

FR FAMILY
SOFTWARE TOOL
FME FR FLASHPROGRAMMER

USER GUIDE

Revision History

Date	Issue
2008-07-10	v1.0 Markus Heigl Initial Version
2008-10-20	Markus Heigl v 1.1 Added Feature Description: passing several mhx Files
2008-12-17	Markus Heigl V1.2 Added the MB91F469Q and MB91FV460B in the documentation

This document contains 19 pages.

Warranty and Disclaimer

To the maximum extent permitted by applicable law, Fujitsu Microelectronics Europe GmbH restricts its warranties and its liability for **all products delivered free of charge** (e.g. software include or header files, application examples, target boards, evaluation boards, engineering samples of IC's etc.), its performance and any consequential damages, on the use of the Product in accordance with (i) the terms of the License Agreement and the Sale and Purchase Agreement under which agreements the Product has been delivered, (ii) the technical descriptions and (iii) all accompanying written materials. In addition, to the maximum extent permitted by applicable law, Fujitsu Microelectronics Europe GmbH disclaims all warranties and liabilities for the performance of the Product and any consequential damages in cases of unauthorised decompiling and/or reverse engineering and/or disassembling. **Note, all these products are intended and must only be used in an evaluation laboratory environment.**

1. Fujitsu Microelectronics Europe GmbH warrants that the Product will perform substantially in accordance with the accompanying written materials for a period of 90 days from the date of receipt by the customer. Concerning the hardware components of the Product, Fujitsu Microelectronics Europe GmbH warrants that the Product will be free from defects in material and workmanship under use and service as specified in the accompanying written materials for a duration of 1 year from the date of receipt by the customer.
2. Should a Product turn out to be defect, Fujitsu Microelectronics Europe GmbH's entire liability and the customers exclusive remedy shall be, at Fujitsu Microelectronics Europe GmbH's sole discretion, either return of the purchase price and the license fee, or replacement of the Product or parts thereof, if the Product is returned to Fujitsu Microelectronics Europe GmbH in original packing and without further defects resulting from the customer's use or the transport. However, this warranty is excluded if the defect has resulted from an accident not attributable to Fujitsu Microelectronics Europe GmbH, or abuse or misapplication attributable to the customer or any other third party not relating to Fujitsu Microelectronics Europe GmbH.
3. To the maximum extent permitted by applicable law Fujitsu Microelectronics Europe GmbH disclaims all other warranties, whether expressed or implied, in particular, but not limited to, warranties of merchantability and fitness for a particular purpose for which the Product is not designated.
4. To the maximum extent permitted by applicable law, Fujitsu Microelectronics Europe GmbH's and its suppliers' liability is restricted to intention and gross negligence.

NO LIABILITY FOR CONSEQUENTIAL DAMAGES

To the maximum extent permitted by applicable law, in no event shall Fujitsu Microelectronics Europe GmbH and its suppliers be liable for any damages whatsoever (including but without limitation, consequential and/or indirect damages for personal injury, assets of substantial value, loss of profits, interruption of business operation, loss of information, or any other monetary or pecuniary loss) arising from the use of the Product.

Should one of the above stipulations be or become invalid and/or unenforceable, the remaining stipulations shall stay in full effect

Contents

REVISION HISTORY	2
WARRANTY AND DISCLAIMER	3
CONTENTS	4
1 INTRODUCTION	5
1.1 Supported Devices	5
2 SOFTWARE INSTALLATION	6
2.1 System Requirements	6
2.2 Installation	6
3 USING THE SOFTWARE	7
3.1 The Graphical User Interface	7
3.2 Flash Programming	8
3.2.1 Basic settings	8
3.2.2 Automatic reset and other signals	8
3.2.3 Automatic Mode	9
3.2.4 Manual programming	10
3.2.4.1 How to start in manual mode?	10
3.2.4.2 BootROM Functions	10
3.2.4.3 Flash Functions	12
3.3 External Flash programming	14
3.4 “Options” tab	15
3.5 Command line options	16
4 APPENDIX	19
4.1 Information on Files to be programmed (MHX-Format)	19
4.2 Using “K-line”	19

1 Introduction

Please note that Fujitsu does not allow the use of this software for mass production!

The FME FR Flashprogrammer is a development tool which can be used to program Fujitsu MB91xxx series microcontrollers over a standard UART connection.

The Flashprogrammer uses Motorola™ hex format (.mhx) files as input. These files can be generated by the Softune toolchain automatically.

1.1 Supported Devices

- ADA_91V460_91F467D
- EMAMB91V460A
- EMAMB91FV460B **NEW**
- Cremson Starterkit (Extflash16MB, Extflash1MB, Extflash8MB)
- MB91F361
- MB91F362
- MB91F364G
- MB91F365G
- MB91F365
- MB91F376G
- MB91F463N
- MB91F464A
- MB91F464H
- MB91F465B
- MB91F465C
- MB91F465K
- MB91F465P
- MB91F465X
- MB91F467B
- MB91F467C
- MB91F467D
- MB91F467M
- MB91F467R
- MB91F467S
- MB91F467T
- MB91F469G
- MB91F469Q **NEW**

2 Software Installation

2.1 System Requirements

486 or later processor

1 Free Serial com port, (USB optional)

1.2 MB of free disk space

Windows 95/98/2000/XP

2.2 Installation

In order to install the FME FR Flashprogrammer you have to run the FME FR Flashprogrammer Setup. You can download the Setup and this Manual from the following Website.

http://mcu.emea.fujitsu.com/mcu_tool/detail/FLASH_PROGRAMMER_FR_FME.htm



After the setup was started it will guide you through the rest of the installation process

3 Using the Software

3.1 The Graphical User Interface

After starting the flashprg.exe you will get a window similar to the one shown in Figure 1.

