Application Note

Programming Flash Microcontroller

F²MC-16LX Family

History

13th Aug. 99	TKa	V1.0	New Format, new updated version
17th Dec. 99	TKa	V1.1	3.5 Program Code Download, MDx mode pin setting description changed
01th Mar. 00	MSt	V1.2	MB90F497 added
16th Mar. 00	TKa	V1.3	Updated for new Flash500 programming software
17th Jan. 01	TKa	V1.4	Schematic for serial programming changed, recommedation added to connect HST directly to RST

Warranty and Disclaimer

To the maximum extent permitted by applicable law, Fujitsu Mikroelektronik GmbH restricts its warranties and its liability for **all products delivered free of charge** (eg. software include or header files, application examples, application Notes, 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 Mikroelektronik 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 Mikroelektronik GmbH warrants that the Product will perform substantially in accordance with the accompanying written materials for a period of 90 days form the date of receipt by the customer. Concerning the hardware components of the Product, Fujitsu Mikroelektronik 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 Mikroelektronik GmbH´s entire liability and the customer´s exclusive remedy shall be, at Fujitsu Mikroelektronik 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 Mikroelektronik 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 Mikroelektronik GmbH, or abuse or misapplication attributable to the customer or any other third party not relating to Fujitsu Mikroelektronik GmbH.

3. To the maximum extent permitted by applicable law Fujitsu Mikroelektronik 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 Mikroelektronik 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 Mikroelektronik 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.

1. Fujitsu Flash Microcontroller Overview

Fujitsu offers at this time the following Flash Microcontroller of the 16LX Family

MB90F497, 64KB Flash Memory, single CAN, external bus interface
MB90F523, 128KB Flash Memory
MB90F543, 128KB Flash Memory, double CAN, external bus interface
MB90F548, 128KB Flash Memory, single CAN, external bus interface
MB90F549, 256KB Flash Memory, single CAN, external bus interface
MB90F553A, 128KB Flash Memory
MB90F562/L, 64KB Flash Memory, Low Cost
MB90F568, 128KB Flash Memory, Low Cost
MB90F574, 256KB Flash Memory
MB90F583B, 128KB Flash Memory, 5 UART's
MB90F598, 128KB Flash Memory, single CAN interface
MB90F594/A, 256KB Flash Memory, dual CAN interface

1.1 Main Flash Features

- Single 5V power supply
- 10000 erase cycles
- 10 years data retention
- Different sector sizes available
- Sectors can erased individually
- In system programmability

1.2 Evaluation boards

For the ***MB90F523 and MB90F574*** series the FLASH-EVA2-120P-M13 evaluation board is available as a low cost multifunctional evaluation board. This Flash-Test-Board can be used as a simple target board for the emulator and as an evaluation board to download and program the software via RS232 connection and test the user application. This board is shipped with an MB90F523, programmed with a boot-loader to download software via RS232. The board does not offer a special ROM monitor debugger.
To download the user application the SKWizard, which is a terminal program with integrated software download function, or the HexloadW utility program from Fujitsu can be used.
Note: For Windows NT it could happen that the current version of the SKWizard does not work correctly.

For the ***MB90F553A, MB90F543, MB90F548, MB90F549, MB90F583, MB90F594/A, and the MB90F598*** the ***Flash-CAN2 –100P-M06*** evaluation board can be used. In the main it is used as an emulator target board. But it is also possible to use this board as a quite simple Flash evaluation board to test some functions of the user application. For the ***MB90F497, MB90F562/L, MB90F568*** the ***Flash-CAN-64P-M01/M09*** evaluation board can be used.

2.0 Programming Flash Microcontrollers

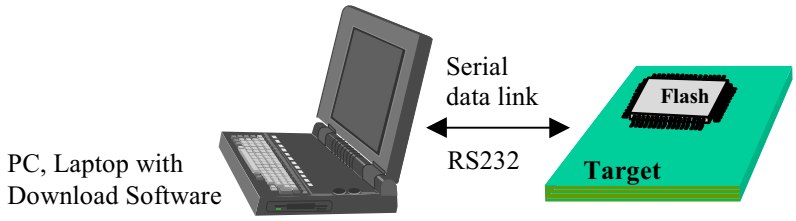
To program Flash microcontrollers there exist three possibilities:

2.1 Use an ordinary EPROM Programmer

- **Minato 1890A,** with Base OU910 unit, with programming adapter
- **Data I/O, SMS Sprint Programmer,** with programming adapter
- **Conitec Datensysteme (GalepIII) ,** with programming adapter
- **BP Microsystems,** with programming adapter
- **Stag,** with programming adapter

