE wired/wireless LAN interface. If Intel® AMT is disabled, "Error 9257: Cannot run the command since Intel® AMT is not available" is returned.
-BISTRESULT	Returns last BIST results.
-ERRLIST <test name>	Return a list of available codes.



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, BIOS Version, and Gbe 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 ME12.0 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® MEmanuf 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® MEmanuf 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® ME state check, Intel® MEmanuf verifies Intel® ME 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® ME is in abnormal state, Intel® MEmanuf will report error without running BIST test.</p> <p>ICC data check, Intel® MEmanuf 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).</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>
-ALL	<p>Use this option to generate all possible tests for configuration file.</p> <p>All BIST, EOLConfig, and EOLVAR types of tests will be included in the generated XML.</p> <p>Note: Intel recommended tests will be enabled regardless of <code>-all</code> parameter to meet corresponding dependencies</p>
-VERBOSE <file>	<p>Displays the debug information of the tool or stores it in a log file.</p>



Option	Description
-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.
-NOGFX	This option will skip KVM related test.
-GFX	This option will force KVM related test.
-NOLAN	Note: This option is not applicable for Consumer Intel® ME FW SKU. This option only applies to the Intel® AMT test so that the user can skip the wired LAN NIC test if there is no wired LAN NIC attached to the hardware. Note: -S0 can only be used on the platform which Intel® AMT is present and can be enabled in the field.
-LAN	This option will force LAN test

Note: The KVM test will be skipped if the platform being tested contains both internal and external GFX and BIOS has disabled internal GFX.



Table 5-2. Intel® MEmanuf Test Matrix

		CM3 Supported SKU	Consumer SKU
BIST Disabled in the ME BOOT	No option	-1st time: Run full BIST test (with ME 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 (with no reset)
	-Test	-Run full BIST test with Intel ME 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 ME 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 ME 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: ICC data check is performed for all options.

Note: The Full BIST test for ME12.0 is a combination of M0_HW, Live_HW and M0_Config. The Runtime BIST is a combination of M0_HW and M0_Config.

Intel® MEmanuf 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. ME/BIOS VSCC validation, Intel® MEmanuf 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® ME state check, Intel® MEmanuf verifies Intel® ME 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® ME is in abnormal state, Intel® MEmanuf will report error without running BIST test.
3. ICC data check, Intel® MEmanuf 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 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 MEmanuf.xml File

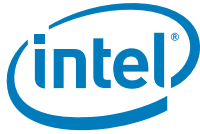
The MEmanuf.xml file includes all the test configurations for MEmanuf -EOL check. It needs to be at the same folder that MEmanuf is run. If there is no MEmanuf.xml file on that folder, MEmanuf -EOL config runs the Intel recommended default check only.

Note: Only MAC address, Wireless MAC address and System UUID tests allow the user to set the ReqVal option.

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="VDM - General : VDM engine">
    <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
    <!-- Description>Test VDM.</Description -->
    <!-- IntelRequired>True</IntelRequired -->
    <!-- Dependencies></Dependencies -->
```



```
<!-- TestType>M0_HW</TestType -->
<!-- End of uneditable fields -->
<!-- edit the fields below ONLY with the State or ErrAction -->
<State>Enabled</State>
<ErrAction>ErrorContinue</ErrAction>
</csmebist>
<csmebist name="GFX - General : Sampling engine">
  <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
  <!-- Description>Test KVM sampling engine.</Description -->
  <!-- IntelRequired>True</IntelRequired -->
  <!-- Dependencies></Dependencies -->
  <!-- TestType>M0_HW</TestType -->
  <!-- End of uneditable fields -->
  <!-- edit the fields below ONLY with the State or ErrAction -->
  <State>Enabled</State>
  <ErrAction>ErrorContinue</ErrAction>
</csmebist>
<csmebist name="SMBus - SMBus : Read byte">
  <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
  <!-- Description>Read one byte from SmBus ICH device (offset 0x44), if fails,
read DIMM0 (offset 0xA0 >> 1), if fails, read DIMM1 (0xA2 >> 1) and so on (0xA4 >> 1,
0xA6 >> 1, 0xA8 >> 1, 0xAA >> 1). Test fails if all trials failed.</Description -->
  <!-- IntelRequired>True</IntelRequired -->
  <!-- Dependencies></Dependencies -->
  <!-- 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 Password : Validate MEBx password">
  <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
  <!-- Description>Verify password is acceptable.</Description -->
  <!-- IntelRequired>True</IntelRequired -->
  <!-- Dependencies></Dependencies -->
  <!-- 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="Policy Kernel - Boot Guard : Self Test">
  <!-- The commented fields bellow 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 -->
  <!-- Dependencies></Dependencies -->
  <!-- TestType>M0_HW</TestType -->
  <!-- End of uneditable fields -->
  <!-- Please edit the fields below ONLY with the State or ErrAction -->
  <State>Enabled</State>
```



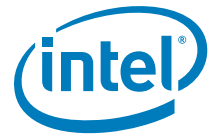
```

        <ErrAction>ErrorContinue</ErrAction>
    </csmebist>
    <csmebist name="Policy Kernel - ME Configuration : M3 Power Rails Available">
        <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
        <!-- Description>Only on mobile or desktop. Test fails if M3 power well rule
is not set to MEFWCAPS_M3_PWR_RAILS_AVAILABL.</Description -->
        <!-- IntelRequired>True</IntelRequired -->
        <!-- Dependencies></Dependencies -->
        <!-- 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="Policy Kernel - ME Configuration : PROC_MISSING">
        <!-- The commented fields bellow 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 -->
        <!-- Dependencies></Dependencies -->
        <!-- 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="Policy Kernel - ME Configuration : Wlan Power Well">
        <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
        <!-- Description>WLAN power well setting.</Description -->
        <!-- IntelRequired>True</IntelRequired -->
        <!-- Dependencies></Dependencies -->
        <!-- 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="Policy Kernel - Power Package : Live Heap Test">
        <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
        <!-- Description>Allocate memory in live heap in M0, write in M3, read back in
M0.</Description -->
        <!-- IntelRequired>True</IntelRequired -->
        <!-- Dependencies></Dependencies -->
        <!-- TestType>LIVE_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 - Embedded Controller : Power source type">
        <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored

```



```
by the tool -->
    <!-- Description>Only on mobile, if power source is DC, test fails.</
Description -->
    <!-- IntelRequired>True</IntelRequired -->
    <!-- Dependencies></Dependencies -->
    <!-- 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="USBr - General : Storage">
    <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
    <!-- Description>Test USBr Storage.</Description -->
    <!-- IntelRequired>True</IntelRequired -->
    <!-- Dependencies></Dependencies -->
    <!-- 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="USBr - General : KVM">
    <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
    <!-- Description>Test USBr KVM.</Description -->
    <!-- IntelRequired>True</IntelRequired -->
    <!-- Dependencies></Dependencies -->
    <!-- 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="Common Services - LAN : Connectivity to NIC in M3">
    <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
    <!-- Description>LAN test runs only if AMT is not permanantly disabled and we
are not in small business mode or mDNSProxy is not disabled.</Description -->
    <!-- IntelRequired>True</IntelRequired -->
    <!-- Dependencies>LAN</Dependencies -->
    <!-- TestType>LIVE_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="Common Services - LAN : Connectivity to NIC in M0">
    <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
    <!-- Description>LAN test runs only if AMT is not permanantly disabled and we
are not in small business mode or mDNSProxy is not disabled.</Description -->
    <!-- IntelRequired>True</IntelRequired -->
    <!-- Dependencies>LAN</Dependencies -->
```



```

    <!-- 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="Common Services - LAN : Connectivity to NIC in M3">
    <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
    <!-- Description>LAN test runs only if AMT is not permanently disabled and we
are not in small business mode or mDNSProxy is not disabled.</Description -->
    <!-- IntelRequired>True</IntelRequired -->
    <!-- Dependencies>LAN</Dependencies -->
    <!-- TestType>LIVE_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="Common Services - LAN : Connectivity to NIC in M0">
    <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
    <!-- Description>LAN test runs only if AMT is not permanently disabled and we
are not in small business mode or mDNSProxy is not disabled.</Description -->
    <!-- IntelRequired>True</IntelRequired -->
    <!-- Dependencies>LAN</Dependencies -->
    <!-- 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="Common Services - EHBC State : EHBC and Privacy Level states
compatibility">
    <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
    <!-- Description>Check while both EHBC and privacy level are available,
(PrivLevel != PRIVACY_LEVEL_DEFAULT) && (EHBCState == EHBC_STATE_ENABLE).</
Description -->
    <!-- IntelRequired>True</IntelRequired -->
    <!-- Dependencies></Dependencies -->
    <!-- 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="Common Services - EHBC State : Valid Embedded Host Based
Configuration (EHBC) state">
    <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
    <!-- Description>Check if EHBC state is available.</Description -->
    <!-- IntelRequired>True</IntelRequired -->
    <!-- Dependencies></Dependencies -->
    <!-- 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="Common Services - Privacy Level : Valid Privacy Level settings">
  <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
  <!-- Description>Check if privacy level is available.</Description -->
  <!-- IntelRequired>True</IntelRequired -->
  <!-- Dependencies></Dependencies -->
  <!-- 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="Common Services - General : Valid FOV number %d">
  <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
  <!-- Description>Checks if there were any issues when FOV's were copied into
system.</Description -->
  <!-- IntelRequired>True</IntelRequired -->
  <!-- Dependencies></Dependencies -->
  <!-- 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="AMT - Power : M3 power rail supported">
  <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
  <!-- Description>Run the tests verifying the internal variables.</Description
-->
  <!-- IntelRequired>True</IntelRequired -->
  <!-- Dependencies></Dependencies -->
  <!-- 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="AMT - Power : Valid LAN power well">
  <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
  <!-- Description>Run the tests verifying the internal variables.</Description
-->
  <!-- IntelRequired>True</IntelRequired -->
  <!-- Dependencies>LAN</Dependencies -->
  <!-- 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>
<!-- END OF CSME BIST TESTS -->
<!-- EOL CONFIG TESTS -->
<eolconfig name="GuC Encryption Key FPF">
  <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
  <!-- Description>Check GuC Encryption Key against expected value</Description
-->
  <!-- IntelRequired>False</IntelRequired -->
  <!-- Dependencies></Dependencies -->
  <!-- End of uneditable fields -->
  <!-- Please edit the fields below ONLY with the State or ErrAction -->
  <State>Disabled</State>
  <ErrAction>ErrorContinue</ErrAction>
  <RequiredValue format="32 hex pairs"
example="04ABF345031DEFA2B7E898791045ABDEF23549A00135782937ABDEEFFA10EF33"> </
RequiredValue>
</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 -->
  <!-- End of uneditable fields -->
  <!-- Please edit the fields below ONLY with the State or ErrAction -->
  <State>Enabled</State>
  <ErrAction>ErrorContinue</ErrAction>
</eolconfig>
<eolconfig name="BSMM SVN FPF">
  <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
  <!-- Description>Check fpf bsmm svn against expected value</Description -->
  <!-- IntelRequired>False</IntelRequired -->
  <!-- Dependencies></Dependencies -->
  <!-- End of uneditable fields -->
  <!-- Please edit the fields below ONLY with the State or ErrAction -->
  <State>Disabled</State>
  <ErrAction>ErrorContinue</ErrAction>
  <RequiredValue format="Not set/2 digit hex number with 0x prefix"
example="0xB4"> </RequiredValue>
</eolconfig>
<eolconfig name="KM SVN FPF">
  <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
  <!-- Description>Check fpf km svn against expected value</Description -->
  <!-- IntelRequired>False</IntelRequired -->
  <!-- Dependencies></Dependencies -->
  <!-- End of uneditable fields -->
  <!-- Please edit the fields below ONLY with the State or ErrAction -->
  <State>Disabled</State>
  <ErrAction>ErrorContinue</ErrAction>
  <RequiredValue format="Not set/2 digit hex number with 0x prefix" example="Not
set"> </RequiredValue>

```



```
</eolconfig>
<eolconfig name="ACM SVN FPF">
  <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
  <!-- Description>Check fpf acm against expected value</Description -->
  <!-- IntelRequired>False</IntelRequired -->
  <!-- Dependencies></Dependencies -->
  <!-- End of uneditable fields -->
  <!-- Please edit the fields below ONLY with the State or ErrAction -->
  <State>Disabled</State>
  <ErrAction>ErrorContinue</ErrAction>
  <RequiredValue format="Not set/2 digit hex number with 0x prefix"
example="0xB4"> </RequiredValue>
</eolconfig>
<eolconfig name="Enforcement Policy FPF">
  <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
  <!-- Description>Check fpf enf against expected value</Description -->
  <!-- IntelRequired>False</IntelRequired -->
  <!-- Dependencies></Dependencies -->
  <!-- End of uneditable fields -->
  <!-- Please edit the fields below ONLY with the State or ErrAction -->
  <State>Disabled</State>
  <ErrAction>ErrorContinue</ErrAction>
  <RequiredValue format="2 digit hex number with 0x prefix" example="0x4A"> </
RequiredValue>
</eolconfig>
<eolconfig name="BSP Initialization FPF">
  <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
  <!-- Description>Check fpf bsp against expected value</Description -->
  <!-- IntelRequired>False</IntelRequired -->
  <!-- Dependencies></Dependencies -->
  <!-- End of uneditable fields -->
  <!-- Please edit the fields below ONLY with the State or ErrAction -->
  <State>Disabled</State>
  <ErrAction>ErrorContinue</ErrAction>
  <RequiredValue format="Enabled/Disabled" example="Disabled"> </RequiredValue>
</eolconfig>
<eolconfig name="CPU Debugging FPF">
  <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
  <!-- Description>Check fpf cpu debug against expected value</Description -->
  <!-- IntelRequired>False</IntelRequired -->
  <!-- Dependencies></Dependencies -->
  <!-- End of uneditable fields -->
  <!-- Please edit the fields below ONLY with the State or ErrAction -->
  <State>Disabled</State>
  <ErrAction>ErrorContinue</ErrAction>
  <RequiredValue format="Enabled/Disabled" example="Enabled"> </RequiredValue>
</eolconfig>
<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>Disabled</State>
        <ErrAction>ErrorContinue</ErrAction>
        <RequiredValue format="Not set/Enabled/Disabled" example="Not set"> </
RequiredValue>
    </eolconfig>
    <eolconfig name="Key Manifest ID FPF">
        <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
        <!-- Description>Check kmid against expected value</Description -->
        <!-- IntelRequired>False</IntelRequired -->
        <!-- Dependencies></Dependencies -->
        <!-- End of uneditable fields -->
        <!-- Please edit the fields below ONLY with the State or ErrAction -->
        <State>Disabled</State>
        <ErrAction>ErrorContinue</ErrAction>
        <RequiredValue format="2 digit hex number with 0x prefix" example="0xA4"> </
RequiredValue>
    </eolconfig>
    <eolconfig name="Verified Boot FPF">
        <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
        <!-- Description>Check fpf verified boot against expected value</Description
-->
        <!-- IntelRequired>False</IntelRequired -->
        <!-- Dependencies></Dependencies -->
        <!-- End of uneditable fields -->
        <!-- Please edit the fields below ONLY with the State or ErrAction -->
        <State>Disabled</State>
        <ErrAction>ErrorContinue</ErrAction>
        <RequiredValue format="Enabled/Disabled" example="Disabled"> </RequiredValue>
    </eolconfig>
    <eolconfig name="Measured Boot FPF">
        <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
        <!-- Description>Check fpf measure boot against expected value</Description -
->
        <!-- IntelRequired>False</IntelRequired -->
        <!-- Dependencies></Dependencies -->
        <!-- End of uneditable fields -->
        <!-- Please edit the fields below ONLY with the State or ErrAction -->
        <State>Disabled</State>
        <ErrAction>ErrorContinue</ErrAction>
        <RequiredValue format="Enabled/Disabled" example="Enabled"> </RequiredValue>
    </eolconfig>
    <eolconfig name="Protect BIOS Environment FPF">
        <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
        <!-- Description>Check fpf protect bios env against expected value</
Description -->
        <!-- IntelRequired>False</IntelRequired -->
        <!-- Dependencies></Dependencies -->

```



```
<!-- End of uneditable fields -->
<!-- Please edit the fields below ONLY with the State or ErrAction -->
<State>Disabled</State>
<ErrAction>ErrorContinue</ErrAction>
<RequiredValue format="Enabled/Disabled" example="Disabled"> </RequiredValue>
</eolconfig>
<eolconfig name="Force Boot Guard ACM FPF">
  <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
  <!-- Description>Check fpf force boot against expected value</Description -->
  <!-- IntelRequired>False</IntelRequired -->
  <!-- Dependencies></Dependencies -->
  <!-- End of uneditable fields -->
  <!-- Please edit the fields below ONLY with the State or ErrAction -->
  <State>Disabled</State>
  <ErrAction>ErrorContinue</ErrAction>
  <RequiredValue format="Enabled/Disabled" example="Enabled"> </RequiredValue>
</eolconfig>
<eolconfig name="OEM Public Key Hash FPF">
  <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
  <!-- Description>Check fpf oem key hash against expected value</Description -
->
  <!-- IntelRequired>False</IntelRequired -->
  <!-- Dependencies></Dependencies -->
  <!-- End of uneditable fields -->
  <!-- Please edit the fields below ONLY with the State or ErrAction -->
  <State>Disabled</State>
  <ErrAction>ErrorContinue</ErrAction>
  <RequiredValue format="32 hex pairs"
example="04ABF345031DEFA2B7E898791045ABDEF23549A00135782937ABDEEFA10EF33"> </
RequiredValue>
</eolconfig>
<eolconfig name="Wireless LAN micro-code mismatch">
  <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
  <!-- Description>Check ucode WLAN against programmed ucode</Description -->
  <!-- IntelRequired>False</IntelRequired -->
  <!-- Dependencies>VPRO|WLAN|CORP</Dependencies -->
  <!-- End of uneditable fields -->
  <!-- Please edit the fields below ONLY with the State or ErrAction -->
  <State>Disabled</State>
  <ErrAction>ErrorContinue</ErrAction>
  <RequiredValue format="Yes/No -OR- 1/0" example="1"> </RequiredValue>
</eolconfig>
<eolconfig name="GBE version">
  <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
  <!-- Description>Check Gbe Version against expected value</Description -->
  <!-- IntelRequired>False</IntelRequired -->
  <!-- Dependencies>LAN</Dependencies -->
  <!-- End of uneditable fields -->
  <!-- Please edit the fields below ONLY with the State or ErrAction -->
  <State>Disabled</State>
  <ErrAction>ErrorContinue</ErrAction>
```