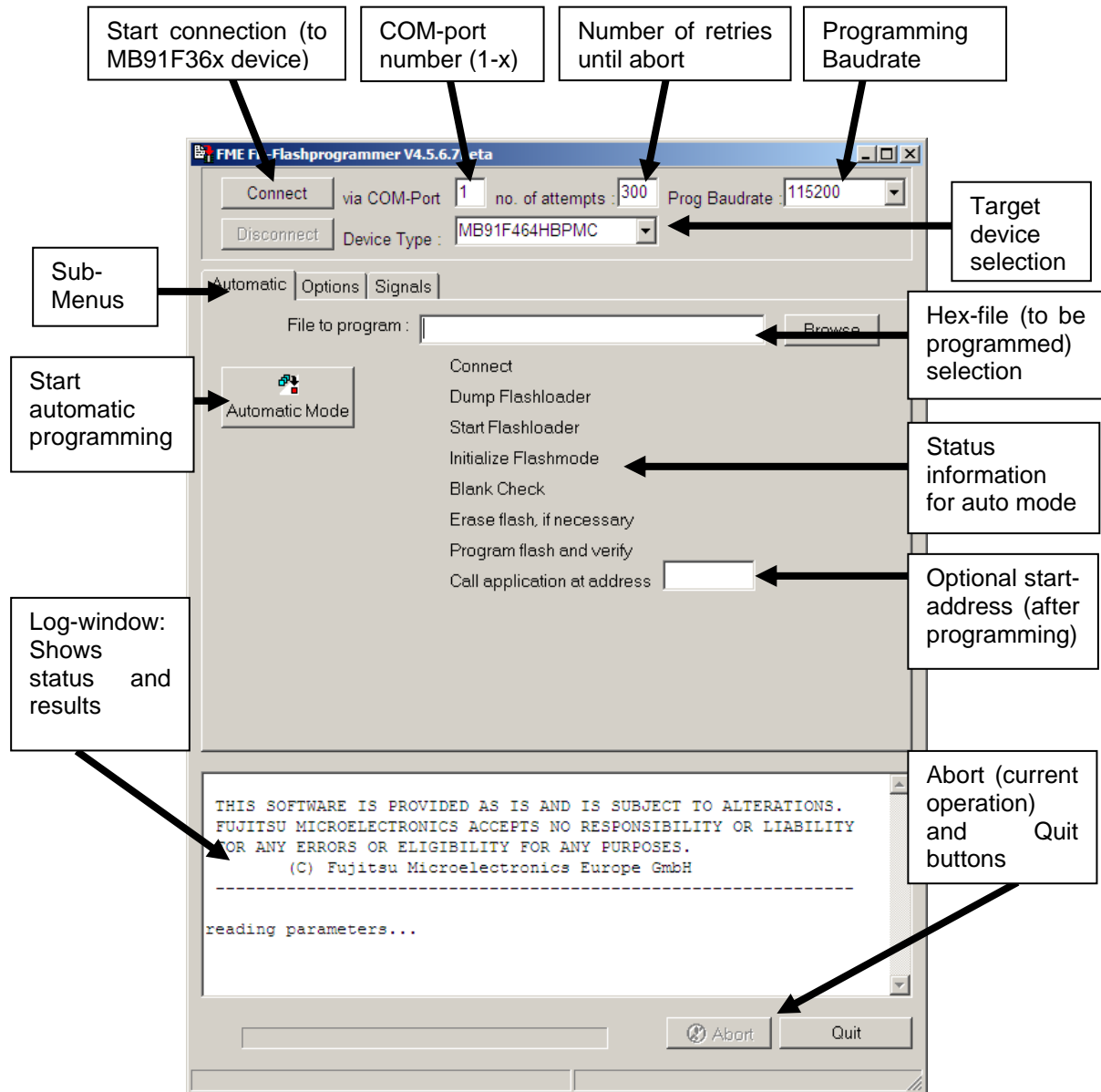


Figure 1 Main menu

3.2 Flash Programming

3.2.1 Basic settings

Before programming a microcontroller you will have to configure some settings. The settings are saved when you quit the program therefore you will have to do this steps only the first time you use the program unless you are changing something in your development setup.

