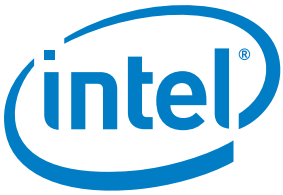


System Tools for Intel® 8 Series Chipset Family Intel® Management Engine Firmware 9.0 SKU's

User Guide

October 2013

Revision: 1.0

Intel Confidential

Contents

Figures

Tables

Revision History

Revision	Description	Date
1.0	Final Revision	October 2013

§

1 Introduction

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

1.1 Terminology

Acronym/Term	Definition
3PDS	3rd Party Data Storage
AC	Alternating Current
Agent	Software that runs on a client PC with OS running
API	Application Programming Interface
ASCII	American Standard Code for Information Interchange
BBBS	BIOS Boot Block Size
BIN	Binary file
BIOS	Basic Input Output System
BIOS-FW	Basic Input Output System Firmware
BIST	Built In Self Test
CCM	Client Control Mode (Host Based Setup and Configuration)
CLI	Command Line Interface
CPT	Cougar Point
CPU	Central Processing Unit
CRB	Customer Reference Board
DHCP	Dynamic Host Configuration Protocol
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

Acronym/Term	Definition
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
FITC	Flash Image Tool
FLOCKDN	Flash Configuration Lock-Down
FMBA	Flash Master Base Address
FOV	Fixed Offset Variable
FPSBA	Flash PCH Strap Base Address
FPT	Flash Programming Tool
FPTW	Flash Programming Tool Window
FQDN	Fully Qualified Domain Name
FRBA	Flash Region Base Address
FW	Firmware
FWUpdate	Firmware Update
G3	A system state of Mechanical Off where all power is disconnected from the system. A G3 power state does not necessarily indicate that RTC power is removed.
GbE	Gigabit Ethernet
PCH	Peripheral Controller Hub
GPIO	General Purpose Input/Output
GUI	Graphical User Interface
GUID	Globally Unique Identifier
HECI (deprecated)	Host Embedded Controller Interface
Host or Host CPU	The processor running the operating system. This is different than the management processor running the Intel® ME FW.
Host Service/ Application	An application running on the host CPU
HostIF	Host Interface
HTTP	HyperText Transfer Protocol
HW	Hardware

Acronym/Term	Definition
IBEN	Input Buffer Enable
IBV	Independent BIOS Vendor
ICC	Integrated Clock Configuration
ID	Identification
IDER	Integrated Drive Electronics Redirection
INF	An information file (.inf) used by Microsoft operating systems that support the Plug & Play feature. When installing a driver, this file provides the OS with the necessary information about driver filenames, driver components, and supported hardware.
Intel® AMT	The Intel® AMT Firmware running on the embedded processor
Intel® AT	Intel® Anti-Theft Technology
Intel® DAL	Intel® Dynamic Application Loader (Intel® DAL)
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® ME Setting Checker Tool
Intel® MEInfoWin	Windows version of Intel® MEINFO
Intel® MEManuf	Intel® MEManuf validates Intel® ME functionality on the manufacturing line
Intel® MEManufWin	Windows version of Intel® MEManuf
ISV	Independent Software Vendor
IT User	Information Technology User. Typically very technical and uses a management console to ensure multiple PCs on a network function.
JEDECID	Joint Electronic Device Engineering Councils ID. Standard Manufacturer's Identification Code that is assigned, maintained and updated by the JEDEC office
JTAG	Joint Test Action Group
KVM	Keyboard, Video, Mouse
LAN	Local Area Network
LED	Light Emitting Diode
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
M0	Intel® ME power state where all HW power planes are activated. Host power state is S0.

Acronym/Term	Definition
M1	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.**
M3	Intel® ME power state where all HW power planes are activated but the host power state is different than S0. (Some host power planes are not activated.) The Host PCI-E* interface is unavailable to the host SW. The main memory is not available for Intel® ME use.
M-Off	No power is applied to the management processor subsystem. Intel® ME is shut down.
MAC address	Media Access Control address
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
OVR	Override
PAVP	Protected Video and Audio Path
PC	Personal Computer
PCH	Platform Controller Hub
PCI	Peripheral Component Interconnect
PCIe*	Peripheral Component Interconnect Express
PDR	Platform Descriptor Region
PHY	Physical Layer

Acronym/Term	Definition
PID	Provisioning ID
PKI	Public Key Infrastructure
PM	Power Management
PRTC	Protected Real Time Clock
PSK	Pre-Shared Key
PSL	PCH Strap Length
RCS	Remote Connectivity Service
RCFG	Remote Configuration
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 state where the host CPU and memory are not active.
S5	A system state where all power to the host system is off but the power cord is still connected.
SDK	Software Development Kit
SEBP	Single Ended Buffer Parameters
SHA	Secure Hash Algorithm
SMB	Small Medium Business mode
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.
Sx	All S states which are different than S0
SW	Software

Acronym/Term	Definition
System States	Operating System power states such as S0, S1, S2, S3, S4, and S5.
TCP/IP	Transmission Control Protocol/Internet Protocol
TLS	Transport Layer Security
UI	User Interface
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
VE	Virtualization Engine
VLAN	Virtual Local Area Network
VSCC	Vendor Specific Component Capabilities
Windows* PE	Windows* Preinstallation Environment
WIP	Work in Progress
WLAN	Wireless Local Area Network
XML	Extensible Markup Language. Intel® AMT's XML-based protocol has 3 parts: An envelope that defines a framework for describing what is in a message and how to process it A set of encoding rules for expressing instances of application-defined data types A convention for representing remote procedure calls and responses
ZTC	Zero Touch Configuration

1.2 Reference Documents

Document	Document No./Location
FW Bring Up Guide	Release kit
Firmware Variable Structures for Intel® Management Engine and Intel® Active Management Technology 9.0	ANACAPA document
PCH EDS	CDI
Intel® 8 Series Chipset Family SPI Programming Guide	Release kit

§

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, see the 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:

- Intel® 8 Series Chipset Family platform – All tools in the Intel® 8 Series Chipset Family FW release kit are designed for Intel® 8 Series Chipset Family platforms only. These tools do not work properly on any other legacy platforms (Santa Rosa, Weybridge, Montevina, McCreary, and Capella/Piketon). Tools designed for other platforms also do not work properly on the Intel® 8 Series Chipset Family platform.
- Intel® vPro™ platform – All features listed in this document are available for Intel® vPro™ platforms with Intel® ME FW 9.0. There are some features that are specifically designed for the Intel® vPro™ platform and only work on it.
- Intel® ME Firmware 9.0 SKU – A common set of tools are provided for the following Intel® ME FW 9.0 SKUs: 1.5MB Intel® ME FW SKU and 5MB Intel® ME FW SKU. The following features are only available for 5MB Intel® ME FW SKUs and 1.5MB 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 1.5MB Intel® ME FW SKU contains a note indicating this.

2.2 ME8 System Tools Changes

Intel developed the following system tools enhancements for ME8 platforms:

- FPT supports the flashing without verifying
- FPT support flashing while retaining the MAC address
- One image for both FITC and FW update.
- FW Update supports partial FW update.

- Intel® MEMANUF will save test result in SPI
- Intel® MEMANUF option changes , no –R, –S4, S5 and new –test option
- Intel® MEMANUF support BIST into early boot

Note: More details are available in each tool's documentation.

2.3 Image Editing Tools

The following tools create and write flash images:

- FITC:

  Combines the Descriptor, GbE, BIOS, PDR, and Intel® ME FW binaries into one image.

  Configures softstraps and NVARs for Intel® ME settings that can be programmed by a flash programming device or the FPT Tool.

- FPT:

  Programs the flash memory of individual regions or the entire flash device.

  Modifies some Intel® ME settings (FOV) 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 SPI image. (**Note:** The firmware update tool provided by Intel only works on the platforms that support this feature.)

2.4 Manufacturing Line Validation Tools

The manufacturing line validation tools (Intel® MEMANUF) allow the Intel® ME and Intel® AMT functionality to be tested immediately after the PCH chipset is generated. These tools are designed to be able to run quickly. They can run on simple operating systems, such as EFI, MS-DOS 6.22, Windows* 98 DOS, FreeDOS, and DRMK DOS. The Windows versions are written to run on Windows* XP (SP1/2), Windows* 7, Windows* 8 and Win* PE32 and 64. These tools are mostly run on the manufacturing line to do manufacturing testing.

2.5 Intel® ME 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.

2.6 Operating System Support

Table 1: OS Support for Tools

Intel® ME and Manufacturing Tools	MS DOS*	Windows* 98 DOS	DRMK DOS	Free DOS	PC DOS Version 7.01	PC DOS Version 7.00	EFI (64bit)	Windows* PE 32 (version 3 & 4)	Windows* PE 64 (version 3 & 4)	Windows* XP 32	Windows* XP 64	Windows* 7 32	Windows* 7 64	Windows* Server 2003/2008 32 With the latest SP	Windows* Server 2003/2008 64 With the latest SP	Windows* 8 32 (MBR & uEFI)	Windows* 8 64 (MBR & uEFI)
FITC										x	x	x	x	x	x	x	x
FPT	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
MEMANUF	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
MEINFO	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
FWUPDLCL	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
UpdParam	x	x	x	x	x	x											

NOTES:

1. 64 bit support does 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.

2.7 Generic System Requirements

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

- Intel® MEI driver.
- Intel® AMT LMS – not applicable to 1.5MB Intel® ME FW SKU.

See the description of each tool for its exact requirements.

Table 2: Tools Summary

Tool Name	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
FPT	Programs the image onto the flash memory	X
FWUpdate	Updates the FW code while maintaining the previously set values	X

2.8 Error Return

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

2.9 Usage of the Double-Quote Character (")

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

FPTEfi –r "^" this is an example"^".

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.

2.10 PMX Driver Limitation

Several tools (Intel® MEINFO, Intel® MEMANUF, and 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.

§

3 Flash Image Tool

The Flash Image tool (**FITC.exe**) creates and configures a complete SPI image file for Intel® 8 Series Chipset Family platforms in the following way:

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

   Binary files of the following regions:

   BIOS

   Intel integrated LAN (GbE)

   Intel® ME

   Platform Descriptor Region

   The Flash Descriptor Region created by FITC
3. The user can manipulate the completed SPI image via a GUI and change the various chipset parameters to match the target hardware. Various configurations can be saved to independent files, so the user does not have to recreate a new image each time.

FITC 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: FITC just generates a complete SPI image file; it does not program the flash device. This complete SPI 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

FITC runs on Windows* XP, Windows* 7, and Windows* 8. The tool does not have to run on an Intel® ME-enabled system.

3.2 Flash Image Details

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

Figure 1: SPI Flash Image Regions

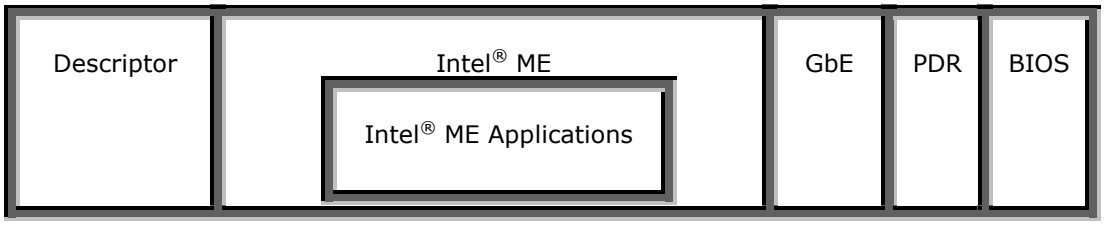


Table 3: Flash Image Regions – Description

Region	Description
Descriptor	This region contains information such as the space allocated for each region of the flash image, read-write permissions for each region, and a space which can be used for vendor-specific data. It takes up a fixed amount of space at the beginning of the flash memory. **Note:** This region MUST be locked before the serial flash device is shipped to end users. Please see 0 below for more information. Failure to lock the Descriptor Region leaves the Intel® ME device vulnerable to security attacks.
Intel® ME	This region contains code and configuration data for Intel® ME applications, such as Intel® AMT technology and Intel® AT. It takes up a variable amount of space at the end of the Descriptor.
GbE	This region contains code and configuration data for an Intel Integrated LAN (Gigabit Ethernet). It takes up a variable amount of space at the end of the Intel® 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. If there is leftover space and Intel® ME region is not implemented, the BIOS region expands to occupy the remaining space.
4. If there is leftover space and the BIOS region is not implemented, then the GbE region expands to occupy the remaining space.
5. If only the Descriptor region is implemented, it expands to occupy the entire flash.

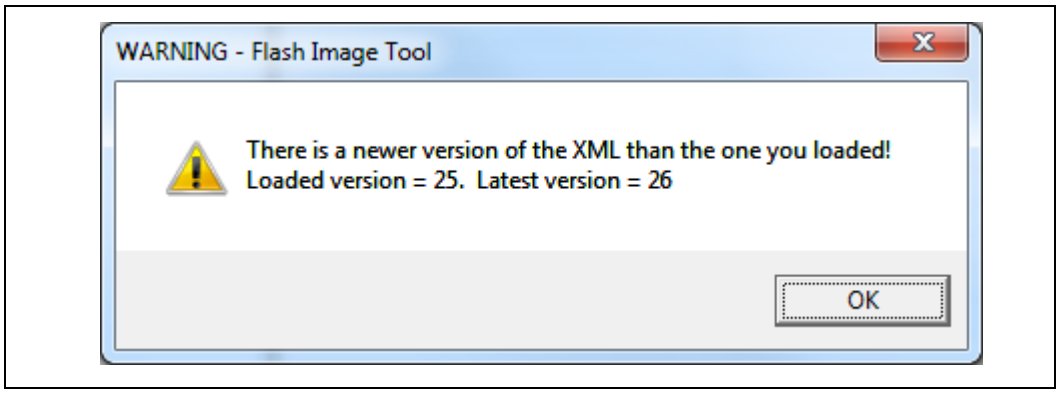
3.3 Required Files

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

- fitctmpl.xml
- newfiletmpl.xml
- vsccommn.bin
- fitcwizardhelp.chm
- fitc.ini

FITC does not run correctly if any of the .xml and .bin files listed above are missing. FITC creates a blank **fitc.ini** file if there is no **fitc.ini** file in the folder.

Note: When using a 'Newfiletmp.xml' from previous kit releases FITc will display a message to the user that the file being used is older than the version FITc expecting (See example below).



After the user selects the OK radio button FITc will automatically update the 'Newfiletmp.xml' with any missing / new or changed variables and pre-populates those variables with the firmware defaults. Once this is completed the user can then re-save this new 'Newfiletmp.xml' back in order to retain the updates made by FITc.

3.4 FITC

See the following for further information:

- General configuration information – See the FW Bring Up Guide from the appropriate Intel® ME FW kit.
- Detailed information on how to configure PCH Soft Straps and VSCC information – See the Intel® 8 Series Chipset Family SPI programming guide

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. FITC 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 a New Configuration

FITC provides a default configuration file that the user can use to build a new image. This default configuration file can be loaded by clicking **File** > **New**.

3.4.3 Opening an Existing Configuration

To open an existing configuration file:

1. Choose File **> Open**; the **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 a 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.

4. Select the path and enter the file name for the configuration.
5. 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 FITC 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 **> Environment Variables**; a dialog appears displaying the current working directory on top, followed by the current values of all the environment variables:

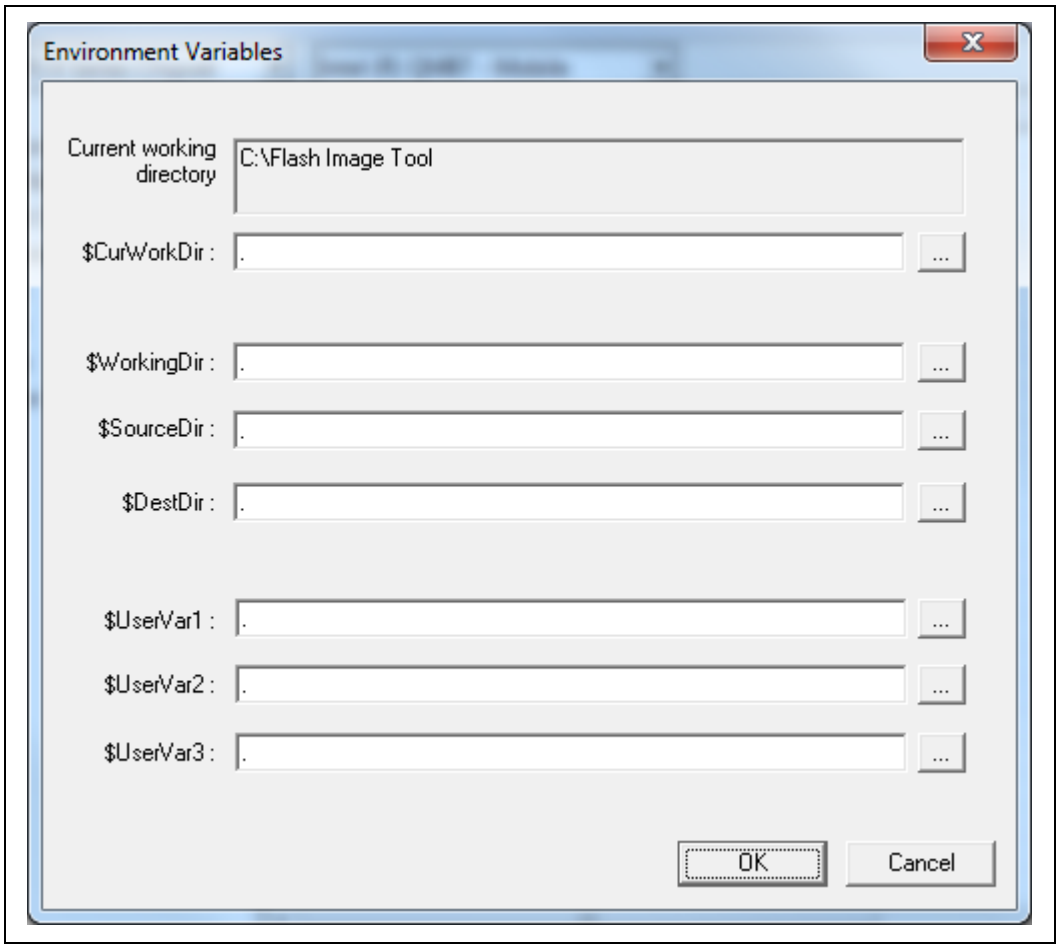
   $WorkingDir – the directory where the log file is kept and where the components of an image are stored when an image is decomposed.

