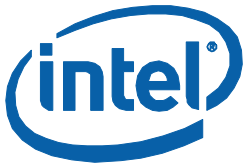


Intel® Management Engine System Tools User Guide

User Guide

November 2008

Revision 1.92

Intel Confidential

Intel Software License Agreement

IMPORTANT—READ BEFORE COPYING, INSTALLING OR USING.

Do not use or load this software or any associated materials (collectively, the “Software”) until you have carefully read the following terms and conditions. By loading or using the Software, you agree to the terms of this Agreement. If you do not wish to so agree, do not install or use the Software.

LICENSE—Subject to the restrictions below, Intel Corporation ("Intel") grants you the following limited, revocable, non-exclusive, non-assignable, royalty-free copyright licenses in the Software.

The Software may contain the software and other property of third party suppliers, some of which may be identified in, and licensed in accordance with, the “license.txt” file or other text or file in the Software:

DEVELOPER TOOLS—including developer documentation, installation or development utilities, and other materials, including documentation. You may use, modify and copy them internally for the purposes of using the Software as herein licensed, but you may not distribute all or any portion of them.

RESTRICTIONS—You will make reasonable efforts to discontinue use of the Software licensed hereunder upon Intel’s release of an update, upgrade or new version of the Software.

You shall not reverse-assemble, reverse-compile, or otherwise reverse-engineer all or any portion of the Software.

Use of the Software is also subject to the following limitations:

You,

(i) are solely responsible to your customers for any update or support obligation or other liability which may arise from the distribution of your product(s)

(ii) shall not make any statement that your product is "certified," or that its performance is guaranteed in any way by Intel

(iii) shall not use Intel's name or trademarks to market your product without written permission

(iv) shall prohibit disassembly and reverse engineering, and

(v) shall indemnify, hold harmless, and defend Intel and its suppliers from and against any claims or lawsuits, including attorney's fees, that arise or result from your distribution of any product.

OWNERSHIP OF SOFTWARE AND COPYRIGHTS—Title to all copies of the Software remains with Intel or its suppliers. The Software is copyrighted and protected by the laws of the United States and other countries, and international treaty provisions. You will not remove, alter, deface or obscure any copyright notices in the Software. Intel may make changes to the Software or to items referenced therein at any time without notice, but is not obligated to support or update the Software. Except as otherwise expressly provided, Intel grants no express or implied right under Intel patents, copyrights, trademarks, or other intellectual property rights. You may transfer the Software only if the recipient agrees to be fully bound by these terms and if you retain no copies of the Software.

LIMITED MEDIA WARRANTY—If the Software has been delivered by Intel on physical media, Intel warrants the media to be free from material physical defects for a period of ninety (90) days after delivery by Intel. If such a defect is found, return the media to Intel for replacement or alternate delivery of the Software as Intel may select.

EXCLUSION OF OTHER WARRANTIES—EXCEPT AS PROVIDED ABOVE, THE SOFTWARE IS PROVIDED "AS IS" WITHOUT ANY EXPRESS OR IMPLIED WARRANTY OF ANY KIND INCLUDING WARRANTIES OF MERCHANTABILITY, NONINFRINGEMENT, OR FITNESS FOR A PARTICULAR PURPOSE. Intel or its suppliers do not warrant or assume responsibility for the accuracy or completeness of any information, text, graphics, links or other items contained in the Software.

LIMITATION OF LIABILITY—IN NO EVENT SHALL INTEL OR ITS SUPPLIERS BE LIABLE FOR ANY DAMAGES WHATSOEVER (INCLUDING, WITHOUT LIMITATION, LOST PROF¬ITS, BUSINESS INTERRUPTION, OR LOST INFORMATION) ARISING OUT OF THE USE OF OR IN¬ABILITY TO USE THE SOFTWARE, EVEN IF INTEL HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. SOME JURISDICTIONS PROHIBIT EXCLUSION OR LIMITA¬TION OF LIABILITY FOR IMPLIED WARRANTIES OR CONSEQUENTIAL OR INCIDENTAL DAMAGES, SO THE ABOVE LIMITA¬TION MAY NOT APPLY TO YOU. YOU MAY ALSO HAVE OTHER LEGAL RIGHTS THAT VARY FROM JURISDICTION TO JURISDICTION.

Contents

Figures

Tables

Revision History

Revision Number	Description	Revision Date
0.3	Pre Alpha	07/20/2007
0.31	iTPM mods	08/08/2007
0.40	Alpha 1 release	08/28/2007
0.60	Alpha 2 release	10/24/2007
0.70	Alpha 3 release	12/05/2007
0.80	Beta 1 Release	02/04/2008
1.2	PC Release	05/23/2008
1.3	PV Release	06/03/2008
1.6	Hotfix Release	07/30/2008
1.7	Hotfix Release – New TPM parameters in FITC have been added.	08/13/2008
1.8	Updates to FITC parameters	08/29/2008
1.91	Hotfix Release new TPM setting in FPT tool and a new feature in FWupdate tool	11/07/2008
1.92	Add Win7 support	05/29/2009

§

1 Introduction

The purpose of this document is to provide guidance on the usage of the tools that are used in the BIOS programming process and its testing.

1.1 Terminology

Term	Description
BIOS	Basic Input-Output System
Complete SPI Image	An image that contains Descriptor, BIOS, Intel® Management Engine (Intel® ME) Firmware, GBE and PDR Region
FW	Firmware, specifically firmware executing on the Intel® ME.
IDE-R	IDE Redirection
Intel® AMT	Intel® Active Management Technology.
Intel® TPM	Intel® Trusted Platform Module—Compliant with TPM 1.2 Specification
RCFG	Remote Configuration
SNMP	Simple Network Management Protocol
SOAP	Standard Object Access Protocol
SOL	Serial Over LAN
TLS	Transport Layer Security

1.2 Reference Documents

Document	Document No./Location
OEM Bring Up Guide	Release kit
Intel® AMT Web UI Guide	<TBD>
System Tools User Guide	Release Kit
Users Guide to the Setup and Configuration Application	iAMT tools\iamtconfiguration
Firmware Variable Structures for Intel® Management Engine and Intel® Active Management Technology - Intel® Centrino® Pro Processor Technology (Cantiga)	Anacapa# 24988
All trusted computing literature	http://www.trustedcomputing.com
Intel® TPM Tools User Guide	Release Kit

§

2 Preface

2.1 Overview

The system tools described in this document create, modify and write binary image files. A brief overview of the tools follows. Instructions on the usage of these tools are divided between:

- The Validation Tools User Guide
- Intel® TPM Tools User Guide.

The following tools are described in the Validation Tools User Guide:

- AMTVTL—AMT Validation Tool Local
- AMTVTR—AMT Validation Tool Remote
- AMTRedirection.

2.2 Manufacturing Line Validation Tools

Manufacturing line validation tools allow testing of Intel® AMT and Intel® TPM functionality immediately after the platform silicon is generated. These tools are designed to be able to run quickly and on simple operating systems, such as, MS-DOS 6.22, Windows* 98 DOS, FreeDOS*, and DRMK DOS*. The Windows versions are written to run on Windows* XP (SP1/2) and Windows Vista*.

MeManuf and MEManufWin—these tools are used to validate Intel® AMT and Intel® TPM functionality on the manufacturing line.

2.3 Image Editing Tools

The following tools create and write flash images:

- Flash Image Tool (FITC)—combines the GBE, BIOS, PDR and Intel® ME firmware into one image that can be programmed by a flash programming device or the Flash Programming Tool (FPT).
- Flash Programming Tool—programs the flash memory. This tool can program individual regions or the entire flash device.
- FWUpdate—updates the firmware code of a flash device that has already been programmed with a complete SPI image.

2.4 Requirements

Manufacturing line validation tools are qualified to run on the following operating systems:

- MS-DOS* 6.22
- Windows* 98 DOS
- Windows* XP (SP1/2)
- Windows Vista* 32/64.

Integration validation tools run on Windows (Windows*XP SP1/2, Windows* PE, and Windows Vista*) or DOS.

Note: Not all tools are supported in DOS. Not all tools are supported in Windows Vista* 64

Integration validation tools that run locally on the Intel® AMT device require one or more of the following services to be installed:

- Intel® Management Engine Interface (Intel® MEI) driver.
- Intel® AMT Local Manageability Service (LMS)
- Intel® TPM driver.
- Microsoft .NET* Framework version 2.0 Redistributable package (x86)
  To download, please visit http://www.microsoft.com/downloads

Check individual tool descriptions for the exact requirements.

Table 1. Tools Summary

Tool Name	Feature Tested	Runs on Intel® AMT device	Runs on Management System
MEManuf and MEManufWin	Connectivity between Intel® ME Devices	X	
MEInfo and MEInfoWin	Firmware Aliveness—outputs certain Intel® ME parameters	X	
Flash Programming Tool (FPT)	Programs the image onto the flash memory	X	
Flash Image Tool (FITC)	Prepares the image files to be programmed onto the flash programming tool	X	X
Firmware Update	Updates the firmware code while maintaining the values previously set	X	

§

3 Flash Image Tool (FITC)

The Flash Image Tool (FITC) creates and configures a complete SPI image file for the Montevina platform. The FITC takes a combination of the following regions in the form of binary files, and assembles them into a single flash image:

- BIOS
- Gigabit Ethernet
- Intel® Management Engine (Intel® ME)
- Platform Descriptor Region (PDR).

The user is able to manipulate the complete SPI image via a Graphical User Interface (GUI) and change the various chipset parameters to match the target hardware. Various configurations can be saved to independent files, obviating the need to recreate a new image each time.

The FITC supports a set of command line parameters that can be used to build an image from the command prompt or from a makefile. A previously stored configuration can be used to define the image layout making interacting with the GUI unnecessary.

Note that the FITC does not program the flash device. FITC only generates a complete SPI image file. This complete SPI image must be programmed into the flash, either using the FPT or another third-party tool.

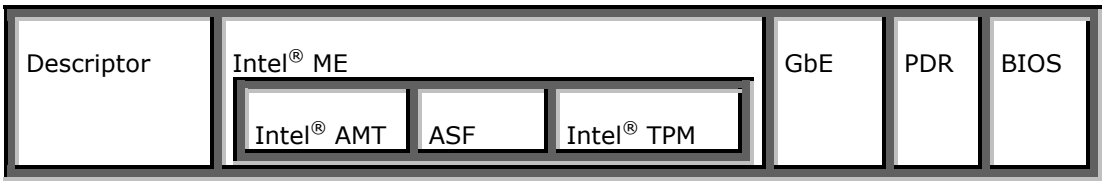
3.1 System Requirements

The FITC will run on Windows* XP or Windows Vista* (32 bit). It is not necessary for the tool 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. Firmware Image Components



The following is a description of these regions:

- Descriptor—takes up a fixed amount of space at the beginning of the flash memory. The descriptor contains information, such as, 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.

Note: This region MUST be locked before the Intel® AMT device is shipped to end users. Please see the 4.11 below for more information. Failure to lock the Descriptor Region will leave the Intel® AMT device vulnerable to security attacks.

- ME—region that takes up a variable amount of space at the end of the descriptor. Contains code and configuration data for Intel® ME applications, such as Intel® AMT technology, ASF 2.0, Intel® TPM.
- GbE—region that takes up a variable amount of space at the end of the ME region. Contains code and configuration data for Gigabit Ethernet.
- BIOS—region that takes up a variable amount of space at the end of flash memory. The BIOS contains code and configuration for the entire computer.
- PDR—Platform Descriptor Region allows system manufactures to 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 no fixed space is assigned, then the region will occupy only as much space as it requires (in 4 Kbyte increments).
2. If there is still space left in the flash after allocating space for all of the regions, the ME region will expand to fill the remaining space.
3. If there is leftover space and the ME region is not implemented, then the BIOS region will expand to use the remaining space.
4. If there is leftover space and the BIOS region is not implemented (an unlikely scenario), then the GbE region will expand to occupy the remaining space.
5. Lastly, if only the Descriptor region is implemented, it will expand to occupy the entire flash.

3.3 Required Files

The FITC main executable is fitc.exe. This program requires that the following files be in the same directory as fitc.exe:

- fitcmplc.xml
- newfiletmpl.xml

The FITC will not run correctly if either of these files are missing.

3.4 Configuration Files

The flash image can be configured in many different ways, depending on the target hardware and the firmware options required. The FITC enables the user to 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.

3.4.1 Creating a new configuration

The FITC provides a default configuration file from which the user can build a new image. This default configuration can be loaded by clicking **File** > **New** from the menu bar.

3.4.2 Opening an existing configuration

To open an existing configuration file:

1. Click **File** on the menu bar.
2. Select **Open**. This will cause the Open File dialog to appear.
3. Select the XML file you want to load
4. Click **Open**.

It is also possible to open a file by dragging and dropping a configuration file onto the main window of the application.

3.4.3 Saving a configuration

To save the current configuration in an XML file:

1. Click **File** on the menu bar.
2. Select **Save**.

—OR—

1. Click **File** on the menu bar.
2. Select **Save As…**. If **Save As…** is selected or if the configuration has not been given a name, the **Save File** dialog will appear.
3. Select the path and file name under which to save the configuration.
4. Click **Save**.

3.5 Environment Variables

To modify the environment variables:

1. Click **Build** on the menu bar.
2. Select **Environment Variables….** A dialog box will appear showing the current working directory on top, followed by the current values of all the environment variables.

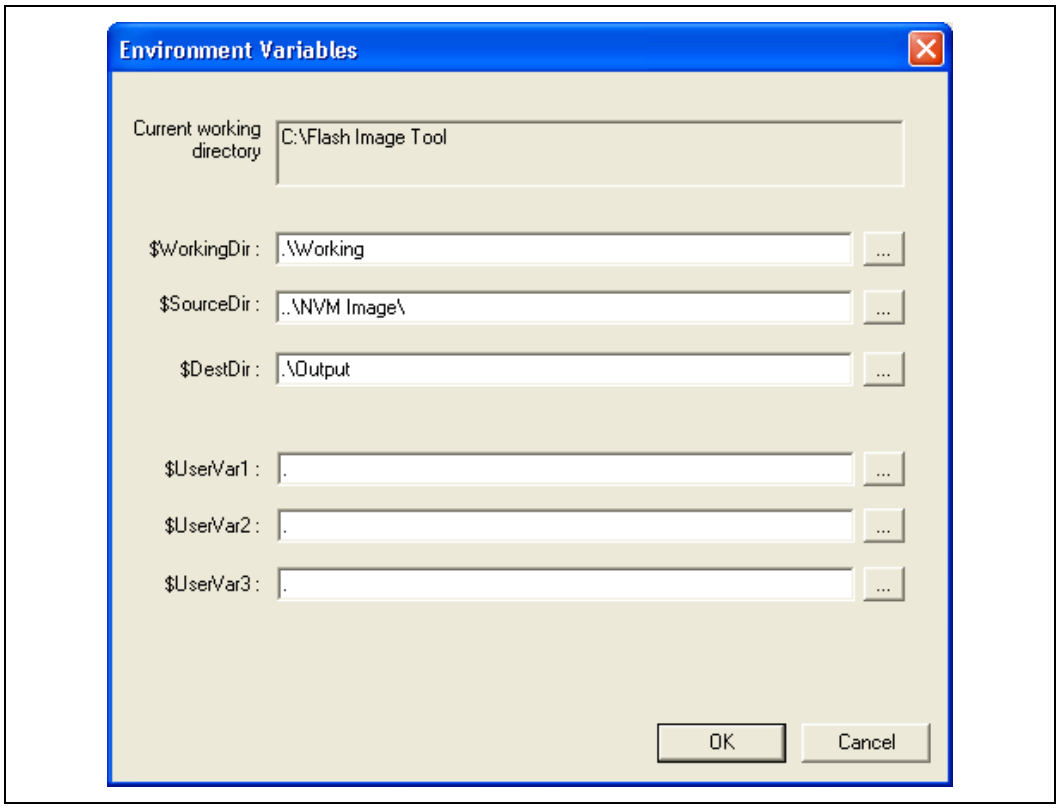
   A set of environment variables are provided to make the image configuration files more portable. By making all of the paths in the configuration relative to environment variables, the configuration is not tied to a particular root directory structure. Each user can set their environment variables appropriate for their computer, or override the variables using command line options.

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

   The variables are:

   - $WorkingDir—the directory where the log file is kept. This is also where the components of an image are stored when an image is decomposed.
   - $SourceDir—the directory that contains base image binary files from which a complete flash image will be prepared. Usually these base image binary files are obtained from Intel® ARMS on the Web, a BIOS programming resource, or other source.
   - $DestDir—the directory in which the final combined image will be saved, including all intermediate files generated during the build.
   - $UserVar1-3 – are used when the above variables are not populated

Figure 2. Environment Variables Dialog



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

3.6 Build Settings

To modify the build setting:

1. Click **Build** on the menu bar.
2. Select **Build Settings…** A dialog box will appear showing the current build settings.

The FITC allows the user to set several options that control how the image is built. The Output path is the path and filename where the final image should be saved after it is built. (Use the $DestDir environment variable to make the configuration more portable.)

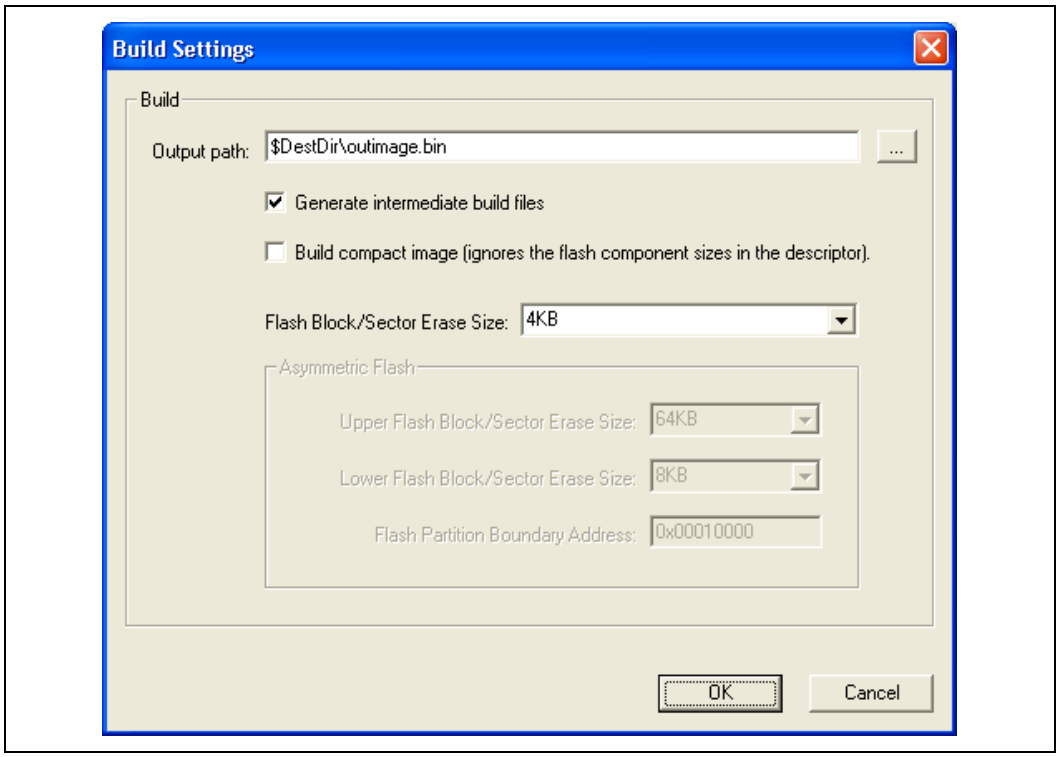
An option is provided (the **Generate intermediate build files** checkbox) that causes the application to generate separate (intermediate) binary files for each region, in addition to the final image file (see Figure 3). These files will be located in the "int" folder located inside the specified output folder. These image files can be programmed individually using the Flash Programming Tool (FPT).