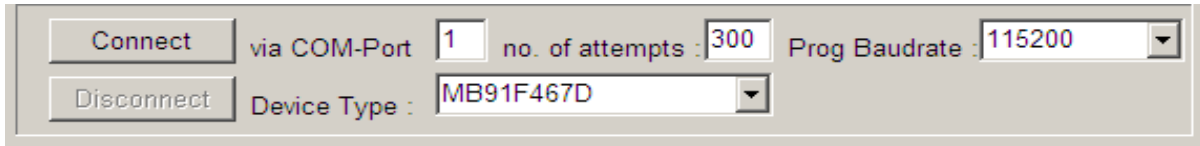


Figure 2 Connection settings

First of all you will have to choose the serial port (COM port) you are using to connect the MCU to your PC and the baud rate you want to use for programming. Then you should choose your Device Type from the dropdown box. If your Hardware supports reset generation via the DTR or RTS line you can configure the FME FR Flashprogrammer to trigger such a reset before programming. If not you will have to trigger the reset manually and you can skip the next paragraph

3.2.2 Automatic reset and other signals

Depending on your hardware the FME FR Flashprogrammer can use the RTS and DTR line of the serial port to trigger certain actions. For example you can use either the RTS or DTR signal to generate a reset.

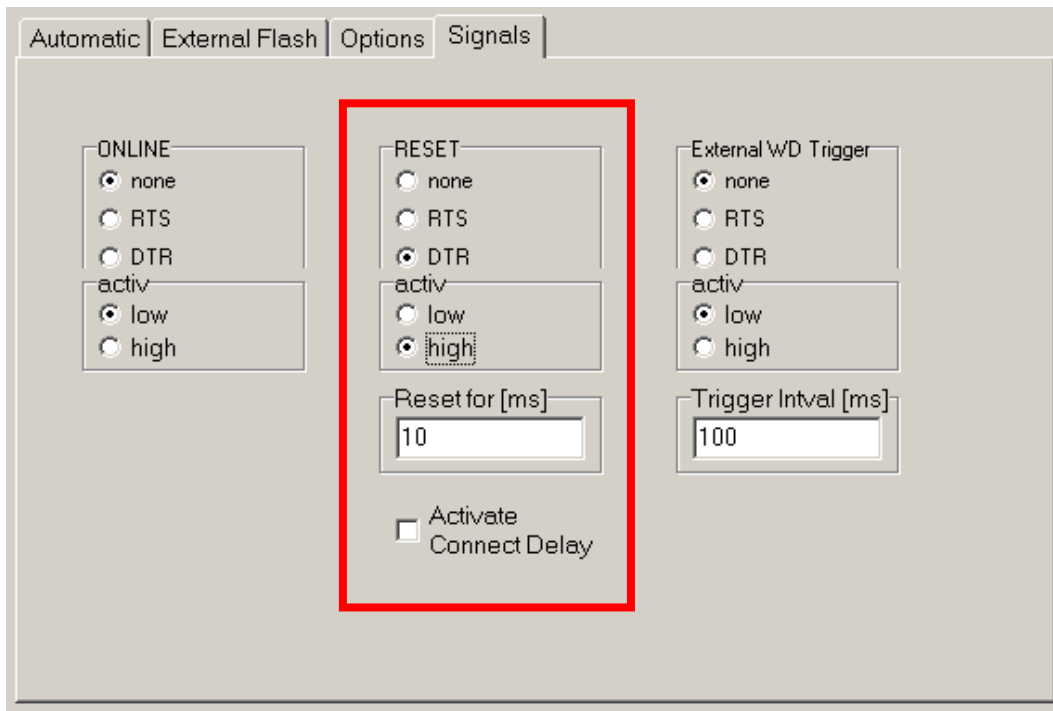


Figure 3 Signal Tab

On most Fujitsu starterkits reset can be generated via DTR. To use this feature you have to set Reset to DTR and active high as highlighted in Figure 3. You can change the reset time to fit to your needs or leave it at its default value which is sufficient for Fujitsu starterkits. Of course you will have to set the jumpers on the starterkit accordingly. Additionally you can check the “Activate connect Delay“ checkbox which will delay the first connection attempt.

The FME FR Flashprogrammer also can be used to generate a periodic signal on the DTR or RTS line which can be used to trigger an external watchdog component while the MCU is in programming mode.

Additionally you can set RTS or DTR to either low or high signal when a connection attempt was successful. This can be configured by the radio buttons on the left and may be useful if you want to trigger additional actions on your board while programming.

3.2.3 Automatic Mode

The fastest and easiest way to program a microcontroller is to use the automatic mode. This mode will run through the programming procedure without much user interaction.