```

        <RequiredValue format="major_ver.minor_ver" example="0.6"> </RequiredValue>
    </eolconfig>
    <eolconfig name="BIOS version">
        <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
        <!-- Description>Check BIOS Version against expected value</Description -->
        <!-- IntelRequired>False</IntelRequired -->
        <!-- Dependencies></Dependencies -->
        <!-- End of uneditable fields -->
        <!-- Please edit the fields below ONLY with the State or ErrAction -->
        <State>Disabled</State>
        <ErrAction>ErrorContinue</ErrAction>
        <RequiredValue format="Customer specific"
example="H5WLPTU1.86C.0117.R00.1303102001"> </RequiredValue>
    </eolconfig>
    <eolconfig name="ME FW version">
        <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
        <!-- Description>Check Firmware Version against expected value</Description -
->
        <!-- IntelRequired>False</IntelRequired -->
        <!-- Dependencies></Dependencies -->
        <!-- End of uneditable fields -->
        <!-- Please edit the fields below ONLY with the State or ErrAction -->
        <State>Disabled</State>
        <ErrAction>ErrorContinue</ErrAction>
        <RequiredValue format="major_ver.minor_ver.hotfix_ver.build_num H | LP | ULT"
example="12.0.0.xxxx LP"> </RequiredValue>
    </eolconfig>

    <eolconfig name="System UUID">
        <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
        <!-- Description>Check System UUID against programmed value</Description -->
        <!-- IntelRequired>True</IntelRequired -->
        <!-- Dependencies>VPRO</Dependencies -->
        <!-- End of uneditable fields -->
        <!-- Please edit the fields below ONLY with the State or ErrAction -->
        <State>Enabled</State>
        <ErrAction>ErrorContinue</ErrAction>
        <RequiredValue format="See example" example="550e8400-e29b-41d4-a716-
446655440000"> </RequiredValue>
    </eolconfig>
    <eolconfig name="Wireless MAC address">
        <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
        <!-- Description>Check Wireless MAC address</Description -->
        <!-- IntelRequired>True</IntelRequired -->
        <!-- Dependencies>VPRO|WLAN</Dependencies -->
        <!-- End of uneditable fields -->
        <!-- Please edit the fields below ONLY with the State or ErrAction -->
        <State>Disabled</State>
        <ErrAction>ErrorContinue</ErrAction>
        <RequiredValue format="6 hex pairs separated by ':'"
example="00:01:12:A2:3B:45"> </RequiredValue>

```



```
</eolconfig>
<eolconfig name="MAC address">
  <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
  <!-- Description>Check MAC address</Description -->
  <!-- IntelRequired>True</IntelRequired -->
  <!-- Dependencies>VPRO</Dependencies -->
  <!-- End of uneditable fields -->
  <!-- Please edit the fields below ONLY with the State or ErrAction -->
  <State>Enabled</State>
  <ErrAction>ErrorContinue</ErrAction>
  <RequiredValue format="6 hex pairs separated by ':'"
example="00:01:12:A2:3B:45"> </RequiredValue>
</eolconfig>
<eolconfig name="CF9GR lock check">
  <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
  <!-- Description>Check CF9CR lock register</Description -->
  <!-- IntelRequired>True</IntelRequired -->
  <!-- Dependencies></Dependencies -->
  <!-- End of uneditable fields -->
  <!-- Please edit the fields below ONLY with the State or ErrAction -->
  <State>Enabled</State>
  <ErrAction>ErrorContinue</ErrAction>
</eolconfig>
<eolconfig name="Security Descriptor Override (SDO) check">
  <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
  <!-- Description>Check SDO pin</Description -->
  <!-- IntelRequired>True</IntelRequired -->
  <!-- Dependencies></Dependencies -->
  <!-- End of uneditable fields -->
  <!-- Please edit the fields below ONLY with the State or ErrAction -->
  <State>Enabled</State>
  <ErrAction>ErrorContinue</ErrAction>
</eolconfig>
<eolconfig name="Flash Region Access Permissions">
  <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
  <!-- Description>Check flash access</Description -->
  <!-- IntelRequired>True</IntelRequired -->
  <!-- Dependencies></Dependencies -->
  <!-- End of uneditable fields -->
  <!-- Please edit the fields below ONLY with the State or ErrAction -->
  <State>Enabled</State>
  <ErrAction>ErrorContinue</ErrAction>
</eolconfig>
<eolconfig name="ME Manufacturing Mode status">
  <!-- The commented fields bellow 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 -->
  <!-- End of uneditable fields -->
```



```

        <!-- Please edit the fields below ONLY with the State or ErrAction -->
        <State>Enabled</State>
        <ErrAction>ErrorContinue</ErrAction>
    </eolconfig>
    <eolconfig name="BIOS VSCC check">
        <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
        <!-- Description>Check programmed BIOS VSCC against Intel recommended value</
Description -->
        <!-- IntelRequired>True</IntelRequired -->
        <!-- Dependencies></Dependencies -->
        <!-- End of uneditable fields -->
        <!-- Please edit the fields below ONLY with the State or ErrAction -->
        <State>Enabled</State>
        <ErrAction>ErrorContinue</ErrAction>
    </eolconfig>
    <eolconfig name="ME VSCC check">
        <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
        <!-- Description>Check programmed ME VSCC against Intel recommended value</
Description -->
        <!-- IntelRequired>True</IntelRequired -->
        <!-- Dependencies></Dependencies -->
        <!-- 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 bellow CANNOT be edited. Any edits will be ignored
by the tool -->
        <!-- Description>Check that EOP was sent/recieved</Description -->
        <!-- IntelRequired>True</IntelRequired -->
        <!-- Dependencies></Dependencies -->
        <!-- 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="GuC Encryption Key">
        <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
        <!-- Description>Test variable against expected value</Description -->
        <!-- IntelRequired>False</IntelRequired -->
        <!-- Dependencies></Dependencies -->
        <!-- End of uneditable fields -->
        <!-- Please edit the fields below ONLY with the State or ErrAction -->
        <State>Disabled</State>
        <ErrAction>ErrorContinue</ErrAction>
        <RequiredValue format="32 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"> </RequiredValue>
    </eolvar>

```



```
<eolvar name="BSP Initialization">
  <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
  <!-- Description>Test variable against expected value</Description -->
  <!-- IntelRequired>False</IntelRequired -->
  <!-- Dependencies></Dependencies -->
  <!-- End of uneditable fields -->
  <!-- Please edit the fields below ONLY with the State or ErrAction -->
  <State>Disabled</State>
  <ErrAction>ErrorContinue</ErrAction>
  <RequiredValue format="False/True/00/01" example="False"> </RequiredValue>
</eolvar>
<eolvar name="CPU Debugging">
  <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
  <!-- Description>Test variable against expected value</Description -->
  <!-- IntelRequired>False</IntelRequired -->
  <!-- Dependencies></Dependencies -->
  <!-- End of uneditable fields -->
  <!-- Please edit the fields below ONLY with the State or ErrAction -->
  <State>Disabled</State>
  <ErrAction>ErrorContinue</ErrAction>
  <RequiredValue format="False/True/00/01" example="False"> </RequiredValue>
</eolvar>
<eolvar name="Boot Guard Profile Configuration">
  <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
  <!-- Description>Test variable against expected value</Description -->
  <!-- IntelRequired>False</IntelRequired -->
  <!-- Dependencies></Dependencies -->
  <!-- End of uneditable fields -->
  <!-- Please edit the fields below ONLY with the State or ErrAction -->
  <State>Disabled</State>
  <ErrAction>ErrorContinue</ErrAction>
  <RequiredValue format="No_FVME/VE/VME/VM/FVE/FVME" example="No_FVME"> </
RequiredValue>
</eolvar>
<eolvar name="Key Manifest ID">
  <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
  <!-- Description>Test variable against expected value</Description -->
  <!-- IntelRequired>False</IntelRequired -->
  <!-- Dependencies></Dependencies -->
  <!-- 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="0x0"> </RequiredValue>
</eolvar>
<eolvar name="OEM Public Key Hash">
  <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
  <!-- Description>Test variable against expected value</Description -->
  <!-- IntelRequired>False</IntelRequired -->
  <!-- Dependencies></Dependencies -->
```




```

    <!-- End of uneditable fields -->
    <!-- Please edit the fields below ONLY with the State or ErrAction -->
    <State>Disabled</State>
    <ErrAction>ErrorContinue</ErrAction>
    <RequiredValue format="32 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"> </RequiredValue>
  </eolvar>
  <eolvar name="Embedded Host Based Configuration Enabled">
    <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
    <!-- Description>Test variable against expected value</Description -->
    <!-- IntelRequired>False</IntelRequired -->
    <!-- Dependencies>CORP</Dependencies -->
    <!-- End of uneditable fields -->
    <!-- Please edit the fields below ONLY with the State or ErrAction -->
    <State>Disabled</State>
    <ErrAction>ErrorContinue</ErrAction>
    <RequiredValue format="Enabled/Disabled/00/01" example="Enabled"> </
RequiredValue>
  </eolvar>
  <eolvar name="PKI Domain Name Suffix">
    <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
    <!-- Description>Test variable against expected value</Description -->
    <!-- IntelRequired>False</IntelRequired -->
    <!-- Dependencies>CORP</Dependencies -->
    <!-- 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="0x0"> </RequiredValue>
  </eolvar>
  <eolvar name="MCTP PCIe Enabled">
    <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
    <!-- Description>Test variable against expected value</Description -->
    <!-- IntelRequired>False</IntelRequired -->
    <!-- Dependencies></Dependencies -->
    <!-- End of uneditable fields -->
    <!-- Please edit the fields below ONLY with the State or ErrAction -->
    <State>Disabled</State>
    <ErrAction>ErrorContinue</ErrAction>
    <RequiredValue format="False/True/00/01" example="False"> </RequiredValue>
  </eolvar>
  <eolvar name="MCTP eSPI Enabled">
    <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
    <!-- Description>Test variable against expected value</Description -->
    <!-- IntelRequired>False</IntelRequired -->
    <!-- Dependencies></Dependencies -->
    <!-- End of uneditable fields -->
    <!-- Please edit the fields below ONLY with the State or ErrAction -->
    <State>Disabled</State>
    <ErrAction>ErrorContinue</ErrAction>

```



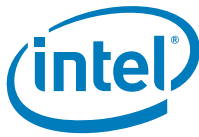
```
<RequiredValue format="False/True/00/01" example="False"> </RequiredValue>
</eolvar>
<eolvar name="MCTP Device Ports">
  <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
  <!-- Description>Test variable against expected value</Description -->
  <!-- IntelRequired>False</IntelRequired -->
  <!-- Dependencies></Dependencies -->
  <!-- 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="0x0"> </RequiredValue>
</eolvar>
<eolvar name="Reserved ID used by Intel (R) Service">
  <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
  <!-- Description>Test variable against expected value</Description -->
  <!-- IntelRequired>False</IntelRequired -->
  <!-- Dependencies></Dependencies -->
  <!-- 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="0x0"> </RequiredValue>
</eolvar>
<eolvar name="System Integrator ID used by Intel (R) Service">
  <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
  <!-- Description>Test variable against expected value</Description -->
  <!-- IntelRequired>False</IntelRequired -->
  <!-- Dependencies></Dependencies -->
  <!-- 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="0x0"> </RequiredValue>
</eolvar>
<eolvar name="ODM ID used by Intel (R) Service">
  <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
  <!-- Description>Test variable against expected value</Description -->
  <!-- IntelRequired>False</IntelRequired -->
  <!-- Dependencies></Dependencies -->
  <!-- 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="0x0"> </RequiredValue>
</eolvar>
<eolvar name="Intel(R) PTT initial power-up state">
  <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
  <!-- Description>Test variable against expected value</Description -->
```



```

    <!-- IntelRequired>False</IntelRequired -->
    <!-- Dependencies></Dependencies -->
    <!-- End of uneditable fields -->
    <!-- Please edit the fields below ONLY with the State or ErrAction -->
    <State>Disabled</State>
    <ErrAction>ErrorContinue</ErrAction>
    <RequiredValue format="Enabled/Disabled/00/01" example="Enabled"> </
RequiredValue>
  </eolvar>
  <eolvar name="Intel(R) AMT initial power-up state">
    <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
    <!-- Description>Test variable against expected value</Description -->
    <!-- IntelRequired>False</IntelRequired -->
    <!-- Dependencies></Dependencies -->
    <!-- End of uneditable fields -->
    <!-- Please edit the fields below ONLY with the State or ErrAction -->
    <State>Disabled</State>
    <ErrAction>ErrorContinue</ErrAction>
    <RequiredValue format="Enabled/Disabled/00/01" example="Enabled"> </
RequiredValue>
  </eolvar>
  <eolvar name="Intel(R) ME Network Services Supported">
    <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
    <!-- Description>Test variable against expected value</Description -->
    <!-- IntelRequired>False</IntelRequired -->
    <!-- Dependencies></Dependencies -->
    <!-- End of uneditable fields -->
    <!-- Please edit the fields below ONLY with the State or ErrAction -->
    <State>Disabled</State>
    <ErrAction>ErrorContinue</ErrAction>
    <RequiredValue format="No/Yes/00/01" example="No"> </RequiredValue>
  </eolvar>
  <eolvar name="Transport Layer Security Supported">
    <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
    <!-- Description>Test variable against expected value</Description -->
    <!-- IntelRequired>False</IntelRequired -->
    <!-- Dependencies></Dependencies -->
    <!-- End of uneditable fields -->
    <!-- Please edit the fields below ONLY with the State or ErrAction -->
    <State>Disabled</State>
    <ErrAction>ErrorContinue</ErrAction>
    <RequiredValue format="No/Yes/00/01" example="No"> </RequiredValue>
  </eolvar>
  <eolvar name="KVM Redirection Supported">
    <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
    <!-- Description>Test variable against expected value</Description -->
    <!-- IntelRequired>False</IntelRequired -->
    <!-- Dependencies></Dependencies -->
    <!-- End of uneditable fields -->
    <!-- Please edit the fields below ONLY with the State or ErrAction -->
    <State>Disabled</State>

```



```
<ErrAction>ErrorContinue</ErrAction>
<RequiredValue format="No/Yes/00/01" example="No"> </RequiredValue>
</eolvar>
<eolvar name="PAVP Supported">
  <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
  <!-- Description>Test variable against expected value</Description -->
  <!-- IntelRequired>False</IntelRequired -->
  <!-- Dependencies></Dependencies -->
  <!-- End of uneditable fields -->
  <!-- Please edit the fields below ONLY with the State or ErrAction -->
  <State>Disabled</State>
  <ErrAction>ErrorContinue</ErrAction>
  <RequiredValue format="No/Yes/00/01" example="No"> </RequiredValue>
</eolvar>
<eolvar name="Intel(R) AMT Supported">
  <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
  <!-- Description>Test variable against expected value</Description -->
  <!-- IntelRequired>False</IntelRequired -->
  <!-- Dependencies></Dependencies -->
  <!-- End of uneditable fields -->
  <!-- Please edit the fields below ONLY with the State or ErrAction -->
  <State>Disabled</State>
  <ErrAction>ErrorContinue</ErrAction>
  <RequiredValue format="No/Yes/00/01" example="No"> </RequiredValue>
</eolvar>
<eolvar name="Intel(R) PTT Supported">
  <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
  <!-- Description>Test variable against expected value</Description -->
  <!-- IntelRequired>False</IntelRequired -->
  <!-- Dependencies></Dependencies -->
  <!-- End of uneditable fields -->
  <!-- Please edit the fields below ONLY with the State or ErrAction -->
  <State>Disabled</State>
  <ErrAction>ErrorContinue</ErrAction>
  <RequiredValue format="Enabled/Disabled/00/01" example="Enabled"> </
RequiredValue>
</eolvar>
<eolvar name="Auto BIST Config Status">
  <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
  <!-- Description>Test variable against expected value</Description -->
  <!-- IntelRequired>False</IntelRequired -->
  <!-- Dependencies></Dependencies -->
  <!-- End of uneditable fields -->
  <!-- Please edit the fields below ONLY with the State or ErrAction -->
  <State>Disabled</State>
  <ErrAction>ErrorContinue</ErrAction>
  <RequiredValue format="Enabled/Disabled/00/01" example="Enabled"> </
RequiredValue>
</eolvar>
<eolvar name="OEM Tag">
  <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
```



