

Intel[®] Active Management Technology

Validation Tools User Guide

April 2007

Revision 1.1



DISCLAIMER: 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. Intel products are not intended for use in medical, life saving, or life sustaining applications.

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.

Intel® Active Management Technology requires the platform to have an Intel® AMT-enabled chipset, network hardware and software, connection with a power source and a network connection

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

Intel, Intel vPro™ and Intel logo are trademarks or registered trademarks of Intel® Corporation or its subsidiaries in the United States and other countries.

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

Copyright © 2006-2007, Intel® Corporation



Contents

1	Intel Software License Agreement.....	8
2	Document Information	10
2.1	Revision History.....	10
2.2	Terminology	10
2.3	Reference Documents.....	11
3	Preface.....	12
3.1	Intel® Management Engine.....	12
3.2	Intel® Active Management Technology (Intel® AMT)	12
4	Introduction	14
4.1	Overview	14
4.1.1	Manufacturing Line Validation Tools	14
4.1.2	Image Editing Tools	14
4.1.3	Integration Validation Tools	14
4.1.4	Requirements.....	15
4.1.5	Tools Summary Table.....	15
5	Flash Image Tool	18
5.1	System Requirements	18
5.2	Flash Image Details.....	18
5.2.1	Flash Space Allocation.....	19
5.3	Required Files.....	19
5.4	Getting Started.....	20
5.4.1	Creating a New Configuration	20
5.4.2	Opening an Existing Configuration	20
5.4.3	Saving a Configuration	20
5.5	Environment Variables.....	20
5.6	Build Settings.....	22
5.7	Modifying the Flash Descriptor Region	22
5.7.1	Setting the Number and Size of the Flash Components.....	23
5.7.2	Region Access Control	24
5.8	MCH Strap 0.....	28
5.8.1	ME Boot from Flash.....	28
5.9	ICH Strap 0.....	29
5.9.1	SM Bus.....	29
5.9.2	TCO mode	29
5.10	VSCC Table	29
5.10.1	Adding a New table.....	30
5.10.2	Removing an existing table	30
5.11	Modifying the ME Region.....	31
5.11.1	Setting the ME Region Binary File	31
5.11.2	Enabling/Disabling the ME Region.....	31



5.12	Modifying the GbE (LAN) Region	32
5.12.1	Setting the GbE Region Binary File.....	32
5.12.2	Enabling/Disabling the GbE Region	33
5.13	Modifying the BIOS Region.....	33
5.13.1	Setting the BIOS Region Binary File	33
5.13.2	Enabling/Disabling the BIOS Region	34
5.14	Building a Flash Image	34
5.15	Decomposing an Existing Flash Image	34
5.16	Command Line Usage.....	35
6	Flash Programming Tool	38
6.1	System Requirements	38
6.2	Required Files.....	38
6.3	Acquire DOS4GW.EXE.....	39
6.4	Fparts.txt file	39
6.5	Programming the flash device	40
6.6	Usage.....	40
6.7	Examples.....	42
7	MEManuf	47
7.1	Requirements	48
7.2	Windows PE requirements.....	48
7.3	Firmware Counter	49
7.4	Complete Test	49
7.5	Partial Test	50
7.6	Usage.....	50
7.7	Examples.....	51
7.8	Error Codes.....	52
8	AMTNVM.....	55
8.1	Requirements	55
8.2	Usage.....	55
8.3	Examples.....	56
8.4	Parameter Fields Description	57
9	Fixed Address Programming (FAUPD)	65
9.1	Requirements	65
9.2	Usage.....	65
9.3	FAUPD tool Flow	67
9.4	PSK File format.....	67
9.5	MAC address File format	68
9.6	Examples.....	68
9.7	Parameters Definition	69
9.8	Error Codes.....	70
10	EEUpdate.....	73
10.1	Requirements.....	73
10.2	Usage.....	73
10.3	Example	73



11	FWUpdLcl	76
	11.1 Requirements	76
	11.2 Non-Secure Dos Requirements	76
	11.3 Non-Secure Windows Requirements	76
	11.4 Secure Windows Requirements	77
	11.5 Windows* PE Requirements	77
	11.6 Enabling and Disabling Local Firmware Update	77
	11.7 Usage DOS Version	78
	11.8 Usage Windows* Version	78
	11.9 Examples	79
12	AMTSD	81
	12.1 Requirements	81
	12.2 Usage	81
	12.3 Execution Flow	82
13	AMTFeaturesLocal	83
	13.1 Requirements	83
	13.2 Usage	83
14	AMTFeaturesRemote	85
	14.1 Requirements	85
	14.2 Usage	85
	14.2.1 Errors	87
	14.2.2 Warnings	87
15	MEInfo	89
	15.1 Requirements	89
	15.2 Windows* PE requirements	89
	15.3 Usage	89
	15.4 Examples	91
	15.5 Error Codes	92
16	AMTRedirection	96
	16.1 Requirements	96
	16.2 Usage	97
	16.3 Verify SOL/IDE-R in TLS mode	99
17	Setup and Configuration Application (Configuration Server)	102
	17.1 Tool Requirements	102
	17.2 Usage	103
	17.3 PSK/PID File Format	103
	17.4 Sample TLS Certificates	104
	17.5 Configuring an Intel® AMT system in Enterprise mode	104
	17.5.1 Modify PSK Repository	105
	17.5.2 USBfile.exe	105
	17.5.3 Configure Setup Parameters	105
	17.5.4 Prepare Intel® AMT Systems	106
	17.5.5 Install sample Certificates	106



18	Intel® AMT Privacy Icon	108
18.1	System Requirements	108
18.2	Usage.....	108



Figures

Figure 1. Firmware image components	18
Figure 2. Environment Variables Dialog	21
Figure 3. Build Settings Dialog	22
Figure 4. Editable Flash Image Region List	23
Figure 5. Descriptor Region <input type="checkbox"/> Descriptor Map Options	23
Figure 6. Descriptor Region <input type="checkbox"/> Fast Read Support Options	24
Figure 7. Descriptor Region <input type="checkbox"/> Component Section Options	24
Figure 8. Descriptor Region <input type="checkbox"/> Master Access Section Location	26
Figure 9. Descriptor Region <input type="checkbox"/> Master Access Section Options	26
Figure 10. ROM bypass warning	28
Figure 11. Message to determine if Firmware image contains ROM bypass section	29
Figure 12. Add New VSCC table entry	30
Figure 13. Add VSCC table entry	30
Figure 14. VSCC table entry	30
Figure 15. Remove VSCC table entry	31
Figure 16. Enabling the ME Region via Right-Click Context Menu	32
Figure 17. GbE Region Options	32
Figure 18. Disabling the GbE Region via Right-Click Context Menu	33
Figure 19. BIOS Region Options	33
Figure 20. BIOS Region Disabled	34
Figure 22. Full Firmware with image components	55
Figure 23. Intel® AMT Privacy Icon	108
Figure 24. Intel® AMT Status window	109

Tables

Table 1. Revision History	10
Table 2. Terminology	10
Table 3. Reference Documents	11
Table 4. Tools Summary	15
Table 5. Region Access Control table	24
Table 6. Minimum Read/Write Values	27
Table 7. Minimum Read/Write Values	28
Table 8. MEmanuf Error Codes	52
Table 9. AMTNVM Parameters Table	57
Table 10. FAUPD Error Codes	70
Table 11. Firmware Override Update Variables	78
Table 12. MEInfo Error Codes	92
Table 13. PSKRepository.xml format	104



1 Intel Software License Agreement

IMPORTANT - READ BEFORE COPYING, INSTALLING OR USING.

Do not use or load this software and 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 to 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:

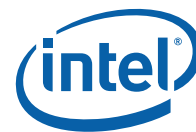
- **Developer Tools** include developer documentation, installation or development utilities, and other materials, including documentation. You may use, modify and copy them internally for the purposes of using the Software as herein licensed, but you will not distribute all or any portion of them.

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 PROFITS, BUSINESS INTERRUPTION, OR LOST INFORMATION) ARISING OUT OF THE USE OF OR INABILITY TO USE THE SOFTWARE, EVEN IF INTEL HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. SOME JURISDICTIONS PROHIBIT EXCLUSION OR LIMITATION OF LIABILITY FOR IMPLIED WARRANTIES OR CONSEQUENTIAL OR INCIDENTAL DAMAGES, SO THE ABOVE LIMITATION MAY NOT APPLY TO YOU. YOU MAY ALSO HAVE OTHER LEGAL RIGHTS THAT VARY FROM JURISDICTION TO JURISDICTION.

TERMINATION OF THIS AGREEMENT: Intel may terminate this Agreement at any time if you violate its terms. Upon termination, you will immediately destroy the Software or return all copies of the Software to Intel.

APPLICABLE LAWS: This license, and all claims arising under this license, shall be governed by the laws of Delaware, excluding its principles of conflict of laws and the United Nations Convention on Contracts for the Sale of Goods. You will not export the Software in violation of applicable export laws and regulations. Intel is not obligated under any other agreements unless they are in writing and signed by an authorized representative of Intel.

GOVERNMENT RESTRICTED RIGHTS: The Software is provided with "RESTRICTED RIGHTS." Use, duplication, or disclosure by the Government is subject to restrictions as set forth in FAR52.227-14 and DFAR252.227-7013 et seq. or its successor. Use of the Software by the Government constitutes acknowledgment of Intel's proprietary rights therein. Contractor or Manufacturer is Intel Corporation, 2200 Mission College Blvd., Santa Clara, CA 95052.

§



2 Document Information

2.1 Revision History

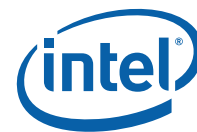
Table 1. Revision History

Revision Number	Description	Revision Date
0.3	Pre-Alpha	06/13/2006
0.374	Alpha 1	09/13/2006
0.42	Alpha 2	10/20/2006
0.52	Beta 1	11/17/2006
0.651	Beta 2	12/22/2006
0.821	Beta 3	01/26/2007
0.89	Production Candidate	02/20/2007
0.93	Production Version	02/22/2007
1.1	Hot Fix release	04/10/2007

2.2 Terminology

Table 2. Terminology

Term	Description
Intel® AMT	Intel® <u>A</u> ctive <u>M</u> anagement <u>T</u> echnology.
BIOS	<u>B</u> asic <u>I</u> nput- <u>O</u> utput <u>S</u> ystem
Complete SPI Image	Complete SPI image contains a Descriptor, BIOS, GBE and ME region
FW	<u>F</u> irm <u>W</u> are, specifically firmware executing on the ME.
SOL	<u>S</u> erial <u>O</u> ver <u>L</u> AN
IDE-R	<u>I</u> DE <u>R</u> edirection
Intel® QST	Intel® <u>Q</u> uiet <u>S</u> peed <u>T</u> echnology. 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® MEI	Intel® Intel Management Engine Interface



Term	Description
SOAP	<u>S</u> tandard <u>O</u> bject <u>A</u> ccess <u>P</u> rotocol
TLS	<u>T</u> ransport <u>L</u> ayer <u>S</u> ecurity
SNMP	<u>S</u> imple <u>N</u> etwork <u>M</u> anagement <u>P</u> rotocol

2.3 Reference Documents

Table 3. Reference Documents

Document	Document No./Location
OEM Bring Up Guide	Production Candidate kit
Intel® AMT WEB UI Guide	Production Candidate kit
Users Guide to the Setup and Configuration Application	iAMT tools\iamtconfiguration
Intel® I/O Controller Hub 8 (ICH8) Family datasheet	http://download.intel.com/design/chipsets/datashts/31305602.pdf
Intel® AMT SDK	http://softwarecommunity.intel.com/isn/home/manageability.aspx

3 Preface

3.1 Intel® Management Engine

In 2007 Intel® platforms, a new hardware architecture will replace the previous solution. The primary features of this architecture are:

1. A separate processing engine, the Intel® Management Engine (ME), provided within the MCH, which serves as a processor for Intel® AMT capabilities, including out-of-band (OOB) operation.
2. Dedicated memory space within main memory to execute ME code and store ME run-time data.
3. A LAN controller that supports OOB activity.

3.2 Intel® Active Management Technology (Intel® AMT)

Intel® AMT's features can be summarized as follows:

1. Highly-Available OOB Remote Management
 - Provides remote management capabilities in all system power and health states
 - Runs on auxiliary power
2. OS-Independent
 - Runs outside the context of the OS
 - Works the same way regardless of the installed OS
 - Immune from OS configuration issues
3. Tamper Resistant
 - Intel® AMT agent bound to the PC and configured by IT
 - Difficult for end-user to modify or disable
 - Robust network and local host interface security



4. Capabilities

- Discovers: Hardware and software inventory
- Heals: Remote control, event management, IDE Redirection, and Serial Over LAN
- Infrastructure: Firmware update, Setup and Configuration, network and security administration, local communications over SOAP/TLS, and mutual authentication
- Protect: System Defense for network outbreak containment and detection with agent presence

§

4 Introduction

4.1 Overview

Software tools described in this document are designed to assist in qualifying and verifying the implementation of Intel® AMT on a new platform. A brief overview of the tools follows.

4.1.1 Manufacturing Line Validation Tools

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

- **MeManuf and MEManufWin:** Validates Intel® AMT functionality on the manufacturing line.

4.1.2 Image Editing Tools

- **Flash Image tool:** This image combines the GBE, BIOS and ME firmware into one image that can be programmed by a flash programming device or the Flash Programming Tool.
- **AMTNVM:** Edits parameters within the complete SPI binary image. The edited image can be programmed onto an Intel® AMT system that needs to be setup and configured.
- **FWUpdLcl:** Updates the firmware code of a flash device already programmed with a full firmware image.
- **Flash Programming tool:** Programs the flash device on the Intel® AMT system. This tool can program individual regions, or the entire flash device.
- **EEUpdate:** This tool can update the Ethernet MAC address after the firmware has been fully programmed.

4.1.3 Integration Validation Tools

Integration validation tools are used by system integrators to check and validate various aspects of Intel® AMT functionality.

- **AMTSD:** Provides an interactive demonstration of Intel® AMT's System Defense feature.



- **AMTFeaturesLocal:** Queries the Intel® AMT subsystem for a local feature and reports if this feature is available or not. Tested features include LMS, the Intel Management Engine Interface driver, FW Intel® Management Engine Interface network interface, and access to NVM storage.
- **AMTFeaturesRemote:** Queries the Intel® AMT subsystem for a remote feature and reports if this feature is available or not. Remote features include event manager, remote control, hardware assets, network administration, security administration, System Defense, storage and storage administration, agent presence remote interface and wireless configuration.
- **MEInfo and MEInfoWin:** Queries the ME and returns information such as BIOS, FW, and Intel Management Engine Interface driver versions.

4.1.4 Requirements

Manufacturing line validation tools run on MS-DOS* 6.22, Windows* 98 DOS, Windows* XP (SP1/2) or Windows Vista*. Integration validation tools run on a Windows (Win2K SP4, XP SP1/2, PE, and Vista) machine or on DOS (Not all tools are supported in DOS). Integration validation tools that run locally on the Intel® AMT machine requires the following services to be installed: Intel® AMT Local Manageability Service (LMS) and the Intel Management Engine Interface driver. Check individual tool descriptions for the exact requirements.

4.1.5 Tools Summary Table

Table 4 Tools Summary

Tool Name	Feature Tested	Runs on Intel® AMT System	Runs on Management System
MEManuf and MEManufWin	Connectivity between Intel® AMT Devices	X	
MEInfo	Firmware Aliveness; Outputs certain Intel® AMT parameters	X	
AMTFeaturesLocal	Local access to NVM storage area	X	
AMTFeaturesRemote	Remote administrative features		X
AMTSD	Interactive System Defense test		X
AMTNVM	Edits Intel® AMT specific parameters and outputs a new firmware image file to program onto the flash devices	X	X
Fixed Variable Programming Tool (FAUPD)	Updates board specific parameters on a full firmware image file or on the SPI flash device	X	
Flash Programming Tool	Programs the image onto the flash device of the Intel® AMT system	X	



Tool Name	Feature Tested	Runs on Intel® AMT System	Runs on Management System
Flash Image Tool	Prepares the image files to be programmed onto the flash programming tool	X	X
Firmware Update Local	Updates the firmware code while maintain the values previously set	X	
EEUpdate	Updates the Ethernet MAC Address	X	

§



5 Flash Image Tool

The purpose of the Flash Image Tool is to simplify the creation and configuration of the Flash image for the Crestline/ICH8M platform. The Flash Image Tool can take the following image files form of binary files:

- BIOS
- Gigabit Ethernet
- Intel Management Engine

And assembles them into a complete flash image. The user is able to manipulate the image layout through a graphical user interface (GUI) and change the various chipset parameters to match the target hardware. Different configurations can be saved to a file so image layouts do not need to be recreated each time.

The user does not need to interact with the GUI each time they need to create an image. The tool 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 Flash Image Tool does not program the flash. The Flash Image tool only generates a binary image file. This image must be burned onto the flash by another means.

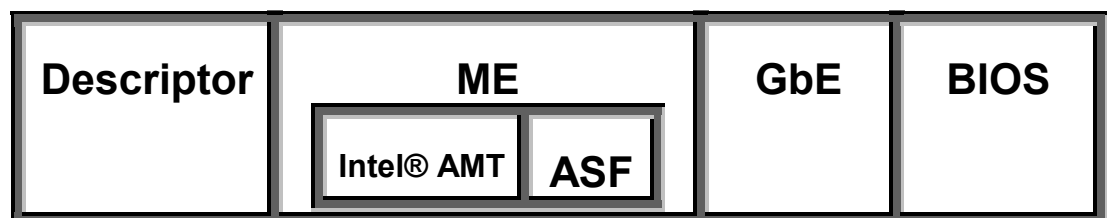
5.1 System Requirements

The Flash Image Tool will run on Windows* XP. The system on which the tool is run does not need to be ME-enabled.