The user can also elect to build a compact image which creates the smallest flash image possible. (By default, the application uses the flash component sizes in the descriptor to determine the image length.)

Finally, the user must select the flash component sector erase size. It is critical that this option is set correctly to ensure that the flash regions can be properly updated at runtime. All regions in the flash conform to the 4Kbyte sector erase size.

The **Asymmetric** option allows the user to specify a different sector erase size for the upper and lower flash block. This option also allows the user to modify the flash partition boundary address.

Figure 3. Build Settings Dialog



Note: The build settings are saved in the XML configuration file.

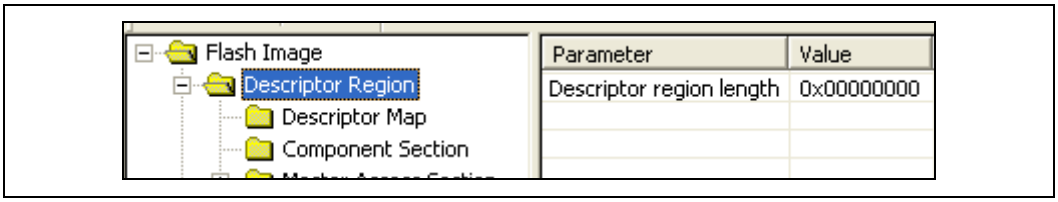
3.7 Modifying the Flash Descriptor Region (FDR)

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

3.7.1 Descriptor Region length

Selecting the Descriptor Region will allow the user to specify the size of the region. If a non-zero value is entered, this value will be used to determine the length of the region.

Figure 4. Descriptor Region length

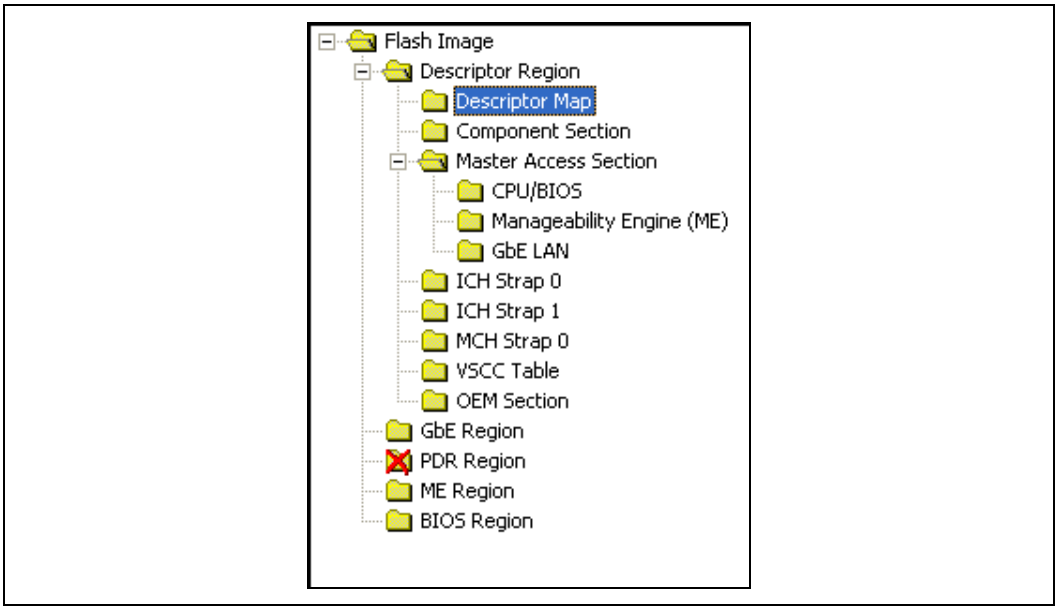


3.7.2 Setting the number and size of the flash components

To set the number of flash components:

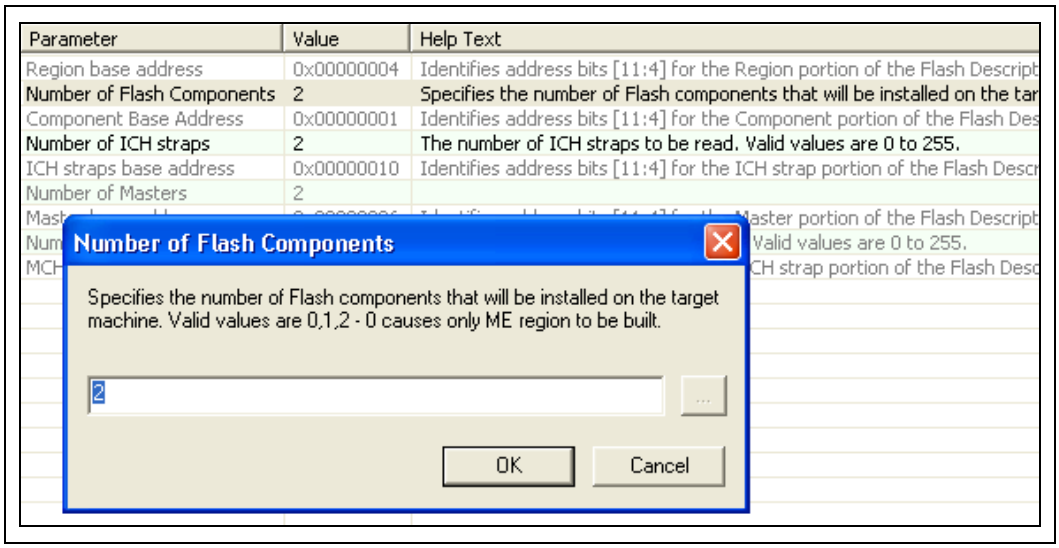
1. Expand the Descriptor Region node of the tree in the left pane of the main window.
2. Select **Descriptor Map** (see Figure 5). All of the parameters for the Descriptor Map section will appear in the list in the right pane of the main window.

Figure 5. Editable Flash Image Region List



3. Double-click the list item named **Number of Flash Components** (see Figure 6). A dialog will appear allowing the user to enter the number of flash components (valid values are 1 or 2).
4. Click **OK** to update the parameter.

Figure 6. Descriptor Region Map Options

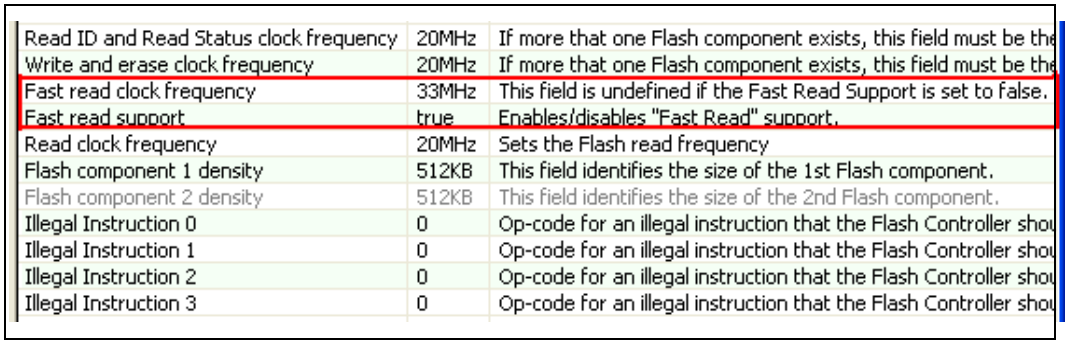


Some SPI flash devices support both standard and fast read speeds. For the ICH10 to support the faster read speeds, the fast read clock frequency must be set to 33 MHz and fast read support must be set to true.

If the system has two SPI devices, the system will not recognize the 2nd SPI device until the 1st SPI device is programmed. For this reason, the SPI flash devices needs to be programmed twice before both SPI devices are recognized. The first time the first device is programmed the image should specify two devices. The first image file should contain the Descriptor region and the BIOS Region only. This can be done using any hex editor. A future release of the flash image tool will support this feature. After the system returns from a G3 state, both SPI devices will be recognized and both will be programmable.

If the SPI devices are programmed using a flash programmer, both devices will be present after the first program.

Figure 7. Descriptor Region Fast Read Support Options

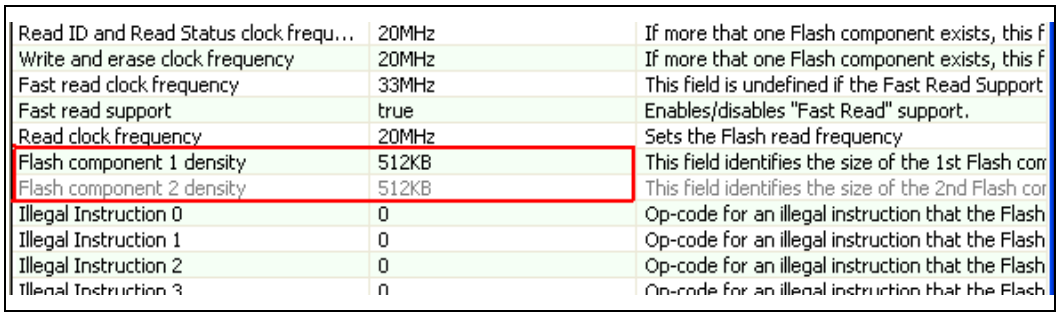


To set the size of each flash component:

1. Expand the Descriptor Region tree node and select the **Component Section** node. The parameters Flash component 1 density and Flash component 2 density specify the size of each flash component.
2. Double-click on each parameter and select the correct component size from the drop-down list.
3. Click **OK** to update the parameters.

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

Figure 8. Descriptor Region Component Section Options



3.7.3 Region access control

Regions of the flash can be protected from read or write access by setting a protection parameter in the Descriptor Region. Before Intel® ME devices are shipped, the Descriptor Region must be locked. 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 2. Region Access Control Table

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

Three parameters in the Descriptor exist to specify access for each chipset. The bit structure of these parameters are shown below.

Key:

0—denied access

1—allowed access

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

CPU /BIOS gets…

Read Access

	Unused			PDR	GbE	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	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

For example, if the CPU/BIOS needs read access to the GbE and ME and write access to ME, then the bits will be set to:

Read Access—0b 0000 1110

Write Access—0b 0000 0110

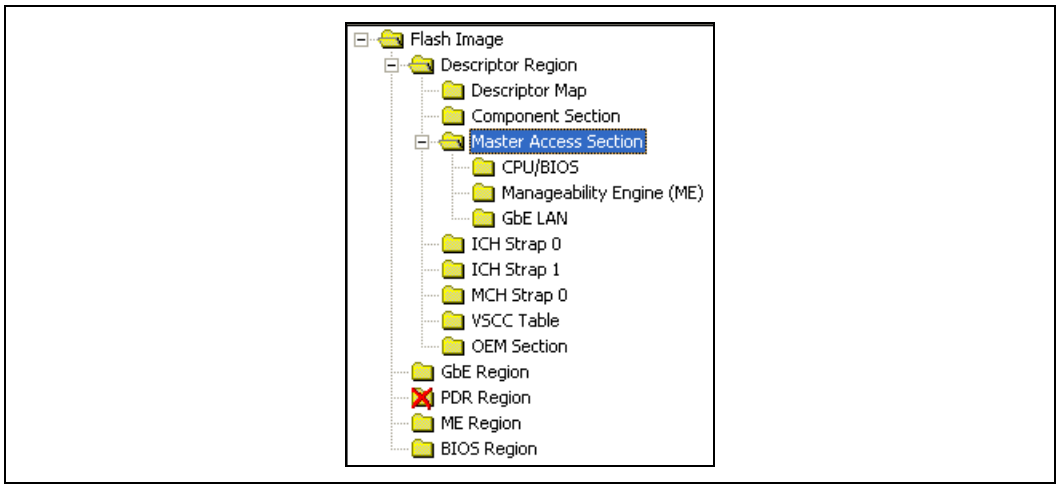
In hexadecimal:

Read Access—0x 0E

Write Access—0x 06

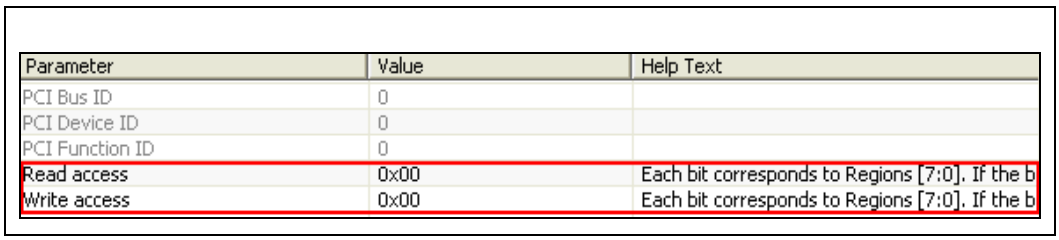
In the FITC these access values can be set by selecting the Descriptor Region tree node and selecting CPU/BIOS under the Master Access Section (see Figure 9).

Figure 9. Descriptor Region Master Access Section Location



The read and write access hexadecimal values can be specified in the appropriate parameters (see Figure 10).

Figure 10. Descriptor Region Master Access Section Options



Parameter	Value	Help Text
PCI Bus ID	0	
PCI Device ID	0	
PCI Function ID	0	
Read access	0x00	Each bit corresponds to Regions [7:0]. If the b
Write access	0x00	Each bit corresponds to Regions [7:0]. If the b

As a reference, the bit layout for ME/MCH and the GbE controller are given below.

ME/MCH gets…

Read Access

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

Write Access

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

GbE Controller gets…

Read Access

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

Write Access

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

The following is the minimum recommended settings for the read/write parameters. This sample will provide the Descriptor Region with an acceptable level of security while still allowing reasonable access to the rest of the regions on the flash device.

Note: The settings below will lock the flash region and prevent any future changes to the flash device.

This includes any changes made via the fixed address mechanism. If using the fixed address mechanism, manufacturers can alternatively lock the descriptor region during manufacturing. Locking the Descriptor Region late in the manufacturing flow allows the manufacturer more flexibility in the programming of the flash device. As stated above, once the region is locked, changes to the flash device will be more difficult.

Table 3. Recommended Read/Write Values

Master Access	PDR Region	GBE Region	ME Region	BIOS Region	Descriptor Region
ME read access	N	Y	Y	N	Y
ME write access	N	Y	Y	N	N
GbE read access	N	Y	N	N	N
GbE write access	N	Y	N	N	N
BIOS read access	Y	Y	N	Y	Y
BIOS write access	Y	Y	N	Y	N

The table below shows the values to be used in the FITC. These values provide the access levels described in the table above.

Table 4. Recommended Read/Write Values

	ME	GbE	BIOS
Read	0b 0000 1101 = 0x0D	0b 0000 1000 = 0x08	0b 0001 1011 = 0x1b If the PDR region is not present the value should be: 0b 0000 1011 = 0x0b
Write	0b 0000 1100 = 0x0C	0b 0000 1000 = 0x08	0b 0001 1010 = 0x1a If the PDR region is not present the value 0b 0000 1010 = 0x0a

3.8 MCH Strap 0

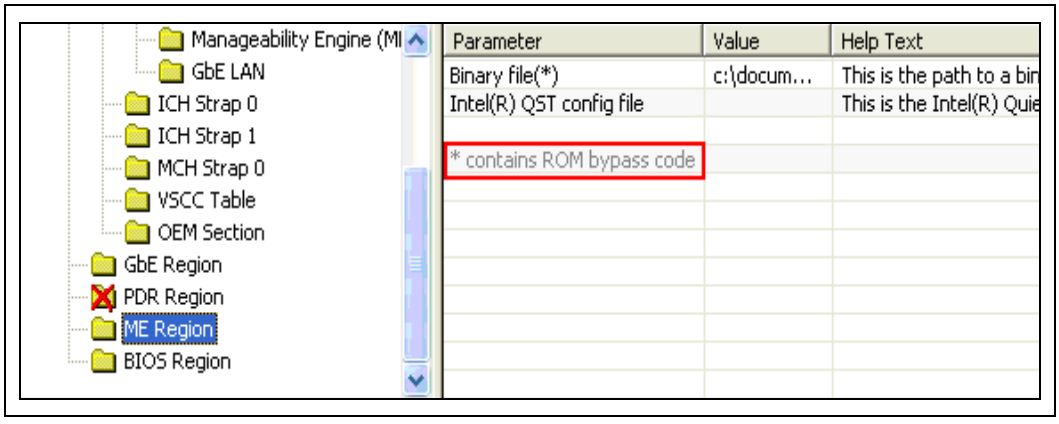
This section contains the variable to edit the address at which the Intel® ME begins to read.

3.8.1 Intel® ME boot from flash

If the firmware used contains the boot from flash option, FITC will automatically select the Intel® ME boot from flash option.

When this option is loaded, a small amount of code is run before the firmware, allowing the user to avoid known hardware or software problems until a permanent solution can be found. If the firmware loaded in the ME Region does not include a ROM bypass section the user will not see the not seen below.

Figure 11. Message Determining Whether Firmware Image Contains ROM Bypass Section



3.9 ICH Strap 0

This section contains variables for configuring LAN specific components and PCI express configuration details.

Integrated GbE or PCI express - When using non-Intel LAN port 6 this value should be set to "PCI Express". By default the values is 'Integrated GBE'

SMBUS Address - The SMBus address should be set to 0x64 in order to ensure that the correct address location is given to the SMBus. Providing an incorrect address will result in the code starting at an incorrect address.

3.10 VSCC Table

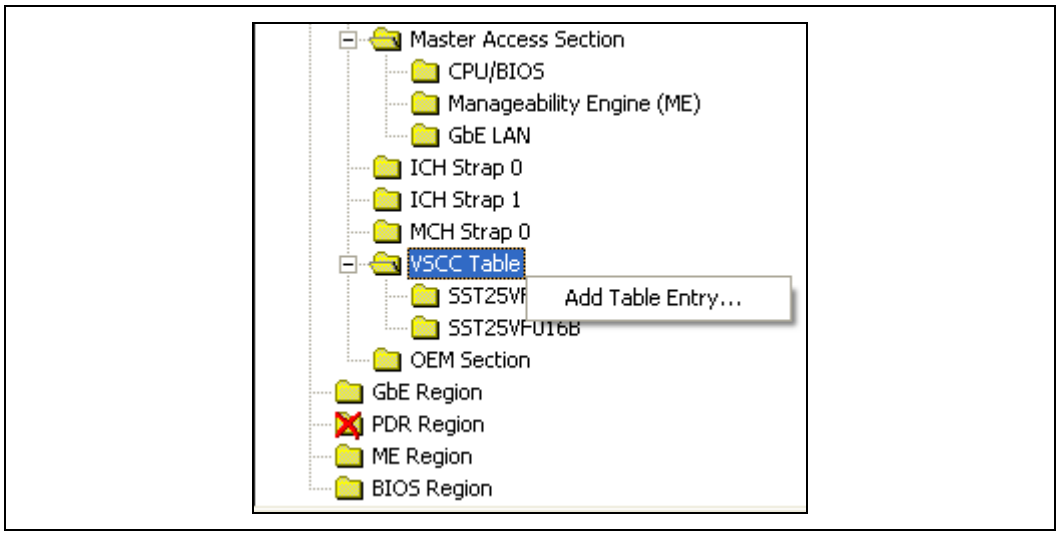
This section is used to store information regarding the flash devices used on the system and is **REQUIRED** by the Intel® ME firmware and BIOS. If the information in this section is incorrect, the Intel® ME will not communicate with the flash device. The information provided here is specific to the flash device used on the system. Please contact your flash vendor for this information.

