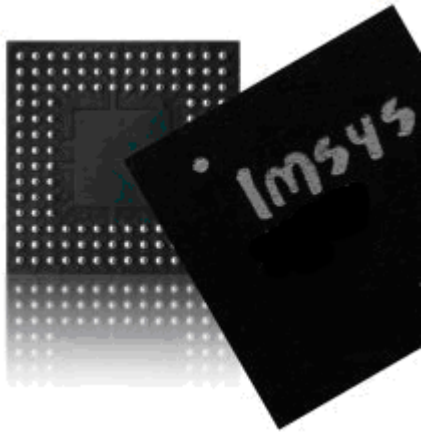
Firmware Installation and Upgrade

IM3000 Based Systems

Revision 1.1



Imsys

Imsys AB, Sweden
Johanneslundsvägen 3
SE-194 61 Upplands Väsby
Sweden
+46 8 594 110 70 phone
+46 8 591 110 89 fax

Imsys, USA
9903 North Poetry Ln
Terrell, TX 75160
USA
+1 877 775 1627

838 Henderson Ave
Sunnyvale, CA 94086
USA

VAT:
556453-3247-01

Web:
www.imsystech.com

Copyright

Disclaimer

Imsys AB makes no warranties for the contents of this document or the accuracy or completeness thereof and disclaims any implied warranties of merchantability or fitness for any particular purpose. Further, Imsys AB reserves the right to revise this document and to make changes without notifications to any person of such changes. The information contained in this document is provided solely for use in connection with Imsys AB products. This document should not be construed as transferring or granting a license to any intellectual property rights, neither expressed nor implied.

Imsys AB products have not been designed, tested, or manufactured for use in any application where failure, malfunction, or inaccuracy carries a risk of death, bodily injury, or damage to tangible property, including, but not limited to, use in factory control systems, medical devices or facilities, nuclear facilities, aircraft, watercraft or automobile navigation or communication, emergency systems, or other applications with a similar degree of potential hazard.

Revision History

Document: STO-DEV7237-LZ	
Previous revision: -	
Section	**Changes since last revision**
1.0	First Release
1.1	Updated table in section 2.2

Table of Contents

1. Firmware

1.1. Firmware overview

The firmware for IM3000-based systems consists of a few modules. They all are essential for the system to boot up and operate. These modules are:

- Microprogram Boot Loader – assembled as one microprogram file (.mp) intended for a specific flash format and size. This file is stored in the first sector (boot sector) of the flash memory and fetched by the processor into the processor's internal RAM on power up or reset. The Microprogram Boot Loader starts the boot process by loading other firmware components.
- Main Microprogram – assembled as one microprogram file (.mp) specific to each target platform. The microprogram file is stored in a special area of the flash memory and fetched by the Microprogram Boot Loader into the processor's internal RAM. The Main Microprogram provides the complete hardware- and processor-level operating environment and base functionality.
- Boot Loader Program – assembled as one executable file (.gpx). This file is stored in a special area of the flash memory and loaded by the Microprogram Boot Loader into the DRAM. The Boot Loader Program loads the main executable file from the flash file system and transfers the execution control to it.
- Main Executable – assembled as one executable file (.gpx). This file is stored on the flash memory as an ordinary file within the file system. This is the main code that the target system executes while operating.

The Microprogram Boot Loader presents only once in the system and it is always located in the boot sector of the flash memory. It finds which Main Microprogram file and the Boot Loader Program file to load by checking corresponding fields in a boot blocks. There are two of them: Read-only Boot Block (RBB) and Writable Boot Block (WBB). The RBB is mandatory and it must have valid pointers to the Main Microprogram file and to the Boot Loader Program file. The WBB is optional and the pointer to it is stored in the RBB. The WBB can have pointers to another Main Microprogram and Boot Loader Program files, which can differ from those pointed by the RBB.

The Microprogram Boot Loader first reads the RBB and checks the pointer to WBB. If it is zero, i.e. no WBB present, it loads Main Microprogram and Boot Loader Program files specified in the RBB. Otherwise it reads the WBB and checks its pointers to the Main Microprogram and to the Boot Loader Program. If any of them are non-zero, it loads the corresponding module. Otherwise the module pointed by RBB is loaded.

1.2. Boot fault protection

The Microprogram Boot Loader with the help of two boot blocks and two instances of Main Microprogram and the Boot Loader Program provides the basic protection against boot faults.

The Microprogram Boot Loader starts from trying to clear the WBB pointer in the RBB. If the write-protection of the flash is enabled the WBB pointer will not

be cleared and the boot loader continues to execute its normal procedure of checking WBB. Otherwise it will continue with the RBB. So, it is always possible to force the boot process to take place from the RBB by disabling the flash write-protection.