```

by the tool -->
    <!-- Description>Test variable against expected value</Description -->
    <!-- IntelRequired>False</IntelRequired -->
    <!-- Dependencies></Dependencies -->
    <!-- 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="0x0"> </RequiredValue>
</eolvar>
<eolvar name="Processor Emulation">
    <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
    <!-- Description>Test variable against expected value</Description -->
    <!-- IntelRequired>False</IntelRequired -->
    <!-- Dependencies></Dependencies -->
    <!-- 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/vPro/Core/Celeron/Pentium/Xeon/Xeon
Manageability Capable" example="No Emulation"> </RequiredValue>
</eolvar>
<eolvar name="PROC_MISSING">
    <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
    <!-- Description>Test variable against expected value</Description -->
    <!-- IntelRequired>False</IntelRequired -->
    <!-- Dependencies></Dependencies -->
    <!-- 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" example="No onboard glue
logic"> </RequiredValue>
</eolvar>
<eolvar name="Firmware Update OEM ID">
    <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
    <!-- Description>Test variable against expected value</Description -->
    <!-- IntelRequired>False</IntelRequired -->
    <!-- Dependencies></Dependencies -->
    <!-- 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="0x0"> </RequiredValue>
</eolvar>
<eolvar name="Intel(R) ME Region Flash Protection Override">
    <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
    <!-- Description>Test variable against expected value</Description -->
    <!-- IntelRequired>False</IntelRequired -->
    <!-- Dependencies></Dependencies -->
    <!-- End of uneditable fields -->

```



```
<!-- Please edit the fields below ONLY with the State or ErrAction -->
<State>Disabled</State>
<ErrAction>ErrorContinue</ErrAction>
<RequiredValue format="False/True/00/01" example="False"> </RequiredValue>
</eolvar>
<eolvar name="M3 Power Rail Available">
  <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
  <!-- Description>Test variable against expected value</Description -->
  <!-- IntelRequired>False</IntelRequired -->
  <!-- Dependencies></Dependencies -->
  <!-- End of uneditable fields -->
  <!-- Please edit the fields below ONLY with the State or ErrAction -->
  <State>Disabled</State>
  <ErrAction>ErrorContinue</ErrAction>
  <RequiredValue format="False/True/Not Available/Available/00/01"
example="False"> </RequiredValue>
</eolvar>
<eolvar name="Debug Override Production Silicon">
  <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
  <!-- Description>Test variable against expected value</Description -->
  <!-- IntelRequired>False</IntelRequired -->
  <!-- Dependencies></Dependencies -->
  <!-- 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="0x0"> </RequiredValue>
</eolvar>
<eolvar name="Debug Override Pre-Production Silicon">
  <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
  <!-- Description>Test variable against expected value</Description -->
  <!-- IntelRequired>False</IntelRequired -->
  <!-- Dependencies></Dependencies -->
  <!-- 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="0x0"> </RequiredValue>
</eolvar>
<eolvar name="WLAN Power Well">
  <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
  <!-- Description>Test variable against expected value</Description -->
  <!-- IntelRequired>False</IntelRequired -->
  <!-- Dependencies></Dependencies -->
  <!-- End of uneditable fields -->
  <!-- Please edit the fields below ONLY with the State or ErrAction -->
  <State>Disabled</State>
  <ErrAction>ErrorContinue</ErrAction>
  <RequiredValue format="Disabled/Sus Well/ME Well/SLP_M#||SPDA/WLAN Sleep via
SLP_WLAN#/80/82/83/84/85/86" example="Disabled"> </RequiredValue>
</eolvar>
```



```

    <eolvar name="LAN Power Well">
      <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
      <!-- Description>Test variable against expected value</Description -->
      <!-- IntelRequired>False</IntelRequired -->
      <!-- Dependencies></Dependencies -->
      <!-- End of uneditable fields -->
      <!-- Please edit the fields below ONLY with the State or ErrAction -->
      <State>Disabled</State>
      <ErrAction>ErrorContinue</ErrAction>
      <RequiredValue format="Core Well/Sus Well/ME Well/SLP_LAN#(MGPIO3)/00/01/02/
03" example="Core Well"> </RequiredValue>
    </eolvar>
    <eolvar name="Firmware KVM Screen Blanking">
      <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
      <!-- Description>Test variable against expected value</Description -->
      <!-- IntelRequired>False</IntelRequired -->
      <!-- Dependencies>CORP</Dependencies -->
      <!-- End of uneditable fields -->
      <!-- Please edit the fields below ONLY with the State or ErrAction -->
      <State>Disabled</State>
      <ErrAction>ErrorContinue</ErrAction>
      <RequiredValue format="No/Yes/00/01" example="No"> </RequiredValue>
    </eolvar>
    <eolvar name="Intel(R) AMT Idle Timeout">
      <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
      <!-- Description>Test variable against expected value</Description -->
      <!-- IntelRequired>False</IntelRequired -->
      <!-- Dependencies>CORP</Dependencies -->
      <!-- 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="0x0"> </RequiredValue>
    </eolvar>
    <eolvar name="Redirection Privacy / Security Level">
      <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
      <!-- Description>Test variable against expected value</Description -->
      <!-- IntelRequired>False</IntelRequired -->
      <!-- Dependencies>CORP</Dependencies -->
      <!-- End of uneditable fields -->
      <!-- Please edit the fields below ONLY with the State or ErrAction -->
      <State>Disabled</State>
      <ErrAction>ErrorContinue</ErrAction>
      <RequiredValue format="Default/Enhanced/Extreme/01/02/03" example="Default">
</RequiredValue>
    </eolvar>
    <eolvar name="OEM Default Certificate 5 Stream">
      <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
      <!-- Description>Test variable against expected value</Description -->
      <!-- IntelRequired>False</IntelRequired -->

```




```
<!-- Dependencies>CORP</Dependencies -->
<!-- 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="0x0"> </RequiredValue>
</eolvar>
<eolvar name="OEM Default Certificate 5 Friendly Name">
  <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
  <!-- Description>Test variable against expected value</Description -->
  <!-- IntelRequired>False</IntelRequired -->
  <!-- Dependencies>CORP</Dependencies -->
  <!-- End of uneditable fields -->
  <!-- Please edit the fields below ONLY with the State or ErrAction -->
  <State>Disabled</State>
  <ErrAction>ErrorContinue</ErrAction>
  <RequiredValue format="String" example="Any"> </RequiredValue>
</eolvar>
<eolvar name="OEM Default Certificate 5 Active">
  <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
  <!-- Description>Test variable against expected value</Description -->
  <!-- IntelRequired>False</IntelRequired -->
  <!-- Dependencies>CORP</Dependencies -->
  <!-- End of uneditable fields -->
  <!-- Please edit the fields below ONLY with the State or ErrAction -->
  <State>Disabled</State>
  <ErrAction>ErrorContinue</ErrAction>
  <RequiredValue format="False/True/Not Active/Active/00/01" example="False">
</RequiredValue>
</eolvar>
<eolvar name="OEM Default Certificate 4 Stream">
  <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
  <!-- Description>Test variable against expected value</Description -->
  <!-- IntelRequired>False</IntelRequired -->
  <!-- Dependencies>CORP</Dependencies -->
  <!-- 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="0x0"> </RequiredValue>
</eolvar>
<eolvar name="OEM Default Certificate 4 Friendly Name">
  <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
  <!-- Description>Test variable against expected value</Description -->
  <!-- IntelRequired>False</IntelRequired -->
  <!-- Dependencies>CORP</Dependencies -->
  <!-- End of uneditable fields -->
  <!-- Please edit the fields below ONLY with the State or ErrAction -->
  <State>Disabled</State>
  <ErrAction>ErrorContinue</ErrAction>
  <RequiredValue format="String" example="Any"> </RequiredValue>
```




```

    </eolvar>
    <eolvar name="OEM Default Certificate 4 Active">
        <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
        <!-- Description>Test variable against expected value</Description -->
        <!-- IntelRequired>False</IntelRequired -->
        <!-- Dependencies>CORP</Dependencies -->
        <!-- End of uneditable fields -->
        <!-- Please edit the fields below ONLY with the State or ErrAction -->
        <State>Disabled</State>
        <ErrAction>ErrorContinue</ErrAction>
        <RequiredValue format="False/True/Not Active/Active/00/01" example="False">
</RequiredValue>
    </eolvar>
    <eolvar name="OEM Default Certificate 3 Stream">
        <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
        <!-- Description>Test variable against expected value</Description -->
        <!-- IntelRequired>False</IntelRequired -->
        <!-- Dependencies>CORP</Dependencies -->
        <!-- 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="0x0"> </RequiredValue>
    </eolvar>
    <eolvar name="OEM Default Certificate 3 Friendly Name">
        <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
        <!-- Description>Test variable against expected value</Description -->
        <!-- IntelRequired>False</IntelRequired -->
        <!-- Dependencies>CORP</Dependencies -->
        <!-- End of uneditable fields -->
        <!-- Please edit the fields below ONLY with the State or ErrAction -->
        <State>Disabled</State>
        <ErrAction>ErrorContinue</ErrAction>
        <RequiredValue format="String" example="Any"> </RequiredValue>
    </eolvar>
    <eolvar name="OEM Default Certificate 3 Active">
        <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
        <!-- Description>Test variable against expected value</Description -->
        <!-- IntelRequired>False</IntelRequired -->
        <!-- Dependencies>CORP</Dependencies -->
        <!-- End of uneditable fields -->
        <!-- Please edit the fields below ONLY with the State or ErrAction -->
        <State>Disabled</State>
        <ErrAction>ErrorContinue</ErrAction>
        <RequiredValue format="False/True/Not Active/Active/00/01" example="False">
</RequiredValue>
    </eolvar>
    <eolvar name="OEM Default Certificate 2 Stream">
        <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
        <!-- Description>Test variable against expected value</Description -->

```



```
<!-- IntelRequired>False</IntelRequired -->
<!-- Dependencies>CORP</Dependencies -->
<!-- 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="0x0"> </RequiredValue>
</eolvar>
<eolvar name="OEM Default Certificate 2 Friendly Name">
  <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
  <!-- Description>Test variable against expected value</Description -->
  <!-- IntelRequired>False</IntelRequired -->
  <!-- Dependencies>CORP</Dependencies -->
  <!-- End of uneditable fields -->
  <!-- Please edit the fields below ONLY with the State or ErrAction -->
  <State>Disabled</State>
  <ErrAction>ErrorContinue</ErrAction>
  <RequiredValue format="String" example="Any"> </RequiredValue>
</eolvar>
<eolvar name="OEM Default Certificate 2 Active">
  <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
  <!-- Description>Test variable against expected value</Description -->
  <!-- IntelRequired>False</IntelRequired -->
  <!-- Dependencies>CORP</Dependencies -->
  <!-- End of uneditable fields -->
  <!-- Please edit the fields below ONLY with the State or ErrAction -->
  <State>Disabled</State>
  <ErrAction>ErrorContinue</ErrAction>
  <RequiredValue format="False/True/Not Active/Active/00/01" example="False">
</RequiredValue>
</eolvar>
<eolvar name="OEM Customizable Certificate 3 Stream">
  <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
  <!-- Description>Test variable against expected value</Description -->
  <!-- IntelRequired>False</IntelRequired -->
  <!-- Dependencies>CORP</Dependencies -->
  <!-- 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="0x0"> </RequiredValue>
</eolvar>
<eolvar name="OEM Customizable Certificate 3 Friendly Name">
  <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
  <!-- Description>Test variable against expected value</Description -->
  <!-- IntelRequired>False</IntelRequired -->
  <!-- Dependencies>CORP</Dependencies -->
  <!-- End of uneditable fields -->
  <!-- Please edit the fields below ONLY with the State or ErrAction -->
  <State>Disabled</State>
  <ErrAction>ErrorContinue</ErrAction>
```



```

        <RequiredValue format="String" example="Any"> </RequiredValue>
    </eolvar>
    <eolvar name="OEM Customizable Certificate 3 Active">
        <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
        <!-- Description>Test variable against expected value</Description -->
        <!-- IntelRequired>False</IntelRequired -->
        <!-- Dependencies>CORP</Dependencies -->
        <!-- End of uneditable fields -->
        <!-- Please edit the fields below ONLY with the State or ErrAction -->
        <State>Disabled</State>
        <ErrAction>ErrorContinue</ErrAction>
        <RequiredValue format="False/True/Not Active/Active/00/01" example="False">
</RequiredValue>
    </eolvar>
    <eolvar name="OEM Customizable Certificate 2 Stream">
        <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
        <!-- Description>Test variable against expected value</Description -->
        <!-- IntelRequired>False</IntelRequired -->
        <!-- Dependencies>CORP</Dependencies -->
        <!-- 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="0x0"> </RequiredValue>
    </eolvar>
    <eolvar name="OEM Customizable Certificate 2 Friendly Name">
        <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
        <!-- Description>Test variable against expected value</Description -->
        <!-- IntelRequired>False</IntelRequired -->
        <!-- Dependencies>CORP</Dependencies -->
        <!-- End of uneditable fields -->
        <!-- Please edit the fields below ONLY with the State or ErrAction -->
        <State>Disabled</State>
        <ErrAction>ErrorContinue</ErrAction>
        <RequiredValue format="String" example="Any"> </RequiredValue>
    </eolvar>
    <eolvar name="OEM Customizable Certificate 2 Active">
        <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
        <!-- Description>Test variable against expected value</Description -->
        <!-- IntelRequired>False</IntelRequired -->
        <!-- Dependencies>CORP</Dependencies -->
        <!-- End of uneditable fields -->
        <!-- Please edit the fields below ONLY with the State or ErrAction -->
        <State>Disabled</State>
        <ErrAction>ErrorContinue</ErrAction>
        <RequiredValue format="False/True/Not Active/Active/00/01" example="False">
</RequiredValue>
    </eolvar>
    <eolvar name="OEM Customizable Certificate 1 Stream">
        <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->

```



```
<!-- Description>Test variable against expected value</Description -->
<!-- IntelRequired>False</IntelRequired -->
<!-- Dependencies>CORP</Dependencies -->
<!-- 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="0x0"> </RequiredValue>
</eolvar>
<eolvar name="OEM Customizable Certificate 1 Friendly Name">
  <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
  <!-- Description>Test variable against expected value</Description -->
  <!-- IntelRequired>False</IntelRequired -->
  <!-- Dependencies>CORP</Dependencies -->
  <!-- End of uneditable fields -->
  <!-- Please edit the fields below ONLY with the State or ErrAction -->
  <State>Disabled</State>
  <ErrAction>ErrorContinue</ErrAction>
  <RequiredValue format="String" example="Any"> </RequiredValue>
</eolvar>
<eolvar name="OEM Customizable Certificate 1 Active">
  <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
  <!-- Description>Test variable against expected value</Description -->
  <!-- IntelRequired>False</IntelRequired -->
  <!-- Dependencies>CORP</Dependencies -->
  <!-- End of uneditable fields -->
  <!-- Please edit the fields below ONLY with the State or ErrAction -->
  <State>Disabled</State>
  <ErrAction>ErrorContinue</ErrAction>
  <RequiredValue format="False/True/Not Active/Active/00/01" example="False">
</RequiredValue>
</eolvar>
<eolvar name="OEM Default Certificate Stream">
  <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
  <!-- Description>Test variable against expected value</Description -->
  <!-- IntelRequired>False</IntelRequired -->
  <!-- Dependencies>CORP</Dependencies -->
  <!-- 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="0x0"> </RequiredValue>
</eolvar>
<eolvar name="OEM Default Certificate Friendly Name">
  <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
  <!-- Description>Test variable against expected value</Description -->
  <!-- IntelRequired>False</IntelRequired -->
  <!-- Dependencies>CORP</Dependencies -->
  <!-- End of uneditable fields -->
  <!-- Please edit the fields below ONLY with the State or ErrAction -->
  <State>Disabled</State>
```



```

        <ErrAction>ErrorContinue</ErrAction>
        <RequiredValue format="String" example="Any"> </RequiredValue>
    </eolvar>
    <eolvar name="OEM Default Certificate Active">
        <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
        <!-- Description>Test variable against expected value</Description -->
        <!-- IntelRequired>False</IntelRequired -->
        <!-- Dependencies>CORP</Dependencies -->
        <!-- End of uneditable fields -->
        <!-- Please edit the fields below ONLY with the State or ErrAction -->
        <State>Disabled</State>
        <ErrAction>ErrorContinue</ErrAction>
        <RequiredValue format="False/True/Not Active/Active/00/01" example="False">
    </RequiredValue>
    </eolvar>
    <eolvar name="Config Server FQDN">
        <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
        <!-- Description>Test variable against expected value</Description -->
        <!-- IntelRequired>False</IntelRequired -->
        <!-- Dependencies></Dependencies -->
        <!-- End of uneditable fields -->
        <!-- Please edit the fields below ONLY with the State or ErrAction -->
        <State>Disabled</State>
        <ErrAction>ErrorContinue</ErrAction>
        <RequiredValue format="String" example="Any"> </RequiredValue>
    </eolvar>
    <eolvar name="FeatureShipState">
        <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
        <!-- Description>Test variable against expected value</Description -->
        <!-- IntelRequired>False</IntelRequired -->
        <!-- Dependencies></Dependencies -->
        <!-- 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="0x0"> </RequiredValue>
    </eolvar>
    <eolvar name="OEMSKURule">
        <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->
        <!-- Description>Test variable against expected value</Description -->
        <!-- IntelRequired>False</IntelRequired -->
        <!-- Dependencies></Dependencies -->
        <!-- 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="0x0"> </RequiredValue>
    </eolvar>
    <eolvar name="MEBxPassword">
        <!-- The commented fields bellow CANNOT be edited. Any edits will be ignored
by the tool -->

```



```

<!-- Description>Test variable against expected value</Description -->
<!-- IntelRequired>False</IntelRequired -->
<!-- Dependencies></Dependencies -->
<!-- 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="0x0"> </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® MEmanuf 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 MEmanuf for testing.