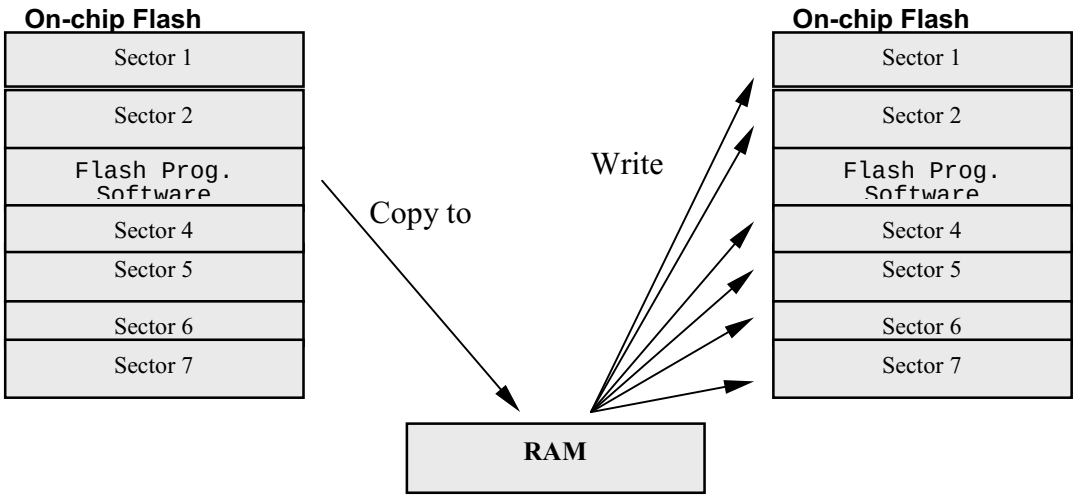
2.2 Fujitsu embedded Burn-IN ROM serial programming mode .

In this mode a standard asynchronous RS232 connection to a PC can be used to download the software and program the Flash Microcontroller (even blank devices) directly in the system. Therefore an IN-ROM bootstrap loader is used which is enabled by a special mode pin setting of the microcontroller. On the PC side a special Software is used. On the target system a RS232 interface must be assigned for the corresponding UART of the microcontroller. Additionally the Mode pins and two port pins must be used to set-up the microcontroller into this programming mode after Reset or power on. The download rate is about 9600Bit/s if a 4MHz crystal is used for the Microcontroller.
Note: Be aware that not all Flash microcontroller support this special serial-programming mode, therefore please see the table below.



2.3 User application software (also called **bootloader**)

With this method a user application software is started to download the program data and program the Flash memory. Therefore such a user application (bootloader) must be located in the Flash memory (programmed into the Flash via method 1) or 2)). This bootloader could be located in a small sector and is used by a special command to download the data (e.g. via RS232 or CAN interface) and to program the data into the Flash memory. During Flash programming the bootloader programming software itself must be running in the RAM area.
This method is available as an example application on the QFP120 Flash-Test-Board using the RS232 interface for data download. Furthermore bootloaders for the MB90F523, MB90F543, MB90F574, MB90F583, MB90F598, MB90F553A and MB90F594/A are available. Of course it is possible to modify the bootloaders corresponding to the needs of the application.



The user Flash programming software is copied to the internal RAM. After that, the program is started in the RAM and the Flash programming is executed.

The following table gives an overview about the Flash Microcontrollers and the Flash programming possibilities for these devices.

Device	SMS Programmer (Programming adapter)	Minato (Programming adapter)	Burn-In ROM Serial Programming Mode (UART used by µC)	Program example for a User Bootloader available (current Version)
MB90F497	--	--	UART1	--
MB90F523PFV	S5024	H910-262	Not supported	V3.0
MB90F543PF	S5023	H910-288A	UART1	V1.0
MB90F548PF	S5023	H910-288A	UART1	Same as for MB90F543
MB90F549PF	S5023	H910-288A	UART1	Same as for MB90F543
MB90F553APF	S5023	H910-288A	UART0	V4.0
MB90F562/L	--	--	UART1 (P60/61)	--
MB90F574PFV	S5024	H910-262	UART0	V1.0
MB90F583PF	S5023	H910-288A	UART0	V1.0
MB90F594PF	S5023	H910-288A	Not supported	V2.0
MB90F594APF	S5023	H910-288A	UART0	V2.0
MB90F598PF	S5023	H910-288A	UART1	Same as for MB90F594

3. Fujitsu embedded Burn-In ROM serial programming

Fujitsu Flash microcontrollers offer a quite simple way to program the Flash memory directly in the system, without the need of a special programmer. The Serial Asynchronous Programming Mode allows the user to program the microcontroller directly on the target system via a standard RS232 connection. So Software updates can be done very easily.

All you need to program the Flash Memory inside the microcontroller is a PC and a RS232 cable. For the microcontroller some additional circuits have to be assigned on the target board. The Block diagram (Figure1) shows all necessary connections, which has to be done to program the Flash memory.