5.2 Flash Image Details

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

Figure 1. 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, and a space which can be used for vendor-specific data. *This region must be locked before the Intel® AMT system is shipped to end users. Please see Region Access control section for more information. Failure to lock the descriptor region will leave the Intel® AMT system vulnerable to attacks.*
- ME: Optional region that takes up a variable amount of space at the end of the descriptor. Contains code and configuration data for ME functions such as Intel® AMT and ASF 2.0
- GbE: Optional 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. BIOS contains code and configuration for the entire platform.

5.2.1 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 occupies only as much space as it requires (in 4 Kbyte increments).
2. If after allocation for all regions there is still space left in flash, then the ME region expands to fill the remaining space.
3. If there is leftover space and the ME region is not implemented, then the BIOS region is expands 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 is expands to contain the remaining space.
5. Lastly, if only the Descriptor region is implemented, it expands to contain the entire flash.

5.3 Required Files

The Flash Image Tool main executable is ftoolc.exe. This program requires that the following files be in the same directory as ftoolc.exe:

- ftooltmplc.xml
- newfiletmpl.xml

The Flash Image Tool will not run correctly if these files are missing.



5.4 Getting Started

It is recommended that the user set the environment variable paths first before working with a new configuration (see 5.5 Environment Variables). Next, the user should modify the build settings to the desired configuration (see 5.6 Build Settings). Performing these 2 steps first will make configuring the flash image easier and help ensure the configuration is portable to other machines.

The flash image can be configured in many different ways, depending on the target hardware and the firmware options desired. The Flash Image Tool enables the user to change this configuration in a graphical manner (through 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.

5.4.1 Creating a New Configuration

The Flash Image Tool provides a default configuration file for the user to build from. This default configuration can be loaded by selecting "File → New" from the main menu.

5.4.2 Opening an Existing Configuration

An existing configuration file can be loaded by selecting "File → Open..." from the main menu. This will cause the Open File dialog to appear. Select the XML file you want to load and click "Open". It is also possible to open a file by simply dragging-and-dropping a configuration file onto the main window of the application.

5.4.3 Saving a Configuration

The current configuration can be saved to an XML file by selecting "File → Save" or "File → Save As..." from the main menu. If "Save As..." is selected or if the configuration has not been given a name, the Save File dialog will appear. Select the path and file name to save the configuration under and click "Save".

5.5 Environment Variables

To modify the environment variables, select "Build → Environment Variables..." from the main menu. 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 machine, or override the variables using command line options.

It is recommended that the environment variables be the first thing the user sets when working with a new configuration. This ensures that the Flash Image Tool can properly substitute environment variables into paths to keep them relative. Doing this

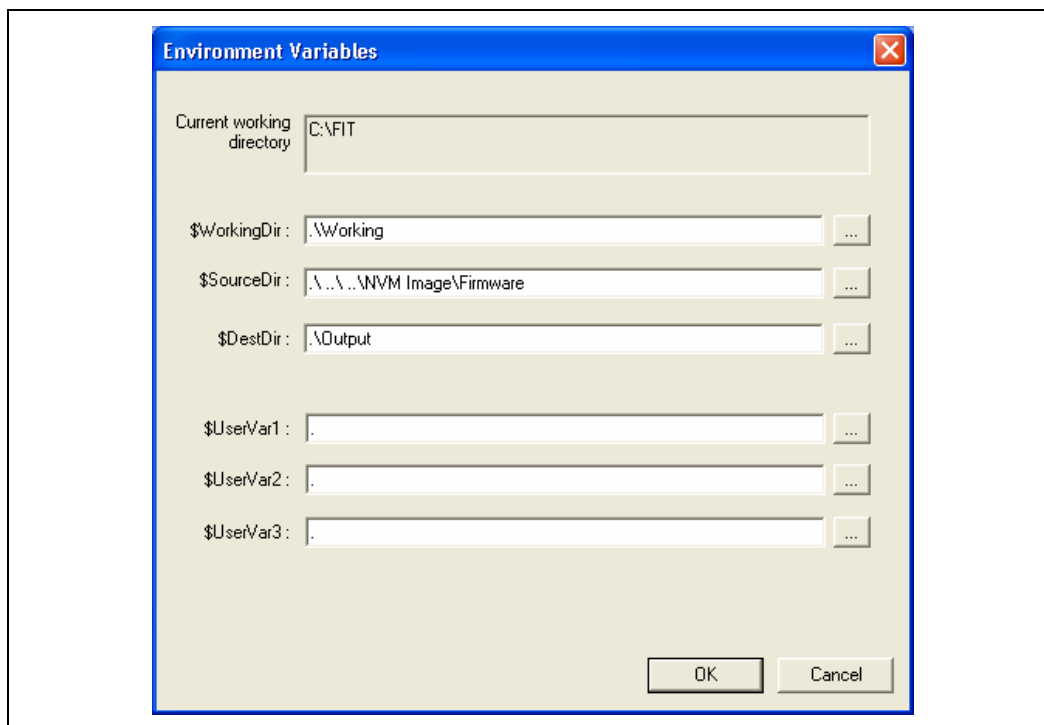
also speeds up configuration because many of the Open File dialogs default to particular environment variable paths.

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 (see section 3.9).
- **\$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, or from 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 (see section 3.8).

In addition to the core \$SourceDir, \$DestDir, and \$WorkingDir environment variables, the Flash Image Tool provides a set of user-defined variables. These variables can be used in addition to the core variables to provide more relative paths.

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.

5.6 Build Settings

To modify the build setting, select “Build → Build Settings...” from the main menu. A dialog box will appear showing the current build settings.

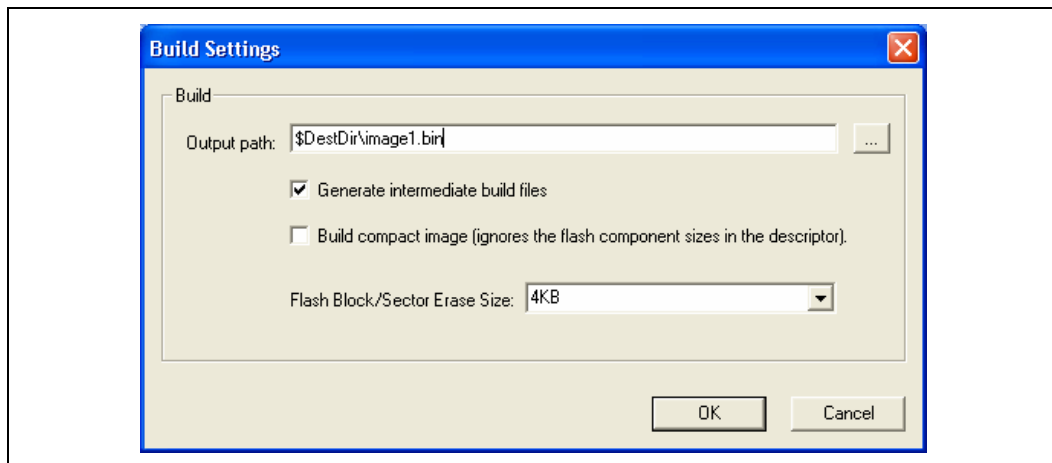
The Flash Image Tool allows the user to set several options that control how the image is built. The “Output path” is the path and filename where the final image should be saved after it is built. (Use the \$DestDir environment variable to make the configuration more portable)

An option is provided that instructs 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 folder called “int” located inside the specified output folder. These image files can be individually programmed using the Flash Programming Tool.

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. This is critical to correctly set this option to ensure the flash regions can be properly updated at runtime. All the regions in the flash are aligned to the 4Kbyte sector erase size.

Figure 3. Build Settings Dialog



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

5.7 Modifying the Flash Descriptor Region

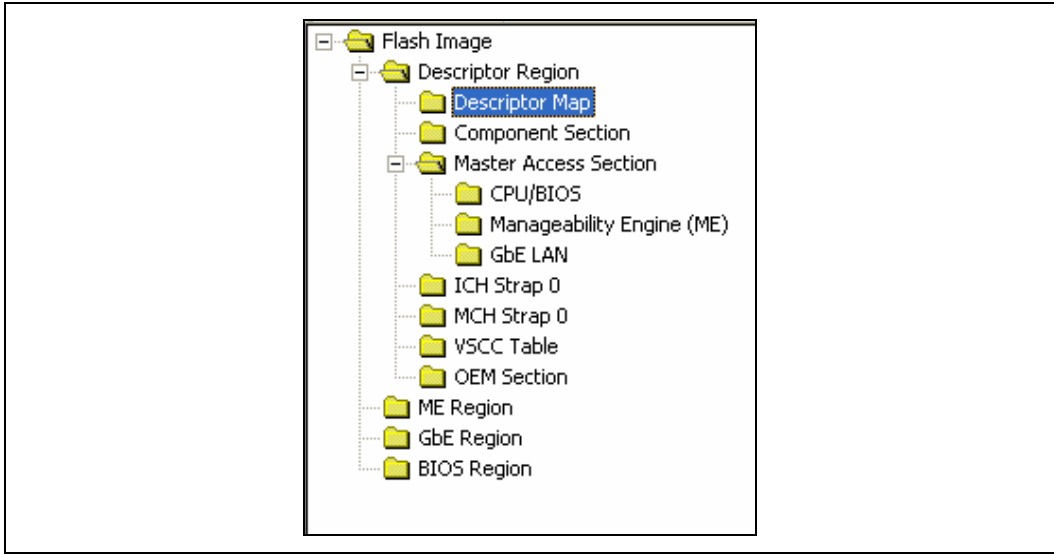
The flash descriptor region contains information about the flash image and the target hardware. It is important for this region to be configured correctly or else the target system may not function as desired.



5.7.1 Setting the Number and Size of the Flash Components

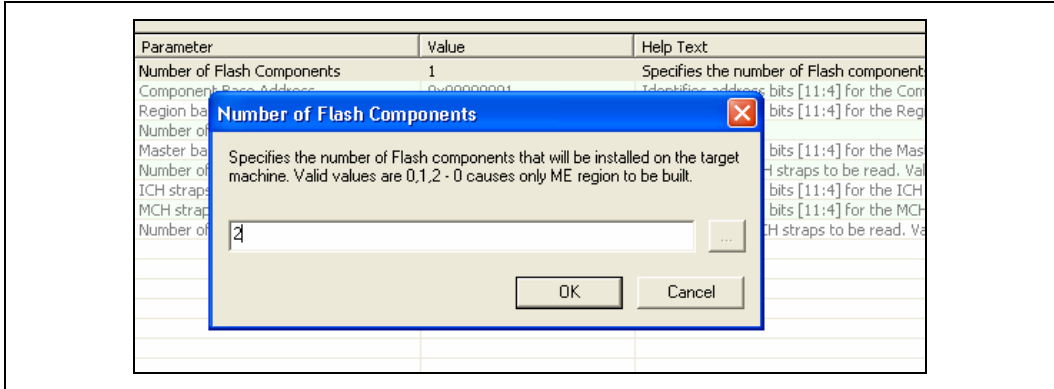
To set the number of flash components, expand the “Descriptor Region” node in the tree on the left side of the main window. Then, select the “Descriptor Map” node (See Figure 4). All of the parameters for the descriptor map section will appear in the list on the right side of the main window.

Figure 4. Editable Flash Image Region List



Double-click the list item named “Number of Flash Components” (See Figure 5). A dialog will appear allowing the user to enter the number of flash components (valid values are 1 or 2). Click “Ok” to update the parameter.

Figure 5. Descriptor Region ☐ Descriptor Map Options



Some SPI flash devices support both standard and fast read speeds. For the ICH8 to support these faster read speeds the fast read clock frequency must be set to 33Mhz and fast read support must be set to true.

Figure 6 Descriptor Region □ Fast Read Support Options

Read ID and Read Status clock frequency	20MHz	If more that one Flash component exists, this field must be th
Write and erase clock frequency	20MHz	If more that one Flash component exists, this field must be th
Fast read clock frequency	33MHz	This field is undefined if the Fast Read Support is set to false.
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 component.
Flash component 2 density	512KB	This field identifies the size of the 2nd Flash component.
Illegal Instruction 0	0	Op-code for an illegal instruction that the Flash Controller shd
Illegal Instruction 1	0	Op-code for an illegal instruction that the Flash Controller shd
Illegal Instruction 2	0	Op-code for an illegal instruction that the Flash Controller shd
Illegal Instruction 3	0	Op-code for an illegal instruction that the Flash Controller shd

To set the size of each flash component, 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 (See Figure 7). Double-click on each parameter and select the correct component size from the drop-down list. 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 7. Descriptor Region □ Component Section Options

Parameter	Value	Help Text
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	2MB	This field identifies the size of the 1st Flash com
Flash component 2 density	2MB	This field identifies the size of the 2nd Flash cor
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

5.7.2 Region Access Control

Regions of the flash can be protected from read or write access by setting a protection parameter in the Descriptor. Before Intel® AMT systems can be shipped, the Descriptor region MUST be locked. If the description region is not locked down, the Intel® AMT system remains vulnerable. The level of read/write access is at the OEM/ODM's choice. A cross-reference of access settings is shown below.

Table 5 Region Access Control table

Region (#) to which Access is Allowed or Denied	Chipset which is Allowed or Denied Access		
	CPU and BIOS	ME/MCH	GbE Controller
Descriptor (0)	None / Read / Write	None / Read / Write	None / Read / Write



	Chipset which is Allowed or Denied Access		
BIOS (1)	Write only. CPU and BIOS can always read from and write to BIOS region	None / Read / Write	None / Read / Write
ME (2)	None / Read / Write	Write only. ME/MCH can always read from and write to ME/MCH region	None / Read / Write
GbE (3)	None / Read / Write	None / Read / Write	Write only. GbE controller can always read from and write to GbE region

Three parameters in the descriptor exist to specify access for each chipset. The bit structure of these parameters are shown below (0 = denied access, 1 = allowed access, NC = bit may be either 0 or 1 since it is unused).

CPU / BIOS gets...

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

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

For example, if the CPU/BIOS needs read access to 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

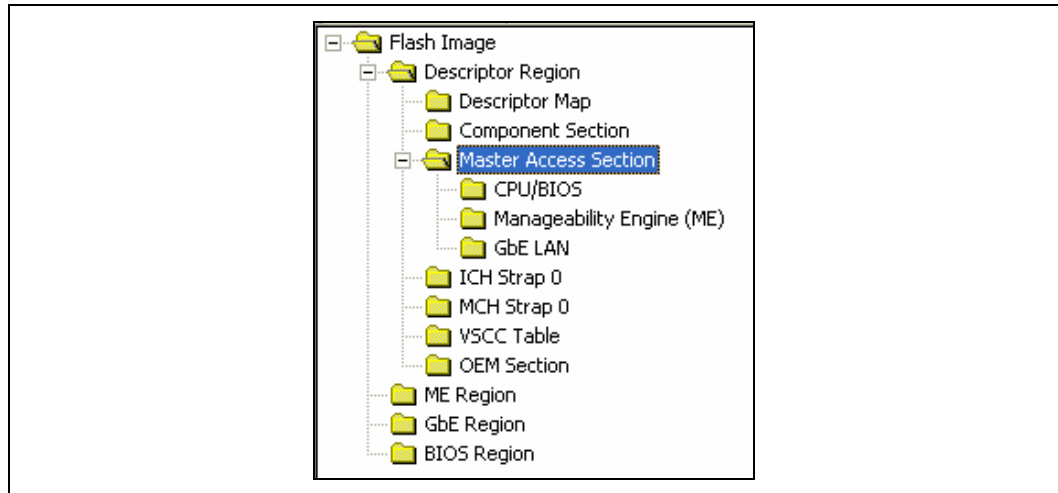
In hexadecimal:

Read Access: 0x 0A

Write Access: 0x 06

In the Flash Image Tool these access values can be set by selecting the "Descriptor Region" tree node and selecting "CPU/BIOS" under "Master Access Section" (see Figure 8).

Figure 8. Descriptor Region □ Master Access Section Location



The read and write access hexadecimal values can be specified in the appropriate parameters (see Figure 9).

Figure 9. 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				GbE	ME	BIOS	Desc
Bit Value	NC	NC	NC	NC	0/1	NC	0/1	0/1
Bit Number	7	6	5	4	3	2	1	0

Write Access								
	Unused				GbE	ME	BIOS	Desc
Bit Value	NC	NC	NC	NC	0/1	NC	0/1	0/1



Write Access								
Bit Number	7	6	5	4	3	2	1	0

GbE Controller gets...

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

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

The following is the minimum set of the read/write parameters. This sample will lock down descriptor region with an acceptable level of security and still provide access to the rest of the regions on the flash device. 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. The descriptor region must be locked before the Intel® AMT system is shipped to end users. By locking the descriptor region late in the manufacturing flow, the manufacturer has 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 6 Minimum Read/Write Values

	ME	GbE	BIOS
ME read access	Y	N	N
ME write access	Y	N	N
GbE read access	Y	Y	Y
GbE write access	Y	Y	Y
BIOS read access	N	N	Y
BIOS write access	N	N	Y

Descriptor read access	Y	N	Y
Descriptor write access	N	N	N

The table below shows the values to be inserted into the Flash image tool. The values below will provide the access levels described in the table above.