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

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

   $UserVar1-3 – used when the above variables are not populated.

Figure 2. Environment Variables Dialog



2. Click 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 **OK**.

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

3.4.6 Build Settings

FITC lets the user set several options that control how the image is built. The options that can be modified are described in Table 4.

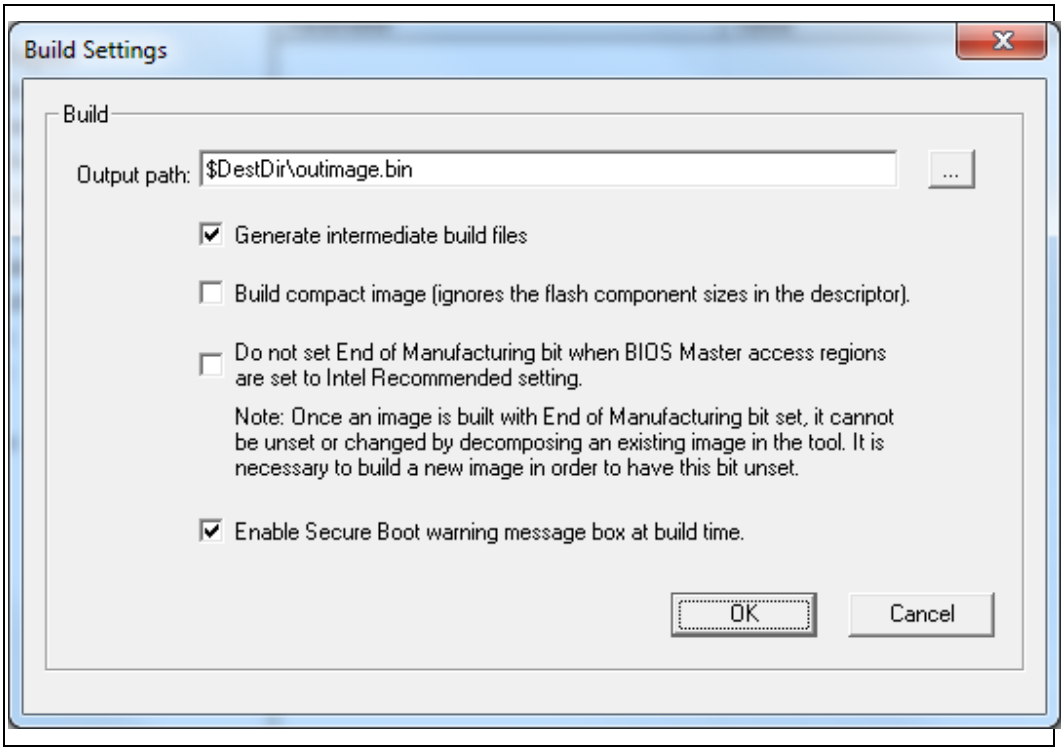
To modify the build setting:

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 4: Build Settings Dialog Options

Option	Description
Output path	The path and filename where the final image should be saved after it is built. (**Note**: Using the $DestDir environment variable makes the configuration more portable.)
Generate intermediate build files	Causes the application to generate separate (intermediate) binary files for each region, in addition to the final image file (see Figure 3). These files are located in the specified output folder's INT subfolder. These image files can be programmed individually with the FPT.
Build Compact Image	Creates the smallest flash image possible. (By default, the application uses the flash component sizes in the Descriptor to determine the image length.)
Do not set End of Manufacturing bit …	When descriptor permissions are set to production values, do not select the **Do not set End of Manufacturing bit** box unless not closing End of Manufacturing is explicitly desired. Intel strongly recommends that the Global Lock Bit/End of Manufacturing bit be set on all production platforms.
Flash Block/Sector Erase Size	All regions in the flash conform to the **4KB sector erase size**. It is critical that this option is set correctly to ensure that the flash regions can be properly updated at runtime.
Asymmetric Flash	Lets the user specify a different sector erase size for the upper and lower flash block. **Only 4KB erase is supported for Intel® ME FW**. This option also lets user modify the flash partition boundary address.

Figure 3. Build Settings Dialog



End of manufacturing bit is simply a byte in the image. This is not an NVAR, or FOV. In previous generation, when creating an image, the user can set the global valid bit automatically based on BIOS being set to production Master Access section, but to allow some customers not to set it, we show this checkbox. This checkbox only does something if:

Intel® ME manufacturing done bit is not set, BIOS is not set to production → FITc will not set Intel® ME manufacturing done bit – independent of this checkbox

Intel® ME manufacturing done bit is not set, BIOS is set to production, checkbox is unchecked → FITc will set Intel® ME manufacturing done bit

Intel® ME manufacturing done bit is not set, BIOS is set to production, checkbox is checked → FITc will not set Intel® ME manufacturing done bit

Intel® ME manufacturing done bit set → will stay set

A dumped image is never reflected in this checkbox – it does not show the actual value of Intel® ME manufacturing done bit. It shows what should be done in the next build. But if Intel® ME manufacturing done bit is set, this checkbox will never uncheck it.

3.4.7 Selecting the Platform SKU

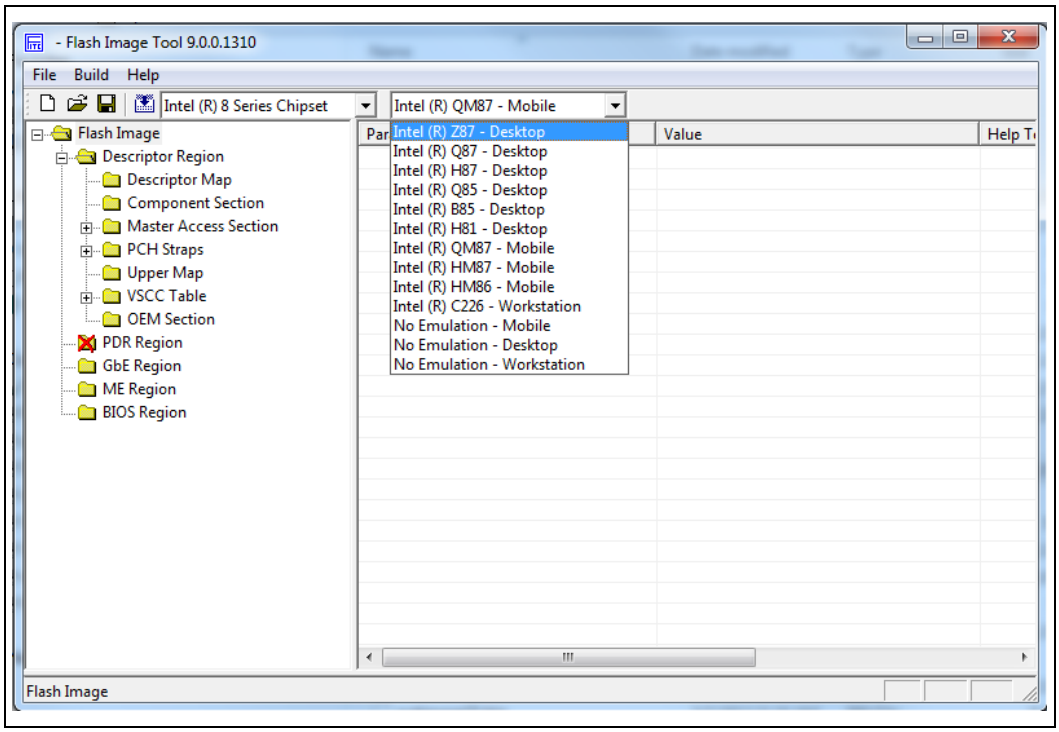
The ability to select the Platform SKU lets the user configure "Full Featured Engineering samples" to test how the firmware behaves like the production Intel® 8 Series Chipset, with the following reservations:

- Certain features only work with particular Chipset SKUs and FW kits (e.g., Intel® AMT only works with corporate SKUs with the 5MB Intel® ME FW kit).
- SKU Manager Selection has no effect on the Production PCH chipset

To select a Platform SKU:

1. Load the Intel® ME region (**Note**: Loading the Intel® ME region first ensures that the proper FW settings are loaded into FITC.
2. Select the appropriate platform type for the specific chipset from the SKU Manager drop-down list; the "Full Featured Engineering Samples" behaves as if it were the selected SKU PCH chipset.

Figure 4: Selected an SKU Platform in FITC



3.4.8 Modifying the Flash Descriptor Region

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

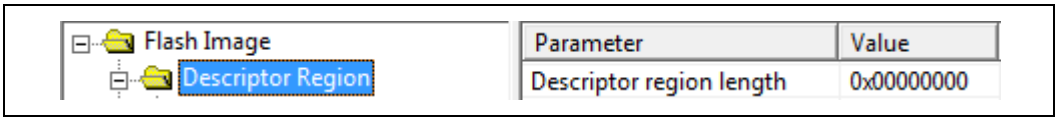
3.4.9 Descriptor Region Length

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

To set the value of the Descriptor Region Length parameter:

1. Select **Descriptor Region** in the left pane; the **Descriptor Region Length** parameter appears in the right pane.
2. Double-click the **Descriptor Region Length** parameter; the **Descriptor Region Length** dialog appears.
3. Enter any non-zero value into the dialog to set the length of the region and click **OK**.

Figure 5. Descriptor Region Length Parameter

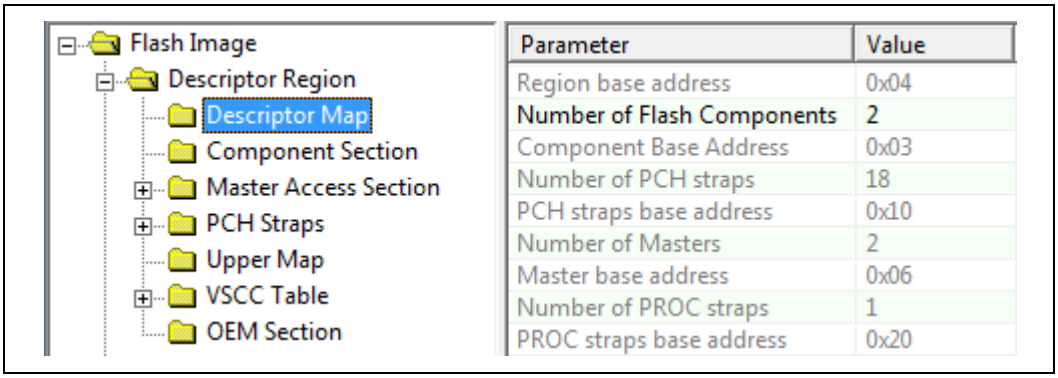


3.4.10 Setting the Number and Size of the Flash Components

To set the number of flash components:

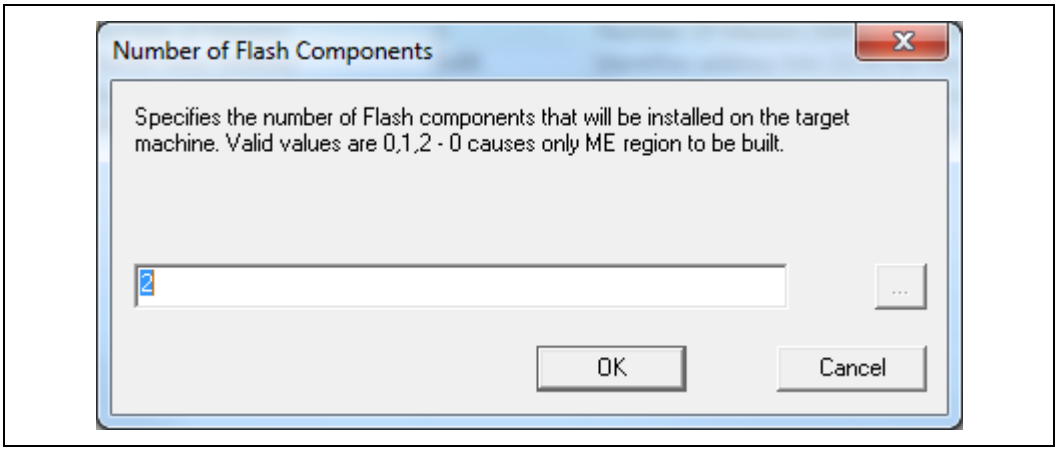
1. Expand the **Descriptor Region** node of the tree in the left pane.
2. Select **Descriptor Map** (see Figure 6); all the parameters in the Descriptor Map section are listed in the right pane.

Figure 6: Descriptor Region > Descriptor Map Parameters



3. Double-click **Number of Flash Components** in the right pane (see Figure 7); the Flash Components dialog appears.
4. Enter the number of flash components (valid values are 0, 1 or 2).
5. Click **OK**; the parameter is updated.

Figure 7: Flash Components Dialog



To set the size of each flash component:

1. Expand **Descriptor Region** node in the left pane and select **Component Section**; the Component Section parameters appear in the right pane. The **Flash component 1 density** and **Flash component 2 density** parameters specify the size of each flash component.
2. Double-click on one of these parameters; a dialog appears.
3. Select the correct component size from the dialog's drop-down list and click **OK**; that parameter is updated.
4. 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.

Note: Setting the number of flash components to 0 will cause FITc to generate just the ME region binary with any associated setting customizations.

Figure 8: Descriptor Region > Component Section Parameters

Parameter	Value	Help Text
Read ID and Read Status clock frequency	33MHz	If more that one Flash component exists, this field must be the lowest c...
Write and erase clock frequency	33MHz	If more that one Flash component exists, this field must be the lowest c...
Fast read clock frequency	33MHz	This field is undefined if the Fast Read Support is set to false.
Fast read support	false	Enables/disables Fast Read support.
Read clock frequency	20MHz	Sets the Flash read frequency
Flash component 1 density	8MB	This field identifies the size of the 2nd Flash component.
Flash component 0 density	8MB	This field identifies the size of the 1st Flash component.
Dual Output Fast Read Support	false	false: Not Supported. true: Dual Output Fast Read instruction is issued in...
Invalid Instruction 3	0	Op-code for an invalid instruction that the Flash Controller should prote...
Invalid Instruction 2	0	Op-code for an invalid instruction that the Flash Controller should prote...
Invalid Instruction 1	0	Op-code for an invalid instruction that the Flash Controller should prote...
Invalid Instruction 0	0	Op-code for an invalid instruction that the Flash Controller should prote...
Invalid Instruction 7	0	Op-code for an invalid instruction that the Flash Controller should prote...
Invalid Instruction 6	0	Op-code for an invalid instruction that the Flash Controller should prote...
Invalid Instruction 5	0	Op-code for an invalid instruction that the Flash Controller should prote...
Invalid Instruction 4	0	Op-code for an invalid instruction that the Flash Controller should prote...

3.4.11 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 5: Region Access Control Table

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

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

Key:

0 – denied access

1 – allowed access

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

Table 6: 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 FITC:

1. Select **Descriptor Region > Master Access, Manageability Engine and GBE > CPU/BIOS** in the left pane; the access parameters are listed in the right pane (see Figure 9).
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 three sections.

   To generate a production image with BIOS access to the PDR region select read access 0x0B and write access 0x0A.

   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 setting for production (e.g., select 0x0D for Intel® ME read access and 0x0C for Intel® ME write access).

Note: If all Read/Write Master access settings for Intel® ME are set to production platform values, then the Intel® ME manufacturing mode done(Global Lock) bit is automatically set. If the Intel® ME manufacturing mode done (Global Lock) bit is set, the FOV mechanism is not available.

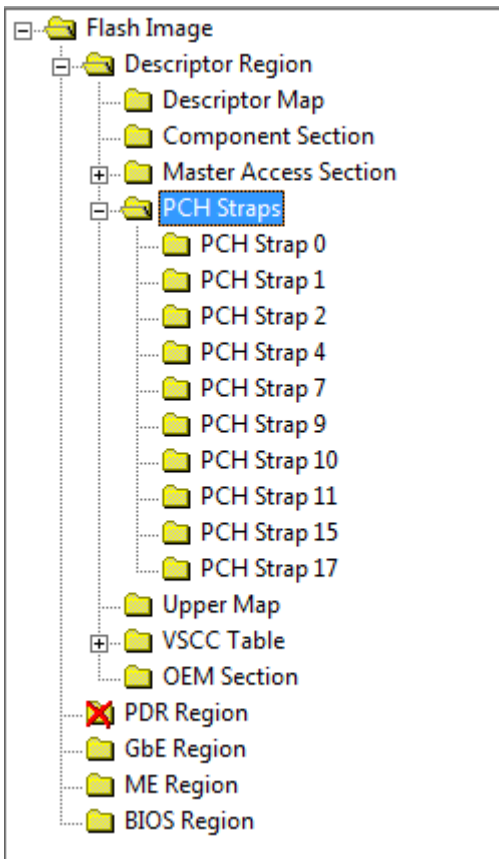
Figure 9: Descriptor Region > Master Access Section

Parameter	Value	Help Text
PCI Bus ID	0	
PCI Device ID	0	
PCI Function ID	0	
Read Access	0xFF	0xFF = Debug/Manufacturing, 0x0D = Production. Each bit corresponds to Regio...
Write Access	0xFF	0xFF = Debug/Manufacturing, 0x0C = Production. Each bit corresponds to Regio...

3.4.12 PCH Soft Straps

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

Figure 10: PCH Straps



3.4.13 VSCC Table

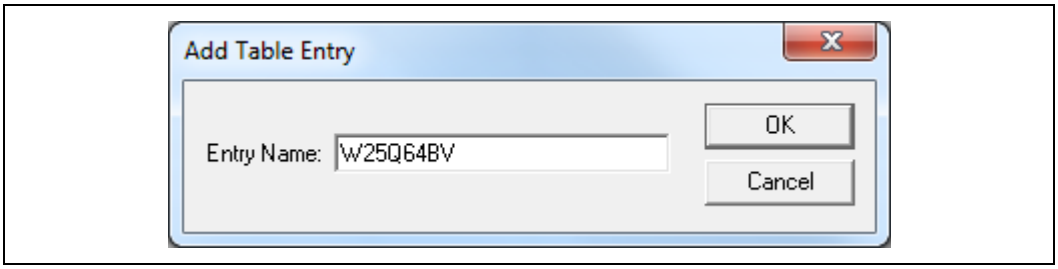
This section is used to store information to setup flash access for Intel® ME. This does not have any effect on the usage of the FPT. **If the information in this section is incorrect, Intel® ME FW may not communicate with the flash device.** The information provided is dependent on the flash device used on the system. (For more information, see the Intel® 8 Series Chipset SPI Programming Guide, Section 6.4.)