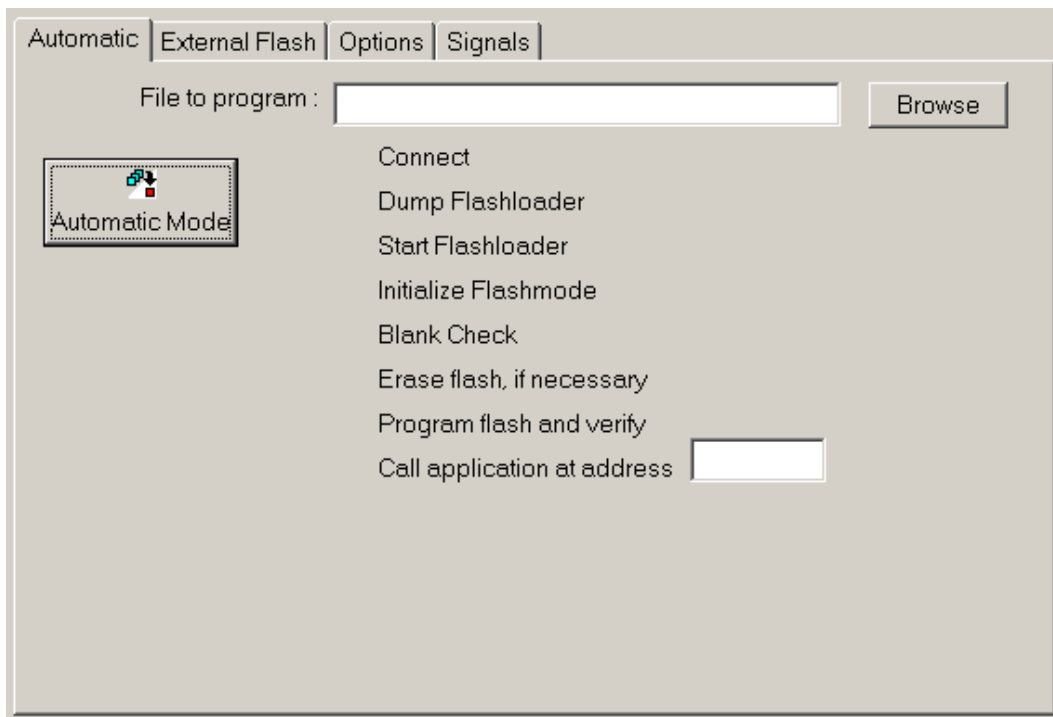


Figure 4 Automatic Mode

By clicking on the browse button you will be prompted with a standard windows open dialog where you can choose the mhx file to be programmed. It is also possible to directly enter the full path and filename into the textbox to the left of the browse button.

If you want to program more than one mhx file at once you can do this as well by entering the full path and filenames of all files that you want to program separated by comma.

After choosing the file you can press the automatic mode button which will start the programming procedure. When you have configured the FME FR Flashprogrammer to generate a reset you just have to wait until the Flashprogrammer states that automatic mode has finished. Otherwise you will have to reset your target system manually before the maximum numbers of connection attempts have been made. After that the Flashprogrammer will proceed with programming as usual and will inform you when programming has finished.

3.2.4 Manual programming

All the steps that are done during automatic mode can also be triggered manually. This involves doing the right things in the right order. Although this procedure is more complex and not practical if you just want to flash a program to the microcontroller it gives you more control over the process. This can be useful if you want to test a special setup which will not work with automatic mode or if you have to debug your development setup.

3.2.4.1 How to start in manual mode?

The basic settings mentioned in 3.2 and 3.2.2 are applicable for manual mode as well. Manual mode is started by clicking on the Connect button. The Log Window should now show something similar to Figure 5.

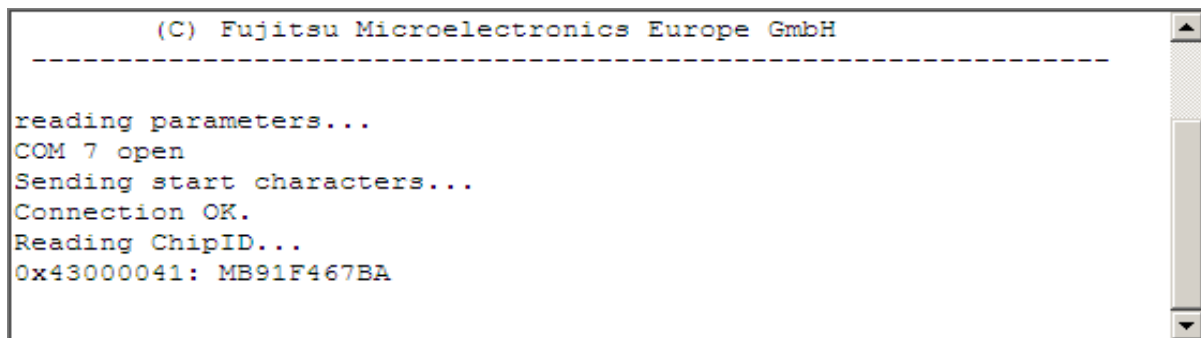


Figure 5 Log window after Connect

The log window states that a connection could be established and it prints out the ChipID read from the microcontroller. At this point we have a connection to the BootROM which allows us to write data via the serial Interface to the microcontroller. We will use this interface to write some settings to the microcontroller and to load a kernel to the microcontroller which enables us to write to the flash memory. To do so we will switch to the tab labelled BootROM_Functions.