Table 7 Minimum Read/Write Values

	ME	GbE	BIOS
Read	0b 0000 1101 = 0x0d	0b 0000 1000 = 0x08	0b 0000 1011 = 0x0B
Write	0b 0000 1100 = 0x0c	0b 0000 1000 = 0x08	0b 0000 1010 = 0x0A

5.8 MCH Strap 0

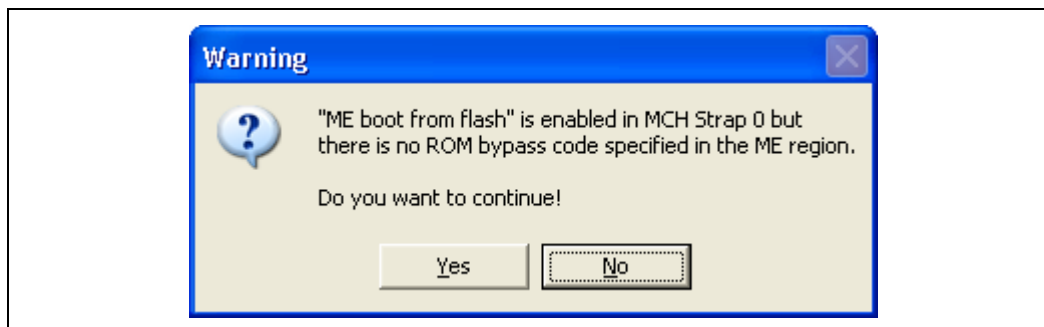
This section contains the variable to edit the address at which the Intel® Management Engine begins to read.

5.8.1 ME Boot from Flash

When this option is set to true firmware will begin to read from the ROM bypass section of code. If this section is set to true, the firmware loaded must contain a ROM bypass section. This option can be selected to avoid known hardware or software problems associated with the ME.

When this option is loaded a small amount of code is ran 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 a warning will occur when attempting to build the image file. Clicking "No" will cancel the build process.

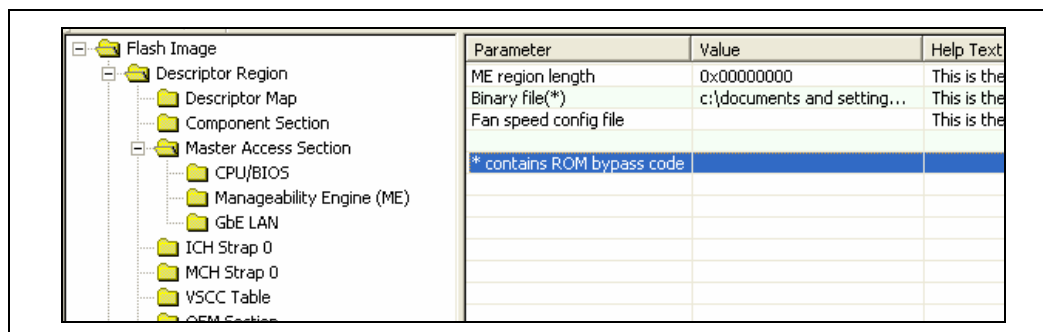
Figure 10 ROM bypass warning



If this warning occurs the user can ignore the warning and continue to build the image file, or choose another firmware image file, that does contain an ROM bypass file. If

the firmware image does contain a ROM bypass section, a message will be displayed in the ME Region section.

Figure 11. Message to determine if Firmware image contains ROM bypass section



5.9 ICH Strap 0

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

5.9.1 SM Bus

The SMBus address should be set to 0x64. This is to ensure the correct address location is given to the SMBus. Giving an incorrect address will result in the code starting at an incorrect address.

By default, the SMBus 2 address should be set to 0x63 and the SMBus 2 select should be set to 1(SMLink pins)

5.9.2 TCO mode

There are two options available: Legacy mode and TCO mode. For Intel® AMT to function the user should select TCO mode.

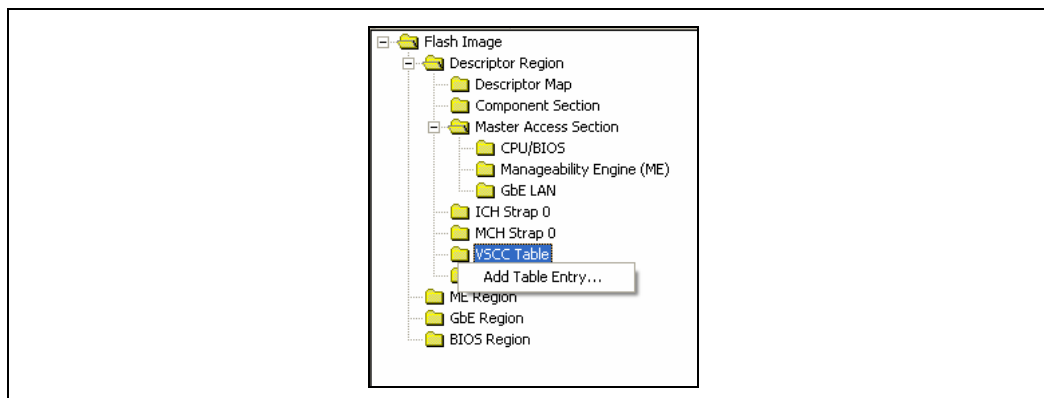
5.10 VSCC Table

This section is used to store information regarding the flash devices used on the system. This information is used by the firmware. If the information in this section is incorrect, the ME may not communicate with the flash device. This information provided here is dependent on the flash device used on the system. Please contact your flash vendor for this information. Please have your flash vendor refer to section 19.2.5.5(VSCC0—Vendor Specific Component Capabilities) in the ICH 8 data sheet (<http://download.intel.com/design/chipsets/datashts/31305602.pdf>) to locate the correct information.

5.10.1 Adding a New table

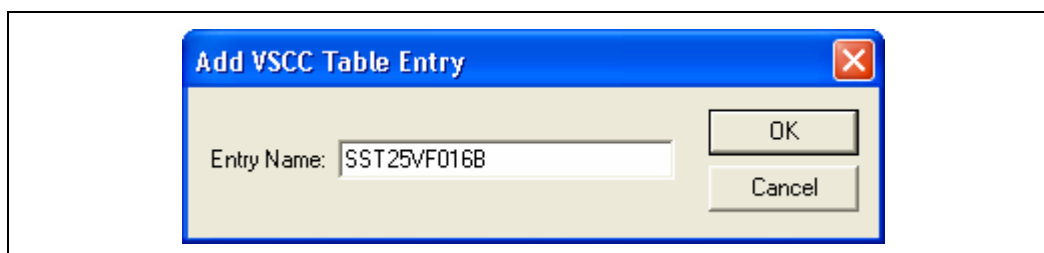
To add a new table, right click on vscc table and select add a new table entry.

Figure 12 Add New VSCC table entry



The program will then prompt the user for a table entry name. To avoid confusion it is recommended that each table entry be unique. Ftoolc will not create an error message for table entries that have the same name

Figure 13 Add VSCC table entry



After a table entry has been added, the user will be able to fill in values for the flash device. The values in the VSCC table are provided by your flash vendor. The information in the VSCC table entry is the same information that is displayed in the fparts.txt file from the Flash Programming tool. The picture below shows the values for the flash part SST25vf016b.

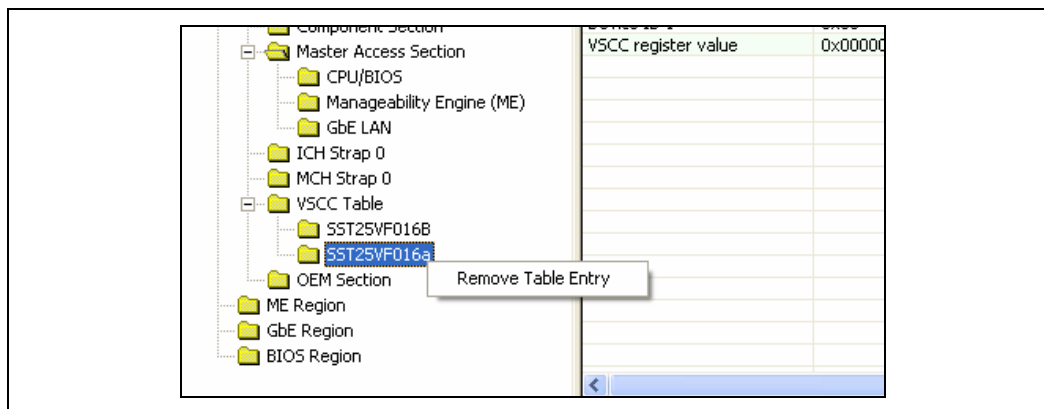
Figure 14 VSCC table entry

Parameter	Value	Help text
Device ID 0	0x25	The first device specific byte of the JEDEC ID.
VSCC register value	0x00002008	The device specific VSCC register value.

5.10.2 Removing an existing table

To remove an existing table, right click the table that needs to be removed and select remove table. All information in the table along with the table entry will be removed.

Figure 15 Remove VSCC table entry



5.11 Modifying the ME Region

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

5.11.1 Setting the ME Region Binary File

To set the ME region binary file, select the “ME Region” tree node. Double-click the “Binary file” parameter from the list. A dialog box will appear allowing the user to specify the ME file to use. 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 be left alone. A value of “0x00000000” indicates that the ME region will be auto-sized as described in 5.2.1: Flash Space Allocation.

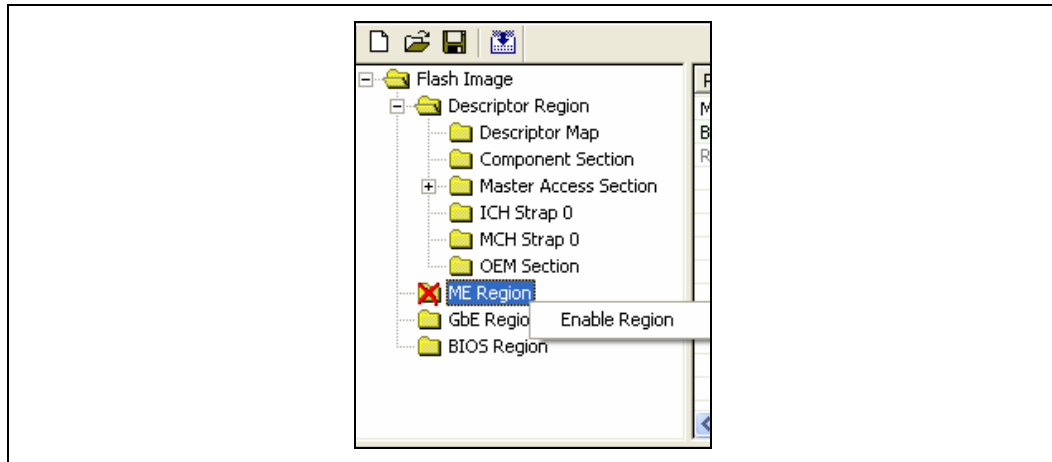
If the user has specified, in the MCH Strap 0 section, the ME to boot from flash, the firmware loaded must contain a ROM Bypass section. If the firmware does not contain a ROM bypass section, a section will become available to enter the location of the ROM bypass file.

5.11.2 Enabling/Disabling the ME Region

The ME region can be excluded from the flash image by disabling it in the Flash Image Tool. Simply right-click on the “ME Region” tree node and select “Disable Region” from the pop-up menu. When the flash image is built, there will be no ME region in it.

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

Figure 16. Enabling the ME Region via Right-Click Context Menu



5.12 Modifying the GbE (LAN) Region

The GbE region contains various configuration parameters (like the MAC address) for the embedded Ethernet controller.

5.12.1 Setting the GbE Region Binary File

To set the GbE region binary file, select the "GbE Region" tree node. Double-click the "Binary input file" parameter from the list. A dialog box will appear allowing the user to specify the GbE file to use. 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 be left alone. A value of "0x00000000" indicates that the GbE region will be auto-sized as described in 5.2.1: Flash Image Details.

Figure 17. GbE Region Options

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

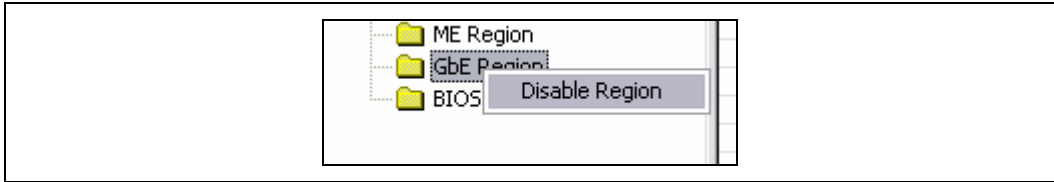
This is the location where the user can modify the Ethernet MAC address. To configure the Ethernet MAC address, simply double click the "MAC address" parameter from the list. A dialog box will appear allowing the user to specify the Ethernet MAC address. Click "Ok" to update the parameter.



5.12.2 Enabling/Disabling the GbE Region

The GbE region can be excluded from the flash image by disabling it in the Flash Image Tool. Simply right-click on the “GbE Region” tree node and select “Disable Region” from the pop-up menu. When the flash image is built, there will be no GbE region in it.

Figure 18. Disabling the GbE Region via Right-Click Context Menu



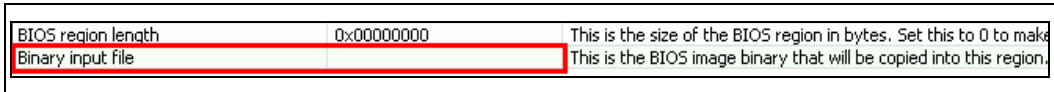
The GbE region can be enabled by right-clicking on the “GbE Region” tree node and selecting “Enable Region” from the pop-up menu.

5.13 Modifying the BIOS Region

The BIOS region contains the BIOS code run by the host processor. The Flash Image Tool always aligns this region with the end of the flash image. The reason for this is because if the flash descriptor gets corrupted, the ICH will default to legacy mode and look for the reset at the end of flash. 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 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 the end.

5.13.1 Setting the BIOS Region Binary File

Figure 19. BIOS Region Options



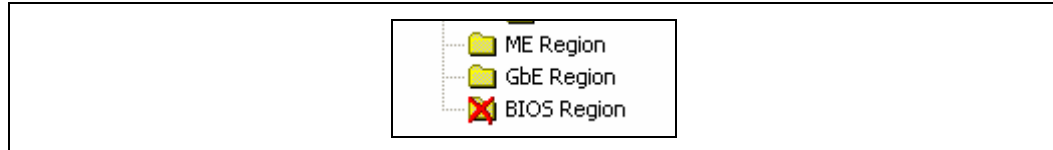
To set the BIOS region binary file, select the “BIOS Region” tree node. Double-click the “Binary input file” parameter from the list. A dialog box will appear allowing the user to specify the BIOS file to use. Click “OK” to update the parameter. When the flash image is built, the contents of this file will be copied into the BIOS region.

The BIOS region length option should be left alone. A value of “0x00000000” indicates that the BIOS region will be auto-sized as described in 5.2.1: Flash Space Allocation.

5.13.2 Enabling/Disabling the BIOS Region

The BIOS region can be excluded from the flash image by disabling it in the Flash Image Tool. Simply right-click on the “BIOS Region” tree node and select “Disable Region” from the pop-up menu. When the flash image is built, there will be no BIOS region in it.

Figure 20. BIOS Region Disabled



The BIOS region can be enabled by right-clicking on the “BIOS Region” tree node and selecting “Enable Region” from the pop-up menu.

5.14 Building a Flash Image

The flash image can be built using the tool’s GUI interface. Through this interface the user can create image files by using the currently loaded configuration and selecting “Build → Build Image” from the main menu, or by specifying an XML file with the /b option on the command line.

The Flash Image Tool uses an XML configuration file and the corresponding binary files to build a BW/ICH8M flash image. The output of the build will be a binary file representing the image, a text file detailing the different regions in the image, an optional set of intermediate files (see 3.3 Build Settings), and (if 2 flash components are specified) multiple binary files containing the image broken up according to the flash component sizes.

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

5.15 Decomposing an Existing Flash Image

The Flash Image Tool is capable of taking an existing flash image and decomposing it to create the corresponding configuration. This configuration can be edited in the GUI just like any other configuration (see the following sections). A new image can be built from this configuration that is identical to the original except for the changes made by the user.

There are several ways to decompose a binary image file. An image can be decomposed by selecting “File → Open...” from the main menu, changing the file type filter to the appropriate file type, and clicking “Open”. The image will automatically be decomposed and the GUI updated to reflect the new configuration. An alternate way to decompose an image is by simply dragging-and-dropping the file onto the main window.



5.16 Command Line Usage

The Flash Image Tool supports command line options. To view all of the supported commands, run the application with the `/?` option. The command line syntax for the Flash Image Tool is:

```
ftoolc  [<xml_file>]
        [/b]
        [/o <file>]
        [/me <file>]
        [/gbe <file>]
        [/bios <file>]
        [/w <path>]
        [/s <path>]
        [/d <path>]
        [/u1 <value>]
        [/u2 <value>]
        [/u3 <value>]
        [/flashcount <1|2>]
        [/flashsize1 <0|1|2|3|4|5>]
        [/flashsize2 <0|1|2|3|4|5>]
```

- <xml_file>** The XML configuration file is used when generating a flash image file. A sample xml file is provided along with the tool. When an xml file is used with the **/b** option the flash image file will be built automatically.
- /b** Automatically builds the flash image. The UI will NOT be shown if this flag is specified. This 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
- /o <file>** The path and filename where the image will be saved. This command overrides the output file path in the XML file.
- /me <file>** Overrides the binary source file for the ME region with the specified binary file.
- /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
- /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 Bringup Guide.