3.1 Serial Interface of the Microcontroller

To connect the microcontroller to the serial interface only the Transmit (SOT) and the Receive line (SIN) of the UART are necessary. For voltage level conversion a RS232 driver has to be used (e.g. MAX232). The RS232 driver can be assigned directly on the target board or it can be mounted on an adapter, which is connected to the target to program the device. This solution saves some space on the target and minimises the costs.

Note: The MB90F553A, MB90F574, MB90F583, MB90F594A use the UART0 interface, the MB90F497, MB90F562/L, MB90F543, MB90F548, MB90F549 and MB90F598 use the UART1 interface. For the MB90F594A exist a separate PC programming software, which must be used for the in-circuit programming.
At this time only these devices can be programmed using the Fujitsu Burn-In ROM Serial Asynchronous Programming Mode.

3.2 Serial interface of the PC

To establish a connection to the microcontroller the RTS (Request To Send) and the CTS (Clear To Send) lines of the PC interface must be connected together. Also the DTR (Data Terminal Ready) must be connected to the DSR (Data Set Ready) signal. Otherwise the program cannot send any data to the microcontroller and a “Communication Error” occurs. The Transmit line (TXD) of the PC interface has to be connected to SIN, the Receive line (RXD) has to be connected to SOT via the RS232 driver. The GND line of the RS232 interface must also be connected.

3.3 Mode Setting for the Microcontroller

To program the Flash memory of the microcontroller it is necessary to change the operating mode of the controller to the Asynchronous Serial Programming Mode. Therefore the mode pins MD0, MD1, MD2 and the port pins P00 and P01 are used. Table1/2 show the required settings.

Pin	**MD2**	**MD1**	**MD0**
Level	**HIGH**	**HIGH**	**LOW**

Table 1

Pin	**P01**	**P00**
Level	**LOW**	**LOW**

Table 2

With these settings the microcontroller enters the Asynchronous Serial Programming Mode after power-on or after generating a Reset using /RST.
It is recommended to connect /HST directly to /RST. It might be useful to connect the reset line /RST to a button to generate a Reset. Otherwise Power-on must be used to change the operating mode.
Note: Using only the /HST instead of /RST to generate a microcontroller Reset will not set the microcontroller into the Asynchronous Serial Programming Mode. The usage of /RST is mandatory to change the operating mode! Nevertheless the /HST has its own functionality which must be considered in the design.

3.4 Installing the PC Software

To install the PC Software just run the self-extracting file *Flash500.*. This will install the corresponding Flash programming software by default in c:\Softune\Utility\Flash500. But any other directory can also be used.

Note1: In some rare cases "error 103, wrong write file" can occur. As workaround please copy the motorola S-Record which should be programmed to the internal Flash memory (myname.mhx) to the installation directory of the Flash500 software or try to install the Flash500 directly in the root directory.

Note2: In some cases it can happen, that this software version of the software does not fully support Windows NT. This is because some procedures are done using batch files. So it could happen that an error occurs using Windows NT (Error 103, wrong write file). In case of error, please look at the c:\softune\flash500 directory and make a copy of the temporary created batch file. Start the batchfile separately first. After that the Flash programming software itself can be started. The batchfile creates a work.bin file, which is used by the Flash programming software.

3.5 Program Code Download

To download data to the microcontroller, the Flash programming Software must be started. After that, first the corresponding device must be selected from the device list and the clock frequency must be set corresponding to the external clock frequency used for the controller. Only the following external clock frequencies are allowed: 4, 8, and 16MHz. Other frequencies will not work because the baud rate is fixed in the Burn-In ROM. After that select the Com port which is used for the download.
Now the <Download> command can be executed. Now, first the Flash programming routines itself will be downloaded into the RAM of the microcontroller. A "Download ok" message confirms that the connection to the controller could be established and the software is now ready to start. After the download, the Baudrate is increased to download the application at a higher transfer speed. The following table gives an overview.

External Clock Frequency	Baudrate used to download Flash Programming routines	Baudrate used to download the application itself (used with erase -, write & verify -, read -, blank check -, auto command)
4MHz	4800	9600
8MHz	9600	19200
16MHz	19200	38400

For the next step the file which has to be downloaded must be selected. This is done via the <search> command. The file itself must be a Motorola S2-Record and the S-Record must contain a start record (S0) and an end record (S8). Otherwise a temporary generated binary file (_w_o_r_k.5, which is used for the download) cannot be generated and the error message "unable to open _w_o_r_k.5" occurs. The S0 and S8 record is only used to recognise the start and the end of the data section during the transfer.
Note: Check the Motorola S-Rec for S0 and S8 Record!
With the <Auto> command a complete programming sequence is started: Download, Erase complete Flash memory, Blank check, write file to Flash memory and verify data.
Of course single commands like <Erase>, <Blank Check>, <Write&Verify>, <Read and Compare> and <copy> can be executed as well.
After the programming sequence the application can be started. Therefore the operating mode of the microcontroller must be changed to the corresponding setting which is necessary for the application (e.g. single chip mode: MD2=0, MD1=1, MD0=1). After that, a Power-on Reset or pressing the Reset button will start the application in the Flash memory.