3.2.4.2 BootROM Functions

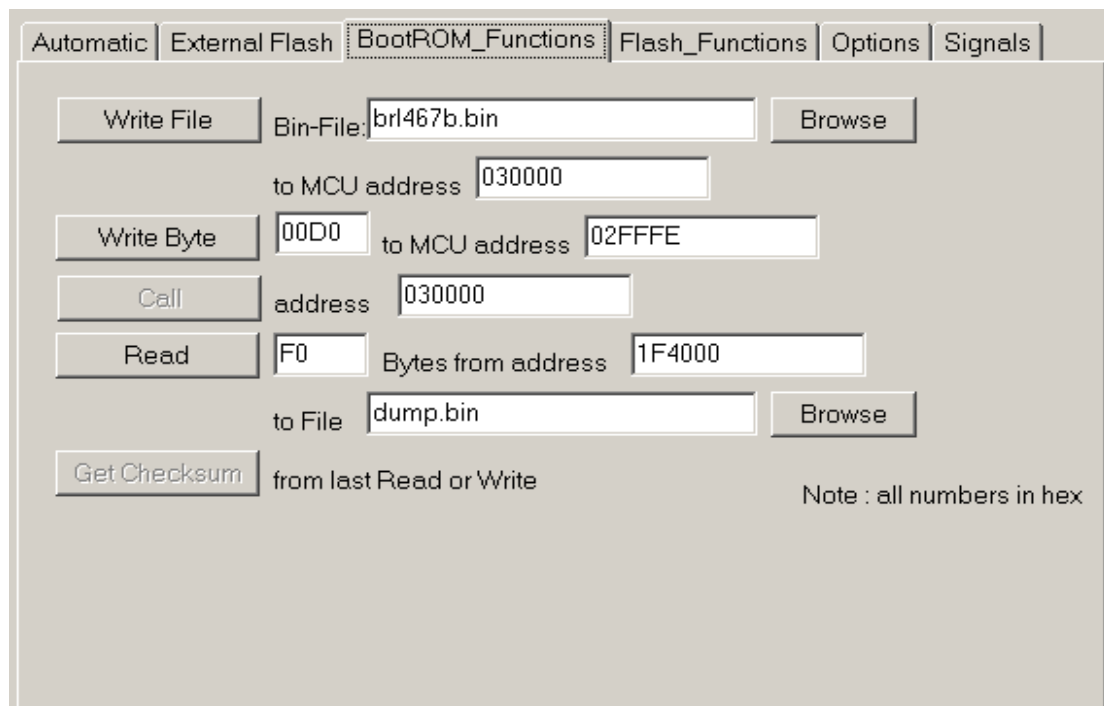


Figure 6 BootROM Functions tab

Button	Description
Write File	Dumps a binary file into the internal RAM beginning at the specified address. The transmission will be verified by a checksum.
Write Byte	Writes one single byte to the given location (RAM)
Call	Calls the program or function at the given address. If there is a "RET"-statement at the end, the operation will continue (return value is displayed)
Read	Reads out a number of bytes from a given address and creates a binary file.
Checksum	Displays the checksum from the last operation

You will now have to load the kernel to the microcontroller. The kernel is located in a binary file. These files are normally stored together with the flashprg.exe. The Flashprogrammer has already chosen the default kernel according to the device you have set in the device drop down box. If it is necessary you can exchange this kernel with another kernel. In the to MCU address field you have to enter the memory address the binary file was linked for. This value is also preset to the value for the default flashloader. Pressing the Write File button will load the kernel into the memory of the microcontroller. After that you will have to click on the Write Byte button which will store some configuration data to the microcontroller memory which are evaluated by the kernel to for example set the communication speed. Pressing the call button will trigger the MCU to executing the kernel. Now you can switch to Flash Functions tab.

