

# Intel® Management Engine System Tools User Guide

## User Guide

*May 2009*

*Revision 1.71*

**Intel Confidential**



INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL® PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN INTEL'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER, AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

UNLESS OTHERWISE AGREED IN WRITING BY INTEL, THE INTEL PRODUCTS ARE NOT DESIGNED NOR INTENDED FOR ANY APPLICATION IN WHICH THE FAILURE OF THE INTEL PRODUCT COULD CREATE A SITUATION WHERE PERSONAL INJURY OR DEATH MAY OCCUR.

Intel may make changes to specifications and product descriptions at any time, without notice. Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined." Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. The information here is subject to change without notice. Do not finalize a design with this information.

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

Contact your local Intel sales office or your distributor to obtain the latest specifications and before placing your product order.

This document contains information on products in the design phase of development.

All products, platforms, dates, and figures specified are preliminary based on current expectations, and are subject to change without notice. All dates specified are target dates, are provided for planning purposes only and are subject to change.

This document contains information on products in the design phase of development. Do not finalize a design with this information. Revised information will be published when the product is available. Verify with your local sales office that you have the latest datasheet before finalizing a design.

Intel® Active Management Technology requires the computer system to have an Intel® AMT-enabled chipset, network hardware and software, as well as connection with a power source and a corporate network connection. Setup requires configuration by the purchaser and may require scripting with the management console or further integration into existing security frameworks to enable certain functionality. It may also require modifications of implementation of new business processes. With regard to notebooks, Intel AMT may not be available or certain capabilities may be limited over a host OS-based VPN or when connecting wirelessly, on battery power, sleeping, hibernating or powered off. For more information, see www.intel.com/technology/platform-technology/intel-amt/

Code names featured are used internally within Intel to identify products that are in development and not yet publicly announced for release. Customers, licensees and other third parties are not authorized by Intel to use code names in advertising, promotion or marketing of any product or services and any such use of Intel's internal code names is at the sole risk of the user.

Intel, Intel vPro, and the Intel logo are trademarks of Intel Corporation in the U.S. and other countries.

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

Copyright © 2007-2008, Intel Corporation. All rights reserved.





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


1 Introduction ..... 11
1.1 Terminology ..... 11
1.2 Reference Documents ..... 12
2 Preface ..... 13
2.1 Overview ..... 13
2.2 Manufacturing Line Validation Tools ..... 13
2.3 Image Editing Tools ..... 13
2.4 Requirements ..... 14
3 Flash Image Tool (FITC) ..... 15
3.1 System Requirements ..... 15
3.2 Flash Image Details ..... 15
3.2.1 Flash Space Allocation ..... 16
3.3 Required Files ..... 16
3.4 Configuration Files ..... 17
3.4.1 Creating a new configuration ..... 17
3.4.2 Opening an existing configuration ..... 17
3.4.3 Saving a configuration ..... 17
3.5 Environment Variables ..... 17
3.6 Build Settings ..... 19
3.7 Modifying the Flash Descriptor Region (FDR) ..... 20
3.7.1 Descriptor Region length ..... 21
3.7.2 Setting the number and size of the flash components ..... 21
3.7.3 Region access control ..... 23
3.8 MCH Strap 0 ..... 28
3.8.1 ME boot from flash ..... 28
3.9 ICH Strap Section ..... 29
3.10 VSCC Table ..... 29
3.10.1 Adding a new table ..... 29
3.10.2 Removing an existing table ..... 31
3.11 Modifying the ME Region ..... 31
3.11.1 Setting the ME Region binary file ..... 31
3.11.2 Enabling/disabling the ME Region ..... 32
3.12 Modifying the GbE (LAN) Region ..... 33
3.12.1 Setting the GbE Region binary file ..... 33
3.12.2 Enabling/disabling the GbE Region ..... 33
3.13 Modifying the PDR Region ..... 34
3.13.1 Setting the PDR Region binary file ..... 34
3.13.2 Enabling/disabling the PDR Region ..... 35
3.14 Modifying the BIOS Region ..... 35
3.14.1 Setting the BIOS Region binary file ..... 36
3.14.2 Enabling/disabling the BIOS Region ..... 36
3.15 Configuration Tab ..... 37
3.15.1 ME Section ..... 38




3.15.2 AMT Section ..... 39
3.15.3 PET Section ..... 40
3.15.4 Power Packages Section ..... 41
3.15.5 Features Supported ..... 41
3.15.6 Setup and Configuration Section ..... 42
3.15.7 Intel® TPM Section ..... 43
3.15.8 Intel® Remote Connectivity Service Section ..... 45
3.16 Building a Flash Image ..... 47
3.17 Change the region order on the SPI device ..... 47
3.18 Decomposing an Existing Flash Image ..... 48
3.19 Command Line Interface ..... 48
3.20 Example – Decomposing and Image and Extracting Parmeters ..... 51

4 Flash Programming Tool (FPT) ..... 53
4.1 System Requirements ..... 53
4.2 Flash Image Details ..... 54
4.3 Windows* Required Files ..... 54
4.4 DOS Required Files ..... 55
4.5 Where to find dos4gw.exe ..... 55
4.6 Programming the Flash Device ..... 55
4.7 Programming fixed offset variables ..... 56
4.8 Usage ..... 56
4.9 Update Hash Certificate through FOV ..... 59
4.10 fparts.txt File ..... 61
4.11 End of Manufacture ..... 62
4.12 Examples ..... 62
4.12.1 Example 1 ..... 62
4.12.2 Example 2 ..... 63
4.12.3 Example 3 ..... 63
4.12.4 Example 4 ..... 63
4.12.5 Example 5 ..... 64
4.12.6 Example 6 ..... 64
4.12.7 Example 7 ..... 65

5 MEManuf and MEManufWin ..... 68
5.1 Requirements ..... 69
5.2 Windows PE requirements ..... 70
5.3 Firmware Counter ..... 70
5.4 Complete Test ..... 70
5.4.1 First invocation ..... 70
5.4.2 Second invocation ..... 71
5.4.3 Last invocation ..... 71
5.5 Partial Test ..... 71
5.6 Intel® TPM Impact on MEManuf ..... 72
5.7 Usage ..... 72
5.8 Examples ..... 72
5.8.1 Example 1 ..... 73
5.8.2 Example 2 ..... 73
5.8.3 Example 3 ..... 73

6 MEInfo ..... 74




6.1 Requirements ..... 74
6.2 Windows* PE requirements ..... 74
6.3 Usage ..... 75
6.4 Examples ..... 78
6.4.1 Example 1 ..... 78
6.4.2 Example 2 ..... 79
6.4.3 Example 3 ..... 79
6.4.4 Example 4 ..... 79
6.4.5 Example 5 ..... 80
6.4.6 Example 6 ..... 80
6.4.7 Example 7 ..... 80

7 Firmware Update (FWUpdLcl) ..... 83
7.1 Requirements ..... 83
7.2 Non-Secure Dos Requirements ..... 84
7.3 Non-Secure Windows Requirements ..... 84
7.4 Secure Windows Requirements ..... 84
7.5 Windows* PE Requirements ..... 84
7.6 Enabling and Disabling Local Firmware Update ..... 84
7.7 Usage DOS Version ..... 86
7.8 Usage Windows* Version ..... 86
7.9 Examples ..... 88
7.9.1 Example 1 ..... 88
7.9.2 Example 2 ..... 88

Appendix A Fixed offset Variables ..... 90

Appendix B Error Codes ..... 94



# Figures


Figure 1. Firmware Image Components ..... 15
Figure 2. Environment Variables Dialog ..... 19
Figure 3. Build Settings Dialog ..... 20
Figure 4. Descriptor Region length ..... 21
Figure 5: Editable Flash Image Region List ..... 21
Figure 6: Descriptor Region Map Options ..... 22
Figure 7: Descriptor Region Fast Read Support Options ..... 22
Figure 8: Descriptor Region Component Section Options ..... 23
Figure 9: Descriptor Region Master Access Section Location ..... 25
Figure 10: Descriptor Region Master Access Section Options ..... 26
Figure 11: Message Determining Whether Firmware Image Contains ROM Bypass Section ..... 28
Figure 12: ICH Strap 0 Section ..... 29
Figure 13: Add New VSCC table entry ..... 30
Figure 14: Add VSCC table entry ..... 30
Figure 15: VSCC Table Entry ..... 31
Figure 16: Remove VSCC table entry ..... 31
Figure 17: Enabling the ME Region ..... 32
Figure 18: GbE Region Options ..... 33
Figure 19: Disabling the GbE Region ..... 34
Figure 20: PDR Region Options ..... 35
Figure 21: Disabling the PDR Region ..... 35
Figure 22: BIOS Region Options ..... 36
Figure 23: Disabling the BIOS Region ..... 36
Figure 24: Configuration Tab ..... 37
Figure 25: Configuration Tab ..... 37
Figure 26: ME Section ..... 38
Figure 27: AMT Section ..... 40
Figure 28: PET Section ..... 41
Figure 29: Power Packages Section ..... 41
Figure 30: Features Supported Section ..... 42
Figure 31: Setup and Configuration Section ..... 42
Figure 32: Intel® TPM Section ..... 43
Figure 33: Intel® Remote Connectivity Service Section ..... 45
Figure 34: Region Order ..... 48
Figure 35: Firmware Image Components ..... 54
Figure 36: Raw Hash value from certificate file ..... 60
Figure 37: Sample Hash BIN file ..... 60
Figure 38: McCreary Chipset Layout ..... 69


# Tables


Table 1. Tools Summary ..... 14
Table 2. Recommended Read/Write Values ..... 27
Table 3. Recommended Read/Write Values ..... 27
Table 4. Firmware Override Update Variables ..... 39
Table 5. Intel® TPM Permanent Flags ..... 44
Table 6. Dictionary Attack Flags ..... 44
Table 7 Remote Connectivity Service Parameters ..... 46



Table 8. List of components for which version information must be retrieved ..... 76
Table 9. Firmware Override Update Variables ..... 85



# Revision History

| Revision Number | Description | Revision Date |
|---|---|---|
| 0.3 | Pre Alpha | 11/16/2007 |
| 0.5 | Alpha | 12/05/2007 |
| 0.8 | Alpha 1 | 03/06/2008 |
| 0.81 | Alpha 2 | 04/05/2008 |
| 0.92 | Alpha 3 | 05/06/2008 |
| 0.96 | Beta | 06/03/2008 |
| 0.97 | PC | 06/30/2008 |
| 1.1 | PV | 07/17/2008 |
| 1.4 | Hotfix release- Added auto verify feature in FPT and ME Configuration paramters | 08/13/2008 |
| 1.5 | Updates to FWUpdate options | 08/18/2008 |
| 1.6 | Update to FPT FOV area | 09/22/2008 |
| 1.7 | Update HaltCFG option<br>Fix some errors in FOV list | 04/01/2009 |
| 1.71 | Add Win 7 support | 05/22/2009 |

§



# 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, ME Firmware, GBE and PDR Region |
| FW | Firmware, specifically firmware executing on the ME. |
| IDE-R | IDE Redirection |
| Intel® AMT | Intel® Active Management Technology. |
| Intel® QST | Intel® Quiet System Technology. Embedded hardware and firmware solution that allows for algorithmic relationship between system cooling fans and temperature monitors so as to reduce noise without losing thermal efficiency. |
| Intel® Remote Connectivity Service | Allows OEMs, managed service providers (MSP) and IT Outsourcers to connect with end user systems over the public internet and remotely manage enabled systems regardless of system state |
| Intel® TPM | Integrated TPM—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 5.0 | 24571 |
| All trusted computing literature | http://www.trustedcomputing.com |
| Intel® TPM Tools User Guide | Release Kit |
| ICH10 EDS | 373635 |
| ICH SPI Programming Guide | 355834 |

§



# 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 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 (XP SP1/2, PE, and 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.