3.6 QFP120 evaluation board Preparing

If the QFP120 Flash evaluation board is used to program the MB90F574 device make sure that the Reset LED is off after power-on. Otherwise change the DTR polarity with jumper J20 or just remove the jumper to avoid any influence of the DTR line during the programming sequence. To generate a correct Reset, the jumper J21 must be set to /RST. To enable the Serial Programming Mode, the Mode pins must be set correctly. Therefore the

switch S1 must be set for MD2, MD1 and MD0 to position ON. The connection of the port pins P01 and P00 to GND must be done with two additional wires. The UART0 of the MB90F574 is connected to the DB-9 connector by plugging the RN4 resistor network in the position MB90570 and setting the jumpers J12, J13, J14 to Burn-In position. The RTS / CTS and the DTR / DSR signals can be wired together on the evaluation board as well (RTS pin 7, CTS pin 8 of the DB-9 connector). The serial cable must be a straight 1:1 connection.
After the download the Mode pin setting must be changed to the corresponding setting of the application (e.g. single chip mode: MD2 Off, MD1 ON, MD0 Off – 0 1 1). A Power-on Reset or pressing the Reset button will start the application in the Flash memory.

3.7 Preparing the Flash-CAN evaluation board

To program a microcontroller on the Flash-CAN evaluation board it is necessary to connect the RS 232 interface to the PC. Take care that the correct serial channel of the microcontroller is connected to the RS232 interface connector. Therefore the jumpers JP7, JP8, JP9, JP10 must be used. The following Table shows the settings for the corresponding devices (Flash-CAN board Rev. 1.2 or later):

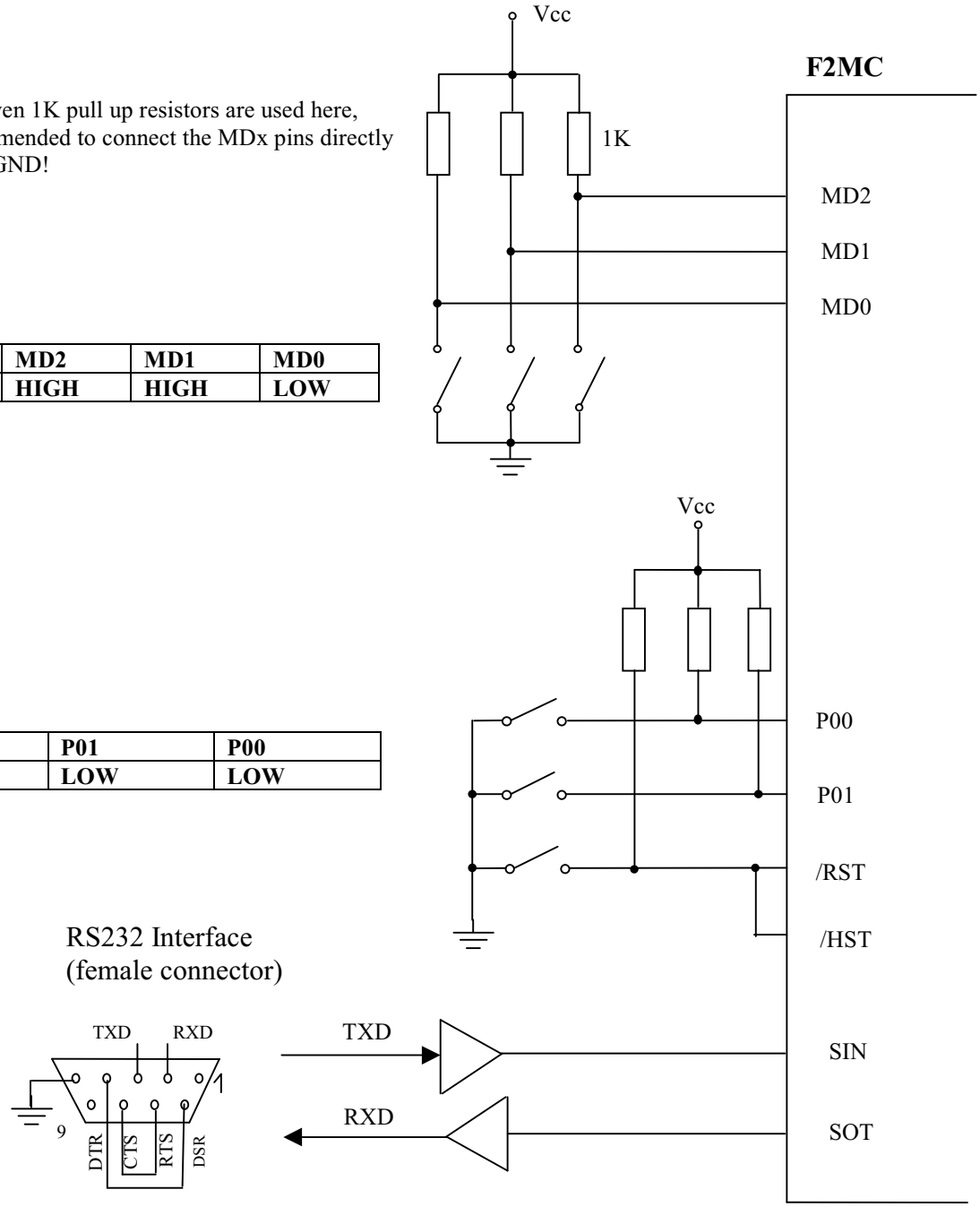
Device	**JP7**	**JP8**	**JP9**	**JP10**
MB90F543	--	--	1-2 (P24)	1-2 (P21)
MB90F548	--	--	1-2 (P24)	1-2 (P21)
MB90F549	--	--	1-2 (P24)	1-2 (P21)
MB90F553A	1-2 (P19)	1-2 (P20)	--	--
MB90F583B	1-2 (P19)	--	JP9 P18 - JP10 pin2	--
MB90F594A	2-3 (P14)	2-3 (P16)	--	--
MB90F598	--	--	1-2 (P24)	1-2 (P21)