3.4.14 Adding a New Table

To add a new table:

1. Right-click on **Descriptor Region > VSCC table**.
2. Choose **Add Table Entry** from the pop-up menu; the **Add Table Entry** dialog appears.

Figure 11: Add VSCC Table Entry Dialog

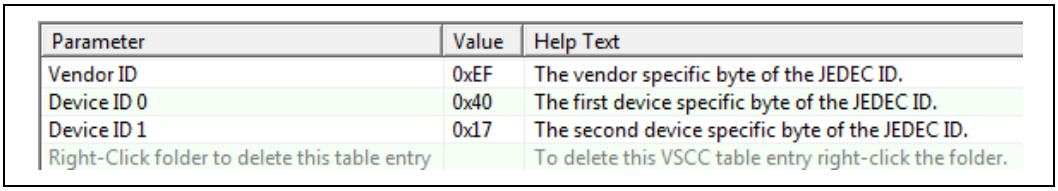


3. Enter a name into the **Entry Name** field. (**Note**: To avoid confusion it is recommended that each table entry name be unique. There is no checking mechanism in FITC to prevent table entries that have the same name and no error message is displayed in such cases.)
4. Click **OK**; the new table is listed in the left pane under **VSCC Table** and user can enter into it the values for the flash device. (See Figure 12, which shows the parameters of a new VSCC table.)

Note: The VSCC register value will be automatically populated by FITc using the vscccommn.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 Intel® 8 Series Chipset Family SPI Programming Guide should be used to calculate the VSSC values.

Figure 12: Sample VSCC Table Entry



Parameter	Value	Help Text
Vendor ID	0xEF	The vendor specific byte of the JEDEC ID.
Device ID 0	0x40	The first device specific byte of the JEDEC ID.
Device ID 1	0x17	The second device specific byte of the JEDEC ID.
Right-Click folder to delete this table entry		To delete this VSCC table entry right-click the folder.

3.4.15 Removing an Existing VSCC Table

To remove an existing table:

1. Right-click on the name of the table in the left pane that the user wants to remove.
2. Choose Remove Table Entry; the table and all of the information will be removed.

3.4.16 Modifying the Intel® ME Region

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

3.4.17 Setting the Intel® ME Region Binary File

To select the Intel® ME region binary file:

1. Select the Intel® ME Region tree node.
2. Double-click on the **Binary file parameter** in the list; a dialog appears that lets the user 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.

Note: If the user specifies in the PCH Strap Section (0) that Intel® ME must boot from flash, the loaded FW must contain a ROM Bypass section. If the FW does not contain a ROM bypass section this field is set as read-only and cannot be changed.

3.4.18 Intel® ME FW Configuration

Intel® ME FW parameters are visible and editable after a valid Intel® ME FW image has been loaded.

If any of the parameters do not have the Intel-recommended value, the offending row is highlighted yellow but no errors are reported. The highlighted yellow is designed to draw attention to these values to ensure these parameters are set correctly.

3.4.19 Intel® ME Section

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

The Intel® ME section lets the user define the computer's manageability features. The parameter values can be found in the Help Text next to the parameter value as shown in Figure 13.

Figure 13: Intel® ME Section

Parameter	Value
FW Update OEM ID	00000000-0000-0000-0000-000000000000
LAN Power Well Config	3
WLAN Power Well Config	0x86
M3 Power Rails Availability	true
Host ME Region Flash Protection Override	true
PROC_MISSING	No onboard glue logic
Processor Emulation	No Emulation
OEM Tag	0x00000000
Hide FW Update Control	false
Debug Si Features	0x00000000
Prod Si Features	0x00000000
M3 Autotest Enabled	false
Enable hash file creation	true
Independent Firmware Recovery Enable	true

3.4.20 Manageability Application Section

Note: This section and its sub-sections are not applicable to 1.5MB Intel® ME FW SKU.

This section describes the Manageability Application parameters. (See the FW Bringup 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).

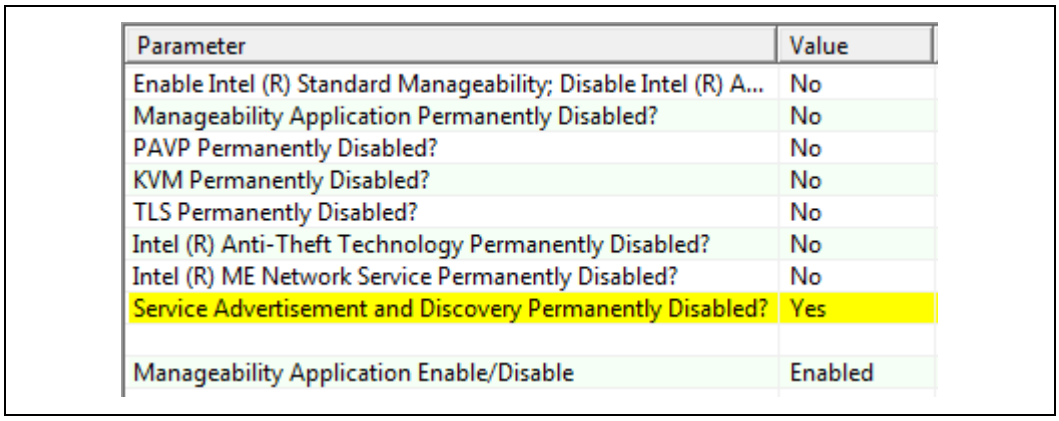
Figure 14: Manageability Application Section

Parameter	Value
Boot into BIOS Setup Capable	false
Pause during BIOS Boot Capable	false
BIOS Reflash Capable	false
BIOS Secure Boot	false
USBr EHCI 1 Enabled	11b Enabled
USBr EHCI 2 Enabled	10b Disabled
Privacy/Security Level	Default

3.4.21 Features Supported

The Features Supported 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. (See the FW Bringup guide for general information and see Appendix E for more details.)

Figure 15: Features Supported Section



Parameter	Value
Enable Intel (R) Standard Manageability; Disable Intel (R) A...	No
Manageability Application Permanently Disabled?	No
PAVP Permanently Disabled?	No
KVM Permanently Disabled?	No
TLS Permanently Disabled?	No
Intel (R) Anti-Theft Technology Permanently Disabled?	No
Intel (R) ME Network Service Permanently Disabled?	No
Service Advertisement and Discovery Permanently Disabled?	Yes
Manageability Application Enable/Disable	Enabled

These options control the availability and visibility of FW features.

In cases where a specific feature is configurable in the Intel® MEBx, permanently disabling it through the **Features Supported** section hides/disables that feature in Intel® MEBx.

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: The Intel® Manageability Application setting combines several manageability technologies that are related to each other. This setting controls the following manageability technologies:

- Intel® Active Management Technology
- Intel® Standard Management
- Fast Call for Help
- Intel® KVM Remote Assistance Application

Setting **Intel® Manageability Application Permanently Disabled?** To "Yes"will permanently disable all of the features listed above the only way to re-enable these features prior to close manufacture on the platform by using Fixed Offset Variables. The only way to re-enable these features is to completely re-burn the Intel® ME region with this setting set to "No". A FW update using ***FWUpdLcl.exe*** cannot re-enable features.

3.4.22 Setup and Configuration Section

The Setup and Configuration section allows the end user to specify the configuration settings, Intel® AT and Intel® DAL. (See the FW Bringup guide for general information and see Appendix E for more details.

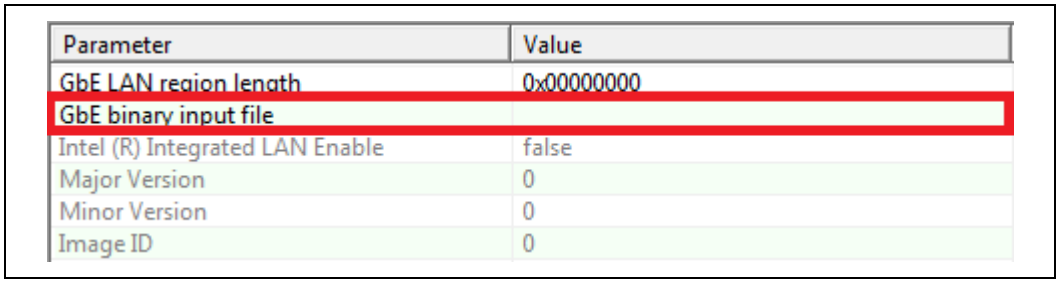
Figure 16: Setup and Configuration Section

Parameter	Value
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
MCTP Static EIDs	0x323130
MCTP Info 3G	0x4032
Permit Period Timer Resolution	Days
PKI DNS Suffix	
OEM Default Certificate Active	false
OEM Default Certificate Friendly Name	
OEM Default Certificate Stream	
OEM Default Certificate 2 Active	false
OEM Default Certificate 2 Friendly Name	
OEM Default Certificate 2 Stream	
OEM Default Certificate 3 Active	false
OEM Default Certificate 3 Friendly Name	
OEM Default Certificate 3 Stream	

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 17: GbE Region Options



3.4.24 Setting the 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 the GbE Region Binary File

To select the GbE Region binary file:

1. Select **GbE Region** in the left pane; the GbE Region parameters are listed in the right pane.
2. Double-click on the **Binary input file** parameter; a dialog appears that lets the user select the GbE file to use.
3. Select a file.
4. Click **OK** to update the parameter; when the flash image is built, the contents of this file is copied into the GbE Region.

3.4.26 Enabling/Disabling the GbE Region

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

To disable the GbE Region:

1. Right-click on **GbE Region** in the left pane.
2. Choose **Disable Region** from the pop-up menu; when the flash image is built it will not contain a GbE Region.

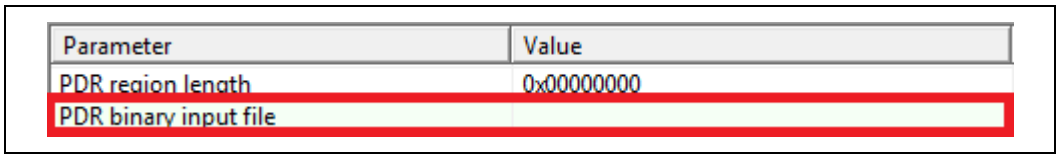
To enable the GbE Region:

1. Right-click on **GbE Region** in the left pane.
2. Choose **Enable Region** from the pop-up menu.

3.4.27 Modifying the PDR Region

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

Figure 18: PDR Region Options



3.4.28 Setting the 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 the PDR Region Binary File

To select the PDR region binary file:

1. Select **PDR Region** in the left pane; the PDR Region parameters are listed in the right pane.
2. Double-click the **Binary input file** parameter; a dialog appears that lets the user specify which PDR file to use.
3. 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 the PDR Region

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

To disable the PDR Region:

1. Right-click on **PDR Region** in the left pane.
2. Choose **Disable Region** from the pop-up menu; when the flash image is built, there is no PDR Region in it.

Note: This region is disabled by default.

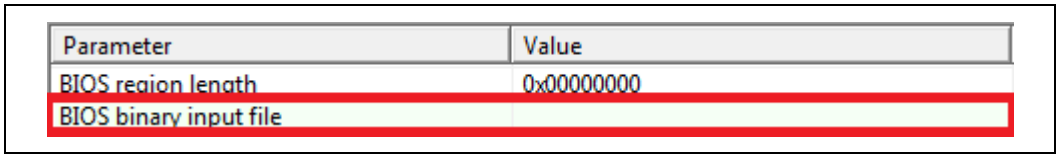
To enable the PDR Region:

1. Right-click on **PDR Region** in the left pane.
2. Choose **Enable Region** from the pop-up menu.

3.4.31 Modifying the BIOS Region

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

Figure 19: BIOS Region Parameters



3.4.32 Setting the BIOS Region Length Parameter

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

3.4.33 Setting the BIOS Region Binary File

To select the BIOS region binary file:

1. Select **BIOS Region** in the left pane; the BIOS Region parameters are listed in the right pane.
2. Double-click the **Binary input file** parameter; a dialog appears that lets the user specify which BIOS file to use.
3. 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 FITC.

To disable the BIOS Region:

1. Right-click on **BIOS Region** in the left pane.
2. Choose **Disable Region** from the pop-up menu; when the flash image is built, there is no BIOS Region in it.

To enable the PDR Region:

1. Right-click on **BIOS Region** in the left pane.
2. Select **Enable Region** from the pop-up menu.

3.4.35 Building a Flash Image

The flash image can be built with the FITC 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.

FITC 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:

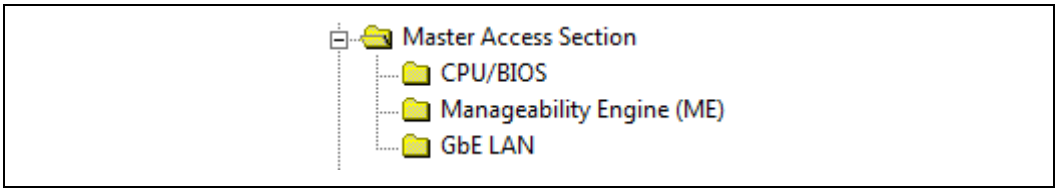
- Binary file representing the image
- Text file detailing the various regions in the image
- Optional set of intermediate files (see Section 3.4.6).
- Multiple binary files containing the image broken up according to the flash component sizes (**Note**: These files are only created if two flash components 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 Change the Region Order on the SPI Device

The order and placement of the regions in the full SPI image created by FITC can be altered. The location of each region is determined by the order of the PDR, GbE, ME and BIOS regions as they are displayed in left pane of the FITC window.

Figure 20: Region Order



Each region is added to the full SPI image in the order in which they appear in the list. The order of the regions in the full SPI image created from the regions listed in Figure 20 in order immediately after the Descriptor Region:

1. BIOS Region
2. GbE Region
3. ME Region

This can be useful when programming a system with two SPI devices. It is possible to change the order of the PDR, GbE, ME and BIOS regions by clicking and dragging the region to the required location. Figure 20 shows that the BIOS is placed on the first

SPI device and the Intel® ME Region is placed on the second SPI device. The length of each region and the order determines if that region is on the first or second SPI device.

3.4.37 Decomposing an Existing Flash Image

FITC 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 (see 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 FITC. FITC 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 FITc customization is applied.

3.4.38 Command Line Interface

FITC supports command line options.

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

The command line syntax for FITC is:

```
FITC [/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>] [XML or BIN file]
```

Table 7: FITC Command Line Options

Option	Description
<XML_file>	Used when generating a flash image file. A sample xml file is provided along with the FITC. 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.

Option	Description
-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.
-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.

Option	Description
-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.
-SKU <value>	This option is used to change the SKU configuration being built. Use the words Q87, QM87, etc. as a reference to a SKU from the drop-down menu (e.g., `/sku Q87`).

3.4.39 Example – Decomposing an 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:

```
Fitc.exe 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.

3.4.40 More Examples of FITC 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:

```
Fitc.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:

```
Fitc.exe /b /me ".\..\..\Image
Components\Firmware\ME9.0_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:

```
Fitc.exe /b /gbe ".\..\..\Image
Components\GbE\NAHUM6_CLARKSVILLE_DESKTOP_11.bin" <file.bin or file.xml>
```

§

4 Flash Programming Tool

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 fixed offset variables.

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 MS DOS 6.22, DRMKDOS, and 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* 7 64/32 bit and Windows* 8 64/32 bit.

The Windows 64 bit version (fpt64.exe) is designed for running in native 64 bit OS environment which does not have 32 bit compatible mode available for example Win*PE 64.

FPT requires that the platform is bootable (i.e. working BIOS) and an operating system to run on. It is designed to deliver a custom image to a computer that is already able to boot and is not a means to get a blank system up and running. 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

A flash image is composed of up to five 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 21: Flash Image Regions

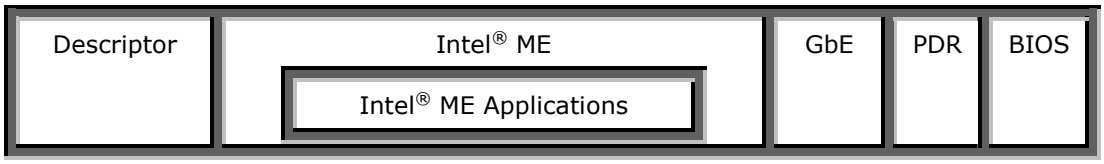


Table 8: Flash Image Regions – Description

Component	Description
Descriptor	Region that takes up a fixed amount of space at the beginning of the flash memory. Contains information such as: Space allocated for each region of the flash image. Read/write permissions for each region. A space that can be used for vendor-specific data.
Intel® ME	Contains code and configuration data for Intel® ME applications, such as Intel® AMT technology and Intel® AT.
GbE	Contains code and configuration data for GbE.
BIOS	Contains code and configuration data for the entire platform.
PDR	Region that allows system manufacturers to define custom features for the platform.

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 9: 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 in the same 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.

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 the Flash Device

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

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 HMRFPO ENABLE Intel® MEI command to Intel® ME (for more information see the 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 Fixed Offset Variables

FPT can program the fixed offset variables 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. The fixed offset variables can be continuously changed until the Intel® ME manufacturing mode done (***globallocked)*** bit is set to 0x01. The parameters can **NOT** be modified after this bit is set. To modify the default settings for the parameters, the entire flash device must be re-programmed.

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

Table 10: Fixed Offset Variables Options

Option	Description
fpt.exe –FOVs	Displays a list of the supported variables.
fpt.exe –cfggen	Creates an empty text file that lets the user update multiple fixed offset variables. The variables have the following format in the text file: <Parameter name> = <Value> In the created text file:
fpt.exe –U –IN <Text file>	Updates the fixed offset variables with the values as they are entered in the text file.

See Appendix A for a description of all the Fixed Offset Variable parameters.

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 -? option.

The commands in 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] [-D] [-DESC] [-BIOS] [-ME] [-GBE] [-PDR]
[-SAVEMAC] [-C] [-B] [-E] [-ADDRESS|A] [-LENGTH|L] [-FOVS] [-CFGGEN]
[-U] [-O] [-IN] [-N] [-ID] [-V] [-LOCK] [-DUMPLOCK] [-PSKFILE]
[-CLOSEMNF] [-GRESET] [-PAGE] [-SPIBAR] [-R] [-VARS] [-COMMIT]
[-COMPARE] [-HASHED] [-DisableMe]
```

Table 11: 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.
-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.
-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, erases the flash, and then programs the binary into the flash. After a successful flash, FPT verifies that the SPI flash matches the provided image. Without specify the length with –L option, FPT will use the total SPI size instead of an image size. The NOVERFY sub-option *must* 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.