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 ME Devices | X | |
| MEInfo and MEInfoWin | Firmware Aliveness—outputs certain 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 McCreary 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 (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.

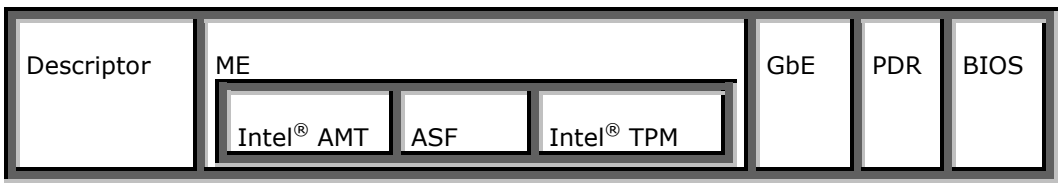
## 3.1 System Requirements

The FITC will run on Windows* XP or Windows Vista* (32 bit). It is not necessary for the tool to run on an 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 ME applications, such as Intel® AMT technology, ASF 2.0, Intel® TPM and Intel® Quiet System Technology (Intel® QST).
- 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 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.

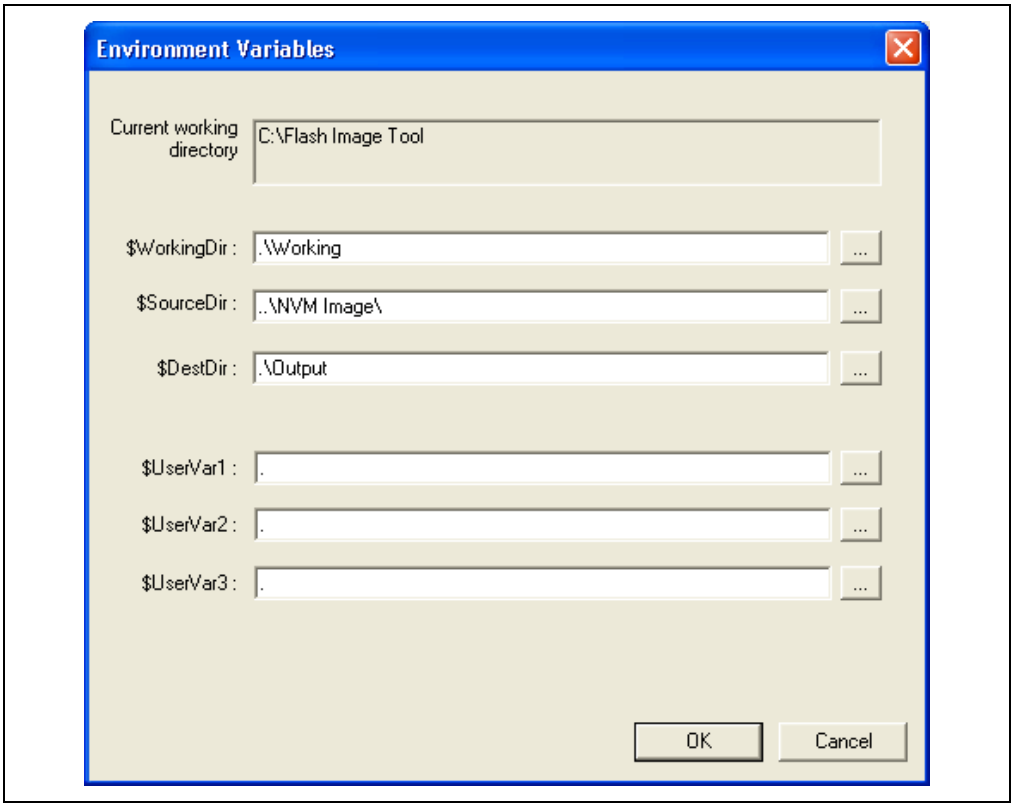
   A set of environment variables are provided to make the image configuration files more portable. By making all of the paths in the configuration relative to environment variables, the configuration is not tied to a particular root directory structure. Each user can set their environment variables appropriate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

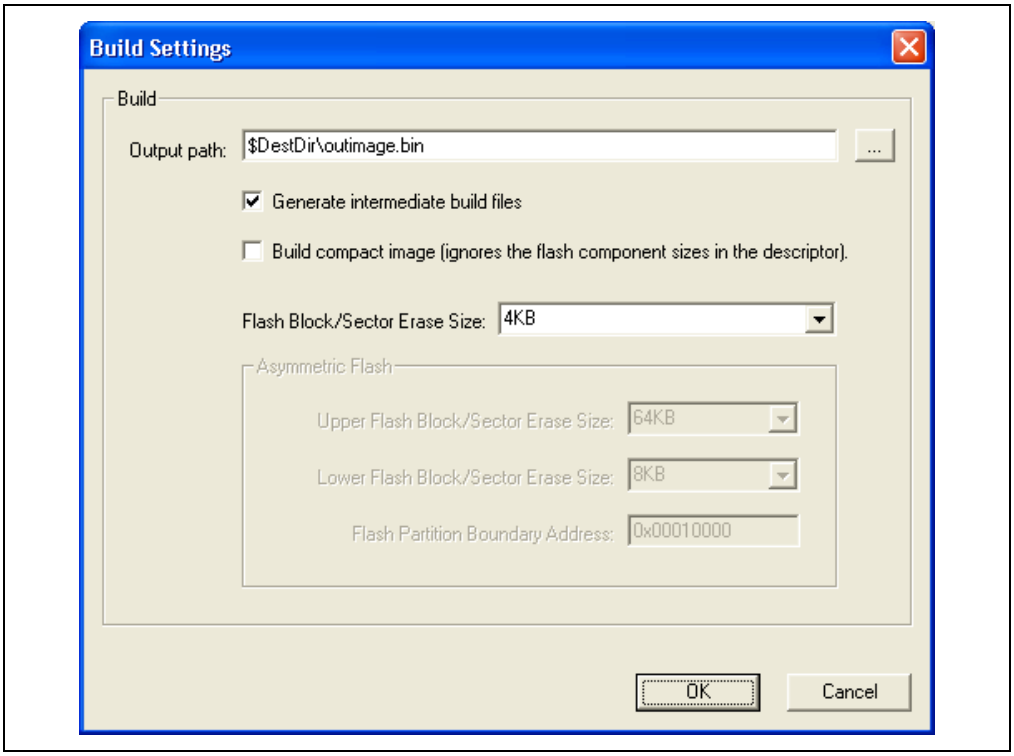
An option is provided (the **Generate intermediate build files** checkbox) that causes the application to generate separate (intermediate) binary files for each region, in addition to the final image file (see Figure 3). These files will be located in the int folder located inside the specified output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

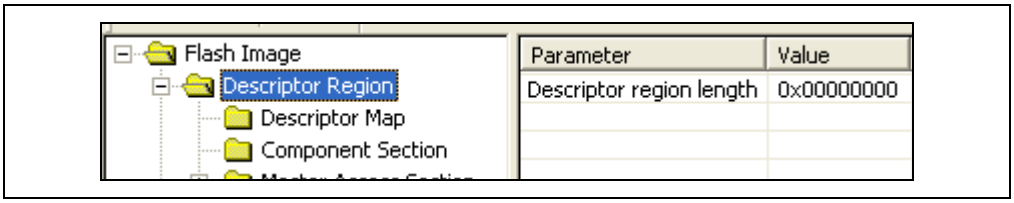
## 3.7 Modifying the Flash Descriptor Region (FDR)

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**

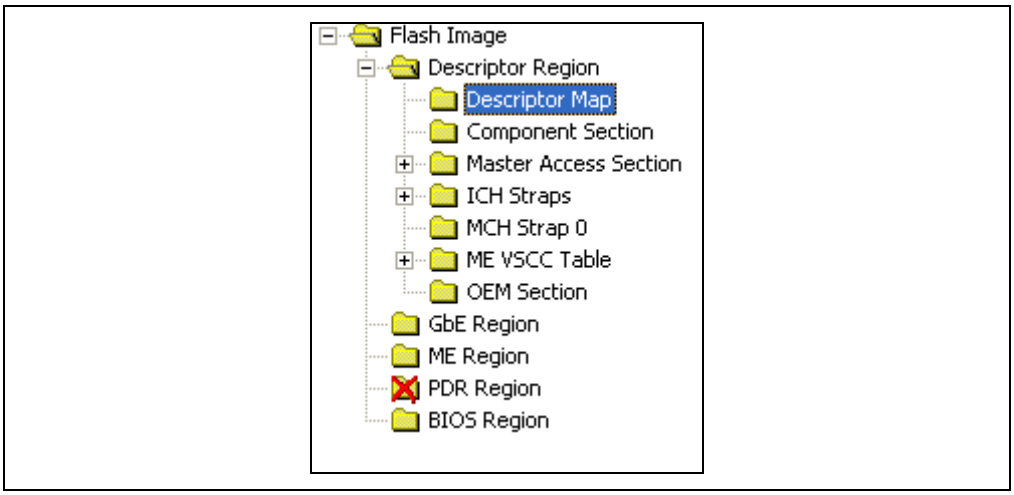



### 3.7.2 Setting the number and size of the flash components

To set the number of flash components:

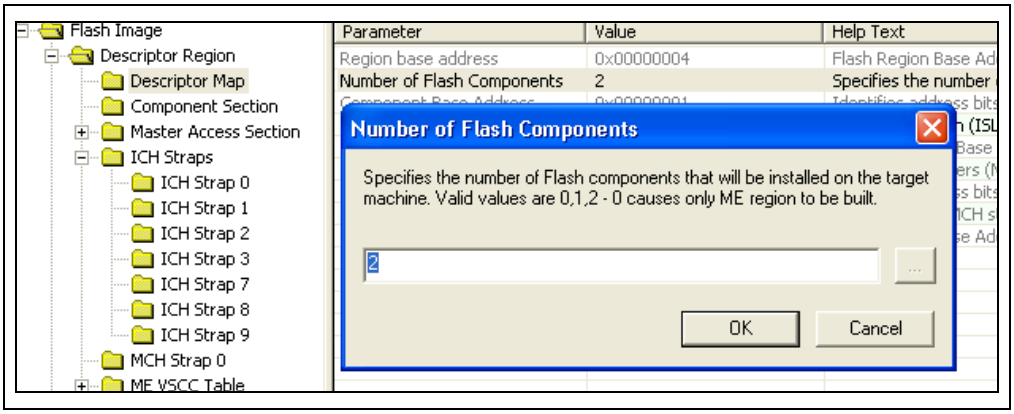
1. Expand the Descriptor Region node of the tree in the left pane of the main window.
2. Select **Descriptor Map** (see Figure 5). All of the parameters for the Descriptor Map section will appear in the list in the right pane of the main window.