For the serial PC interface it is also necessary to connect RTS with CTS. DTR is already connected to DSR on the board.

After that the Mode pins and the port pins P00 and P01 must be switched to the right position.
To set the mode pins and the port pins for serial async. programming the DIL switch S3 must be set as follows:
SW1 ON, SW2 OFF, SW3 OFF, SW4 OFF, SW5 ON, SW6 OFF, SW 7 ON, SW8 ON. SW1 to 3 are responible for the Mode pins, SW5 to connect RTS to CTS, SW7 and SW8 set the port pins P00 and P01to GND. Additionally jumper JP12 must be removed to avoid that the DTR signal resets the board.
After these settings the Flash programming software can be started. After programming the application can be started. Therefore the switches SW7 and SW8 should be set to OFF and the Mode pins must be set corresponding to the application. For single chip Mode the setting is: SW1 OFF, SW2 OFF, SW3 ON. After Power-on or Reset the application will start now.

NOTE: MB90F543/548/549 programming
The MB90F543 has a Flash security feature, which is enabled by setting bit 0 of address $FE0001. This enables a read protection for the internal flash memory, so nobody can read out the flash memory anymore (For detailed description please see Hardware Manual of the MB90540/5 series).
If the flash security bit is set, it is not possible to program or erase the Flash with the Flash361/400/500 program anymore!
The reason for this is that only a chip erase command is excepted when the security is set. But the Flash362/400/500 software uses subsequent single sector erase commands to erase the chip.

Figure 1
Hardware configuration for Serial Asynchronous Programming



NOTE: Even 1K pull up resistors are used here, it is recommended to connect the MDx pins directly to Vcc or GND!

Pin	MD2	MD1	MD0
Level	HIGH	HIGH	LOW

Table 1

Pin	P01	P00
Level	LOW	LOW

Table 2

4. Working with a User Application Software (Bootloader)

4.1 General Description

As already mentioned there exist an Application Note for an examples of a user bootloader for the MB90F523, MB90F543, MB90F553, MB90F574, MB90F583, MB90F594/A and the MB90F598. The functionality of such a bootloader will be described now, with this bootloader for the MB90F523 as a reference. Take care that the bootloader examples supplied for other devices may slightly be different. There are some small differences regarding baud rate and handling of the application reset vector. Therefore please see the table „Bootloader Selector Guide“ afterwards.
NOTE:
The bootloaders have been carefully checked and are believed to be reliable. However, no responsibility is assumed for any inaccuracies. Fujitsu does not accept any liability arising out of the use of this application example.

All bootloaders use the UART of the microcontroller for communication and download. Only the transmit and receive line of the corresponding UART interface are used. After Power-on the user has 1sec to send the character <ESC> via a terminal to the microcontroller. Therefore e.g. the SKWizard terminal emulation program can be used. If no <ESC> character is sent, the application is called after a timeout of approximately 1-second.
The SKWizard offers some more features, especially to download software. After sending the character, the bootloader enters a monitor mode and a prompt (>) appears on the terminal. Now some commands can be entered (commands are listed below) or the download of the application software can be started. With the SKWizard the download can be started by the load button or with the following command line:

```
SKWizard 1 -sk16 -i19200 -r -c %x\%A.cnv
```

Instead of using the SKWizard it is also possible to use the HEXLOADW download-utility to transfer the linker-output file. The settings (*Option-SetUtility*-menu) should be

```
HEXLOADW.EXE 1 -Flash -w -c -i19200 %x\%A.cnv
```

HEXLOADW automatically erases the flash memory sectors respectively and downloads the application program code.