- /s <path>** Overrides the source file directory environment variable (\$SourceDir). It is recommended that the user set these environmental variables first
- /d <path>** Overrides the destination directory environment variable (\$DestDir). It is recommended that the user set these environmental variables first
- /u1 <value>** Overrides the \$UserVar1 environment variable with the value specified. This variable is free for the user to use as they see fit.
- /u2 <value>** Overrides the \$UserVar2 environment variable with the value specified. This variable is free for the user to use as they see fit.
- /u3 <value>** Overrides the \$UserVar3 environment variable with the value specified. This variable is free for the user to use as they see fit.
- /flashcount <0|1|2>**
- Overrides the number of flash components in the descriptor region. If this value is zero, then only the ME region will be built.
- /flashsize1 <0|1|2|3|4|5>**
- Overrides the size of the 1st flash component with the size of the option selected. 0 = 512KB, 1 = 1MB, 2 = 2MB, 3 = 4MB, 4 = 8MB, 5 = 16MB.
- /flashsize2 <0|1|2|3|4|5>**
- Overrides the size of the second flash component with the size of the option selected: 0 = 512KB, 1 = 1MB, 2 = 2MB, 3 = 4MB, 4 = 8MB, 5 = 16MB.





6 *Flash Programming Tool*

The purpose of the Flash Programming Tool is to program an image file to the flash. The Flash Programming Tool can program the following “regions”, in the form of binary files, into flash:

- BIOS
- Gigabit Ethernet
- Intel® Management Engine

This tool can program an individual region, or the entire flash device.

6.1 System Requirements

The DOS version of the Flash Programming Tool (fprog.exe) will run on MS Dos 6.22, DRMKDos and FreeDOS.

The windows version (fprogw.exe) will run in Windows* XP (Sp2) and Win PE.

Both versions of the Flash Programming Tool require an operating system to run on. The tool is designed to deliver a custom image to a system that is able to boot, instead of a means to get a blank system up and running. The tool must run on the system with the flash that the user desires to program.

6.2 Required Files

The DOS version of the Flash Programming Tool main executable is fprog.exe. This version requires that the following files be in the same directory as fprog.exe:

- fprog.exe
- dos4gw.exe (DOS version only)
- fparts.txt

The windows version of the Flash Programming Tool executable is fprogw.exe. This version requires the following files to be in the same directory as fprogw.exe

- fprogw.exe
- fparts.txt
- ssedrvidl132e.dll
- ssePmxdll32e.dll
- ssepmxdrv.sys

The Flash Programming Tool will not run correctly if these files are missing.



6.3 Acquire DOS4GW.EXE

Due to licensing issues, dos4gw.exe is not distributed in the same package as the Flash Programming Tool. Open a web browser and navigate to the following URL:
<http://www.scene.org/file.php?file=%2Fresources%2Fdos4gw.exe&fileinfo>

Download the 'dos4gw.exe' file and save it in the same directory as the 'fprog.exe'. This file is only needed for the DOS version.

6.4 Fparts.txt file

This document contains a list of all flash devices that this tool supports. The flash devices listed in this file must contain a 4KB erase size. If the flash device is not listed below 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 4 KB erase sector and the total size of the SPI Flash device must be a multiple of 4 KB. The description and order of the fields is listed below:

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



6.5 Programming the flash device

Once the ME is programmed the ME 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. It is important to note that programming the flash device while ME is running may cause the flash device to become corrupt. The ME should be disabled before programming the full flash device. The ME can be disabled by:

- Disabling the ME through the BIOS or the MEBX
- Pulling down gpio33 (Manufacturing mode jumper) when powering on the system – If the parameters are configured to ignore this jumper, the bullet will not be a valid way to disable the ME.
- Removing memory from Channel 0 – This method will cause the ME to boot up in an error state and the error will be logged on the flash device. Programming the flash device should be done after the OS has fully booted.
- Set ME disable bits in the strap sections of the descriptor region – Please see the ICH EDS (sections 24.2.5.1 and 24.2.5.2) for more information
- Do not program the ME onto the flash device

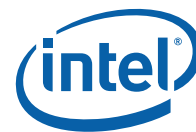
The ME does NOT need to be disabled when writing to the fixed offset region

Note: *After programming the device the system must go into a G3 state. The system may not function properly until the system goes into a G3 state.*

6.6 Usage

Warning: *To prevent possible corruption of the SPI device, it is recommended that the ME be disabled before programming the flash device. This can be done through the MEBX.*

Both the Windows version and DOS version of the Flash Programming Tool can be utilized through command line options. To view all of the supported commands, run the application with the /? option. The commands in DOS and Windows have the same syntax. The only difference would be in the name of the executable. The command line syntax for fprog.exe and fprogw.exe is:



```
fprog  [/?]
        [/c]
        [/b]
        [/i]
        [/f:<file>]
        [/v:<file>]
        [/d:<file>]
        [/address:<value>]
        [/length:<value>]
        [/desc]
        [/bios]
        [/me]
        [/gbe]
        [/y]
        [/q]
        [/e]
        [/p:<file>]
```

/?	Displays help screen.
/c	After the user confirms 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	Check to see if the flash part is erased, which will return in text if the flash is blank or not blank. If there are 2 flash devices and both are not 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, and if the flash descriptor is valid.
/f	Load binary file into flash starting at address 0x0000. The flash device must be written in 4 KB sections. The total size of the flash device must also be in increments of 4 KB
/v	Compare binary file to flash. If the binary file is not the 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 4 KB sections. The total size of the flash device must also be in increments of 4 KB
/d	Dump flash contents to a file or onto 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	Used in conjunction with load, verify or dump, allows the user to load, verify or dump a file beginning at the specified



	address. This option can not be used with the /desc, /bios, /me nor the /gbe option
/length	Used in conjunction with the load, verify or dump options, allows the user to specify the number of bytes to load, verify or dump. This option can not be used with the /desc, /bios, /me nor the /gbe option
/desc	Used in conjunction with the load, verify or dump options, allows the user to load, verify or dump to the descriptor region leaving the rest of the flash untouched. This option can not be used with the /address nor the /length option
/bios	Used in conjunction with the load, verify or dump options, allows the user to load, verify or dump to the BIOS region leaving the rest of the flash untouched. This option can not be used with the /address nor the /length option
/me	Used in conjunction with the load, verify or dump options, allows the user to load, verify or dump to the ME region leaving the rest of the flash untouched. This option can not be used with the /address nor the /length option
/gbe	Used in conjunction with load, verify or dump, allows the user to load, verify or dump to the GBE region leaving the rest of the flash untouched. This option can not be used with the /address nor the /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 area before writing to flash.
/p	Specifies a different flash part definition file to use as opposed to the one located within the executable

6.7 Examples

The following examples are from the DOS version (Fprog.exe) of the tool; however the windows version (Fprogw.exe) will behave in the same manner but from a windows command prompt.

Example 1

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:

```
C:\ fprog /me /d STDOUT
```

**Example 2**

Will load 2 Kbytes of the binary file (image.bin) starting at address 0x0000. The starting address and the length must be a multiple of 4KB

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

Example 3

Will loads file "bios.rom" into the BIOS region and verify the operation ran successfully

```
C:\ fprog /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)
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.
```

Example 4

Write the contents of the descriptor region to the file "descdump.bin"

```
C:\ fprog /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)
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
```

Example 5

Displays information about the flash devices present in the system if the flash device is specified within the fparts.txt file. The base address refers to the start location of the particular regions and the limit address refers to the end of the region

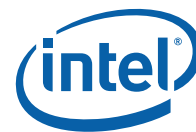
```
C:\ fprog /i
```

```
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)  
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: 0x1FFFFFFF  
ME - Base: 0x001000, Limit: 0x0FDFFF  
GbE - Base: 0x0FE000, Limit: 0x0FFFFFFF  
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
```

Example 6

Compares the ME region programmed on the flash with the specified firmware image file (CL_ICH8_REL_IAMT_BYP_ME.BIN). If the /y option is not inserted the user will be notified that the file is smaller than the binary image. This is an expected warning due to extra padding that is added during the program process. The padding is ignored when performing a comparison:



```

C:\ Fprog /me /v CL_ICH8_REL_IAMT_BYP_ME.BIN /y

-----
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 ---
    SST25VF016BID:0xBF2541 Size: 2048KB (16384Kb)
    SST25VF016BID:0xBF2541 Size: 2048KB (16384Kb)

Using software sequencing.
Reading file "CL_ICH8_REL_IAMT_BYP_ME.BIN" into memory...

Verifying Flash [0x0FE000]... 1012KB of 1012KB - 100% complete.

RESULT: The data is identical.

```

Example 7

Below is the sample output when comparing the file (outimage.bin) with the contents on the flash devices:

```

C: \ fprog /v 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

```



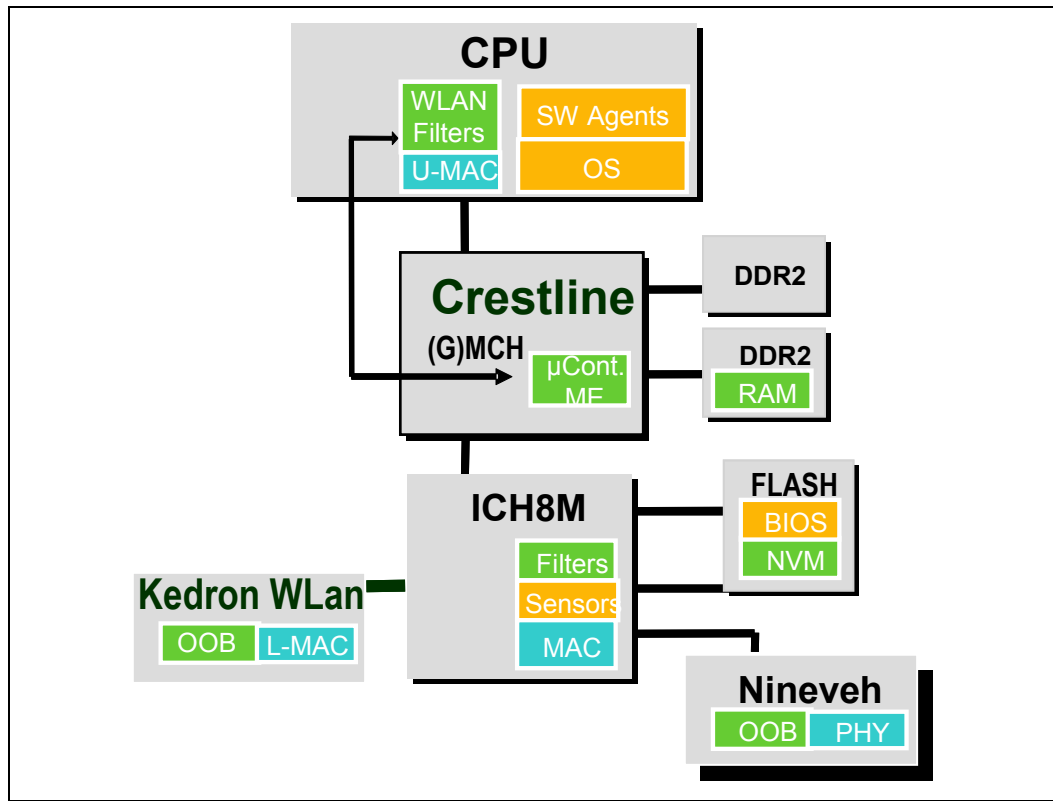
7 MEManuf

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

- SMBus Master and Slave
- BIOS, and BIOS/FW connectivity
- NVM Functionality
- Intel Management Engine Interface
- M-Link Connectivity
- AC power source
- Wireless card present
- S5 Functionality (not available for the Partial Test, see below)

MEManuf does not check for LAN functionality. Using the tool carries the assumption that all Intel® AMT components on the test board have been validated by their vendors. The tool verifies these components have been assembled together correctly.

MEManuf can run either a Complete Test or a Partial Test (see below).



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

7.1 Requirements

MEManuf runs on an Intel® AMT enabled machine running MS-DOS 6.22, Windows 98 DOS, FreeDOS, DRMK DOS or on a Windows (XP SP1/2, Vista 32bit/64bit) machine. The windows version of MEManuf (MEManufWin) requires the installation of the Intel Management Engine Interface driver. MEManuf and MEManufWin have a runtime of less than 30 seconds.

- In the ME BIOS Extension, **Intel® AMT** must be selected in the "Manageability Feature Selection" menu.
- MEManuf will **NOT** work with any other SKU.
- MEManuf will only work on a system running on AC power
- MEManuf must be run by a user with Administrator privileges.

7.2 Windows PE requirements

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

- The Intel® Management Engine Interface (MEI) driver must be installed in the Windows PE image
- The Windows PE image must be WMI enabled

7.3 Firmware Counter

For security reasons, the firmware counter indicates the maximum number of times a manufacturing test command was sent to the Host Interface. When the counter reaches zero, a manufacturing test command issued to the Host Interface is no longer acknowledged.

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

Once the counter has reached zero the image needs to be reprogrammed onto 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.

MEManuf can decrement the FW counter to 0 ("zero") using the following command:

```
MEManuf.exe -block
```

MEManuf can return the present value of the FW counter using the following command:

```
MEManuf.exe -counter
```

7.4 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.

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 results to registers.
- Decrements the FW counter.
- System reboots



Second Invocation

- MEManuf will be invoked by autoexec.bat if there were no failed tests from the 1st invocation.
- MEManuf obtains results of the test (in First Invocation) from registers. If a test has failed before the 2nd invocation, the test will return the results and the 2nd invocation will not be called.
- MEManuf reports these results to the user (see below).
- MEManuf clears the register of results that have now been extracted and reported.
- System performs **all** other manufacturing line tests. These can be specified manually in the *autoexec.bat* file that invokes MEManuf.

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.

7.5 Partial Test

MEManuf may optionally be run in Partial Test mode. The Partial Test is identical to the Complete Test (see above) except:

- 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.

7.6 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 -<option> [-nowlan]
```

Option is one of the following options listed below

- **-full** – This will run the partial test along with a system reset. The system reset will verify that ME is able to run s5 state. See the section above for more details on the full test. The power package selected must support ME in an S5 state.



- **-graceful** (Windows Version only) – This test is similar to the full test. However the graceful test will put the system into hibernate mode (S4) and bring the system back to S0. This test is only available if:
 - The system can go into hibernate mode (S4)
 - The Power package selected must support ME in the S4 state.

If the above options are not supported, MEManuf will not run the test and return with an error message.

- **-part** – Invokes the partial test only. See the section above
- **-nowlan** – This option may be added to any of the test (part, graceful, full). This option will have MEManuf ignore the wireless card tests. If this option is not used and there is no wireless card present, MEManuf will return a Kedron C link error.
- **-block** – Blocks all the future invocations of the full and graceful tests
- **-counter** – Displays the number of full test remaining
- **-version** – Displays the version of the MEManuf

7.7 Examples

Example 1:

```
MEManufWin.exe -graceful -nowlan
```

This will run the full test and ignore the wireless card tests. Instead of a hard power cycle, MEManuf will send windows into Hibernate (S4) mode and bring the system back into s0 state. This command should be used again to view the test results.

Example 2:

```
MEManufWin.exe -block
```

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

Example 3:

```
MEManufWin.exe -Full
```



The system will immediately go into an S5 state and power back on. To view the results the user must run the "-Full" option again. If this command is invoked in windows, the user may lose unsaved data.

7.8 Error Codes

The error codes fall into 3 categories, General Intel® AMT errors codes, BIST error codes, MEManuf application codes. A success has a return code of 0 and an appropriate success message will be displayed

Table 8 MEManuf Error Codes

General Intel® AMT error codes:

0	Intel® AMT Status Success
1	Intel® AMT Status Internal Error
2	Intel® AMT Status NOT Ready
3	Intel® AMT Status INVALID Intel® AMT MODE
4	Intel® AMT Status INVALID MESSAGE LENGTH
5	Intel® AMT Status TABLE FINGERPRINT NOT AVAILABLE
6	Intel® AMT Status INTEGRITY CHECK FAILED
7	Intel® AMT Status UNSUPPORTED ISVS VERSION
9	Intel® AMT Status INVALID REGISTRATION DATA
10	Intel® AMT Status APPLICATION DOES NOT EXIST
11	Intel® AMT Status NOT ENOUGH STORAGE
12	Intel® AMT Status INVALID NAME
13	Intel® AMT Status BLOCK DOES NOT EXIST
14	Intel® AMT Status INVALID BYTE OFFSET
15	Intel® AMT Status INVALID BYTE COUNT
16	Intel® AMT Status NOT PERMITTED
17	Intel® AMT Status NOT OWNER
18	Intel® AMT Status BLOCK LOCKED BY OTHER
19	Intel® AMT Status BLOCK NOT LOCKED
20	Intel® AMT Status INVALID GROUP PERMISSIONS
21	Intel® AMT Status GROUP DOES NOT EXIST

22	Intel® AMT Status INVALID MEMBER COUNT
23	Intel® AMT STATUS MAX LIMIT REACHED
24	Intel® AMT Status INVALID AUTH TYPE
25	Intel® AMT Status AUTHENTICATION FAILED
26	Intel® AMT Status INVALID DHCP MODE
27	Intel® AMT Status INVALID IP ADDRESS
28	Intel® AMT Status INVALID DOMAIN NAME
30	Intel® AMT Status REQUEST UNEXPECTED
31	Intel® AMT Status INVALID TABLE TYPE
32	Intel® AMT Status INVALID PROVISIONING MODE
33	Intel® AMT Status UNSUPPORTED OBJECT
34	Intel® AMT Status INVALID TIME
35	Intel® AMT Status INVALID INDEX
36	Intel® AMT Status INVALID PARAMETER
37	Intel® AMT Status INVALID NETMASK
38	Intel® AMT Status FLASH WRITE LIMIT EXCEEDED
39	Intel® AMT Status INVALID IMAGE LENGTH
40	Intel® AMT Status INVALID IMAGE SIGNATURE
41	Intel® AMT Status PROPOSE ANOTHER VERSION
42	Intel® AMT Status INVALID PID FORMAT
43	Intel® AMT Status INVALID PPS FORMAT
44	Intel® AMT Status BIST COMMAND BLOCKED
0x800	Intel® AMT Status NETWORK IF ERROR BASE

Built in Self Test (BIST) error codes:

201	An internal error has occurred
203	A flash read/write error has occurred
204	A problem sending commands on the SMBus was encountered
207	A power down error has occurred
208	A BIOS error has occurred

**MEManuf application error codes:**

210	Failed to connect to Intel® Management Engine Interface
211	An Intel® Management Engine Interface error has occurred
212	Memory Allocation Error
213	Usage Error
214	Unsupported OS
216	ME EC Error
217	Kedron C Link Error

§

8 AMTNVM

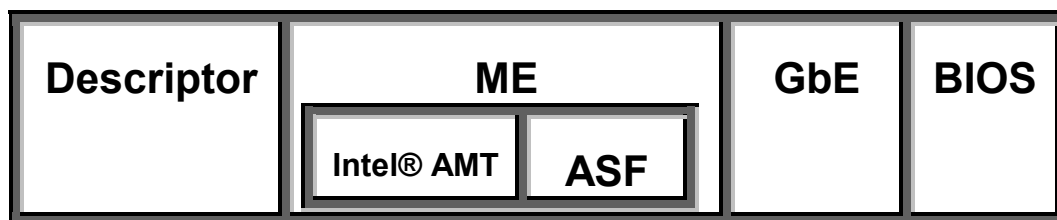
AMTNVM edits the default values of Intel® AMT parameters in an ME binary image. Some of the parameters include:

- Configuration Server port and name
- TCP/IP Settings
- Firmware Counter

A sample parameters file is included in the folder. This file is an example of how an edited parameters file may be used when editing an image file.

Warning: This is only a sample file and should **NOT** be used modify image files.

Figure 22 Full Firmware with image components



8.1 Requirements

AMTNVM runs on Windows (Windows XP SP1/2, Vista 32bit/64bit). It is a command-line executable and may be run on any machine with a compliant operating system.

8.2 Usage

The executable can be invoked by:

```
AMTNVM.exe    [-parse <ME Image File>]
               [-edit <ME Image File> <Params File>]
               -out <Output File>
```

- **parse** – Targets a BIOS image file or parameter specification file (see Section 5.4) for parsing
 - **ME Image File** – Complete SPI binary image filename to be parsed

- **edit** – Targets and loads an ME binary image file for editing
 - **ME Image File** – Complete SPI binary image filename to be edited
 - **Params File** – A file containing parameters and the default values (see Section 8.4: Parameter) that can be altered and then written to the BIOS image file
- **out** – Specifies the name and location of the generated output file.
 - **Output File** – In parse mode, the output is a human readable file with parameters and their values. In edit mode, the output is a complete SPI firmware image.

Note: If the Flash Descriptor Override Pin-Strap Ignore parameter is set to 1 (one) the flash override pin will always be ignored. If the flash device becomes corrupted, another method will be needed to restore the flash device to a working state.

8.3 Examples

Example 1:

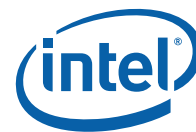
```
AMTNVM.exe -parse outimage.bin -out parms.txt
```

Creates a human-readable file (parms.txt) which lists the editable parameters in the input image file (outimage.bin) and their default values.

Example 2:

```
AMTNVM.exe -edit outimage.bin parms.txt -out outfile.bin
```

Receives an image file (outimage.bin) and a parameters file (parms.txt), edits the image file with the input parameters from the parameters file, and creates an output image file (outfile.bin) with the parameters from parms.txt.



8.4 Parameter Fields Description

It is recommended that the "BIOS Reflash Capable", "Boot into BIOS Setup Capable", and "Pause during BIOS Boot Capable" parameters remain unchanged and left as their default values (0,0,0).

A sample parameter text file is included in the AMTNVM folder. This sample file is an example of how a edited parameters list may appear when editing an image file.

Warning: This is only a sample file and should **NOT** be used modify image files.

These parameters enable the various features of an Intel® AMT system. If the system needs to go from the M1 to M-Off state (ME WOL) the "Remote Wake Timer" should be set to a value other than "00". If the value is set to 25, the ME will switch to M-off after being idle for 25 minutes. The minimum value for this is 2 minutes. Any value lower than 2 minutes will be ignored and the 2 minute value will be used. If the Idle Timeout parameter is modified in the MEBX, the value of remote wake timer will **NOT** be used. The idle timeout timer will determine the amount of idle time before the ME goes into an M-off state.

The parameters that are listed by AMTNVM are the default values. If these parameters are manually changed by the user, AMTNVM will use the values entered by the user. Values that can not be changed by the user will remain unchanged even if the system is un-provisioned. All other values will be returned to the values as modified by AMTNVM.

For more details on each of the parameters please see the AMTNVM_editable_fields document. This document provides description, possible values, and default values for each parameter. The AMTNVM_editable_fields document is located in the same folder as the AMTNVM executable. A table of all of the values is listed below.

Warning: AMTNVM will not give warnings for invalid settings. The user should verify that the values are valid and do not conflict with other parameters. The table below shows all possible values with some warnings about incorrect configuration.

Table 9 AMTNVM Parameters Table

Parameter Name	Description	Parameter Type/Range	Mandatory
Manageability Mode	The ME Mode (Intel® AMT, ASF or NONE)	0 for NONE 1 for Intel® AMT 2 for ASF If Intel® AMT modes is selected: a) LAN Well should NOT be powered from the Core Well b) WLAN should NOT powered from the Core well	Yes



		<p>c) Power Package 2 MUST be supported</p> <p>d) Power Package 3 MUST be supported</p> <p>e) Intel® AMT MUST be supported</p> <p>If ASF mode is selected:</p> <p>a) LAN well should NOT be powered from the Core Well</p> <p>b) Package 3 MUST be supported</p> <p>c) ASF MUST be supported</p>	
Manageability Mode Lock	This will lock the ME Mode. Once this bit is set, the user will no longer be able to change the ME Mode through MEBx.	0 to unlock 1 to lock	Yes
Local Firmware Update Enabled	Determines whether Local FW Update is enabled	0 to disable 1 to enable	Yes
Local FWU Override Counter	Determines whether to override the MEBx Local FW Update configuration and enable the ME local FW Update channel. If -1 (always) is chosen, override will work only if allowed by Local FWU Override Qualifier	0 for never n (255 > n > 0) to allow override for precisely n host boot cycles -1 for always	Yes
Local FWU Override Qualifier	Determines whether the override is allowed if the Local FWU Override Counter is set to -1 (minus one)	0 (always) to allow override 1 (never) to disqualify override 2 (restricted) this will allow override only until ME is configured	Yes
Flash Descriptor Override Pin-Strap Ignore	0 - ME enters a Temporary Disabled state in order to safely recover the full flash image 1 - ME completely ignores the strap and remain fully functional	a bit value (0/1)	Yes
Intel® AMT Compatibility Mode	Determines whether the Intel® AMT is in Intel® AMT	a bit value (0/1) (1 for true)	Yes



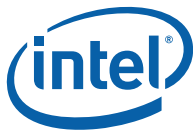
	1.0 mode (1 for true)		
Intel® AMT Configuration Mode	Determines whether the Intel® AMT is in Enterprise (0) or Small Business mode (1)	0 for enterprise 1 for small business	Yes
ZTC Enable	Determines whether the ZTC(Zero Touch Enable) is set or not	0 = No 1 = Yes	Yes
Configuration Server Port	The port of the configuration server	a two byte integer	No
Configuration Server Name	The name of the provisioning server (client will look for this server in DNS if no IP is set)	a multibyte character string of length 1 to 33	No
Configuration Server IP	The IP address of the configuration server	an IP address	No
Intel® AMT Host Name	The hostname	a multibyte character string of length 1 to 32	No
Intel® AMT Domain Name	The domain name	a multibyte character string of length 1 to 222	No
DHCP Enabled	Determines whether DHCP is enabled by default	a bit value (0/1) 0 is disabled; 1 is enabled	Yes
Intel® AMT Ping Response Enabled	Determines whether the Intel® AMT device responds on ping	a bit value (0/1) 0 is disabled; 1 is enabled	Yes
Intel® AMT Static IP Address	The Intel® AMT device IP address	an IP address	Yes
Intel® AMT Static IP Subnet Mask	The Intel® AMT device Subnet Mask	an IP address	Yes
Intel® AMT Static IP Default Gateway Address	IP address of the Default Gateway	an IP address	Yes
Intel® AMT Static IP Primary DNS Address	IP address of the primary DNS	an IP address	Yes
Intel® AMT Static IP Secondary DNS Address	IP address of the secondary DNS	an IP address	Yes
Intel® AMT Dedicated MAC Address	The Intel® AMT MAC address	a source mac address (xx-xx-xx-xx-xx-xx where every x is either a decimal digit (0-9) or one of characters a,b,c,d,e or f)	Yes



		(The address must be non-zero, and the second hex-digit must be even.)	
VLAN	The VLAN tag that is used for the manageability VLAN	a two byte integer	Yes
PET Language Code	Language Code used in PET events by default.	a two character hexadecimal value (0xhh where each h is either a decimal digit (0-9) or one of the characters a,b,c,d,e or f)	No
PET OEM Custom Fields 00-15	OEM custom fields used in PET events	16 two-character hexadecimal values, separated by single whitespace characters	No
PET OEM Custom Fields 16-31	OEM custom fields used in PET events	16 two-character hexadecimal values, separated by single whitespace characters	No
PET OEM Custom Fields 32-47	OEM custom fields used in PET events	16 two-character hexadecimal values, separated by single whitespace characters	No
PET OEM Custom Fields 48-63	OEM custom fields used in PET events	16 two-character hexadecimal values, separated by single whitespace characters	No
PET OEM Custom Fields Length	The number of valid OemCustomField bytes	a single byte integer between 0 and 64 Note: when the value is 64, the final OEM custom byte is ignored	No
PET Community String	The event manager community string used in PET events	a multi-byte character string of length 1 to 16	Yes
AMTManuf Test Counter	The number of full AMTManuf tests allowed	a single byte integer	Yes
IDER Boot Capable	Determines whether IDE Redirection is supported.	a bit value (0/1) 0 is disabled; 1 is enabled	Yes
SOL Boot Capable	Determines whether Serial over LAN is supported.	a bit value (0/1) 0 is disabled; 1 is enabled	Yes
BIOS Reflash Capable	Determines whether BIOS Reflash is supported.	a bit value (0/1) 0 is disabled; 1 is enabled	Yes



Boot into BIOS Setup Capable	Determines whether BIOS boot into setup screen is supported.	a bit value (0/1) 0 is disabled; 1 is enabled	Yes
Pause during BIOS Boot Capable	Determines whether BIOS pause before booting operating system is supported.	a bit value (0/1) 0 is disabled; 1 is enabled	Yes
HostIf SOL Enabled	Determines whether Serial over LAN is enabled. Note: "SerialOverLAN" must be included in the list of Intel® AMT enabled interfaces in order for SOL to be allowed. (See remote SOAP commands Get/SetEnabledInterfaces.)	a bit value (0/1) 0 is disabled; 1 is enabled	Yes
HostIf IDER Enabled	Determines whether IDE Redirection is enabled. Note: "IdeRedirection" must be included in the list of Intel® AMT enabled interfaces in order for IDER to be allowed. (See remote SOAP commands Get/SetEnabledInterfaces.)	a bit value (0/1) 0 is disabled; 1 is enabled	Yes
Remote Wake Timer	Determines the Remote Wake Timer. This value is the same as the Idle Timeout timer in the MEBX. The value will not be seen in the MEBX. If the Idle timeout timer is modified in the MEBX, the Remote wake timer value will not be used.	a two byte integer This value is in minutes. A value of 5 will be read as 5 minutes. The minimum value is 2 minutes. Any value lower than 2 minutes will be ignored and the 2 minute value will be used.	No
ME Visual LED Indicator Disabled	Determines whether the ME Visual LED Indicator is disabled.	a bit value (0/1) 0 is available; 1 is not available	Yes
LAN Power Well	Determines where the LAN Well is powered from	a bit value (0/1/2) 0 = Core Well 1 = Suspend Well 2 = ME Well ORed with GPIO9 (WOL_EN)	Yes



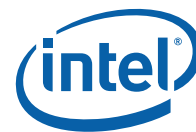
WLAN Power Well	Determines where the WLAN Well is powered from	a bit value (0/1/2/3) 0 = No WLAN Well 1 = Core Well 2 = Suspend Well 3 = ME Well	Yes
Power Package <N> Supported (<Power Package Description>)	Determines if Power package <N> is supported. Power package <N> can be selected user only if the power package is supported.	A bit value (0/1) 0 = NOT Supported 1 = Supported 1) Power package 1 MUST be supported 2) Power package 2 MUST be supported if the system supports Intel® AMT 3) Power Package 3 MUST be supported if the system supports Intel® AMT or ASF 4) Power Package 4 can NOT be supported if the LAN Well is powered from the Core Well 5) Power Package 5 can NOT be supported if the LAN Well is powered from the Core Well	Yes
Default Power Package	Displays the default Power Package from the available Power Packages.	A bit value (1-n), where n = the valid available power package number The default power package selected must be a supported power package.	Yes
Kedron Driver WA	Determines if the workaround for ICH B0 system is invoked.	A bit value (0/1). If there is no Kedron card installed on the platform the value must be set to 1 (one). If this bit is set to 1, the system will perform a global reset every time the Suspend well loses power	Yes



Intel® AMT Legacy Provisioning Mode Supported	Displays the whether the legacy Provision mode supported or not.	A bit value (0/1). 0 is disabled; 1 is enabled If enabled Intel® AMT must be supported	Yes
Intel® AMT VLAN Local Configuration Blocked	Displays the whether the VLAN configuration is blocked or not.	A bit value (0/1). 0 is disabled; 1 is enabled	Yes
iQST Supported	Displays the whether the iQST supported by the OEM or not.	A bit value (0/1). 0 is disabled; 1 is enabled	Yes
ASF Supported	Displays the whether the ASF supported by the OEM or not.	A bit value (0/1). 0 is NOT Supported; 1 is supported If manageability mode is set to "ASF", ASF MUST be supported.	Yes
Intel® AMT Supported	Displays the whether the Intel® AMT supported by the OEM or not.	A bit value (0/1). 0 is NOT Supported; 1 is supported If manageability mode is set to "AMT", Intel® AMT MUST be supported	Yes
PKI DNS Suffix	Displays the DNS Suffix.	A multi-byte character string of length 1 to 255	Yes

§





9 Fixed Address Programming (FAUPD)

The fixed address program tool can:

- Extract parameters from the fixed offset location
- Update multiple parameters in the fixed offset location
- Update single parameter in the fixed offset location

This tool can only read and write directly from the flash. This tool can not write to any other region on the flash device.

If the CPU does not have write access to the ME section, this tool will not work. This tool must be run on the system with the flash device that requires the update. The fixed address programming tool is located inside the system tools folder of the release kit.

9.1 Requirements

FAUPD runs on DOS (MS-DOS 6.22, Windows 98 DOS, FreeDOS, DRMK DOS). FAUPDw runs on Windows (Windows XP SP1/2, 2003). Both tools are a command-line executable and may be run on any machine with a compliant operating system.

FAUPD requires DOS4GW.exe. Please see section 5.3 for instructions on how to download this file.

The Fixed address program and all the files in the folder need to be in the same directory as fprog. The DOS version of the fixed address programming tool must be placed with the DOS version of fprog tool. The windows version of FAUPD must be placed with the windows version of fprog.

9.2 Usage

FAUpd will only program the parameters in the fixed offset region. The ME will not use these parameters until they are copied to their final location. To copy the parameters to the final location the global valid bit must be written to. After the globalvalid bit has been flipped, the system must go into a G3 state, before the parameters are copied to their final location. After this bit is flipped the parameters can not be modified using FAUpd. To modify the parameters, the entire flash device needs to be reprogrammed.

Please see the following section on the tool flow

Both tools FAUPD and FAUPDW have the same usage and functionality. The only difference is the operating system. The below examples will use the DOS tool



example, but can also be used for windows tool (FAUPDW). The executable can be invoked by:

```
FAUPD.exe [options]
```

The options are listed below are the most common use cases.

- **-e -o <Parameters text file>** – This option will extract a list of parameters from the SPI flash device that can be updated by FAUPD in a text file format. If a binary file is not specified, the program will extract the list of parameters from the flash device. The “-out” option must be used in with the “-extract” option.
- **-u -i <Parameters Text file>** – Updates the parameters inside the supplied text file directly to the SPI flash device.
- **-u [-n <parameter name> -v <parameter Value>]** – Updates the specified parameters with the given value. The update will be directly to the flash. **All numeric values must have “0x” before the value.**

Below are all of the options listed in a list format

- **-e** – Extracts the parameters that can be updated from the SPI flash device. The parameters text file will be outputted to a text file as specified from the “-out” option.
- **-u** - Updates the flash device. Updates can be made to individual parameters using the “-n” option or multiple parameters using the parameters text file and the “-i” option.
- **-o <parameter Text file>** - Specifies the parameters text file from the “-e” option. This will allow the user to specify the name of the parameters text file. When enabled a parameter using the outputted text file, the value to enable a parameter should be 0x00. Parameters that do not have the enable value set to 0x00 will not be copied to the fixed offset region. See below for the definitions of the parameters.
- **-i <parameter Text file>** - Allows the user to provide the name of parameters text file to read from when using the “-u” option
- **-n <Parameter name> - v <Parameter Value>** - Used with “-u” option, this allows the user to update a single parameter with the value given. If the parameter name or its value contains a space, the entire name/value needs to be in quotation marks. To update more than 1 (one) parameter, the user can use the “-o” option or the tool will need to be called again.
All numeric values must have “0x” before the value.
- **-PSKfile <PSK File>** - This will allow manufactures to give each system a unique PID,PPS and password triplet from a single source file. The format of the PSK file is the same as the USB key provisioning format. Please see the Setup and Configuration user guide for more information
- **-MACfile <MAC File>** - This allows manufactures to give each system a unique MAC address. Please see the section below regarding the MAC Address



file format. The GBE MAC address and Manageability MAC. To update both MAC address two calls to the file must be made or the parameter output file can be used.

- **-lock** - This option allows the user to lock down all three regions to a predefined value. The read and write values will be set to the values that Intel recommends. Please see the Region Access control section (5.7.2) for more details. This option should be done AFTER the Global Valid bit is set to 0x00. The system needs to go into g3 before the descriptor region is locked.
- **-y** - Overwrites the output file if the file already exists. The user will be asked for confirmation before overwriting an existing file if this option is not used.

9.3 FAUPD tool Flow

When using the Fixed Address programmer, the parameters modified are not used by the system until the global valid bit is set. FAUPD requires the FPROG tool to be in the same folder

- 1) Program full SPI Image onto flash device
- 2) Restart the system
- 3) Extract parameters from Flash Device (Please see Example 1)
- 4) Modify parameters from the text file that FAUPD outputs
- 5) Update parameters by text file (Please see Example 2) or by individual names using the "-n" and "-v" option (Please see Example 3)
- 6) Set Global Valid bit (Please see Example 4)
- 7) Lock Descriptor Region using FAUPD -lock
(**optional**) See section 5.7.2 for details on read/write values set by FAUPD
- 8) Put system in G3 state

After the global valid bit is set and the system recovers from the g3 state, the parameters modified will be used by the Intel® AMT system. The modified parameters can be verified by using MEINFO or AMTNVM.

9.4 PSK File format

The PSK file must be a bin file. To create the bin file there is a tool called **USBfile.exe** located in the "iamttools\iamtConfiguration\ConfigScripts". This file will create random PID/PPS pairs in a binary file and an XML file. The binary file created by this tool is also used to create USB setup and configuration key. The full usage of the tool can be found later in the document. The usage below will create the binary file for PSK file format for FAUPD

```
USBFile.exe PSKfaupd.bin PSKfaupd.xml admin Admin@98 10
```



This example will create a bin file (PSKfaupd.bin) with 10 PID/PPS pairs. The PID/PPS pairs can be viewed in the XML file (PSKfaupd.xml). The username (admin) and password (Admin@98) is the current password for the MEBX menu. Please see Section 17 for full usage details.

9.5 MAC address File format

The MAC address file can be used to update multiple MAC addresses from a single file source. The MAC Addresses should be written without dashes and in a text file. A sample file can be seen below.

```
000AC45D7800
000AC45D7801
000AC45D7802
```

The MAC address format also supports the use of brackets, [], after any given MAC address. If the bracket notation is used after a MAC address, the subsequent MAC addresses will be consecutive. The above example can be written in the following format.

```
000AC45D7800 [3]
```

9.6 Examples

Example 1:

```
FAUPD.exe -e -o flashparams.txt
```

Reads the binary image on the flash device and outputs a list of parameters (flashparams.txt) and their current values. The list of parameters is outputted in a text format

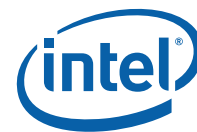
Example 2:

```
FAUPD.exe -u -i flashparams.txt
```

Updates the flash device with all the values from the parameters text file (flashparams.txt)

Example 3:

```
FAUPD.exe -u -n Password -v Admin@98
```



This updates the password value with the new value "Admin`12". In addition to this value, the user must also set the "**initvector**" parameter. The init vector parameter will create a hash key to store the password within the ME. If the "**initvector**" is not set, the ME password can **NOT** be changed. No error message will be given if the **initvector** is not programmed.

Example 4:

```
FAUPD.exe -u -n globalvalid -v 0x00
```

This will set the global valid bit. After the system returns from a G3 state the parameters set using FAUPD will be copied to their final location. The parameters in the fixed offset region will be cleared. Once the global valid bit is set, the only way to change the values is to re program the entire SPI device. After this value is set you will be able to verify the parameters using the validation tools.

9.7 Parameters Definition

For more information on the specifics of the fixed offset region, please see the Intel Manageability Engine Fixed Variable Offset for more information on this algorithm. This document outlines the length and valid values for each of the parameters listed below. When enabled a parameter using the outputted text file, the value to enable a parameter should be 0x00. Parameters that do not have the enable value set to 0x00 will not be copied to the fixed offset region and will not be used by the ME.

MngMacAdd – Manageability MAC address. If the system is in DHCP mode, this parameter will be ignored, and Intel® AMT will use the GBE MAC address. If the system is in Static mode the Manageability MAC address must be different from the GBE MAC address. The address is written as a 12 byte MAC address in HEX format(ex. 0x001122334455)

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.

Note: The value should be entered with a dash "-"(e.g. 1234-1234).

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.

Note: The value should be entered with a dash "-"(e.g. 1234-abcd-1234-abcd-1234-1234-abcd-1234).

MngFeatureLock – This parameter is used to lock management setting. Once enabled, the end user can not choose between management features such as Intel® AMT, ASF.

Allow Changes: 0x00

Dis-Allow Changes: 0x02

MngMode - This parameter is used to determine the default manageability feature.

None: 0x00



Intel® AMT: 0x01
ASF: 0x02

Note: If Intel® **AMT** mode is selected:

- a) LAN Well should **NOT** be powered from the Core Well
- b) WLAN should **NOT** powered from the Core well
- c) Power Package 2 **MUST** be supported
- d) Power Package 3 **MUST** be supported

If **ASF** mode is selected:

- a) LAN well should **NOT** be powered from the Core Well
- b) Package 3 **MUST** be supported
- c) ASF **MUST** be supported

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)

InitVector – Any pseudo Random number 32 bytes long in HEX format. This will create a hash value to securely store the MEBX password. Setting this parameter is required if setting the password through FAUPD. Please see the Intel Manageability Engine Fixed Variable Offset for more information on this algorithm (ex. 0x00112233445566778899012345678912)

Full Test Counter – This specifies the number of Full/Graceful test MEManuf can run. Once this counter reaches 0 (zero), Full and Graceful test can not be run. This value must be entered in HEX format.

HostMacAdd – The GBE Mac Address. The address is written as a 12 byte MAC address in HEX format (ex. 0x001122334455)

Globalvalid – The global valid parameter should be set to 0x00 after all parameters in the fixed offset region have been configured. After setting the global valid bit all of the parameters will be copied to their final location. The parameters in the fixed offset region will no longer be used and all of the parameters in this region will be deleted. If the parameters need to be fixed, the entire flash device will need to be reprogrammed. This parameter is not listed when using the "-e" option

9.8 Error Codes

Below are the error codes that FAUPd will output. These codes can be used to script FAUPd on the manufacturing line

Table 10 FAUPD Error Codes

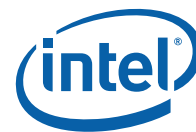
Error code	Error Message
------------	---------------



0	Success
1	Failure
2	Invalid Parameter value.
3	Image file already open.
4	Invalid Image.
5	Error occurred while opening image file
6	Parsing of image file failed.
7	ME region not found in image file.
8	Error occurred while opening/creating param file.
9	Param file already open.
10	Param file is not open
11	Corrupted Image file.
12	Invalid header marker.
13	File Exists.
14	File doesnot exists.
15	Invalid operation.
16	Param File Exists
18	Invalid variable name.
19	Error during parse.
20	Configuration file not exists.
21	Fprog application not exists.
22	Invalid length of value.
23	Invalid password.
24	Application expects parameters. Use -? or -h for help.
25	Error occurred during file write.
26	Error occurred during file read.
27	Invalid value passed for Globalvalid bit.
28	Descriptor region not found.
29	Invalid Lock region specified.
30	The setup file header has an invalid UUID.
31	The setup file version is unsupported.
32	A record has been encountered that does not contain an entry with the Current MEBx Password.
33	The given buffer length is invalid.
34	The header chunk count cannot contain all of the setup file header data.
35	The record chunk count cannot contain all of the setup file record data.
36	The requested index is invalid.
37	The setup file header indicates that there are no valid records.
38	The given buffer is invalid.
39	A record entry with an invalid Module ID was encountered.
40	A record was encountered with an invalid record number.



41	The setup file header contains an invalid module ID list.
42	The setup file header contains an invalid byte count.
43	The setup file record id is invalid.
44	The list of data record entries is invalid.
45	Failed to write to the given file.
46	Failed to read from the given file.
47	Failed to create random numbers.
48	The CurrentMEBx password is invalid.
49	The NewMEBx password is invalid.
50	The PID is invalid.
51	The PPS is invalid.
52	The data record is missing a CurrentMEBx password entry.
53	The data record is missing a NewMEBx password entry.
54	The data record is missing a PID entry.
55	The data record is missing a PPS entry.



10 EEUpdate

EEUpdate is used to program the MAC address after the firmware image has already been programmed onto the flash device.

10.1 Requirements

EEUpdate runs in DOS (MS-DOS 6.22, Windows 98 DOS). It is a command-line tool that must be ran on the system that requires the updated ethernet MAC address

10.2 Usage

The number of options available in EEUpdate is quite extensive. For more detail on all of these options please refer to the readme.txt file. The executable can be invoked by:

```
eeupdate.exe [options]
```

- **<Blank>** – running EEUpdate without any options will the devices that can be updated
- **/nic = xx** – This will select the nic device to update where “xx” is the number reported by EEUpdate.
- **/mac=<MAC Address>** - This will update the ethernet MAC address of the device specified by the /nic option. The ethernet MAC address must contain exactly 12 hexadecimal digits without spaces or dashes.

Note: The /mngmac parameter will **NOT** program the Manageability MAC address. This parameter will only program the Manageability Mac address for Broadwater and Lakeport platforms.

10.3 Example

The example below will update the ethernet MAC address of device 1. This is the most common usage for the EEUpdate tool.

```
Eeupdate.exe /nic=1 /mac=001122334455
```



§



11 FWUpdLcl

FWUpdLcl allows the end user IT Administrator to update part of the system's firmware without having to reprogram the entire flash device, and verifies that the update was successful.

FWUpdLcl does not update the BIOS, GbE or descriptor region. It only updates the firmware code portion that Intel provides on the OEM website.

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, **CL_ICH8_REL_IAMT_BYP_ME_UPD.BIN**, is located in the kit's NVM image folder.

Note: After updating the firmware, local firmware updates will be disabled. The admin will need to re-enable local firmware updates if future updates are required.

11.1 Requirements

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

FWUpdLcl can be run on machines with one of the following operating systems:

- **DOS** (MS-DOS 6.22, Windows 98 DOS, FreeDOS, DRMK DOS)
- **Windows** (Windows XP SP2, x64, PE, Vista)

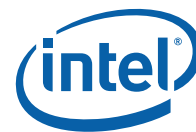
The requirements for running FWUpdLcl depend on the Intel® AMT system's operating system. The requirements also depend on the type of Intel® AMT connection desired (secure or a non-secure).

11.2 Non-Secure Dos Requirements

- User needs host administrator access

11.3 Non-Secure Windows Requirements

- ME Firmware Local Update must be enabled in the MEBX
- In the ME BIOS Extension, Intel® AMT must be selected in the "Manageability Feature Selection" menu.
- Intel® Management Engine Interface (MEI) driver must be installed.



11.4 Secure Windows Requirements

- In the ME BIOS Extension, Intel® AMT must be selected in the “Manageability Feature Selection” menu.
- In the Intel® AMT Configuration menu (in the Intel® AMT BIOS extension), **Secure Firmware Update** must be enabled.
- The Intel® AMT Local Manageability Service (LMS) must be installed.

11.5 Windows* PE Requirements

- The Intel® Management Engine Interface (MEI) driver must be installed in the Windows PE image.
- The Windows PE image is WMI-enabled.

11.6 Enabling and Disabling Local Firmware Update

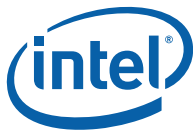
Disabling **Firmware Local Update** in the ME Bios Extension prevents updating the firmware. However, even if **Firmware Local Update** is disabled, you can still enable updating the firmware for a limited number of times. This is done during manufacturing. To do this, you configure the two variables **Local FWU Override Counter** and **Local Firmware Override Qualifier** to temporarily override the ME BIOS Extension settings. These parameters can be modified by using the AMTNVM tool on the full firmware image file.

When **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 settings. After the flash is programmed, each time the machine 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 by the ME BIOS Extension 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 **Local FWU Override Counter** is set to -1 and **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 11 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 <u>NOT</u> Allowed	Local Firmware Updates <u>NOT</u> Allowed	Local Firmware Updates <u>NOT</u> Allowed
Local FWU Override Counter = -1 (minus one)	Local Firmware Updates Allowed	Local Firmware Updates <u>NOT</u> Allowed	Local Firmware Updates Allowed only until Intel® AMT is configured
Local FWU Override Counter = 0 < n < 255	Local Firmware Updates Allowed	Local Firmware Updates Allowed	Local Firmware Updates Allowed

If **Local FWU Override Qualifier** is set to 2 (two) and Local FWU Override counter is set to -1 (minus one), local firmware updates will be allowed until Intel® AMT has completed the setup and configuration process. After the setup and configuration process is complete, local firmware updates will be **NOT** allowed.

11.7 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 Flash Image tool to program the entire flash memory). To differentiate between the image files used for updating and those used for programming the entire flash memory, note that the files used by FWUpdLcl for updating include the string **UPD** in their file names.

Note: After updating the firmware, local firmware updates will be disabled. The admin will need to re-enable local firmware updates if future updates are required.

The executable can be invoked by:

```
FWUpdLcl.exe <Image File>
```

- **Image File** – Image file of the firmware to be updated. This image file is not the same image file used in the Flash Image tool.

11.8 Usage Windows* 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 Flash Image tool to program the entire flash memory). To differentiate between the image files used for updating and those used for programming the entire flash memory, note that the files used by FWUpdLcl for updating include the string **UPD** in their file names.

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



- **Image File** – Image file of the firmware to be updated. This image file is not the same image file used in the Flash Image tool.
- **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>** – This is the hostname of the firmware. If TLS is used this must be the name of the TLS certificate in the firmware.
 - **- Cert <Certificate>** – The name of the certificate used if TLS mutual Authentication mode is enabled.
 - **- generic** – This option will update the firmware without credentials even if the system is setup and configured already. If this option is used all other options are ignored.

11.9 Examples

Example 1:

```
FWUpdLcl.exe CL_ICH8_REL_IAMT_BYP_ME_UPD.BIN -user Admin -pass Admin`12 -
TLS -host Cert_Name
```

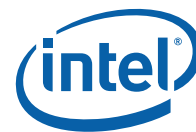
The above will update the local firmware using TLS connection to the firmware. The Certificate name (Cert_Name) matches the certificate name provided in the Firmware.

Example 2:

```
FWUpdLcl.exe CL_ICH8_REL_IAMT_BYP_ME.BIN -user admin -pass Admin`12
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.





12 AMTSD

AMTSD demonstrates the System Defense feature of an Intel® AMT machine. The System Defense feature will help protect the system and the network from viruses, worms and other malicious code.

12.1 Requirements

AMTSD runs on a Windows (Windows XP SP1/2, 2003) machine. It is a command-line executable and can be run on a remote machine connected to an Intel® AMT machine.

12.2 Usage

If the policy is invoked twice, the user will receive the error message: "Error: A tool policy already exists. Please use the cleanup option first. AMTSD ended successfully". The user should use the "-cleanup" option before trying to invoke the policy. The "-cleanup" option will allow communication on **BOTH** the wired and wireless connection.

AMTSD.exe [options]

Options are mentioned below. If Kerberos authentication is used, the username and password should not be entered

- **-host <Hostname/IP address>** – IP Address(Intel® AMT IP Address) or Host Name (displayed in the MEBX) of the Intel® AMT enabled system. If TLS mode the Hostname must be used
- **-user <Username>** – Username used to access the Intel® AMT enabled system. If Kerberos authentication is used, the username and password should not be entered.
- **-pass <Password>** – Password used in conjunction with the username to access the Intel® AMT enabled system. If Kerberos authentication is used, the username and password should not be entered
- **-TLS** – Must be entered if TLS mode is enabled.
- **-cert <certificate>** – The common name of the client certificate is provided here. The certificate is used only when Mutual Authentication is enabled and in TLS mode. If TLS is used the hostname provided must be the same as the hostname used in the TLS certificate in the firmware.
- **-interface [LAN | WLAN | both]** – Specifies the interface that the policy is applied to. WLAN and both options will only work if the machine has a WLAN



adapter and a wireless profile that has been configured with Intel® AMT. Please see the *OEM WEB UI Guide* for more information. Default value is LAN

- **-cleanup** – Deactivate the tool policy and complete the System Defense test. If the test is prematurely stopped and System Defense filter remains active, this option will complete the test and allow network connectivity for **BOTH** wired and wireless connection. This option should **NOT** be used with the “-interface” option.

