



# **Programming Flash Microcontrollers through the Controller Area Network (CAN) Interface**

## Programming Flash Microcontrollers through the Controller Area Network (CAN) Interface

### Abstract

This application note describes how to program the Fujitsu Flash Microcontrollers through a CAN interface. For reference purposes, this document includes sample programs and a hardware configuration.

### Introduction

Fujitsu offers microcontroller families with both on-chip Flash memory and CAN controller. These features allow Fujitsu microcontrollers to be widely used in industrial automation as well as in automotive and mobile machines.

This application note provides a solution for programming the Fujitsu Flash Microcontrollers through the CAN interface.

Numbers in parentheses throughout this application note refer to references at the end of the document.

### Programming Fujitsu Flash Microcontrollers

There are three ways to program Flash Microcontrollers (1):

1. Using the EPROM Programmer
2. Using the Fujitsu embedded Burn-in ROM serial programming mode:

In this mode, a standard asynchronous RS232 connection to a PC downloads the software and programs the Flash Microcontroller directly in the system. The built-in ROM bootstrap loader is enabled by a special mode pin setting of the microcontroller.

3. Using the bootloader:

The bootloader, a user application, is located in the Flash memory and is programmed into the Flash with Method 1 or 2 above. During Flash programming, the bootloader program itself must be running in the RAM area. 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 application described in this document uses this method. By adding the CAN control function and other control schemes, the user bootloader can program the Flash via the CAN interface.

### Implementation

This example application allows the user to program the Fujitsu Flash Microcontrollers in the field through the CAN interface. Figure 1 shows the configuration of this application and assumes the following: There is a CAN network; all microcontrollers (nodes) have been mounted; CAN is the only connection among these nodes. If a need should arise requiring an upgrade or change of functionality of some of the nodes, the Flash microcontrollers can be reprogrammed using the CAN interface.

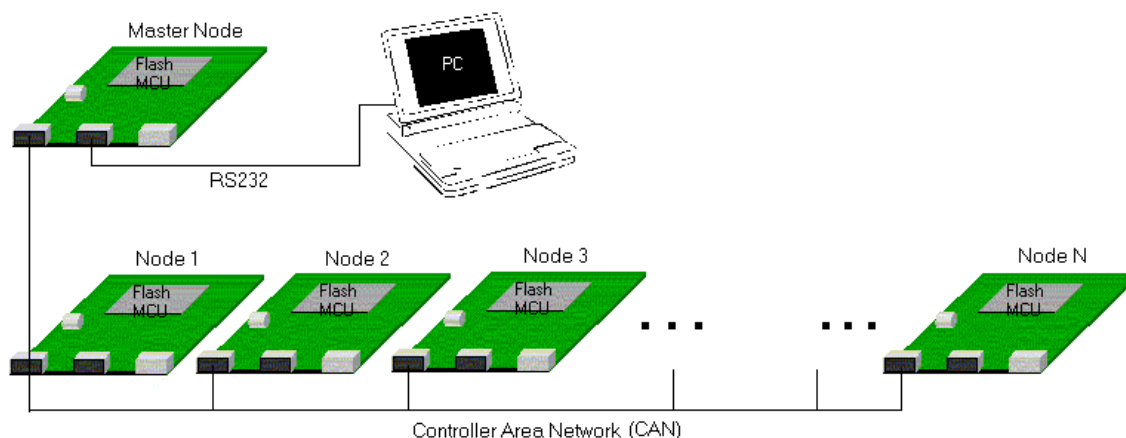


Figure 1. Programming a Flash Microcontroller through the CAN Interface

# Programming Flash Microcontrollers through the CAN Interface

## Configuration

This application uses a master-slave structure. All nodes in the system have their corresponding bootloaders (a master bootloader and a slave bootloader) programmed in advance. By running a simple terminal emulator (starter-kit Wizard), the user can communicate with the master node and thus communicate with all the nodes in the CAN. After power ON, the user has one second to send a dummy character from the PC terminal to the master node. If no character is received by the master node, the default application is called after a timeout; otherwise, the master bootloader enters a flash program monitor mode and a prompt (>) will appear on the terminal.

## Master Programming Mode

Once the master node enters the monitor mode, a simple menu will appear on the PC terminal as shown in Figure 2.

The user has the option of selecting the master node or the slave node for programming. For programming the master node, the user is able to download code from a PC terminal and then program the master node's flash memory. At the same time, all the slave nodes will call, respectively, their default applications because there is no request for them to be programmed.