3.2.4.3 Flash Functions

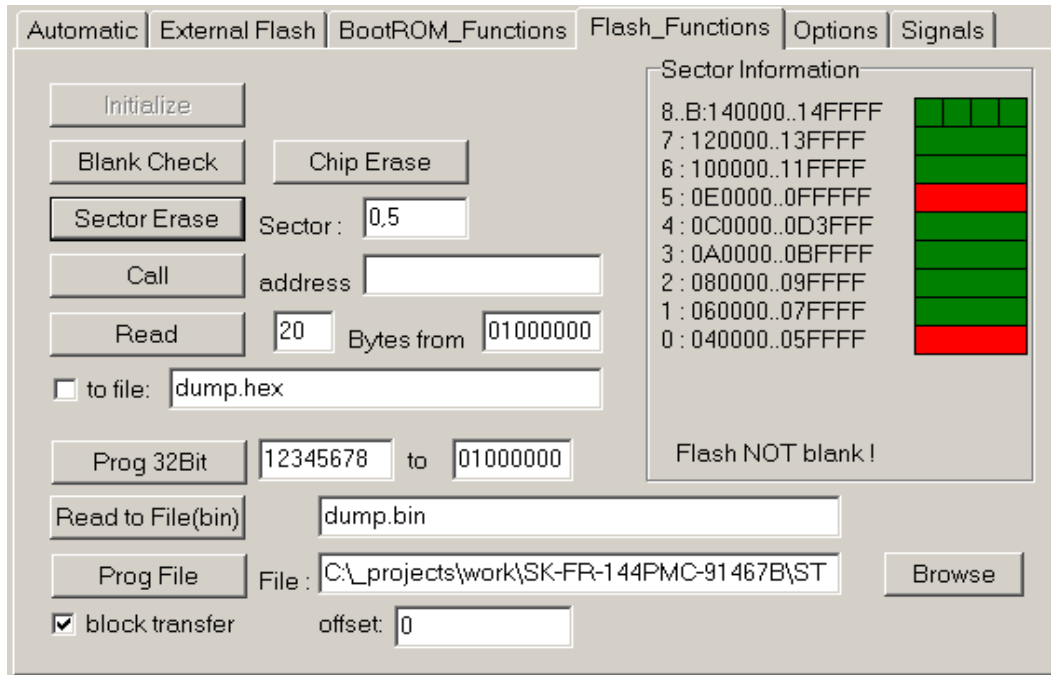


Figure 7 Flash Functions tab

Button	Description
Initialize	Checks whether the flashloader which was dumped to RAM is available, initialized at the correct baud rate and ready to operate. This function checks for a prompt character (“>”) and displays the flashloader version. If this operation fails (e.g. no flashloader present), none of the following functions will be enabled.
Blank Check	Reads out every byte of each flash-sector to check for non-blank (≠ “FF”) cells. The result is displayed in the flash-diagram shown (red = not blank sectors; green=blank sectors). This operation is performed automatically after each programming/erasing operation. All not blank sectors are also listed in the Sector Erase-field.
Sector Erase	Erases the sectors indicated by the numbers in the Sector Erase field (separated by “,” e.g.: “1,2,3” will erase sectors 1,2,3). The result is displayed in the log-window.
Chip Erase	Erases the entire flash-ROM (all sectors). The result is displayed in the log-window.
Call :	This function will divert operation of the device to the specified location. This terminates the flashloader.

Button	Description
Read :	<p>Reads out a number of bytes beginning at the specified address and displays the results in hexadecimal format on the log-window. This function can be used to verify certain areas (e.g. security-vector).</p> <p>It is also possible to save the data to a file by checking the “to file”checkbox and providing a filename in the textbox next to it.</p> <p>If the external memory feature is selected, reading of external memory addresses is possible too.</p>
Prog 32 bit	<p>Programs one word to flash at the specified address. The result is displayed in the log-window.</p> <p>If the external memory feature is selected, programming of external memory addresses is possible too.</p>
ProgFile	<p>Handles the transfer of the specified MHX-file to the flashloader. Each line will be processed (programmed) sequentially. If an error occurs, programming will be aborted with an error-message (e.g. “Loading Error”). During programming, a progress-bar is displayed.</p>
Block Transfer:	<p>This option enables the block transfer of the MHX-file (instead of transmitting only single lines they are compressed to bigger chunks). This option will increase the programming speed and also enables the option to uses an offset for the target address.</p>
Read to File	<p>This function will read and save the complete flash memory to a binary file. This function can be used to verify the flash.</p>

After pressing the Initialize button the Flash programmer connects to the kernel executing on the microcontroller. The Software will check whether the flash is empty or not. A detailed status of the individual sectors is presented on the right. This tab enable you to read or program single words, to erase single sectors or the complete flash, to program a mxh file or to readout the content of the flash.

If you just want to program a file you will have to press the Chip Erase button if the flash is not empty. After the flash has been erased you can choose a file with the browse button and press the “Prog File” button to program this file to the flash memory.

3.3 External Flash programming

Note: The Flashprogrammer currently only supports MBM29x series devices from Spansion with embedded auto algorithm

The FME FR Flashprogrammer can be used to write data to certain flash devices which are connected to the external bus of the microcontroller. External flash programming is only working with Spansion Flash devices which have an embedded flash auto algorithm. Configuring the external flash programming feature currently requires a good understanding of the microcontrollers external bus interface as well as a good understanding of the type of flash in use.