3.10.1 Adding a new table

To add a new table:

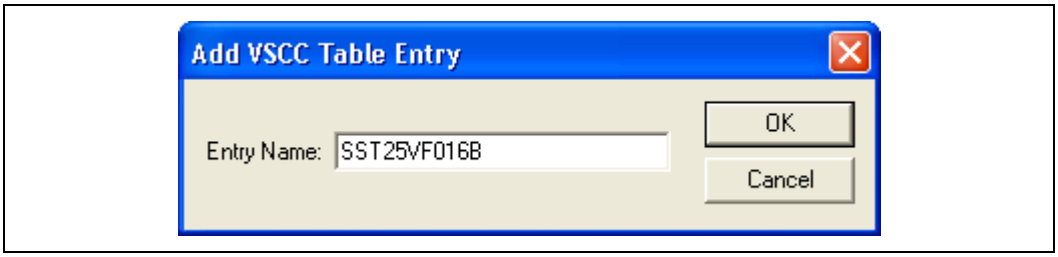
1. Right-click on **VSCC table**.
2. Select **Add Table Entry…**

Figure 12. Add New VSCC table entry



The program will then prompt the user for a table entry name. To avoid confusion it is recommended that each table entry be unique. There is no checking mechanism in FITC to prevent table entries that have the same name and no error message will be displayed in such cases.

Figure 13. Add VSCC table entry



After a table entry has been added, the user will be able to fill in values for the flash device. The values in the VSCC table are provided by your flash vendor. Users should contact their flash vendor for the specific values mentioned in this table. For Intel® Blanchard flash devices, the values would be as follows:

Vendor ID—0x89

Device ID 0—0x89

Device ID 1—0x11

VSCC Register Value—0xD81ED81F

The screenshot below shows the values for the flash part SST25vf016b.

Figure 14. VSCC Table Entry

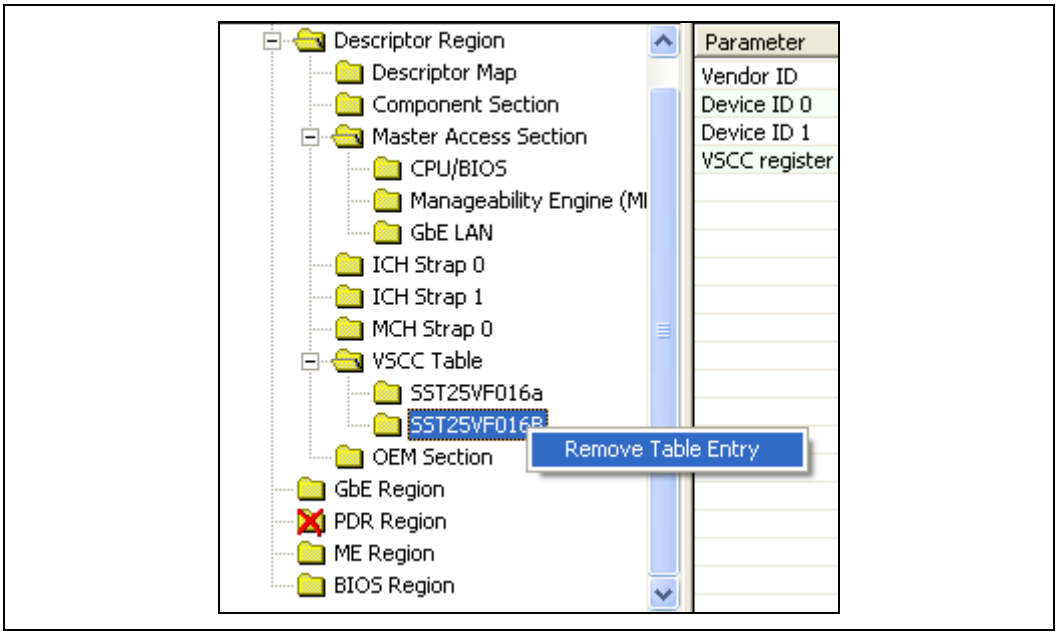
Parameter	Value	Help Text
Vendor ID	0xBF	The vendor specific byte of the JEDEC ID.
Device ID 0	0x25	The first device specific byte of the JEDEC ID.
Device ID 1	0x41	The second device specific byte of the JEDEC ID.
VSCC register value	0x00002009	The device specific VSCC register value.

3.10.2 Removing an existing table

To remove an existing table:

1. Right-click on the table that needs to be removed.
2. Select **Remove Table Entry**. All information in the table along with the table entry will be removed.

Figure 15. Remove VSCC table entry



3.11 Modifying the Intel® ME Region

The Intel® ME Region contains all of the firmware and data for the Intel® ME (which includes the kernel and Intel® AMT).

3.11.1 Setting the Intel® ME Region binary file

To set the ME region binary file:

1. Select the ME Region tree node.
2. Double-click on the **Binary file parameter** in the list. A dialog box will appear allowing the user to specify the ME file to use.
3. Click **OK** to update the parameter.

When the flash image is built, the contents of this file will be copied into the ME Region.

The ME Region length option should not be altered. A value of 0x00000000 indicates that the ME Region will be auto-sized as described in Section 4.2, Flash space allocation.

If the user has specified in the MCH Strap 0 Section 3.8 that the ME must boot from flash, the firmware loaded must contain a ROM Bypass section. If the firmware does not contain a ROM bypass section, a section will become available in which to enter the location of the ROM bypass file.

3.11.2 Enabling/disabling the Intel® ME Region

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

To disable the ME Region:

1. Right-click on the ME Region tree node.
2. Select **Disable Region** from the pop-up menu.

The user will then need to increase the size in one of the other regions. FITC will "pad" the remaining space. For example, if the user wants to disable the ME Region and "pad" the GbE Region he would subtract the size of the BIOS Region, PDR Region (if a PDR Region is included) and the Descriptor Region from the full SPI image size. This will determine the new size of the GbE Region.

3.11.2.1 Example 1

The example below assumes a symmetric 4kb flash with a 1 KB BIOS with no PDR Region.

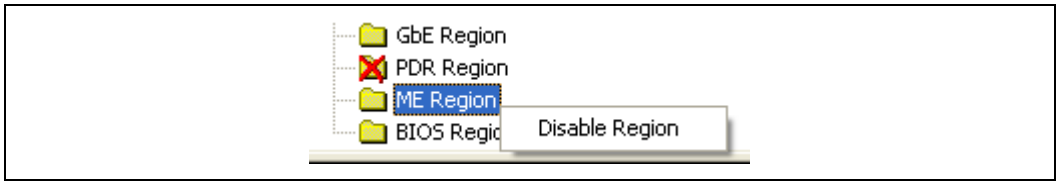
Full SPI Image size – BIOS Region size – Descriptor Region size = GBE Region Size

0x400000 – 0x100000 – 0x1000 = 0x2ff000

The GbE Region size value should be entered for the GbE LAN Region length in the GbE section. "Padding" the BIOS Region is not recommended.

The ME Region can be enabled by right-clicking on the ME Region tree node and selecting **Enable Region** from the pop-up menu.

Figure 16: Enabling the Intel® ME Region



3.12 Modifying the GbE (LAN) Region

The GbE Region contains various configuration parameters (such as, the MAC address) for the embedded Ethernet controller.

3.12.1 Setting the GbE Region binary file

To set the GbE Region binary file:

1. Select the GbE Region tree node.
2. Double-click on the **Binary input file parameter** from the list. A dialog box will appear allowing the user to specify which GbE file to use. Select a file.
3. Click **OK** to update the parameter.

When the flash image is built, the contents of this file will be copied into the GbE Region.

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.

Figure 17: GbE Region Options

GbE LAN region length	0x00000000	This is the size of the ME region in bytes. Set this to 0 to make the region leng...
Binary input file		This is the Gbe image binary that will be copied into this region.
MAC address	00 00 00 00 00 00	This is the 48-bit Ethernet MAC.
Major Version	0	
Minor Version	0	
Image ID	0	

This is the location where the user can modify the Ethernet MAC address.

To configure the Ethernet MAC address:

1. Double-click the MAC address parameter from the list. A dialog box will appear allowing the user to specify the Ethernet MAC address.
2. Enter the required value.
3. Click **OK** to update the parameter.

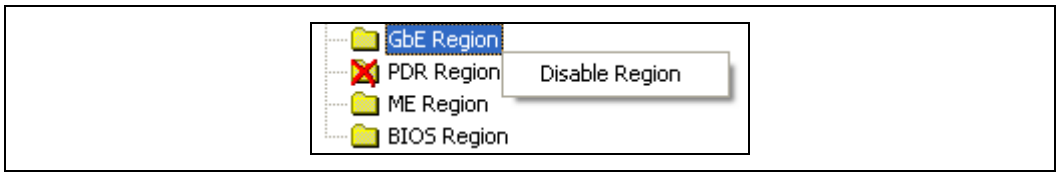
3.12.2 Enabling/disabling the GbE Region

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 the GbE Region tree node.
2. Select **Disable Region** from the pop-up menu. When the flash image is built it will not contain a GbE Region.

Figure 18: Disabling the GbE Region



To enable the GbE Region:

1. Right-click on the GbE Region tree node.
2. Select **Enable Region** from the pop-up menu.

3.13 Modifying the PDR Region

The PDR Region contains various configuration parameters that allow for the customization of the computer's behavior.

3.13.1 Setting the PDR Region binary file

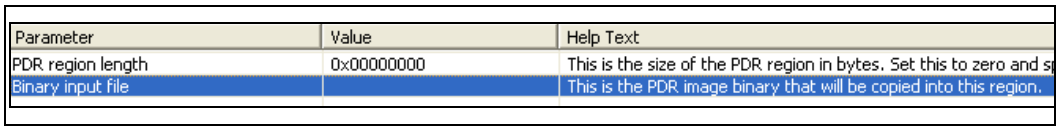
To set the PDR region binary file:

1. Select the PDR Region tree node.
2. Double-click the **Binary input file parameter** from the list. A dialog box will appear allowing the user to specify the PDR file to use.
3. Click **OK** to update the parameter. When the flash image is built, the contents of this file will be copied into the BIOS region.

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.

Note: If the system supports VA, the PDR region length must be set to 0x8000.

Figure 19: PDR Region Options



3.13.2 Enabling/disabling the PDR Region

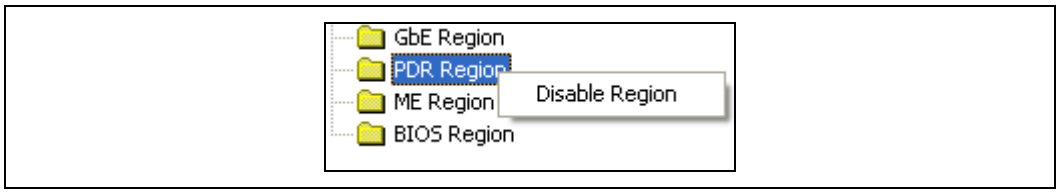
The PDR Region can be excluded from the flash image by disabling it in the FITC.

To disable the PDR Region:

1. Right-click on the PDR Region tree node.
2. Select **Disable Region** from the pop-up menu. When the flash image is built, there will be no PDR Region in it.

By default this region is disabled.

Figure 20. Disabling the PDR Region



To enable the PDR Region:

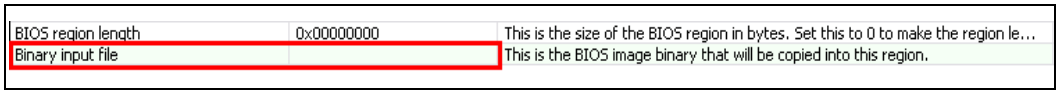
1. Right-click on the PDR Region tree node.
2. Select **Enable Region** from the pop-up menu.

3.14 Modifying the BIOS Region

The BIOS Region contains the BIOS code run by the host processor. The FITC always aligns this region with the end of the flash image. This is done so that in the event that the flash descriptor becomes corrupt for any reason, the ICH will default to legacy mode and look 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 will be 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 will be padded at the beginning instead of at the end.

3.14.1 Setting the BIOS Region binary file

Figure 21. BIOS Region Options



To set the BIOS region binary file:

1. Select the BIOS Region tree node.
2. Double-click the **Binary input file parameter** from the list. A dialog box will appear allowing the user to specify the BIOS file to use.
3. Click **OK** to update the parameter. When the flash image is built, the contents of this file will be copied into the BIOS region.

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

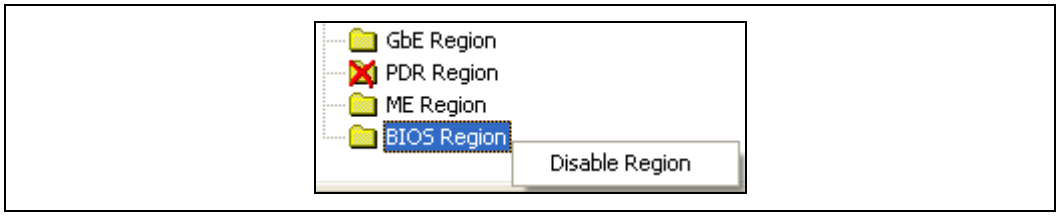
3.14.2 Enabling/disabling the BIOS Region

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

To disable the BIOS Region:

1. Right-click on the BIOS Region tree node.
2. Select **Disable Region** from the pop-up menu. When the flash image is built, there will be no BIOS Region in it.

Figure 22. Disabling the BIOS Region



To enable the PDR Region:

1. Right-click on the BIOS Region tree node.
2. Select **Enable Region** from the pop-up menu.

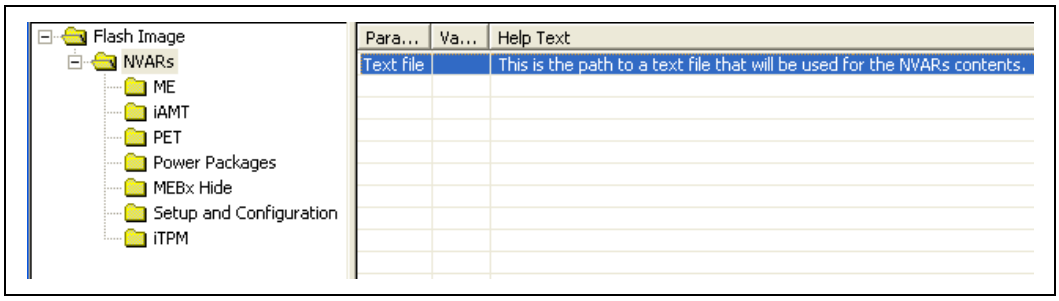
3.15 NVARs Tab (Modify Default ME Parameters)

The NVARs tab located at the bottom of the window allows the user to set specific parameters. This option replaces AMTNVM from previous generations.

If any of the parameters are changed from the Intel recommended value the offending row will be highlighted yellow. No errors will be reported. The highlighted yellow is designed to draw attention to these values were ensure these parameters were set correctly.

The first value will allow the user to set the NVARs text file. The NVARS text file contains the values of all the parameters set in the NVARS region. This text file can be used in the command line argument to modify the default ME parameters using the "/nvars" option.

Figure 23. NVARs Tab

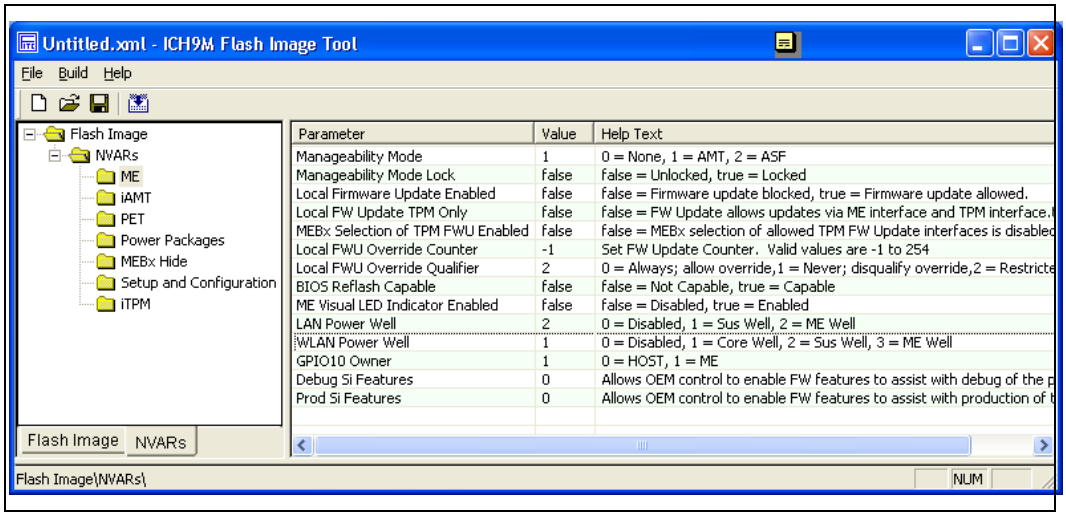


3.15.1 Intel® ME Section

The Intel® ME section allows the user to specify the computer's manageability features. The parameters values are can be found in the Help Text alongside to the parameter value as shown in Figure 24.

Note: The ME LED bit is reserved and default settings should be maintained

Figure 24. ME Section



WLAN Power Well – This value determines how the wireless LAN is powered. If this setting is incorrect AMT will not function correctly through the WLAN. The settings for this parameter are explained below

0 (None) – If there is no plans to add a WLAN device on the system the WLAN Power Well value should be set to 0 (zero).

1 (Core Well) – States that the WLAN device can only communicate with the ME when the Core well is powered.
Note – This is an invalid value for Intel® AMT. Intel® AMT will not work through the WLAN with this setting

2 (SUS well) - States that the WLAN device can only communicate with the ME when the SUS well is powered.
Note – This setting supports ME Wake on LAN through the WLAN.

3 (ME Well) – States that the WLAN device can only communicate with the ME when the ME well is powered.
Note – This setting does NOT support ME Wake on LAN through the WLAN.

Local FW Update TPM Only – Determines the interface that firmware updates are performed. If this value is set to false firmware updates will be allowed through both the TPM interface and the MEI interface. If the value is set to true, firmware updates will only be allowed through the TPM interface

False – Firmware updates allowed through TPM and MEIN interface. (Default value)

True – Firmware update only allowed through the TPM interface

MEBX Selection of TPM FW Enabled – Determines if the TPM update interface is visible. If the option is visible, the user will be able to modify the value. A value of False will not display the option to enable/disable Firmware updates through the TPM interface. A value of True will display the firmware update via TPM interface.

False – Firmware update via TPM interface parameter is NOT displayed in the MEBX. (Default value)

True – Firmware update via TPM interface parameter is displayed and can be modified.

GPIO10 – Determines the owner of GPIO10. If the system is an Intel® TPM only SKU, this value should set to 1 (one). If the system uses GPIO10 for another purpose the owner of GPIO10 should be set to 0 (zero). If this parameter was not previously set, the value will remain at the default value of 1 (one)

1 (one) – ME is the owner of GPIO10. (Default value)

0 (zero) – Host is the owner of GPIO10

Prod Si Features – Enables or Disables ME Debug Display Device (MDDD). MDDD will not display post code information if this value is not set correctly. This value should be set to 0 in the final production image.

1 (one) – MDDD displays post code information through the device

0 (zero) – MDDD will NOT display post code information through the device

3.15.1.1 Temporary firmware update parameters

If the Local FWU Override Counter has a value between 1 and 255, firmware updates are allowed even if updates are disabled in the Intel® Management Engine BIOS Extension (Intel® MEBX) settings. After the flash is programmed, each time the computer restarts it causes the Local FWU Override Counter to be decremented. When the Local FWU Override Counter reaches 0, firmware updates are no longer allowed if they are not enabled in the Intel® MEBX settings.