12.3 Execution Flow

AMTSD configures a System Defense policy that will cause System Defense to block all incoming traffic. The following procedure is used:

- AMTSD creates a System Defense policy named “IntelCBTest”. The policy instructs System Defense to cut off all network traffic if any non- Intel® AMT network traffic is received.
- The user continuously pings the host machine’s IP address. The IP address is specified in dotted decimal notation ddd.ddd.ddd.ddd.

```
ping -t ddd.ddd.ddd.ddd
```

- AMTSD prompts the user to activate the “IntelCBTest” policy.
- The first message which arrives after the policy is set will cause the filter to match.
- System Defense will block all incoming traffic.
- Ping responses will stop.
- The tool will prompt the user to cleanup the policy. The tool supports a “cleanup” mode to cleanup the policy. If the tools clean up option is used, the clean up option will restore communication to **BOTH** wired and wireless connections.
- Ping responses return.



13 AMTFeaturesLocal

AMTFeaturesLocal is used to verify the functionality of local Intel® AMT storage. Note that the Web GUI, AMTFeaturesLocal tool, and AMTFeaturesRemote tool do not cover checking of all features of Intel® AMT. The feature(s) checked by the AMTFeaturesLocal tool are:

- Local access to NVM Storage

Components that must be working correctly in order to perform this test:

- LMS
- Intel® Management Engine Interface driver

Note: These features cannot be tested directly.

13.1 Requirements

AMTFeaturesLocal runs on a Windows (Win2K SP4, XP SP1/2, Vista 32bit/64bit) machine. It is a command-line executable and must be run locally on an Intel® AMT machine. AMTFeaturesLocal (like Intel® AMT) needs the following services to be installed: Intel® AMT Local Manageability Service (LMS) and the Intel Management Engine Interface driver.

- The user must have Administrator authorization (MEInfoWin Only)

13.2 Usage

The executable can be invoked by:

```
AMTFeaturesLocal.exe [-user <username>] [-pass <password>] [-TLS] [-cert  
<certificate>]
```

- **-user <username>** – Username to authenticate with for access to Intel® AMT features on local machine. If Kerberos authentication is used, the username and password should not be entered.
- **-pass <password>** – Password corresponding to username to authenticate with for access to Intel® AMT features on local machine. If Kerberos authentication is used, the username and password should not be entered.
- **-TLS** – Must be entered if TLS mode is enabled. If TLS mode is enabled, AMTFeaturesLocal will prompt the user for the host name. If TLS is used the



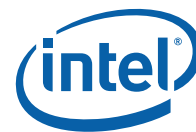
hostname provided must be the same as the hostname used in the TLS certificate in the firmware.

- **-cert <certificate>** – The common name of the client certificate is provided here. The certificate is used only when Mutual Authentication is enabled and in TLS mode. If TLS is used the hostname provided must be the same as the hostname used in the TLS certificate in the firmware.

AMTFeaturesLocal will return success or failure based on the availability of the tested features. In case of failure, AMTFeaturesLocal will give a meaningful error message.

PTSDK_STATUS_HARDWARE_ACCESS_ERROR: The Library has identified a HW Internal error.

§



14 AMTFeaturesRemote

AMTFeaturesRemote is used to check the functionality of a remote Intel® AMT feature which may not be available through the Web GUI. Note that the Web GUI, AMTFeaturesLocal tool, and AMTFeaturesRemote tool do not cover checking of all features of Intel® AMT. The feature(s) checked by the AMTFeaturesRemote tool are:

- Event Manager (Note that access to the Event Log must be on NVM, not through listener)
- Remote Control (boot optional)
- HW Assets
- Network Admin
- Security Admin
- System Defense
- Storage and Storage Admin
- Agent Presence Remote Interface
- Wireless Configuration

14.1 Requirements

AMTFeaturesRemote runs on a Windows (Windows XP SP1/2, Vista 32bit/64bit) machine. It is a command-line executable and can be run on a remote machine connected to an Intel® AMT machine.

14.2 Usage

The executable can be invoked by:

```
AMTFeaturesRemote.exe [-host <Hostname/IP>] [-user <username>] [-pass  
  <password>] [-TLS] [-cert <certificate>] [-feat ALL| STO| HWINV| RC| CB| EM|  
  NA| SA| AP| WC] [boot]
```

- **-host <Hostname/IP>** – The Hostname(Hostname displayed in the MEBX) or the IP address(Intel® AMT IP address) of the remote Intel® AMT machine specified in a dotted decimal notation ddd.ddd.ddd.ddd (e.g. 134.176.185.2). If TLS mode is used the Hostname must be used



- **-user <username>** – Username to authenticate with for access to Intel® AMT features on remote machine. If Kerberos authentication is used, the username and password should not be entered.
- **-pass <password>** – Password corresponding to username to authenticate with for access to Intel® AMT features on remote machine. If Kerberos authentication is used, the username and password should not be entered.
- **-TLS** – Must be entered if TLS mode is enabled.
- **-cert <certificate>** – The common name of the client certificate is provided here. The certificate is used only when Mutual Authentication is enabled and in TLS mode. If TLS is used the hostname provided must be the same as the hostname used in the TLS certificate in the firmware.
- **- feat** – Run the feature to be tested. Only one option may be used with each call to the program.
 - **ALL** – Check all features
 - **STO** – Tests the Storage and Storage Administration SOAP interfaces. Tests writing to and reading from the flash.
Note: that if there is EACL entries with "Enterprise Name = Intel2", the "sto" option will erase this entry.
 - **HWINV** – Enumerates the hardware assets types and retrieves hardware details.
Note: When additional asset data information is not captured by the Intel® AMT device due to container space limitation, an appropriate warning will be displayed.
 - **CB** – Check only the System Defense feature. Enumerates System Defense filters and policies.
 - **EM** – Reads the event log records and log timestamp. **Note:** A warning will be displayed in case no event entries exist and boot
 - **RC** – Get the current and supported power states of the Intel® AMT system
 - **NA** – Get TCP/IP parameters
 - **SA** – Get firmware and Configuration mode
 - **AP** – Enumerate console watch dog timers
 - **WC** – Test Wireless profiles that are currently added
- **Boot** – When this option is used in conjunction with the "all" or the "rc" option, the system will perform a power cycle test.



14.2.1 Errors

AMTFeaturesRemote will return success or failure based on the availability of the tested feature. In case of failure, AMTFeaturesRemote will give a meaningful error message as seen below.

Error: failed while calling Add Storage Eacl Entry
Status = 4103
PTSDK_STATUS_NETWORK_ERROR: A network or authentication error has occurred while processing the call.

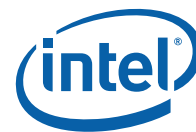
The "Status" value returns a zero ("0") if all tests succeeded. Upon failure a non-zero value (such as shown above) is returned.

14.2.2 Warnings

Some of the commands may result in a warning. A warning, unlike an error, will not end the execution of the tool and the tool will continue working. For example, no events in event log, no storage space available. The tool will have an option to suppress warnings (the default leaves warnings on).

Warning: Event log time is not set.





15 MEInfo

MEInfoWin and MEInfo provide a simple aliveness test of the ME FW. Both tools perform the same test, query the Intel® AMT FW and retrieve data. Below is a list of the data that each tool will return.

15.1 Requirements

MEInfo runs on MS-DOS* 6.22, Windows 98 DOS, FreeDOS, DRMK DOS, or Windows (Windows XP SP1/2, Vista 32/64). MEInfo and MEInfoWin is a command-line executable.

- Intel® AMT must be the selected "Manageability Feature Selection" in the MEBx
- Intel® Management Engine Interface and the LMS drivers to be installed (MEInfoWin only)
- The user must have Administrator authorization (MEInfoWin Only)

15.2 Windows* PE requirements

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

- The Intel® MEI 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. The LMS driver can not be installed on Windows PE.

15.3 Usage

The executable can be invoked by:

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

- **feat <feature name> <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/ value must be enclosed in quotation marks. If the values are identical a message will display



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. Any value that is returned by MEInfo can be validated, please see the list below. When verifying a MAC address, the MAC Address can be verified with or without dashes. Both entry methods will have the same result.

- Retrieves the following information:
 - MEInfo tool version
 - BIOS version
 - Intel® AMT firmware version – Displays a list of parameters specific to Firmware version currently in use. Such as: Flash, NetStack, AMTApps, Intel® AMT, SKU, Vendor ID, Build Number, Recovery Version, Recovery Build Num, Legacy Mode. Some of the descriptions can be found below:
 - SKU – Displays which Firmware SKU is currently in use in plain text. (AMT/ASF/FSC/CB/CRYPTO) The value is decimal representation of the following bit-field, indicating the features supported by the firmware SKU. The SKU for Intel® AMT is 12.
 - Intel® AMT mode setting – Displays 1.0 (2005) or 2.0 (2006) or 2.5 (2007)
 - Link status – Displays if the system is connected to and has network connectivity. (Link up or Link down)
 - Cryptography fuse state – Enabled or Disabled
 - Flash protection state – Enabled or Disabled
 - Last ME reset reason – Displays reason for last ME reset in plain text. (Power up, Firmware reset or Global system reset)
 - Setup and Configuration state – Displays if the system has been setup and configured. (Not Started, In process or Completed)
 - BIOS Mode – Displays the state of the BIOS (Pre Boot, In Boot, or Post Boot)
 - Dedicated Mac Address: Displays the Intel® AMT MAC Address
 - Host Mac Address: Displays the ethernet MAC Address
 - FWU Override Counter: Displays override counter for local firmware updates.
 - FWU Override Qualifier: Displays the override qualifier for local firmware updates.
 - FW on Flash Desc Override: Displays the Descriptor override for flash protection.



- Kedron Driver Version (Windows version only)
- Kedron Hardware Version (Windows version only)
- UNS version (Windows version only)
- LMS version (Windows version only)
- Intel® Management Engine Interface driver version (Windows version only)

15.4 Examples

Example 1: This is the simple firmware aliveness test that will return machine specific parameters. The output is from the windows version. The DOS version will not display the Kedron driver version, Kedron HW version, UNS version, Intel Management Engine Interface or LMS version numbers.

```
C:\ MEInfoWin.exe

Intel® MEInfo Win Version: 2.5.0.1052

BIOS Version:          SROSA001.86C.0024.D.0611081602

Intel® AMT code versions:

Flash:                2.5.0
Netstack:             2.5.0
Apps:                 2.5.0
Intel® AMT:           2.5.0
Sku:                  0
VendorID:             8086
Build Number:         1052
Recovery Version:     2.5.0
Recovery Build Num:   1052
Legacy Mode:          False

Link status:           Link up
Cryptography fuse:     Enabled
Flash protection:      Enabled
Last reset reason:     Power up
Setup and Configuration: Completed
BIOS Mode:             Post Boot
Dedicated Mac Address: 00-11-22-33-44-55
Host Mac Address:      00-AA-BB-CC-DD-55
FWU Override Counter:  Never
FWU Override Qualifier: Always
FW on Flash Desc Override: Disable
Kedron Driver Version: Not Available
Kedron HW Version:     Not Available
UNS Version:           2.5.3.1051
```



LMS Version:	2.5.5.1051
MEI Version:	2.5.22.1051

Example 2: This example compares the flash version stored on the flash device with the command line argument. The value provided does not match the value stored, so the value stored is displayed.

```
C:\ MEInfo.exe -feat flash -value 2.0.0
```

Failed. The values are not identical.

Flash: 2.5.0

Example 3: This example checks if the system has completed the setup and configuration process

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

Success. The values are identical

Example 4: This example checks to verify the Manageability MAC address is correct. The MAC address entered can be entered with or without dashes. Both options will have the same result.

```
C:\ MEInfo.exe -feat "Dedicated MAC Address" -value "554433221100"
```

Success. The values are identical

Example 5: This example checks to verify the Host MAC address is correct. The MAC address entered can be entered with or without dashes. Both options will have the same result.

```
C:\ MEInfo.exe -feat "Host MAC Address" -value "00-AA-BB-CC-DD-55"
```

Success. The values are identical

15.5 Error Codes

The error codes for the windows version fall into 2 categories, Application errors codes and Firmware error codes. A success has a return code of 0 and an appropriate success message will be displayed for both versions

Table 12 MEInfo Error Codes

Windows version error codes:

Application error codes:

2059	PT_STATUS_EVENTLOG_FROZEN: Event Log operation failed due to the current freeze status of the log.
4099	PTSDK_STATUS_INVALID_PARAM: One of the parameters is invalid (usually indicates a NULL pointer or an invalid session handle is specified).
4096	PTSDK_STATUS_INTERNAL_ERROR: Intel® AMT has identified an internal HW error.
4101	PTSDK_STATUS_HARDWARE_ACCESS_ERROR: The Library has identified a HW Internal error.

FW error codes returned by the interface:

2	AMT_STATUS_NOT_READY: Intel® AMT device has not progressed far enough in its initialization to process the command.
4	AMT_STATUS_INVALID_MESSAGE_LENGTH: Length field of header is invalid.
3	AMT_STATUS_INVALID_AMT_MODE: Command is not permitted in current operating mode.
11	PT_STATUS_NOT_ENOUGH_STORAGE: The requested number of bytes cannot be allocated in ISV storage.
33	AMT_STATUS_UNSUPPORTED_OBJECT: Unsupported object.

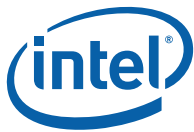
DOS version error codes:

FW error codes returned by the interface:

2	AMT_STATUS_NOT_READY : AMT device not ready
3	AMT_STATUS_INVALID_AMT_MODE : Command is not permitted in current operating mode
4	AMT_STATUS_INVALID_MESSAGE_LENGTH : Length of field header is invalid

Application error codes:

0x200a	MEI_STATUS_TIMEOUT_ERROR : AMT Device Unavailable
0x2001	MEI_STATUS_LOCATE_DEVICE_ERROR : Cannot locate ME Device
0x2002	MEI_STATUS_MEMORY_ACCESS_ERROR : Memory access failure
0x2003	MEI_STATUS_WRITE_REGISTER_ERROR : Write register failure
0x2004	MEI_STATUS_MEMORY_ALLOCATION_ERROR : Cannot allocate memory
0x2005	MEI_STATUS_BUFFER_OVERFLOW_ERROR : Circular buffer overflow
0x2006	MEI_STATUS_NOT_ENOUGH_MEMORY : Not enough memory in circular buffer
0x2007	MEI_STATUS_MSG_TRANSMISSION_ERROR : ME device not ready for transmission
0x2008	MEI_STATUS_VERSION_MISMATCH : Unsupported Intel® MEI bus message protocol
0x200b	MEI_STATUS_UNEXPECTED_RESPONSE : Unexpected ME device response
0x200c	MEI_STATUS_UNKNOWN_MESSAGE : Unsupported message type
0x2010	MEI_STATUS_NO_FREE_CONNECTION : No free connection available



0x200a	MEI_STATUS_TIMEOUT_ERROR : AMT Device Unavailable
0x2013	Intel® _STATUS_NO_MESSAGE : No message
0x2016	MEI_STATUS_BUFFER_NOT_EMPTY : Circular buffer not empty
0x2017	AMT_STATUS_INTERNAL_ERROR : AMT device internal error

§



16 AMTRedirection

AMTRedirection will allow the use of IDER and SOL through an interactive menu.

IDE Redirection (IDER) emulates an IDE floppy or CD drive over a standard network connection. IDER enables a management machine to attach one of its floppy or CD drives to a managed client over the network. Once an IDER session is established, the managed client can use the IDE device as if it were directly attached to one of its own IDE channels. This can be used for remotely booting an otherwise unresponsive system. IDER can transfer data from an internal IDE CD-ROM drive, from a standard internal 1.44MByte floppy drive, or from an LS-120 high capacity drive with either a 120 MB disk or a standard 1.44 MB disk. IDER does not support DVD format.

Serial Over LAN (SOL) is the ability to emulate serial port communication over a standard network connection. SOL can be used for most management applications where a local serial port connection would normally be required. When an active SOL session is established between an Intel® AMT enabled client and a management console using the Intel® AMT redirection library, the client's serial traffic is redirected through Intel® AMT over the LAN connection and made available to the management console. Similarly, the management console may send serial data over the LAN connection that appears to have come through the client's serial port.

Only *.IMG files are supported for floppy disk image files and *.ISO files are supported for CD ROM image files.

For more information regarding the SOL/IDER, see Intel® AMT SDK documentation.

Note: If an SOL or IDER session is open, the **Turn Power off** Command from the WEB UI will **not** work. If the user attempts to send the **Turn Power off** command an Invalid remote control error message will be displayed.

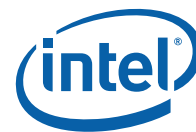
16.1 Requirements

Intel® AMT must be provisioned prior to running the AMTRedirection tool. When configuring Intel® AMT in ME Bios extension, SOL and IDER features should be enabled in the appropriate submenu.

In addition, LMS and SOL drivers should be installed on Intel® AMT host.

AMTRedirection requires the following files to be located in the same folder as the executable:

- Imrsdk.dll
- MC.ini
- SOLHTcfg.ht



- StatusStrings.dll

16.2 Usage

The executable can be invoked by:

```
AMTRedirection.exe [-host <hostname>] [-user <username> -pass <password>] [-  
TLS] [-cert <certificate>]
```

- **-host <hostname>** – This is the hostname(As displayed in the MEBX) of the system or the IP address(Intel® AMT IP address) of the remote Intel® AMT machine specified in a dotted decimal notation ddd.ddd.ddd.ddd (e.g. 134.176.185.2). In TLS mode the hostname must be used.
- **-user <username>** – Username to authenticate with for access to Intel® AMT features on remote machine. If Kerberos authentication is used, the username and password should not be entered.
- **-pass <password>** – Password corresponding to username to authenticate with for access to Intel® AMT features on remote machine. If Kerberos authentication is used, the username and password should not be entered.
- **-TLS** – Must be entered if TLS mode is enabled. To verify IDE-R and SOL in TLS mode the Intel® AMT system needs to be configured in TLS mode. Please see below for details to prepare iamtredirection to use TLS mode.
- **-cert <certificate>** – The common name of the client certificate is provided here. The certificate is used only when Mutual Authentication is enabled and in TLS mode. If TLS is used the hostname provided must be the same as the hostname used in the TLS certificate in the firmware.