Here is the example that explain how to use this feature:

```

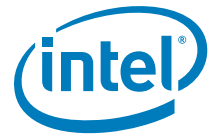
<eolconfig name="PTT FPF">
    <!-- The commented fields bellow CANNOT be edited. Any edits will be
    ignored by the tool -->
    <!-- Description>Check ptt against expected value</Description -->
    <!-- IntelRequired>False</IntelRequired -->
    <!-- Dependencies>PlatformTrust</Dependencies -->
    <!-- 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>

```

5.4.2 MEmanuf –EOL Variable Check

MEmanuf -EOL variable check is designed to check the Intel® ME 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® ME manufacturing mode or No EOP Message Sent.

Note: -EOL Variable check. The system must be in Intel® ME manufacturing mode when -EOL Variable check is run or No EOP Message Sent.



5.4.3 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
EOP status check	Enabled
Intel® ME VSCC check	Set according to the Intel-recommended value.
BIOS VSCC check	Set according to the Intel-recommended value.
Intel® ME Manufacturing Mode status	Disabled.
Flash Region Access Permissions	Set according to the Intel-recommended value.
Flash Descriptor Override Strap (HDA_SDO)	Disabled.
MAC address	None, all 0, or f
Wireless MAC address	None, all 0, or f
System UUID	None, all 0.

Note: `-EOL Config` check. If the system is in Intel® ME manufacturing mode when `-EOL Config` check is run there will be an error report or No EOP Message Sent.

5.4.4 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

5.5.1 Example for Consumer Intel® ME FW SKU

MEmanuf `-verbose`

Intel(R) MEmanuf Version: XX.XX.XX.xxxx



Copyright(C) 2005 - 2014, Intel Corporation. All rights reserved.

```
FW   Status   Register1: 0x86000255
      FW   Status   Register2: 0x6085012E
      FW   Status   Register3: 0x00000000
      FW   Status   Register4: 0x00004000
      FW   Status   Register5: 0x00000000
      FW   Status   Register6: 0x00000000
```

```
CurrentState:           Normal
ManufacturingMode:      Enabled
FlashPartition:         Valid
OperationalState:       CM0 with UMA
InitComplete:           Complete
BUPLoadState:           Success
ErrorCode:              No Error
ModeOfOperation:        Normal
ICC:                    Valid OEM data, ICC programmed
```

Get FWU info command...done

Get FWU version command...done

Get FWU feature state command...done

Get ME FWU platform type command...done

Get ME FWU feature capability command...done

Feature enablement is 0x1001C60

gFeatureAvailability value is 0x1

System is running on consumer/4M image, start Intel(R) ME Runtime Test
OEM ICC data valid and programmed correctly

Request Intel(R) ME test result command...done

vsccommn.bin was created on 23:32:28 05/05/2010 GMT

SPI Flash ID #1 ME VSCC value is 0x2005

SPI Flash ID #1 (ID: 0xEF4017) ME VSCC value checked

SPI Flash ID #1 BIOS VSCC value is 0x2005

SPI Flash ID #1 (ID: 0xEF4017) BIOS VSCC value checked

SPI Flash ID #2 ME VSCC value is 0x2005

SPI Flash ID #2 (ID: 0xEF4017) ME VSCC value checked

SPI Flash ID #2 BIOS VSCC value is 0x2005

SPI Flash ID #2 (ID: 0xEF4017) BIOS VSCC value checked

FPBA value is 0x0

No Intel Wireless device was found

Request Intel(R) ME Runtime BIST test command...done

Get Intel(R) ME test data command...done

Total of 22 Intel(R) ME test result retrieved

Micro Kernel - Blob Manager: Set - Passed

Micro Kernel - Blob Manager: Get - Passed

Micro Kernel - Blob Manager: Remove - Passed

Policy Kernel - SMBus: Read byte - Passed

Policy Kernel - ME Password: Valid MEBx password - Passed



```

Policy Kernel - ME Configuration: Wlan Power Well - Passed
Policy Kernel - ME Configuration: CPU Missing Logic - Passed
Policy Kernel - ME Configuration: CM3 Power Rails Available - Passed
Policy Kernel - Embedded Controller: Get power source - Passed
Common Services - General: Low power idle timeout - Passed
Common Services - Provisioning: Valid MEBX password change policy - Passed
Common Services - Provisioning: Zero-Touch configuration enabled - Passed
Common Services - Provisioning: Client Config mode is valid - Passed
Common Services - General: Vlan not enabled on mobile - Passed
Common Services - Provisioning: Both PID and PPS are set - Passed
Common Services - Provisioning: MEBX password set when PID and PPS set - Passed
Common Services - Wireless LAN: Connectivity to NIC - Skipped
AMT - Privacy Level: Valid Privacy Level settings - Passed

```

Clear Intel(R) ME test data command...done

MEmanuf Test Passed

5.5.2 Example for Corporate Intel® ME FW SKU

MEmanuf -verbose

```

Intel(R) MEmanuf Version: XX.XX.XX.xxxx
Copyright(C) 2005 - 2014, Intel Corporation. All rights reserved.

```

```

FW   Status   Register1:   0x86000255
FW   Status   Register2:   0x6085012E
FW   Status   Register3:   0x00000000
FW   Status   Register4:   0x00004000
FW   Status   Register5:   0x00000000
FW   Status   Register6:   0x00000000

```

```

CurrentState:           Normal
ManufacturingMode:      Enabled
FlashPartition:         Valid
OperationalState:       CM0 with UMA
InitComplete:           Complete
BUPLoadState:           Success
ErrorCode:              No Error
ModeOfOperation:        Normal
ICC:                    Valid OEM data, ICC programmed

```

Get FWU info command...done

Get FWU version command...done

Get FWU feature state command...done

Get ME FWU platform type command...done

```

Get ME FWU feature capability command...done
Feature enablement is 0xDF65C65

```



gFeatureAvailability value is 0x1

Request Intel(R) ME test result command...done

ME initialization state valid

ME operation mode valid

Current operation state valid

ME error state valid

Verifying FW Status Register1...done

OEM ICC data valid and programmed correctly

Request Intel(R) ME test result command...done

vsccommn.bin was created on 03:08:01 01/25/2011 GMT

SPI Flash ID #1 ME VSCC value is 0x2005

SPI Flash ID #1 (ID: 0xEF4017) ME VSCC value checked

SPI Flash ID #1 BIOS VSCC value is 0x2005

SPI Flash ID #1 (ID: 0xEF4017) BIOS VSCC value checked

FPBA value is 0x0

No Intel Wireless device was found

Request Intel(R) ME Full BIST test command...done

Get Intel(R) ME test data command...done

Total of 31 Intel(R) ME test result retrieved

Common Services - LAN: Connectivity to NIC in CM3 - Passed

MicroKernel - Internal Hardware Tests: Internal Hardware Tests - Passed

Policy Kernel - SMBus: Read byte - Passed

Policy Kernel - ME Password: Validate MEBx password - Passed

MicroKernel - Blob Manager: Set - Passed

MicroKernel - Blob Manager: Get - Passed

MicroKernel - Blob Manager: Remove - Passed

Policy Kernel - ME Configuration: Wlan Power Well - Passed

Policy Kernel - ME Configuration: PROC_MISSING - Passed

Policy Kernel - ME Configuration: CM3 Power Rails Available - Passed

Policy Kernel - Embedded Controller: Power source type - Passed

Common Services - General: Low power idle timeout - Passed

Common Services - Privacy Level: Valid Privacy Level settings - Passed

Common Services - General: Vlan not enabled on mobile - Passed

Common Services - Provisioning: Both PID and PPS are set - Passed

Common Services - Provisioning: MEBx password set when PID and PPS set - Passed

Common Services - LAN: Connectivity to NIC in CM0 - Passed

AMT - Power: Valid LAN power well - Passed

AMT - Power: Valid WLAN power well (Mobile) - Failed

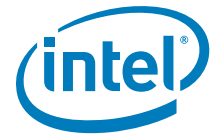
Error 9357: WLAN power well setting is set incorrectly

AMT - KVM: USBx is enabled when KVM is enabled - Passed

AMT - EC: Basic connectivity - Passed

AMT - Hardware Inventory: BIOS tables - Passed

AMT - KVM: Compare engine - Passed



AMT - KVM: Compression engine - Passed
AMT - KVM: Sampling engine - Skipped
AMT - KVM: VDM engine - Passed
AMT - USB: Hardware - Passed

Clear Intel(R) ME test data command...done

Error 9296: MEmanuf Test Failed

§ §



6 Intel® MEInfo

MEInfoWin and Intel® MEInfo provide a simple test to check whether the Intel® ME FW is alive. Both tools perform the same test; query the Intel® ME FW including Intel® AMT – and retrieve data.

Table 18 contains a list of the data that each tool returns.

The Windows® version of MEInfo (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.

Meinfo reports an LMS error. This behavior is expected as the LMS driver cannot be installed on Windows® PE.

6.2 Usage

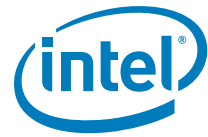
The executable can be invoked by:

```
MEInfo.exe [-EXP] [-H|?] [-VER] [-FITVER] [-FEAT] [-VALUE] [-FWSTS]  
[-VERBOSE] [-PAGE]
```

```
MEInfo.efi [-EXP] [-H] [-VER] [-FITVER] [-FEAT] [-VALUE] [-FWSTS]  
[-VERBOSE] [-PAGE]
```

Table 6-1. Intel® MEInfo Command Line Options

Option	Description
-FEAT <name> <column>	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".
-VALUE <value>	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



Option	Description																										
-FEAT <name> <column>	<p>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® MEINFO.</p> <p>Intel® MEINFO can retrieve all of the information detailed below. However, depending on the SKU selected, some information may not appear.</p> <p>Note: For the EFI shell version you need to add additional "^" to enclose the text string in order for it to be properly parsed.</p> <p>Example: MEINFO.efi -feat "^BIOS boot state"^</p>																										
-FWSTS	<p>Decodes the Intel® ME FW status register value field and breaks it down into the following bit definitions for easy readability:</p> <pre>FW Status Register1: 0x90000255 FW Status Register2: 0x00F10506 FW Status Register3: 0x00000020 FW Status Register4: 0x00004004 FW Status Register5: 0x00000000 FW Status Register6: 0x00400000</pre> <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>ME File System Corrupted:</td><td>No</td></tr> <tr> <td>PhaseStatus:</td><td>PROTECTED_START</td></tr> <tr> <td>FPF and ME 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	ME File System Corrupted:	No	PhaseStatus:	PROTECTED_START	FPF and ME 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																										
ME File System Corrupted:	No																										
PhaseStatus:	PROTECTED_START																										
FPF and ME 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® MEINFO 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.																										



Table 6-2. List of Components that Intel® MEINFO Displays

Feature Name	Feature Data Source (Intel® ME Kernel/ Intel® AMT/ SW/ Other)	Consumer SKU	Corporate SKU	Specific Feature Dependency	Field Value
Tools Version	SW (Intel® MEInfo)	X	X	N/A	Version string Example: 12.x.y.ZZZZ; where x=minor, y = HF/MR, ZZZZ = Build Number.
BIOS Version	Intel® ME Kernel	X	X	MEBx needs to be present. Not available on Corporate Sku	Version string
MEBx Version	Intel® ME Kernel	X	X	MEBx needs to be present. Not available on Corporate Sku	Version string 12.x.y.ZZZZ; where x=minor, y = HF/MR, ZZZZ = Build Number.
GbE Version	Other (Directly reading from SPI)	X	X	GbE Region to be present in the image	A version string
PMC Firmware Version	Other (Directly reading from SPI)	X	X	PMC Region to be present in the image	A version string Unknown if partition does not exist. 0 if empty
Descriptor Version	Other (Directly reading from SPI)	X	X	SPI Image	A version string
VendorID	Intel® ME Kernel	X	X	N/A	A number (in Hex)



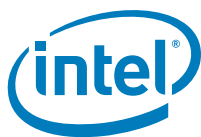
Feature Name	Feature Data Source (Intel® ME Kernel/ Intel® AMT/ SW/ Other)	Consumer SKU	Corporate SKU	Specific Feature Dependency	Field Value
FW Version	Intel® ME Kernel	X	X	N/A	Version string XX.x.y.ZZZZ A B; where XX=major, x=minor, y = HF/MR, ZZZZ = Build Number, A=LP/H, B=SKU type [Consumer/ Corporate].
Security Version (SVN)	Intel® ME Kernel	X	X	N/A	Version Number
LMS version*	Other (Reading Windows® registry entries)	X	X	Only when Windows® LMS driver is installed	A version string
Intel® MEI Driver version*	Other (Reading Windows® registry entries)	X	X	Only when Windows® Intel® MEI driver is installed	A version string
Wireless Driver/ Hardware Version*	Other (Reading Windows® registry entries)	X	X	Only when wireless HW is present, and wireless Windows® driver is installed	A version string
PCH Information	Intel® ME Kernel	X	X	N/A	Display of PCH Information including: <ul style="list-style-type: none"> • Version • Device ID • Step Data • SKU Type • PCH Replacement Counter • PCH Replacement Counter State • PCH Unlocked State



Feature Name	Feature Data Source (Intel® ME Kernel/ Intel® AMT/ SW/ Other)	Consumer SKU	Corporate SKU	Specific Feature Dependency	Field Value
FW Capabilities	Intel® ME Kernel	X	X	N/A	<p>Combination of feature name list breakdown (with a Hexadecimal value)</p> <p>*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.</p> <p>Information Includes:</p> <ul style="list-style-type: none"> • Intel(R) Active Management Technology • Protect Audio Video Path • Intel(R) Dynamic Application Loader • Service Advertisement & Discovery • Intel(R) Platform Trust Technology • Persistent RTC and Memory • Intel(R) Precise Touch and Stylus
FW Type	Intel® ME Kernel	X	X	N/A	Pre-Production/Production
Intel® AMT State	Intel® ME Kernel		X	Both Full Manageability and Manageability Application have to be PRESENT (Capable)	Enabled/Disabled



Feature Name	Feature Data Source (Intel® ME Kernel/ Intel® AMT/ SW/ Other)	Consumer SKU	Corporate SKU	Specific Feature Dependency	Field Value
TLS	Intel® ME Kernel	X	X	N/A	Enabled/Disabled
Last Intel® ME Reset Reason	Intel® ME Kernel	X	X	N/A	Power up/ Firmware reset/ Global system reset/ Unknown
Local FWUpdate	Intel® ME Kernel	X	X	N/A	Enabled/Disabled/ Password Protected
BIO	Other (Directly reading from SPI)	X	X	N/A	Enabled/Disabled/ Unknown
GbE Config Lock	Other (Directly reading from SPI)	X	X	N/A	Enabled/Disabled/ Unknown
Host Read Access to Intel® ME	Other (Directly reading from SPI)	X	X	N/A	Enabled/Disabled/ Unknown
Host Write Access to Intel® ME	Other (Directly reading from SPI)	X	X	N/A	Enabled/Disabled/ Unknown
Host Read Access to EC/Host Write Access to EC	Other (Directly reading from SPI)	X	X	N/A	Enabled/Disabled/ Unknown
SPI Flash ID	Other (Directly reading from SPI)	X	X	Only when there are flash parts HW installed	A JEDEC ID number (in Hex)



Feature Name	Feature Data Source (Intel® ME Kernel/ Intel® AMT/ SW/ Other)	Consumer SKU	Corporate SKU	Specific Feature Dependency	Field Value
ME/BIOS VSCC register values	Other (Directly reading from SPI)	X	X	Only when there are flash parts HW installed	A 32bit VSCC number (in Hex)
BIOS Boot State	Intel® ME Kernel	X	X	N/A	Pre Boot/ In Boot/ Post Boot
OEM Id	Intel® ME Kernel	X	X	Only if fw image supports OEM Id	UUID for OEM to check during FW Update
Capability Licensing Service	Intel® ME Kernel	X	X	Not available on Corporate Sku. Not shown unless Fw feature capability supports it	Enabled/Disabled
OEM Tag	Intel® ME Kernel	X	X	N/A	A 32bit Hexadecimal number
Report on Revenue Sharing ID Fields	Intel® ME Kernel Firmware Host Interface	Both	All	N/A	3 slot of 32-bit integer values (in Hex)
M3 Autotest	Intel® ME Kernel		X	FIT CM3 Autotest Enabled set to 'true'	Enabled/Disabled
C-Link Status	Intel® ME Kernel		X	Intel® Wireless LAN	Enabled/Disabled