Note: The restart that takes place after the flash memory has been programmed also causes the Local FWU Override Counter to be decremented. Therefore, if it is necessary to enable updating the firmware N times, you need to assign the Local FWU Override Counter the initial value N+1.

If the Local FWU Override Counter is set to -1 and the Local Firmware Override Qualifier is set to 0, firmware updates are always allowed regardless of the settings in the Intel® MEBX.

The following table shows the possible value combinations for the two variables. To enable local firmware updates, make sure both variables are assigned the correct values.

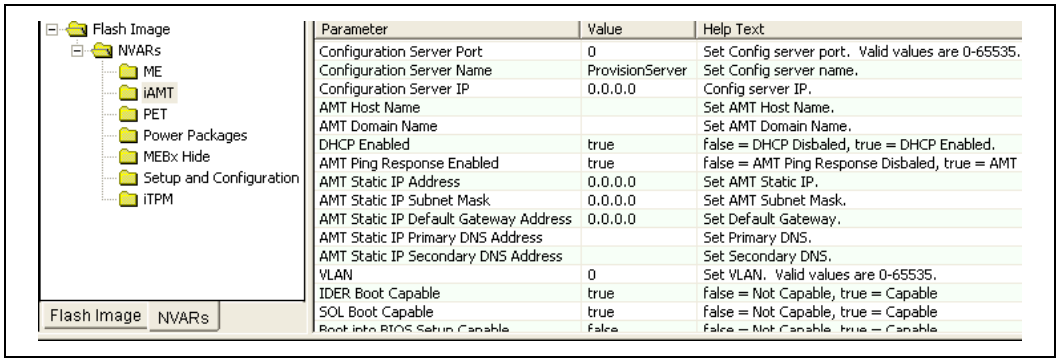
Table 5. Firmware Override Update Variables

	Local FWU Override Qualifier = 0 (zero)	Local FWU Override Qualifier = 1 (one)	Local FWU Override Qualifier = 2 (two)
Local FWU Override counter = 0 (zero)	Local Firmware Updates NOT Allowed	Local Firmware Updates NOT Allowed	Local Firmware Updates NOT Allowed
Local FWU Override Counter = -1 (minus one)	Local Firmware Updates Allowed	Local Firmware Updates NOT Allowed	Local Firmware Updates Allowed only until ME is configured
Local FWU Override Counter = 0<n<255	Local Firmware Updates Allowed	Local Firmware Updates Allowed	Local Firmware Updates Allowed

3.15.2 Intel® AMT Section

The Intel® AMT section allows the user to specify the default Intel® AMT parameters. The values specified in this section will be used after the Intel® AMT device is un-provisioned (full or partial).

Figure 25. Intel® AMT Section



Be careful when setting these parameters as some of them cannot be modified by the end user, such as the Boot into BIOS setup Capable.

Configuration Server Port – Specifies the port number that will be used for remote configuration. Default value is 9971.

Configuration Server Name – Specifies the FQDN of the configuration server. The Intel® AMT system will look for the specified FQDN when attempting remote configuration. Default value is "ProvisionServer".

IDE-R Boot Capable – Determines if the system is capable of opening an IDE-R session. If this is set to false, the system can **NOT** open an IDE-R session. This parameter can **NOT** be modified in the Intel® MEBX. Default value is True.

SOL Boot Capable – Determines if the system is capable of opening an SOL session. If this is set to false, the system can **NOT** open an SOL session. This parameter can **NOT** be modified in the Intel® MEBX. Default value is True.

HostIF IDER Enabled – Determines if IDE-R sessions are permitted. Before an IDE-R session can be opened, this option must be set to true. This parameter can be modified in the Intel® MEBX. Default value is True.

HostIF SOL Enabled – Determines if SOL sessions are permitted. Before an SOL session can be opened, this option must be set to true. This parameter can be modified in the Intel® MEBX. Default value is True.

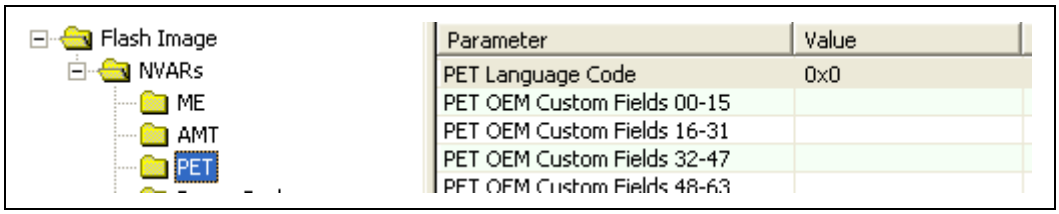
Idle Timeout -Specifies the amount of time (in minutes) before the system goes into an M-off state if ME-WOL is enabled. This value can be modified by the end user in the Intel® MEBX. To reduce the amount of end user configuration time, this value should be set to a reasonable value. Default value is 1 minute.

3.15.3 PET Section

The PET section allows the OEM to specify custom field data. If no value is specified for the "PET Community String" AMT will use the default value of "public".

For more information on these values please see Platform Event Trap Format Specification document publicly available online.

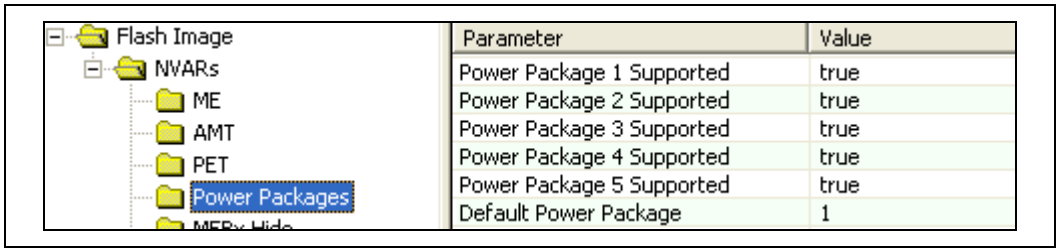
Figure 26. PET Section



3.15.4 Power Packages Section

The Power Packages section allows the OEM/ODM to specify which power packages are supported.

Figure 27. Power Packages Section



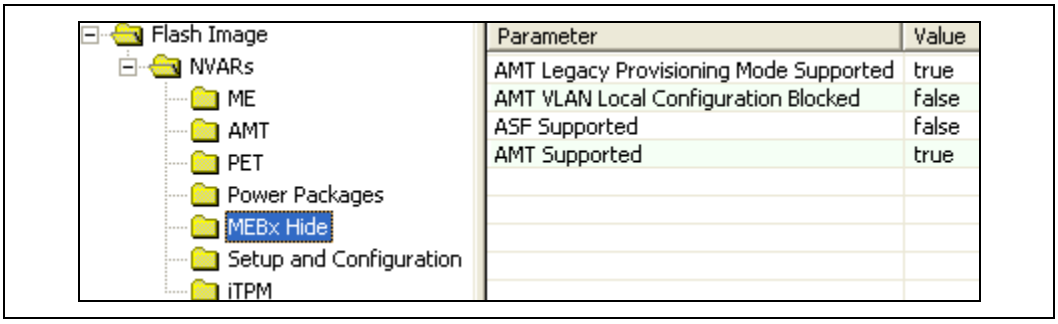
If the Power Package Supported value is set to false, that specific power package cannot be selected and will not be visible to the end user.

The Default Power Package selected must be supported. This is the value that will be selected when the system is shipped. This value will affect energy star compliance if not set correctly.

3.15.5 Intel® MEBX Hide Section

The Intel® MEBX Hide section allows the OEM/ODM to specify which fields will be visible in the Intel® MEBX.

Figure 28. Intel® MEBX Hide Section

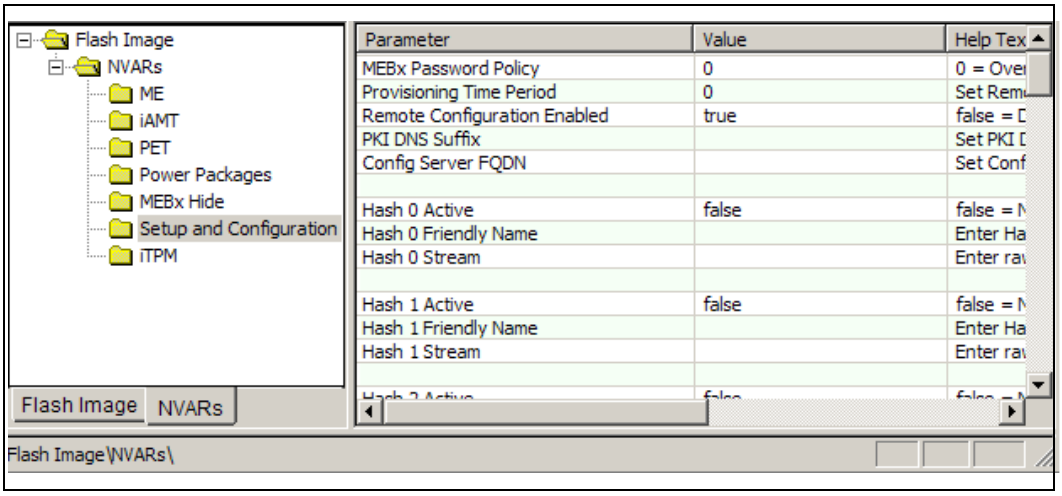


Any parameter that is set to false will not be visible to the end user in the Intel® MEBX. If the value is not visible to the end user, the value cannot be modified by the end user. Please be certain that the values in the Intel® AMT and Intel® ME sections are set correctly.

3.15.6 Setup and Configuration Section

The Setup and Configuration section allows the end user to specify the configuration settings. These values determine the mode of the Intel® AMT device after the system has been configured.

Figure 29. Setup and Configuration Section



Provisioning Time Period - Specifies the amount of time (in hours) allowed to configure the Intel® AMT device. This time period begins when the Intel® ME starts to

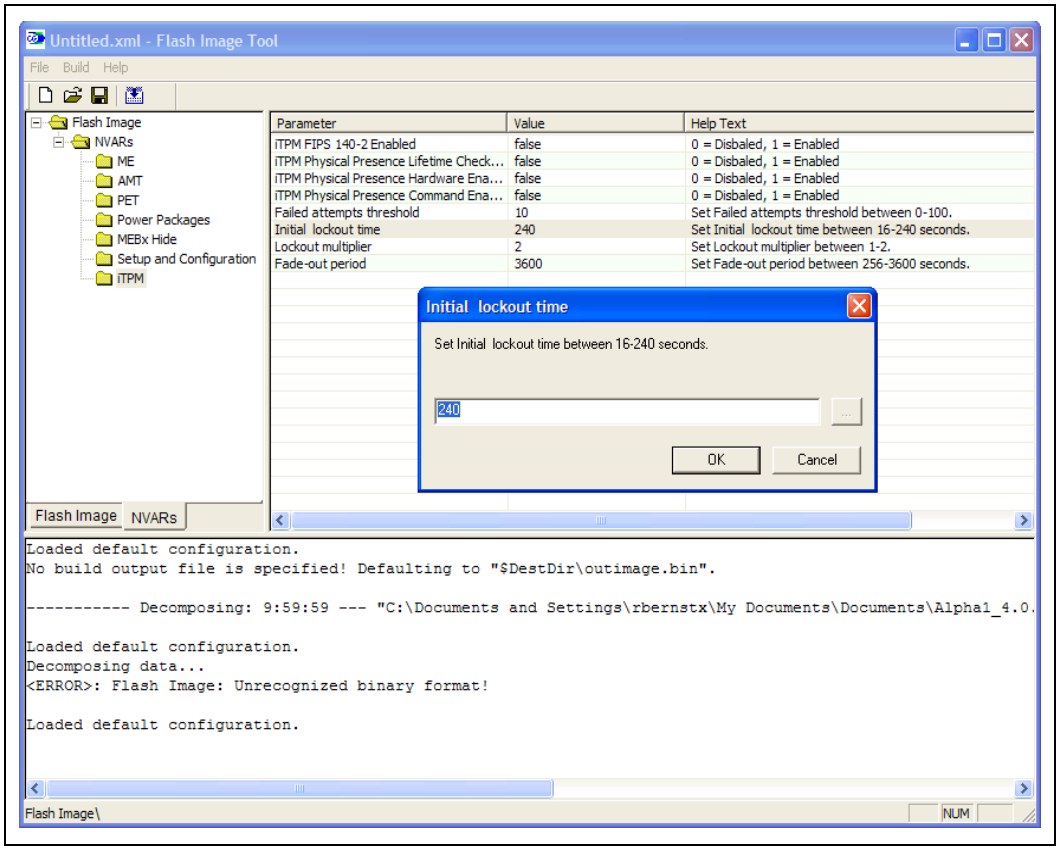
run on the system and stops when the Intel® AMT system is provisioned. The provisioning time period will continue to decrement if the ME is running on the system. Once this period of time has elapsed, the system will enter a timed mode. During this timed mode the system only has one hour (per boot) to be configured by the management console.

Remote Configuration Enabled – Specifies if remote configuration is enabled. If the value is set to *True* and the Provisioning time period is not 0, The Intel® AMT system will send out directed packets across the network. Default value is True.

The Hashes specified allow the system to be remotely configured. At least one hash must be active and Remote Configuration Enabled must be set to true to allow remote configuration. The Hash Certificates entered through FITC will be set as default. These Hash certificates will be preserved after a full un-provision.

3.15.7 iTPM Section

Figure 30. iTPM Section



The following is a list of the Intel® TPM parameters along with their defaults that can be modified.

Table 6. Intel® TPM Permanent Flags

Bit	Flag	Description	Default
0	FIPS	TRUE: This TPM operates in FIPS mode FALSE: This TPM does NOT operate in FIPS mode	FALSE
1	Physical Presence Lifetime Lock	FALSE: The state of either physicalPresenceHWEnable or physicalPresenceCMDEnable MAY be changed TRUE: The state of either physicalPresenceHWEnable or physicalPresenceCMDEnable MUST NOT be changed for the life of the Intel® TPM	FALSE
2	Physical Presence HW Enable	FALSE: Disable the hardware signal indicating physical presence TRUE: Enables the hardware signal indicating physical presence	FALSE
3	Physical Presence CMD Enable	FALSE: Disable the command indicating physical presence TRUE: Enables the command indicating physical presence	TRUE

Table 7. Dictionary Attack Flags

Bit	Field	Description	Default	Min	Max
0	Auth Fail Threshold	Number of failed auth attempts which will trigger lockout	10	1	100
1	Initial Lockout Time	Duration in seconds of first lockout period	240	16	0xFFFF
2	Lockout Increase Factor	Factor by which lockout period is multiplied with every additional failed auth	2	1	0xFFFF

Bit	Field	Description	Default	Min	Max
3	Fade Out Time	Every time this period (in seconds) passes with no auth failures, lockout time will be reduced (divided by Lockout Increase Factor)	3600	256	0xFFFF

3.16 Building a Flash Image

The flash image can be built using the FITC GUI interface.

To build a flash image using the currently loaded configuration:

1. Click **Build** on the menu bar.
2. Select **Build Image**.

—OR—

3. Specify an XML file with the /b option on the command line.

The FITC uses an XML configuration file and the corresponding binary files to build a Montevina flash image. The following will be produced when building an image:

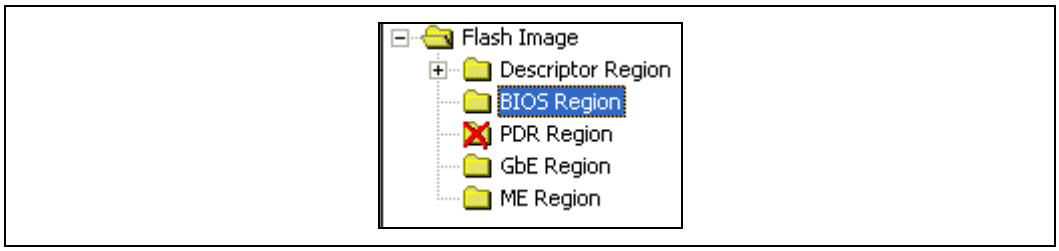
- Binary file representing the image
- Text file detailing the various regions in the image
- Optional set of intermediate files (see Section 3.6).
- And, if two flash components are specified, multiple binary files containing the image broken up according to the flash component sizes.

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

3.17 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 region as they are displayed in left hand pane of the FITC window.

Figure 31. Region Order



Each region will be added to the full SPI image in the order in which they appear in the list. In Figure 32: Region Order, the Descriptor Region will be the first region in the full image, followed by the BIOS Region. The Intel® ME Region will be the last to be added to the full SPI image file.

This can be useful when programming a system with two SPI devices. It is possible to change the order of the regions by clicking and dragging the region to the required location. Figure 24 shows that the BIOS will be placed on the first SPI device and the ME Region will be placed on the second SPI device. The length of each region and the order will determine if that region will be on the first or second SPI device.

3.18 Decomposing an Existing Flash Image

The 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 just as with any other configuration (see the following sections). A new image can be built from this configuration that is almost identical to the original apart for the changes made by the user.

To decompose an image:

1. Click **File** on the menu bar.
2. Select **Open… ,** change the file type filter to the appropriate file type.
3. Select the required file and click **Open**. The image will automatically be decomposed and the GUI updated to reflect the new configuration.

Alternatively, it is possible to decompose an image by simply dragging and dropping the file onto the main window.

A folder will be created with each of the regions in a separate binary file.

3.19 Command Line Interface

The FITC supports command line options. To view all of the supported options, run the application with the /? option. The command line syntax for the FITC is:

```
fitc    [<xml_file>]
        [<BIN File>]
                [/?]
                [/b]
                [/o <file>]
                [/me <file>]
                [/meoffset <value>]
                [/gbe <file>]
                [/bios <file>]
                [/pdr <file>]
                [/nvars <file>]
                [/fpba <address>]
                [/fpba_or <number>]
                [/ubs <value>]
                [/lbs <value>]
                [/w <path>]
                [/s <path>]
                [/d <path>]
                [/u1 <value>]
                [/u2 <value>]
                [/u3 <value>]
                [/i <enable|disable>]
                [/flashcount <1|2>]
                [/flashsize1 <0|1|2|3|4|5>]
                [/flashsize2 <0|1|2|3|4|5>]
```

<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 will be built automatically.

<Bin File>—decomposes the BIN file. The individual regions will be separated and placed in a folder with the same name as the BIN file name.

/?—displays the command line options.

/b—automatically builds the flash image. The GUI will not be shown if this flag is specified. This option causes the program to run in auto-build mode. If there is an error, a valid message will be displayed and the image will not be built.

If a bin file is included in the command line, this option will decompose the bin file.

/o <file>—path and filename where the image will be saved. This command overrides the output file path in the XML file.

/me <file>—overrides the binary source file for the ME Region with the specified binary file.

/me_offset <value>—overrides the offset of the ME region.

/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.

/nvars <file>—overrides the NVARS file with the file specified

/pdr <file>—overrides the binary source file for the PDR Region with the specified binary file.

/fpba <address>—overrides the flash partition boundary address.

/ubs <value>—overrides the upper block size.

/lbs <value>—overrides the lower block size.

/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 ME Region will be built.

/flashsize1 <0, 1, 2, 3, 4 or 5>—overrides the size of the first flash component with the size of the option selected as follows:

- 0 = 512KB
- 1 = 1MB
- 2 = 2MB
- 3 = 4MB

- 4 = 8MB
- 5 = 16MB.