Option	Description
-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.
-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.
-C:	Chip erase. Erases the contents of SPI flash device(s). This function does NOT erase block by block.
-B:	Blank Check. Checks whether the SPI flash is erased. If the SPI flash is not empty, the application halts as soon as contents are detected. The tool reports the address at which data was found.
-E:	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.
-FOVS:	Supported Fixed Offset Variables. Displays all supported FOVs supported by FPT. This option displays names and IDs of supported FOVs.
-U:	Update. Updates the FOVs in the flash. 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 `–cfggen` command). The `–in <file>` option is used to specify the input file.
-O <file>	Output File. The file used by FPT to output FOV information.
-IN <file>	Input File. The file used by FPT for FOV input. This option flag must be followed by a text file (i.e., `fpt –u –in FPT.cfg`). The tool updates the FOVs contained in the text file with the values provided in the input file. User can also use FPT –cfggen to generate this file.
-N <value>	Name. Specifies the name of the FOV that the user wants to update in the image file or flash. The name flag must be used with Value (`–v`).

Option	Description
-ID <value>	ID. The names of certain FOVs are quite lengthy. This option lets the user update the FOV by providing its unique identification number instead of its name. The ID for each FOV is specified in the configuration file.
-V <value>	Value. Specifies the value for the FOV variable. The name of variable is specified in the Name flag. The Value flag must follow the Name flag.
-LOCK:	Region Lock. Sets the SPI flash region access to the Intel recommended values (see ** Return **value 0 indicates successful completion. In the second case, FPT –closemnf returns 1 (= error) because it is unable to set the ME Mfg Done bit, because flash permissions are already set to Intel recommended values (host cannot access ME Region).** Table 13)
-DUMPLOCK:	Dump Lock Settings. Displays the current lock settings on the screen. The lock settings are read from the descriptor region.
-PSKFILE <file>	PID/PPS/Password pair file. Specifies the input file that contains the one or more PID/PPS/Password key value pairs. This option is used to update the PID, PPS, and Password FOVs whose values are read from the input file. This option only support version 1 FiletypeHeader UUID
-CLOSEMNF <NO> <PDR>:	End of Manufacturing. This option is executed at the end of manufacturing phase. This option does the following: Sets the Intel® ME manufacturing mode done bit (Global Locked bit). Verifies that the Intel® ME manufacturing mode done bit (Global Locked) is set. Sets the master region access permission in the Descriptor region to its Intel-recommended value Verifies that flash regions are locked. 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. 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. The "PDR" addition will allow CPU\BIOS Read & Write access to the PDR region of flash. **Note**: Running `FPT-closemnf` also sets the default value for any unprovisioning process. Run `FPT –closemnf` 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. **Note:** For DeepSx enabled mobile platforms the following command sequence should be used to avoid platform hangs: **> FPT –closemnf no** **> FPT –greset no**

Option	Description
-GRESET <NO> :	Global Reset. FPT performs a global reset. On mobile platforms this includes driving GPIO30 low. Mobile platforms require a SUS Well power-down acknowledge-driven low before the global reset occurs or the platform may not boot up from the reset. The "NO" afterwards disables the driving of GPIO30 for mobile SKUs. **Note:** For DeepSx enabled mobile platforms the "NO" option should be used with the –GRESET command to avoid platform hangs.
-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.
-CFGGEN	FOV Input file generation option. This creates a file which can be used to update the FOVs. If no file name is specified the default name "FPT.CFG" will be used.
-SPIBAR:	Display SPI BAR. FPT uses this option to display the SPI BAR.
-R <name>	NVAR Read. FPT uses this option to read a variable stored as a NVAR in the FW. The value of the variable is displayed. By default, all non- secure variables are displayed in clear-text and secure NVAR will be displayed in HASH. The `–hashed` option can be used to display the hash of a value instead of the clear-text value.
-VARS:	Display Supported Variables. FPT uses this option to display all variables supported for the `–R` and `–COMPARE` commands.
-COMMIT:	Commit. FPT uses this option to commit FOVs 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.
-COMPARE <file>	NVAR Compare. FPT uses this option to compare a NVAR with the expected value filled in a text file. The compare entry should have the following format: `"<name>" = <value>` **Note**: <value> should have the form "xx ", where xx is a hexadecimal value. Each byte must be separated by a space and start with the least significant followed by the next significant byte.
-PAGE	Pauses the screen when a page of text has been reached. Hit any key to continue.
-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	This option will allow the tool operator to temporarily disable the Intel® Management Engine until the next Global Reset or G3.

Table 12: FPT –closemnf Behavior

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

Table 13: 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 FOV

Note: This section is not applicable for 1.5MB Intel® ME FW SKU.

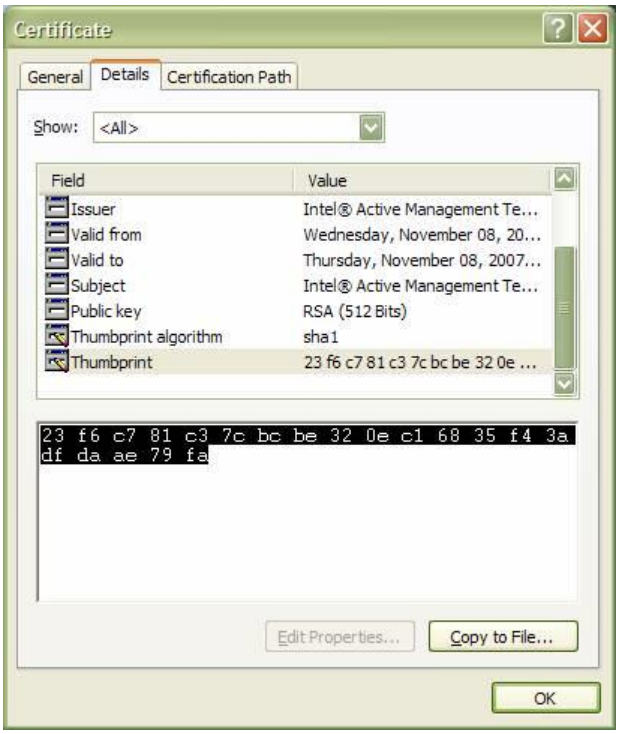
There are 3 OEM Customizable certificate hash values that can be stored in the Intel® ME region:

- 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 FOV (with FPT or other flash programming methods) or FITC.

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

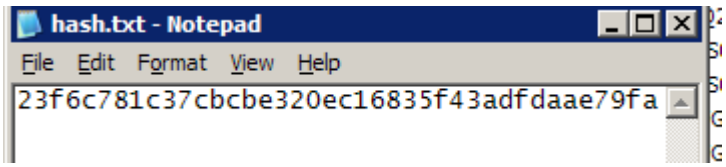
1. Copy the raw hash values from a valid certificate file.

Figure 22: 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 23: 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 FOV:

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   = 23f6c781c37cbce320ec16835f43adfdaae79fa
```

6. Flash Hash FOV with FPT's -u –in option (e.g., fpt –u –in sampleparam.txt).

Note: **FTP.CFG** is the file that is used to update multiple FOVs (fpt.exe /ex /o FPT.CFG).

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) 2007-2011, 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 Intel® 8 Series Chipset 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
- Chip Erase Command.

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

```
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 a 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) 2007-2011, 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 a 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) 2007-2011, 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) 2007-2011, 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:  x.x.x.xxxx
Copyright (c) 2007-2011, 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)
   --- Flash Image Information --
   Signature: VALID
   Signature: VALID
       Number of Flash Components: 1
           Component 1 - 8192KB (65536Kb)
       Regions:
           Descriptor - Base: 0x000000, Limit: 0x000FFF
           BIOS       - Base: 0x580000, Limit: 0x7FFFFF
           ME         - Base: 0x003000, Limit: 0x57FFFF
           GbE        - Base: 0x001000, Limit: 0x002FFF
           PDR        - Not present
       Master Region Access:
           CPU/BIOS - ID: 0x0000, Read: 0xFF, Write: 0xFF
           ME       - ID: 0x0000, Read: 0xFF, Write: 0xFF
           GbE      - ID: 0x0118, Read: 0xFF, Write: 0xFF
Total Accessible SPI Memory: 8192KB, Total Installed SPI Memory: 16384KB
FPT Operation Passed
```

This command displays information about the flash devices present in the computer. The base address refers to the start location of that region and the limit address refers to the end of the region. If the flash device is not specified in **fparts.txt**, 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) 2007-2011, 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) 2007-2011, 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 “Power Package 1”
-----------------------------------------
Intel (R) Flash Programming Tool. Version:  x.x.x.xxxx
Copyright (c) 2007-2011, 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: "Power Package 1"
Value: True / 01
Retrieve Operation: Successful
```

Please note that 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 Compare Intel® ME settings

FPT –verbose –compare vars.txt compares variables with suggested values in vars.txt, and report result on the screen. Vars.txt can have the following data with verbose information: FPT –VARS can be used to get the VAR list for the platform and get the value/format from FITC advanced mode. There are settings in the ME which are stored encrypted. Users will not be able to compare them using clear text values. Please use FPT –R option to read the hash value of those settings and use them as baseline for the expected value.

```
"MEBxPassword" = 76 3C BE 3E B5 75 5F 6D 2D 5D 94 43 FD 79 A1 9D 54 D2
D5 9C 87 F8 FF 0E 6C 59 6F D2 17 37 13 5B
"OEMSkuRule" = EF DC EE 0F
"FeatureShipState" = EF FF EE 03
"OEM_TAG" = 78 56 34 12
"PID" = 8F DE B9 92 C3 88 03 71 12 A9 A7 3D FC 18 80 78 64 58 0A E1 D9
E4 19 54 EF 6A 9F 33 F9 74 93 8C
"PPS" = 1A D3 16 1B A1 84 9A 7E 65 9E FB 67 1D 39 8E C0 06 92 81 67 4D
76 FB E4 09 1F 73 27 85 20 84 88
"USBrSettings" = 0B
"LAN Well Power Config" = SLP_LAN#(MGPIO3)
"WLAN Well Power Config" = Disabled
"Debug Si Features" = 00 00 00 00
"Prod Si Features" = 00 00 00 00
"M3 Power Rails Availability" = True
"HECI ME Region Unlockable" = True
"Sub System Vendor ID" = 00 00
"FW Update OEM ID" = 12345678-AABB-CCDD-EEFF-55AA11223344
"PROC_MISSING" = No onboard glue logic
"Power Package 1" = True
"Power Package 2" = True
"Default Power Package" = Power Package 2
"Enable Intel(R) Standard Manageability; Disable Intel(R) AMT" = No
"Manageability Application Permanently Disabled?" = No
"PAVP Permanently Disabled?" = No
"KVM Permanently Disabled?" = No
"TLS Permanently Disabled?" = No
"Intel(R) Anti-Theft Technology Permanently Disabled?" = No
"Manageability Application Enable/Disable" = Enabled
"BIOS Reflash Capable" = False
"Boot into BIOS Setup Capable" = False
"Pause during BIOS Boot Capable" = False
"USBr EHCI 1 Enabled" = 11b Enabled
"USBr EHCI 2 Enabled" = 10b Disabled
"PrivacyLevel" = Default
"Host Based Setup and Configuration" = True
"Allow Unsigned Assert Stolen" = False
"Intel(R) Anti-Theft BIOS Recovery Timer" = Disabled
"MEBx Password Policy" = 00
"Hash 0 Active" = True
```

```
"Hash 0 Friendly Name" = VeriSign Class 3 Primary CA-G1
"Hash 0 Stream" = 74 2C 31 92 E6 07 E4 24 EB 45 49 54 2B E1 BB C5 3E 61
74 E2
"ODM ID used by Intel(R) Service" = <hashed value>
```

4.11.11 FOV Configuration File Generation (-cfggen)

It creates an input file which can be used to update multiple (any or all) FOV's. The file includes all the current FOV's. When creating the file, it extracts the fixed offset variables from flash. Note, the file generated will change every time the list of FOV's changes.

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

        < none >                Creates an input file which can be
                                modified to update multiple FOVs.  If
                                no output file name is provided, the
                                default "FPT.cfg" file will be created.
        -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.
;
MEBxPassword =

DefPwrPackage =

;   OEMSkuRule:  Entering a value for the complete 32-bit FOV entry
```

```
;   below and bit-wise entries are mutually exclusive.  Entering a value
for
;   the complete FOV will cause the program to ignore any bit-wise
entries.
;
;   Valid entries for the bit-wise values are "enable", "disable",
;   "NoChange", or no value at all (i.e. blank). The values are not case
;   sensitive. Invalid bit-wise values will cause FPT to display a
warning
;   and ignore the bit-wise entry being updated.
;
OEMSkuRule =
    Enable Intel (R) Standard Manageability; Disable Intel (R) AMT =
    Manageability Application =
    Intel (R) Anti-Theft Technology =
    PAVP                    =
    Intel (R) ME Network Service =
    KVM                     =
    TLS                     =
    Service Advertisment & Discovery =
    Near Field Communication Enabled =

;   FeatureShipState:  Entering a value for the complete 32-bit FOV entry
;   below and bit-wise entries are mutually exclusive.  Entering a value
for
;   the complete FOV will cause the program to ignore any bit-wise
entries.
;
;   Valid entries for the bit-wise values are "enable", "disable",
;   "NoChange", or no value at all (i.e. blank). The values are not case
;   sensitive. Invalid bit-wise values will cause FPT to display a
warning
;   and ignore the bit-wise entry being updated.
;
FeatureShipState =
    Manageability Application =


SetWLANPowerWell =

OEM_TAG =

PID =

PPS =

MEIdleTimeout =

;  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      =
OEMCustomCert1 FriendlyName  =
OEMCustomCert1 RawHashFile   =

;  OEMCustomCert2 Certificate
```

```
;  All data is required to update the certificate.
;  See the Tools Users Guide for detailed explanation
;  of required data and format.
OEMCustomCert2 IsActive      =
OEMCustomCert2 FriendlyName  =
OEMCustomCert2 RawHashFile   =

;  OEMCustomCert3 Certificate
;  All data is required to update the certificate.
;  See the Tools Users Guide for detailed explanation
;  of required data and format.
OEMCustomCert3 IsActive      =
OEMCustomCert3 FriendlyName  =
OEMCustomCert3 RawHashFile   =

USBrSettings =

Privacy/SecurityLevel =

ODM_ID =

SystemIntegratorId =

ReservedId =

ATFPOPHard =

ATFPOPSoft =
```

§

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® MEManuf (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* 7 64/32 bit and Windows* 8 64/32 bit.

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.

VSCCCOMN.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:

```
MEMANUF [-EXP] [-H|?] [-VER] [-S0] [-F] [-TEST] [-NETON][-NETOFF]
        [-NOWLAN] [-WLAN] [-EOL] [-NEXTREBOOT] [-CFGGEN] [-VERBOSE]
        [-PAGE] [-NOGFX][-GFX] [-NOLAN][-LAN][-NONFC][-NFC]
```

Table 14: Options for the Tool

Option	Description
No option	There are differences depending on the firmware SKU type the system is running on: If BIST is disabled in the Intel® ME Boot: The first time running Intel® MEManuf, since there is no M3 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 M0 to M3, and then attempt automatically transition back from M3 to M0 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. If BIST is enabled in the Intel® ME Boot: If there is no M3 test result, the tool will report error and request user to use –test to run a full BIST. If there is M3 test result, the tool will execute the runtime BIST and report the result. If running on a 1.5MB SKU image, the tool will request the FW to run a complete BIST which doesn't involve any power transition at the end of the test. Test result will be reported back right after the test is done and cleared. If BIST test result isn't 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. 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.
-EXP	Shows examples of how to use the tools.
-H or -?	Displays the help screen.
-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
-NETON	**Note**: This option is not applicable for 1.5MB 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.

Option	Description
-NETOFF	**Note**: This option is not applicable for 1.5MB Intel® ME FW SKU. This option re-enables the integrated GbE wired/wireless LAN interface so that network traffic can go in/out of it. If Intel® AMT is disabled, "Error 9257: Cannot run the command since Intel® AMT is not available" is returned.
-NOWLAN	**Note**: This option is not applicable for 1.5MB 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:** For Intel® vPro platform this test will only be skipped if the FW image is built with the WLAN power well set to 0x84 or 0x85 and there is a WLAN adapter present in the platform). **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**
-EOL <Var|Config> -F <filename>	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 `config` or `var` is optional. Using `–EOL` without a sub option is equivalent to the `–EOL config`. VSCC test and ICC data check are performed for all options. 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. Host based tests 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. 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. 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). Intel® MEMANUF –EOL Check.) When –f flag is used along with a file name, the tool will load the file as the configuration file, instead of using MEManuf.cfg.

Option	Description
-NEXTREBOOT	Upon successful platform reboot M3 Autotest will be performed. **Note:** This is a standalone command and will only work if M3 Autotest has been enabled in the firmware image. M3 Autotest will be executed on the next Moff – M0 transition (example: Cold Reset), Global Reset or G3. The option itself will not trigger any platform reboots.
-CFGGEN <filename>	Use this option along with a filename to generate a default configuration file. This file (with or without modification) can be used for the –EOL option. Rename it **MEManuf.cfg** before using it. It is highly recommended to use this option to generate a new **MEManuf.cfg** with an up-to-date variable names list before using the Intel® MEManuf End-Of-Line check feature.
-VERBOSE <file>	Displays the debug information of the tool or stores it in a log file.
–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 1.5MB Intel® ME FW SKU. This option only applies to the 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
-NONFC	This option will skip NFC test
-NFC	This option will force NFC test. NFC BIST consists of two tests: 1. HW connectivity between ME and the NFC module 2. RF test of the module

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 15: Intel® MEMANUF Test Matrix

		M3 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 M3 test result in SPI - After: Run Runtime BIST and query M3 test result from SPI without reset	Run runtime BIST test (with no reset)
	-Test	-Run full BIST test with ME triggered reset in DOS and host triggered hibernation in Windows - Save the M3 test result in SPI	Run runtime BIST test (with no reset)
	-S0	Run runtime BIST test (with no reset)	Same as M3 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 M3 test result retrieved, return error	Run runtime BIST test (with no reset)
	-Test	-Run full BIST test with ME triggered reset in DOS and host triggered hibernation in Windows - Save the M3 test result in SPI	Run runtime BIST test (with no reset)
	-S0	Run runtime BIST test (with no reset)	Same as M3 Supported SKU

Note: VSCC test and ICC data check are performed for all options.

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 introduced in the Intel® 7 Series Chipset Family platform 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. Please refer that section.

5.4.1 MEMANUF.cfg File

The **MEMANUF.cfg** 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.cfg** 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 **MEMANUF.cfg** file:

```
// The end-of-line checks are broken into two categories. One is
// Variable Check, and the other is Configuration Check. If either
// of these check fails, by default MEManuf will report error and
// continue on to the next check. If a user doesn't wish to continue
// when an error is found, ErrAction field can be used. Please see
// the examples here for detailed explaination:
//
//    SubTestName="ME VSCC check", ErrAction="ErrorStop"
//
// If the above test fails, MEManuf will report error and stop. There
// are total of three different error actions user can choose from:
//
// ErrorContinue - report error and continue on to the next check
// ErrorStop - report error and stop any check after the current one
// WarnContinue - report warning and continue on to the next check
```

```
//
// To add comment or take out a specific test, leave // at the start
// of a line. This file is processed by MEManuf line by line as text
// file. Duplication of the same sub-tests are allowed, but MEManuf
// will always perform the last test to the first test from the file.