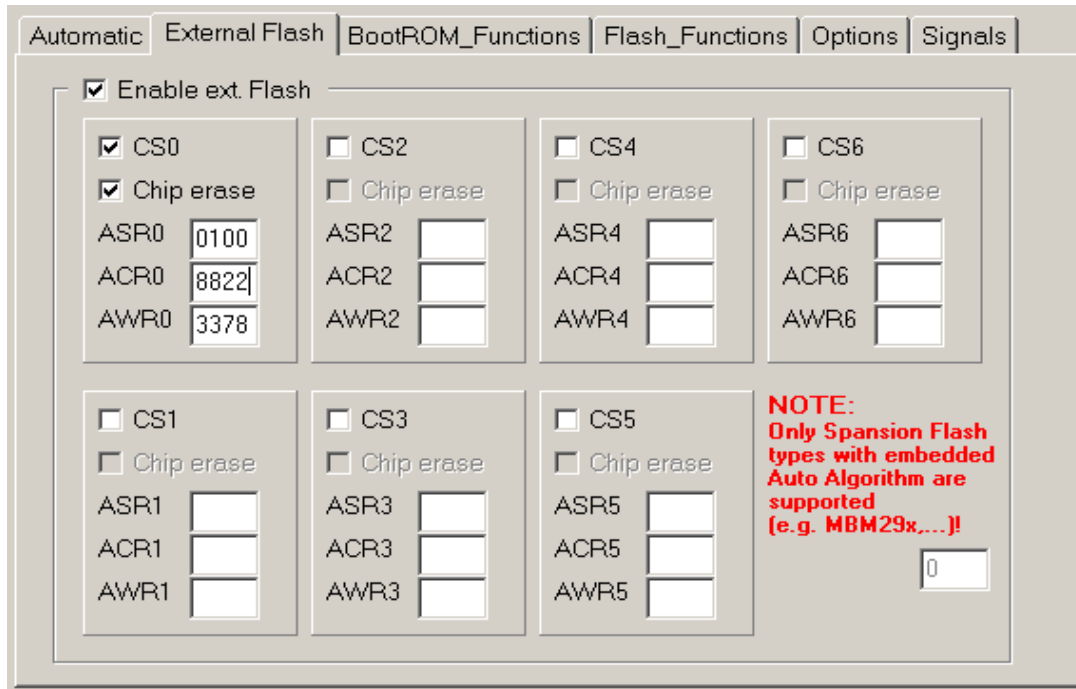


Figure 8 External flash configuration tab

The settings on the External Flash configuration tab are the same settings you would have to do if you would want to use the external flash in your microcontroller application. In real the Flashprogrammer just writes the values you enter in the edit boxes into the corresponding registers of the microcontroller. If you already have developed your application to use the external flash memory you can just copy the settings for the ASR, ACR and AWR register from your project. The settings shown in Figure 8 are working with most of the external flash chips we use on our starterkits. To use the external flash programming feature you will have to check the Enable ext. Flash checkbox and check the checkbox of the Chip select you are using on your hardware. You can also choose whether a Chip Erase should be done on the external flash device by checking the chip erase checkbox.

3.4 “Options” tab

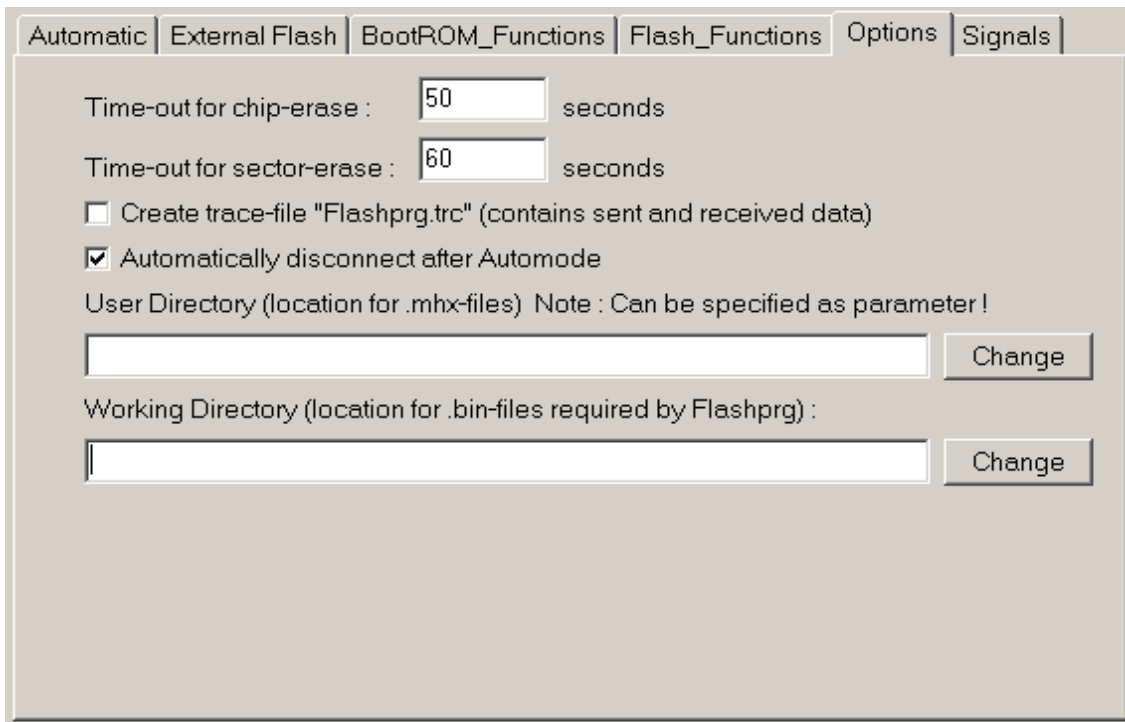


Figure 9 Option tab

Field	Description
Chip erase timeout	This is the time the Flashprogrammer waits during a chip erase before it aborts because of time out
Sector erase timeout	This is the time the Flashprogrammer waits during a sector erase before it aborts because of time out
Create Trace-File	If this box is checked, a file “flashprg.trc” will be created which contains all in- and out-going characters from this application. This file can be used to check for errors (knowledge of the protocol is required). Note: this file will be available after FLASHPRG has terminated.
Working Directory	Whenever this program is started, the actual location of the FLASHPRG.EXE will be treated as “Working Directory” where all necessary files (e.g. brloadxx.bin) must be placed. You may change this setting manually if binary files are placed in a different directory.
User Directory	The user directory is the location for the files to be programmed (mhx-files). This directory can be chosen independently from the working directory and can also be specified as command parameter (instead of programming file).

3.5 Command line options

The FME FR Flashprogrammer has various command line options which allow setting certain parameters. They can be used to integrate the FME FR Flashprogrammer into Softune Workbench or any other environment as scriptable tool.

The com port has to be passed first and the file to program has to be the last option. All other options can be passed in any order.

FLASHPRG [com] [...] [file]

e.g.: flashprg 1 -D13 c:\project\Standalone\ABS\sample.mhx

This will program sample.mhx to a MB91F467D via com port 1.