4.2 Different methods to reprogram the flash memory

There are two methods to download programs into Flash memory, ***Method A*** and ***Method B***.
In the previous example, the application has been programmed into the flash-memory using *Method A*.
Method A: The entire sector will be erased before a byte will be written to it. Method A does not check, whether the content of the sector is different from the data that should be written to it or not. Hexloadw supports only this method. SKWizard also works with *Method B*.
Method B: In this case the code will be transferred twice to the Flash-Board. During the first transmission, the flash-monitor checks for differences between the actual contents and the received data. During the second transmission, it will only erase those sectors, to which new data should be written. This method also takes into account, that a flash memory cell only has to be erased, if their data-bit has to be reprogrammed from logical “0” to “1”.
To choose Method B with the SKWizard add an the *–flash* option in the command line:

```
SKWIZARD.EXE 1 –flash -sk16 -i19200 –d
```

4.3 Application Call

After Power-on of the MCU, the bootloader always takes control, at least for the first steps to check the cause of reset.
After *Power On* the version string of the bootloader “MB90523 Flash loader - V2.0” is output and you have the chance to respond with *ESC* in order to enter the monitor mode of the bootloader.
If „ESC“ is pressed during the timeout period, the program execution branches to the Flash Monitor.

Note that the loader reads the WDTC (WatchDog Timer-Control) register in order to find out the cause for reset. While reading WDTC, the content is destroyed, but through the R0 register, a copy of the content is passed to the application software for further evaluation.
If you download an application, which normally has its own reset vector at H'FFFFDC, this "application vector" will be stored in INT#7 location at address H'FFFFE0. Therefore a vector call for INT#7 is no longer available in conjunction with this boot loader. The boot loader itself takes care of this **vector displacement**. This way of handling with the reset vector is very flexible because the "application vector" can be defined in the software project inside *Softune* as the normal reset vector and is not fixed to a certain address.

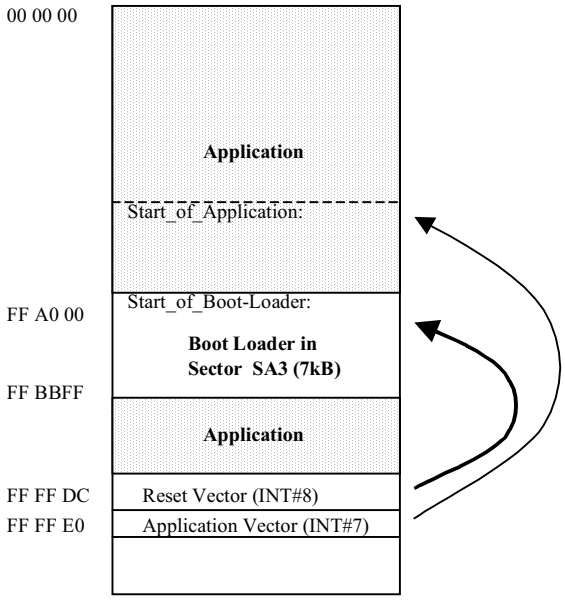


Figure 13: Location of reset vector and application vector

4.4 Special Note for MB90F553 V2.3 Bootloader

The bootloader for the MB90F553 V2.3 does not support this INT#7 feature. For this bootloader the application start address is fixed to H'FF0000. So if the bootloader V2.3 of the MB90F553A is used, the application must start at address H'FF0000 and no Reset Vector may be generated by the compiler for the application. Otherwise the Reset Vector of the bootloader would be overwritten, which is protected by the bootloader software. But if a reset vector is included the download will be stopped.

4.5 Bootloader Commands

After Power-on the message "MB90523 Flash loader - V2.0" appears as the first prompt at the terminal. Respond with "ESC" within 1 second enters the monitor mode and a prompt ">" appears. In the monitor mode of the bootloader the following commands can be entered:

4.5.1 Command overview

- RM AAAAAA NN : "Read Memory": NN bytes from location AAAAAA[*)]
- ES NN : "Erase Sector" : Erases sector NN
- G : "Go" : Call user application
- CALL AAAAAA : "Call" : Call address AAAAAA
- PROTECT_RESET_VECTOR OFF : Disables write-protect function for reset-vector
- PROTECT_RESET_VECTOR ON : Enables write-protect function for reset-vector
- PROTECT_SECTORS OFF : Disables erase-protect function
- PROTECT_SECTORS ON : Enables erase-protect function
- CS OFF : Switch checksum-algorithm off
- RST : Software reset of MCU
- HEXDWL : Switch to Download-Mode (Method A)

- HEXDWL1 : Switch to Download-Mode (Method B – first step)
- HEXDWL2 : Switch to Download-Mode (Method B – second step)
- (Escape) : any ESC aborts the current function

*) AAAAAA: 24-bit upper case hex address, NN: two-digit upper case hex number

Defaults:
PROTECT_SECTORS **ON**
PROTECT_RESET_VECTOR **ON**
to protect the monitor from erasing the reset vector and itself.