// All string comparisions given in this file are case insensitive
// compare. There might be multiple field name/value pairs in one
// entry, but each field needs to be specified in the following
// format where <field name> can be replaced by SubTestName, ReqVal
// or ErrAction, <field value> can be replaced by any string including
// dash and/or spaces surrounded by double quotation marks, or hex-
// decimal number(s) that not surrounded by double quotation marks.
// In case of numeric value, each value (without 0x prefix) needs to
// be specified in byte and deliminated by spaces if there are multiple
// bytes. No line Wrapping is supported:
//
//    <field name>="<field value>", such as ReqVal=" ", or
//    <field name>=<numeric value>, such as ReqVal=78, or
//    <field name>=<numeric value>, such as ReqVal=01 0A 0F FE 7B CD

////////////////////////////////////////////////////////////////////////
   /////
// Intel recommands default end-of-line checks includes the following
// list. If a user chooses to use his/her own version of MEManuf.cfg
// to skip or modify the error action of these checks as WarnContinue,
// MEManuf will report failure with warnings when these checks are
   skipped,
// or have errors. It's suggested that a user should perform these
   Intel(R)
// recommanded check on all type of SKUs.

SubTestName="EOP status check"
SubTestName="ME VSCC check"
SubTestName="BIOS VSCC check"
SubTestName="ME Manufacturing Mode status"
SubTestName="Flash Region Access Permissions"
SubTestName="Security Descriptor Override (SDO) check"
SubTestName="CF9GR lock check"
SubTestName="MAC address"
SubTestName="Wireless MAC address"
SubTestName="System UUID"

////////////////////////////////////////////////////////////////////////
   /////
// Please note that MAC address check will be skipped if Intel Gbe
   region
// is not present in SPI image. Wireless MAC address check will be
   skipped
// if Intel wireless device is not found on the PCI bus. System UUID
   check
// will be skipped if platform is not vPro platform.
//
```

```
// MAC address check, Wireless MAC address check and UUID check
// will be skipped if Intel(R) AMT is permanently disabled or not
   present.
//
// MAC address and System UUID Checks can work with an optional ReqVal
   field,
// which allows a user to specify his/her custom values to compare
   against.
//
// For example, the test shown here checks the current wired LAN MAC
   address
// against user provided value of 01-02-03-04-05-06:
//
//    SubTestName="MAC address", ReqVal="01-02-03-04-05-06"
//
// Here is the default values MEManuf uses if ReqVal field if obmitted:
//
// System UUID - all zeros and 0xff are considered as errors
// MAC address - all zeros and 0xff are considered as errors
// Wireless MAC address - all zeros and 0xff are considered as errors
//
// MAC address takes the format as XX-XX-XX-XX-XX-XX
// System UUID takes the format as XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX
////////////////////////////////////////////////////////////////////////
   /////

////////////////////////////////////////////////////////////////////////
   ////////
// The following Configuration Check requires a user to enter an
   expected
// value after ReqVal=, otherwise the lines without ReqVal field values
   will
// be ignored.
//
// Please note that GBE version check will be skipped if Intel Gbe
   region
// is not present in SPI image.
//
// ME FW version is a string as <major ver>.<minor ver>.<hotfix
   ver>.<build num>
// GBE version is a string as <major ver>.<minor ver>.<revision ver>
// BIOS version is string that vendor specific
////////////////////////////////////////////////////////////////////////
   ////////

// SubTestName="ME FW version", ReqVal=
// SubTestName="BIOS version", ReqVal=
// SubTestName="GBE version", ReqVal=
// SubTestName="Wireless LAN micro-code mismatch", ReqVal=

////////////////////////////////////////////////////////////////////////
// Variable Check - user needs to put an expected value after ReqVal,
// otherwise the lines without ReqVal field values will be ignored
//
// There are variables that stored in encrypted format. When comparing
```

```
// with these variables, ReqVal can only specified as numeric values
// (in encrypted form) in byte order as mentioned aboved. ReqVal needs
// to be surrounded by double quotation marks if they are string input.
//
// To get a up-to-dated MEManuf.cfg with a complete variable names list,
// please run MEManuf -cfggen <filename>. Please note that variables
// that have # need to be replace by a number. Here defines the number:
//
// Note: The '#' for hash variables should be replaced with an entry
   index.
//       The valid range is 0 to 22.
//
// !!! Please be sure to disable sending EOP or leave platform in ME
// !!! manufacturing mode to run this test, otherwise MEManuf will
// !!! report failure because this feature is only available in factory
// !!! mode environment.
//////////////////////////////////////////////////////////////////////

// SubTestName="Allow Unsigned Assert Stolen", ReqVal=
// SubTestName="BIOS Reflash Capable", ReqVal=
// SubTestName="Boot into BIOS Setup Capable", ReqVal=
// SubTestName="Debug Si Features", ReqVal=
// SubTestName="Default Power Package", ReqVal=
// SubTestName="Enable Intel (R) Standard Manageability; Disable Intel
   (R) AMT", ReqVal=
// SubTestName="FeatureShipState", ReqVal=
// SubTestName="Flash Protection Override Policy Hard", ReqVal=
// SubTestName="Flash Protection Override Policy Soft", ReqVal=
// SubTestName="FW Update OEM ID", ReqVal=
// SubTestName="HECI ME Region Unlockable", ReqVal=
// SubTestName="Idle Timeout - Manageability Engine", ReqVal=
// SubTestName="Intel (R) Anti-Theft BIOS Recovery Timer", ReqVal=
// SubTestName="Intel (R) Anti-Theft Technology Permanently Disabled?",
   ReqVal=
// SubTestName="Intel (R) Dynamic Application Loader Permanently
   Disabled?", ReqVal=
// SubTestName="Intel (R) ME Network Service Permanently Disabled?",
   ReqVal=
// SubTestName="KVM Permanently Disabled?", ReqVal=
// SubTestName="LAN Well Power Config", ReqVal=
// SubTestName="M3 Autotest Enabled", ReqVal=
// SubTestName="M3 Power Rails Availability", ReqVal=
// SubTestName="Manageability Application Enable/Disable", ReqVal=
// SubTestName="Manageability Application Permanently Disabled?",
   ReqVal=
// SubTestName="MCTP Info 3G", ReqVal=
// SubTestName="MCTP Static EIDs", ReqVal=
// SubTestName="MEBxPassword", ReqVal=
// SubTestName="ODM ID used by Intel (R) Services", ReqVal=
// SubTestName="OEM Customizable Certificate 1", ReqVal=
// SubTestName="OEM Customizable Certificate 2", ReqVal=
// SubTestName="OEM Customizable Certificate 3", ReqVal=
// SubTestName="OEM Default Certificate", ReqVal=
```

```
// SubTestName="OEM_TAG", ReqVal=
// SubTestName="OEMSkuRule", ReqVal=
// SubTestName="Pause during BIOS Boot Capable", ReqVal=
// SubTestName="PAVP Permanently Disabled?", ReqVal=
// SubTestName="Permit Period Timer Resolution", ReqVal=
// SubTestName="PID", ReqVal=
// SubTestName="PKI DNS Suffix", ReqVal=
// SubTestName="Power Package 1", ReqVal=
// SubTestName="Power Pkg 2 Supported", ReqVal=
// SubTestName="PPS", ReqVal=
// SubTestName="Privacy/Security Level", ReqVal=
// SubTestName="PROC_MISSING", ReqVal=
// SubTestName="Prod Si Features", ReqVal=
// SubTestName="Reserved ID used by Intel (R) Services", ReqVal=
// SubTestName="Sub System Vendor ID", ReqVal=
// SubTestName="System Integrator ID used by Intel (R) Services",
   ReqVal=
// SubTestName="TLS Permanently Disabled?", ReqVal=
// SubTestName="USBr EHCI 1 Enabled", ReqVal=
// SubTestName="USBr EHCI 2 Enabled", ReqVal=
// SubTestName="USBrSettings", ReqVal=
// SubTestName="WLAN Well Power Config", ReqVal=
```

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: Create a line that begins with SubTestName="<specific sub test name>".

Here are some other examples that explain how to use this feature:

- To run a GbE version check defined under "Platform Configuration Checkings", a valid GbE version should be equal to string 1.2.3:

  SubTestName="GBE version", Reqval="1.2.3"

- To run the Variable check defined for "Remote Connectivity Service Enabler ID", a valid ID should be equal to string 550e8400-e29b-41d4-a716-446655440000:

  ```
  SubTestName="Remote Connectivity Service Enabler ID", Reqval="
  550e8400-e29b-41d4-a716-446655440000"
  ```

5.4.2 MEMANUF –EOL Variable Check

`MEMANUF –EOL var` 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 var` 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 16: 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 1

5.5.1.1 Example for 1.5MB Intel® ME FW SKU

```
MEMANUF –verbose

   Intel(R) MEManuf Version: 9.0.0.xxxx
   Copyright(C) 2005 - 2012, Intel Corporation. All rights reserved.

   FW Status Register1: 0x1E000255
   FW Status Register2: 0x62000006

     CurrentState:                          Normal
     ManufacturingMode:                     Enabled
     FlashPartition:                        Valid
     OperationalState:                      M0 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 - Power Package: Package 1 supported - Passed
Policy Kernel - Power Package: Default package supported - Passed
Policy Kernel - ME Configuration: Wlan Power Well - Passed
Policy Kernel - ME Configuration: CPU Missing Logic - Passed
Policy Kernel - ME Configuration: M3 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
Policy Kernel - Power Package: Live Heap Test - Passed

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

MEManuf Test Passed
```

5.5.1.2 Example for 5MB Intel® ME FW SKU

```
MEMANUF –verbose


Intel(R) MEManuf Version: 9.0.0.xxxx
Copyright(C) 2005 - 2012, Intel Corporation. All rights reserved.


FW Status Register1: 0x1E000255
FW Status Register2: 0x68000006

  CurrentState:                    Normal
  ManufacturingMode:               Enabled
  FlashPartition:                  Valid
  OperationalState:                M0 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

Policy Kernel - Power Package: Live Heap Test - Passed

Common Services - LAN: Connectivity to NIC in M3 - Passed

MicroKernel - Internal Hardware Tests: Internal Hardware Tests - Passed

Policy Kernel - SMBus: Read byte - Passed
Policy Kernel - ME Password: Validate MEBx password - Passed
Policy Kernel - Power Package: Package 1 supported - Passed
Policy Kernel - Power Package: Default package supported - 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: M3 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 M0 - 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 - Power: Power-package 2 supported - Passed
AMT - KVM: USBr 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 - USBr: Hardware - Passed

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

Error 9296: MEManuf Test Failed
```

§

6 MEInfo

MEInfoWin and Intel® MEInfo provide a simple test to check whether the Intel® ME FW is alive or not. Both tools perform the same test; query the Intel® ME FW 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* 7 64/32 bit and Windows* 8 64/32 bit.

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] [-FITCVER] [-FEAT] [-VALUE] [-FWSTS]
           [-VERBOSE] [-PAGE][-PID <filename>] [-DUMPIDLM <filename>]

MEInfo.efi [-EXP] [-H|?] [-VER] [-FITCVER] [-FEAT] [-VALUE] [-FWSTS]
           [-VERBOSE] [-PAGE][-PID <filename>] [-DUMPIDLM <filename>]
```

Table 17: Intel® MEInfo Command Line Options

Option	Description
-FEAT < name> -VALUE <value>	Compares the value of the given feature name 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. 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.
-FITCVER	Displays FITC version information
-FEAT <name>	Retrieves the current value for the specified feature. If the feature name is more than one word, the entire feature name must be enclosed in quotation marks. The feature name entered must be the same as the feature name displayed by Intel® MEInfo. Intel® MEInfo can retrieve all of the information detailed below. However, depending on the SKU selected, some information may not appear. **Note:** When using the EFI based version you must enclose the feature name designation within "^" in order for the EFI shell to parse it properly. Example: meinfo.efi -feat "^"FW Version"^"

–FWSTS	Decodes the Intel® ME FW status register value field and breaks it down into the following bit definitions for easy readability:
	`FW Status Register1: 0x1E000255`
	`FW Status Register2: 0x69000006`
	`CurrentState: Normal`
	`ManufacturingMode: Enabled`
	`FlashPartition: Valid`
	`OperationalState: M0 with UMA`
	`InitComplete: Complete`
	`BUPLoadState: Success`
	`ErrorCode: No Error`
	`ModeOfOperation: Normal`
	`ICC: Valid OEM data, ICC programmed`

-VERBOSE <filename>	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". This option works with no option and `–feat`.
-H or -?:	Displays the list of command line options supported by the Intel® MEInfo tool.
-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.
-PID <filename>	Append/Export Platform ID to the binary file
-DUMPIDLM<filename>	Displays Platform ID list in an IDLM binary

Option	Description
No option:	If the tool is invoked without parameters, it reports information for all components listed in Table 18 below for full SKU FW.

Table 18: List of components that Intel® MEInfo displays

Feature Name	Feature Data Source (ME Kernel/AMT /SW/Other)	Consumer SKU	Corporate SKU	Specific Feature Dependency	Field Value
Tools Version	SW (MEInfo)	X	X	N/A	Version string Example: 9.x.y.ZZZZ; where x=minor, y = HF/MR, ZZZZ = Build Number.
PCH Version	Intel® ME Kernel	X	X	N/A	A version string
FW Version	Intel® ME Kernel	X	X	N/A	Version string 9.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 4M Sku	Version string
GbE Version	Other (Directly reading from SPI)	X	X	GbE Region to be present in the image	A version string
MEBx Version	Intel® ME Kernel	X	X	MEBx needs to be present. Not available on 4M Sku	Version string 9.x.y.ZZZZ; where x=minor, y = HF/MR, ZZZZ = Build Number.
VendorID	Intel® ME Kernel	X	X	N/A	A number (in Hex)
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

Feature Name	Feature Data Source (ME Kernel/AMT /SW/Other)	Consumer SKU	Corporate SKU	Specific Feature Dependency	Field Value
NFC FW Version	NFC	Both	All	N/A	A version string. If NFC HW device is not found/accessible, display "Not Available"
NFC Loader Version	NFC	Both	All	N/A	A version string. If NFC HW device is not found/accessible, display "Not Available"
Link Status	Intel® AMT	X	X	AMT CEM (a.k.a Common Service) is used. Not available on 4M Sku	Link up/down
FW Capabilities	Intel® ME Kernel	X	X	N/A	Combination of feature name list breakdown (with a Hexadecimal value) *This is a display of the Feature State for the Intel® ME. Is enabled / disabled on the system. Each bit in the value represents a feature state. Intel® ME features including Full manageability, standard manageability, Anti-theft technology etc.
TLS Support	Intel® ME Kernel	X	X	N/A	Enabled/Disabled

Feature Name	Feature Data Source (ME Kernel/AMT /SW/Other)	Consumer SKU	Corporate SKU	Specific Feature Dependency	Field Value
BIOS and GbE Config Lock	Other (Directly reading from SPI)	X	X	N/A	Enabled/Disabled/ Unknown If shown as enabled, both FLOCKDN for BIOS and Gbe are set. If shown as disabled, either/all FLOCKDN for BIOS and Gbe are not set.
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
Last Intel® ME Reset Reason	Intel® ME Kernel	X	X	N/A	Power up/ Firmware reset/ Global system reset/ Unknown
Intel® AMT State	Intel® ME Kernel	N/A	X	Both Full Manageability and Manageability Application has to be PRESENT (Capable)	Enabled/Disabled
Intel® Standard Manageability State	Intel® ME Kernel	N/A	X	Full Manageability should not be PRESENT (Capable), but Manageability Application has to be PRESENT	Enabled/Disabled
BIOS Boot State	Intel® ME Kernel	X	X	N/A	Pre Boot/ In Boot/ Post Boot

Feature Name	Feature Data Source (ME Kernel/AMT /SW/Other)	Consumer SKU	Corporate SKU	Specific Feature Dependency	Field Value
System UUID	Intel® AMT	N/A	X	AMT CEM (a.k.a. Common Service) is used. Not available on 4M Sku	UUID of the system
OEM Id	Intel® ME Kernel	X	X	Only if fw image supports OEM Id	UUID for OEM to check during FW Update
Configuration State	Intel® AMT	N/A	X	AMT CEM (a.k.a. Common Service) is used. Not available on 1.5M Sku	Not started/ In process/ Completed/ Unknown
Provisioning Mode	Intel® AMT	N/A	X	AMT CEM (a.k.a. Common Service) is used. Not available on 1.5M Sku	PKI/PSK/ Unknown
MAC Address	Intel® AMT	X	X	AMT CEM (a.k.a. Common Service) is used only when wired Hw is present. Not available on 1.5M Sku	A MAC address (in Hex separated by "=")
Wireless MAC Address	Intel® AMT	X	X	AMT CEM (a.k.a. Common Service) is used only when wireless HW is present. Not available on 1.5M Sku	A MAC address (in Hex separated by "=")