## Slave Programming Mode

The Master Flash monitor acquires a slave node ID selected by the user, and then broadcasts it on the CAN bus. All slave nodes listen to the initial control information including to the selected slave node ID, and then compare it with their own node ID. Thus, the selected slave node will establish the flash programming connection with the master node and PC. All other slave nodes call their default application individually. In this case, the master node acts as a bridge between the PC terminal via UART and slave nodes via CAN.

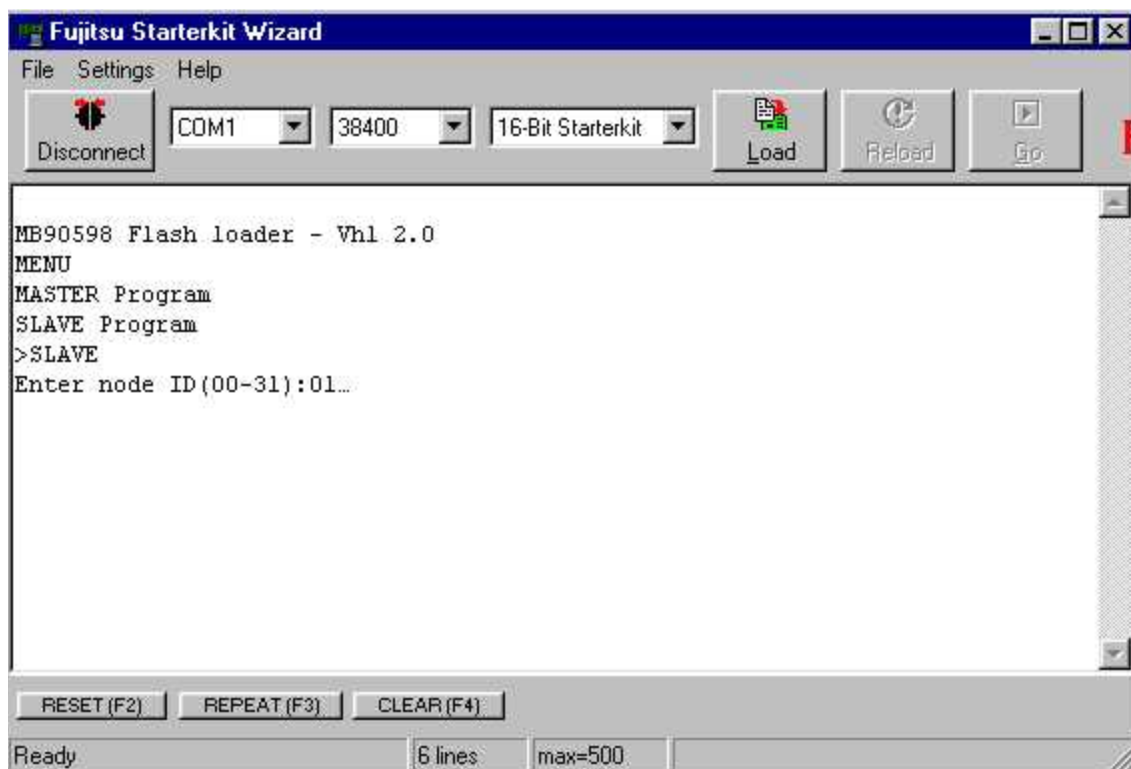


Figure 2. Programming Menu

## Bootloader

The bootloader for the master node has the ability to download the program code from the PC through UART, identify the program record type and format, communicate individually with all the slave nodes via the CAN interface, and also program its own flash memory. The flowchart of the master bootloader is shown in Figure 3.

The bootloader for slave nodes can communicate and receive program code through the CAN interface, identify which node is being programmed, and program its own flash memory. The flowchart of the slave bootloader is shown in Figure 4.

## CAN Implementation (2)

According to the CAN Specification 2.0, Part A and B (3), an 11-bit standard CAN message identifier is used in this application. The format of this ID is shown in Figure 5.

10	9	8	7	6	5	4	3	2	1	0
Type ID				X	Node ID					

**Figure 5. 11-bit Standard CAN Message Identifier**

Each message identifier contains two parts: Type ID (4-bit) and Node ID (5-bit). The four types of identifiers used in this application are listed below in Table 1.

