



4 Flash Programming Tool (FPT)

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

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

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

4.1 System Requirements

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

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

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

One possible flow for using the FPT is:

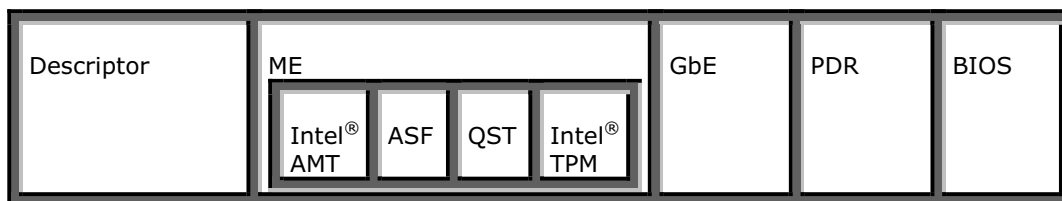
1. Pre-programmed flash with legacy or generic BIOS image is plugged into a new computer.
2. Computer boots.
3. The FPT is run and a custom BIOS/ME/GbE image is written to flash.
4. Computer powers down.
5. Computer powers up, boots, and is able to access its ME/GbE capabilities as well as any new custom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



4.2 Flash Image Details

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

Figure 35: Firmware Image Components



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

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

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

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

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

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

4.3 Windows* Required Files

The Windows version of the FPT executable is called fptw.exe. The following files must be in the same directory as fptw.exe:

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



- ssePmxdll32e.dll—supported library file.
- ssepmdrv.sys—supported system file.

4.4 DOS Required Files

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

- fpt.exe—the executable used to program the final image file into the flash.
- dos4gw.exe (DOS version only)
- fparts.txt—contains a comma separated list of attributes for supported flash devices. The text in the file explains each field. An additional entry may be required in this file to describe the flash part which is on the target system. Examine the target board before adding in the appropriate attribute values. The file is supplied already populated with default values for SPI devices used with Intel Reference Boards (CRBs).

4.5 Where to find dos4gw.exe

This file is only needed for the DOS version.

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

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

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

4.6 Programming the Flash Device

Once the ME has been programmed it will be running at all times. The ME is capable of writing to the flash device at any time, even when the management mode is set to none and it may appear that no writing would occur.

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



To disable the ME use one of the following options:

1. Disable the ME via the BIOS or the MEBx.
2. Pull down GPIO33 (manufacturing mode jumper) while powering on the system. If the parameters are configured to ignore this jumper, this will not be a valid method of disabling the ME.
3. Remove the memory from Channel 0—this method will cause the ME to boot up in an error state and the error will be written to the flash device. Programming the flash device should be done only after the OS has fully booted.
4. Set the ME disable bits in the strap sections of the descriptor region. Refer to the ICH EDS (sections 24.2.5.1 and 24.2.5.2) for more information.

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

4.7 Programming fixed offset variables

FPT can program the fixed offset variables. FPT will change the default values of the parameters that are programmed. The ME will not use the default values of the parameters until **globallocked** bit is set to 0x0. After the **globallocked** bit has been set the system must go into a G3 state, before the parameters are used by the ME. After this bit is set the parameters can **NOT** be modified. To modify the default settings for the parameters, the entire flash device needs to be reprogrammed.

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

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

Fpt.exe -ex -o <Text File> will create a text file that will allow the user to update multiple fixed offset variables. The variables will be displayed in the following format:

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

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

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

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

4.8 Usage

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

Both the Windows version and DOS version of the FPT can run with command line options. To view all of the supported commands, run the application with the /?



option. The commands in both the DOS and Windows versions have the same syntax. The command line syntax for fpt.exe and fptw.exe is:

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

/? or /h—displays the help screen.

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

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

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

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



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

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

/d—dumps the flash contents to a file or to the screen using the STDOUT option. The flash device must be written in 4KB sections. The total size of the flash device must also be in increments of 4 KB.

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

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

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

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

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

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

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

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

/q—do not display output to the screen.