/flashsize2 <0, 1, 2, 3, 4 or 5>—overrides the size of the second flash component with the size of the option selected as follows:

- 0 = 512KB
- 1 = 1MB
- 2 = 2MB
- 3 = 4MB
- 4 = 8MB
- 5 = 16MB.

3.20 Examples – Decomposing an Image and Extracting parameters

The NVARS variables and the current value parameters can be seen by dragging and dropping the 4mb image. The current parameter value will be displayed.

The parameters can also be extracted using the command line method by the following:

Fitc.exe output.bin /b

The above command will create a folder labeled "output". The folder will contain the individual regions (Descriptor, GBE, ME, BIOS), Map file (labeled <FILENAME>_MAP.txt and NVARS.txt file.

The NVARS.txt file will contain the current ME parameters.

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

§

4 Flash Programming Tool (FPT)

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

Each region can be programmed individually or all of the regions can be programmed in a single command. The user can perform various functions on the contents of the flash, such as:

- View the contents on the screen.
- Write the contents to a log file.
- Perform a binary file to flash comparison.
- Write to a specific address block.
- Program fixed offset variables

4.1 System Requirements

The DOS version of the FPT fpt.exe will run on MS DOS* 6.22, DRMKDOS* and FreeDOS*.

The Windows version fptw.exe will run on Windows* XP (Sp2), Windows* PE and Windows Vista* (32-bit).

The FPT requires an operating system to run on and is designed to deliver a custom image to a computer that is already able to boot, instead of a means to get a blank system up and running. The FPT must be run on the system with the flash memory that the user is programming.

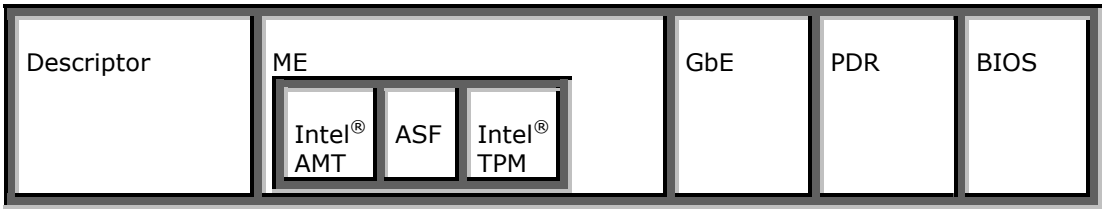
One possible flow for using the FPT is:

1. Pre-programmed flash with legacy or generic BIOS image is plugged into a new computer.
2. Computer boots.
3. The FPT is run and a custom BIOS/ME/GbE image is written to flash.
4. Computer powers down.
5. Computer powers up, boots, and is able to access its ME/GbE capabilities as well as any new custom BIOS features.

4.2 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 overall layout of the flash memory.

Figure 32: Firmware Image Components



Descriptor—takes up a fixed amount of space at the beginning of flash memory. The descriptor contains information, such as:

- Space allocated for each region of the flash image.
- Read/write permissions for each region.
- A space which can be used for vendor-specific data.

Intel® ME—optional region that takes up a variable amount of space at the end of the descriptor. Contains code and configuration data for ME functions, such as, Intel® AMT and Intel® TPM.

GbE—optional region that takes up a variable amount of space at the end of the ME Region. Contains code and configuration data for GbE.

BIOS—region that takes up a variable amount of space at the end of flash memory. The BIOS contains code and configuration for the entire platform.

PDR—Platform Descriptor Region that allows system manufacturers to define custom features for the platform.

4.3 Windows* Required Files

The Windows version of the FPT executable is called 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 file is supplied already populated with default values for SPI devices used with Intel Reference Boards (CRBs).
- fptw.exe—the executable used to program the final image file into the flash.

- sseIdrvdll32e.dll—supported library file.
- ssePmxdll32e.dll—supported library file.
- ssepmxdrv.sys—supported system file.

4.4 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.
- dos4gw.exe (DOS version only)
- 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 file is supplied already populated with default values for SPI devices used with Intel Reference Boards (CRBs).

4.5 Where to find dos4gw.exe

This file is only needed for the DOS version.

Due to licensing issues, dos4gw.exe is not distributed in the same package as the FPT, but may be downloaded from:

http://www.scene.org/file.php?file=%2Fresources%2Fdos4gw.exe&fileinfo

Download the dos4gw.exe file to the same directory where fpt.exe is located.

4.6 Programming the Flash Device

Once the Intel® ME has been programmed it will be running at all times. The 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.

Note: It is important to note that programming the flash device while the ME is running may cause the flash device to become corrupted. The ME should be disabled before programming the full flash device.

To disable the Intel® ME use one of the following options:

1. Disable the ME via the BIOS or the Intel® MEBX.
2. Pull down gpio33 (manufacturing mode jumper) while powering on the system. If the parameters are configured to ignore this jumper, this will not be a valid method of disabling the ME.
3. Remove the memory from Channel 0—this method will cause the ME to boot up in an error state and the error will be written to the flash device. Programming the flash device should be done only after the OS has fully booted.
4. Set the ME disable bits in the strap sections of the descriptor region. Refer to the ICH EDS (sections 24.2.5.1 and 24.2.5.2) for more information.

The ME does **not** need to be disabled when writing to the fixed offset region.

4.7 Programming fixed offset variables

FPT can program the fixed offset variables. FPT will change the default values of the parameters. The modified parameters will be used by the ME firmware after a Global reset or upon returning from a G3 state. The fixed offset variables can be continuously changed until the ***globallocked*** bit is set to 0x0. After this bit is set the parameters can **NOT** be modified. To modify the default settings for the parameters, the entire flash device needs to be re-programmed. The variables can be modified individually or all at once via a text file.

Fpt.exe –fovs will display a list of the variables supported.

Fpt.exe –ex –out <Text File> will create a text file that will allow the user to update multiple variables.

Fpt.exe –u –in <Text file> will update the fixed offset variables with the values as they are entered in the text file.

A list of all of the parameters and their description can be found in the Appendix

4.8 Usage

Note: To prevent possible firmware corruption, the user should disable the firmware before programming any SPI flash devices. Refer to the previous section.

Both the Windows version and DOS version 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 both the DOS and Windows versions have the same syntax. The command line syntax for fpt.exe and fptw.exe is:

```
fpt     [/?]
                [/h]
                [/c]
                [/b]
                [/i]
                [/f:<file>]
                [/verify:<file>]
                [/d:<file>]
                [/address:<value>]
                [/length:<value>]
                [/l]
                [/desc]
                [/bios]
                [/me]
                [/gbe]
                [/pdr]
                [/y]
                [/q]
                [/e]
                [/erase]
                [/p:<file>]
                [/log]
                [/list]
                [/iTPM <Enabled/Disable>]
                [/fovs]
                [/ex]
                [/u]
                [/o]
                [/in]
                [/n]
                [/id]
                [/v]
                [/MacFile]
                [/Lock]
                [/DumpLock]
                [/PskFile]
                [/CloseMnf] [/txtconf:<file>]
```

/? or /h—displays the help screen.

/c—asks the user to confirm that the entire flash part will be erased. It is not necessary to erase the flash before a load. The load command will erase the region before a load is performed. If two flash devices are present, both devices will be erased.

/b—checks to see whether the flash has been erased and generates a message stating whether or not the flash is blank. If there are two flash devices and neither are blank, the program will return with a non-blank message.

/i—displays information about the flash image. This information includes:

- Start and end of each region
- Read and write permissions
- Whether or not the flash descriptor is valid.

/f—loads a binary file into the flash starting at address 0x0000. The flash device must be written in 4kB sections. The total size of the flash device must also be in increments of 4kB.

/verify—compare binary file to the image in the flash. If the binary file is not identical to the flash, the address and expected value of the first 5 bytes will be displayed on the screen. The flash device must be written in 4kB sections. The total size of the flash device must also be in increments of 4 KB. This must be performed immediately after programming the SPI flash device.

/d—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 4 KB.

/address or /a—used in conjunction with load, verify or dump, and allows the user to load, verify or dump a file beginning at the specified address. This option cannot be used with the /desc, /bios, /me or /gbe options.

/length or /l—used in conjunction with the load, verify or dump options, and allows the user to specify the number of bytes to load, verify or dump. This option cannot be used with the /desc, /bios, /me or /gbe option.

/desc—used in conjunction with the load, verify or dump options, and allows the user to load, verify or dump to the descriptor region leaving the rest of the flash untouched. This option cannot be used with the /address or /length option.

/bios—used in conjunction with the load, verify or dump options, and allows the user to load, verify or dump to the BIOS Region leaving the rest of the flash untouched. This option cannot be used with the /address or /length option.

/me—used in conjunction with the load, verify or dump options, and allows the user to load, verify or dump to the ME Region leaving the rest of the flash untouched. This option cannot be used with the /address or /length option.

/gbe—used in conjunction with load, verify or dump, and allows the user to load, verify or dump to the GBE Region leaving the rest of the flash untouched. This option cannot be used with the /address or /length option.

/pdr—used in conjunction with load, verify or dump, and allows the user to load, verify or dump to the PDR Region leaving the rest of the flash untouched. This option cannot be used with the /address or /length option.

/y—do not prompt when a warning occurs. If a warning occurs, the warning will be displayed, however, the specified command will continue to run.

/q—do not display output to the screen.

/e—do not erase any area before writing to the flash.

/erase – Erase the contents of the flash

/p—specifies a different flash part definition file to use instead of the one located within the executable.

/log— specifies the name of the log file created

/list—list all the SPI devices supported

/iTPM <Enabled/Disable>—enable/disable integrated TPM

/FOVs—list the names and id numbers of all fixed offset variables (FOVs) supported.

/ex /o <List filename>—extracts list of variables and the current value to the text file specified

/u—updates parameter specified by /n or /id option.

/in <FOV filename>—specifies the fixed offset parameter file to update all fixed offset variables.

/n—specifies the name of the variable to update using the /u and /v option

/id—specifies the name of the variable to update using the /u and /v option

/v—specifies the value of the variable. Used with /u and /n or /id option

/MacFile —specifies the MAC address file that FPT can read and program MAC addresses for multiple systems

/lock—locks the descriptor region according to Intel® recommendation. Please see section 4.7.3 Region Access Control for more information.

/dumplock—displays the current descriptor lock settings

/PSKFile <PSK filename>—species the name of the PSK file that FPT can read and program PSK value for multiple systems

/closeMnf—Option used at the end of the manufacturing line. Please see Section 4.11 End of Manufacture for more details

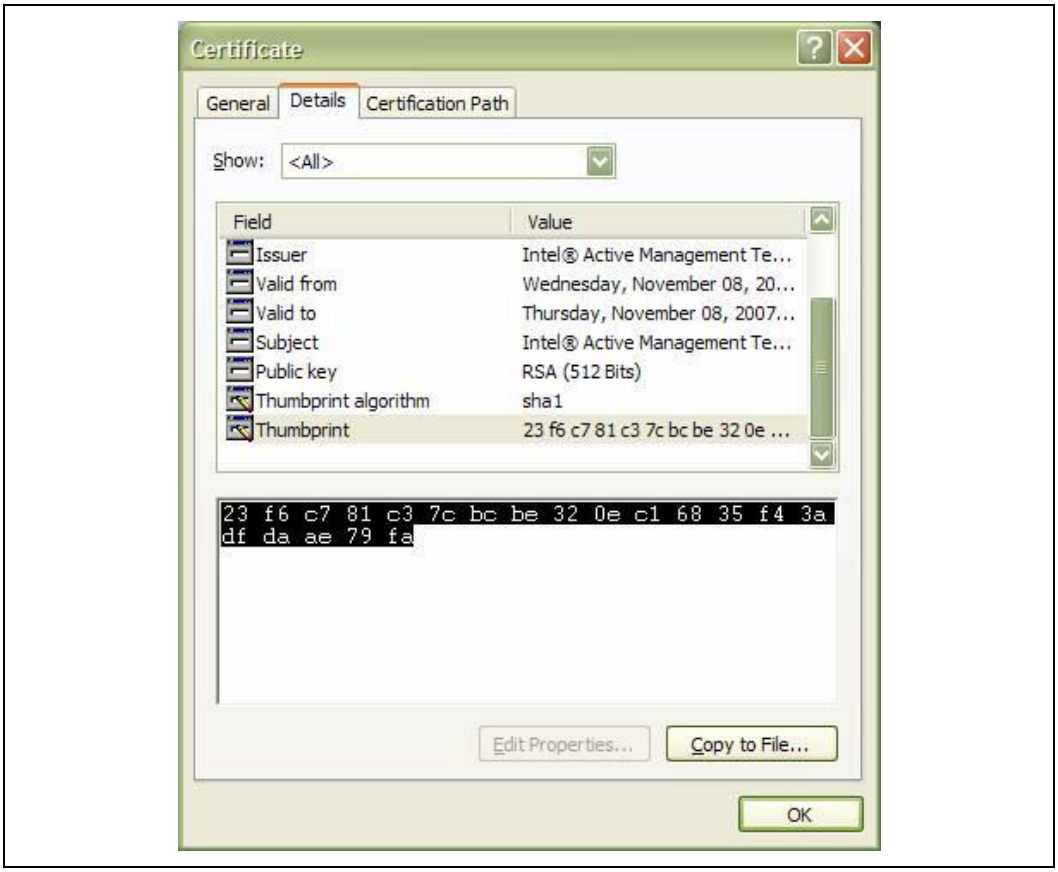
/txtconf –option used with /closeMnf to do TXT setting for iTPM

4.9 How to Update Hash, Certificate and Profile FOVs

Hash Certificates that are entered through the FOV mechanism can not be set as default certificates. For this reason, the Hash Certificates that are entered through the FOV mechanism will be deleted after a full un-provision. Only Hash certificates entered through FITC will remain after a full un-provision is performed.

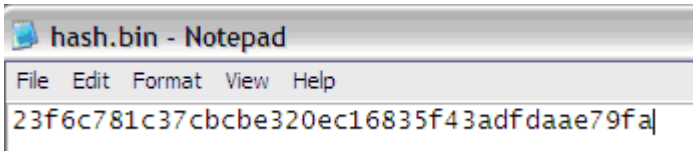
1) User must copy the raw hash values from a valid certificate file

Figure 33. Raw Hash vale from certificate file



2) Paste the raw hash contents to a text file and remove all the spaces from there and save that file as hash.bin

Figure 34. Hash BiN file



3) Hash FOV can only be flashed using FPT's **-u –in** option like this:

fpt –u –in sampleparam.txt

Where sampleparam.txt is the file that is used to update multiple FOVs together. In this case we want to update FOV as well. So user must include following entries to the sampleparam.txt file:

[ZTCEnable]

Enabled = 0x0

Value = 0x00

[Hash1]

Enabled = 0x1

IsActive = 0x1

FriendlyName = myHash3

RawHashFile = hash.bin

[CfgSrvFQDN]

Enabled = 0x0

Value = Intel.com

4.10 fparts.txt File

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

```
Flash Programming Tool.  Version X.X.X
Reading LPC BC register... 0x00000000
BIOS space write protection is enabled
Disabling BIOS space write protection
Reading LPC RCBA register... 0xFED1C001
SPI register base address... 0xFED1F020
Loading the flash definition file
Reading file "fparts.txt" into memory...
Initializing SPI utilities
Reading HSFSTS register... Flash Descriptor: Valid

        --- Flash Devices Found ---
>>> Error: There is no supported SPI flash device installed!
```

If the device is not located in the fparts.txt file, the user is expected to provide information about their device and insert the values into the file using the same format as the rest of the devices. 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 End of Manufacture

Before a platform leaves the manufacturing floor, the descriptor region must be locked, the MEManuf counter must be set to 0, and the Global valid bit must be set.

Specifically for Intel® TPM SKUs, the NV area must also be locked.

In the past, steps 1 to 3 were performed individually by separate tools.

To end manufacture, perform the following actions:

1. Set descriptor permissions for each region. (In the past this was performed by running FITC or FAUPD.)
2. Set MEManuf Counter to zero. (In the past this was performed by MEManuf or FAUPD.)
3. Set Global Valid bit. (In the past this was performed by FAUPD.)

When –txtconf:<file> option is applied, FPT will read the txt file specified by the user and set policies and other configuration in iTPM to satisfy TXT requirement. The detail process is

- Define Auxiliary index
- Define Platform Default index
- Create platform default policy. Default policy are taken from the user provided file. User can override this by updating the file.
- Write-protect the Platform Default index by a write to the Platform Default index with dataSize of 0.
- Lock NV area

;; PLATFORM DEFAULT POLICY EXAMPLE
;;
;; Version (byte)
;; HashAlg (byte)
;; PolicyType (byte)

;; SINIT (byte)
;; Policy Control Field (dword)
;; Revocation Counters (word for each)
;; Hash (# of bytes depends on the HashAlg)
;;
Version = 0x01
HashAlg = 0x00
PolicyType = 0x3
SINIT = 0x10
PolicyControlField = 0x00000000
RevocationCounters = 0x0000, 0x0000, 0x0000
Hash = 0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07, 0x08, 0x09, 0x0A, 0x0B, 0x0C, 0x0D, 0x0E, 0x0F, 0x10, 0x11, 0x12, 0x13, 0x14

4.12 Examples

The following examples illustrate the usage of the DOS version Fpt.exe of the tool. The Windows version Fptw.exe will behave in the same manner apart from running in a Windows environment.

4.12.1 Example 1

```
C:\ fptw.exe /me /d STDOUT
```

This usage displays the entire contents of the ME Region one screen at a time. Pressing Enter will display the next page, pressing q will exit the program.

4.12.2 Example 2

```
C:\ fpt.exe /f image.bin /address 0x100 /length 0x800
```

This usage loads 2KB of the binary file image.bin starting at address 0x0000. The starting address and the length must be a multiple of 4KB.

4.12.3 Example 3

```
C:\ fpt.exe /f bios.rom /bios

-------------------------------------------
Flash Programming Tool. Version X.X.X

Reading file "fparts.txt" into memory...
Initializing SPI utilities
Reading HSFSTS register... Flash Descriptor: Valid

        --- Flash Devices Found ---
        SST25VF016B      ID:0xBF2541                 Size: 2048KB
(16384Kb)

Using software sequencing.
Reading LPC BC register... 0x00000001
Reading file "BIOS.ROM" into memory...
- Erasing Flash Block [0x101000]... - 100% complete.
- Programming Flash [0x100400]... - 100% complete.
Write Complete
```

This usage loads the bios.rom file into the BIOS Region and verifies that the operation ran successfully.

4.12.4 Example 4

```
C:\ fptw.exe /desc /d descdump.bin
-------------------------------------------

Flash Programming Tool. Version X.X.X

Reading file "fparts.txt" into memory...

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

        --- Flash Devices Found ---
        SST25VF016B      ID:0xBF2541                  Size: 2048KB
(16384Kb)

Using software sequencing.

- Reading Flash [0x000040]... 4KB of    4KB - 100% complete.
Writing flash contents to file "descdump.bin"...

Memory Dump Complete
```

This usage writes the contents of the Descriptor Region to the file descdump.bin.