Note: When passing a filename as command line option the FME FR Flashprogrammer will automatically start programming in automatic mode.

Option	Description	Usage
com	Com Port 1-x	FLASHPRG 1
file	Filename and full path of the File to be programmed.	FLASHPRG 1 ↵ c:\project\test.mhx
-Dn	D0 = MCM D1 = MB91F361 D2 = MB91F365 D3 = MB91F362 (default) D4 = MB91F365,6,7,8,9Gx D5, D6, D7= ext. Flash (Cremson Starterkit) D8 = MB91F364 D9 = MB91F376G D11 = Flash16MB_CS2456 D12 = ADA_91V460_91F467D D13 = MB91F467D D14 = MB91F464A D15 = MB91F465K D16 = MB91F469G D17 = EMA91V460 D18 = MB91F463N D19 = MB91F467B D20 = MB91F465X D21 = MB91F467R D22 = MB91F467C	-D19

Option	Description	Usage
	D23 = MB91F467M D24 = MB91F467T D25 = MB91F465P D26 = MB91F467S D27=MB91F465C D28=MB91F465B D29=MB91F464H D30=MB91F469Q D31 =MB91FV460B	
-C (address)	call-address after writing the file	-C30000
-Q	Quit FLASHPRG after successful programming	-Q
-B(Baudrate)	selects programming baudrate (9600,19200,38400,57600,115200)	-B9600
-S5	ignore S5 record	-S5
-E (logfile)	creates an error-log file for batch-processing (see below) Error codes : 0 = success 1 = wrong command shell parameters 2 = flash erase error 3 = download error 4 = user abort 5 = file open error 6 = no connect 7 = command error 8 = wrong echo (protocol) received	-E filename
-ONLINE= [NONE/RTS/DTR]	select signal for online state	-ONLINE=DTR
-RESET= [NONE/RTS/DTR]] select signal for reset insertion	-RESET=RTS
-ICE	chip erase will not be executed in automatic session	ICE
-ATP=n	n = number of attempts	-ATP=42
-PSV	security vectors are programmed (MB91F46xx)	-PSV
-XEN	enable external flash feature	-XEN

Option	Description	Usage
-XCSn	enable chip select CHIPSELECT n for ext. flash n=1..6	-XCS1
-XCER	execute chip erase on external flash	-XCER
-XAWR= [WaitRegister]	wait register setting for external bus chip select area WaitRegister : 3378 (default)	-XAWR=3378
-XASR= [AreaSelect]	base memory address for chip select AreaSelect: 0100	-XASR=0100
-XACR= [Configuration]	Configuration for external bus chip select area chip select Configuration: 8822 (default: MB91F467D)	-XACR= [Configuration]
-RT=[t]	Reset time t (in ms): defines how long the reset signal is send.	-RT=100
-RD=[t]	Delay time t (in ms) after RESET at 'connect'.	-RD=50
-ASE=[sectors]	Will erase the given sectors in automatic mode instead of a chip erase (only if they are not blank), e.g.: -ASE=0,5,6	-ASE=[sectors]
-TB	transfers mhx-file as block (if supported by the flashloader)	-TB
-TBDB	Creates a new mhx-file (same name as the old +'_n') containing the compressed S-Record strings used by the block-transfer function.	-TBDB
-OFFSET=[offset]	defines an offset for the flash-programming (only if using -TB)	-OFFSET=[offset]
-TRACE	enables trace mode	-TRACE
-UCLK=[freq]	selects the frequency of the used oscillator (default 4 [MHz])	-UCLK=[freq]
-NOKLINE	disables automatic Kline detection	-NOKLINE
-TOCE= [seconds]	Chip erase timeout	-TOCE=60
-TOSE= [seconds]	Sector erase timeout	-TOSE=30
-TOECE= [seconds]	ext Chip erase timeout	-TOECE=200

4 Appendix

4.1 Information on Files to be programmed (MHX-Format)

The hex-file to be programmed to flash-ROM must be specified in the “file to program”-field (use the browse-button to select). This file must be a converted linker output file from Softune Workbench in the Motorola™ -Format (.MHX). If you can't find this file in your project directory, you probably haven't included the converter in your tool-chain. To do so, select “Project – Setup – Make/Build” from the Softune main menu and tick “Absolute module converter is started”. Be sure you have selected “Motorola S-Format” in your Converter settings (tool options). After a successful “Make” or “Build”, you should be able to find the “<projectname>.MHX”-file in the ABS-directory.

4.2 Using “K-line”

This software as well as the Boot-ROM are capable of using a “K-line”-type connection. This is a special 1-wire communication standard which multiplexes serial input and output to one line. K-line is mainly used in automotive applications for diagnostic purposes and therefore ideal for re-programming a microcontroller inside a control unit.

If you are using a “K-line”-connection in your application you can use the Flash Programmer as described in this manual without any further actions. The software will automatically detect the presence of a K-line connection because of the transmission echoes. If such a connection was detected, a message “K-line detected!” is displayed in the log-window and on the status bar.

Note that some K-line adapters use transistors instead of integrated circuits for multiplexing the signals. This can lead to a limited maximum baud rate. If you experience problems while flash-programming using a K-line connection, try to reduce the baud-rate.