Application Note

The Bootconcept

of Fujitsu's MB91360 Devices

History

13th Aug. 99	MM	V1.0	New Format, new updated version
04th Jul. 00	MEN	V1.1	Updated Application

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.

The development of the new 32-Bit RISC MCU-family MB91360 will be completed soon with the flash version MB91F361. In order to support serial flash programming, a new bootconcept was developed for these devices.

Having 512kB FlashROM onboard, the ‘360-devices are well prepared for complex applications which require large code and constant areas. In all Flash-MCUs, Fujitsu offers three different ways to (re-)program the internal FlashROM :

1.) Parallel programming using a standard EPROM-programmer together with the appropriate socket (footprint-converter).

2.) Serial programming using a built-in Boot-ROM for handling the data transfer.

3.) In-Circuit programming – the application uses it’s own software-method in order to customize the programming algorithms (e.g. re-programming via CAN)

The serial programming is the most flexible method, because it has the advantage of being available even when the device is mounted on the board and already in use. It can be used for field-updates (complete or partly updates the FlashROM), for diagnostic purposes (e.g. reading SW-version-information), low-cost developments (where no in-circuit emulator can be used) or prototype productions.

Boot-ROM

Software-routines which are required for handling the serial data are stored inside a fixed Boot-ROM, which is present in all MB91360 devices. Compared with the concept used in the 16-Bit devices, where the internal Boot-ROM is only accessible when external Mode-Pins will be changed, the Boot-ROM of the MB91360 devices is mapped into memory at all times and will be called after each reset (see fig.1).

The BootROM is located at address FF000 and has a size of 2kB. Since it’s connected to the internal bus system, no chip-select areas must be assigned before executing code from this location.