4.6 Modifying the Bootloader

The provided bootloader is just an example how to generate such kind of "bootloader". It can be adapted to the needs of the application. For that purpose the source code of the current monitor must be modified. Therefore the following instructions must be considered:

a) During erasing and reprogramming of the on-chip flash, the code must be **executed from the RAM**! Therefore, the monitor copies part of the code from ROM into the RAM area. For the Softune Workbench, the linker can provide a dedicated mechanism for that purpose: RAMCode. This feature allows to link code to ROM, which will be executed in the RAM later on.
(**Note**: When you are still working with the old Softune V01 development environment the following issues must be considered: For the linker, the flash routines of the code must be placed in a RAM-location during compilation-time. So this part of code has to be copied back to ROM before programming a new monitor into the Flash memory. To do so, use the BINHEX-tool (Version 2.1 or later):
BINHEX /z=FFA000 *NAME*.cnv /y /a
This will copy all code linked to RAM-locations (below 1000_{hex}) to $FFAxxx_{hex}$. The output file with the corrected load-module will be in Motorola S-format with the extension *.mhx by default.)

b) If the device supports the serial Burn-In ROM programming, it is possible to download the bootloader via RS232 to a blank device using the special serial programming software (described in chapter 2).

c) A blank MB90F523 or MB90F594 devices can only be programmed using an EPROM-programmer like the MINATO 1890A or the Data I/O Plus48. A special programming adapter is necessary for each programmer.

4.7 Updating the Bootloader

If a bootloader is already programmed in the flash memory it is possible to update the bootloader by using the bootloader itself. Therefore the following procedure must be done:
Before downloading the new bootloader to the flash device , also the reset-vector protection and sector-protection of the current bootloader must be switched off. Otherwise, any attempt to overwrite the current version of the bootloader will result in an error. Use the commands:

PROTECT_RESET_VECTOR **OFF** and
PROTECT_SECTORS **OFF**

from the SKWizard or any terminal program to allow overwriting of the bootloader area and download the new *bootloader* file to the flash memory. After reset (/HST or Power On) the new version of the bootloader should appear. Take care that during the programming sequence the power supply is not switched off. Otherwise the programming could fail and the bootloader will not work anymore.

5. Command Line Options

5.1 HEXLOADW

HEXLOADW {PORT} [-R|-C] [-W] [-Flash] [-Ibaud] [file]

PORT This is the communication port that the board is connected to.
-W Wait for target (Power on)
-C Close HexloadW after download
-R Run program, do not use for Flash boot loader
-Ibaud Initialise serial port to baud (300, 1200, 2400, 4800, 7200, 9600, 19200, 38400).
-Flash Flash-programming using Method A

Examples:

HEXLOADW 1 –i38400 -flash –w myprog.cnv
Waits for the target to be reset/switched on, programs myprog to flash via COM-port 1 using method A

HEXLOADW 2 –i19200 –flash –c myprog.cnv
Programs myprog to flash via COM-port 2 using method A, and shuts down

5.2 SKWIZARD

SKWIZARD [P] [-IBaud] [-SKType] [-D] [-R [-C]] [File]

P COM-Port Number (1,2,3,4)
Baud Baudrate (300,1200,2400,4800,9600,19200,38400)
Type Starterkit-Type (8,16,32)
-D Detect Baudrate (synchronises Baudrate with board)
-R Run after load (only useful if file is specified)
-C Close program after a successful load and execution (only if -R is specified)
-F Flash-programming using Method B

File (last parameter given) should contain full path and filename+ext of hex-file

Examples:

SKWIZARD 1 –SK16 –i19200
Simply opens the application for use with the Flash-Board via COM-port 1

SKWIZARD 1 -SK16 -i19200 myprog.cnv
As above, but will program myprog to flash using method A

SKWIZARD 2 -SK16 -i19200 –f –r -c myprog.cnv
Now uses COM 2 and will program myprog to flash using method B, then execute it and shuts down

5.3 BINHEX

BINHEX /o=-FF0000 /z=FFA000 /m test.cnv /o=FF0000 /a

/o is the read offset (in this case negative)
/z is new offset for records below 1000 hex
/m is for Motorola format
test.cnv is the input file
/o is the write offset (to put the record back where it started)
/a is to change addresses only