**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**

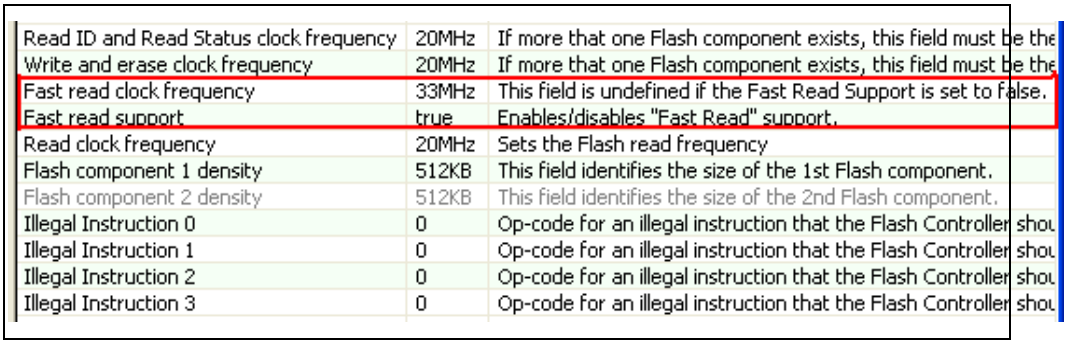



Some SPI flash devices support both standard and fast read speeds. For the ICH10 to support the faster read speeds, the fast read clock frequency must be set to 33 MHz and fast read support must be set to true.

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:**

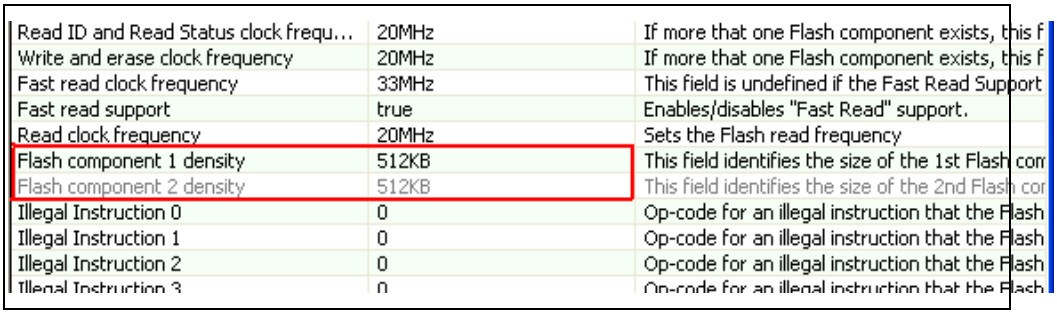
1. Expand the Descriptor Region tree node and select the **Component Section** node. The parameters Flash component 1 density and Flash component 2 density specify the size of each flash component.



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**



| | | |
|---|---|---|
| Read ID and Read Status clock frequ... | 20MHz | If more that one Flash component exists, this f |
| Write and erase clock frequency | 20MHz | If more that one Flash component exists, this f |
| Fast read clock frequency | 33MHz | This field is undefined if the Fast Read Support |
| Fast read support | true | Enables/disables "Fast Read" support. |
| Read clock frequency | 20MHz | Sets the Flash read frequency |
| Flash component 1 density | 512KB | This field identifies the size of the 1st Flash com |
| Flash component 2 density | 512KB | This field identifies the size of the 2nd Flash com |
| Illegal Instruction 0 | 0 | Op-code for an illegal instruction that the Flash |
| Illegal Instruction 1 | 0 | Op-code for an illegal instruction that the Flash |
| Illegal Instruction 2 | 0 | Op-code for an illegal instruction that the Flash |
| Illegal Instruction 3 | 0 | Op-code for an illegal instruction that the Flash |

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

**Table 5. Region Access Control Table**

| | Regions that can be accessed | | | | |
|---|---|---|---|---|---|
| **Region to Grant Access** | **PDR** | **ME** | **GBE** | **BIOS** | **Descriptor** |
| **ME** | None / Read / Write | None / Read / Write | Write only. 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

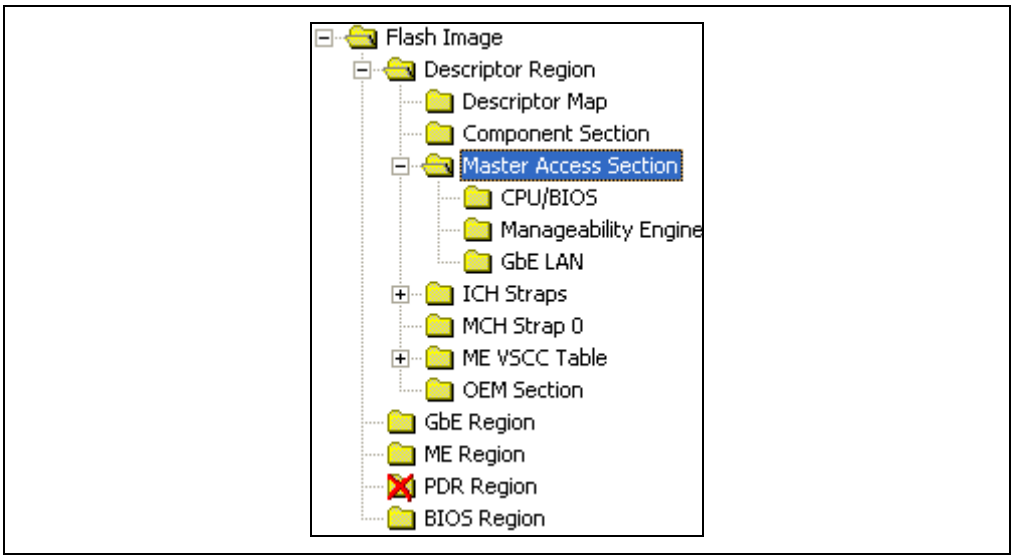
In hexadecimal:

Read Access—0x 0E

Write Access—0x 06

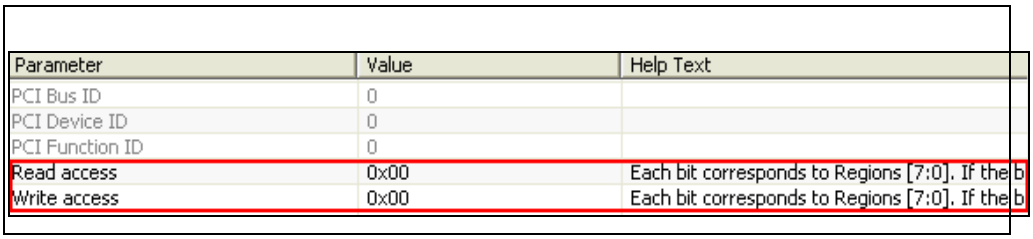
In the FITC these access values can be set by selecting the Descriptor Region tree node and selecting CPU/BIOS under the Master Access Section (see Figure 9).

**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 2. 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 3. Recommended Read/Write Values**

| | ME | GbE | BIOS |
|---|---|---|---|
| Read | 0b 0000 1101 = 0x0d | 0b 0000 1000 = 0x08 | 0b 0001 1011 = 0x1b<br><br>If the PDR region is not present the value |

| | | | |
|---|---|---|---|
| | | | should be:<br>0b 0000 1011 = 0x0b |
| Write | 0b 0000 1100 = 0x0c | 0b 0000 1000 = 0x08 | 0b 0001 1010 = 0x1a<br><br>If the PDR region is not present the value<br>0b 0000 1010 = 0x0a |

## 3.8 MCH Strap 0

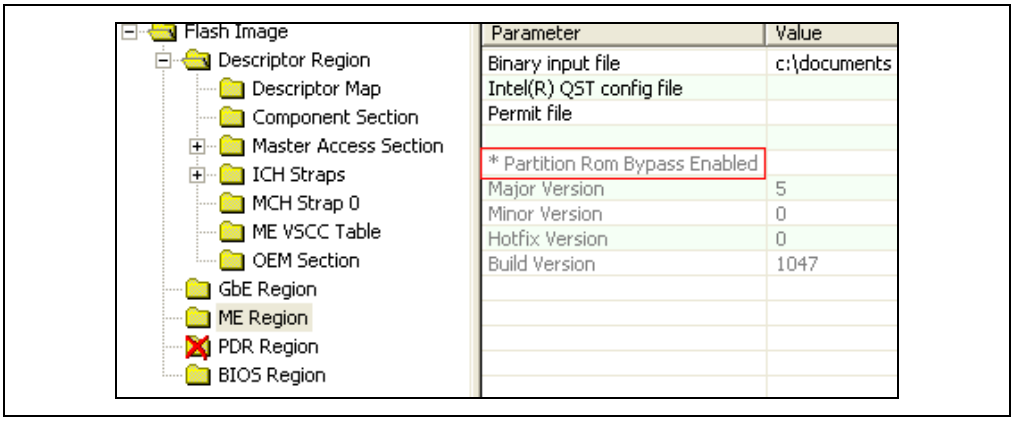
This section contains the variable to edit the address at which the Intel® ME begins to read.

### 3.8.1 ME boot from flash

If the firmware used contains the boot from flash option, FITC will automatically select the 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**



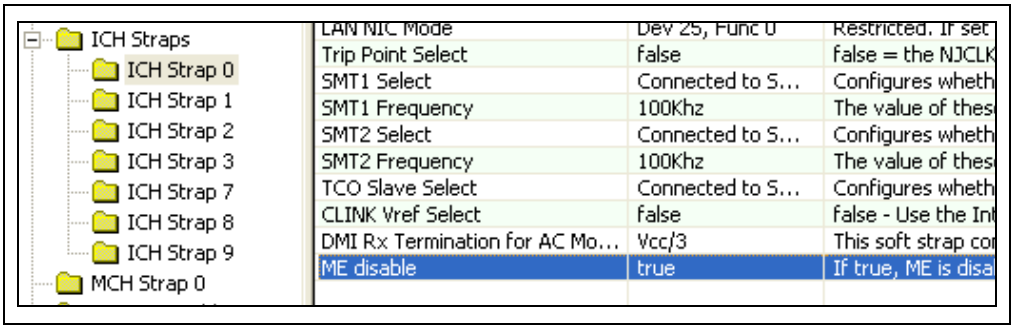


## 3.9 ICH Strap Section

This section contains variables for configuring LAN specific components and PCI express configuration details. Please see the ICH EDS for more details on this section.

Please be certain the ME disable bit is set correctly under the ICH Strap 0 section

**Figure 12: ICH Strap 0 Section**




## 3.10 VSCC Table

This section is used to store information regarding the flash devices used on the system and is REQUIRED by the firmware. If the information in this section is incorrect, the 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.

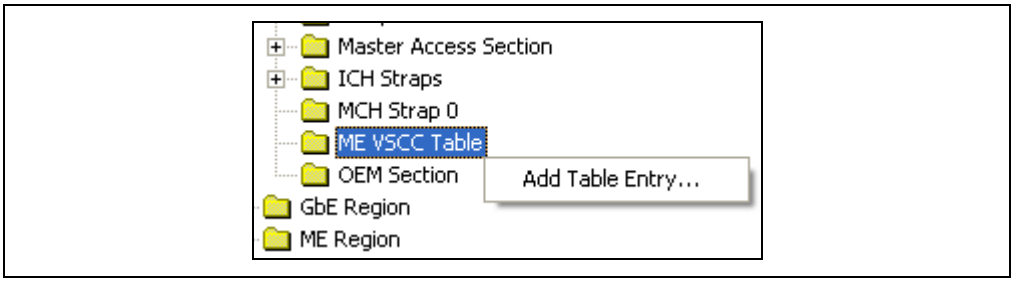
Additional information on VSCC values can be found in the ICH SPI Programming Guide (Document #355834). Failure to set these bits correctly will cause pre-mature wear out of the SPI flash device as well as erratic ME behavior.