4.12.5 Example 5

```
C:\ fptw.exe /i
Flash Programming Tool. Version X.X.X

Reading LPC BC register... 0x00000001
Reading LPC RCBA register... 0xFED1C001
SPI register base address... 0xFED1F020
Loading the flash definition file
Reading file "fparts.txt" into memory...
Initializing SPI utilities
Reading HSFSTS register... Flash Descriptor: Valid

        --- Flash Devices Found ---
        SST25VF016B      ID:0xBF2541                  Size: 2048KB
(16384Kb)

Using software sequencing.

        --- Flash Image Information --
        Signature: VALID
        Number of Flash Components: 1
                 Component 1 - 2048KB (16384Kb)
        Regions:
                 Descriptor - Base: 0x000000, Limit: 0x000FFF
                 BIOS       - Base: 0x100000, Limit: 0x1FFFFF
                 ME         - Base: 0x001000, Limit: 0x0FDFFF
                 GbE        - Base: 0x0FE000, Limit: 0x0FFFFF
        Master Region Access:
                 CPU/BIOS - ID: 0x0000, Read: 0xFF, Write: 0xFF
```

```
ME        - ID: 0x0000, Read: 0xFF, Write: 0xFF
GbE       - ID: 0x0218, Read: 0xFF, Write: 0xFF
```

This usage displays information about the flash devices present in the computer. The base address refers to the start location of the particular regions and the limit address refers to the end of the region. If the flash device is not specified in fparts.txt, Fpt will return the error message “There is no supported SPI flash device installed”.

4.12.6 Example 6

```
C: \ fpt.exe /verify outimage.bin
Flash Programming Tool. Version X.X.X
Reading LPC BC register... 0x00000001
Reading LPC RCBA register... 0xFED1C001
SPI register base address... 0xFED1F020
Loading the flash definition file
Reading file "fparts.txt" into memory...
Initializing SPI utilities
Reading HSFSTS register... Flash Descriptor: Valid
        --- Flash Devices Found ---
        SST25VF016B      ID:0xBF2541                 Size: 2048KB
(16384Kb)
        SST25VF016B      ID:0xBF2541                 Size: 2048KB
(16384Kb)
Using software sequencing.
Reading file "outimage.bin" into memory...

RESULT: Data does not match!
        0x00000000: 0x5A - 0x5A
        0x00000001: 0xA5 - 0xA5
        0x00000002: 0xF0 - 0xF0
        0x00000003: 0x0F - 0x0F
        0x00000004: 0x01 - 0x01
```

This usage compares the ME Region programmed on the flash with the specified firmware image file outimage.bin. If the /y option is not used, the user will be 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. They /y option will proceed with the comparison without warning.

4.12.7 Example 7

```
C: \ fpt.exe /verify outimage.bin

Flash Programming Tool. Version 0.8.11

Reading file "fparts.txt" into memory...
Initializing SPI utilities
Reading HSFSTS register... Flash Descriptor: Valid

      --- Flash Devices Found ---
      SST25VF016B ID:0xBF2541 Size: 2048KB (16384Kb)

Using software sequencing.
Reading file "outimage.bin" into memory...

RESULT: Data does not match!
      [0x000000] Expected: 0x0B, Found: 0x5A
Total mismatches found in 64 byte block: 27
```

This usage compares the file 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 will result in a mismatch.

§

5 MEManuf and MEManufWin

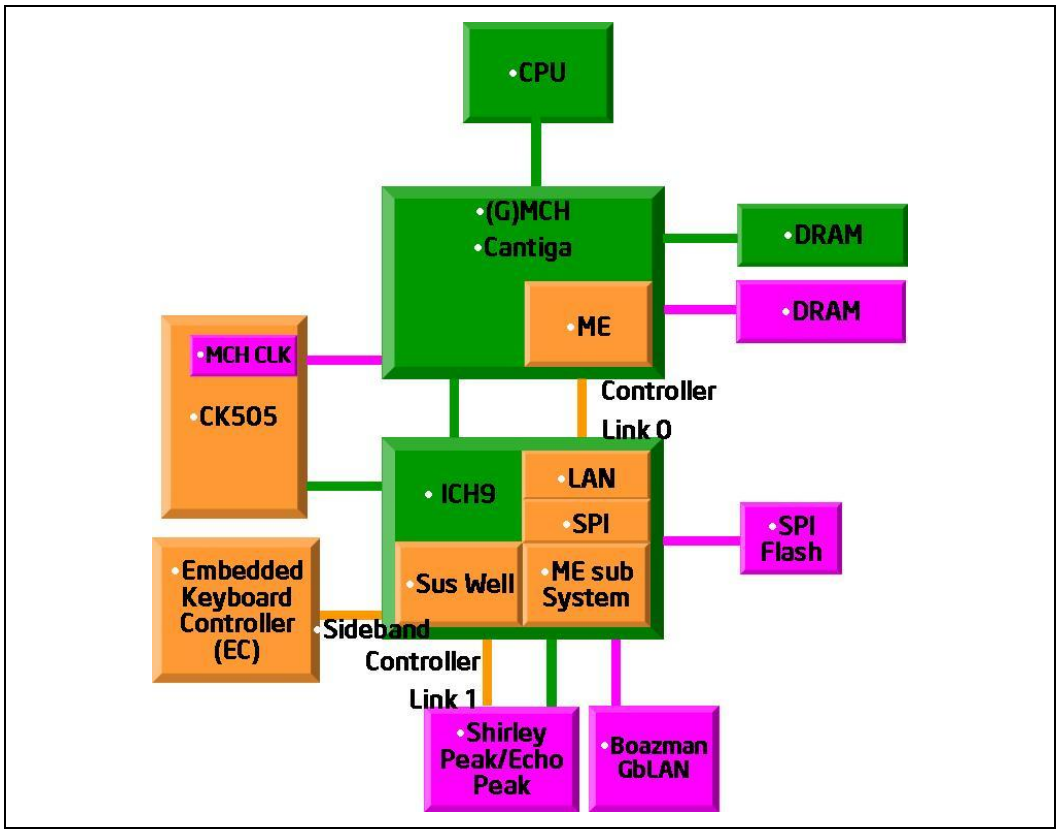
MEManuf validates Intel® ME functionality (verifies that all its components have been assembled together correctly) on the manufacturing line. The tool accomplishes this by invoking the test program embedded in the ME firmware. The test covers the following features:

- SMBus Interface
- BIOS, and BIOS/FW connectivity
- C-Link (ME-ICH and ICH-Shilo)
- EC
- Intel® TPM
- Wireless connectivity.

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

MEManuf can run either a complete or partial test (see below).

Figure 35. Montevina Chipset Layout



MEManuf can also decrement the FW counter to 0 (zero) or return its present value (see below).

5.1 Requirements

MEManuf runs on an Intel® AMT-enabled computer running any of the following:

- MS-DOS* 6.22
- Windows* 98 DOS
- FreeDOS* v 1.1.32a
- DRMK DOS*
- MEManuf will only work on a system running on AC power

MEManuf will run on a Windows (Windows* XP SP1/2, Windows* XP32/64, Windows Vista* 32/64 or Windows* PE) computer. The Windows version of MEManuf requires

the installation of the Intel ME Interface driver, and if testing Intel® TPM also requires the Intel® TPM driver. MEManuf has a runtime of less than 30 seconds.

5.2 Windows* PE requirements

Windows PE has specific requirements for AMTManuf. The usage for the tool remains unchanged.

For Intel® AMT the following drivers are required:

- The ME Interface driver must be installed in the Windows PE image.
- The Windows PE image must be WMI enabled.

For Intel® TPM the Intel® TPM driver is required.

5.3 Firmware Counter

For security reasons, the firmware counter tracks the number of times a manufacturing test command has been sent to the Host Interface. When the counter reaches zero, any manufacturing test command issued to the Host Interface is no longer acknowledged.

Use of the MEManuf complete test decrements this counter with each run. This limits the number of times a test system can be repaired in order to have it pass the manufacturing test.

Once the counter has reached zero, the image needs to be reprogrammed into the SPI flash device. If the CPU does not have write access to the Descriptor Region, the counter can only be reset by reprogramming the image using the Security Override Strap if needed.

5.4 Complete Test

MEManuf is run in three stages or invocations. Its use is expected to be automated so the tool is called from a batch file autoexec.bat at test system boot. The following sequence describes the recommended usage model for MEManuf.

5.4.1 First invocation

- MEManuf is invoked by autoexec.bat. This is the first manufacturing line test that is performed on the test system.
- MEManuf issues the Host Interface manufacturing test command.
- FW saves the results to registers.

- Decrements the FW counter.
- System reboots.

5.4.2 Second invocation

- MEManuf will be invoked by autoexec.bat if there were no failed tests from the first invocation.
- MEManuf obtains results of the test (from the first invocation) from the registers. If a test has failed before the second invocation, the test will return the results and the second invocation will not be called.
- MEManuf reports these results to the user.

5.4.3 Last invocation

- MEManuf is invoked by autoexec.bat.
- MEManuf decrements FW counter to 0 (zero).
- A sample autoexec.bat is included in the kit. The autoexec.bat included only runs in the supported DOS environments mentioned above.

5.5 Partial Test

- MEManuf may optionally be run in partial test mode. The partial test is identical to the complete test (see above) except that:
- S5 functionality is not tested. As a result of this, the partial test is much faster.
- The FW counter is not decremented.

Note: Intel recommends that each manufacturer perform the complete test for manufacturing line validation.

5.6 Intel® TPM Impact on MEManuf

Intel® TPM functionality has been added to the tool.

It is possible to execute this ordinal irrespective of the state of Intel® TPM (disabled, deactivated, un-owned) as long as the image is an Intel® TPM SKU.

5.7 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:

```
MEManufWin.exe [AMT|TPM|TPM+AMT] <option>
```

Options are only available with AMT or iTPM+AMT.

-full—will run the partial test plus a system reset. The system reset will verify that the ME is able to run in the S5 state.

-part—invokes the partial test only. See the section above.

-graceful—similar to the full test, but will test to see if the ME can run in the S4 (hibernate) state. A graceful test can only be run on Windows and the system must be able to go into hibernate mode.

Note: The graceful test will not run if the system cannot go into hibernate mode or the power package selected does not support the ME running in the S4 state.

-nowlan—ignores the wireless LAN test. If a wireless card is not present on the machine MEManuf will return an error message. This optioncan be used with –full, -part or –graceful test options.

-block—blocks all future invocations of the full and graceful tests.

-counter—displays the number of full tests remaining.

-version—displays the version of the MEManuf.

5.8 Examples

5.8.1 Example 1

```
MEManufWin.exe –graceful
```

This usage runs the full test, however, instead of a hard power cycle, MEManuf will send Windows into the S4 hibernate mode and then bring the system back to the S0

state. This command should be used again to view the test results. If the power package selected does not support the ME in the S4 state, MEManuf will not run and will return the following error message:

"Intel® AMT power policy prevents ME from bringing the system back from hibernation, so hibernation will not be performed. All other tests ran successfully."

5.8.2 Example 2

```
MEManufWin.exe -block
```

This usage sets the MEManuf full counter to 0 (zero) and prevents any more full or graceful tests from being executed. Partial test will still be allowed. If the user needs to run additional full or graceful tests, the complete SPI image must to be reprogrammed.

5.8.3 Example 3

```
MEManufWin.exe -Full
```

This usage will immediately send the computer into an S5 state and then power back on. To view the results, the user must run the -Full option again. If this command is invoked on Windows, the user may lose unsaved data.

§

6 MEInfo

MEInfoWin and MEInfo provide a simple test to check whether the Intel® ME firmware is alive or not. Both tools perform the same test, query the Intel® ME firmware including Intel® AMT and Intel® TPM, and retrieve data. Below is a list of the data that each tool will return.

6.1 Requirements

MEInfo runs on the following:

- MS-DOS 6.22
- Windows 98 DOS
- FreeDOS 8.0
- DRMK DOS v8.0

MEInfo Hill run on Windows (Windows PE, Windows XP 32/64, Windows XP SP1/2, Windows Vista32/64 Windows 7 32/64). MEInfo and MEInfoWin are also command-line executables.

The Intel® MEI and the LMS drivers must be installed (MEInfoWin only).

If testing Intel® TPM functionality, only the Intel® TPM driver must be installed.

6.2 Windows* PE requirements

Windows* PE has specific requirements for MEInfo. The usage for the tool remains unchanged.

For Intel® AMT the Intel ME Interface driver must be installed in the Windows PE image.

The Windows PE image must be WMI enabled.

MEInfo will report an LMS error. This is expected behavior as the LMS driver cannot be installed on Windows PE.

For Intel® TPM the Intel® TPM driver is required.

6.3 Usage

The executable can be invoked by:

```
MEInfo.exe [-feat <feature name> -value <value>]
MEInfo.exe [-feat <feature name>]
```

-feat <feature 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 will display indicating success. If the values are not identical, the actual value of the feature will be returned. Only one feature may be requested in a command line.

-feat <feature 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 MEInfo

Note: If the name or value is more than one word the name/value must be between quotations marks. Ex. "Configuration State"

MEInfo can retrieve all of the information detailed below, however, depending on the SKU selected, some information may not appear.

Table 8. List of components for which version information must be retrieved

Component	Intel® AMT SKU 'Manageability Mode'			ASF	Intel® TPM	Field value
	AMT	ASF	None			
Tools version	X	X	X	X	X	A version string
BIOS version	X	X	X	X	X	A version string
GbE version	X	--	---	---	---	A version string
Intel® MEBX Version	X	X	X	X	X	A version string
Intel® AMT Netstack version	X	X	X	---	---	A version string
Intel® AMT version	X	X	X	---	---	A version string
Intel® AMT build number	X	X	X	---	---	A number
Kernel Version	X	X	X	X	X	A version string
Kernel Build Num	X	X	X	X	X	A number
Vendor ID	X	X	X	---	---	A number
Wireless Hardware Version	X	X	X	X	X	A version string

Component	Intel® AMT SKU 'Manageability Mode'			ASF	Intel® TPM	Field value
Link status	X	--- Tool will report n/a	--- Tool will report n/a	---	---	Link up/ down
Hardware SKU	X	X	X	X	X	AMT, ASF, TPM and the possible combinations
Cryptography fuse	X	X	X	---	X	Enabled/ Disabled
Flash protection	X	X	X	X	X	Enabled/ Disabled
Last ME reset reason	X	X	X	X	X	Power up/ Firmware reset/ Global system reset
BIOS boot State	X	X	X	X	X	Pre Boot/ In Boot/ Post Boot
Configuration state	X	--- Tool will report n/a	--- Tool will report n/a	---	---	Not started/ In process/ Completed
Manageability Mode	X	X	X	---	---	Intel® AMT/ ASF/ None
User Notification State	X	--- Tool will report n/a	--- Tool will report n/a	---	---	Enabled/ Disabled
Manuf-mode override behavior	X	X	X	X	X	Disable/ Continue
Host MAC Address	X	X	X	X	X	A MAC address
Wireless MAC address	X	X	X	X	X	The MAC address of the wireless NIC if ME Wireless management is enabled
FWU Override Counter	X	X	X	X	X	(A number)/ Always/ Never
FWU Override Qualifier	X	X	X	X	X	Never/Always/Restricted
Local FWUpdate	X	X	X	X	X	Enabled/ Disabled
Secure FWUpdate	X	X	X	X	X	Enabled/ Disabled
Intel® MEI Driver version*	X	X	X	X	---	A version string
LMS version*	X	X	X	---	---	A version string
UNS version*	X	X	X			
Wireless Driver Version*	X	X	X	X	X	A version string
Intel® TPM fuses (MCH/ICH/soft strap MCH/ soft strap ICH)	X	X	X	X	X	Enabled/ Disabled

Component	Intel® AMT SKU 'Manageability Mode'			ASF	Intel® TPM	Field value
Intel® TPM Vendor ID	---	---	---	---	X	A version string
Intel® TPM SPEC Version	---	---	---	---	X	A version string
Intel® TPM FW Version	---	---	---	---	X	A version string
Intel® TPM FW Build	---	---	---	---	X	A number
Intel® TPM State	---	---	---	---	X	Operational / Failed state
Intel® TPM Operational Mode	---	---	---	---	X	Active, Enabled, Owned
iTPM – FIPS 140-2	---	---	---	---	X	False/True
iTPM - Physical presence life time lock flag	---	---	---	---	X	False/True
iTPM - Physical presence command enabled flag	---	---	---	---	X	False/True
iTPM - Physical presence HW enabled flag	---	---	---	---	X	False/True
Failed attempts threshold	---	---	---	---	X	
Initial lockout time	---	---	---	---	X	
Lockout multiplier	---	---	---	---	X	
Fade-out period	---	---	---	---	X	

6.4 Examples

6.4.1 Example 1

This is a simple test that indicates whether the firmware is alive and if so, will return device specific parameters. The output is from the Windows version. The DOS version will not display the UNS version, Intel® Management Engine Interface or LMS version numbers.

```
C:\ MEInfoWin.exe
Copyright (C) 2005-07, Intel Corporation

AMT SKU Found.
Intel(R) MEInfo Win Version: 4.0.0.XXX

BIOS Version      VVXXXX.XXX

Intel(R) AMT code versions:
        Flash:    4.0.0
        Netstack: C.0.0
        AMTApps:  C.0.0
        AMT:      4.0.0
```

```
        SKU:      ASF IAMT
        VendorID: 8086
        Build Number:    XXXX
        Recovery Version:         4.0.0
        Recovery Build Num:       XXXX
        Legacy Mode:     False
Intel(R) Mode:           SMB
Link status:             Link up
Cryptography fuse:       Enabled
Flash protection:        Enabled
Last ME reset reason:    Power Up
Configuration state:     Not Completed
BIOS boot State:         Post Boot
Host Mac Address:        00-dd-bb-cc-aa-00
Wireless Mac Address:    00-55-44-33-22-11
FWU Override Counter:    Never
FWU Override Qualifier:  Always
FW on Flash Desc Override:        Disable
Wireless Driver Version: 4.0.X.XXXX
Shilo Hardware Version:  4.0.X.XXXX
UNS Version:             4.0.X.XXXX
LMS Version:             4.0.X.XXXX
MEI Version:             4.0.X.XXXX
TPM Fuses (MCH/ICH/soft strap MCH/ soft strap ICH):  Disabled
```

6.4.2 Example 2

This example retrieves the current value of the Flash version

```
C:\ MEInfo.exe -feat “AMT”

Intel(R) AMT:            4.0.0
```

6.4.3 Example 3

This example compares the dedicated MAC address stored in the ME with the MAC address provided in the command line argument. The value should be entered as it would appear in MEInfo. If the values do not match, an error message will be printed and the correct value will be displayed, as seen below.

```
C:\ MEInfo.exe -feat “Host MAC Address” –value 005544332211

Host Mac Address: Error – actual value is: 00-55-44-33-22-11
```

6.4.4 Example 4

This example checks whether the computer has completed the setup and configuration process. If the parameter name or the value has a space, the value or name should be entered in quotes.

```
C:\ MEInfo.exe -feat "Configuration State" -value "Not Completed"

Success.  The values are identical
```

6.4.5 Example 5

If Intel® AMT mode is not selected, MEInfo will display different results. ASF and TPM will have the same results. The results below are from a computer without Intel® AMT mode selected.

```
Copyright (C) 2005-07 Intel Corporation. All Rights Reserved.
No AMT Found.
Intel(R) MEInfo version: 4.0.0.XXXX
Intel(R) ME code versions:
        Flash:                  4.0.0
        SKU:                    ASF IAMT
        VendorID:               8086
        Build Number:           XXXX
        Recovery Version:       4.0.0
        Recovery Build Num:     XXXX
```

§

7 Firmware Update (FWUpdLcl)

FWUpdate allows an end user, such as an IT administrator, to update the ME firmware 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 Region. It only updates the firmware code portion that Intel provides on the OEM website. FWUpdate will update the entire ME code area.

The image file that the tool uses for the update is not the image file used to create the complete SPI firmware image file. A sample firmware image file for updating, MV_ICH9_REL_IAMT_BYP_ME_UPD.BIN, is located in the kit's NVM image folder.

Please be aware that firmware update takes approximately 1-4 minutes to complete, based on flash device.

7.1 Requirements

FWUpdate is a command-line executable that can be run on an Intel® AMT- enabled system that needs updated firmware.