Another level of fault protection is an integrity verification. When the Microprogram Boot Loader tries to load the firmware from WBB pointers it first verifies the integrity of each module. If it finds the module corrupted it clears the corresponding pointer in the WBB, so on the next boot it will take that module from the RBB.

1.3. Types of flashing

The Read-only Boot Block and corresponding “read-only” versions of Main Microprogram and Boot Loader Program files can only be written during so-called “hard-flashing” procedure. This procedure requires the target system to be connected though the debug connector to the PC with the Imsys Developer installed. The write-protection of a read-only part of flash memory has to be disabled. The hard-flashing procedure is used for initial firmware installation or in the case when the damaged system needs to be restored.

The Writable Boot Block and its “writable” versions of Main Microprogram and Boot Loader Program is written via a “soft-flashing” procedure. The Main Executable file along with other files and folders that forms the content of the target file system is also written the same way. The soft-flashing means the installation of a firmware by the software, which is already running on the target system. The soft-flashing procedure can be performed on the live system through the network or terminal connection.

1.4. SNP packages

The firmware installed through the soft-flashing procedure comes in a form of firmware distribution packages (SNP packages) – specially formatted archive files which may or may not contain the following firmware components:

- Main Microprogram file
- Boot Loader Program file
- Flash file system content: Main Executable, configuration files, any other files and directories that needs to placed on the target system.

Any number of custom SNP packages can be created for different systems or for different purposes.

As soon as the soft-flashing procedure modifies only the WBB, the Main Microprogram and the Boot Loader Program files contained in the SNP package are installed as “writable” versions. The RBB and “read-only” versions stays untouched. So the soft-flashing procedure provides the way to upgrade the firmware even on the live system without a need to be “hardly flashed” and to keep the system bootable with “read-only” version of the firmware.

2. FlashIt project

2.1. FlashIt project overview

The FlashIt project is an automated firmware installation tool for IM3000-based systems. It can perform both hard- and soft-flashing and thus allows to completely initialize the target system with either original or customized firmware as well as to create customized firmware distribution packages. The FlashIt project comes with each IM3xxx target profile and can be found under the <Profile_Path>\Tools\FlashIt directory.

2.2. FlashIt utility

The FlashIt project includes the FlashIt utility, a IM3000 executable module (FlashIt.gpx), which is downloaded and executed on the target system. It uses the RAM disk with predefined directory structure to maintain files necessary for firmware installation. When FlashIt executes it reads files from the RAM disk and writes firmware files onto the flash memory.

Here is the description of RAM disk directory structure:

- /HIS – a place for a text file containing the Hardware Identification String to be stored in the RBB. This string can be up to 32 characters. By default this directory contains the reference to the HIS.TXT file which is located in <Profile_Path>\Tools\FlashIt
- /NETINFO – a place for a text file with the MAC address to be stored in the RBB. For example: 00:0b:b9:01:02:03. By default this directory contains the reference to the MAC.TXT file which is located in <Profile_Path>\Tools\FlashIt.
- /RBB_MPGM_BOOTLOADER – a place for the Microprogram Boot Loader file. By default this directory contains the reference to the Boot_sf_64_m210.mp file which is located in <Profile_Path>\Bin.
- /RBB_MPGM – a place for the read-only Main Microprogram file. By default this directory contains the reference to the <Target_Name>.mp file which is located in <Profile_Path>\Bin.
- /RBB_BOOTLOADER – a place for read-only Boot Loader Program file. By default this directory contains the reference to the bootloader.gpx file which is located in <Profile_Path>\Bin.
- /SNP – a place for SNP packages to be installed (soft-flashing stage). By default this directory contains the reference to the system.snp file which is located in <Profile_Path>\Bin.

Each directory except /SNP must contain only one file. The name of file is insignificant. The /SNP directory may have multiple files, but the installation order is undefined. If the directory is empty the corresponding information or firmware part will not be written.

The execution of the FlashIt utility can be altered by command-line options. The following options are currently supported:

Option	Meaning
`-i`	Prints the current firmware information. Nothing will be written to the flash memory.
`-w`	Enables the write-protection on the flash memory after the flashing procedure is complete.
`-z`	Reset the boot sector. This option must be specified when the first time initialization is performed.
`-r`	Write read-only files. Implied with -z. Used to change RBB programs.
`-n`	Write NETINFO. Implied with -z. Used to change MAC address.
`-h`	Write HIS. Implied with -z. Used to change Hardware Identification string.
`-f`	Format the flash memory. This option must be specified when the file system is initialized for the first time.
`-d`	Output messages to the Imsys Developer's Output window.
`-c1`	Output messages to the COM1 port
`-c2`	Output messages to the COM2 port
`-c3`	Output messages to the COM3 port