### 3.10.1 Adding a new table

**To add a new table:**

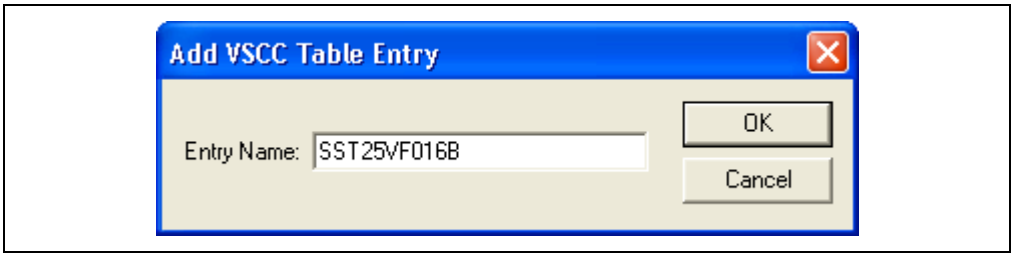
1. Right-click on **VSCC table**.
2. Select **Add Table Entry...**

**Figure 13: 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 14: 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 15: VSCC Table Entry**

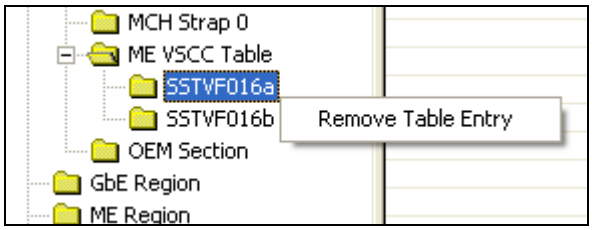
| Parameter | Value | Help Text |
|---|---|---|
| Vendor ID | 0xBF | The vendor specific byte of the JEDEC ID. |
| Device ID 0 | 0x25 | The first device specific byte of the JEDEC ID. |
| Device ID 1 | 0x41 | The second device specific byte of the JEDEC ID. |
| VSCC register value | 0x00002009 | The 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 16: Remove VSCC table entry**




## 3.11 Modifying the ME Region

The 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 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 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.

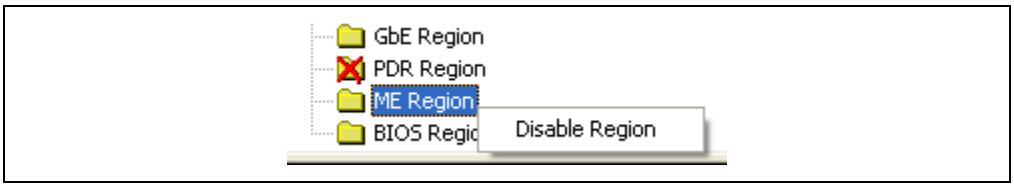
Full SPI Image size – BIOS Region size – Descriptor Region size = GBE Region Size

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 17: Enabling the 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

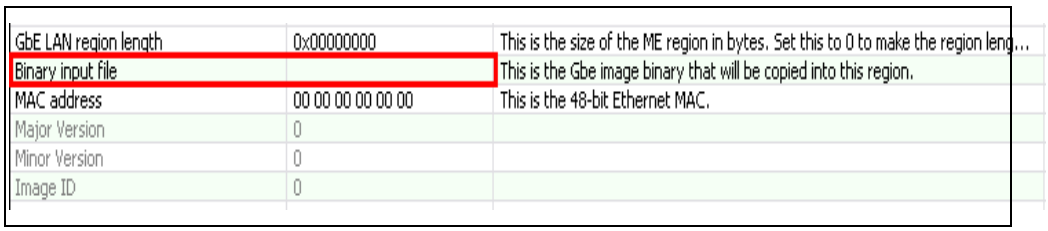
**To set the GbE Region binary file:**

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 18: GbE Region Options**




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.

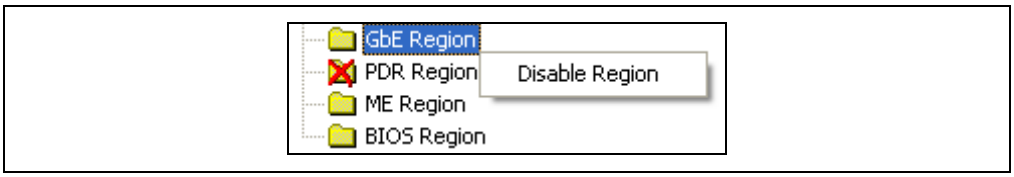
### 3.12.2 Enabling/disabling the GbE Region

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 19: 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

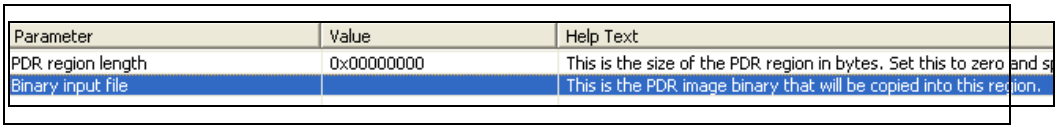
**To set the PDR region binary file:**

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 20: PDR Region Options**




## 3.13.2 Enabling/disabling the PDR Region

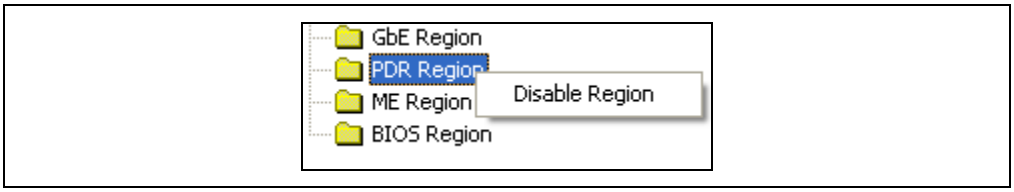
The PDR Region can be excluded from the flash image by disabling it in the FITC.

**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 21: Disabling the PDR Region**




**To enable the PDR Region:**

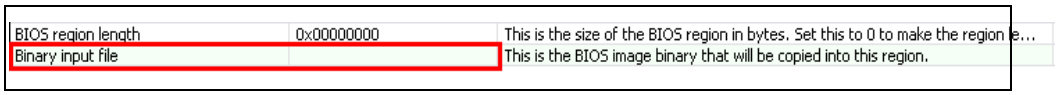
1. Right-click on the PDR Region tree node.
2. Select **Enable Region** from the pop-up menu.

# 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 22: 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.

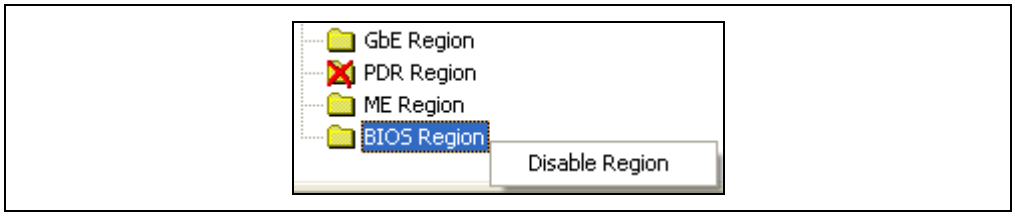
### 3.14.2 Enabling/disabling the BIOS Region

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 23: 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.

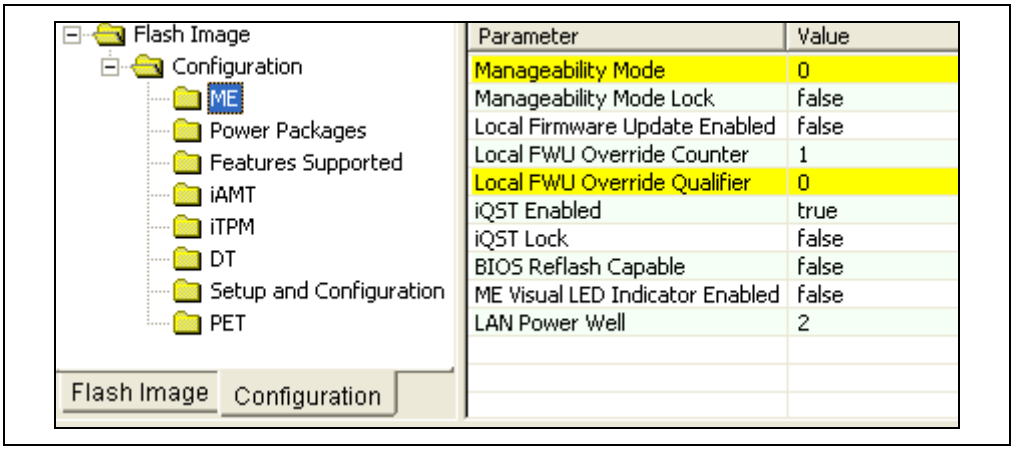


## 3.15 Configuration Tab

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

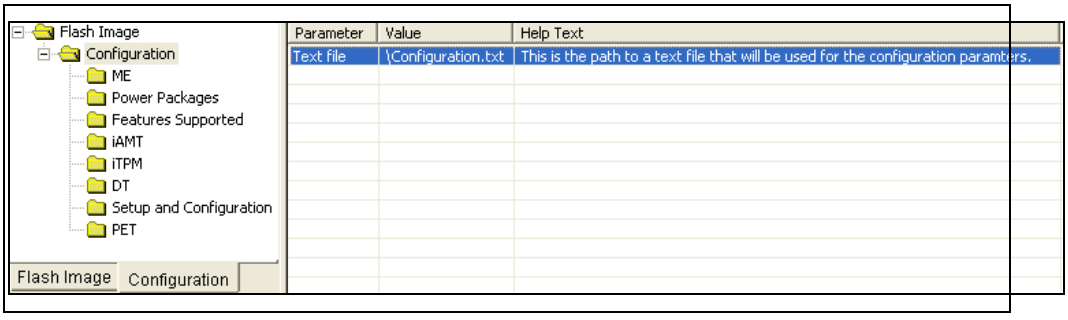
If any of the parameters are changed from the Intel recommended value the offending row will be highlighted yellow. No errors will be reported. The highlighted yellow is designed to draw attention to these values were ensure these parameters were set correctly

**Figure 24: Configuration Tab**




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 25: Configuration Tab**



### 3.15.1 ME Section

The 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 26.

**Figure 26: ME Section**

| Parameter | Value | Help Text |
|---|---|---|
| Manageability Mode | 1 | 0 = None, 1 = Intel® AMT, 2 = ASF2 |
| Manageability Mode Lock | false | false = Unlocked, true = Locked |
| Local Firmware Update Enabled | false | false = Firmware update blocked, true = Firmware update allowed. |
| Local FWU Override Counter | -1 | Set FW Update Counter. Valid values are -1 to 254 |
| Local FWU Override Qualifier | 2 | 0 = Always; allow override,1 = Never; disqualify override,2 = Restricted;... |
| Flash Descriptor Override Pin-Strap Ignore | false | 0 = FW Update Strap Eanbled, 1 = FW Update Strap Blocked |
| Intel® iQST Enabled | true | false = Intel® iQST Disable, true = Enable |
| Intel® iQST Lock | false | false = Intel® iQST Unlocked, true = Locked |
| BIOS Reflash Capable | false | false = Not Capable, true = Capable |
| ME Visual LED Indicator Enabled | false | false = Disable, true = Enable |
| LAN Power Well | 2 | 0 = Disable, 1 = Sus Well, 2 = ME Well |
| Debug Si Features | 0x00000000 | Allows OEM control to enable FW features to assist with debug of the platform |
| Prod Si Features | 0x00000000 | Allows OEM control to enable FW features to assist with production of the plat. |
| ME Flash Protection Override Enabled | false | false = Disable, true = Enable |

#### 3.15.1.1 Flash Descriptor Override Pin Strap Ignore

This bit determines if ME will be disabled when the manufacturing override jumper set.