**Table 1.**

Type ID	Value	Memo
Control ID	0x0040	Contains the information about flash program mode and record length
Program ID	0x0020	Contains the program code record in CAN message
Acknowledge ID	0x0030	Usually sent by slave nodes; contains information about program errors
Initialization ID	0x0010	Initial message sent by master node; contains selected node ID

The node ID is presented by 5 least significant bits of the 11-bit message identifier shown in Figure 5. This allows up to 32 nodes in the CAN loop. Each slave node has its own unique ID. The node ID list can be found in the application program (4). The software flow charts for the master and slave node CAN controller are shown in Figure 6 and Figure 7.

# Programming Flash Microcontrollers through the CAN Interface

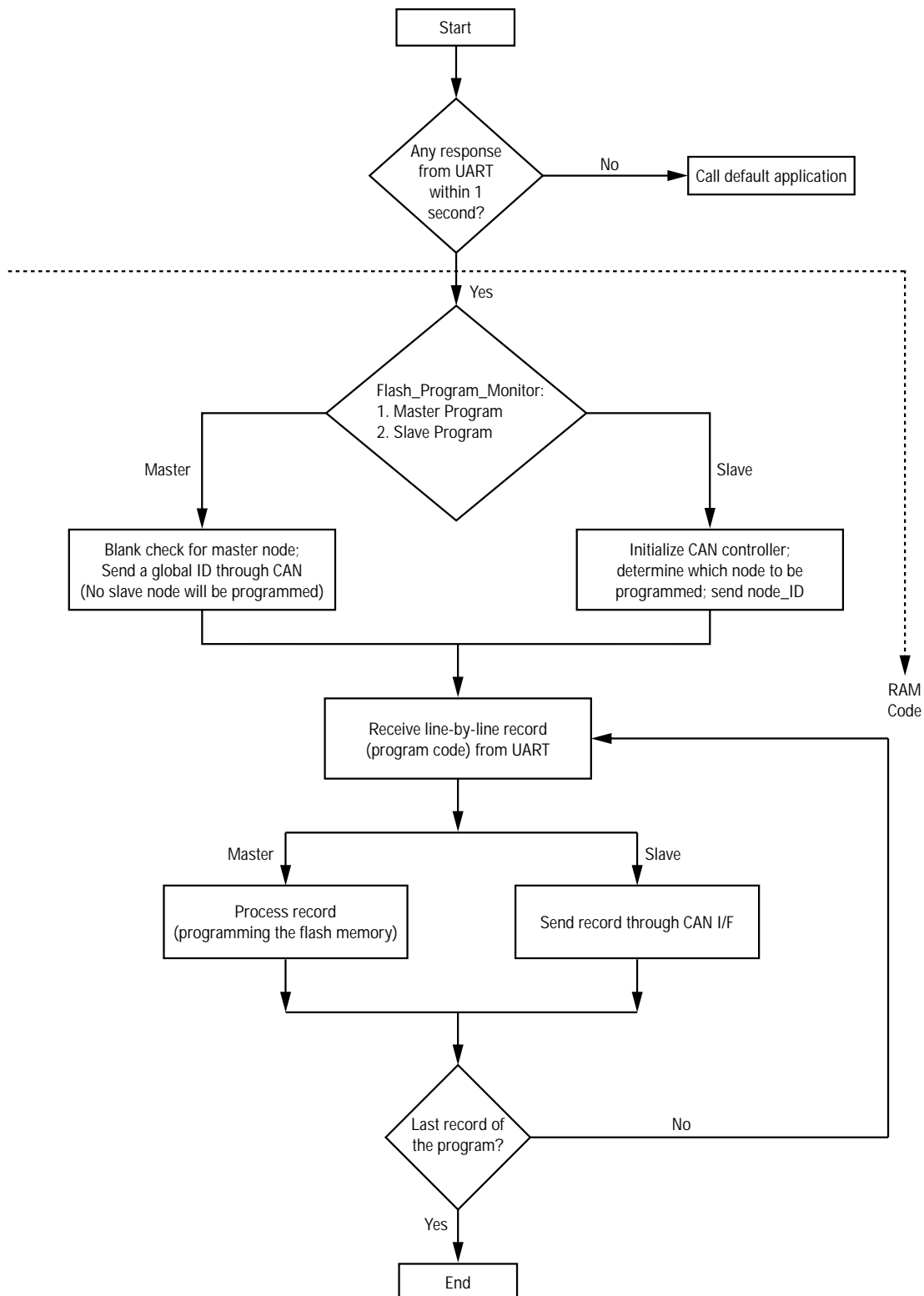
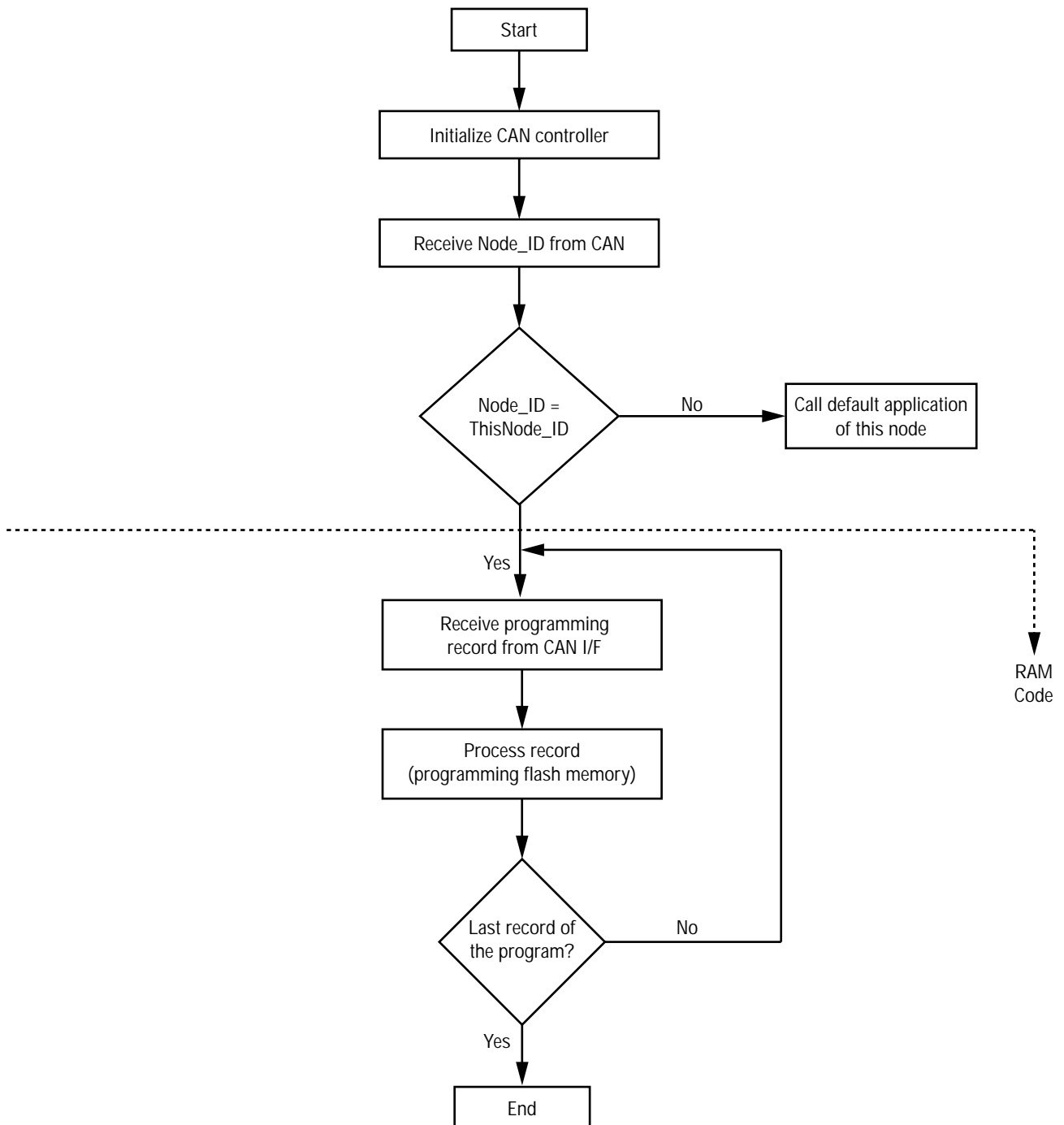
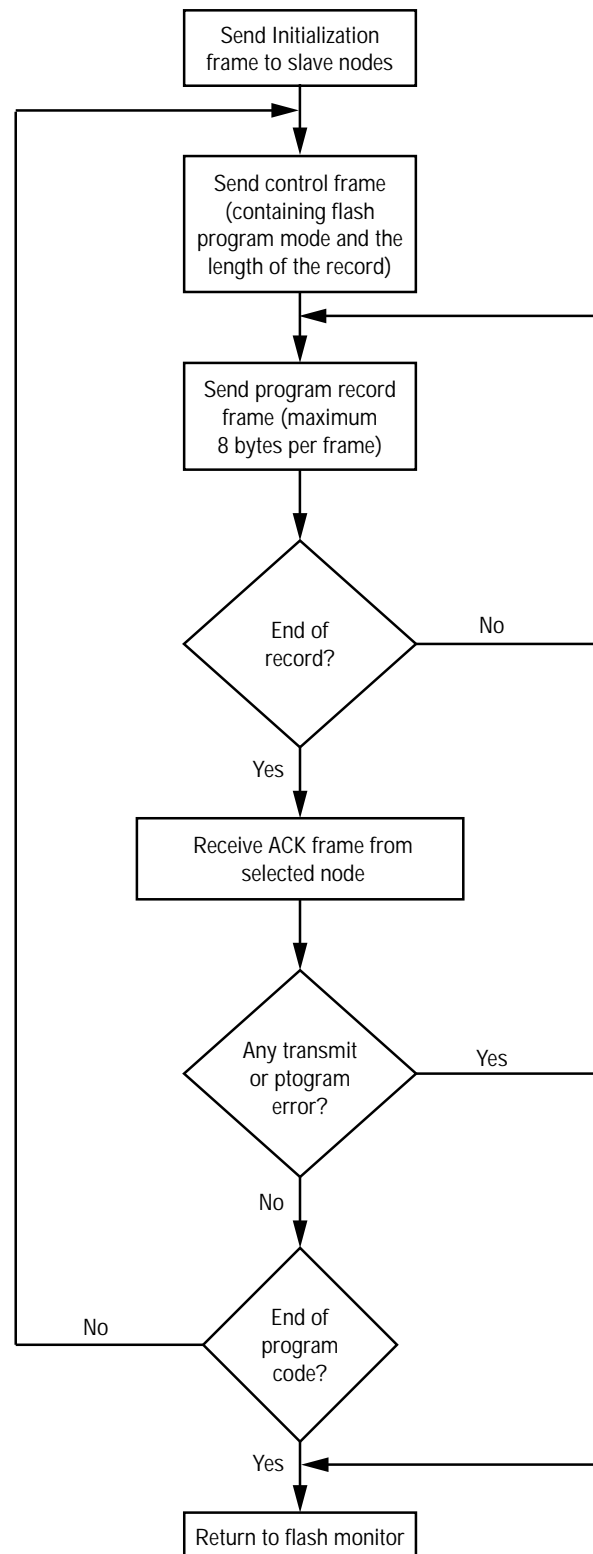