Please be aware that firmware update takes approximately 1-4 minutes to complete, based on flash device.

Note: FWUpdate only supports upgrading firmware. Downgrading firmware is not supported in Montevina.

FWUpdate can be run on computers with one of the following operating systems:

- MS-DOS 6.22
- Windows* 98 DOS
- FreeDOS 8.0
- DRMK DOS 1.1.32a
- Windows (Windows* XP SP1/SP2, Windows* XP32/64, Windows* PE or Windows Vista* 32/64, Windows 7 32/64).

The requirements for running FWUpdate depend on the computer's OS. The requirements also depend on the type of Intel® AMT manageability connection required (secure or a non-secure).

7.2 Non-Secure Dos Requirements

The user must have administrator access.

7.3 Non-Secure Windows Requirements

Intel® ME Firmware Local Update must be enabled in the Intel® MEBX.

In the Intel® MEBX, Intel® AMT must be selected in the Intel® Manageability Feature Selection menu.

The Intel® MEI driver must be installed.

7.4 Secure Windows Requirements

In the Intel® MEBX, Intel® AMT must be selected in the Intel® Manageability Feature Selection menu.

In the Intel® AMT Configuration menu (in the Intel® MEBX), Local Firmware Update must be enabled.

The Intel® AMT LMS must be installed.

7.5 Windows* PE Requirements

The Intel® ME driver must be installed in the Windows* PE image.

The Windows* PE image must be WMI-enabled.

7.6 Enabling and Disabling Local Firmware Update

Disabling Firmware Local Update in the Intel® MEBX prevents any updating of the firmware. However, even if Firmware Local Update is disabled, you can still enable updating the firmware for a limited number of times which can be done during manufacturing. To do this, configure the two variables Local FWU Override Counter and Local Firmware Override Qualifier to temporarily override the Intel® MEBX settings. These parameters can be modified by using the FITC or FPT.

When Local FWU Override Counter has a value between 1 and 255, firmware updates are allowed even if updates are disabled in the Intel® MEBX settings. After the flash is programmed, each time the computer restarts it causes Local FWU Override Counter to be decremented. When Local FWU Override Counter reaches 0, firmware updates are no longer allowed if they are not enabled in the MEBx settings.

Note: The restart that takes place after the flash memory has been programmed also causes Local FWU Override Counter to be decremented. Therefore if you want to enable updating the firmware N times, you need to assign Local FWU Override Counter the initial value N+1.

If the Local FWU Override Counter is set to -1 and the Local Firmware Override Qualifier is set to 0, firmware updates are always allowed regardless of the settings in the Intel® ME BIOS extension

The following table shows the possible value combinations for the two variables. To enable local firmware updates, make sure both variables are assigned the correct values.

Table 9. Firmware Override Update Variables

	Local FWU Override Qualifier = 0 (zero)	Local FWU Override Qualifier = 1 (one)	Local FWU Override Qualifier = 2 (two)
Local FWU Override counter = 0 (zero)	Local Firmware Updates NOT Allowed	Local Firmware Updates NOT Allowed	Local Firmware Updates NOT Allowed
Local FWU Override Counter = -1 (minus one)	Local Firmware Updates Allowed	Local Firmware Updates NOT Allowed	Local Firmware Updates Allowed only until ME is configured
Local FWU Override Counter = 0<n<255	Local Firmware Updates Allowed	Local Firmware Updates Allowed	Local Firmware Updates Allowed

7.7 Usage DOS Version

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

To differentiate between the image files used for updating and those used for programming the entire flash memory, files used for FWUpdate include the string UPD in their file names.

Please be aware that firmware update takes approximately 1-4 minutes to complete, based on flash device.

The executable can be invoked by:

```
FWUpdLcl.exe <Image File> – [options]
```

Options—these options are only valid if the system has Intel® AMT selected in the MEBx. The options can be one or more of the following:

-h- Displays command line options

-TPM –Required if TPM is enabled on the system.

-verbose –Prints out debug information to the screen

-HaltRCFG – Halts all remote configuration network traffic and prevents remote configuration. The system can not be remotely configured until a local agent, such as Activator or ZTCLocalAgent, is run to initiate delayed provisioning mode. Only valid with firmware 4.1.3 and greater. The haltRCFG command can NOT be used as a command line argument while performing firmware update.

Image File—image file of the firmware to be updated. This image file is not the same image file used by the FITC.

7.8 Usage Windows* Version

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

To differentiate between the image files used for updating and those used for programming the entire flash memory, files used for FWUpdate include the string UPD in their file names.

Please be aware that firmware update takes approximately 1-4 minutes to complete, based on flash device.

The executable can be invoked by:

```
FWUpdate.exe <Image File> – [options]
```

Image File—image file of the firmware to be updated. This image file is not the same image file used by the FITC.

Options—these options are only valid if the system has Intel® AMT selected in the MEBx. The options can be one or more of the following:

– user <User Name>—admin user name.

– pass <Password>—password that corresponds with the admin user.

–TLS—connect to the firmware using TLS. If the –TLS option is specified without the –user and –pass options, the program attempts to use Kerberos Authentication.

–host <hostname>—The hostname specified in the firmware

– Cert <Certificate>—name of the certificate used if TLS mutual authentication mode is enabled.

–EOI—connect to the firmware using legacy protocol. If this option is not specified WS-MAN will be sued

–DASH—connect to the firmware using DASH protocol. If this option is not specified WS-MAN will be used

–TPM —Update through the TPM interface. If -TPM option is used –key or –msf is required

–key <owner_key>–TPM owner password if the TPM is in an owned state

-MSF <file_name>– Microsoft* Vista* generated AuthData file if TPM is in an owned state. This required if the user must input a Windows* Vista* specific auth data file for authentication

– generic—will update the firmware without credentials, even if the system is already set up and configured. If this option is used, all other options are ignored. The -generic option requires that the ME Interface driver is installed. When using the -generic option the firmware update will occur via the ME Interface driver.

-verbose–Prints debug information to the screen

-HaltRCFG – Halts all remote configuration network traffic and prevents remote configuration. The system can not be remotely configured until a local agent, such as Activator or ZTCLocalAgent, is run to initiate delayed provisioning mode. Only valid with firmware 4.1.3 and greater. The haltRCFG command can NOT be used as a command line argument while performing firmware update.

7.9 Examples

7.9.1 Example 1

```
FWUpdLcl.exe MV_ICH9_REL_IAMT_BYP_ME_UPD.BIN –user Admin –pass Admin@98 –
TLS –host Cert_Name
```

The above will update the local firmware using a TLS connection to the firmware. The certificate name Cert_Name matches the certificate name provided in the firmware.

7.9.2 Example 2

```
FWUpdLcl.exe MV_ICH9_REL_IAMT_BYP_ME.BIN –user admin –pass Admin@98
Error: Bad seek
Error: failed to parse image file
```

The above is the error message that is seen if the wrong firmware binary file is used. When updating the firmware, the correct file for this tool name contains the string UPD in the filename.

7.9.3 Example 3

```
FWUpdLcl.exe -haltRCFG
```

This option is only valid after firmware 4.1.3 or greater has been applied. After the firmware has been updated to at least 4.1.3 calling the haltRCFG option will halt all remote configuration traffic and prevent remote configuration. The haltRCFG command can NOT be used as a command line argument while performing firmware update.

§

Appendix A Fixed offset Variables

All of the fixed offset variables have an id and a name. The "-fov" option will display a list of the ID and their respective name. The variable name must be entered exactly as displayed below.

FOV ID	Variable Name	Description	Value Type
0x0001	MEStateLock	Determine if user can modify ME State. Enabling the lock means the user can NOT change the ME state	0x00 – Disable Lock 0x02 – Enable Lock
0x0002	MEStateControl	Determines the default state of the ME	0x00 – Disabled 0x02 – Enabled
0x0004	MEPwrFeature	This option will prevent the user from changing the power package selected	0x00 – Allow user to modify power package 0x02 – Do NOT allow user to modify power package
0x0005	DefPowerPackage	Selects the default power package. The default power package must be supported	Hex value integer indicating the power package.
0x0006	FWUpdEnable	Determines if non-secure firmware updates are allowed	0x00 – Firmware update disabled 0x02 – Firmware update enabled
0x0007	FWUpdOverideQualifier	Please see firmware update section for more information on this parameter	0x00 – Always 0x01 – Never 0x02 - Restricted
0x0008	FWUpdOverrideCounter	Please see firmware update section for more information on	0x00 – Never 0<N<255 – Allow N boot cycles 0xFF - Always

FOV ID	Variable Name	Description	Value Type
0x2001	PID	Platform ID that is used to uniquely identify the system when Intel® AMT is configured in Enterprise mode. The platform ID is a 64bit value consisting of 8 characters. Valid characters are capital letters (A-Z) and numbers (0-9). Please see the Intel Manageability Engine Fixed Variable Offset for more information on this algorithm.	Value should be entered with a dash "-"(e.g. 1234-1234).Integer value
0x2002	PPS	Pre-shared Passphrase is a 256 bit value consisting of 32 characters. Valid characters are capital letters (A-Z) and numbers (0-9). This is used to validate the authenticity of the Intel® AMT system when in Enterprise mode. Please see the Intel Manageability Engine Fixed Variable Offset for more information on this algorithm.	Value should be entered with a dash "-"(e.g. 1234-abcd-1234-abcd-1234-1234-abcd-1234).
0x2003	MngFeatureLock	Determine if user can modify manageability mode. Enabling the lock means the user can NOT change the manageability mode	0x00 – Disable Lock 0x02 – Enable Lock
0x2004	MngMode	Determines the default manageability mode	0x00 – None 0x01 – Intel® AMT 0x02 - ASF
0x2005	EncryptionEnable	Determines if TLS encryption is used	0xFF– Enabled TLS 0x0O– Disable TLS
0x2006	FullTestCounter	The number of Full/Gracefull test allowed by MEManuf	Integer in HEX format
0x2007	AMTConfigMode	Determines if the default AMT mode is small business mode or Enterprise	0x00 – Enterprise 0x02 – Small Business mode
0x2008	MEIdleTimeout	If the power package supports ME-Wol. This parameter determines the number of minutes before going into an M-off state	Integer (in minutes) in Hex format Value Range =0x0000 <N 0xFFFF

FOV ID	Variable Name	Description	Value Type
0x2009	RCFGEnable	Remote configuration parameter. This parameter can not be modified by FPT	This parameter can only be modified by FITC.exe
0x0003	Password	This specifies the new password the MEBX screen. This password must be a strong password and between 8 and 32 characters. This password must be entered in string format (e.g. Admin@98)	String format between 8 and 32 characters
0x200a	CfgSrvFQDN	The fully qualified domain name for the setup and configuration server.	String between 1 and 256 characters
0x200E	MEBx Password Change Policy	The policy that controls when the MEBx password and the ME Network Password are synched Default Password Only – Passwords are synched only after the password is changed from the default value. During Setup and Configuration – Passwords are synched after the Setup and Configuration process is complete. Always – Password are synched when either password is changed	0x00 – Default Password Only 0x01 – During Setup and Configuration 0x02 - Always

§

Appendix B Error Codes

B.1 Common Tool Errors – Applies to all Tools

B.1.1 Host and Network Interface Errors

Error Number	Error String	Possible Corrective Actions
0	The request succeeded	Refer to SDK documentation
1	An internal error in the Intel® AMT device has occurred	
2	Intel® AMT device has not progressed far enough in its initialization to process the command.	
3	Command is not permitted in current operating mode.	
4	Length field of header is invalid.	
5	The requested hardware asset inventory table checksum is not available.	
6	The Integrity Check Value field of the request message sent by Intel® AMT enabled device is invalid.	Refer to SDK documentation
7	The specified ISV version is not supported	
8	The specified queried application is not registered.	
9	Either an invalid name or a not previously registered Enterprise name was specified	
10	The application handle provided in the request message has never been allocated.	
11	The requested number of bytes cannot be allocated in ISV storage.	
12	The specified name is invalid.	Refer to SDK documentation
13	The specified block does not exist.	
14	The specified byte offset is invalid.	
15	The specified byte count is invalid.	
16	The requesting application is not permitted to request execution of the specified operation.	
17	The requesting application is not the owner of the block as required for the requested operation.	

Error Number	Error String	Possible Corrective Actions
18	The specified block is locked by another application.	
19	The specified block is not locked.	
20	The specified group permission bits are invalid.	
21	The specified group does not exist.	
22	The specified member count is invalid.	
23	The request cannot be satisfied because a maximum limit associated with the request has been reached.	
24	The specified key algorithm is invalid	
25	Authentication failed	
26	The specified DHCP mode is invalid.	Refer to SDK documentation
27	The specified IP address is not a valid IP unicast address.	
28	The specified domain name is not a valid domain name.	
29	Unsupported version	
30	The requested operation cannot be performed because a prerequisite request message has not been received.	
31	Invalid Table type	Refer to SDK documentation
32	The specified provisioning mode code is undefined.	
33	Unsupported object	
34	The specified time was not accepted by the Intel® AMT device since it is earlier than the baseline time set for the device.	
35	Starting Index is invalid.	
36	Specified parameter is invalid.	
37	An invalid netmask was supplied a valid netmask is an IP address in which all '1's are before the '0' – e.g. FFFC0000h is valid FF0C0000h is invalid).	
38	The operation failed because the Flash wear-out protection mechanism prevented a write to an NVRAM sector.	
39	ME FW did not receive the entire image file.	Refer to SDK documentation
40	ME FW received an image file with an invalid signature.	
41	LME can not support the requested version.	

Error Number	Error String	Possible Corrective Actions
42	The PID must be a 64 bit quantity made up of ASCII codes of some combination of 8 characters – capital alphabets (A–Z) and numbers (0–9).	
43	The PID must be a 256 bit quantity made up of ASCII codes of some combination of 8 characters – capital alphabets (A–Z) and numbers (0–9).	
44	Full BIST test has been blocked	Refer to SDK documentation
45	A TCP/IP connection could not be opened on with the selected port.	
46	Max number of connection reached. LME can not open the requested connection.	

B.1.2 Network Interface Errors

Error Number	Error String	Possible Corrective Actions
2049	The OEM number specified in the remote control command is not supported by the Intel® AMT device	Refer to SDK documentation
2050	The boot option specified in the remote control command is not supported by the Intel® AMT device	
2051	The command specified in the remote control command is not supported by the Intel® AMT device	
2052	The special command specified in the remote control command is not supported by the Intel® AMT device	
2053	The handle specified in the command is invalid	
2054	The password specified in the User ACL is invalid	Refer to SDK documentation
2055	The realm specified in the User ACL is invalid	
2056	The FPACL or EACL entry is used by an active registration and cannot be removed or modified.	
2057	Essential data is missing on CommitChanges command.	
2058	The parameter specified is a duplicate of an existing value	
2059	Event Log operation failed due to the current freeze status of the log.	
2060	The device is missing private key material.	
2061	The device is currently generating a keypair. Caller may try repeating this operation at a later time.	

Error Number	Error String	Possible Corrective Actions
2062	An invalid Key was entered.	Refer to SDK documentation
2063	An invalid X.509 certificate was entered.	
2064	Certificate Chain and Private Key do not match.	
2065	The request cannot be satisfied because the maximum number of allowed Kerberos domains has been reached. The domain is determined by the first 24 Bytes of the SID.)	
2066	The requested configuration is unsupported	Refer to SDK documentation
2067	A profile with the requested priority already exists	
2068	Unable to find specified element	
2069	Invalid User credentials	
2070	Passphrase is invalid	
2072	Need to associate a key pair with signing Key pair handle	

B.1.3 SDK Specific Errors

Error Number	Error String	Possible Corrective Actions
4096	An internal SDK error occurred	Refer to SDK documentation
4097	An ISV operation was called while the library is not initialized	
4098	The requested library I/F is not supported by the current library implementation.	
4099	One of the parameters is invalid (usually indicates a NULL pointer or an invalid session handle is specified)	
4100	The SDK could not allocate sufficient resources to complete the operation.	
4101	The Library has identified a HW Internal error.	
4102	The application that sent the request message is not registered. Usually indicates the registration timeout has elapsed. The caller should reregister with the Intel AMT enabled device.	Refer to SDK documentation
4103	A network error has occurred while processing the call.	
4104	Specified container can not hold the requested string	
4105	ISVS_InitializeCOMinThread was not called by the current thread.	

Error Number	Error String	Possible Corrective Actions
4106	URL required	

B.2 Intel® MEI Errors – Applies to all Tools

Error Number	Error String	Possible Corrective Actions
8192	Intel ® ME Interface : Internal error	Tool failed due to an internal error. Please report error
8193	Intel ® ME Interface : Cannot locate ME device	
8194	Intel ® ME Interface : Memory access failure	
8195	Intel ® ME Interface : Write register failure	
8196	Intel ® ME Interface : Cannot alocate memory	Close other applications and retry
8197	Intel ® ME Interface : Circular buffer overflow	Tool failed due to an internal error. Please report error
8198	Intel ® ME Interface : Not enough memory in circular buffer	
8199	Intel ® ME Interface : ME Device not ready for data transmission	
8200	Intel ® ME Interface : Unsupported bus message protocol version	
8201	Intel ® ME Interface : Unexpected interrupt reason	
8202	Intel ® ME Interface : Intel ® AMT device unavailable	
8203	Intel ® ME Interface : Unexpected ME device response	
8204	Intel ® ME Interface : Unsupported message type	
8205	Intel ® ME Interface : Cannot find host client	
8206	Intel ® ME Interface : Cannot find ME client	
8207	Intel ® ME Interface : Client already connected	Tool failed due to an internal error. Please report error
8208	Intel ® ME Interface : No free connection available	
8209	Intel ® ME Interface : Illegal parameter	
8210	Intel ® ME Interface : Flow control error	
8211	Intel ® ME Interface : No message	
8212	Intel ® ME Interface : Buffer too large	
8213	Intel ® ME Interface : Buffer too small	
8214	Intel ® ME Interface : Circular buffer not empty	

B.3 Firmware Update Errors

Error Number	Error String	Possible Corrective Actions
8704	Firmware update operation not initiated due to a SKU mismatch.	Check that the FW image is meant for this SKU
8705	Firmware update not initiated due to version mismatch.	Check that the FW image is of a later rev than the one already on the system
8706	Firmware update not initiated due to invalid signature.	Check that the FW image is a valid Intel supplied FW image
8707	Firmware update failed due to an internal error.	Tool failed due to an internal error. Please report error
8708	Firmware Update operation not initiated because a firmware update is already in progress.	A FW update is already in progress. Retry later
8709	Firmware update failed due to an invalid code partition.	Retry operation. Partition may have been invalidated due to a previously failed FW Update operation
8710	Firmware update failed due to insufficient memory.	Close other applications and retry operation
8711	Firmware update not initiated because FW is not ready.	Retry operation
8712	Firmware update failed due to authentication failure	Check credentials supplied
8713	Firmware update not initiated due to an invalid FW image header.	Check that the FW image is a valid Intel supplied FW image
8714	Firmware update not initiated due to a file open or read failure.	Check that the image file and it's path are correct
8715	Firmware update failed due a HTTP operation failure.	Retry the operation- network errors may be transient
8716	Incorrect or insufficient number of arguments.	Check help page for usage info
8717	Firmware update not initiated due to invalid hostname specified.	Check that the hostname is valid
8718	Firmware update operation timed-out. Can not determine if the operation succeeded.	Run MEInfo to check if operation succeeded
8719	Firmware update cannot be initiated because 'Local Firmware Update' is disabled.	Local FW update must be enabled in the image
8720	Firmware update cannot be initiated because Secure FW update is disabled.	1. Enable secure FW Update in the MEBx OR 2. Use unsecured FW update (-generic)
8721	Firmware update failed due to an internal error.	Tool failed due to an internal error. Please report error

Error Number	Error String	Possible Corrective Actions
8722	Cannot receive the current version from the firmware after update	FW Update is done, Tool failed to get the version after update. Please check with MEInfo and report the error.
8723	No Firmware update is happening	FW Update is not happening, internal error please report the error
8724	Update finished but version mismatch after the update	FW Update finished, Please check the FW version using MEInfo or report the error.
8725	Failed to receive last update status from the firmware	Tool failed to receive last updated status, Please try after a reboot or report the error
8726	Firmware update through TPM is not enabled	FW update through TPM is disabled, Please try to update through MEI or AMT
8727	Firmware update tool failed to get the firmware parameters	FW update tool failed to get firmware parameters from FW. Please report the error.
8728	Firmware update iAMT communication failed, Failed to find certificate '%s' in certificate store	FW update failed due to certificated issues, please use the right certificate or install valid certificate.
	Firmware update iAMT communication failed, Failed to set HTTP certificate options (%d): %s	
	Firmware update iAMT communication failed, Failed to find certificate names	
	Firmware update iAMT communication failed, Failed to open system certificate store (%d): %s	
	Firmware update iAMT communication failed, HTTP request failed: Certificate rejected	
8729	TPM in owned state, owner password is required	TPM ownership has been taken, please use owner password
8730	Firmware update tool failed to encrypt the data	Tool failed to encrypt/decrypt the data, please report the error
8731	Firmware update tool failed to set physical presence	Tool failed to set Software physical presence, please enable HW physical presence.
8732	Firmware update tool failed to send TPM ordinal	Tool failed communicate with TPM, please report the error.