**False** – ME will enters a disabled state to safely program the full SPI device if the manufacturing mode jumper is set

**True** – ME will NOT enter a disabled if the manufacturing mode jumper is set.

#### 3.15.1.2 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 ME BIOS Extension (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 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 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 4. 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 |

### 3.15.1.3 Debug Si Features/ Prod Si Features

This option will output the firmware status register to a specified bus address. This is a debug feature that should not be enabled in the production image.

The D*ebug Si feature* parameter should be used for non-production hardware. The P*rod Si feature* parameter should be used for production hardware.
Both of these parameters should not be set in the final production image.

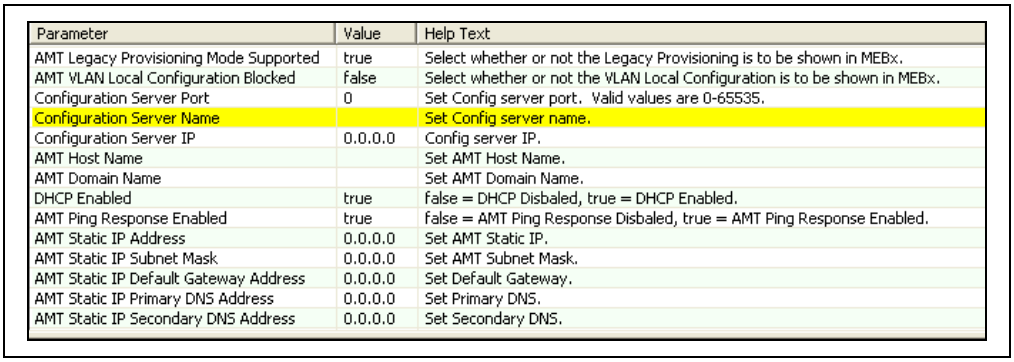
**1 –** Firmware status register to the default bus address of the MDDD device inserted in Channel 0 memory slot

**Bus Address** – Firmware status register will output to the specified bus address. Bits 7:1 are used for the Bus Address. Bit 0 is used for the enable bit with 1 denoting enabled and 0 is disabled.

## 3.15.2 AMT Section

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

**Figure 27: AMT Section**



| Parameter | Value | Help Text |
|---|---|---|
| AMT Legacy Provisioning Mode Supported | true | Select whether or not the Legacy Provisioning is to be shown in MEBx. |
| AMT VLAN Local Configuration Blocked | false | Select whether or not the VLAN Local Configuration is to be shown in MEBx. |
| Configuration Server Port | 0 | Set Config server port. Valid values are 0-65535. |
| Configuration Server Name | | Set Config server name. |
| Configuration Server IP | 0.0.0.0 | Config server IP. |
| AMT Host Name | | Set AMT Host Name. |
| AMT Domain Name | | Set AMT Domain Name. |
| DHCP Enabled | true | false = DHCP Disbaled, true = DHCP Enabled. |
| AMT Ping Response Enabled | true | false = AMT Ping Response Disbaled, true = AMT Ping Response Enabled. |
| AMT Static IP Address | 0.0.0.0 | Set AMT Static IP. |
| AMT Static IP Subnet Mask | 0.0.0.0 | Set AMT Subnet Mask. |
| AMT Static IP Default Gateway Address | 0.0.0.0 | Set Default Gateway. |
| AMT Static IP Primary DNS Address | 0.0.0.0 | Set Primary DNS. |
| AMT Static IP Secondary DNS Address | 0.0.0.0 | Set Secondary DNS. |

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.

**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 MEBX.

**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 MEBX.

**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 MEBX

**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 MEBX.

**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 MEBx. To reduce the amount of end user configuration time, this value should be set to a reasonable value.

### 3.15.3 PET Section

The PET section allows the OEM to specify custom field data.



**Figure 28: PET Section**

| Parameter | Value | Help Text |
|---|---|---|
| PET Language Code | 0x0 | Select PET Language code. |
| PET OEM Custom Fields 00-15 | 00-00-00-00-00-00-... | Enter OEM Custom Field data. |
| PET OEM Custom Fields 16-31 | 00-00-00-00-00-00-... | Enter OEM Custom Field data. |
| PET OEM Custom Fields 32-47 | 00-00-00-00-00-00-... | Enter OEM Custom Field data. |
| PET OEM Custom Fields 48-63 | 00-00-00-00-00-00-... | Enter OEM Custom Field data. |
| PET OEM Custom Fields Length | 0 | Select PET OEM Custom Fields Length. |
| PET Community String | public | Select PET Community String. |

## 3.15.4 Power Packages Section

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

**Figure 29: Power Packages Section**

| Parameter | Value |
|---|---|
| Power Package 1 Supported (Desktop: ON in S0) | true |
| Power Package 2 Supported (Desktop: ON in S0, S3) | true |
| Power Package 3 Supported (Desktop: ON in S0, S3, S4-5) | true |
| Power Package 4 Supported (Desktop: ON in S0, ME WoL in S3) | true |
| Power Package 5 Supported (Desktop: ON in S0, ME WoL in S3, S4-5) | true |
| Power Package 6 Supported (Desktop: ON in S0, S3, S4-5, OFF After Power Loss) | true |
| Power Package 7 Supported (Desktop: ON in S0, ME WoL in S3, S4-5, OFF After Power Loss) | true |
| Default Power Package | 1 |

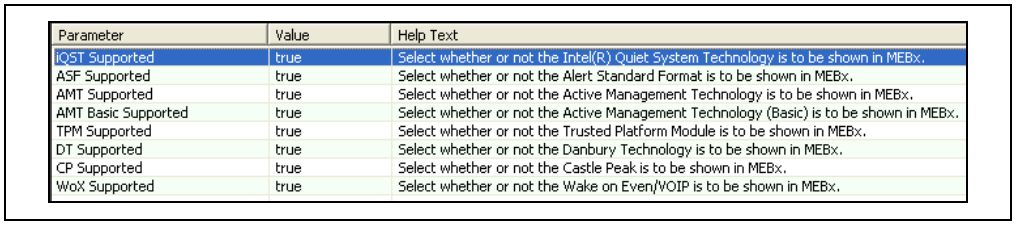
If the Power Package Supported value is set to false, that specific power package cannot be selected and will not be visible to the end user.

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 Features Supported

The Features supported section will determine which features are supported by the system. If a system does not meet the minimum hardware requirements, no error message will be given when programming the image

**Figure 30: Features Supported Section**



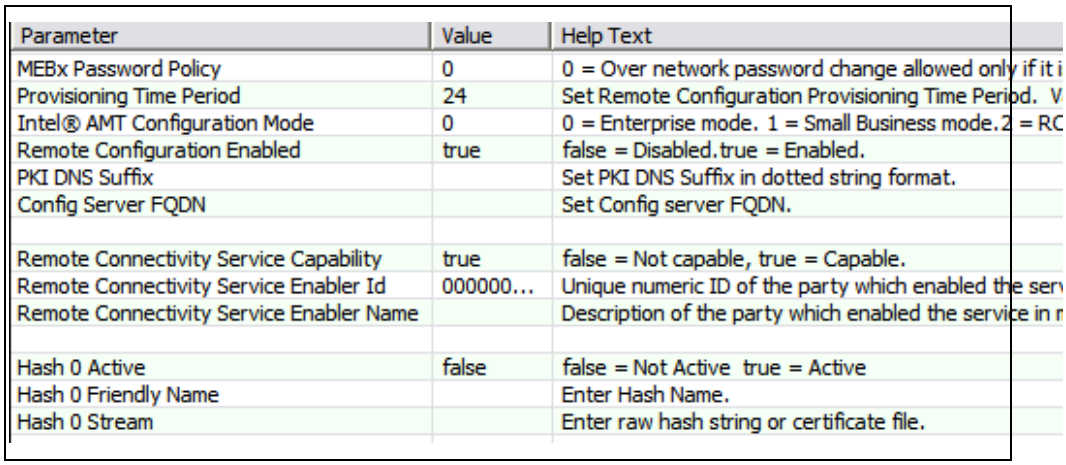
| Parameter | Value | Help Text |
|---|---|---|
| IQST Supported | true | Select whether or not the Intel(R) Quiet System Technology is to be shown in MEBx. |
| ASF Supported | true | Select whether or not the Alert Standard Format is to be shown in MEBx. |
| AMT Supported | true | Select whether or not the Active Management Technology is to be shown in MEBx. |
| AMT Basic Supported | true | Select whether or not the Active Management Technology (Basic) is to be shown in MEBx. |
| TPM Supported | true | Select whether or not the Trusted Platform Module is to be shown in MEBx. |
| DT Supported | true | Select whether or not the Danbury Technology is to be shown in MEBx. |
| CP Supported | true | Select whether or not the Castle Peak is to be shown in MEBx. |
| WoX Supported | true | Select whether or not the Wake on Even/VOIP is to be shown in MEBx. |

Any parameter that is set to false will not be supported.

## 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 31: Setup and Configuration Section**



| Parameter | Value | Help Text |
|---|---|---|
| MEBx Password Policy | 0 | 0 = Over network password change allowed only if it i |
| Provisioning Time Period | 24 | Set Remote Configuration Provisioning Time Period. V |
| Intel® AMT Configuration Mode | 0 | 0 = Enterprise mode. 1 = Small Business mode.2 = RC |
| Remote Configuration Enabled | true | false = Disabled.true = Enabled. |
| PKI DNS Suffix | | Set PKI DNS Suffix in dotted string format. |
| Config Server FQDN | | Set Config server FQDN. |
| | | |
| Remote Connectivity Service Capability | true | false = Not capable, true = Capable. |
| Remote Connectivity Service Enabler Id | 000000... | Unique numeric ID of the party which enabled the serv |
| Remote Connectivity Service Enabler Name | | Description of the party which enabled the service in r |
| | | |
| Hash 0 Active | false | false = Not Active true = Active |
| Hash 0 Friendly Name | | Enter Hash Name. |
| Hash 0 Stream | | Enter raw hash string or certificate file. |

The **Provisioning Time Period** specifies the amount of time (in hours) allowed to configure the Intel® AMT device. This time period begins when the 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 Connectivity Services allows OEMs, managed service providers (MSP) and IT Outsourcers to connect with enabled systems over the public internet and remotely manage them regardless of system state. This mode can be set instead of Small Businesses or Enterprise mode.

**Remote Connectivity Service capability** specifies whether the platform allows configuration of Remote Connectivity Service provisioning mode or not. When the value is "true", the platform has three possible AMT Provisioning Modes – Enterprise, Small Business and Remote Connectivity Service. When the value is "false" the platform has two possible AMT Provisioning Modes – Enterprise and Small Business.

**AMT Configuration Mode –** when the mode is **Remote Connectivity Service,** the platform is automatically configured (no additional user configuration required) to interact with Intel Remote Connectivity infrastructure in the Internet.

**Remote Connectivity Service enabler ID** specifies the unique ID of the party (e.g. OEM) which enabled the platform for Remote Connectivity Service mode.

**Remote Connectivity Service enabler Description** specifies the textual description (string) of the party (e.g. OEM) which enabled the platform for Remote Connectivity Service mode.