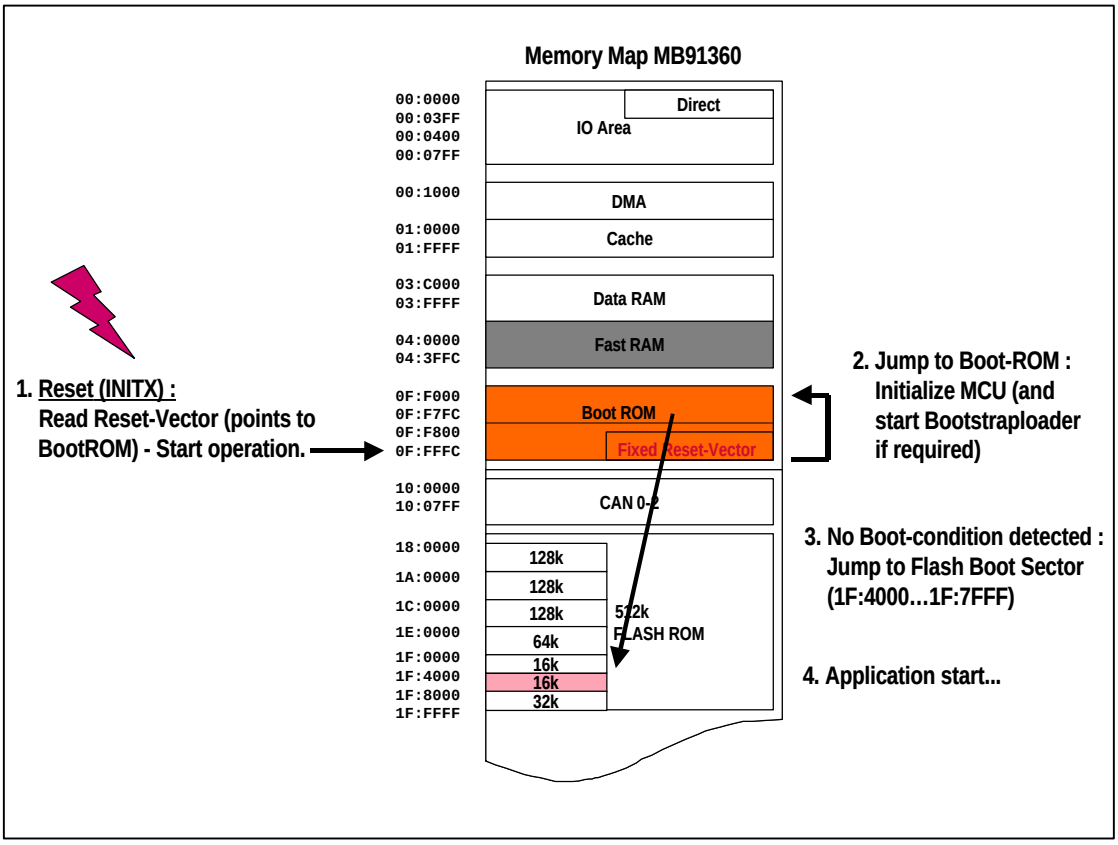

Figure 1 : Reset Sequence of MB91360

The device will start executing the Boot-ROM immeadiatly after each reset and initializes the chip (CS-settings, clock-supply etc.). Then a number of conditions will be checked to determine wheather to start the actual bootloader or to start the current application.

When the device does not start from a INITX (external power-on reset), the bootloader cannot be started. Then the Boot-ROM enters a short checkloop in order to verify the actual boot-conditions. This is either a logic “1” on the external bootpin or the reception of a certain character from the internal UART0 at 9600 Baud (serial trigger). Figure 2 shows the boot program flow.

The possibilty to use a serial trigger is a major advantage. This is because in the final application, no further hardware modifications are required to enable the re-programming. Since the actual communicaton for downloading the new data to FlashROM will usually be transmitted over this same serial line, the only necessary preparation is to have the serial line available outside. In many cases, there is a serial interface for diagnostic purposes anyway. So, one would have to connect to this interface using the programmer-software, power-up the application and re-programming will start automatically.