Feature Name	Feature Data Source (ME Kernel/AMT /SW/Other)	Consumer SKU	Corporate SKU	Specific Feature Dependency	Field Value
IPv4 Address (Wired and Wireless)	Intel® AMT	X	X	AMT CEM (a.k.a. Common Service) is used only when wired/wireless Hw is present. Not available on 1.5M Sku	IPv4 IP address (in decimal separated by ".")
IPv6 Address (Wired and Wireless)	Intel® AMT	N/A	X	AMT CEM (a.k.a. Common Service) is used only when wired/wireless Hw is present. Not available on 1.5M Sku	All IPv6 IP addresses
IPv6 enabled (Wired and Wireless)	Intel® AMT	N/A	X	AMT CEM (a.k.a. Common Service) is used only when wired/wireless Hw is present. Not available on 1.5M Sku	Enabled/Disabled
Local FWUpdate	Intel® ME Kernel	X	X	N/A	Enabled/Disabled/ Password Protected
MEI Driver version*	Other (Reading Windows registry entries)	X	X	Only when Windows MEI driver is installed	A version string
LMS version*	Other (Reading Windows registry entries)	X	X	Only when Windows LMS driver is installed	A version string
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 (ME Kernel/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)
Capability Licensing Service	Intel® ME Kernel	X	X	Not available on 4M Sku. Not shown unless Fw feature capability supports it	Enabled/Disabled
Capability Licensing Service Status	Intel® ME Kernel	X	X	Not available on 4M Sku. Not shown unless FW feature capability supports it. This feature is only shown if there is a Level III PCH devices, or the feature is enabled	Permit info not available/ Upgraded/ Not Upgraded/ Not Upgradable
CPU Upgrade State	Intel® ME Kernel (ICLS)	N/A	H65, H67, H61, HM65, HM67	Not available on 4M SKU. Not shown unless Fw feature capability supports it	Upgraded/ Upgrade Capable/ Not Upgradable
Privacy / Security Level	Intel® AMT	X	X	Not available on 4M SKU. Only shown when AMT is enabled	Default/Enhanced/ Extreme/Unknown
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)

Feature Name	Feature Data Source (ME Kernel/AMT /SW/Other)	Consumer SKU	Corporate SKU	Specific Feature Dependency	Field Value
FWSTS	Intel® ME Kernel	X	X	N/A	Two 32bit Hexadecimal numbers and their bit definition breakdown
M3 Autotest	Intel® ME Kernel		X	FITc M3 Autotest Enabled set to 'true'	Enabled/Disabled
Wireless Micro-code Mismatch	FWU	Corporate	All	N/A	Yes: FW has detected a ucode mismatch, and partial FWUpdate needs to be performed
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
Independent Firmware Recovery	FWU	All	All	Only when Windows IFR Agent is installed and the FW image has IFR set to 'true'	Enabled/Disabled

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.

6.3.1 1.5MB Intel® ME FW SKU

MEINFOWIN.exe

```
Intel(R) MEInfo Version: 9.0.0.xxxx
Copyright(C) 2005 - 2011, Intel Corporation. All rights reserved.

Intel(R) Manageability and Security Application code versions:

BIOS Version:                           ACRVMBY1.86C.0038.B00.1103291453
MEBx Version:                           9.0.0.xxxx
Gbe Version:                            1.3
VendorID:                               8086
PCH Version:                            2
FW Version:                             9.0.0.xxxx

FW Capabilities:                        0x00101C60

    Intel(R) Capability Licensing Service - PRESENT/ENABLED
    Protect Audio Video Path - PRESENT/ENABLED
    Intel(R) Dynamic Application Loader - PRESENT/ENABLED

Intel(R) AMT State:                     Enabled
CPU Upgrade State:                      Not Upgradable
TLS Support:                    Enabled
Last ME reset reason:                   Power up
Local FWUpdate:                         Enabled
BIOS and GbE Config Lock:               Disabled
Host Read Access to ME:                 Enabled
Host Write Access to ME:                Enabled
SPI Flash ID #1:                        C22017
SPI Flash ID VSCC #1:                   20052005
SPI Flash BIOS VSCC:                    20052005
BIOS boot State:                        Post Boot
OEM Id:                                 00000000-0000-0000-0000-000000000000
Capability Licensing Service:           Enabled
Capability Licensing Service Status:    Permit info not available
OEM Tag:                                0x00000001
Slot 1 Board Manufacturer:              0x0000203B
Slot 2 System Assembler:                Unused
Slot 3 Reserved:                        Unused
M3 Autotest:                            Disabled
Independent Firmware Recovery:          Disabled
```

6.3.2 5MB Intel® ME FW SKU

```
MEINFOWIN.exe
Intel(R) MEInfo Version: 9.0.0.xxxx
Copyright(C) 2005 - 2011, Intel Corporation. All rights reserved.

Intel(R) Manageability and Security Application code versions:

BIOS Version:                           ACRVMBY1.86C.0038.B00.1103291453
MEBx Version:                           9.0.0.xxxx
Gbe Version:                            1.3
VendorID:                               8086
PCH Version:                            2
FW Version:                             9.0.0.xxxx

FW Capabilities:                        0x0CFE5C47

    Intel(R) Active Management Technology - PRESENT/ENABLED
    Intel(R) Anti-Theft Technology - PRESENT/ENABLED
    Intel(R) Capability Licensing Service - PRESENT/ENABLED
    Protect Audio Video Path - PRESENT/ENABLED
    Intel(R) Dynamic Application Loader - PRESENT/ENABLED

Intel(R) AMT State:                     Enabled
CPU Upgrade State:                      Not Upgradable
TLS Support:                  Enabled
Last ME reset reason:                   Power up
Local FWUpdate:                         Enabled
BIOS and GbE Config Lock:               Disabled
Host Read Access to ME:                 Enabled
Host Write Access to ME:                Enabled
SPI Flash ID #1:                        C22017
SPI Flash ID VSCC #1:                   20052005
SPI Flash BIOS VSCC:                    20052005
BIOS boot State:                        Post Boot
OEM Id:                                 00000000-0000-0000-0000-000000000000
Link Status:                            Link up
System UUID:                            24580e80-ff1d-11e0-96c0-3cd92b79c328
MAC Address:                            3c-d9-2b-79-c3-28
IPv4 Address:                           192.168.0.100
IPv6 Enablement:                        Disabled
Privacy/Security Level:                 Default
Configuration state:                    Completed
Provisioning Mode:                      PKI
Capability Licensing Service:           Enabled
Capability Licensing Service Status:    Permit info not available
OEM Tag:                                0x00000001
Slot 1 Board Manufacturer:              0x0000203B
Slot 2 System Assembler:                Unused
Slot 3 Reserved:                        Unused
M3 Autotest:                            Disabled
Wireless Micro-code Mismatch:           No
Wireless Micro-code ID in Firmware:     0x0082
Wireless LAN in Firmware:               Intel(R) Centrino(R) Ultimate-N 6205
Wireless Hardware ID:                   No Intel WLAN card installed
Wireless LAN Hardware:                  No Intel WLAN card installed
Localized Language:                     English
Independent Firmware Recovery:          Disabled
```

6.3.3 Retrieve the current value of the Flash version

```
C:\ MEInfo.exe -feat "BIOS boot state"
Intel(R) MEInfo Version: 9.0.0.xxxx
Copyright(C) 2005 - 2011, Intel Corporation. All rights reserved.

BIOS boot State: Post Boot

> MEInfo.efi -feat "^"BIOS boot state"^"
Intel(R) MEInfo Version: 9.0.0.xxxx
Copyright(C) 2005 - 2011, Intel Corporation. All rights reserved.
```

BIOS boot State: Post Boot

6.3.4 Checks whether the computer has completed the setup and configuration process

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

Intel(R) MEInfo Version: 9.0.0.xxxx
Copyright(C) 2005 - 2011, 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: 9.0.0.xxxx
Copyright(C) 2005 - 2011, 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 and LOCL partitions that Intel provides on the OEM website. Intel® FWUpdate updates the entire Intel® ME code area. In addition FWUpdate local can perform a partial update to change / update the WCOD or LOCL portions.

The image file that the tool uses for the update is the same image file that is used by the FITC tool to create a firmware image for use in the SPI. A sample FW image file for updating would be '**ME9_5M_Production.bin**'. These files are located in the 'Image Components\ME' sub-folder of the firmware kit.

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.

If Intel® Anti-theft technology is enabled, a system restart must occur to complete the FW update process.

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.

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 please refer to Intel® MEBx user guide.

7.4 Usage

Note: In this section, <Image File> refers to an Intel-provided image file of the section of the FW to be updated, not the image file used in FITC to program the entire flash memory.

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


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

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

Table 19: 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.
-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).
-HALTRCFG	Halt Remote Configuration. The tool halts remote configuration. **Note**: This is NOT an option used with updating the FW image.
–PARTID <wcod or locl>	This option is always used along with the –F option. The partition ID is requested using the “partid” option, which takes in wcod or locl string as input. 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 and invalid file error will be returned by the tool. Also, if the requested partition is not expected by the firmware and 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.
-PASS <PASSWORD>	This is used to specify the Intel® MEBx password to perform the update. A valid password is required to perform the update especially when FW Update setting in Intel® Mebx is set to “password protected”.

Option	Description
-GENERIC	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. **Note**: This option is only supported in the Windows version of the tool.
-FWVER	Display FW version
-H or -?	Displays the list of command line options supported by the Intel® MEInfo tool.
-EXP	Shows examples about how to use the tools.
-VER	Shows the version of the tools.

7.5 Examples

7.5.1 Updates Intel® ME with Firmware binary file

```
This command updates ME with FW.BIN file. If the firmware on current
platform is newer than then version in FW.BIN file, tools will promote a
warning to let user know there will be a firmware downgrade (rollback)
event and let user choose Y/N to continue. User can always use –y to
skip this warning automatically. If the firmware on the platform is the
same as the version in FW.BIN, tools will return an error. User can use
–allowsv to allow same version update.

FWUpdLcl.exe –f FW.BIN

EFI:
FWUpdLcl.efi –f FW.BIN
```

7.5.2 Halt Remote Configuration

```
FWUpdLcl.exe -haltRCFG

EFI:
FWUpdLcl.efi -haltRCFG
```

Calling the `-haltRCFG` option halts all remote configuration traffic and prevents remote configuration. `-haltRCFG` can NOT be used as a command line argument while performing FWUpdate.

7.5.3 Partial Firmware Update

This command will perform a partial update of the FW via MEI for either the wcod or locl partitions.

```
FWUpdLcl.exe -f FW.BIN.bin -partid <wcod or locl>

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

Non-Verbose Mode

```
C:\>FWUpdlcl.exe –f FW.BIN.bin –partid WCOD

Intel (R) Firmware Update Utility version 9.0.0.xxxx
Copyright (C) 2007-2010, 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 9.0.0.xxxx
Copyright (C) 2007-2010, Intel Corporation.  All rights reserved.

Communication Mode: MEI
Sending the update image to FW for verification:  [ COMPLETE ]

Firmware last update status = Firmware update success
Firmware last update reset type = 2

FW Update is completed successfully.
```

7.5.4 Display supported commands

Display a list of supported command line sequences based on the arguments provided. The arguments relevant for this usage are any of the command line options with the prefix '-' removed. The tool will display all valid command sequences based on the options provided. Below is an example which displays valid command sequences with the –ipu option

```
C:\> FWUpdLcl.exe -exp partid

Intel (R) Firmware Update Utility version 9.0.0.xxxx
Copyright (C) 2007-2010, 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 9.0.0.xxxx
Copyright (C) 2007-2010, 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.
```

§

8 Update Parameter Tool

Note: This section is not applicable for 1.5MB Intel® ME FW SKU.

8.1 Purpose of the Tool

UPdParam is used to change certain Intel® ME FW parameters (both Intel® AMT and Kernel) even after the Intel® ME manufacturing mode done bit (global locked bit) is set and the Descriptor region is locked. This tool only works on DOS when BIOS does not send an EOP message.

8.2 Usage of the Tool

```
UpdParam.exe [-?] [-h] [-f] [-v] [-r] [-u] [-ver] [-s] [-c] [-exp] [-
verbose <file>]
```

Table 20: Update Parameter Tool Options

Option	Description
-H|?	Displays help screen
-F <filename>	Inputs USB file name
-V <MeBXCurrPwd>	Overrides Intel® MEBx Admin password
-R	Global reset
-U	Unprovisioning (use this option with `-f<fname>`)
-S	Saves updated parameters as factory defaults on FW image. This feature was implemented in Intel® 7 Series Chipset Family to save the updated parameter as the factory default. This saves the settings even after CMOS is cleared. **Note**: All the other Intel® ME settings – except Intel® MEBx password change – should be saved after the `-s` command is sent.
-C	Commit Option (used with `-f<filename>`). The use of the commit option is the same as in FPT. Based on which parameter gets updated, the tool performs either Intel® ME reset, Global reset, or none. Reset gets performed at the very end (after all the parameters are updated). Global reset is easy to verify that the system is rebooting. To verify whether or not the Intel® ME reset was performed successfully: Run `Meinfo -fwsts`.
-EXP	Displays sample usage of this tool.
-VERBOSE <FILE>	Displays the tool's debug information.

Table 21: Required Reset for Updated Parameters

Parameter	Required Reset
FW Update Local	Intel® ME Reset
Current MEBx password	Intel® ME Reset
New MEBx password	Intel® ME Reset
Manageability Feature selection (Enable AMT)	Intel® ME Reset
PID	Intel® ME Reset
PPS	Intel® ME Reset
PKIDNSSuffix	Intel® ME Reset
ConfiServerFQDN	Intel® ME Reset
ZeroTouchSetupEnabled	Intel® ME Reset
PreInstalledCertEnabled	Intel® ME Reset
UserDefinedCertEnabled	Intel® ME Reset
UserDefinedCertAdd	Intel® ME Reset
SolIderConfig	Intel® ME Reset
HostName	Intel® ME Reset
DomainName	Intel® ME Reset
DHCP	Intel® ME Reset
Idle Timeout	Intel® ME Reset
StaticIPv4Parameters	Intel® ME Reset
KVM State (Enable/Disable)	Intel® ME Reset
KVM Remote IT	Intel® ME Reset
KVM User	Intel® ME Reset
Manual Setup and Configuration	Intel® ME Reset

Note: This table might get updated in future.

8.3 USB Utility

Intel® UPDParam uses as an input a binary file that is created with a USB Utility (**USBfile.exe**).

8.3.1 Syntax

The following parameters can be set in **USBfile.exe** to generate the binary file.

```
USBfile -create <usb output file name> <current MEBx password>
          <new MEBx password> [-v 1|2|2.1|3|4] [-amt] [-rpsk]
          [-v1file <version 1 outfile>]
          [-dns <DNS suffix>] [-fqdn <prov server fqdn>]
          [-consume 0|1]
          [-ztc 0|1]
          [-dhcp 0|1]
          [-sfwu 0|1]
          [-fwu 0|1|2]
          [-pm 0|1]
          [-fwuq 0|1|2]
          [-pspo <port number>]
          [-psadd <ipv4|ipv6 addr>]
          [-ito <4 byte of idle time out>]
          [-nrec <num of records>]
          [-gen <num of records>]
          [-xml <xml file name>]
          [-pid <pid> -pps <pps>]
          [-hash <cert file name> <friendly name>[sha1|sha256|sha384]]
          [-redir <n>]
          [-s4p <StaticIPv4Params>]
          [-hostname <hostname>]
          [-domname <domain name>]
          [-vlan <0|1-VlanTag>]
          [-passPolicyFlag <0|1|2>]
          [-ipv6 <ipv6 xml file name>]
          [-sdFqdn 0|1]
          [-dDnsUpdate 0|1]
          [-kvm 0|1]
          [-userConsentOption 0|1|255]
          [-userConsentPolicy 0|1]
          [-prov 0|1]
          [-conf 0|1]
          [-scIden <4 bytes of support channel identifier>]
          [-scDesc <support channel description>]
          [-sano <service account number>]
          [-enrPass <enrollment passcode>]
          [-servType 1|2|4]
          [-spIden <16 byte GUID>]
```

Table 22: USB Utility Options

Option	Description
-v 1|2|2.1	Setup file version; 2.1 by default
-v1file <version 1 outfile>	Creates a version 1 setup file
-dns <DNS suffix>	Sets the PKI DNS suffix name (up to length 255)
-ztc 0|1	Disables/enables PKI Configuration
-dhcp 0|1	Disables/enables DHCP
-fwu 0|1	Disables/enables FW local update
-pm 0|1	Enterprise/SMB provisioning mode
-pspo <port number>	Provision server port number
-psadd <ip addr>	IP address for provision server (e.g., 123.222.222.121)
-ito <4 byte of idle time out>	4 char of idle time out
-gen <n>	Number of records to create
-xml <xml file name>	Configuration xml file
-pid <pid> -pps <pps>	PSK pair. This is ignored if –gen was chosen
-hash <certificate file name> <friendly name>	Computes and adds the hash of the given root certificate file. Up to three certificate hashes may be specified.
-redir <n>:	An integer that is calculated as follows: bit 0 : 1 (Enable) or 0 (Disable) - SOL feature bit 1 : 1 (Enable) or 0 (Disable) - IDER feature bit 2 : 1 (Enable) or 0 (Disable) - Username/password authentication type of the SOL/IDER in the Intel® ME FW
-s4p <localHost:SubnetMask:GatewayAddr:DNSaddr:SecondaryDNSaddr>	E.g., 10.0.0.1:255.255.255.0:10.0.0.2:10.0.0.3:10.0.0.4 **Note**: The DHCP flag should be disabled.
-hostname <hostname>	ASCII representation of host name. Maximum length 63.
-domname <domain name>	Domain name. Maximum length 255
-vlan <0|1-VlanTag(1-4096)>	VlanStatus enable/disable, e.g., 0-4011
-passPolicyFlag <0|1|2>	Default/block in post/always open

For more details on how to use **USBfile.exe**, use the help command in the USB file utility. Once all parameter modifications have been completed (along with the current Intel® MEBx password) **USBfile.exe** creates a binary file.

For example, the user could enter the command `Usbkey.exe –create test.bin Admin Admin@98` (supposing the System current Intel® MEBx password is Admin). When the user runs **USBfile.exe**, this command creates a binary file named **test.bin** that sets the new password for Intel® MEBx to Admin@98.

Once the binary file is created it is used by the UpdateParam tool as an input.

To use the binary file created by USBfile.exe:

- The binary file must contain the current Intel® MEBx password.
- This tool (UpdateParam) must be in either pre-boot or in-boot mode in order to run:

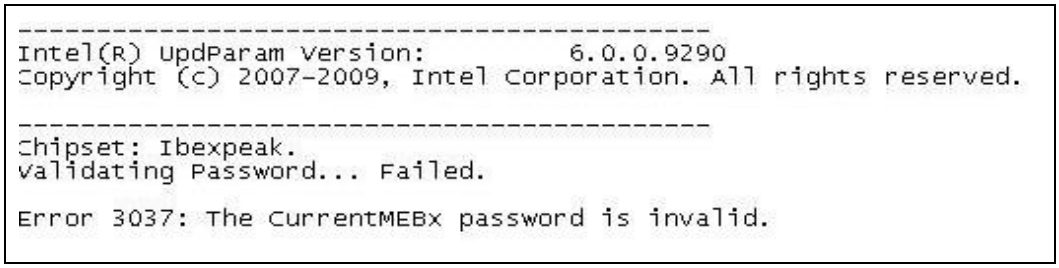
  Pre boot – the platform has just been flashed with an image but default Intel® MEBx password has not been changed yet.

  In-boot – The Intel® MEBx password has been changed and the user has entered the Intel® MEBx interface.
- BIOS does not send an EOP to Intel® ME

8.4 Output

If the binary file contains the right Intel® MEBx password, it proceeds to make the appropriate changes to the settings. It either returns a Success/Fail status for each of the parameters that are in the binary file or the tool returns an error code and error message and exits.

Figure 24: UPDParam Error Message for Incorrect Password



Once the password validation is successfully completed, Intel® UPDParam changes the rest of the parameters as listed in the .bin file. If there is a failure changing/updating any of the parameters, Intel® UPDParam returns the error code and error message associated with the failure.

Figure 25: UPDParam Error Message for Failure to Update Parameter(s)

```
-------------------------------------------
Intel(R) UpdParam Version:         6.0.0.9290
Copyright (c) 2007-2009, Intel Corporation. All rights reserved.