2.3. How to perform flashing

To perform flashing of the target system the following steps should be done in Imsys Developer:

- Open FlashIt project.
- Open Project View window, if not already opened (Menu\View\Project View).
- Modify MAC.TXT and HIS.TXT files if necessary.
- Open RAM Disk window, if not already opened (Menu\Project\RAM Disk).
- Review the RAM disk content and modify it if necessary.
- Check the actual size of the RAM disk (Context Menu\Properties).
- By default the size allocated for the RAM disk is 2000-8500 Kbytes. Increase it if necessary (Menu\Project\Settings -> Debugger\RAM Disk Size).
- Review command-line options and change them if necessary (Menu\Project\Settings -> Debugger\Program Arguments).
- Boot the system (Menu\Debug\Boot or F6).

The flashing process will start automatically.

You could see the following output in the terminal or in the debug output window:

2.4. Archive utility

The FlashIt project includes Archive utility which helps to create any number of customized SNP firmware distribution packages.

The original SNP package, system.snp that comes with IM3xxx target profile can be found in <Profile_Path>\Bin directory.

An Archive utility allows to specify the following components to be included in an SNP package:

- Microprogram – one writable Main Microprogram file. By default it is empty.
- Boot Loader Program – one writable Boot Loader program file. By default it is empty.
- File system - any files and directories that needs to placed on the target system. By default it includes the Main Executable file system.gpx and system configuration files which are located in <Profile_Path>\Bin directory.

With its default settings the Archive utility will build the same SNP package as the original system.snp and place it to the <Profile_Path>\Tools\FlashIt directory.

If several custom SNP packages are supposed to be installed on one system at a time and the installation order is undefined (as in the case with FlashIt) each firmware component should be placed into exactly one SNP package. Otherwise, the file will be overwritten by that from the last installed package.

2.5. How to build SNP package

To build the SNP firmware distribution package the following steps should be done in Imsys Developer:

- Open FlashIt project.
- Open Archive Settings (Menu\Project\Settings -> Archive).
- Specify the path and the file name for created SNP package (Output File).
- Specify the Main Microprogram file if necessary (Microprogram).
- Specify the Boot Loader Program file if necessary (Boot Loader Program).
- Specify whether to include the file system content or not (File System).
- Open Archive window, if not already opened (Menu\Project\Archive).
- Review the file system content and modify it if necessary.
- Build the SNP package (Menu\Build\Build Archive or Ctrl+Alt+F6).

The SNP package will be created and placed into the specified directory.

3. Firmware Upgrade

The firmware can be upgraded or downgraded either with FlashIt project through the flashing procedure described above or via the “live update”.

3.1. Live update

The system software for IM3000-based systems provides the facility for automatically updating the firmware over the network. The firmware is distributed as SNP packages and installed via the soft-flashing procedure.

By the user or application request the live update component of the system software fetches specified SNP packages from a TFTP server and installs them.

The live update facility is exposed to the end-user in two flavors: as a shell command and as a Java method (on Java-enabled systems).

3.2. UPDATE shell command

The command syntax is:

```
root> UPDATE [options] host file
```

where the **host** is a name or an IP-address of a TFTP server and the **file** is a full name of the SNP package on that server.

For example:

```
root> UPDATE tftp.imsys.se /im3000/ver11/system.snp
```

This command also provides the following options:

Option	Meaning
`-t timeout`	Specifies a timeout interval in milliseconds for receiving data from the server. If no data is received within this interval the command will fail. The default value is 2000 ms.
`-f file1`	Instructs to perform a flash memory format before an update. After formatting the command will first install an SNP file specified by **file** and then an SNP file specified by **file1**.
`-e file1`	Instructs to perform a flash memory format, if the installation of **file** fails. After formatting the command will first install an SNP file specified by **file** and then an SNP file specified by **file1**.

For example:

```
root> UPDATE –t 1000 –f /im3000/ver11/microprogram.snp \
tftp.imsys.se /im3000/ver11/system.snp
```

After a successful execution of this command the system has new firmware and needs to be rebooted. This can also be done from the shell with the **REBOOT** command.

3.3. Java updateTFTP() method

The same functionality as the **UPDATE** command is provided by a Java **updateTFTP()** method of the **SNAP** class of the **se.imsys.system** package.

It has the following syntax:

```
public static int updateTFTP(String host, String file)
```

This method will fetch an SNP package specified by **file** from the TFTP server specified by **host** and update the system firmware with files containing in this package.

After the update process is completed the system can be rebooted by the program via **se.imsys.system.Ish.reboot()** method.