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 Intel® TPM Section

**Figure 32: Intel® TPM Section**

| Parameter | Value | Help Text |
|---|---|---|
| iTPM FIPS 140-2 | false | true = The iTPM operates in FIPS mode.false = The iTPM does NOT |
| iTPM Physical Presence Lifetim... | false | true = The state of either physicalPresenceHWEnable or physicalPr |
| iTPM Physical Presence Hardw... | false | true = Enables the hardware signal indicating physical presence. fa |
| iTPM Physical Presence Comma... | true | true = Enables the command indicating physical presence. false = |
| Authentication Fail Threshold | 10 | Number of failed authentication attempts which will trigger lockout b |
| Initial Lockout Time | 240 | Set duration in seconds of first lockout period, between 16-65535. |
| Lockout Increase Factor | 2 | Set Factor by which lockout period is multiplied with every additiona |
| Fade Out Time | 3600 | Every time this period (in seconds) passes with no authentication fa |

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

**Table 5. Intel® TPM Permanent Flags**

| Bit | Flag | Description | Default |
|---|---|---|---|
| 0 | FIPS | TRUE: This TPM operates in FIPS mode<br><br>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<br><br>TRUE: The state of either physicalPresenceHWEnable or physicalPresenceCMDEnable MUST NOT be changed for the life of the TPM | FALSE |
| 2 | Physical Presence HW Enable | FALSE: Disable the hardware signal indicating physical presence<br><br>TRUE: Enables the hardware signal indicating physical presence | FALSE |
| 3 | Physical Presence CMD Enable | FALSE: Disable the command indicating physical presence<br><br>TRUE: Enables the command indicating physical presence | TRUE |

**Table 6. 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 |

## 3.15.8 Intel® Remote Connectivity Service Section

**Intel® Remote Connectivity Service** is a platform extension service which allows PC OEMs, managed service providers (MSP) and IT Outsourcers to reach or "connect" with end user PCs over the public internet and remotely manage them regardless of system state.

Intel® Remote Connectivity Service code and configuration data are embedded into the ME region. The parameters mentioned below are also available in the FOV section. Please see Appendix A. Setting these parameters correctly greatly eases the effort required by the end user to enable Intel® Remote Connectivity Service.

**Remote Connectivity Service capability** - Specifies if the platform allows configuration of Remote Connectivity Service provisioning mode or not. When the value is "**true**", the platform will have Remote Connectivity service as an option for Intel® AMT Provisioning Mode.

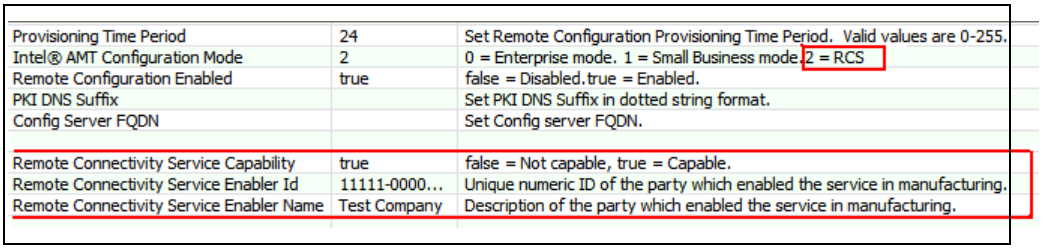
When the value is "**false**" Remote Connectivity Service code is completely disabled in the Firmware and the platform has two possible AMT Provisioning Modes – Enterprise and Small Business.

***Note:** when "Remote Connectivity Service capability" = false, it is not supported to set AMT Configuration Mode =2 ("Remote Connectivity Service") in the FITc tool

**Remote Connectivity Service enabler ID** parameter**-** specifies the unique ID of the party (e.g. OEM) which enabled the platform for Remote Connectivity Service mode.

**Remote Connectivity Service enabler name** parameter **-** specifies the textual description (string) of the party (e.g. OEM) which enabled the platform for Remote Connectivity Service mode.

**Figure 33: Intel® Remote Connectivity Service Section**



Once the AMT configuration mode is set to Remote Connectivity Service no additional user configuration is required to interact with Intel® Remote Connectivity infrastructure in the Internet environment

***Note: When "Remote Connectivity Service capability" = false, the following options are NOT supported values:**

1. Set "AMT Configuration Mode" =2 ("Remote Connectivity Service") in the FITc tool or FOVs.
2. Change the default value of "Remote Connectivity Service enabler ID" in the FITc tool or FOVS.
3. Change the default value of "Remote Connectivity Service enabler name" in the FITc tool or FOVS.

FPT nor FITC will report errors for the incorrect values stated above.

**Table 7 Remote Connectivity Service Parameters**

| Parameter Name | Description | Default value |
|---|---|---|
| **Remote Connectivity Service capability** | Determines if the system supports Remote Connectivity Service<br><br>RCS supported – 0x01<br>RCS NOT Supported – 0x00 | False |
| **AMT Configuration mode** | Sets the default configuration mode for the system<br><br>Enterprise - 0x00<br>Small Business - 0x01<br>Remote Connectivity Service mode - 0x02 | 0x00 |
| **Remote Connectivity Service enabler ID** | Specifies the unique ID of the party (e.g. OEM) which enabled the platform for Remote Connectivity Service mode<br><br>All 16 byte values between 0x00h - 0xFFFFFFFh.<br>0x00h and 0xFFFFFFFFh are not valid values. | None |



| | | |
|---|---|---|
| **Remote Connectivity Service enabler description** | Specifies the textual description (string) of the party (e.g. OEM) which enabled the platform for Remote Connectivity Service mode.<br><br>Textual string. Limited to 60 bytes | None |

## 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 McCreary flash image. The following will be produced when building an image:

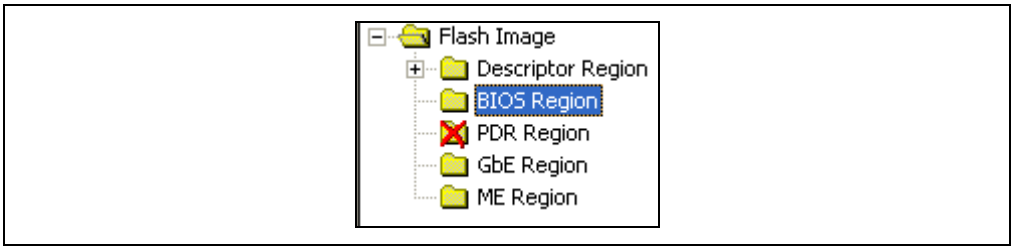
- Binary file representing the image
- Text file detailing the various regions in the image
- Optional set of intermediate files (see Section 3.6).
- And, if two flash components are specified, multiple binary files containing the image broken up according to the flash component sizes.

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 34: 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 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>]
                [/rombypass <true|false>]
                [/me_disable]
                [/me <file>]
                [/gbe <file>]
                [/bios <file>]
                [/pdr <file>]
                [/configparms <file>]
                [/platform <EL_ICH10|EL_ICH10D>]
                [/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.

/rombypass—overrides rombyass settings in the XML file.

/me_disable—overrides the ME disable flag in the descriptor.

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

/gbe <file>—overrides the binary source file for the GbE Region with the specified binary file.

/bios <file>—overrides the binary source file for the BIOS Region with the specified binary file.

/configparms <file>—overrides the Configuration Parameters in the XML file with the values in the file specified





/platform <EL_ICH10 | EL_ICH10D>—Sets the platform setting for the image file to be created

/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 Example – Decomposing and 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.

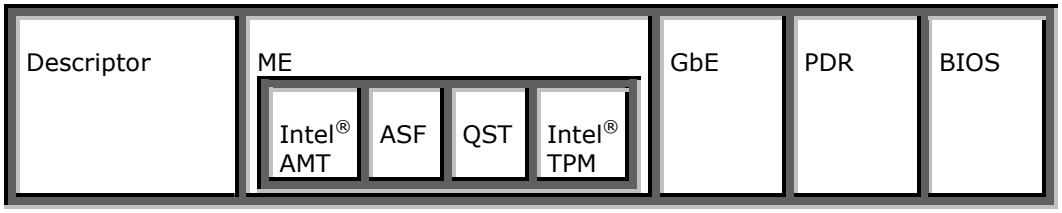
One possible flow for using the FPT is:

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 35: 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.

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, Intel® TPM and Intel® Quiet System Technology (QST).

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 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 ME use one of the following options:**

1. Disable the ME via the BIOS or the 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.

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 initial 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 0x01. 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 –o <Text File>** will create a text file that will allow the user to update multiple fixed offset variables. The variables will be displayed in the following format:

```
[Parmeter name]
Enabled=0xff
Value =
```

In the text file created, variables that NOT enabled (enabled=0xff) will not be modified. Only variables that ARE enabled (enabled=0x1) will be modified.

**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 A

## 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]
                [/Intel® TPM <Enabled/Disable>]
                [/fovs]
                [/ex]
                [/u]
                [/o]
                [/in]
                [/n]
                [/id]
                [/v]
                [/MacFile]
                [/Lock]
                [/DumpLock]
                [/PskFile]
                [/CloseMnf]
                [/GReset]
```

/? 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. After the program process is completed, FPT will automatically verify (/verify) the image was programmed correctly. The verification will always run after using the /f option.

/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. This option is automatically executed after performing the /f option.

/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

/Intel® TPM <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 3.7.3 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 5.9 End of Manufacture for more details

/GReset—Initiate a global reset. A global reset is sufficient to allow the ME to use the Fixed offset variables. Please check with your BIOS to ensure global resets are permitted.

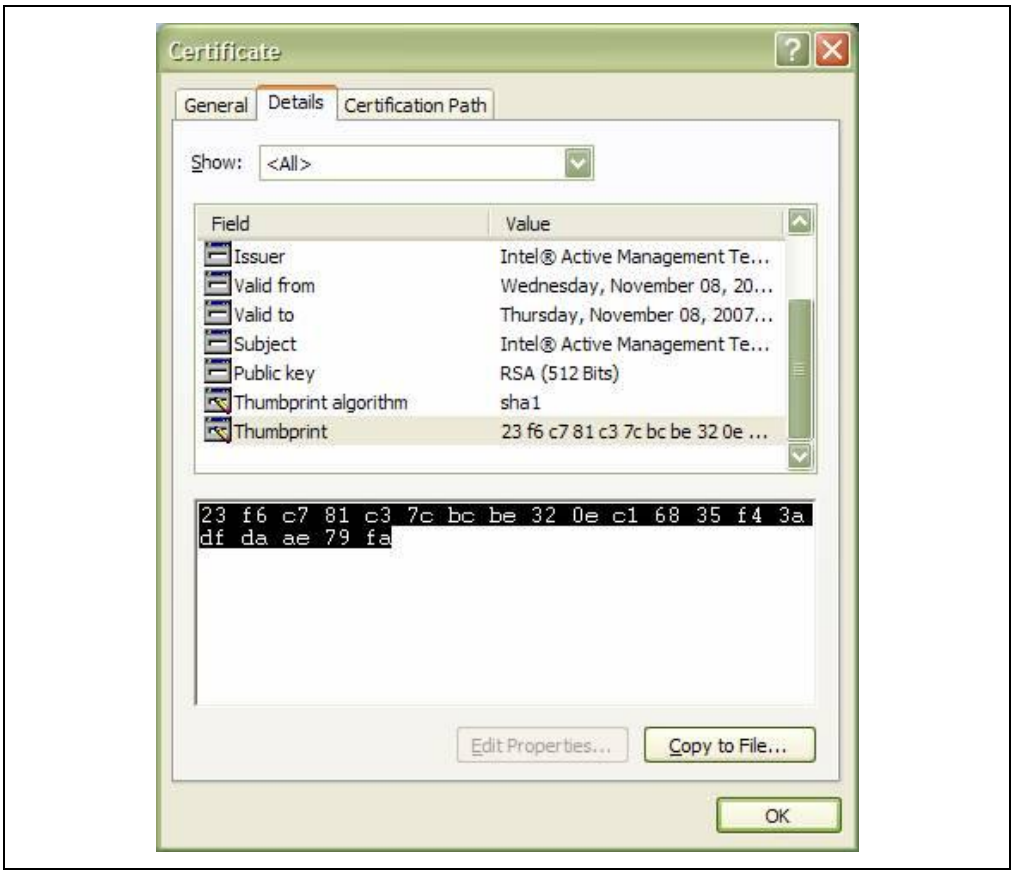
## 4.9 Update Hash Certificate through FOV

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.

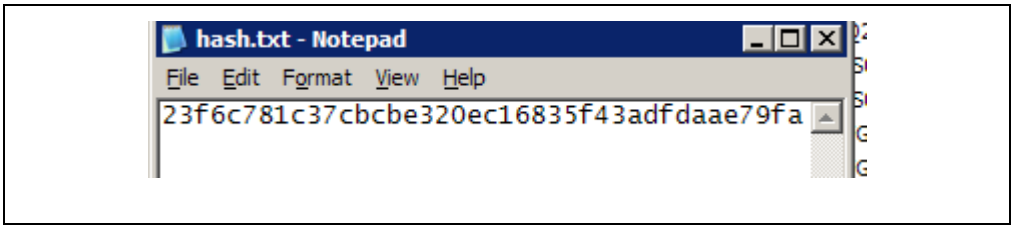
1) User must copy the raw hash values from a valid certificate file

**Figure 36: Raw Hash value 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.txt. This hash file must exist in the same folder as the sampleparam.txt file mentioned in step 3.

**Figure 37: Sample Hash TXT 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 *(fpt.exe /ex /o sampleparm.txt)*. 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.txt**

**[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. The TPM must be locked using another tool.

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 used with the ***-closemnf*** flag, the FPT provides a single command that performs all of these operations.

## 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 – Dump ME Region to the screen