Feature Name	Feature Data Source (Intel® ME Kernel/ Intel® AMT/ SW/ Other)	Consumer SKU	Corporate SKU	Specific Feature Dependency	Field Value
Link Status	Intel® AMT	X	X	Intel® AMT CEM (a.k.a. Common Service) is used. Not available on Corporate Sku	Link up/down
System UUID	Intel® AMT	N/A	X	AMT CEM (a.k.a. Common Service) is used. Not available on Corporate Sku	UUID of the system
Configuration State	Intel® AMT	N/A	X	AMT CEM (a.k.a. Common Service) is used. Not available on Consumer Sku	Not started/ In process/ Completed/ Unknown
MAC Address	Intel® AMT	N/A	X	AMT CEM (a.k.a. Common Service) is used only when wired Hw is present. Not available on Consumer Sku	A MAC address (in Hex separated by "=")



Feature Name	Feature Data Source (Intel® ME Kernel/ Intel® AMT/ SW/ Other)	Consumer SKU	Corporate SKU	Specific Feature Dependency	Field Value
Wireless MAC Address	Intel® AMT	N/A	X	AMT CEM (a.k.a. Common Service) is used only when wireless HW is present. Not available on Consumer Sku	A MAC address (in Hex separated by "=")
IPv4 Address (Wired and Wireless)	Intel® AMT	N/A	X	Intel® AMT CEM (a.k.a. Common Service) is used only when wired/ wireless Hw is present. Not available on Consumer Sku	IPv4 IP address (in decimal separated by ".")
IPv6 Address (Wired and Wireless)	Intel® AMT	N/A	X	Intel® AMT CEM (a.k.a. Common Service) is used only when wired/ wireless Hw is present. Not available on Consumer Sku	All IPv6 IP addresses



Feature Name	Feature Data Source (Intel® ME Kernel/ Intel® AMT/ SW/ Other)	Consumer SKU	Corporate SKU	Specific Feature Dependency	Field Value
IPv6 enabled (Wired and Wireless)	Intel® AMT	N/A	X	Intel® AMT CEM (a.k.a. Common Service) is used only when wired/ wireless Hw is present. Not available on Consumer Sku	Enabled/Disabled
Privacy / Security Level	Intel® AMT	N/A	X	Not available on Consumer SKU. Only shown when AMT is enabled	Default/Enhanced/ Extreme/Unknown
Provisioning Mode	Intel® AMT	N/A	X	Intel® AMT CEM (a.k.a. Common Service) is used only when wired/ wireless Hw is present. Not available on Consumer Sku	
FWSTS	Intel® ME Kernel	X	X	N/A	Firmware status, 32bit Hexadecimal numbers and their bit definition breakdown. Available when -fwsts or -verbose are specified.
Wireless Micro-code Mismatch	FWU	Corporate	All	N/A	Yes: FW has detected a ucode mismatch, and partial FWUpdate needs to be performed



Feature Name	Feature Data Source (Intel® ME Kernel/ Intel® AMT/ SW/ Other)	Consumer SKU	Corporate SKU	Specific Feature Dependency	Field Value
Wireless LAN in Firmware	FWU	Corporate	All	N/A	The "friendly name" matching the WLAN ucode in FW
Wireless Micro-code ID in Firmware	FWU	Corporate	All	N/A	The current WLAN ucode in FW
Wireless LAN Hardware	PCI address	Corporate	All	N/A	The "friendly name" of the Wireless LAN hardware installed on the system
Wireless Hardware ID	PCI address	Corporate	All	N/A	The WLAN DeviceID read from PCI space of the installed WLAN on the system
Localized Language	FWU	All	All	N/A	Displaying the language installed in the flash in English
Keybox	Intel® ME Kernel	All	All	N/A	Enabled/Disabled
Intel® PTT Supported	Intel® ME Kernel	All	All	N/A	Yes/No
Intel® PTT Initial Power State	Intel® ME Kernel	All	All	N/A	Enabled/Disabled
PAVP Supported	Intel® ME Kernel	All	All	Platform Protection	Yes/No
Integrated Sensor Hub Initial Power State	Intel® ME Kernel	All	All		Enabled/Disabled
End of Manufacturing Enable	Intel® ME Kernel	All	All		Yes/No



Feature Name	Feature Data Source (Intel® ME Kernel/ Intel® AMT/ SW/ Other)	Consumer SKU	Corporate SKU	Specific Feature Dependency	Field Value
Post Manufacturing NVAR Config Enabled	Intel® ME Kernel	All	All		Yes/No
Minimum Allowed Anti Rollback SVN	Intel® ME Kernel	All	All	BIOS	
Image Anti Rollback SVN	Intel® ME Kernel	All	All	BIOS	
Trusted Computing Base SVN	Intel® ME Kernel	All	All	BIOS	
ACM SVN FPF	Intel® ME Kernel	All	All	BIOS	
KM SVN FPF	Intel® ME Kernel	All	All	BIOS	
BSMM SVN FPF	Intel® ME Kernel	All	All	BIOS	
OEM Public Key Hash FPF	Intel® ME Kernel	All	All	BIOS	SHA-256bit Hash entry (Set once fuses are burned)
OEM Public Key Hash UEP	Intel® ME Kernel	All	All	BIOS	SHA-256bit Hash entry (Value prior to burning fuses)
OEM Public Key Hash ME FW	Intel® ME Kernel	All	All	BIOS	SHA-256bit Hash entry (Value currently in use by FW)
HW Binding	Intel® ME Kernel	All	All	N/A	Enabled/Disabled



Feature Name	Feature Data Source (Intel® ME Kernel/ Intel® AMT/ SW/ Other)	Consumer SKU	Corporate SKU	Specific Feature Dependency	Field Value
GuC Encryption Key ME	Intel® ME Kernel	All	All	BIOS	256-bit string
Force Boot Guard ACM	Intel® ME Kernel	All	All	BIOS	Yes / No
Key Manifest ID	Intel® ME Kernel	All	All	BIOS	Hash of Public Key to verify Boot Policy Manifest
PTT	Intel® ME Kernel	All	All	BIOS	Enabled / Disabled
SPI Boot Source	Intel® ME Kernel	All	All	BIOS	Enabled / Disabled
Enforcement Policy	Intel® ME Kernel	All	All	BIOS	Unrestricted / Remediation / Restricted
OEM ID	Intel® ME Kernel	All	All	BIOS	Hex Value
TXT Supported	Intel® ME Kernel	All	All	BIOS	Enabled/Disabled
OEM Key Manifest Present	Intel® ME Kernel	All	All	BIOS	Present / Not Present
OEM Platform ID	Intel® ME Kernel	All	All	BIOS	Hex Value
SOC Config Lock	Intel® ME Kernel	All	All	BIOS	Done / Not Done
Persistent PRTC Backup Power	Intel® ME Kernel	All	All	BIOS	Enabled / Disabled
EK Revoke State	Intel® ME Kernel	All	All	BIOS	Revoked / Not Revoked
CPU Debugging	Intel® ME Kernel	All	All	BIOS	Enabled / Disabled



Feature Name	Feature Data Source (Intel® ME Kernel/ Intel® AMT/ SW/ Other)	Consumer SKU	Corporate SKU	Specific Feature Dependency	Field Value
BSP Initialization	Intel® ME Kernel	All	All	BIOS	Enabled / Disabled
Measured Boot	Intel® ME Kernel	All	All	BIOS	Yes / No
Verified Boot	Intel® ME Kernel	All	All	BIOS	Yes / No
Protect BIOS Environment	Intel® ME Kernel	All	All	BIOS	Yes / No
iTouch	SW (Intel® MEInfo)	All	All	iTouch	iTouch information includes: <ul style="list-style-type: none"> • Device ID • HW Revision ID • FW Revision ID • Frame Size • Feedback Size • Sensor Mode • Maximum Number of Touch Point • SPI Frequency • SPI I/O Mode

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. The DOS version does not display the UNS version, Intel® Management Engine Interface, or LMS version numbers.

Note: **If EOM is set, for FPF's the FPF and ME column values both will be displayed**

6.3.1 Consumer Intel® ME FW SKU Sample Output

ÿþ

Intel(R) MEInfo Version: 12.0.0.XXXX

Copyright(C) 2005 - 2017, Intel Corporation. All rights reserved.



Intel(R) ME code versions:

BIOS Version	CNLSFWR1.R00.X100.A01.1708151220
MEBx Version	12.0.0.XXXX
GbE Version	0.2
PMC FW Version	10.0.2.XXXX
Descriptor Version	1.0
Vendor ID	8086
FW Version	12.0.0.XXXX LP Consumer
Security Version (SVN)	1
LMS Version	1726.12.0.XXXX
MEI Driver Version	1726.12.0.XXXX
Wireless Hardware Version	Not Available
Wireless Driver Version	Not Available
PCH Information	
PCH Version	11
PCH Device ID	9D84
PCH Step Data	B1
PCH SKU Type	Pre-Production ES
PCH Replacement Counter	0
PCH Replacement State	Disabled
PCH Unlocked State	Disabled
FW Capabilities	0x31109650
Protect Audio Video Path - PRESENT/ENABLED	
Intel(R) Dynamic Application Loader - PRESENT/ENABLED	
Intel(R) Platform Trust Technology - PRESENT/ENABLED	
Persistent RTC and Memory - PRESENT/ENABLED	
Intel(R) Precise Touch and Stylus - PRESENT/ENABLED	
FW Type	Pre-Production
TLS	Disabled
Last ME reset reason	Global system reset
Local FWUpdate	Enabled
BIOS Config Lock	Enabled
GbE Config Lock	Enabled
Host Read Access to ME	Enabled
Host Write Access to ME	Enabled
Host Read Access to EC	Enabled
Host Write Access to EC	Enabled
SPI Flash ID 1	EF4019
SPI Flash ID 2	Not Available
BIOS boot State	Post Boot
OEM ID	00000000-0000-0000-0000-000000000000
Capability Licensing Service	Enabled
OEM Tag	0x00000000
Slot 1 Board Manufacturer	0x00000000
Slot 2 System Assembler	0x00000000
Slot 3 Reserved	0x00000000
M3 Autotest	Disabled



C-link Status	Disabled
EPID Group ID	0x4DC
Keybox	Not Provisioned
Intel(R) PTT Supported	Yes
Intel(R) PTT initial power-up state	Enabled
PAVP Supported	Yes
Integrated Sensor Hub Initial Power State	Enabled
End of Manufacturing Enable	No
Post Manufacturing NVAR Config Enabled	Yes
ACM SVN FPF	0x0
KM SVN FPF	0x0
BSMM SVN FPF	0x0
OEM Public Key Hash FPF	Not set
OEM Public Key Hash UEP	
4D19B4F23FF9170C2C46B3D76BF05919A7FA8B6B113DF53C86C0E8003C23A8DC	
OEM Public Key Hash ME FW	
4D19B4F23FF9170C2C46B3D76BF05919A7FA8B6B113DF53C86C0E8003C23A8DC	
HW Binding	Disabled

	FPF	UEP *In Use	ME FW
Key Manifest ID	Not set	0x1	0x1
PTT	Not set	Enabled	Enabled
SPI Boot Source	Not set	Enabled	Enabled
Enforcement Policy	Not set	0x0	0x0
OEM ID	Not set	0x0	0x0
TXT Supported	Disabled	Disabled	Disabled
OEM Key Manifest Present	Not Present	Present	Present
OEM Platform ID	Not set	0x0	0x0
SOC Config Lock	Not set	Not Done	Not Done
Persistent PRTC Backup Power	Enabled	Enabled	Enabled
EK Revoke State	Not Revoked	Not Revoked	Not Revoked
CPU Debugging	Not set	Enabled	Enabled
BSP Initialization	Not set	Enabled	Enabled
Measured Boot	Not set	Disabled	Disabled
Verified Boot	Not set	Disabled	Disabled
Protect BIOS Environment	Not set	Disabled	Disabled
Touch - Vendor ID	Not Available		
Touch - Device ID	Not Available		
Touch - HW Revision ID	Not Available		
Touch - FW Revision ID	Not Available		
Touch - Frame Size	Not Available		
Touch - Feedback Size	Not Available		
Touch - Sensor Mode	Not Available		
Touch - Maximum Number of Touch Point	Not Available		
Touch - SPI Frequency	Not Available		
Touch - SPI I/O Mode	Not Available		



6.3.2 Corporate Intel® ME FW SKU Sample Output

Intel(R) MEInfo Version: 12.X.X.X
Copyright(C) 2005 - 2017, Intel Corporation. All rights reserved.

Intel(R) Manageability and Security Application code versions:

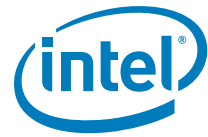
BIOS Version	CNLSFWR1.R00.X114.B03.1711241547
MEBx Version	12.0.0.XXXX
GbE Version	0.2
PMC FW Version	30.1.11.4104
Descriptor Version	1.0
Vendor ID	8086
FW Version	12.0.0.XXXX LP Corporate
Security Version (SVN)	1
LMS Version	1737.12.0.XXXX
MEI Driver Version	1737.12.0.XXXX
Wireless Hardware Version	Not Available
Wireless Driver Version	Not Available

PCH Information	
PCH Version	11
PCH Device ID	9D84
PCH Step Data	A0
PCH SKU Type	Pre-Production ES
PCH Replacement Counter	0
PCH Replacement State	Disabled
PCH Unlocked State	Disabled

FW Capabilities	0x7DF6D655
-----------------	------------

Intel(R) Active Management Technology - PRESENT/ENABLED
Protect Audio Video Path - PRESENT/ENABLED
Intel(R) Dynamic Application Loader - PRESENT/ENABLED
Service Advertisement & Discovery - PRESENT/ENABLED
Intel(R) Platform Trust Technology - PRESENT/ENABLED
Persistent RTC and Memory - PRESENT/ENABLED
Intel(R) Precise Touch and Stylus - PRESENT/ENABLED

FW Type	Pre-Production
Intel(R) AMT State	Enabled
TLS	Enabled
Last ME reset reason	Global system reset
Local FWUpdate	Enabled
BIOS Config Lock	Enabled
GbE Config Lock	Enabled
Host Read Access to ME	Enabled
Host Write Access to ME	Enabled
Host Read Access to EC	Enabled
Host Write Access to EC	Enabled
SPI Flash ID 1	EF4019
SPI Flash ID 2	Not Available



BIOS boot State	Post Boot
OEM ID	00000000-0000-0000-0000-000000000000
Link Status	Link Down
System UUID	88888888-8887-8888-8888-878888888888
MAC Address	88-88-88-88-87-88
IPv4 Address	0.0.0.0
IPv6 Enablement	Disabled
Privacy/Security Level	Default
Configuration State	Not Started
Provisioning Mode	PKI
Capability Licensing Service	Enabled
OEM Tag	0x00000000
Slot 1 Board Manufacturer	0x00000000
Slot 2 System Assembler	0x00000000
Slot 3 Reserved	0x00000000
M3 Autotest	Disabled
C-link Status	Enabled
Wireless Micro-code Mismatch	No
Wireless Micro-code ID in Firmware	0x9DF0
Wireless LAN in Firmware	Intel(R) Dual Band Wireless-AC 9560
Wireless Hardware ID	No Intel WLAN card installed
Wireless LAN Hardware	No Intel WLAN card installed
Localized Language	English
EPID Group ID	0x4DC
Keybox	Not Provisioned
Intel(R) PTT Supported	Yes
Intel(R) PTT initial power-up state	Enabled
PAVP Supported	Yes
Integrated Sensor Hub Initial Power State	Enabled
End of Manufacturing Enable	No
Post Manufacturing NVAR Config Enabled	Yes
Minimum Allowed Anti Rollback SVN	1
Image Anti Rollback SVN	2
Trusted Computing Base SVN	1
ACM SVN FPF	0x0
KM SVN FPF	0x0
BSMM SVN FPF	0x0
OEM Public Key Hash FPF	Not set
OEM Public Key Hash UEP	
4D19B4F23FF9170C2C46B3D76BF05919A7FA8B6B113DF53C86C0E8003C23A8DC	
OEM Public Key Hash ME FW	
4D19B4F23FF9170C2C46B3D76BF05919A7FA8B6B113DF53C86C0E8003C23A8DC	
HW Binding	Disabled

	FPF	UEP *In Use	ME FW
	---	---	----
Key Manifest ID	Not set	0x1	0x1
PTT	Not set	Enabled	Enabled
SPI Boot Source	Not set	Enabled	Enabled
Enforcement Policy	Not set	0x0	0x0
OEM ID	Not set	0x0	0x0
TXT Supported	Disabled	Disabled	Disabled
OEM Key Manifest Present	Not Present	Present	Present



OEM Platform ID	Not set	0x0	0x0
SOC Config Lock	Not set	Not Done	Not Done
Persistent PRTC Backup Power	Enabled	Enabled	Enabled
EK Revoke State	Not Revoked	Not Revoked	Not Revoked
CPU Debugging	Not set	Enabled	Enabled
BSP Initialization	Not set	Enabled	Enabled
Measured Boot	Not set	Enabled	Enabled
Verified Boot	Not set	Enabled	Enabled
Protect BIOS Environment	Not set	Enabled	Enabled

Touch - Vendor ID	Not Available
Touch - Device ID	Not Available
Touch - HW Revision ID	Not Available
Touch - FW Revision ID	Not Available
Touch - Frame Size	Not Available
Touch - Feedback Size	Not Available
Touch - Sensor Mode	Not Available
Touch - Maximum Number of Touch Point	Not Available
Touch - SPI Frequency	Not Available
Touch - SPI I/O Mode	Not Available

6.3.3 Retrieve Current Value of Flash Version

```
C:\ MEINFO.exe -feat "BIOS boot state"
Intel(R) MEINFO Version: XX.XX.XX.xxxx
Copyright(C) 2005 - 2017, Intel Corporation. All rights reserved.
```

BIOS boot State: Post Boot

```
> MEINFO.efi -feat "^"BIOS boot state"^"
Intel(R) MEINFO Version: XX.XX.XX.xxxx
Copyright(C) 2005 - 2017, Intel Corporation. All rights reserved.
```

BIOS boot State: Post Boot

6.3.4 Checks Whether Computer Has Completed Set-up and Configuration Process

```
C:\ MEINFO.exe -feat "Setup and Configuration" -value "Not Completed"
```

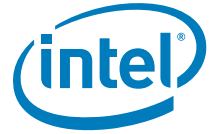
```
Intel(R) MEINFO Version: XX.XX.XX.xxxx
Copyright(C) 2005 - 2017, Intel Corporation. All rights reserved.
```

Local FWUpdate: Success - Value matches FW value.

```
> MEINFO.efi -feat "^"Setup and Configuration"^" -value "^"Not Completed"^"
```

```
Intel(R) MEINFO Version: XX.XX.XX.xxxx
Copyright(C) 2005 - 2017, Intel Corporation. All rights reserved.
```

Local FWUpdate: Success - Value matches FW value.