-------------------------------------------
Chipset: Ibexpeak.

Validating Password... Success.
Updating Local Firmware Update Qualifier... Success.
Updating PID/PPS... Success.
Note: No change in ZTC status required.  Same as input.
Updating PID/PPS... Success.
Updating PKI DNS Suffix... Failed..
Error 3: Command is not permitted in current operating mode
Updating Config Server FQDN... Failed..
Error 1: AMT device internal error
Updating SOL/IDER Configuration... Success.
Setting FW update Parameter... Failed.
Setting Host Name... Success.
Setting Domain Name... Success.
Setting Idle Timeout... Success.
Setting Provisioning Mode... Success.
Setting ProServer Port Parameter... Success.
Setting IPv4 Parameters... Success.
Changing Password... Success.
```

Note: Error messages are displayed in red and warning messages are displayed in yellow.

Since Intel® UpdParam uses Intel® MEI to communicate with different components of the Intel® ME it also returns the Intel® MEI status.

A log file is also created that contains details about all the steps executed. The log file can be found in the same folder as the executable.

8.5 Parameters Intel® UpdParam can Change

- Current Intel® MEBx password
- New Intel® MEBx password
- Manageability Feature selection (Enable Intel® AMT)
- FW Local update
- Power package
- PID
- PPS
- PKIDNSSuffix
- ConfiServerFQDN
- ZeroTouchSetupEnabled
- PreInstalledCertEnabled
- UserDefinedCertEnabled
- UserDefinedCertAdd

- SolIderConfig
- HostName
- DomainName
- DHCP
- Idle Timeout
- Provisioning Server Address
- Provisioning server port
- StaticIPv4Parameters
- KVM
- Configuration Mode
- User Consent Policy
- User Consent Option

8.6 Examples

`UpdParam –f <filename>`

Inputs the binary file and updates the parameters.

`UpdParam –f <filename> -v <CurrentMebxPwd>`

Inputs a binary file containing the MEBX current password entered at the command prompt.

`UpdParam –f <filename> -v <CurrentMebxPwd> -u`

Inputs a binary file containing the following:

- MEBX current password entered at the command prompt.
- An option to do partial unprovisioning.

`Updparam –r`

Performs a global reset.

`Updparam –h`

Displays the help screen.

§

9 Appendix A: Fixed Offset Variables

This appendix only covers fixed offset variables that are directly available to FPT and FPTW. A complete list of fixed offset variables 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 `-fov` 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 23: Fixed Offset Item Descriptions

Fixed Offset Name	FPT ID	Fixed Offset ID	Description	Data Length (in Bytes)	Expected Value	Secure	Reset Type
Non-Application Specific Fixed Offset Item Descriptions							
MEBx Password	1	0x0003	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 **See section 2.7 for format and strong password requirements.**	8<=N<=32	Password	**No**	**ME**

<table>
<tr><th>Fixed Offset Name</th><th>FPT ID</th><th>Fixed Offset ID</th><th>Description</th><th>Data Length (in Bytes)</th><th>Expected Value</th><th>Secure</th><th>Reset Type</th></tr>
<tr>
<td>OEM Sku Rule</td>
<td>7</td>
<td>0x000A</td>
<td>UINT32 (little endian) value. This controls what features are permanently disabled by OEM. See Appendix A - Features Supported (Firmware Variable Structures for Intel Management Engine) and Table 5-5 (Firmware Bring-up Guide) for more details. If a feature is grayed out in Table 5-5 (Firmware Bring-up Guide) for that target HW SKU, then the firmware will disregard that selection.
Notes:
There are reserved bits that must not be changed for proper platform operation. The user should only modify the bit(s) for the feature(s) they wish to change. There is NO ability to change features one at a time. This FOV sets OEM Permanent Disable for ALL features. In addition prior updating or changing any of available settings. It is *highly* recommended that the user retrieve the current value using the "FPT -r OEMSkuRule" and toggling only the desired bits, and then resave using "FPT -u -n OEMSkuRule -v <value>".

This will not enable functionality that is not capable of working in the target hardware SKU. Please see the respective Firmware Bring-up Guide for a list of what features are capable with what firmware bundle and Hardware SKU of LynxPoint-H.</td>
<td>4</td>
<td>Feature Capable: 1
Feature Permanently disabled: 0
<table>
<tr><th>Bit</th><th>Description</th><th>Notes</th></tr>
<tr><td>31</td><td>Near Field Communication</td><td></td></tr>
<tr><td>30</td><td>Service Advertisement & Discovery</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>20:19</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>11:6</td><td>Reserved</td><td></td></tr>
<tr><td>5</td><td>Intel® AT</td><td></td></tr>
<tr><td>4:3</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>
</table>
1. For corporate SKUs (Intel® Q87, Intel® QM87 bits 0 and 2 need to be both set to ‘1’ to allow for Intel® AMT to work.
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’.</td>
<td>No</td>
<td>Global</td>
</tr>
</table>

<table>
<tr><th>Fixed Offset Name</th><th>FPT ID</th><th>Fixed Offset ID</th><th>Description</th><th>Data Length (in Bytes)</th><th>Expected Value</th><th>Secure</th><th>Reset Type</th></tr>
<tr><td>Feature Shipment Time State</td><td>8</td><td>0x000B</td><td>UINT32 (little endian) value. This controls what features are enabled or disabled. These features may be enabled /disabled by mechanisms such as MEBx or provisioning. This setting is only relevant for features NOT permanently disabled by the OEM Permanent Disable.
This will not enable functionality that is not capable of working in the target hardware SKU. Please see the respective Firmware Bring-up Guide for a list of what features are capable with what firmware bundle and Hardware SKU of LynxPoint-H.
Notes:
There are reserved bits that must not be changed for proper platform operation. The user should only modify the bit(s) for the feature(s) they wish to change. There is NO ability to change features one at a time. This FOV sets OEM Permanent Disable for ALL features. In addition prior updating or changing any of available settings. It is *highly* recommended that the user retrieve the current value using the "FPT -r OEMSkuRule" and toggling only the desired bits, and then resave using "FPT -u -n OEMSkuRule -v <value>".</td><td>4</td><td>Feature Enabled: 1
Feature Disabled: 0
<table><tr><th>Bit</th><th>Description</th><th>Notes</th></tr><tr><td>31:3</td><td>Reserved</td><td></td></tr><tr><td>2</td><td>Manageability and Security Application</td><td></td></tr><tr><td>1:0</td><td>Reserved</td><td></td></tr></table></td><td>No</td><td>Global</td></tr>
<tr><td>SetWLANPower Well</td><td>35</td><td>0x000E</td><td>Sets which power well the board uses for WLAN cards</td><td>4</td><td>0x80 = Disabled
0x82 = Sus Well
0x83 = ME Well
0x84 = SLP_M# || SPDA
0x86 = WLAN Sleep via SLP_WLAN#</td><td>No</td><td>ME</td></tr>
<tr><td>OEM_TAG</td><td>34</td><td>0x000F</td><td>A human readable 32-bit number to describe the flash image represented by value</td><td>4</td><td>Readable 32 bit hex value identifying the image. Can be empty (Null).</td><td>No</td><td>ME</td></tr>
<tr><td colspan="8">Intel® AMT Related Fixed Offset Item Descriptions</td></tr>
<tr><td>PID</td><td>9</td><td>0x2001</td><td>A 64 bit quantity made up of ASCII codes of some combination of 8 characters – capital alphabets (A–Z), and numbers (0–9). Must be set along with PPS.</td><td>8</td><td>Please see the PSK algorithm section on how to generate a valid PID.</td><td>No</td><td>ME</td></tr>
</table>

Fixed Offset Name	FPT ID	Fixed Offset ID	Description	Data Length (in Bytes)	Expected Value	Secure	Reset Type
PPS	10	0x2002	A 256 bit quantity made up of ASCII codes of some combination of 32 characters – capital alphabets (A–Z), and numbers (0–9). **Must be set along with PID**.	32	Please see the PSK algorithm section on how to generate valid PPS.	**No**	**ME**
OEM Customizable Certificate 1	14	0x200B	Cert Hash Data. See Certificate Hash Entry Structure definition **Note: If the platform is un-configured the Certificate Hash will be deleted.**	55 => n >= 83	Valid Certificate Hash Entry (SHA1, SHA256 or SHA384)	**No**	**ME**
OEM Customizable Certificate 2	15	0x200C	Cert Hash Data. See Certificate Hash Entry Structure definition **Note: If the platform is un-configured the Certificate Hash will be deleted.**	55 => n >= 83	Valid Certificate Hash Entry (SHA1, SHA256 or SHA384)	**No**	**ME**
OEM Customizable Certificate 3	16	0x200D	Cert Hash Data. See Certificate Hash Entry Structure definition **Note: If the platform is un-configured the Certificate Hash will be deleted.**	55 => n >= 83	Valid Certificate Hash Entry (SHA1, SHA256 or SHA384)	**No**	**ME**
USBr Settings	24	0x2017	USBr feature settings	1	**b11** – Enabled **b10** - Disabled Bit mask: Bits 7:0 Bit 0..1 - EHCI 1 enabled (EHCI1Enabled) Bit 2..3 - EHCI 2 enabled (EHCI2Enabled) Bit 4..7 - reserved **At least one of the EHCIs should be enabled. This is not required but recommended.**	**No**	**Global**

Fixed Offset Name	FPT ID	Fixed Offset ID	Description	Data Length (in Bytes)	Expected Value	Secure	Reset Type
Privacy/Security Level	33	0x2019	Redirection (KVM, SOL, IDE-r) privacy level and configuration (RCFG, CCM) settings.	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	**No**	**ME**
EHBC State		0x201A		1	1 = Enabled 0 = Disabled	**No**	**ME**
Revenue Sharing Related FOV Item Descriptions							
ODM_ID		0x5003	FOV 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**
SystemIntegratorId		0x5004	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**

Fixed Offset Name	FPT ID	Fixed Offset ID	Description	Data Length (in Bytes)	Expected Value	Secure	Reset Type
ReservedID		0x5005	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**
Intel® AT Related FOV Item Descriptions							
AT FW Flash Protection Override Policy Hard HDA_SDO	27	0x6001	Indicates whether Hardware descriptor override (HDA_SDO) is allowed, and under what conditions.	1	Always Allowed: **0x01** Allowed when AT NOT provisioned: **0x02**	**No**	**ME**
AT FW Flash Protection Override Policy Soft HMRPFO	28	0x6002	Indicates whether Software descriptor override (HMRPFO) is allowed, and under what conditions.	1	Always Allowed: **0x01** Allowed when AT NOT provisioned: **0x02**	**No**	**ME**

Note: All Fixed Offset Variables (FOVs) have corresponding Named Variables (NVARs) however not all Named Variables (NVARs) have Firmware Offset Variables (FOVs) associated with them.

Additionally some Fixed Offset Variables (FOVs) have different name designations than Named Variable (NVARs) counterparts.

FPT NVAR Retrieve command:
fpt.exe –r <name> | all [-f <file>] [options]

Required Parameters

<name> Name of NVAR OR All retrieves all the NVARs

FPT FOV / NVAR naming Comparison	
Named Variables (NVARs)	**Fixed Offset Variables (FOVs)**
MEBxPassword	MEBxPassword
Default Power Package	DefPwrPackage
OEMSkuRule	OEMSkuRule
FeatureShipState	FeatureShipState
WLAN Well Power Config	SetWLANPowerWell
OEM_TAG	OEM_TAG
PID	PID
PPS	PPS
Idle Timeout - Manageability Engine	MEIdleTimeout
OEM Customizable Certificate 1	OEMCustomCert1
OEM Customizable Certificate 2	OEMCustomCert2
OEM Customizable Certificate 3	OEMCustomCert3
USBrSettings	USBrSettings
Privacy/Security Level	Privacy/SecurityLevel
ODM ID used by Intel (R) Services	ODM_ID
System Integrator ID used by Intel (R) Services	SystemIntegratorId

FPT FOV / NVAR naming Comparison	
Named Variables (NVARs)	**Fixed Offset Variables (FOVs)**
Reserved ID used by Intel (R) Services	ReservedId
Flash Protection Override Policy Hard	ATFPOPHard
Flash Protection Override Policy Soft	ATFPOPSoft
All remaining NVARS	All remaining NVARs do not have corresponding FOVs to allow configuration post image creation

§

10 Appendix B: Tool Detail Error Codes

A.1 Common Error Code for all Tools

Error Code	Error Message	Response
0	Success	
1	Memory allocation error occurred	Make sure there is enough memory in the system
2	Invalid descriptor region	Check descriptor region
3	Region does not exist	Check region to be programmed
4	Failure. Unexpected error occurred	Contact Intel
5	Invalid data for Read ID command	Contact Intel
6	Error occurred while communicating with SPI device	Check SPI device
7	Hardware sequencing failed. Make sure that access permissions are correct for the target flash area	Check descriptor region access settings
8	Software sequencing failed. Make sure that access permissions are correct for the target flash area	Check descriptor region access settings
9	Unrecognized value in the HSFSTS register	Unrecognized value in the HSFSTS register
10	Hardware Timeout occurred in SPI device	Hardware Timeout occurred in SPI device
11	AEL is not equal to zero	AEL is not equal to zero
12	FCERR is not equal to zero	FCERR is not equal to zero
25	The host CPU does not have write access to the target flash area. To enable write access for this operation the user needs to modify the descriptor settings to give host access to this region.	Check descriptor region access settings
26	The host CPU does not have read access to the target flash area. To enable read access for this operation the user needs to modify the descriptor settings to give host access to this region.	Check descriptor region access settings
27	The host CPU does not have erase access to the target flash area. To enable erase access for this operation the user needs to modify the descriptor settings to give host access to this region.	Check descriptor region access settings

Error Code	Error Message	Response
28	Protected Range Registers are currently set by BIOS, preventing flash access. Contact the target system BIOS vendor for an option to disable Protected Range Registers.	Assert Flash Descriptor Override Strap (GPIO33) to Low, Power Cycle, and Retry. If Protected Range Registers (memory location: SPIBAR + 74h -> 8Fh) are still set, contact the target BIOS vendor.
50	General Erase failure	Attempt the command again. If it fails again, contact Intel.
51	An attempt was made to read beyond the end of flash memory	Check address
52	An attempt was made to write beyond the end of flash memory	Check address
53	An attempt was made to erase beyond the end of flash memory	Check address
54	The address <address> of the block to erase is not aligned correctly	Check address
55	Internal Error	Contact Intel
56	The supplied zero-based index of the SPI Device is out of range.	The supplied zero-based index of the SPI Device is out of range.
57	AEL or FCERR is not equal to zero for Software Sequencing	AEL or FCERR is not equal to zero for Software Sequencing
75	File not found	Check file location
76	Access was denied opening the file	Check file location
77	An unknown error occurred while opening the file	Verify the file is not corrupt
78	Failed to allocate memory for the flash part definition file	Check system memory Verify the file is not corrupt
79	Failed to read the entire file into memory	Check system memory Verify the file is not corrupt
80	Parsing of file failed	Check system memory Verify the file is not corrupt
100	This error can occur if both Software and Hardware sequencing are not available and the SPI Flash configuration registers are write protected by the Flash Configuration Lock-Down bit (FLOCKDN). Contact the BIOS vendor to unlock this bit or enable hardware sequencing in descriptor mode.	Check with BIOS vendor or SPI programming Guide

Error Code	Error Message	Response
101	No SPI flash device could be identified. Please verify if Fparts.txt has support for this part	Verify **Fparts.txt** contains device supported.
102	Failed to read the device ID from the SPI flash part	Verify **Fparts.txt** has correct values
103	There are no supported SPI flash devices installed. Check connectivity and orientation of SPI flash device	Verify **Fparts.txt** has correct values. Check SPI Device
104	The two SPI flash devices do not have compatible command sets	Verify both SPI devices on the system are compatible
105	An error occurred while writing to the write status register of the SPI flash device. This program will not be able to modify the SPI flash	Check SPI Device
202	Confirmation is not received from the user to perform operation.	
203	Flash is not blank	
204	Data verify mismatch found	
205	Unexpected failure occurred	
207	Invalid parameter value specified by user. The option specified cannot be run on a platform with Intel (R) ME Ignition FW	
208	Intel® ME is disabled	
209	Intel® ME failed to reset	
210	Requesting Intel® ME FW Reset failure.	
211	Communications error between FPT and the ME.	
212	The request to disable the ME failed.	
213	Intel® ME disable is not required	
214	Intel® ME is already disabled	
215	The attempt to commit the FOVs has failed.	
216	The Close Manufacturing process failed.	
217	Setting Global Reset Failed	
240	Access was denied opening the file	
241	Access was denied creating the file	
242	An unknown error occurred while opening the file	
243	An unknown error occurred while creating	
244	Not a valid file	
245	file not found error	
246	Failed to read the entire file into memory	
247	Failed to write the entire flash contents to file	

Error Code	Error Message	Response
248	file already exists	
249	The file is longer than the flash area to write.	
250	The file is smaller than the flash area to write.	
251	Length of image file extends past the flash area.	
252	Image file not found.	
253	file does not exist	
254	Not able to open the file	
255	Error occurred while reading the file	
256	Error occurred while writing to the file	
280	Failed to disable write protection for the BIOS space	
281	The Enable bit in the LPC RCBA register is not set. The value of this register cannot be used as the SPI BIOS base address.	