```
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 –Program Specific Address

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

This usage loads 2KB of the binary file image.bin to the address 0x100.

### 4.12.3 Example 3 –Program Specific Region

```
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 – Dump Specific Region

```
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 – Display SPI information

```
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 – Verify Image with errors

```
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. The /y option will proceed with the comparison without warning.

### 4.12.7 Example 7 –Verify Image successfully

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

### 4.12.8 Example 8 – Program FOV parameter

```
C:\ fpt.exe /u /n "AMTConfigMode" /v 0x03

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)

Updating software sequencing.
Reading region information from flash descriptor
Reading FOV configuration file "fptcfg.ini"
Updating variable [AMTConfigMode]..
```

This usage updates the default configuration mode. In this example the Configuration mode was set to **Remote Connectivity Service.** This action is only supported if Remote Connectivity service is supported on the system. FPT will not report dependency errors. Please be sure that the values selected are valid.

### 4.12.9 Example 9 – Update MAC addresses using MAC file

```
C:\ fpt.exe /u /n hostmacadd /macfile mac.txt
Flash Programming Tool. Version X.X.X.XXXX
Copyright (c) Intel Corporation. 2007-2008
Southbridge: ICH10
Reading file "fparts.txt" into memory...
Initializing SPI utilities
Reading HSFSTS register... Flash Descriptor: Valid

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

Flash program registers are locked! HSFSTS[15] (FLOCKDN)
Checking to see if the correct commands are in the OPMENU and PREOP
registers.
Using software sequencing.
Reading region information from flash descriptor.
Reading FOV configuration file "fptcfg.ini"
Reading Input File...[mac.txt]
Using MAC address "001122334455" from "mac.txt"
Updating variable [HostMacAdd]..
```

This usage updates the GBE MAC address using the "-macfile" option. The MAC file is a list of MAC addresses. Each MAC address should be entered on a different line. After a MAC address has been used, a timestamp will indicate when the MAC address was used. A sample MAC.txt file is seen below:

```
001122334455 : 01:17:33  07/13/2008
554433221100
0011aabbccdd
```

When the all address in the MAC file have been used does not have any more valid MAC addresses, FPT will return an error stating all MAC addresses have been used

```
Reading FOV configuration file "fptcfg.ini"
```



```
Reading Input File...[mac.txt]
Error 461: Valid LAN MAC address not found in file "mac.txt".
Error 461: Valid LAN MAC address not found in file "mac.txt".
FPT.exe exited with return code: 461
```

§





# 5 MEManuf and MEManufWin

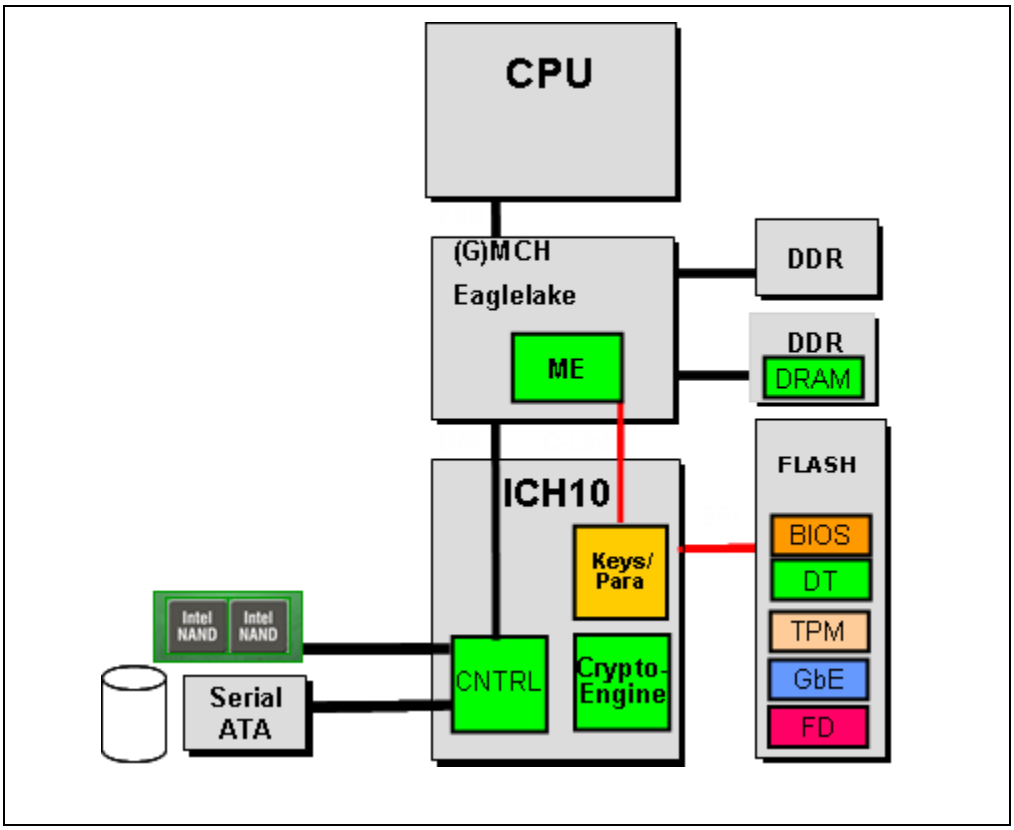
MEManuf validates ME functionality (verifies that all its components have been assembled together correctly) on the manufacturing line. The tool accomplishes this by invoking the test program embedded in the ME firmware. The test covers the following features:

- 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 38: McCreary 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 run on a Windows (XP SP1/2, XP32/64, Vista 32/64 or 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 Intel® TPM+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 ME firmware is alive or not. Both tools perform the same test, query the Intel® ME firmware including 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 will run on Windows (Windows PE, XP 32/64, XP SP1/2, Vista32/64, Windows 7 ). MEInfo and MEInfoWin are also command-line executables.

The Intel® ME Interface 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.

To verify the SKU information the value to be entered should match the SKU number as mentioned below.

Digital Office: 0x00001D7E
- Features Supported:
  - Alert Standard Format (ASF)
  - Intel® Quiet System Technology (QST)
  - Intel® Active Management Technology
  - Intel® Standard Manageability
  - Intel® Trusted Platform Module (TPM)
  - Intel® Remote Wake Technology
  - Intel® Remote PC Assist Technology

Base Corporate : 0x00000806
- Features Supported:
  - Alert Standard Format (ASF)
  - Intel® Quiet System Technology (QST)

Digital Home: 0x00000806
- Features Supported:
  - Alert Standard Format (ASF)
  - Intel® Quiet System Technology (QST)
  - Remote Wake Technology
  - Remote Assist Technology

Base Consumer: 0x00000D06
- Features Supported:
  - Alert Standard Format (ASF)
  - Intel® Quiet System Technology (QST)

-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

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 | AMT SKU 'Manageability Mode' | | | ASF | 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 |
| MEBx Version | X | X | X | X | X | A version string |
| AMT Netstack version | X | X | X | --- | --- | A version string |
| AMT version | X | X | X | --- | --- | A version string |
| 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 |
| 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 |



| Component | AMT SKU 'Manageability Mode' | | | ASF | TPM | Field value |
|---|---|---|---|---|---|---|
| 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 | A MAC address |
| 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 |
| 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 |
| AMT Configuration mode | X | --- | --- | --- | --- | Small Business/Enterprise/ Remote Connectivity |
| Remote Connectivity Service Capability | X | --- | --- | --- | --- | True/False |
| Remote Connectivity Service Enabler ID | X | --- | --- | --- | --- | A number |
| Remote Connectivity Service Enabler name | X | --- | --- | --- | --- | String Description |
| TPM fuses (MCH/ICH/soft strap MCH/ soft strap ICH) | X | X | X | X | X | Enabled/ Disabled |
| TPM Vendor ID | --- | --- | --- | --- | X | A version string |
| TPM SPEC Version | --- | --- | --- | --- | X | A version string |
| TPM FW Version | --- | --- | --- | --- | X | A version string |
| TPM FW Build | --- | --- | --- | --- | X | A number |
| TPM State | --- | --- | --- | --- | X | Operational / Failed state |
| TPM Operational Mode | --- | --- | --- | --- | X | Active, Enabled, Owned |
| Intel® TPM – FIPS 140-2 | --- | --- | --- | --- | X | False/True |
| Intel® TPM - Physical presence life time lock flag | --- | --- | --- | --- | X | False/True |

| Component | AMT SKU 'Manageability Mode' | | | ASF | TPM | Field value |
|---|---|---|---|---|---|---|
| Intel® TPM - Physical presence command enabled flag | --- | --- | --- | --- | X | False/True |
| Intel® TPM - 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.

```
Copyright (C) 2005-08, Intel Corporation.  All Rights Reserved

Intel® MEInfo Win Version: 5.0.0.XXX

BIOS Version              XXXXXX.XXX

Intel(R) AMT code versions:
        Flash:          5.0.0                   Netstack:   5.0.0
        Apps:           5.0.0                   Intel(R) AMT:  5.X.X
        SKU :           1406
                        ASF2
                        Intel(R) AMT
                        Intel Standard Manageability
                        Intel(R) Anti-Theft Technology
                        Intel(R) QST
                        Intel(R) TPM
                        Intel(R) Remote PC Assist Technology
                        Intel(R) Remote Wake Technology

        VendorID        8086
        Build Number :  XXXX

Intel(R) AMT Mode:      Intel(R) AMT 5.0
```



```
UNS Version:              5.0.X.XXXX
LMS Version:              5.0.X.XXXX
MEI Driver version:       5.0.X.XXXX
Link status:                  Link down
Cryptography fuse:            Enabled
Flash protection:             Enabled
Last ME reset reason:         Power up
Configuration state:          Not started
BIOS boot State:              Pre Boot
MAC Address:                  00-11-22-33-44-55
FWU Override Counter:         Never
FWU Override Qualifier:       Always
MEBx Version:                 5.0.X.X
FT Version:                   5.0
FT Build Number:              XXXX
Manageability Mode:           Intel(R)AMT
Local FWUpdate:               Disabled
Secure FWUpdate:              Disabled
Remote Connectivity Service Capability                False
Remote Connectivity Service enabler ID                0
Remote Connectivity Service enabler name
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 "Intel® AMT"

Intel® AMT:               5.0.0
```

## 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 "Setup and Configuration" –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 QST 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® MEInfo version: 5.0.0.XXXX
Intel(R) AMT code versions:
      Flash:       5.0.0     Netstack :          5.0.0
      Apps:        5.0.0     Intel(R) AMT:       5.X.X
      SKU :        1406
                   ASF2
                   Intel(R) AMT
                   Intel Standard Manageability
                   Intel(R) Anti-Theft Technology
                   Intel(R) QST
                   Intel(R) TPM
                   Intel(R) Remote PC Assist Technology
                   Intel(R) Remote Wake Technology
```

### 6.4.6 Example 6

When MEInfo displays the SKU information of the system, MEInfo will display the output on multiple lines.  The SKU information will display all features that are supported on the system

C:\ MEInfo –feat SKU

```
Copyright (C) 2005-07 Intel Corporation. All Rights Reserved.