7 Intel® ME Firmware Update

FWUpdate allows an end user, such as an IT administrator, to update Intel® ME FW without having to reprogram the entire flash device. It then verifies that the update was successful.

FWUpdate does not update the BIOS, GbE, or Descriptor Regions. It updates the FW code portion along with the WCOD, LOCL, IUNP and ISH partitions. Intel® FWUpdate updates the entire Intel® ME code area. In addition FWUpdate local can perform a partial update to change / update the WCOD, LOCL, IUNP and ISH portions.

The image file that the FWUpdate 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 ME firmware stitched together with the PMC binary only.
- The *_full*.bin file contains the ME firmware stitched together with the PMC binary as well as any IUPs and the OEM Key Manifest (when provided).

FWUpdate takes approximately 1-4 minutes to complete depending on the flash device on the system.

After FWUpdate 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® ME Local Firmware Update and Intel® ME Remote Firmware Update. Now there is just a local firmware update tool that is called Intel® ME Firmware Update (FWUpdate).

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® ME FWUpdate must be enabled in the Intel® MEBx or through BIOS.

The Intel® ME 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 Windows® PE Requirements

In order for tools to work under Windows® PE environment, the user will need to manually load a driver by using the .inf file in the Intel® MEI driver installation files. Once the .inf file located, the user will need to use Windows® PE command `drvload *.inf` to load it into the running system each time Windows® PE reboots. Failure to do so causes a tools reporting error.

7.3 Enabling and Disabling Intel® FWUpdate

In Intel® MEBx (or BIOS depending on customer implementation), 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

Password Protected – allows the FW to be updated only if a valid Intel® Mebx password is provided using the “-pass” option. If password does not match the tool will display the appropriate error message. The user will have a maximum of three tries before being asked to reboot the system to try again.

For more details, refer Intel® MEBx user guide.

7.4 FWUpdate Flows

7.4.1 Full FWUpdate

This will help allow to update Intel® ME Firmware. If IUP's are present in the payload image along with Intel® ME Firmware, IUP's will also be updated along with Intel® ME as part of the Full FWUpdate.

Global Reset will be required to complete the FWUpdate operation.

PMC Firmware Update: This will be handled as part of the Full FWUpdate flow and cannot be updated on its own. PMC Firmware needs to be stitched with Intel® ME Firmware using Intel® FIT Tool and that image will be used as the payload to Full FWUpdate Flow for updating PMC Firmware.

Intel® ME Firmware Update: This will be handled as part of the Full FWUpdate Flow. Requirement: Only CSE Image won't be allowed as the payload to execute update. Pre-Stitched ME + PMC binary needs to be used as the payload to execute ME update.



7.4.2 Partial FWUpdate

This will help allow to update IUP's (Independent Updatable Partitions) only i.e. WLAN micro-code, ISH Firmware, Localization, IUnit Loader etc.

For optional IUP's (ISH) Firmware Update only, ISH Firmware can be directly used as the payload to update ISH FW only using Partial FWUpdate. No stitching with Intel® ME Firmware required.

7.5 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.

```
FWUpdLcl.exe [-H|?] [-VER] [-EXP] [-VERBOSE] [-F] [-Y] [-GENERIC]
              [-SAVE] [-FWVER] [-ALLOWSV] [-FORCERESET]
              [-OEMID] [-PARTVER]
```

```
FWUpdLcl.efi [-H|?] [-VER] [-EXP] [-VERBOSE] [-F] [-Y] [-SAVE]
              [-FWVER] [-ALLOWSV] [-FORCERESET] [-OEMID] [-PARTVER]
```

Note: Image File is the image file of the FW to be updated. Is the same image file used by FIT.

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 FWUpdate 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.
-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.



Option	Description
-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 FWUpdate 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 FPT and NFTP partitions. A partial image starts with either WCOD or LOCL partitions.</p>
-GENERIC	<p>Intel® MEI. Specifies that the tool performs the update over the Intel® MEI interface. Intel® MEI is used even if the FW supports a network-based update.</p> <p>Note: This option is only supported in the Windows® version of the tool.</p>
-FWVER	Display FW version
-H or -?	<p>Displays the list of command line options supported by the Intel® MEINFO tool.</p> <p>Note: Use -H for help when running in the EFI Shell.</p>
-EXP	Shows examples about how to use the tools.
-VER	Shows the version of the tools.
-PARTVER	Display flashed ISH FW Version

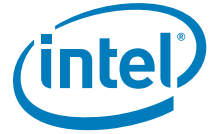
7.6 Examples

7.6.1 Updates Intel® ME with Firmware Binary File

Note: In order to execute FWUpdLcl in EFI, make sure all the payload files and FWUpdate executable are located in the root folder.

This command updates Intel® ME with FW.BIN file. If the firmware on current platform is newer than the 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
```

7.6.2 Partial Firmware Update

This command will perform a partial update of the FW via Intel® MEI for either the wcod, locl, iup and ishc partitions.

```
FWUpdLcl.exe -f FW.bin -partid <wcod, locl, iup or ishc>
```

```
EFI:
FWUpdLcl.efi -f upd.bin -partid <wcod, locl, iup or ishc>
```

Non-Verbos Mode

```
C:\> FWUpdLcl.exe -f FW.BIN.bin -partid WCOD
```

```
Intel (R) Firmware Update Utility version xx.xx.xx.xxxx
Copyright (C) 2007-2017, 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 (R) Firmware Update Utility version xx.xx.xx.xxxx
Copyright (C) 2007-2017, 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.
```

7.6.3 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 '-'. 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 (R) 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.

```
EFI:  
> FWUpdLcl.efi -exp partid
```

Intel (R) 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.

7.6.4 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 FWUpdate API Library

8.1 Getting Started - FWUpdate Library

8.1.1 Introduction

This chapter will describe the Firmware Update Libraries that will be used for Intel® Management Engine (Intel® ME) update. It contains a description of the various APIs to be used.

The Firmware Update process is essential for updating WCOD and LOCL regions by utilizing the APIs provided in the Firmware Update Library.

8.1.2 Environment

The FWUpdate 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 (FWUpdate) 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 FWupdate 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 FWUpdate Sample App



**Note: Please Refer to the Actual Sample App Source Code under EFI/
SampleSource/ Provided in the Kit for Proper Details.**

```
/*++
```

Copyright (c) 2014-2016 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 (R) Firmware Update Utility Sample Application \n");
Print (L"\n Intel (R) 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 ME 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);

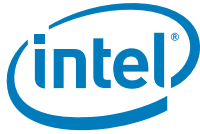
    switch (Upd_Type)
    {
    case DOWNGRADE_SUCCESS:

    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:

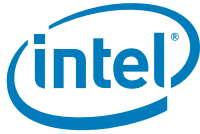
case FWU_IME_NO_DEVICE:
    Print (ID_ERROR_68, FWU_UPDATE_POLLING_FAILED);
    return EFI_SUCCESS;
case FWU_IME_NOT_READY:
    DisplayTextForReturnErrorCode(status);
    return EFI_SUCCESS;
case FWU_ERROR_FW:
    Print (ID_ERROR_69, FWU_ERROR_FW, UpdateStatus);
    return EFI_SUCCESS;
default:
    break;
}

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 FWUpdate 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 FWUpdate 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 fwupdate process just completed LastResetType - indicates Reset type required for the fwupdate 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	Buffer – Buffer with the update image Buffer Length – Length of buffer Password – MEBX Password ForceResetLib – Flag to perform system reset UpdateEnvironment – differentiates various firmware update process environment within the firmware (manufacturing/non-manufacturing) UUID OEMID – OEM ID update_flags – flag to indicate FW of recovery/rollback Func pointer – (bytes of Binary
Returns	0 = Success Non-zero value = Failure

8.3 FW Update Partial Buffer

```
unsigned int FwUpdatePartialBuffer(char* buffer,unsigned int bufferLength, unsigned
int PartitionID, unsigned int Flags, IPU_UPDATED_INFO *IpuUpdatedInfo,
void(*func)(float, float));
```

Purpose: This function performs the Partial FW Update. If the requested partition is expected by the Firmware, it 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. If the requested partition is not expected by the firmware an error will be returned to the user.

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.

FWUpdate API Library supports only Partial FWUpdate for ISH only. -i is the command line switch.

Example Usage: FwUpdLclApp.efi -i <Image.bin>

Arguments	Buffer - Buffer Buffer Length – Length of buffer
-----------	---



Returns	<p>Partition ID - denotes the partition ID, which could be WLAN (wcod) or language (lcl).</p> <p>WOCD ID = 0x244f4357 and LOCL ID = 0x4C434F4C</p> <p>Flags: Bit 0 of the flags is used to set allow same version update. Other bits are reserved and can be used in the future.</p> <p>IpuUpdatedInfo - Contain the information that is actually used to update the IPU partition.</p> <p>0 = Success</p> <p>Non-zero value = Failure</p>
---------	---

8.3.1 PDT Data (Sensor Calibration Data) Update

```
EFI_STATUS
HeciPdt (
    IN  char           *buffer,
    IN  UINT32         bufferLength
);
```

Purpose: The function performs PDT Data Update i.e. Sensor Calibration Data Update.

Command Line Switch -d needs to be used in order to execute PDT Data Update.

Example for Usage:

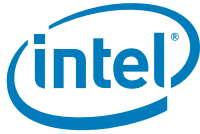
```
FwUpdLclApp.efi -d <Pdt Data Binary>
FWUpdLclApp.efi -d INTC_pdt_SPT_RR3_BOM1_SENSORS
```

Arguments	<p>Buffer - Buffer</p> <p>Buffer Length - Length of buffer</p>
Returns	<p>If Payload is sent to CSME successfully then Send Succeeded Message will be seen.</p>

8.3.2 ISH Firmware Version