Error Number	Error String	Possible Corrective Actions
8733	Firmware update tool failed to locate TPM device driver	Tool failed to connect to TPM, Please check the driver, enable the TPM driver.
8734	Firmware update iAMT communication failed, WSMAN not supported	WSMAN not supported in this version, please use -eoi switch
8735	TPM require a self test to enable the service, please try after a reboot.	TPM in failed state, may need a reboot to start the service.
8736	Firmware update failed, unable to retrieve TPM Capabilities."	tool failed to receive TPM capabilities, please report the error.
8737	Firmware update tool failed to load XML parser, Please install Microsoft XML DOM	Check/Install Microsoft XML DOM
8738	Firmware update not initiated due to AuthData file [%s] open or read failure	Cannot able to find the Auth data file, please check the file name
8739	Firmware update tool failed read ownerAuth data from AuthData file [%s], Invalid file format	Invalid AuthData file fomat, please check the file is valid or not

B.4 MEManuf Errors

Error Number	Error String	Possible Corrective Actions
9216	Manufacturing test failed due to an internal error	Tool failed due to an internal error. Please report error
9217	Manufacturing test failed due to a flash read/write error	1. Check if SPI device is faulty or missing 2. Check if flash image is corrupted
9218	Manufacturing test failed due to a SMBus error	Check SMBUS device functioning
9219	Manufacturing test failed due to a power down error	Check that other ME operations (like SOL or IDER) are not active

Error Number	Error String	Possible Corrective Actions
9220	Manufacturing test failed due to a BIOS error	One or more of following BIOS tables may not exist: 1) FRU table 2) Media device 3) SMBios memory entry 4) SMBios processor entry 5) SMBios battery entry 6) SMBios Computer system entry 7) SMBios baseboard entry 8) SMBios bios entry 9) ASF table Failure may also be caused by ME-BIOS connectivity failure
9221	Manufacturing full test was not performed	Check counter value. If counter value is 0, then no more full tests can be run
9222	Manufacturing test is in progress- Waiting for system to enter S4 state	Manufacturing test is in progress- Waiting for system to enter S4 state
9223	Manufacturing test failed because of a ME-EC error	Check EC power source. EC power source must be AC
9224	Manufacturing test failed because of a malfunctioning wireless device	Check that wireless device is present and plugged in properly
9225	Manufacturing test failed due to a memory allocation error	Close other applications and retry operation
9226	Kernel not ready to run manufacturing test	Retry operation later
9227	**Print help page**	Check help page for usage info
9231	Manufacturing test failed due to ME - MC error	Check EC power source. EC power source must be AC
9232	Manufacturing test failed because of a malfunctioning wireless device	Check that wireless device is present and plugged in properly

B.5 MEInfo Errors

Error Number	Error String	Possible Corrective Actions
9728	ME Information retrieval failed due to a memory allocation error.	Close other applications and retry
9729	Print Help Page.	Check help page for usage info

B.6 FPT Errors

Error Number	Error String	Possible Corrective Actions
0	Success	
1	Memory allocation error occurred	Make sure there is enough memory in the system
2	Command line arguments not specified	Check the command line arguments supported by using the "-?"
3	Invalid Command line option	
4	Invalid parameter Value	
5	ICH not supported	Check ICH values
6	More than one region was specified. Only one region may be specified at a time	Incorrect Usage of FPT. "-?" For usage
7	Invalid descriptor region	Check descriptor region
8	Region specified does not exist	Check region to be programmed
9	Confirmation is not received from the user to perform operation.	User input required
10	Region is not supported on current ICH	Check ICH values
11	Flash is not blank	Attempt to erase the device again
12	Data verify mismatch found	Reprogram the device
13	Failure. Unexpected error occurred	Please file a sighting
14	Invalid data for Read ID command	
51	Initialization of PMX utilities failed.	
52	Initialization of SPI utilities failed	
101	Access was denied opening the file	Check the permissions for the file
102	Access was denied creating the file	Check the permissions for the file
103	An unknown error occurred while opening the file	Verify the file is not corrupt
104	An unknown error occurred while creating the file	Verify the file is not corrupt
105	Invalid filename	Check the filename
106	File not found	Verify the file exist

Error Number	Error String	Possible Corrective Actions
107	Failed to read the entire file into memory	Check system memory
108	Failed to write the entire flash contents to file	
109	An attempt was made to write beyond the SPI flash area	Check flash device size
110	File already Exists	Delete the file that already exist
111	The file is longer than the flash area to write	Check file size
112	The file is smaller than the flash area to write	
113	Length of image file extends past the flash area	
114	Image file not found	Check filename
115	File does not exist	
116	Error occurred while reading the file	
117	Error occurred while writing to the file	
151	Error occurred while communicating with SPI device	Check SPI device
152	Failed to disable write protection for the BIOS space! Unable to perform erase operation!	Verify BIOS does not have write protection enabled
153	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	
154	Hardware sequencing failed. Make sure that you have access to target flash area	Check descriptor region access settings
155	Software sequencing failed. Make sure that you have access to target flash area	
156	Failed to get information about the installed flash devices	
157	Unable to write data to flash.	
158	The flash device does not support an erase command	Check VSCC values
201	The host CPU does not have write access to the target flash area. To enable write access for this operation you must modify the descriptor settings to give host access to this region	Check descriptor region access settings
202	The host CPU does not have read access to the target flash area. To enable read access for this operation you must modify the descriptor settings to give host access to this region	

Error Number	Error String	Possible Corrective Actions
203	The host CPU does not have erase access to the target flash area. To enable erase access for this operation you must modify the descriptor settings to give host access to this region	
251	General Write failure.	Attempt the command again. If symptom persists file a sighting
252	General Read failure	
253	General Erase failure	
254	The address specified s outside the boundaries of flash area	Check address
255	An attempt was made to read beyond the end of flash memory"	
256	An attempt was made to write beyond the end of flash memory	
257	An attempt was made to erase beyond the end of flash memory	
258	The address of the block to erase is not aligned correctly	
259	Error occurred while writing data less than block erase size	
260	Timeout occurred while reading data	Attempt the command again. If symptom persists file a sighting
301	File not found	Check file location
302	Access was denied opening the file	
303	An unknown error occurred while opening the file	Verify the file is not corrupt
304	Failed to allocate memory for the flash part definition file	Check system memory. Verify the file is not corrupt
305	Failed to read the entire file into memory	
306	Parsing of file failed	
351	Flash descriptor does not have correct signature	Verify file is not corrupt
352	An error occurred reading the flash mapping data	Check SPI device
353	An error occurred reading the flash components data	
354	An error occurred reading the flash region base/limit data	

Error Number	Error String	Possible Corrective Actions
355	An error occurred reading the flash master access data	
356	An error occurred reading the flash descriptor signature	
357	System booted in Non-Descriptor mode, but the flash appears to contain a valid signature	
358	Enable to set iTPM enable/disable bit in the descriptor	
401	The SPI Flash configuration registers are write-protected by the Flash Configuration Lock-Down bit (FLOCKDN). Cannot access the flash. Contact BIOS vendor to unlock this bit, or enable hardware sequencing in descriptor mode	Check with BIOS vendor or SPI programming Guide
402	No SPI flash device could be identified. Please verify if Fparts.txt has support for this part	Verify Fparts.txt contains device supported.
403	Failed to read the device ID from the SPI flash part	Verify Fparts.txt has correct values
404	The SPI Flash configuration registers are write protected by the Flash Configuration Lock-Down bit (FLOCKDN). Cannot access the SPI flash. Contact your BIOS vendor to unlock this bit, or enable hardware sequencing in descriptor mode	Check with BIOS vendor or SPI programming Guide
405	There are no supported SPI flash devices installed. Please check connectivity and orientation of SPI flash device	Verify Fparts.txt has correct values. Check SPI Device
406	The 2 SPI flash devices do not have compatible command sets	Verify both SPI devices on the system are compatible
407	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
451	Invalid Fixed Offset variable name	Check Variable name
452	Invalid Fixed Offset variable Id	Check Variable ID
453	Failed to open param file	Check Filename/location
454	Param file is already opened	Close parameter file
455	Invalid name or Id of FOV	Check variable name or ID
456	Invalid length of FOV value. Check FOV configuration file for correct length	Check length of FOV parameter in parmeter file
457	Password does not match the criteria	Password does not meet strong password requirements
458	Error occurred while reading FOV configuration file	

Error Number	Error String	Possible Corrective Actions
459	Valid LAN MAC address not found in file	Verify MAC address is valid
460	Invalid hash certificate file	Check hash certificate file
461	Valid PID/PPS/Password records are not found in	Check PID/PPS/Password records and ensure that all 3 values exist
462	Invalid Global locked value entered	Globallocked value is incorrect. Value should be 0x0
463	Unable to get master base address from the descriptor	Check file integrity
464	Invalid image. Can not proceed	
465	The setup file header has an illegal UUID	UUID must be valid before ME is turned on
466	The setup file version is unsupported	Check setup file integrity
466	A record encountered that does not contain an entry with the Current MEBx password	Current MEBX password must be supplied
467	The given buffer length is invalid	Check buffer length value
468	The record chunk count cannot contain all of the setup file record data	Setup file number exceeded
469	The setup file header indicates that there are no valid records	Setup file has no valid records. Check setup file integrity
470	The given buffer is invalid	Check buffer value
471	A record entry with an invalid Module ID was encountered	Check record values. Check Setup file Integrity
472	A record was encountered with an invalid record number	
473	The setup file header contains an invalid module ID list	
474	The setup file header contains an invalid byte count	
475	The setup file record id is not RECORD_IDENTIFIER_DATA_RECORD	
476	The list of data record entries is invalid	
477	The CurrentMEBx password is invalid	MEBX password does not meet strong password requirements
478	The NewMEBx password is invalid	
479	The PID is invalid	Check to see if value is valid.

Error Number	Error String	Possible Corrective Actions
480	The PPS is invalid	Check file integrity
481	The PID checksum failed	
482	The PPS checksum failed	
483	The data record is missing a CurrentMEBx password entry	Missing value is required
484	The data record is missing a NewMEBx password entry	
485	The data record is missing a PID entry	
486	The data record is missing a PPS entry	
487	Verification of End Of Manufacturing settings failed	Attempt command again. If problem persists, file a sighting
488	End Of Manufacturing Operation failure - Verification failure on Global Locked settings	Verify global locked bit has not been previously set
489	End Of Manufacturing Operation failure - Verification failure on ME Manuf counter	Verify MEManuf counter is valid
490	End Of Manufacturing Operation failure - Verification failure on Descriptor Lock settings	Verify Descriptor region is present and not corrupt
491	Invalid hexadecimal value entered for the FOV	Check value for FOV supplied
501	An error occurred while communicating with the TPM	Refer to TPM Spec for definition
502	One or more input parameters is bad	
503	A specified output pointer is bad	
504	The specified context handle does not refer to a valid context	
505	A specified output buffer is too small	
506	An error occurred while communicating with the TPM	
507	One or more context parameters is invalid	Refer to TPM Spec for definition
508	The TBS service is not running and could not be started	
509	A new context could not be created because there are too many open contexts	
510	A new virtual resource could not be created because there are too many open virtual resources	

Error Number	Error String	Possible Corrective Actions
511	The TBS service has been started but is not yet running	
512	The physical presence interface is not supported	
513	The command was canceled	Refer to TPM Spec for definition
514	The input or output buffer is too large	
515	A compatible Trusted Platform Module (TPM) Security Device cannot be found on this computer	
516	The TBS service has been disabled	

B.7 TPM Errors

As defined by the TPM main spec, TPM has six types of return codes:

- Success (00000000)
- Fatal errors (00000001 - 000003FF)
- Vendor fatal errors (00000400 - 000007FF)
- Non-fatal errors (00000800 - 00000BFF)
- Vendor non-fatal errors (00000C00 - 00000FFF).

For details on the interpretation of these error codes, refer to the TPM specification or to the documentation provided by the TPM vendor for the specific error codes and their interpretation.

The error codes listed here are for reference only. Error codes and their descriptions may be added or deleted at any time by the TCG. Up-to-date TPM specifications and error codes may be found here:

http://www.trustedcomputinggroup.com

B.7.1 Fatal TPM Errors

Error Number	Error String	Possible Corrective Actions
10001	Authentication failed	Refer to TPM Spec for definition
10002	The index to a PCR, DIR or other register is incorrect	

Error Number	Error String	Possible Corrective Actions
10003	One or more parameter is bad	
10004	An operation completed successfully but the auditing of that operation failed.	
10005	The clear disable flag is set and all clear operations now require physical access	
10006	The TPM is deactivated	
10007	The TPM is disabled	
10008	The target command has been disabled	
10009	The operation failed	
10010	The ordinal was unknown or inconsistent	
10011	The ability to install an owner is disabled	Refer to TPM Spec for definition
10012	The key handle can not be interpreted	
10013	The key handle points to an invalid key	
10014	Unacceptable encryption scheme	
10015	Migration authorization failed	
10016	PCR information could not be interpreted	
10017	No room to load key.	
10018	There is no SRK set	
10019	An encrypted blob is invalid or was not created by this TPM	
10020	There is already an Owner	
10021	The TPM has insufficient internal resources to perform the requested action.	
10022	A random string was too short	
10023	The TPM does not have the space to perform the operation.	
10024	The named PCR value does not match the current PCR value.	Refer to TPM Spec for definition
10025	The paramSize argument to the command has the incorrect value	
10026	There is no existing SHA-1 thread.	
10027	The calculation is unable to proceed because the existing SHA-1 thread has already encountered an error.	
10028	Self-test has failed and the TPM has shutdown.	
10029	The authorization for the second key in a 2 key function failed authorization	
10030	The tag value sent to for a command is invalid	

Error Number	Error String	Possible Corrective Actions
10031	An IO error occurred transmitting information to the TPM	
10032	The encryption process had a problem.	
10033	The decryption process did not complete.	Refer to TPM Spec for definition
10034	An invalid handle was used.	
10035	The TPM does not a EK installed	
10036	The usage of a key is not allowed	
10037	The submitted entity type is not allowed	
10038	The command was received in the wrong sequence relative to TPM_Init and a subsequent TPM_Startup	
10039	Signed data cannot include additional DER information	
10040	The key properties in TPM_KEY_PARMs are not supported by this TPM	
10041	The migration properties of this key are incorrect.	
10042	The signature or encryption scheme for this key is incorrect or not permitted in this situation.	
10043	The size of the data (or blob) parameter is bad or inconsistent with the referenced key	Refer to TPM Spec for definition
10044	A mode parameter is bad, such as capArea or subCapArea for TPM_GetCapability, phsicalPresence parameter for TPM_PhysicalPresence, or	
10045	migrationType for TPM_CreateMigrationBlob.	
10046	Either the physicalPresence or physicalPresenceLock bits have the wrong value	
10047	The TPM cannot perform this version of the capability	
10048	The TPM does not allow for wrapped transport sessions	
10049	TPM audit construction failed and the underlying command was returning a failure code also	
10050	TPM audit construction failed and the underlying command was returning success	Refer to TPM Spec for definition
10051	Attempt to reset a PCR register that does not have the resettable attribute	
10052	Attempt to reset a PCR register that requires locality and locality modifier not part of command transport	
10053	Make identity blob not properly typed	

Error Number	Error String	Possible Corrective Actions
10054	When saving context identified resource type does not match actual resource	
10055	The TPM is attempting to execute a command only available when in FIPS mode	
10056	The command is attempting to use an invalid family ID	
10057	The permission to manipulate the NV storage is not available	
10058	The operation requires a signed command	
10059	Wrong operation to load an NV key	
10060	NV_LoadKey blob requires both owner and blob authorization	Refer to TPM Spec for definition
10061	The NV area is locked and not writable	
10062	The locality is incorrect for the attempted operation	
10063	The NV area is read only and can't be written to	
10064	There is no protection on the write to the NV area	
10065	The family count value does not match	
10066	The NV area has already been written to	
10067	The NV area attributes conflict	
10068	The structure tag and version are invalid or inconsistent	
10069	The key is under control of the TPM Owner and can only be evicted by the TPM Owner.	
10070	The counter handle is incorrect	
10071	The write is not a complete write of the area	Refer to TPM Spec for definition
10072	The gap between saved context counts is too large	
10073	The maximum number of NV writes without an owner has been exceeded	
10074	No operator AuthData value is set	
10075	The resource pointed to by context is not loaded	
10076	The delegate administration is locked	
10077	Attempt to manage a family other then the delegated family	
10078	Delegation table management not enabled	
10079	There was a command executed outside of an exclusive transport session	
10080	Attempt to context save a owner evict controlled key	

Error Number	Error String	Possible Corrective Actions
10081	The DAA command has no resources available to execute the command	Refer to TPM Spec for definition
10082	The consistency check on DAA parameter inputData0 has failed.	
10083	The consistency check on DAA parameter inputData1 has failed.	
10084	The consistency check on DAA_issuerSettings has failed.	
10085	The consistency check on DAA_tpmSpecific has failed.	
10086	The atomic process indicated by the submitted DAA command is not the expected process.	
10087	The issuer's validity check has detected an inconsistency	Refer to TPM Spec for definition
10088	The consistency check on w has failed.	
10089	The handle is incorrect	
10090	Delegation is not correct	
10091	The context blob is invalid	
10092	Too many contexts held by the TPM	
10093	Migration authority signature validation failure	
10094	Migration destination not authenticated	
10095	Migration source incorrect	
10096	Incorrect migration authority	
10097	Attempt to revoke the EK and the EK is not revocable	
10098	Bad signature of CMK ticket	

B.7.2 TPM Non Fatal Errors

Error Number	Error String	Possible Corrective Actions
12048	The TPM is too busy to respond to the command immediately, but the command could be resubmitted at a later time The TPM MAY return TPM_Retry for any command at any time.	Refer to TPM Spec for definition
12049	SelfTestFull has not been run.	
12050	The TPM is currently executing a full self test.	

Error Number	Error String	Possible Corrective Actions
12051	The TPM is defending against dictionary attacks and is in some time-out period.	

§