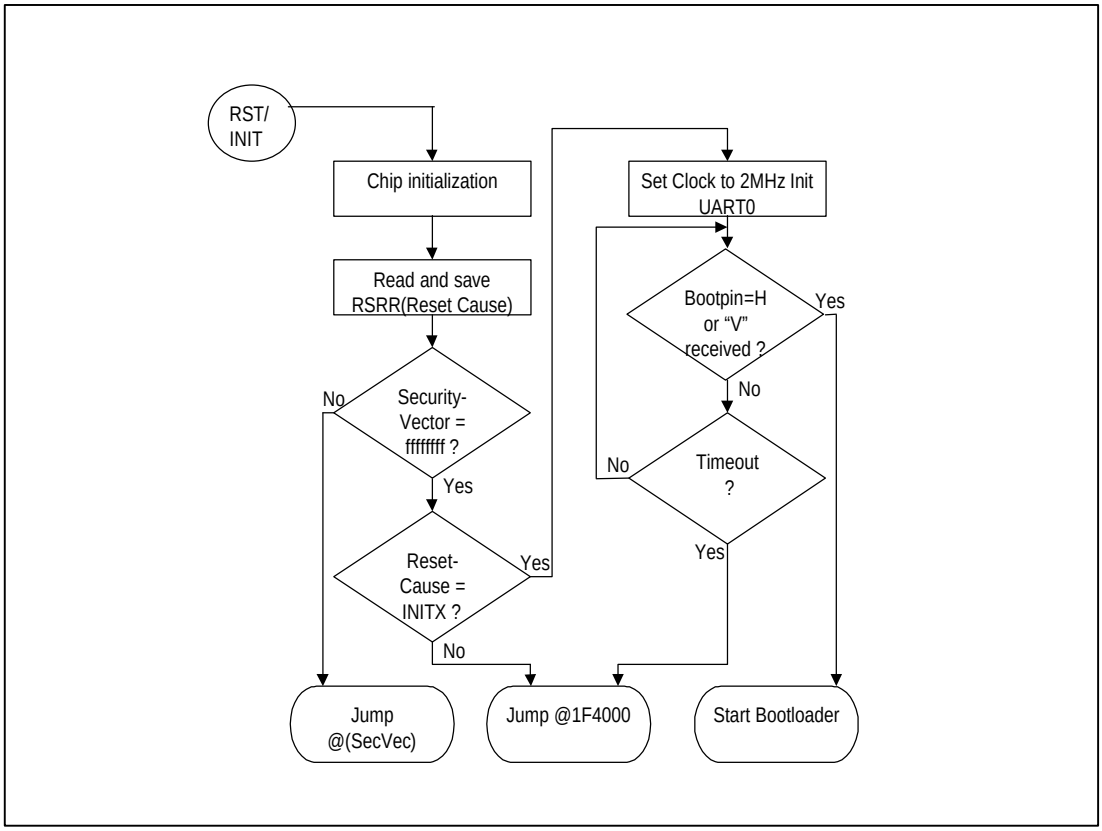

Figure 2 : Flowchart of the internal Boot-ROM

Security

In many applications, protection of the internal FlashROM is an important topic (preventing from unwanted reading, erasing or programming). Therefore, the bootconcept includes the checking of a special vector ("security-vector"). Whenever this vector location is blank (=FFFFFFFF), the BootROM can be used to invoke re-programming. But if any value is written to it, the BootROM will immeadiatly jump to this location after the chip-initializations. No more condition-check will be performed. This protection method has the advantage that the user can either simply disable the bootfeature or to divert the execution to a dedicated location in order to execute customized boot-routines.

Protocol

Once the actual bootloader has been entered (valid boot-condition found), an acknowledge message will be sent to the external device (normally a PC) and the initial communication can start. Four commands are accepted :

- Receive and write to a specified memory block
- Dump the contents of a specified memory block
- Initiate a "CALL" to a certain location
- Re-dump a calculated checksum for verification

Again for security reasons, the BootROM contains no functions, which directly program or even erase the flash. The reason for this is the risk of branching unintentionaly to a location within the BootROM (e.g. by a corrupted software pointer or uninitialized interrupts etc.) which could lead to a complete ROM loss ! Therefore, any routines to handle flash-programming must be downloaded to RAM first and can be executed from there. The so-called “FastRAM”, which allows to load and execute instructions was implemented mainly because of this reason. Quite clearly, any programming routine must not be executed from the flash itself, since the flash-logic will disable flash accesses during any of the “embedded commands”.

Software

A first version of the “MB91360 Flash Programmer” was developed using the emulation device MB91V360. It allows to either invoke all possible BootROM or flash-programming functions manually or to do all necessary steps automatically in one go (“Auto mode”). This tool accepts the converted linker-output from Softune Workbench (the development environment) directy without any pre-processing. It also can be integrated into the tool-chain and would then be controlled by command-line options.

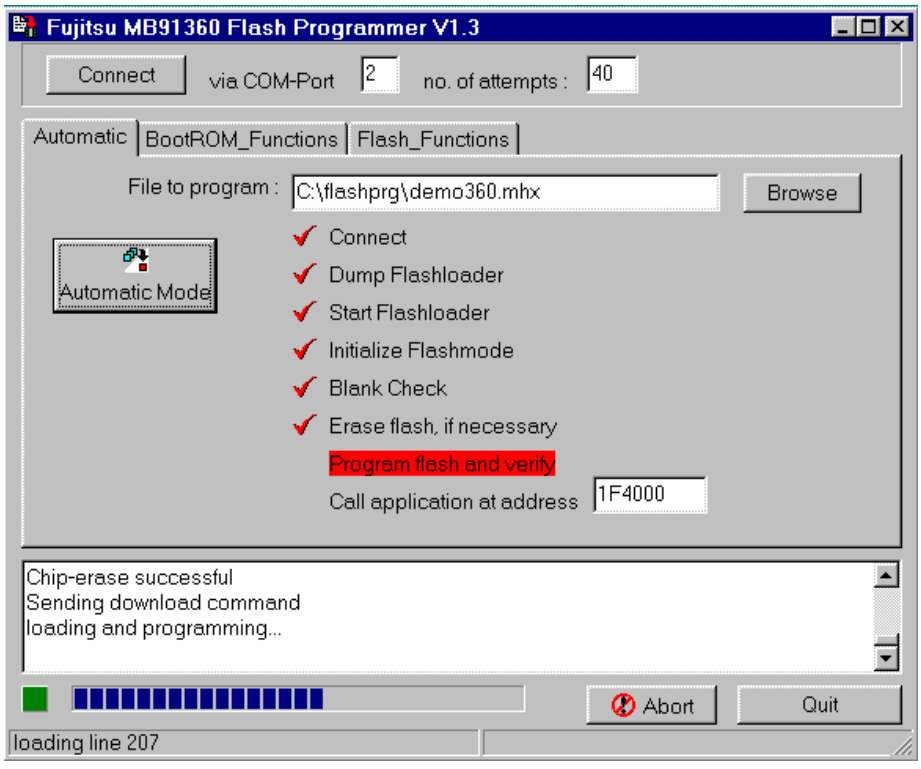

Figure 3 : Serial programming tool for the MB91360 series

Markus Mierse