6. Bootloader Selector Guide

Device Loader name	Loader address	Appl. start address / reset vector	Start condition	Comm	Loading
MB90F594 MB90F598 V2.0	Fixed boot sector: FFA000 – FFBBFF The binary of this loader fits for both MB90F590 and 595 Be aware that the e 590 header files do not completely fit for 595.	Reset vector **must** be used, Vector stored at FFFFE0 (int#7) by loader Dummy application included, which toggles port40	- Boot loader checks WDTC, if POWER-ON only → start main loader, if other reset cause → jump to application pointed by FFFFE0 - Main loader displays WELCOME message - waits 1 second for any character, if not → PLL off, generate software reset, if something → command line mode - WDTC stored in R0	UART0 at 38400 Baud (PLL x4)	loading by SK-Wizard: - power-on target - press ESC after message - proceed as usual loading with Hexloadw: - call "hexloadw 1 -flash -w -c path\name.cnv" - power-on target
MB90F553 V4.0	Sector: FFA000 – FFBBFF	Reset vector **must** be used, vector stored at FFFFE0 (int#7) by loader (vector displacement) dummy application included, that does a endless loop	- Boot loader checks WDTC, if POWER-ON or HST only → start main loader, if other reset cause → generate software reset and start again - Main loader displays WELCOME message - waits 1 second for ESC, if not → PLL off, generate software reset, if ESC → command line mode - WDTC stored in R0	UART0 at 19200 Baud (PLL x4)	loading by SK-Wizard: - power-on target - press ESC after message - proceed as usual loading with Hexloadw: - call "hexloadw 1 -i19200 -flash -w -c path\name.cnv" - power-on target
MB90F523 V3.0	boot sector linked to: FFA000 – FFBBFF	Reset vector **must** be used, Vector stored at FFFFE0 (int#7) by loader Dummy application included, which toggles port P10	- Boot loader checks WDTC, if POWER-ON or HST reset → start main loader, if other reset cause → jump to application pointed by FFFFE0 - Main loader displays WELCOME message - waits 1 second for any character, if not → PLL off, generate software reset, if something → command line mode - WDTC stored in R0	UART0 at 19200 Baud (PLL x4)	loading by SK-Wizard: - power-on target - press ESC after message - proceed as usual loading with Hexloadw: - call "hexloadw 1 -flash -w -c path\name.cnv" - power-on target

Device Loader name	Loader address	Appl. start address / reset vector	Start condition	Comm	Loading
MB90F543 V1.0	Fixed boot sector: FFA000 – FFBBFF	Reset vector **must** be used, Vector stored at FFFFE0 (int#7) by loader Dummy application included, which toggles port40	- Boot loader checks WDTC, if POWER-ON only → start main loader, if other reset cause → jump to application pointed by FFFFE0 - Main loader displays WELCOME message - waits 1 second for any character, if not → PLL off, generate software reset, if something → command line mode - WDTC stored in R0	UART0 at 38400 Baud (PLL x4)	loading by SK-Wizard: - power-on target - press ESC after message - proceed as usual loading with Hexloadw: - call "hexloadw 1 -flash -w -c path\name.cnv" - power-on target
MB90F583 V1.0	Fixed boot sector: FFA000 – FFBBFF	Reset vector **must** be used, Vector stored at FFFFE0 (int#7) by loader Dummy application included, which toggles port40	- Boot loader checks WDTC, if POWER-ON only → start main loader, if other reset cause → jump to application pointed by FFFFE0 - Main loader displays WELCOME message - waits 1 second for any character, if not → PLL off, generate software reset, if something → command line mode - WDTC stored in R0	UART0 at 19200 Baud (PLL x4)	loading by SK-Wizard: - power-on target - press ESC after message - proceed as usual loading with Hexloadw: - call "hexloadw 1 -flash -w -c path\name.cnv" - power-on target
MB90F574 V1.0	Sector: FF8000 – FF9FFF	Reset vector **must** be used, vector stored at FFFFE0 (int#7) by loader (vector displacement) dummy application included, that does a endless loop	- Boot loader checks WDTC, if POWER-ON or HST only → start main loader, if other reset cause → generate software reset and start again - Main loader displays WELCOME message - waits 1 second for ESC, if not → PLL off, generate software reset, if ESC → command line mode - WDTC stored in R0	UART0 at 38400 Baud (PLL x4)	loading by SK-Wizard: - power-on target - press ESC after message - proceed as usual loading with Hexloadw: - call "hexloadw 1 -i19200 -flash -w -c path\name.cnv" - power-on target

NOTE: The bootloaders have been carefully checked and are believed to be reliable. However, no responsibility is assumed for any inaccuracies. Fujitsu does not accept any liability arising out of the use of this application example.