282	Failed to get information about the installed flash devices	
283	Unable to write data to flash.	
284	Fail to load driver (PCI access for Windows). The tool needs to run with an administrator privilege account.	
320	FPT General failure error	
321	The address is outside the boundaries of the flash area.	
360	Invalid Block Erase Size value in	
361	Invalid Write Granularity value in	
362	Invalid Enable Write Status Register Command value	
363	Invalid Chip Erase Timeout value	
360	Invalid Block Erase Size value in	
361	Invalid Write Granularity value in	
362	Invalid Enable Write Status Register Command value	
363	Invalid Chip Erase Timeout value	
360	Invalid Block Erase Size value in	
361	Invalid Write Granularity value in	
362	Invalid Enable Write Status Register Command value	

Error Code	Error Message	Response
363	Invalid Chip Erase Timeout value	
440	Invalid Fixed Offset variable name	
441	FOV invalid variable ID	
442	Param file is already opened	
443	FOV exists already	
444	Invalid name or Id of FOV	
445	Invalid length of FOV value. Check FOV configuration file for correct length	
446	Password does not match the criteria.	
447	Error occurred while reading FOV configuration file	
448	Invalid hash certificate file	
449	Valid PID/PPS/Password records are not found in	
450	Invalid ME Manufacturing Mode Done value entered	
451	Unable to get master base address from the descriptor.	
452	Verification of End Of Manufacturing settings failed	
453	End Of Manufacturing Operation failure - Verification failure on ME Manufacturing Mode Done settings	
454	End Of Manufacturing Operation failure - Verification failure on Intel® ME Manuf counter.	
455	End Of Manufacturing Operation failure - Verification failure on Descriptor Lock settings.	
456	Invalid hexadecimal value entered for the FOV	
457	Parsing of file failed	
480	The setup file header has an illegal UUID	
481	The setup file version is unsupported	
482	a record has been encountered that does not contain an entry with the Current MEBx Password	
483	the given buffer length is invalid	
484	the record chunk count cannot contain all of the setup file record data	
485	the setup file header indicates that there are no valid records (RecordsConsumed >= RecordCount)	
486	the given buffer is invalid	

Error Code	Error Message	Response
487	A record entry with an invalid Module ID was encountered.	
488	A record was encountered with an invalid record number.	
489	The setup file header contains an invalid module ID list.	
490	The setup file header contains an invalid byte count.	
491	The setup file record id is not found	
492	The list of data record entries is invalid.	
493	The CurrentMEBx password is invalid.	
494	The NewMEBx password is invalid.	
495	The PID is invalid.	
496	The PPS is invalid.	
497	The PID checksum failed.	
498	The PPS checksum failed.	
499	The data record is missing a CurrentMEBx password entry.	
500	The data record is missing a NewMEBx password entry.	
501	The data record is missing a PID entry.	
502	The data record is missing a PPS entry.	
503	The header chunk count cannot contain all of the setup file header data.	
504	The requested index is invalid.	
505	Failed to write to the given file.	
506	Failed to read from the given file.	
507	Failed to create random numbers.	
508	The data record is missing a PKI DNS Suffix entry.	
509	The data record is missing a Config Server FQDN entry.	
510	The data record is missing a ZTC entry.	
511	The data record is missing a Pre-Installed Certificate enabled entry.	
512	The data record is missing a User defined certificate config entry.	

Error Code	Error Message	Response
513	The data record is missing a User defined certificate Add entry.	
514	The data record is missing a SOL/IDER enable entry.	
515	OEM Firmware Update Qualifier data missing in USB file.	
1000	Invalid command line option(s)	
1001	Unsupported OS	
8192	General error	
8193	Cannot locate ME device	
8194	Memory access failure	
8195	Write register failure	
8196	OS failed to allocate memory	
8197	Circular buffer overflow	
8198	Not enough memory in circular buffer	
8199	Communication error between application and Intel® ME <HECI command name>	Contact Intel
8200	Unsupported HECI bus message protocol version	
8201	Unexpected interrupt reason	
8202	Intel® AMT device unavailable	
8203	Unexpected result in command response <HECI command name>	Contact Intel
8204	Unsupported message type	
8205	Cannot find host client	
8206	Cannot find Intel® ME client	
8207	Client already connected	
8208	No free connection available	
8209	Illegal parameter	
8210	Flow control error	
8211	No message	
8212	Requesting HECI receive buffer size is too large	
8213	Application or driver internal error	
8214	Circular buffer not empty	

A.2 Firmware Update Errors

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

Error Code	Error Message
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
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. Please verify the update image used.
8787	Password exceeded maximum number of retries.

A.3 Intel® MEManuf Errors

Error Codes	Error Messages
9248	Intel® ME internal communication error (BIST)
9249	Intel® ME internal communication error (FW)
9251	Fail to create verbose log file %s Where %s is the log file name user specified
9255	Internal error
9256	Communication error between host application and Intel® ME FW
9257	Cannot run the command since Intel®AMT is not available

Error Codes	Error Messages
9261	Hibernation isn't supported by the OS, Intel(R) ME test cannot run
9267	Fail to establish a communication with SPI flash interface
9268	Fail to load vsccommn.bin
9269	Zero flash device found for VSCC check
9270	Fail to load driver (PCI access for Windows) Tool needs to run with an administrator priviledge account.
9271	Flash ID 0x%06X Intel® ME VSCC mismatch Programmed value of 0x%X doesn't match the recommended value of 0x%X See PCH SPI programming Guide for more details
9272	No recommended ME VSCC value found for flash ID 0x%06X
9276	Fail to read FW Status Register value 0x%X
9278	Cannot locate hardware platform identification This program cannot be run on the current platform. Unknown or unsupported hardware platform or A %s hardware platform is detected This program cannot be run on the current platform. Unknown or unsupported hardware platform Where %s is the offical name of the hardware platform
9279	SPI flash Intel(R) ME region is not locked
9280	Intel(R) Gbe/ME has read or write access to BIOS region
9281	SPI flash descriptor region is not locked
9282	BIOS has granted Intel(R) Gbe and/or ME access to its region
9283	Region access permissions don't match Intel recommended values

Error Codes	Error Messages
9284	Read firmware flash master region permission failure
9296	MEManuf Test Failed Or MEManuf End-Of-Line Test Failed Or MEManuf Operation Failed
9299	Single flash part found, Flash Partition Boundary Address must be zero
9300	Flash Partition Boundary Address should be in between flash parts
9301	The two flash parts on this platform require different BIOS VSCC values
9303	Memory allocation failed for checking variable "<Variable Name>"
9304	Variable "<Variable Name>" mismatch, actual value is - <Variable Value>
9305	Intel® ME firmware version mismatch, actual value is - <Version String> Intel® Gbe version mismatch, actual value is - <Version String> BIOS version mismatch, actual value is - <Version String>
9306	System UUID mismatch, actual value is - <UUID> System UUID mismatch, feature is not supported
9307	Intel® Wired/Wireless LAN MAC address mismatch, feature is not supported Intel® Wired/Wireless LAN MAC address mismatch, actual value is - <MAC Address>
9308	Security Descriptor Override Strap (SDO) is enabled
9309	End-Of-Post message is not sent
9310	Unable to determine Intel® ME Manufacturing Mode status Intel® ME is still in Manufacturing Mode
9311	Intel® ME test failed to start, error 0x%X returned
9312	Intel® ME test timeout (exceeded 30 seconds)
9315	Intel® ME test is currently running, try again
9316	Intel® ME cannot run Full BIST. Possible Causes: (1) Power package 2 not supported, (2) This is a mobile system with DC power

Error Codes	Error Messages
9317	No valid OEM ICC data programmed
9318	MEManuf End-Of-Line Test config file generation failed
9320	Internal error
9321	MEManuf End-Of-Line Test Failed
9322	MEManuf Operation Failed
9324	M3 results are not available from SPI. Please run –test option to perform the BIST test
9325	Failed to delete M3 results from SPI
9326	M3 test failed
9327	M3 test failed
9328	Internal error
9329	Internal error
9330	Internal error
9331	SMBus hardware is not ready
9332	Internal error
9333	SMBus encountered time-out
9334	Failed to retrieve password from SPI
9335	Internal error
9336	Internal error
9337	Internal error
9338	Failed to retrieve test result from SPI
9339	Failed to retrieve power rule from SPI
9340	Failed to retrieve power source
9341	Failed to retrieve PROC_MISSING_NVAR setting
9342	PROC_MISSING_NVAR setting is set incorrectly
9343	Internal error
9344	Failed to retrieve power package setting

Error Codes	Error Messages
9345	Failed to retrieve M3Power Rails Availability setting
9346	M3 Power Rails Availability setting is set incorrectly
9347	Power source is not AC
9348	Internal error
9349	Internal error
9350	Internal error
9351	Length of OEM Customizable Certificate Friendly Name setting is set incorrectly
9352	OEM Customizable Certificate Stream setting is set incorrectly
9353	OEM Customizable Certificate Hash Algorithm setting is set incorrectly
9354	Length of OEM Customizable Certificate Stream is set incorrectly
9355	Current WLAN does not match micro-code, please update WLAN micro-code in FW
9356	Communication with WLAN device failed
9357	WLAN power well setting is set incorrectly
9358	LAN power well setting is set incorrectly
9359	Power Pkg 2 Supported is set incorrectly
9360	USBr EHCI 1 Enabled and/or USBr EHCI 2 Enabled setting is set incorrectly
9361	KVM device is already in use by other components
9362	Internal error
9363	Internal error
9364	The compressed data is incorrect
9365	Intel integrated LAN setting is set incorrectly
9366	Intel LAN connected Device (PHY) physical connectivity error with ME
9367	Firmware is in recovery mode
9368	SMBus address is not configured correctly
9369	Could not register for SMBus alert
9370	Communication interference

Error Codes	Error Messages
9371	SMBUS connection failed. Check connection or SMBUS address
9372	GPIO connection failed. Check connection or GPIO configuration
9373	NFC Radio – Unknown error
9374	NFC RF Test – Error returned from radio
9375	NFC RF Test – Communication interference or bad response returned from radio
9376	NFC RF Test – Timeout

A.4 Intel® MEInfo Errors

Error Code	Error Messages
9450	Communication error between application and Intel® AMT module (Manageability client)
9451	Communication error between application and Intel® AMT module (PTHI client)
9452	Communication error between application and Intel® ME module (iCLS client)
9455	Failed to read FW Status Register value 0x%X
9457	Failed to create verbose log file %s: Where %s is the log file name user specified
9458	Communication error between application and Intel® ME module (FW Update client)
9459	Internal error (Could not determine FW features information)
9460	Cannot locate hardware platform identification This program cannot be run on the current platform. Unknown or unsupported hardware platform Or A %s hardware platform is detected This program cannot be run on the current platform. Unknown or unsupported hardware platform Where %s is the official name of the hardware platform
9461	Communication error between application and Intel® ME module (HCI client)
9462	Communication error between application and Intel® ME module (Kernel Client)
9467	Cannot use zero as SPI Flash ID index number
9468	Couldn't find a matching SPI Flash ID
9469	Access to SPI Flash device(s) failed

Error Code	Error Messages
9470	Failed to load driver (PCI access for Windows) Tool needs to run with an administrator privilege account.
9471	Invalid feature name XXXXX: Where XXXXX is the feature name
9472	XXXXX feature was not available: Where XXXXX is the feature name
9473	XXXXX actual value is – YYYYY: Where XXXXX is the feature name Where YYYY is the feature value
9474	Error reporting revenue share information – Invalid index used
9475	Error reporting revenue share information – Index already in use
9476	Error reporting revenue share information – Slot is empty

A.5 FPT Errors

Error Code	Error
Invalid Parameters	
200	Invalid parameter value specified by the user. Use -? Option to see help.
Invalid Verbose File	
254	Not able to open the file <FILENAME>.
Unsupported Platform	
201	<EXENAME> cannot be run on the current platform. Please contact your vendor.
Unsupported OS	
9254	Unsupported OS
Commit FOVs Operation	
517	Get NVAR - Read Failed
518	Get NVAR - Invalid NVAR specified
519	Get NVAR - Out of Memory
520	Get NVAR - Blob Integrity Failed
8193	Intel® ME Interface : Cannot locate ME device driver
8199	Intel® ME Interface : ME Device not ready for data transmission
8204	Intel® ME Interface : Unsupported message type
8213	Intel® ME Interface : Buffer too small
Compare FOV(s) Operation	
518	Get NVAR - Invalid NVAR specified
519	Get NVAR - Out of Memory
520	Get NVAR - Blob Integrity Failed
8193	Intel® ME Interface : Cannot locate ME device driver
8199	Intel® ME Interface : ME Device not ready for data transmission
8204	Intel® ME Interface : Unsupported message type
8213	Intel® ME Interface : Buffer too small
Retrieve NVAR Operation	
518	Get NVAR - Invalid NVAR specified
519	Get NVAR - Out of Memory

Error Code	Error
520	Get NVAR - Blob Integrity Failed
8193	Intel® ME Interface : Cannot locate ME device driver
8199	Intel® ME Interface : ME Device not ready for data transmission
8204	Intel® ME Interface : Unsupported message type
8213	Intel® ME Interface : Buffer too small
Updating Parameters Operations	
493	The Current MEBx Password is invalid.
506	Failed to read from the given file.
3003	Error occurred while opening image file
3004	Parsing of image file failed
3005	Heci communication failed
3006	File does not exist
3007	Operating system is not supported
3008	Intel® AMT Internal error occurred
3009	User defined certificate hash table is full
3010	Unable to start HECI
3011	Invalid input file name
3012	Chipset not supported by the tool
3013	PID value is NULL
3014	PPS value is NULL
3015	Configuration Server FQDN value is NULL
3016	PKI DNS Suffix value is NULL
3017	Host Name value is NULL
3018	Domain Name value is NULL
3054	Unable to create Logfile
3055	System failed to retrieve current firmware feature state.
3056	Unable to Save updated parameter as factory defaults on FW image.
3057	Unable to complete FOV commit option.

A.6 UPDPARAM Errors:

Note: This section is not applicable to 1.5MB FW SKU.

Error Codes	Description
0	Success
3001	Invalid arguments specified
3002	Invalid Parameter value
3003	Error occurred while opening image file
3004	Parsing of image file failed
3005	Heci communication failed
3006	File does not exist
3007	Operating system is not supported
3008	Intel® AMT Internal error occurred
3009	User defined certificate hash table is full
3010	Unable to start HECI
3011	Invalid input file name
3012	Chipset not supported by the tool
3013	PID value is NULL
3014	PPS value is NULL
3015	Configuration Server FQDN value is NULL
3016	PKI DNS Suffix value is NULL
3017	Host Name value is NULL
3018	Domain Name value is NULL
3019	The setup file header has an invalid UUID
3020	The setup file version is unsupported
3021	A record has been encountered that does not contain an entry with the Current Intel® MEBx Password
3022	The given buffer length is invalid
3023	The header chunk count cannot contain all of the setup file header data
3024	The record chunk count cannot contain all of the setup file record data
3025	The requested index is invalid
3026	The setup file header indicates that there are no valid records
3027	The given buffer is invalid

Error Codes	Description
3028	A record entry with an invalid Module ID was encountered
3029	A record was encountered with an invalid record number
3030	The setup file header contains an invalid module ID list
3031	he setup file header contains an invalid byte count
3032	The setup file record id is invalid
3033	The list of data record entries is invalid
3034	Failed to write to the given file
3035	Failed to read from the given file
3036	Failed to create random numbers
3037	The CurrentMEBx password is invalid
3038	The NewMEBx password is invalid
3039	The PID is invalid
3040	The PPS is invalid
3041	The data record is missing a CurrentMEBx password entry
3042	The data record is missing a NewMEBx password entry
3043	The data record is missing a PID entry
3044	The data record is missing a PPS entry
3045	The data record is missing a PKI DNS Suffix entry.
3046	The data record is missing a Config Server FQDN entry
3047	The data record is missing a ZTC entry
3048	The data record is missing a Pre-Installed Certificate enabled entry
3049	The data record is missing a User defined certificate config entry
3050	The data record is missing a User defined certificate Add entry
3051	The data record is missing a SOL/IDER enable entry
3052	Firmware feature data missing in USB File
3053	OEM Firmware Update Qualifier data missing in USB file
3054	Unable to create Logfile
3055	System failed to retrieve current firmware feature state.

§

11 Appendix C: Tool Option Dependency on BIOS/Intel® ME Status

Tools' Options	Intel® ME manufacturing mode donebit		End of post		CF9GR locking	
	1	0	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® MEMANUF –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