```
int
GetPartVersion (
    UINT32 partID,
    UINT16 *major,
    UINT16 *minor,
    UINT16 *hotfix,
    UINT16 *build);
```

Purpose: The function helps retrieve ISH Firmware Version flashed on the platform.



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-Updatable 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 an external LZMA tool for compression of the firmware binary. The LZMA tool is supplied with the ISH binary or may be downloaded from <http://7-zip.org/sdk.html>.
- 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 ME FW kit for details on MEU usages, signing & manifesting flows, etc.

§ §



A Intel® ME 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® Management 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					
MEBx Password	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: ASCII(32) <= char <= ASCII(126) Cannot contain these characters: , : " Must contain for complexity: 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. Refer section 2.7 for format and strong password requirements.	8<=N<=32	Password	ME	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>Note: The FPT command now supports changing individual bits of the OEMSKURule. It is strongly recommended to set the individual bits rather than the full 32 bit value.</p> <p>Note: 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. This NVAR sets OEM Permanent Disable for ALL features. In addition, prior to 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. Refer 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> <table><thead><tr><th>Bit</th><th>Description</th><th>Notes</th></tr></thead><tbody><tr><td>31</td><td>Reserved</td><td></td></tr><tr><td>30</td><td>Reserved</td><td></td></tr><tr><td>29:22</td><td>Reserved</td><td></td></tr><tr><td>21</td><td>TLS</td><td></td></tr><tr><td>20</td><td>DAL</td><td></td></tr><tr><td>19</td><td>Reserved</td><td></td></tr><tr><td>18</td><td>KVM</td><td>2</td></tr><tr><td>17</td><td>Reserved</td><td></td></tr><tr><td>16</td><td>ME Network Disable</td><td></td></tr><tr><td>15:13</td><td>Reserved</td><td></td></tr><tr><td>12</td><td>PAVP</td><td></td></tr><tr><td>11</td><td>Reserved</td><td></td></tr><tr><td>10</td><td>ISH</td><td></td></tr><tr><td>9:6</td><td>Reserved</td><td></td></tr><tr><td>4:5</td><td>Reserved</td><td></td></tr><tr><td>3</td><td>Reserved</td><td></td></tr><tr><td>2</td><td>Manageability and Security Application</td><td>1</td></tr><tr><td>1</td><td>Reserved</td><td></td></tr><tr><td>0</td><td>Manageability Full</td><td>1</td></tr></tbody></table> <p>1. For corporate SKUs bits 0 and 2 need to be both set to '1' to allow for Intel® AMT to work.</p> <p>2. KVM (bit 18) should only be set to '1' when Manageability Application (bit 2) is set to '1'. If using a Corporate SKU, then Manageability Full (bit 0) must also be set to '1'</p>	Bit	Description	Notes	31	Reserved		30	Reserved		29:22	Reserved		21	TLS		20	DAL		19	Reserved		18	KVM	2	17	Reserved		16	ME Network Disable		15:13	Reserved		12	PAVP		11	Reserved		10	ISH		9:6	Reserved		4:5	Reserved		3	Reserved		2	Manageability and Security Application	1	1	Reserved		0	Manageability Full	1	Global	No
Bit	Description	Notes																																																															
31	Reserved																																																																
30	Reserved																																																																
29:22	Reserved																																																																
21	TLS																																																																
20	DAL																																																																
19	Reserved																																																																
18	KVM	2																																																															
17	Reserved																																																																
16	ME Network Disable																																																																
15:13	Reserved																																																																
12	PAVP																																																																
11	Reserved																																																																
10	ISH																																																																
9:6	Reserved																																																																
4:5	Reserved																																																																
3	Reserved																																																																
2	Manageability and Security Application	1																																																															
1	Reserved																																																																
0	Manageability Full	1																																																															



Fixed Offset Name	Description	Data Length (in Bytes)	Expected Value	Reset Type	Mfg. Post EOM/ Pre EOP																		
Feature Shipment Time State	<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.</p> <p>This will not enable functionality that is not capable of working in the target hardware SKU. Refer 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>Note: The FPT command now supports changing individual bits of the Feature Ship State. It is strongly recommended to set the individual bits rather than the full 32 bit value.</p>	4	<p>Feature Enabled: 1 Feature Disabled: 0</p> <table><thead><tr><th>Bit</th><th>Description</th><th>Notes</th></tr></thead><tbody><tr><td>31:30</td><td>Reserved</td><td></td></tr><tr><td>29</td><td>PTT</td><td></td></tr><tr><td>28:3</td><td>Reserved</td><td></td></tr><tr><td>2</td><td>Manageability Full</td><td></td></tr><tr><td>1:0</td><td>Reserved</td><td></td></tr></tbody></table> <p>Note: When disabling PTT using Feature Shipment Time state NVAR, execute a reset after executing fpt.efi –commit to ensure PTT is disabled completely.</p>	Bit	Description	Notes	31:30	Reserved		29	PTT		28:3	Reserved		2	Manageability Full		1:0	Reserved		Global	Yes
Bit	Description	Notes																					
31:30	Reserved																						
29	PTT																						
28:3	Reserved																						
2	Manageability Full																						
1:0	Reserved																						
SetWLANPowerWell	Sets which power well the board uses for WLAN cards	4	<p>0x80 = Disabled 0x81 = Core Well SLP_S3 0x82 = Primary Well SLP_SUS 0x83 = ME Well SLP_A 0x86 = WLAN Sleep via SLP_WLAN#</p>	Global	No																		
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
GpioNvar	GPIO	60	<p>GPIO groups and pad range for each</p> <p>grp pad#</p> <p>GPP_A 0-16</p> <p>GPP_B 0-23</p> <p>GPP_C 0-23</p> <p>GPP_D 0-23</p> <p>GPP_E 0-23</p> <p>GPP_F 0-23</p> <p>GPP_G 0-7</p> <p>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. The format for updating the GPIO is as follows...</p> <p>GpioNvar = 0x0000000003000000110000000010000 00000000000100000002000000170000 0001000000000000000008000000030000 0013000000010000000000000000</p> <p>RST = GPP_D_17 IRQ = GPP_C_23 DFU = GPP_D_19</p>	ME	No
FWUpdLcl	Enabled Firmware Update Local Capability	1	<p>0 = disabled</p> <p>1 = enabled</p>	Global	Yes
EDP_PORT_CFG	EDP Port Configuration. Up to two ports can be enabled 0x00 - 0x01 - A 0x02 - B 0x04 - C 0x08 - D 0x10 - E	1	<p>0x00 0x01</p> <p>0x02 0x03</p> <p>0x04 0x05</p> <p>0x06 0x08</p> <p>0x09 0x0A</p> <p>0x0C</p>	Global	No
LSPCON_PORT	LSPCON Port Configuration. 0x00 - 0x02 - B 0x04 - C 0x08 - D	1	<p>0x00</p> <p>0x02</p> <p>0x04</p> <p>0x08</p>	Global	No
URTC	UnConfigure On RTC	1	<p>0 = Disabled</p> <p>1 = Enabled</p>	ME	No



Fixed Offset Name	Description	Data Length (in Bytes)	Expected Value	Reset Type	Mfg. Post EOM/ Pre EOP
DAM	DAM is a feature that allows the SUT to prepare for unlock without actually enabling debug interfaces	1	0 = Disabled 1 = Enabled	ME	No
AMT Related NVARS					
OEM Customizable Certificate 1	Cert Hash Data. Refer Certificate Hash Entry Structure definition Note: If the platform is un-configured the Certificate Hash will be deleted.	55 => n >= 99	Valid Certificate Hash Entry (SHA1, SHA256 or SHA384)	ME	Yes
OEM Customizable Certificate 2	Cert Hash Data. Refer Certificate Hash Entry Structure definition Note: If the platform is un-configured the Certificate Hash will be deleted.	55 => n >= 99	Valid Certificate Hash Entry (SHA1, SHA256 or SHA384)	ME	Yes
OEM Customizable Certificate 3	Cert Hash Data. Refer Certificate Hash Entry Structure definition Note: If the platform is un-configured the Certificate Hash will be deleted.	55 => n >= 99	Valid Certificate Hash Entry (SHA1, SHA256 or SHA384)	ME	Yes
Privacy/ Security Level	Redirection (KVM, SOL, IDE-r) privacy level and configuration (RCFG, CCM) settings. Note: Setting Privacy Level to its default value would cause NVARS to be reverted to their defaults disregarding changes committed to them	1	Default 0x01 Enhanced 0x02 Extreme 0x03 Default: SOL enabled = true IDER enabled = true KVM enabled = true Opt-in can be disabled= true KVM opt-in configurable remotely = true RCFG and CCM = true Enhanced: SOL enabled = true IDER enabled = true KVM enabled = true Opt-in can be disabled= false Opt-in configurable remotely = true RCFG and CCM = true Extreme: SOL enabled = false IDER enabled = false KVM enabled = false Opt-in can be disabled= false KVM opt-in configurable remotely = N/A RCFG and CCM = false	ME	No
EHBC State	Embedded Host Based Configuration State	1	0 = Disabled 1 = Enabled	ME	No



Fixed Offset Name	Description	Data Length (in Bytes)	Expected Value	Reset Type	Mfg. Post EOM/ Pre EOP
ScreenBlankingEn	Screen Blanking Enabled	1	0 = Disabled 1 = Enabled	ME	No
PKI DNS Suffix	PKI DNS Suffix. Null terminated string	32	PKI DNS Suffix in dotted string format Example: "intelFVE.com" Note: dots are acceptable only in the middle of the string	ME	Yes
CfgSrvFqdn	Configuration Server FQDN (Fully Qualified Domain Name)	256	Example: "intelFVE.com"	ME	Yes
Rcfg	R Configuration	1	0 = Disabled 1 = Enabled	ME	Yes
*Redirection	This is a bit-field indicating the enable/disable status of Storage Redirection, SOL, and KVM features in Intel® AMT. bit[0]: 1 – Storage Redirection enabled, 0 – disabled bit[1]: 1 – SOL enabled, 0 – disabled bit[2]: 1 – KVM enabled, 0 – disabled	4	Range: 0-7 Example: Value of 4 (100b) indicates that KVM is enabled. Value of 3 (011b) indicates that Storage Redirection, and SOL are enabled. Value of 7 (111b) indicates that Storage Redirection, SOL, and KVM are enabled.	ME	Yes
*OptinPolicy	Change User Opt-in (lower nibble). NONE = 0, KVM = 1, ALL = F Disable Opt-In Configurable from Remote IT (upper nibble). 0 - Opt-in is NOT Configurable from Remote IT 1 - Opt-in is Configurable from Remote IT	1	0x00 0x10 0x01 0x11 0x0F 0x1F Examples: In addition to the following, the values may not be configured remotely: Value of 0x00 indicates User Consent is not required. Value of 0x01 indicates User Consent is required for KVM only. Value of 0x0F indicates User Consent is required for (ALL). In addition to the following, the values may be configured remotely: Value of 0x10 indicates User Consent is not required. Value of 0x11 indicates User Consent is required for KVM only. Value of 0x1F indicates User Consent is required for (ALL).	ME	Yes
HostName	Set Host Name Only	64	SkyLake SunrisePoint	ME	Yes
DomainName	Set Domain Name Only	192	myserver.intel.com amr.corp.intel.com www.intel.com mymail.somecollege.edu	ME	Yes



Fixed Offset Name	Description	Data Length (in Bytes)	Expected Value	Reset Type	Mfg. Post EOM/ Pre EOP
CfgSrvAdr	Set Provisioning Server (IPv4/IPv6) Address	60	Example of IPV4: 192.168.1.200 255.255.255.0	ME	Yes
CfgSrvPort	Set Provisioning Server (IPv4/IPv6) Port	2	Within Range: 0 – 0xFFFF	ME	Yes
DisCertHash	Disable all Pre-Installed Certificate Hashes	1	0 = Disabled 1 = Enabled	ME	Yes
IdleTO	Change the Idle Timeout in minutes	2	Within Range: 1 – 0xFFFF	ME	Yes
AmtWdAutoReset	Intel® AMT Watchdog Automatic Reset enabled	1	0 = disabled 1 = Enabled	ME	No
Revenue Sharing Related NVAR Descriptions					
ODM_ID	NVAR used for setting the ODM ID Used by Intel® Services Note: This value can only be programmed into FW once.	4	32-bit value Value 0x00000000 < n < 0xFFFFFFFF	ME	No
SystemIntegratorID	Used for setting the System Integrator ID used by Intel® Services Note: This value can only be programmed into FW once.	4	32-bit value Value 0x00000000 < n < 0xFFFFFFFF	ME	No
ReservedID	Used for setting the "Reserved" ID used by Intel® Services Note: This value can only be programmed into FW once.	4	32-bit value Value 0x00000000 < n < 0xFFFFFFFF	ME	No
Field Programmable Fuses					
PTTEnable	Enables / Disables the fTPM / PTT FPFs	1	0 = Disabled 1 = Enabled	ME	No

- Indicates: Intel AMT KVM not supported if both HDCP Internal Display Ports (A, B, C, and D) are configured.

Note: Settings of all AMT Related parameters (All NVARs Listed under AMT Related NVARs Section) will be supported when Intel® AMT is in pre-provisioned mode only. Otherwise the settings will be ignored.

§ §



B Tool Detail Error Codes

B.1 Common Error Code for All Tools

Error Code	Error Message	Response
0	Success	
1		
2	Failure. Unexpected error occurred	Contact Intel
3	Internal Error. Unexpected error occurred	Contact Intel
4	Unsupported OS	
5	Memory allocation error occurred	Make sure there is enough memory in the system
6	Error accessing the function "GetSystemFirmwareTable" from "kernel32.dll"	
7	The function "GetSystemFirmwareTable" failed with Windows Error Code: %d	
8	Error accessing the "kernel32.dll"	
9	Error log is not created because tool is likely running on a read-only media	
10	Error occurred while reading the file "%s"	
11	Error getting current working directory path:	
12	Error getting current working directory permissions:	
13	An unknown error occurred while opening the file	
14	An unknown error occurred while working with the file "%s".	
15	Error occurred while writing to the file "%s"	
21	General error	
22	Cannot locate ME device	
23	Memory access failure	
24	Write register failure	
25	OS failed to allocate memory	
26	Circular buffer overflow	
27	Not enough memory in circular buffer	
28	Communication error between application and Intel(R) ME module	Check that PCI 64 Bit resource Allocation is disabled in BIOS settings for DOS version of tools.



Error Code	Error Message	Response
29	Unsupported HECI bus message protocol version	
31	AMT device unavailable	Check that PCI 64 Bit resource Allocation is disabled in BIOS settings for DOS version of tools.
32	Unexpected result in command response	Contact Intel
33	Unsupported message type	
34	Cannot find host client	
35	Cannot find ME client	
36	Failure occurred during ME disconnect	
37	Client already connected	
38	No free connection available	
39	Illegal parameter	
40	Flow control error	
41	No message	
42	Requesting Intel(R) MEI receive buffer size is too large	
43	Application or driver internal error	
45	Command is not supported	
46	Invalid command line option(s)	
47	The following Parameter is not a valid option: %s	
48	Internal Error (PCH is not supported)	
49	Internal Error (Safe function wrapper error: Invalid size)	
50	Internal Error (Safe function wrapper error: compose string from list)	
51	Internal Error (Safe function wrapper error: compose string)	
52	Internal Error (Safe function wrapper error: memncpy)	
53	Internal Error (Safe function wrapper error: strncpy)	
54	Internal Error (Safe function wrapper error: strncat)	
55	Internal Error (Safe function wrapper error: strtok)	
56	Printf function failed	
57	Failed getting variable "%s" value	
58	The variable "%s" is supported on Corporate SKU only.	
59	"%s".	



Error Code	Error Message	Response
60	Unable to find matching LOCL	
62	Internal Error (Plat info general failure)	
63	Could not access PCI device	
64	Operation is not supported	
65	Internal Error (OS interace: Illegal Param)	
66	Unexpected Failure.	
67	Unable to load library	
68	Unable to change permission	
69	Unable to perform request due to permission failure	
71	Unable to perform CreateFile	
72	The FPF compare failed	
73	Retrieving the FPF Data failed	
74	The CSE File Component requested, "%s", is not valid for this operation.	
75	The CSE File Component requested, ID:"%d", is not valid for this operation.	
78	Communication error between application and Intel(R) ME module (AMT client)	
79	Communication error between application and Intel(R) AMT module (PTHI client)	
80	Communication error between application and Intel(R) ME module (ICLS client)	
81	Fail to read FW Status Register value	
82	Fail to create verbose log file %s	
83	Internal error (Could not determine FW features information)	
84	Unknown or unsupported hardware platform	
85	Communication error between application and Intel(R) ME module (HCI client)	
86	Communication error between application and Intel(R) ME module (FPF client)	
87	Communication error between application and Intel(R) ME module (NFC client)	
88	Communication error between application and Intel(R) ME module (FWU client)	
89	Failed to communicate with CSME. This tool must be run from a privileged account (administrator/root).	



Error Code	Error Message	Response
90	%s feature was not found.	
91	%s feature was not available.	
92	%s actual value is - %s.	
93	Error reporting revenue share information - Invalid index used	
94	Error reporting revenue share information - Index already in use	
95	This slot is unused	
96	%s feature was not available because ME is not in manufacturing mode.	
97	Failed to locate DATA manifest marker	
98	Failed to locate PID module entry	
99	End of file encountered when reading first record	
100	Invalid PID Section Data Marker	
101	This PID cannot be used since the PID matches the known PID for Pre-Production PCHs	
102	Non-Intel chipset found in first record	
103	No PID list marker found	
105	Internal Error (Failed to initialize platinfo)	
106	Error occurred while communicating with SPI device	Check SPI Device
107	Failed to create dependency list for features	
108	Failed to communicate with CSME. This tool must be run from a privileged account (administrator/root).	
109	Unsupported ME Firmware Version	
110	Failed to retrieve feature	
111	Received Status from ME: FPF File Invalid	
112	Internal Error (Failed to check eligibility)	
113	Internal Error (Unexpected error occurred)	
114	Failed to retrieve Intel (R) FIT version	
115	Failed to retrieve Intel (R) Internal Build Version	
116	Unknown or unsupported hardware platform	
117	Unsupported ME Firmware Version %i.%i.%i.%i %s	
118	MEManuf Operation Failed	



Error Code	Error Message	Response
120	Failed to communicate with CSME. This tool must be run from a privileged account (administrator/root).	
121	Communication error between application and Intel(R) ME module (BIST client)	
122	Communication error between application and Intel(R) ME module (AMT client)	
123	Fail to read FW Status Register value	
125	Fail to retrieve error list	
126	Internal error (Could not determine FW features information)	
127	Access to SPI Flash device(s) failed	
128	Failed to create dependency list for features	
129	Intel(R) test failed to start, error 0x%X returned	
130	Intel(R) test timeout (exceeded 30 seconds)	
131	Intel(R) ME test is currently running, try again later	
132	MEManuf End-Of-Line Test config file generation failed	
133	M3 results are not available from SPI. Please run - test option to perform the BIST test	
134	Could not read M3 results from SPI	
135	Internal Error	
136	Internal error - Invalid Parameters	
137	SMBus hardware is not ready	
138	Internal error - SMBus Read Byte PEC failure	
139	SMBus encountered time-out	
140	Internal Error. FW returns status %d	
141	Internal error - Failed to match	
142	Internal error - Out of memory	
143	Internal error - Unable to get current PP	
144	Failed to retrieve test result from SPI	
145	Failed to retrieve power package setting	
146	Failed to retrieve power rule from SPI	
147	WLAN power well setting is set incorrectly	
148	Failed to retrieve test result from SPI	
149	Internal error - Failed to retrieve Platform Attribute	



Error Code	Error Message	Response
150	Failed to retrieve PROC_MISSING NVAR setting	
151	PROC_MISSING NVAR setting is set incorrectly	
156	Failed to retrieve password from SPI	
157	Internal error - Password length is incorrect	
158	Internal error - Modified local password	
159	Internal error - Invalid password	
160	Firmware is in recovery mode	
161	Boot Guard Self Test Failed	
162	Intel integrated LAN setting is set incorrectly	
163	Intel LAN Connected Device (PHY) physical connectivity error with ME	
164	Internal error - Illegal data length	
165	Internal error - Illegal data value	
166	EHBC State Test Failed - Error while reading data from flash	
167	EHBC State Test Failed - Contradiction with current Privacy Level	
168	Current WLAN does not match micro-code, please update WLAN micro-code in FW	
169	Communication with WLAN device failed	
170	Length of OEM Customizable Certificate Friendly Name setting is set incorrectly	
171	OEM Customizable Certificate Stream setting is set incorrectly	
172	OEM Customizable Certificate Hash Algorithm setting is set incorrectly	
173	Length of OEM Customizable Certificate Stream is set incorrectly	
174	Internal error - Unable to compress	
175	The compressed data is incorrect	
176	USB EHCI 1 Enabled and/or USB EHCI 2 Enabled setting is set incorrectly	
177	KVM device is already in use by other components	
178	Internal error	
179	Failed to retrieve power source	
180	Power source is not AC	
194	LAN power well setting is set incorrectly	
195	WLAN power well setting is set incorrectly	