After entering the above line into a command line prompt, the user will be asked to make a selection from the following menu:

```
a: Open SOL Session  
b: Close SOL Session  
c: Open IDER Session  
d: Close IDER Session  
e: Regular boot.  
f: SOL boot.  
g: SOL boot to BIOS setup.  
h: IDER Floppy boot.  
i: IDER CD boot.  
j: SOL + IDER Floppy boot.  
k: SOL + IDER CD boot.  
m: Enable Redirection listener  
n: Disable Redirection listener  
l: Display Menu Option  
x: Exit
```

```
choose an option>
```



Pressing an invalid entry or the letter “L” will display the menu.

A: Open SOL Session: This will invoke windows to open a telnet window. Through this window all text based information from the Intel® AMT enabled system will be viewable in this window. Only one SOL session may be open at any time.

Note: The Redirection listener must be enabled before opening an SOL session in Enterprise mode.

B: Close SOL Session: Will close the SOL session that was previously opened.

C: Open IDER Session: When this option is invoked, the user will be prompted to choose disk or image for floppy device. When floppy disk is chosen, the appropriate drive letter should be entered. If image file is chosen, the appropriate filename with relative path of image file should be entered. This is followed by the same queries for CD-ROM device, where if CD is chosen, the appropriate drive letter should be entered. If image file is chosen, the appropriate filename with relative path of image file should be entered.

Note: opening IDER session with AMTRedirection automatically exposes the virtual CD/Floppy devices to the host by enabling IDE Client registers. One should “Scan for hardware changes” in Device Manager to actually see and use these devices in My Computer. Only one IDER session may be open at any time using this particular instance of the program.

Only *.IMG files are supported for floppy disk image files and *.ISO files are supported for CD ROM image files.

D: Close IDER Session: Will close the IDE-R session that was previously opened.

Note: Closing the IDER session leaves the Client IDE registers enabled, i.e. the virtual devices are still exposed to the host even though there is no open IDER session.

E: Regular Boot: Will cause the Intel® AMT enabled system to perform a regular boot as described in the BIOS.

F: SOL Boot: Will cause the remote session to perform a regular boot with the SOL option. An SOL session must be open before using this menu option.

G: SOL Boot to BIOS Setup: Will cause the remote system to boot directly to the BIOS Screen with SOL. An SOL session must be open before using this menu option.

H: IDER Floppy Boot: Will cause the remote machine to boot from the floppy device in the management console. An IDE-R session must be open before using this menu option.

I: IDER CD Boot: Will cause the remote machine to boot from the CD ROM device in the management console. An IDE-R session must be open before using this menu option.

J: SOL + IDER Floppy boot: This will cause the Intel® AMT host to boot from the bootable remote floppy disk/image in the Management Console. In addition, text and keyboard redirection will be enabled, i.e. text-based information will be redirected from Intel® AMT machine to Management Console, as well as keyboard redirection



from Management Console to Intel® AMT machine can be performed (pressing Del to enter BIOS for example). Both SOL and IDER sessions must be opened before using this option.

K: SOL + IDER CD boot: This will cause the Intel® AMT host to boot from the bootable remote CD disk/image in the Management Console. In addition, text and keyboard redirection will be enabled, i.e. text-based information will be redirected from Intel® AMT machine to Management Console, as well as keyboard redirection from Management Console to Intel® AMT machine can be performed (pressing Del to enter BIOS for example). Both SOL and IDER sessions must be opened before using this option.

M: Enable Redirection Listener: This will enable the Redirection service listener.

Note: By default, the Redirection listener is disabled in Enterprise mode and enabled in Small Business mode.

N: Disable Redirection Listener: This will disable the Redirection service listener.

Note: By default, the Redirection listener is disabled in Enterprise mode and enabled in Small Business mode.

L: Display Menu Option: Display the above menu option.

X: Exit: Will exit the menu option and close any open sessions.

Any Other Option: Will display the above menu options.

16.3 Verify SOL/IDE-R in TLS mode

TLS mode requires TLS certificates installed on the Intel® AMT system. The certificates from the Setup and Configuration tool can be used please refer to the Setup and Configuration User Guide for more information

The setup and configuration application must configure the Intel® AMT system in TLS mutual mode.

To use the certificates from the Setup and Configuration, the certificates will need to be installed from the Setup and Configuration application. These certificates will then need to be copied into the iamtr redirection folder after the certificates have been installed by the Setup and Configuration tool. The list below shows all of the certificates that need to be inside the iamtr redirection folder:

- cacert.cer
- truseted_cert.cer
- remote_client.cer
- remote_client_pem (see below)

To install the certificates, the user should run the **checkcs.bat** file located in the "\\AMTConfiguration\\CertGenerator\\ClientSecScripts" folder. This bat file will create three folders in the same directory labeled, "local_client", "remote_client" and "trusted_rootCA"



The file `remote_client.p12` located in "`iamtconfiguration\CertGenerator\remote_client\`" folder should be copied into the "`iamtconfiguration\Certgenerator\OpenSSL\`" folder. Running `Openssl.exe` will create the modified certificate file to use with AMTRedirection. After the `remote_client.pem` file is modified, it should be copied to the `iamtredirection` folder. The tool will prompt the user for the old password and a new password. The password for this file is "qwerty", The new password can be any value/character. The example below has the password set as "1234"

```
OpenSSL.exe pkcs12 -in "remote_client.p12" -out remote_client.pem
```

To use the certificate just created, the following text should be added to `mc.ini` file located in the `iamtredirection` folder.

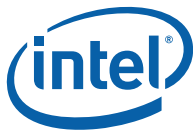
```
[SECURITY]
CA_FILE = cacert.cer
PRIVATE_CERT = remote_client.pem
CERT_PASS = 1234
```

`iamtredirection` is now ready to be tested in TLS mode. The example below shows the tool being executed in TLS mode. If Kerberos is being used the username and password should not be supplied

```
iamtredirection 192.168.1.11 <user> <password> -TLS
```

§





17 Setup and Configuration Application (Configuration Server)

Before management applications can access an Intel® AMT device, the device must be populated with various configuration settings such as usernames, passwords, network parameters, Transport Layer Security (TLS) certificates, and keys necessary for secure communications. This can be done in a couple of ways. The system can be setup up manually as described in the OEM Bring Up Guide or the setup can be done through the use of a Setup and Configuration Application. For more details on full functionality of the application please refer to the Setup and Configuration User Guide located in the same folder as the executable (iAMT Tools\iamtConfiguration \).

17.1 Tool Requirements

Setup and Configuration application runs on Windows Server 2000 and Windows Server 2003 through a command line interface.

- Mutual authentication requires that an Intel® AMT device have a **trusted_root** certificate installed.
- If a DHCP server services are not available then the Intel® AMT device must be configured to use static
- If a DNS server is unavailable, then the SCS IP address must be explicitly set through the MEBx or AMTNVM

All of the files that are included in the subdirectories add features to the configuration server. For more details on all of these features please refer to the Setup and Configuration Users Guide. At a minimum, the following files are required to run:

ConfigurationServer.exe – The Setup and Configuration executable.

BAT Files – Before the configuration server can run, certain files need to be present. These BAT files check for the presence of these files, hence all of the BAT files included in the folders are required for the Server to function. For more details on all of these features please refer to the Setup and Configuration Users Guide

Library files – The following is a list of library files that must be in the same folder as the Setup and Configuration executable: *libeay32.dll*, *msvcr71.dll*, *ssleay32.dll*



psk.repository.xml – A file that contains a set of PID-PPS key pairs for each Intel® AMT platform. The PID is an eight character entry of the form: XXXX-XXXX. The PPS is a thirty-two character quantity of the form: AAAA-BBBB-CCCC-DDDD-EEEE-FFFF-GGGG-HHHH. A sample of the file can be found in the "Configuration\ConfigScripts" subdirectory in the Setup and Configuration Application folder. For more details on this file please see the Setup and Configuration Users Guide

default.conf.xml – A file that contains the desired configuration settings for any Intel® AMT devices to be configured by the SCA. These settings will be applied to all instances of Intel® AMT unless the user creates a unique file for each device. The unique file should be named <UUID>.conf.xml where <UUID> is the actual UUID of the Intel® AMT device. A sample of the file can be found in the "Configuration\ConfigScripts" subdirectory in the Setup and Configuration Application folder. For more details on this file please see the Setup and Configuration Users Guide

17.2 Usage

This tool contains a detailed user guide located with the tool executable. The section below is only a brief summary of the tool.

The application will check for certain signed certificates when executed. If these signed certificates are not present, the application can create signed demo certificates. The application will function correctly with the demo certificates. To create the demo certificates simply agree when prompted by the application. For more details on this file please see the Setup and Configuration Users Guide

This tool does not support the use of multiple client certificates installed on the local host. The certificates used to install the program should not be used on other systems. If the user already has the appropriate certificates, it is unnecessary to install the demo certificates.

```
ConfigurationServer.exe [-port <Port Number>]
```

-port <Port Number> - The port number is the listening port number that the Intel® AMT systems are configured. The sample SCA can be started without any parameters. In this case the default port will be used for listening. The listening port should match the one configured on the Intel® AMT device during setup. By default, port 9971 is used to establish a connection to the Setup and Configuration Application.

17.3 PSK/PID File Format

The sample PSK/PID file (PSK.Repository.xml) is located under the "Configuration\ConfigScripts" subdirectory in the Setup and Configuration Application folder along with the **PskGenerator.exe**.

PskGenerator.exe will output 1 PID-PPS pair directly to the screen. This pair should then be placed in the PSK.REPOSITORY.XML file in format mentioned below. Each Intel



Intel® AMT system needs a unique pair to establish a secure connection to be setup through the application.

Platform OEMs may pre-configure Intel® AMT devices with PID/PPS pairs. The repository will be based on a file delivered by the OEM.

Table 13. PSKRepository.xml format

Variable name	Allowed settings	Usage
<pairs>	<pre><pair> <pid>xxxx-xxxx</pid> <pps>xxxx-xxxx-xxxx-xxxx- xxxx-xxxx-xxxx-xxxx</pps> </pair></pre>	Used to define a PID-PPS key pair. This same PID-PPS should be used during the Factory Mode setup of each Intel® AMT device

17.4 Sample TLS Certificates

The setup and configuration application comes with sample TLS certificates. These are only samples and should not be re-distributed.

The sample certificates are not distributed in the build, but can be installed by running the command checkcs.bat located in

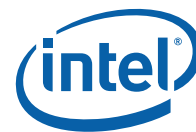
"iamtconfiguration\CertGenerator\ClientSecScripts\" folder. To enable TLS the user should modify default.config.xml or <UUID>.config.xml. These files contain the parameters used by the Setup and Configuration Application.

17.5 Configuring an Intel® AMT system in Enterprise mode

Intel® AMT can be configured in two modes; Small Business and Enterprise. Small business mode can be configured through the MEBX. Enterprise mode requires a Setup and Configuration application. The Setup and Configuration application can be provided by an ISV. The setup and configuration application provided in the validation tools verifies the Intel® AMT system can be Setup and Configured by an application. The system should be tested with ISV software as well. Setting up a system in enterprise requires the following steps:

- Modify PSK repository
- Configure setup parameters file (default.conf.xml)
- Prepare the Intel® AMT system
- Install the sample certificates for the Setup and Configuration Application and run the application

Enterprise mode requires the Setup and Configuration Application to be running on a dedicated system on the same network as the Intel® AMT system



17.5.1 Modify PSK Repository

The PSK repository file (PSK.repository.XML) contains the PID/PPS pairs for each Intel® AMT system the needs to be configured by the Setup and Configuration Application. The format should be maintained for all entries that are added (see Table 13)

The PSK Generator (PSKGenerator.exe) will generate different PID/PPS pairs that can be inserted into the PSK repository file (PSK.repository.XML). USBFile.exe will also create an XML file with PID/PPS pairs. The XML file generated by this tool can be copied and pasted into the PSK.repository.xml.

17.5.2 USBfile.exe

USBfile.exe creates a bin file and an XML file for USB key setup and configuration. However the binary file created can be used for FAUPD and the XML file can be used for the PSK Repository. The usage is as follows:

```
USBfile.exe -Create <Bin File> <XML file> <user name> <Password> <Num>
USBFile.exe -View <Bin File>
```

- **-Create** – This option will create a new bin file and a new XML file to be used for USB key setup and configuration or for the PSK file for FAUPD.
 - **<Bin File>** - The name of the bin file to be created.
 - **<XML File>** - The name of the XML file to be created
 - **<user name>** - The admin user name that is used for USB key setup and configuration.
 - **<Password>** - The Password that is used during the USB key setup and configuration.
 - **<Num>** - A number in decimal format that determines the number of PID/PPS pairs to create in the XML and bin file.
- **-View** – This option will output the contents of a bin file to the screen
 - **<Bin File>** - The name of the bin file to display

17.5.3 Configure Setup Parameters

The setup parameters file need to be modified for each system. The format to use is <UUID>.conf.xml, where <UUID> is the Intel® AMT system's UUID.

If the UUID of the system does not find a unique configuration file, the default configuration file will be used (default.conf.xml)



This file contains certificate information, old username/password, and new username/password. This file also contains a number of options for 2.0 and 2.5. Please read the notes carefully to determine which parameters are desired. For example, if the user would like to access the WEB UI in enterprise mode the WEBUI interface option should be entered under in the "set_enabled_interfaces" section.

The default.conf.xml file located in the same folder as the PSK Repository file (\\AMTConfiguration\\ConfigScripts\\).

17.5.4 Prepare Intel® AMT Systems

The Intel® AMT system must be prepared to be configured in Enterprise mode. The PID/PPS pairs entered in the PSK Repository file must match the Intel® AMT system. Both the PID and the PPS must match the Intel® AMT system or the Setup and Configuration process can not be completed.

Entering the PID/PPS can be done by manually through the MEBX, pre configured through AMTNVM, or with a USB key. For more information on USB key provisioning, please see the Setup and Configuration Users Guide

The Intel® AMT system must also contain the IP address of the Setup and Configuration system along with a port number. The default port number the Setup and Configuration application uses is 9971. This default port number can be modified (see section 17.2)

17.5.5 Install sample Certificates

If the Setup and Configuration application does not have the proper certificates, the application will prompt the user to install the sample certificates. This will allow the user to configure an Intel® AMT system without purchasing certificates. Answering yes to all the prompts will install the sample certificates in their default location

After the certificates are installed, the application will wait for incoming connections. If a port number has not been specified, the application will use port 9971. This must be specified on the Intel® AMT system.

After the AMT system is prepared to be configured, the Intel® AMT system will send out "hello" packets to the specified port and Server IP address. Once this "hello" packet is received, the Setup and Configuration application will acknowledge the request and finish the process.

After the Configuration process is complete, a message will be printed on the screen and the application will wait for the next hello packet. MEInfo can be used to verify that the process is complete.



18 Intel® AMT Privacy Icon

The Intel® AMT privacy icon is to let the end user know that Intel® AMT is enabled and running on the system. The icon will be displayed in the system icon tray. By default the status window will be launched every time Windows starts.

18.1 System Requirements

The Privacy Icon must be installed in windows (XP, Vista 32/64).

The Intel® AMT privacy icon is bundled with the LMS/SOL driver. Once the LMS driver is installed the Intel® AMT privacy icon will automatically be installed and launched.

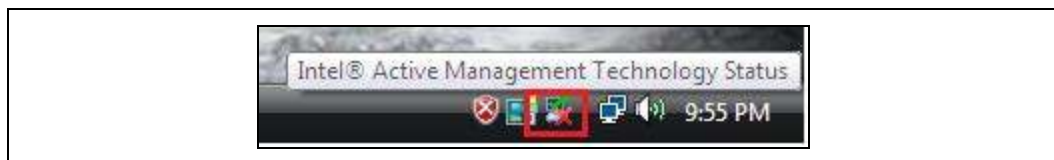
The Intel® Management engine Interface driver should be installed

18.2 Usage

The status window will appear when windows boots or when the status icon is double clicked. When Intel® AMT is enabled the red "x" will not be displayed. When the icon is double clicked the status window will appear.

When the status changes from disabled to enabled, a pop up message will be displayed.

Figure 23. Intel® AMT Privacy Icon

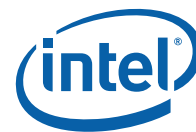


This status window will display the status of Intel® AMT. The status of Intel® AMT can be seen on the first line of the Status window. The word following "Intel® Active Management Technology (Intel® AMT) status of this computer is:" indicates the status of Intel® AMT.

There are three possible values:

Disabled: Setup and Configuration process has not started. ME is disabled or Intel® AMT is not selected as the Manageability mode.

Enabled: A status of enabled does not mean Intel® AMT is fully functional yet. When the Setup and Configuration process is in process, the status will be enabled. The status will also be enabled, when the Setup and Configuration process is complete.



Unknown: Intel® AMT privacy icon service has been disabled.

This status window will appear every time when windows boots up. If the “Do not show this message again” box is selected, the status window will not be displayed when Windows starts. However, the status window will still be displayed when the Intel® AMT icon is double clicked and when the status is changed from disabled to enabled.

Figure 24. Intel® AMT Status window



Once the “Do not show this message again” box is selected, the box will be replaced with a “Reactivate Notification” box. If the “Reactivate Notification” is selected, the Intel® AMT status window will be launched when windows starts and when the status changes from disabled to enabled.

The embedded link in the status window will launch a web browser window and take the user to <http://www.intel.com/vpro>.

§