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

/erase – Erase the contents of the flash

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

/log— specifies the name of the log file created



/list—list all the SPI devices supported

/Intel® TPM <Enabled/Disable>—enable/disable integrated TPM

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

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

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

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

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

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

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

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

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

/dumplock—displays the current descriptor lock settings

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

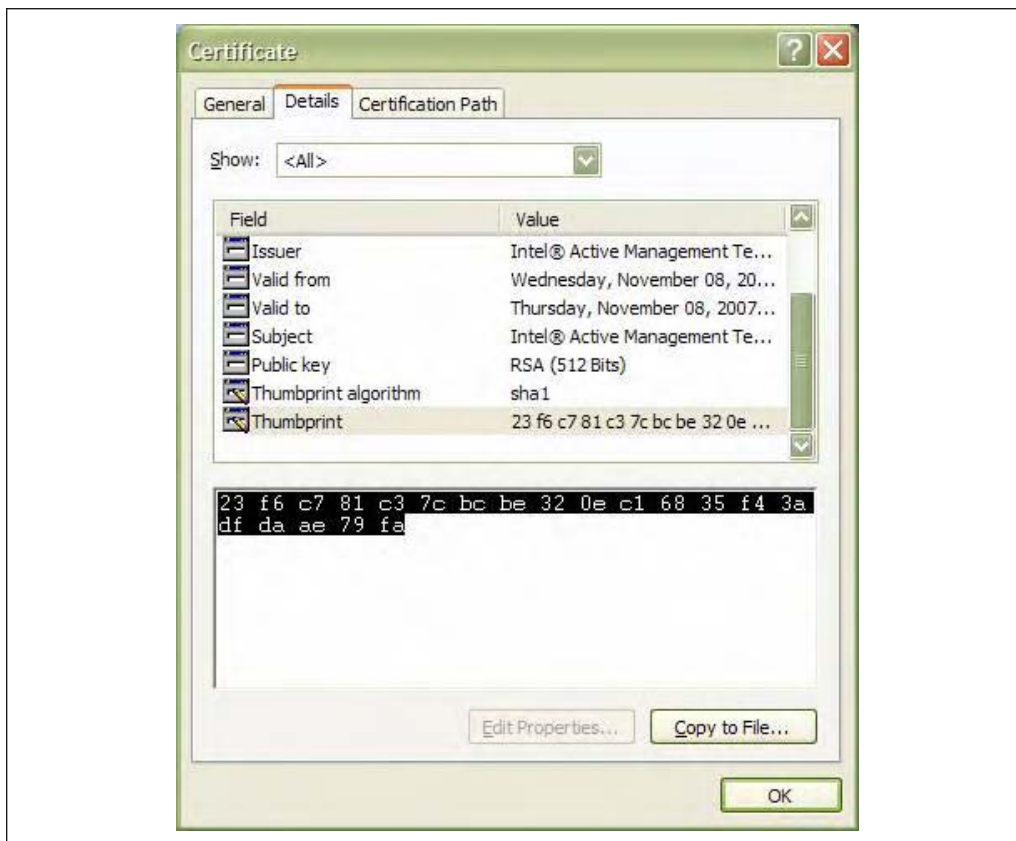
/closeMnf—Option used at the end of the manufacturing line. Please see Section 5.9 End of Manufacture for more details

4.9 Update Hash Certificate through FOV

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

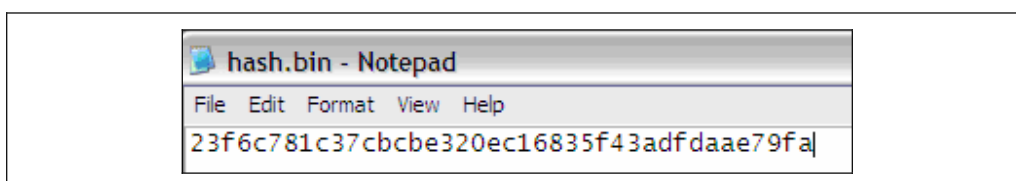
- 1) User must copy the raw hash values from a valid certificate file

Figure 36: Raw Hash value from certificate file



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

Figure 37: Sample Hash BIN file



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

```
fpt -u -in sampleparam.txt
```

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

[ZTCEnable]



Enabled = 0x0
Value = 0x00

[Hash1]
Enabled = 0x1
IsActive = 0x1
FriendlyName = myHash3
RawHashFile = hash.bin

[CfgSrvFQDN]
Enabled = 0x0
Value = Intel.com

4.10 fparts.txt File

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

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

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

If the device is not located in the fparts.txt file, the user is expected to provide information about their device and insert the values into the file using the same format as the rest of the devices. The device must have a 4KB erase sector and the total size of the SPI Flash device must be a multiple of 4KB. The values are listed in columns in the following order:

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



- Chip Erase Command.

4.11 End of Manufacture

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

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

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

To end manufacture, perform the following actions:

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

When used with the **-closemnf** flag, the FPT provides a single command that performs all of these operations, and at the same time enables the NV Lock, thereby restricting access to the NV storage area.

4.12 Examples

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

4.12.1 Example 1 – Dump ME Region to the screen

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

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

4.12.2 Example 2 –Program Specific Address

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



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

4.12.3 Example 3 –Program Specific Region

```
C:\ fpt.exe /f bios.rom /bios
-----
Flash Programming Tool. Version X.X.X

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

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

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

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

4.12.4 Example 4 – Dump Specific Region

```
C:\ fptw.exe /desc /d descdump.bin
-----
Flash Programming Tool. Version X.X.X

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

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

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

Using software sequencing.

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

Memory Dump Complete
```



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

4.12.5 Example 5 – Display SPI information

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

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

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

Using software sequencing.

    --- Flash Image Information --
    Signature: VALID
    Number of Flash Components: 1
    Component 1 - 2048KB (16384Kb)
    Regions:
        Descriptor - Base: 0x000000, Limit: 0x000FFF
        BIOS       - Base: 0x100000, Limit: 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
```

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

4.12.6 Example 6 – Verify Image with errors

```
C: \ fpt.exe /verify outimage.bin
Flash Programming Tool. Version X.X.X
Reading LPC BC register... 0x00000001
Reading LPC RCBA register... 0xFED1C001
SPI register base address... 0xFED1F020
Loading the flash definition file
Reading file "fparts.txt" into memory...
Initializing SPI utilities
```



```

Reading HSFSTS register... Flash Descriptor: Valid
--- Flash Devices Found ---
SST25VF016B      ID:0xBF2541      Size: 2048KB
(16384Kb)
SST25VF016B      ID:0xBF2541      Size: 2048KB
(16384Kb)
Using software sequencing.
Reading file "outimage.bin" into memory...

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

```

This usage compares the ME Region programmed on the flash with the specified firmware image file outimage.bin. If the /y option is not used, the user will be notified that the file is smaller than the binary image. This is due to extra padding that is added during the program process. The padding can be ignored when performing a comparison. The /y option will proceed with the comparison without warning.

4.12.7 Example 7 –Verify Image successfully

```

C: \ fpt.exe /verify outimage.bin

Flash Programming Tool. Version 0.8.11

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

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

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

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

```

This usage compares the file image.bin with the contents of the flash. Comparing an image should be done immediately after programming the flash device. Verifying the contents of the flash device after a system reset will result in a mismatch.

4.12.8 Example 8 – Program FOV parameter

```

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

Flash Programming Tool. Version 0.8.11

```



Flash Programming Tool (FPT)

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

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

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

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

§