SKU:    0x0001D7E
        Alert Standard Format (ASF)
        Intel® Quiet System Technology (QST)
        Intel® Active Management Technology
        Intel® Standard Manageability
        Intel® Trusted Platform Module (TPM)
        Intel® Remote Wake Technology
        Intel® Remote PC Assist Technology
```

### 6.4.7 Example 7

To compare the SKU information MEInfo only compares the SKU value.  MEInfo will not use the feature names as correct arguments.

```
C:\ MEInfo –feat SKU –value 0x00000806
```



```
SKU:    Error - actual value is: 0x00001D7E
        Alert Standard Format (ASF)
        Intel® Quiet System Technology (QST)
        Intel® Active Management Technology
        Intel® Standard Manageability
        Intel® Trusted Platform Module (TPM)
        Intel® Remote Wake Technology
        Intel® Remote PC Assist Technology
```

§





# 7 Firmware Update (FWUpdLcl)

FW Update 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, Base_Corporate_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 depending on the flash device on the system.

-HaltRCFG option will not do any update of the firmware. It only stops the network traffic generated by ME for remote configuration process.

## 7.1 Requirements

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

Firmware can only be updated when the system is in an S0 state. Firmware updates are NOT supported in the S3/S4/S5 state.

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

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

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

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 (XP SP1/SP2, XP32/64, PE or Vista32/64, Windows 7).





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

ME Firmware Local Update must be enabled in the MEBx.

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

The ME Interface driver must be installed.

## 7.4 Secure Windows Requirements

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

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

The Intel® AMT LMS must be installed.

## 7.5 Windows* PE Requirements

The 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 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 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 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 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> -forceReset
```

**– forceReset—**Restarts the system after the firmware update process is complete. Any unsaved data will be lost.

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

```
FWUpdLcl.exe -HaltRCFG
```

**-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.

This command is allowed only **during** remote configuration.
i.e. if configuration was never started (e.g. PRE) or was already completed (POST) or even if the provisioning period was expired, **the command will fail with an error.**

## 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 of the firmware. If TLS is used this must be the name of the TLS certificate in the firmware.

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

**– 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.

**– TPM <pass>—**Initiates a firmware update through the TPM device where <pass> is one of the two options mentioned below

**– key <Owner Key>—**The owner key is the endorsement key of the TPM device in the owned state

**– msf <File_name>—**File name is the Windows Vista generated Authdata file if TPM is in the owned state

**– forceReset—**Restarts the system after the firmware update process is complete. Any unsaved data will be lost.

```
FWUpdLcl.exe -HaltRCFG
```

**-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.

This command is allowed only **during** remote configuration.
i.e. if configuration was never started (e.g. PRE) or was already completed (POST) or even if the provisioning period was expired, **the command will fail with an error.**

## 7.9 Examples

### 7.9.1 Example 1

```
FWUpdLcl.exe Base_Corporate_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 Base_Corporate_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.

If AMT is not in a remote configuration process , i.e. if configuration was never started (e.g. PRE) or was already completed (POST) or even if the provisioning period was expired. Tools will report an error.

§

# Appendix A Fixed offset Variables

This appendix only covers fixed offset variables that are directly available to FPT and FPTW. A complete list of fixed offset variables can be found in the Firmware Variable Structures for Intel® Management Engine (Document number 24571). 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<br>0x02 – Enable Lock |
| 0x0002 | MEStateControl | Determines the default state of the ME | 0x00 – Disabled<br>0x01 – Enabled |
| 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 |
| 0x0004 | MEPwrFeature | This option will prevent the user from changing the power package selected | 0x00 – Allow user to modify power package<br>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<br>0x01 – Firmware update enabled |
| 0x0007 | FWUpdOverideQualifier | Please see firmware update section for more information on this parameter | 0x00 – Always<br>0x01 – Never<br>0x02 - Restricted |



| FOV ID | Variable Name | Description | Value Type |
|---|---|---|---|
| 0x0008 | FWUpdOverrideCounter | Please see firmware update section for more information on | 0x00 – Never<br>0<N<255 – Allow N boot cycles<br>0xFF - Always |
| 0x0009 | PasswordFlag | Password manager allows Intel® AMT to set the MEBX password | 0x01 – Intel® AMT can set MEBX password<br>0x00 – Intel® AMT can NOT set MEBX password |
| 0x000a | OEM SKU Rule | 10 bit value that controls which features are enabled or disabled | 1 – Features enabled<br>0 – Feature Disabled<br>Bit 1 – QST<br>Bit 2 – ASF2<br>Bit 3 – AMT Pro<br>Bit 4 – AMT Basic<br>Bit 5 – TPM<br>Bit 8 – Castle Peak<br>Bit 10 – Wake on Event<br>All other bits are reserved |
| 0x1001 | QSTLock | Determines if the end user can enable or disable QST in the MEBX. **Not used by Mobile platform** | 0x02 – User can NOT enable/disable QST<br>0x00 – User can enable/disable QST |
| 0x1002 | QSTState | Set default QST state to enable or disable. **Not used by Mobile platform** | 0x00- Disable<br>0x01 - Enable |
| 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 |

| FOV ID | Variable Name | Description | Value Type |
|---|---|---|---|
| 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<br>0x02 – Enable Lock |
| 0x2004 | MngMode | Determines the default manageability mode | 0x00 – None<br>0x01 – Intel® AMT<br>0x02 – ASF<br>0x03 – CastlePeak (consumer platform only) |
| 0x2005 | EncryptionEnable | Determines if TLS encryption is used | 0xff – Enabled TLS<br>0x00 – Disable TLS |
| 0x2006 | FullTestCounter | The number of Full/Gracefull test allowed by MEManuf/MEMaanufwin | Integer in HEX format<br>Value must be between 0-255 |
| 0x2007 | AMTConfigMode | Determines if the default AMT mode is small business mode, enterprise, or remote connectivity service mode | 0x01 – Enterprise<br>0x02 - Small Business mode<br>0x03 – Remote Connectivity Service 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<br>Value Range = 0x0000 <N < 0xFFFF |
| 0x2009 | RCFGEnable | Remote configuration parameter. This parameter can only be enabled by FPT/FPTw. | 0x01 – Enable<br>This value can only be set to disabled by FITC |
| 0x200a | CfgSrvFQDN | The fully qualified domain name for the setup and configuration server. | Valid domain name |



| FOV ID | Variable Name | Description | Value Type |
|---|---|---|---|
| 0x200b | Hash1 | Certificate Hash Data | See Firmware Variable Structure document for details |
| 0x200c | Hash2 | Certificate Hash Data | See Firmware Variable Structure document for details |
| 0x200d | Hash3 | Certificate Hash Data | See Firmware Variable Structure document for details |
| 0x200e | MEBx Password Change Policy | The policy that controls when the MEBx password and the ME Network Password are synched<br>Default Password Only – Passwords are synched only after the password is changed from the default value.<br>During Setup and Configuration – Passwords are synched after the Setup and Configuration process is complete.<br>Always – Password are synched when either password is changed | 0x00 – Default Password Only<br>0x01 – During Setup and Configuration<br>0x02 - Always |
| 0x200f | EnablerID | A unique numeric identifier that indicates the enabler of the PC for Remote Connectivity Service. | 16 bytes Hex values<br>All 0 or all f are not valid values |
| 0x2010 | EnablerDescription | A human readable string that describes company represented by the EnablerID | A 60 byte string value |
| 0x2012 | RCSCapabilty | Determines if the platform supports the Remote Connectivity service option | 0x00 – Not Supported<br>0x01 – Supported |

# 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 Inte® 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 | |

| Error Number | Error String | Possible Corrective Actions |
|---|---|---|
| 4105 | ISVS_InitializeCOMinThread was not called by the current thread. | |
| 4106 | URL required | |

## B.2 Intel® ME Interface 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 | |



| Error Number | Error String | Possible Corrective Actions |
|---|---|---|
| 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 |
|---|---|---|
| 8703 | System reset is required to complete firmware update process | Restart the system before attempting to perform another firmware update |
| 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 |

| Error Number | Error String | Possible Corrective Actions |
|---|---|---|
| 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<br>OR<br>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 |

## 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<br>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 |
| 9220 | Manufacturing test failed due to a BIOS error | One or more of following BIOS tables may not exist:<br>1) FRU table<br>2) Media device<br>3) SMBios memory entry<br>4) SMBios processor entry<br>5) SMBios battery entry<br>6) SMBios Computer system entry<br>7) SMBios baseboard entry<br>8) SMBios bios entry<br>9) ASF table<br>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 |



| Error Number | Error String | Possible Corrective Actions |
|---|---|---|
| 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 |

## 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 |

| Error Number | Error String | Possible Corrective Actions |
|---|---|---|
| 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 |
| 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 |



| Error Number | Error String | Possible Corrective Actions |
|---|---|---|
| 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 read access to the target flash area. To enable read access for this operation you must modify the descriptor settings to give host access to this region | Check descriptor region access settings |
| 202 | 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 | |
| 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" | |

| Error Number | Error String | Possible Corrective Actions |
|---|---|---|
| 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 | |
| 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. |



| Error Number | Error String | Possible Corrective Actions |
|---|---|---|
| 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 | |
| 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 0x01 when modifying FOV parameters is no longer desired |
| 463 | Unable to get master base address from the descriptor | Check file integrity |
| 464 | Invalid image. Can not proceed | |

| Error Number | Error String | Possible Corrective Actions |
|---|---|---|
| 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. Check file integrity |
| 480 | The PPS is invalid | |
| 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 | |



| Error Number | Error String | Possible Corrective Actions |
|---|---|---|
| 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 | |
| 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 | |
| 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 intrepreted | |
| 10013 | The key handle points to an invalid key | |



| Error Number | Error String | Possible Corrective Actions |
|---|---|---|
| 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 | |
| 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 | |

| Error Number | Error String | Possible Corrective Actions |
|---|---|---|
| 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 | |
| 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 writtable | |



| Error Number | Error String | Possible Corrective Actions |
|---|---|---|
| 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 | |
| 10081 | The DAA command has no resources availble 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 |

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