Error Code	Error Message	Response
197	Intel(R) ME firmware version mismatch, actual value is - %s	
198	Intel(R) Gbe version mismatch, actual value is - %s	
199	BIOS version mismatch, actual value is - %s	
200	System UUID mismatch, actual value is - %s	
201	System UUID mismatch, feature is not supported	
202	System UUID actual value is all 0x00	
203	System UUID actual value is all 0xFF	
204	Intel(R) Wired LAN MAC address mismatch, actual value is - %s	
205	Intel(R) Wireless LAN MAC address mismatch, actual value is - %s	
206	Security Descriptor Override Strap (SDO) is enabled	
207	End-Of-Post message is not sent	
208	Unable to determine Intel(R) ME Manufacturing Mode status	
209	Intel(R) ME is still in Manufacturing Mode	
210	CF9GR is not locked	
211	Region access permissions don't match given values	
212	BIOS has granted Intel(R) Gbe and/or ME access to its region	
213	SPI flash descriptor region is not locked	
214	Intel(R) Gbe/ME has read or write access to BIOS region	
215	SPI flash Intel(R) ME region is not locked	
216	Wireless LAN micro-code mismatch, actual value is - %s	
217	%s have not been set	
218	%s mismatch, actual value is - %s	
219	Variable %s mismatch, actual value is - %s	
220	Memory allocation failed for checking variable	
221	Invalid test name -	
222	Internal error	
223	Cannot run the command since Intel(R) AMT is not available	
224	No valid OEM ICC data programmed	



Error Code	Error Message	Response
225	MFS is corrupted	
226	Using wrong PCH SKU Emulation via Intel (R) FIT vs whats the actual HW Type	
227	FPFs are not committed	
228	FPFs and ME Config mismatch	
229	Internal error	
230	Cannot perform hibernation. Please manually reboot the system	
231	MEManuf Test Failed	
232	No Intel(R) ME test result to retrieve	
233	MEManuf Operation Passed (with warnings)	
234	Test is enabled by the user but is unknown by the platform -	
235	Some Intel(R) recommended checks have been skipped:	
236	Attempting to add sibling to XML root node	
237	File size is zero	
238	XML parsing failed	
239	XML parsing encountered data overflow	
240	Invalid XML error code conversion	
241	XML parser - out of memory error	
281	Failed to retrieve list of BIST tests to run from FW	
282	Unexpected failure when retrieving BIST results	
283	Test is enabled by the user but is not supported by the current sku -	
285	Retrieving the EOL Config list of tests failed.	
286	Retrieving the EOL Var list of tests failed.	
287	No name attribute specified for test:	
288	Failed to parse configuration file provided.	
289	No output file path specified to write configuration file.	
290	No data to write to configuration file.	
291	Invalid ErrAction specified:	
292	Invalid node name.	
293	The 2 SPI flash devices do not have compatible command sets.	
294	No SPI flash device could be identified. Please verify if Fparts.txt has support	Verify that the device is listed in Fparts.txt.



Error Code	Error Message	Response
295	"%s" file not found.	Check file location.
296	Access was denied opening the file "%s"	Check file location.
297	An unknown error occurred while opening the file "%s"	Check system memory. Verify the file is not corrupt.
298	Failed to allocate memory for the flash part definition file "%s"	Check system memory. Verify the file is not corrupt.
299	Failed to read the entire file "%s" into memory.	Check system memory. Verify the file is not corrupt.
300	Parsing of file "%s" failed.	Check system memory. Verify the file is not corrupt.
301	Protected Range Registers are currently set by BIOS, preventing flash access. Please contact the target system BIOS vendor for an option to disable Protected Range Registers.	Assert Flash Descriptor Override Strap (GPIO33) to Low, Power Cycle, and Retry. If Protected Range Registers (memory location: SPIBAR + 74h -> 8Fh) are still set, contact the target BIOS vendor.
302	Hardware sequencing failed. Make sure that you have access to target flash area!	Check descriptor region access settings.
303	The host CPU does not have read access to the target flash area. To enable read access for this operation you must modify the descriptor settings to give host access to this region.	Check descriptor region access settings.
304	An attempt was made to read beyond the end of flash memory	Check Address.
305	Software sequencing failed. Make sure that you have access to target flash area!	Check descriptor region access settings.
306	Invalid Block Erase Size value in "%s".	
307	Invalid Write Granularity value in "%s".	
308	Invalid Enable Write Status Register Command value in "%s".	
310	Internal Error.	
311	The supplied zero-based index of the SPI Device is out of range.	
312	Invalid descriptor region.	Check descriptor region.
313	Region does not exist.	Check region to be programmed.
314	An attempt was made to write beyond the end of flash memory	Check Address
315	An attempt was made to erase beyond the end of flash memory	Check Address
316	General Erase failure.	Attempt the command again. If it fails, contact Intel.
317	The address 0x%08X of the block to erase is not aligned correctly.	Check Address



Error Code	Error Message	Response
318	The host CPU does not have erase access to the target flash area. To enable erase access for this operation you must modify the descriptor settings to give host access to this region.	Check descriptor region access settings.
320	Hardware timeout occurred in SPI device.	Check SPI device
321	There are no supported SPI flash devices installed. Please check connectivity	Verify Fparts.txt has correct values. Check SPI Device.
322	Unrecognized value in the HSFSTS register.	
323	AEL is not equal to zero.	
324	FCERR is not equal to zero.	
327	Checking variable "%s" failed.	
329	Invalid Manufacturing Line Configurable variable name "%s".	
331	"%s" file does not exist.	
333	End Of Manufacturing Operation failure - Verification failure on Descriptor Lock settings.	
334	Unable to get master base address from the descriptor.	
335	Password does not match the criteria.	
336	Invalid length of Manufacturing Line Configurable value. Check configuration file for correct length.	
337	Invalid hexadecimal value entered for the Manufacturing Line Configurable.	
338	Invalid hash certificate file.	
339	An unknown error occurred while opening the file "%s".	
340	End Of Manufacturing Operation failure - Verification failure on ME Manufacturing Mode Done settings.	
341	The Global Lock Bit has already been set.	
342	Not able to open the file "%s".	
343	Error occurred while reading the file "%s".	
345	Invalid parameter value specified by user. Use -? option to see help.	
346	Failed to communicate with CSME. This tool must be run from a privileged account (administrator/root).	
347	[%s] cannot be run on the current platform. Please contact your vendor.	
349	ME disabled	



Error Code	Error Message	Response
350	Failed to get information about the installed flash devices!	
351	An error occurred reading the flash descriptor signature.	
352	Flash descriptor does not have correct signature.	
354	The attempt to commit the Manufacturing Line Configurables has failed.	
355	"%s" is not a valid file name.	
356	Image file "%s" is not found.	
357	Access was denied opening the file "%s".	
358	Failed to read the entire file into memory. File: %s	Check system memory. Verify the file is not corrupt.
359	The address is outside the boundaries of the flash area.	
360	Unable to write data to flash. Address 0x%x.	
361	Data verify mismatch found.	
362	Access was denied creating the file "%s".	
363	An unknown error occurred while creating %s.	
364	Failed to write the entire flash contents to file.	
365	General Read failure.	
366	An error occurred reading the flash mapping data.	
367	System booted in Non-Descriptor mode, but the flash appears to contain a valid signature.	
368	An error occurred reading the flash components data.	
369	An error occurred reading the flash region base/limit data.	
370	An error occurred reading the flash master access data.	
371	Flash is not blank.	
372	The Close Manufacturing process failed.	Check that PCI 64 Bit resource Allocation is disabled in BIOS settings for DOS version of tools.
373	Setting Global Reset Failed	
374	ME disable not needed	
375	ME already disabled	
376	The request to disable the ME failed.	
377	There was a communications error between FPT and the ME.	



Error Code	Error Message	Response
378	There is a problem with the GbE binary which prevents saving the data.	
381	No FPF Name provided.	
382	No value to compare against.	
383	A required parameter is missing.	
386	Committing the FPF is not allowed at this time.	
387	The FPF has already been committed.	
388	FPF Commit Failed.	
389	Committing a specific FPF is not supported. Consider committing all the FPFs.	
407	Keybox file open failed.	
408	Keybox file size invalid.	
409	Invalid keybox API version in provisioning response.	
410	Invalid keybox command id in provisioning response.	
411	Invalid keybox buffer length in provisioning response.	
412	CEK is invalid.	
413	CEK is not available.	
414	Cannot provision after EOM.	
415	Unknown error for provisioning status.	
416	Bad CRC.	
417	Bad Magic.	
418	Invalid keybox status in provisioning response.	
419	Invalid keybox API version in provisioning response.	
420	Invalid keybox command id in provisioning response.	
421	Invalid keybox buffer length in provisioning response.	
422	Keybox is not provisioned.	
423	The host CPU does not have write access to the target flash area. To enable write access for this operation you must modify the descriptor settings to give host access to this region.	Check descriptor region access settings.
425	User elected to cancel the operation.	
426	Confirmation is not received from the user to perform operation.	



Error Code	Error Message	Response
427	User response was invalid. Please select from the correct responses.	
428	Open Process Token error %d	
429	Adjust Token Privileges error %d	
430	Initiate System Shutdown error %d	
431	Graceful Exit of Windows has failed.	
432	Error determining possible system states.	
434	Cannot locate MEI driver	
435	A test returned from FW does not match known test by the tool: App(%d) Comp(%d) Test(%d)	
439	Invalid "State" found for test -	
440	ISH Internal Error	
441	IUP Not Found	
442	Attempt to communicate to ISH library failed	
443	ISH error. Library status: 0x%08X	
444	ISH error. Unexpected Failure. Please make sure to use correct DLL version.	
445	ISH error. Invalid level selected.	
446	ISH Test failed	
447	Error retrieving ISH configuration	
448	Communication error between application and Intel(R) ME module (Intel(R) Precise Technology client).	
449	Cannot locate HID device.	
450	Incorrect Report ID received.	
451	MCTP SMBUS test failed	
453	Invalid config file. "State" was not found for test -	
454	Invalid config file. "RequiredValue" was not found for test -	
455	Invalid config file. "ErrAction" was not found for test -	
457	Unable to validate address range	
458	Memory window not set or device is not armed for operation	
459	Sensor could not be found. Either no sensor is connected, the sensor has not yet initialized, or the system is improperly configured.	
460	Not enough memory/storage for requested operation	



Error Code	Error Message	Response
461	Unexpected error occurred	
462	Used in TOUCH_SENSOR_HID_READY_FOR_DATA_RSP to indicate sensor has been disabled or reset and must be reinitialized.	
463	Used to indicate compatibility revision check between sensor and ME failed, or protocol ver between ME/HID/Kernels failed.	
464	Indicates sensor went through an unexpected reset	
465	Requested sensor reset failed to complete	
466	Operation timed out	
467	Test mode pattern did not match expected values	
468	Indicates sensor reported fatal error during reset sequence. Further progress is not possible.	
469	Indicates sensor reported non-fatal error during reset sequence. HID/BIOS logs error and attempts to continue.	
470	Indicates sensor reported invalid capabilities, such as not supporting required minimum frequency or I/O mode.	
471	Indicates that command cannot be complete until ongoing Quiesce I/O flow has completed.	
472	Touch - Vendor ID mismatch, actual value is - %s	
473	Cannot find the NVAR file; the system maybe in EOM.	
476	Cannot access the NVAR file attributes.	
477	Cannot access the CVAR file; No CVAR files to commit.	
478	Cannot modify the NVAR file.	
479	Cannot access the CVAR file; file is not allowed to be changed after EOM.	
480	Used an invalid input parameter to access the NVAR file.	
484	FPF is not written.	
491	NVAR access in the system caused a general error.	
492	Some platform eligible tests have been skipped:	
493	HDCP file invalid.	
496	HDCP receive message failed.	
497	Failure. Unexpected error occurred.	
498	Can not provision after EOM.	
499	Certificate verification failed.	



Error Code	Error Message	Response
500	HDCP Rx is not provisioned.	
502	Invalid string value entered for the Manufacturing Line Configurable.	
504	Fail to read FW Status Register value	
505	Detected ME in recovery mode.	
508	FW returned status: Erase token failure.	
510	Detected invalid data size.	
511	Detected invalid hex value.	
512	Integrated Sensor Solution Test Returned Warning.	
513	Variable "%s" found, but unable to display by design.	
514	Failed to retrieve 5K port setting	
515	Failed to retrieve LSPCON Port setting	
516	Display port settings are not correct	
518	Unexpected size found in the file "%s". Expected: 0x%X. Received: 0x%X.	
519	Keybox is not provisioned.	
520	Unexpected keybox failure.	
521	Unable to execute command in this Firmware State. Please reboot.	
523	File "%s" contains GPIO pin assignments that are not multiples of the GPIO pin data structure.	
525	Mismatch on FPF file %s - UEP: %s, FPF HW: %s.	
526	Valid bit FPF is not set on file %s.	
527	FPFs are not committed to HW.	
529	Failed to read FPF HW.	
530	SOC Config Lock is not set.	
531	Lock bit FPF is not set on file %s.	
532	Failed to read FPF in UEP.	
533	FW Update OEM ID incorrectly set to 00 or FF.	
534	Unable to determine FW Update OEM ID status	
535	BIOS Region access permissions don't match Intel recommended values	
536	ME Region access permissions don't match Intel recommended values	
537	GBE Region access permissions don't match Intel recommended values	



Error Code	Error Message	Response
538	EC Region access permissions don't match Intel recommended values	
539	RPMC SPI device did not initialize RPMC support correctly, RPMC SPI device needs replacement/refurbishment.	
540	RPMC SPI device has not been bound to the platform yet, RPMC manufacturing process is not complete.	
541	Invalid Parameter.	
542	"HW Binding" state is not enabled	
544	The "%s" var is not updatable.	
545	Invalid Manufacturing Line Configurable variables file	
546	Failed to locate IntegratedTouch protocol.	
547	The variable "%s" is not supported on this platform.	
548	Warning: There are some addresses that are not defined in any regions. Read/Write/Erase operations are not possible on those addresses.	
552	EOM flow incomplete. Close Manufacturing process was canceled by user.	
553	FPT Operation Successful (with warnings).	
554	PCH is unlocked. Disable Delayed Authentication Mode and retry	
555	Configuration file "%s" not found.	
557	Test required value format is not valid.	
558	Invalid BootGuard configuration.	

B.2 Firmware Update Errors

Error Code	Error Message
0	Success
1	An internal error to the AMT device has occurred haltrcfg related.



Error Code	Error Message
2	Intel® AMT Status is not ready.
3	Invalid Intel® AMT Mode.
4	An internal error to the Intel® AMT device has occurred.
8193	Intel® ME Interface: Cannot locate Intel® ME device driver.
8704	Firmware update operation not initiated due to a SKU mismatch.
8705	Firmware update not initiated due to version mismatch.
8706	Firmware update not initiated due to integrity failure or invalid FW image.
8707	Firmware update failed due to an internal error.
8708	Firmware Update operation not initiated because a firmware update is already in progress.
8710	Firmware update tool failed due to insufficient memory.
8713	Firmware update not initiated due to an invalid FW image header.
8714	Firmware update not initiated due to file open or read failure.
8716	Invalid usage.
8718	Update operation timed-out; cannot determine if the operation succeeded.
8719	Firmware update cannot be initiated because Local Firmware update is disabled.
8722	Intel® ME Interface: Unsupported message type
8723	No Firmware update is happening.
8724	Platform did not respond to update request.
8725	Failed to receive last update status from the firmware
8727	Firmware update tool failed to get the firmware parameters
8728	This version of the Intel I® FW Update Tool is not compatible with the current platform.
8741	FW Update Failed.
8743	Unknown or unsupported Platform.
8744	OEM ID verification failed.
8745	Firmware update cannot be initiated because the OEM ID provided is incorrect
8746	Firmware update not initiated due to invalid image length
8747	Firmware update not initiated due to an unavailable global buffer.
8748	Firmware update not initiated due to invalid firmware parameters.



Error Code	Error Message
8754	Encountered error writing to file.
8757	Display FW Version failed.
8758	The image provided is not supported by the platform.
8759	Internal Error.
8760	Update downgrade vetoed.
8761	Firmware write file failure.
8762	Firmware read file failure.
8763	Firmware delete file failure.
8764	Partition layout NOT compatible.
8765	Downgrade NOT allowed, data mismatched.
8766	Password did not match.
8768	Password Not provided when required.
8769	Polling for FW Update Failed.
8772	Invalid usage, -allowsv switch required to update the same version firmware.
8778	Unable to read FW version from file. Verify the update image used.
8787	Password exceeded maximum number of retries.
8937	Update between engineering build vs regular build is not allowed. Both builds have to be the same type: regular or engineering build. Engineering build is 7000 and above. Regular build is below 7000.

B.3 FWUpdate API Library Errors

Error Code	Error Message	Response
0	FWU_ERROR_SUCCESS	N/A
8193	HECI Device not Found	Perform Reset, Try Running FWUpdate Process Again. Try 3 times else Contact Intel for this Error
8199	Failure to send or receive messages to HECI to get Status Info	Perform Reset, Try Running FWUpdate Process Again. Try 3 times else Contact Intel for this Error
8204	HECI message has incorrect message type	Contact Intel for this Error



Error Code	Error Message	Response
8213	HECI Buffer Size is Small Error	Contact Intel for this Error
8707	Internal error within the library	Contact Intel for this Error
8710	FWU_NO_MEMORY - Insufficient memory Error	Contact Intel for this Error
8713	Invalid Image file header	Contact Intel for this Error
8714	FWU_FILE_OPEN - Failed to open input file	Make sure Appropriate file is used for updating the image. E.g. On Consumer Platform, ME Consumer Image should be used and not Corporate.
8727	Failure to send or receive HECI messages to HECI client	Perform Reset, Try running FWUpdate Process Again. Try 3 times else Contact Intel for this Error
8741	FW returns incorrect Message Type or wrong image ordering	Check ME Image to be updated is appropriate. Try Running FWUpdate Process Again
8746	Invalid image size	Make sure Image passed for FWUpdate is of appropriate size. E.g. Consumer Image, size should be 1,968 KB and for Corporate Image, size should be 7,160 KB
8747	Buffer not available	Make sure enough buffer is available
8748	Invalid parameters sent to Firmware	Contact Intel for this Error
8761	Firmware write file failure	Contact Intel for this error. Potential Cause: After Image is passed to the firmware, firmware is seeing issues with file write or Inappropriate Image is passed in FWUpdate Tool.
8762	Firmware read file failure	Contact Intel for this error. Potential Cause: After Image is passed to the firmware, firmware is seeing issues with file read or Inappropriate Image is passed in FWUpdate Tool.
8763	Firmware delete file failure	Contact Intel for this error. Potential Cause: After Image is passed to the firmware, firmware is seeing issues with file delete or Inappropriate Image is passed in FWUpdate Tool.
8764	Partition layout not compatible	Contact Intel for this Error
8771	Invalid file used for partial FW update (only FULL and Partial images are supported)	Contact Intel for this Error



Error Code	Error Message	Response
8776	Failure in Send or Receive of the Get Partition Attribute Command. Or even when FW returns an error status after receiving command	Contact Intel for this Error
8778	The partition ID requested for update is not expected by the FW	N/A
8793	FW Update/downgrade is not allowed to the supplied FW image	N/A
8794	FW downgrade is not allowed due to SVN restriction	N/A

§ §



C Tool Option Dependency on BIOS/Intel® ME Status

Tools' Options	Intel® ME End-of-Manufacturing NVAR		End of Post		CF9GR Locking	
	Set	Not Set	Yes	No	Yes	No
FPT -Greset	Not related	Not related	Not related	N/A Not related	Fail – DOS	Work
FPT –R	Depends on End of post status	Work	Depends on Intel® ME manufacturing mode donebit status	Work	Not related	Not related
Intel® MEINFO – EOL config	Depends on End of post status	Work	Depends on Intel® ME manufacturing mode donebit status	Work	Not related	Not related
All options for UpdPARAM	Not related	Not related	Fail	Work	Not related	Not related

§ §