Figure 3. Software Flowchart for the Master Bootloader

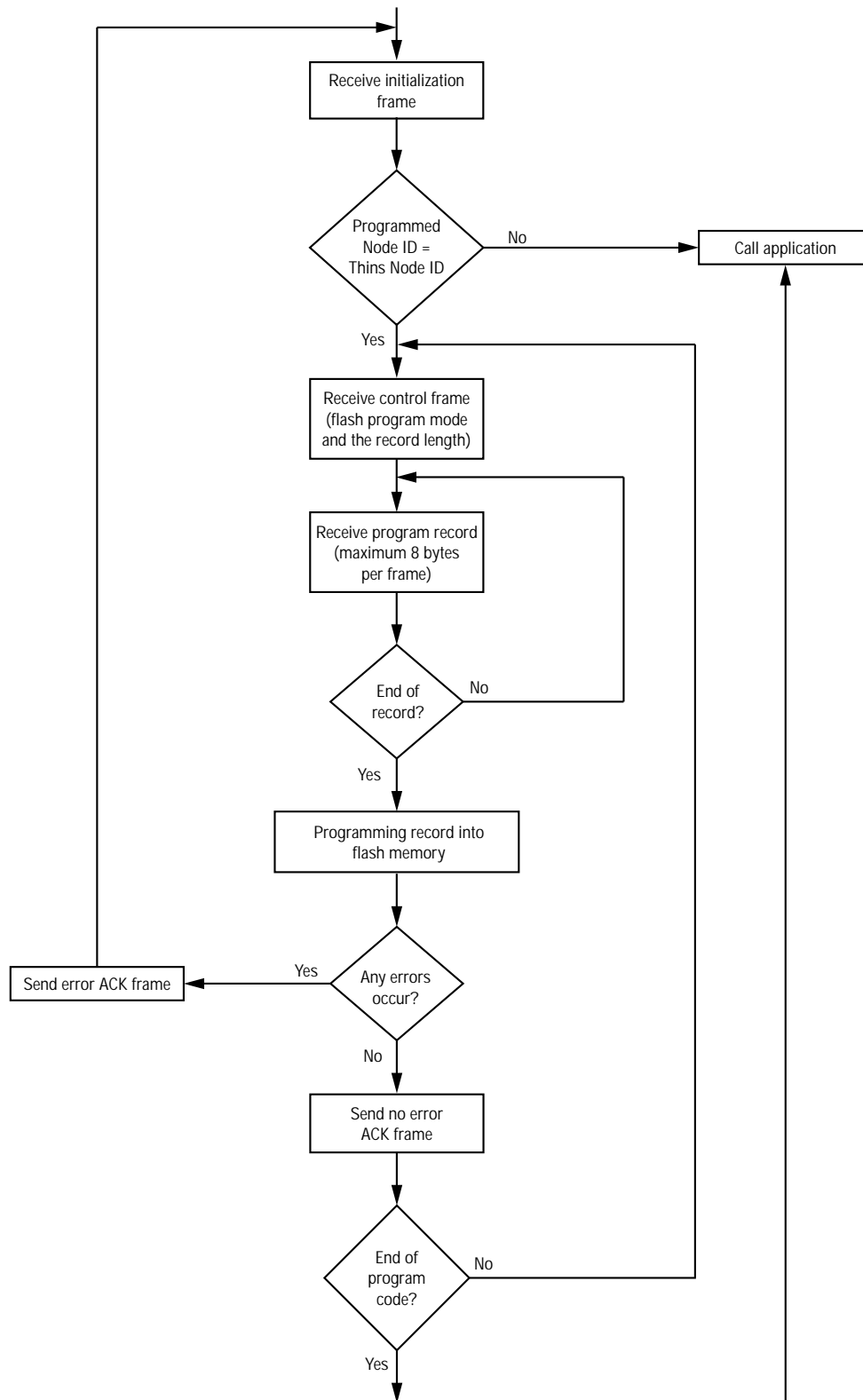


**Figure 4. Software Flowchart for the Slave Bootloader**

# Programming Flash Microcontrollers through the CAN Interface



**Figure 6. Software Flowchart for Master Node CAN Controller**



**Figure 7. Software Flowchart for Slave Node CAN Controller**

# Programming Flash Microcontrollers through the CAN Interface

## Flash Programming Implementation

The actual programming of the Flash Memory is performed by starting the flash memory automatic algorithm. During this phase the flash memory is accessed by the CPU via the internal bus. Once initiated, the automatic algorithm takes care of the setup required by the flow, such as timer settings, modes and registers, and decision programming needed for erasing and/or writing the Flash memory. The various sequences of write accesses are executed in one to six cycles and are called Flash commands. For detailed information about automatic algorithms, refer to Chapter 4.6, entitled Automatic Write/Erase Algorithm, of the MB90595 Hardware Manual (5).

The flash programming method depends on the global variable “w\_e\_mode”(write/erase mode) sent by the master node, and the automatic algorithm processes the data word as follows:

- w\_e\_mode = 1: Checks if the sector to which address “adr” belongs to has already been erased once. If it hasn't, the function erases the sector. In any case, it writes the data to the flash ROM.
- w\_e\_mode = 2: Compares “data” to the corresponding flash ROM word. If any “data” bit is 1 while the flash ROM bit is 0, the sector which “adr” belongs to is erased. No writing is done.
- w\_e\_mode = 3: Writes “data” to the flash ROM without any preceding checks or erases. Flash ROM bits will only be programmed from 1 to 0.  
Note: The address must be even.

By polling the bits of the specific register (Address 0xFFFFFE for MB90595 series), the user can detect the execution state of the automatic algorithm.

## Running the Sample Code and Demo:

The demo is presented on Fujitsu Flash-CAN-100P evaluation board with MB90F598 microcontroller. The following items are required for this demo.

- FLASH-CAN-100P boards with MB90F598 microcontroller. (at least two boards)
- Serial cable
- CAN connection cable
- Power Supply Connectors (7-14 V external DC Power Supply) for FLASH-CAN-100P boards.
- PC or laptop running SOFTUNE V3.0

- Master and Slave Bootloader sample codes, Application sample code. Master and slave bootloader are available for download from <http://www.fma.fujitsu.com/mcu/emb.asp> under application notes.
- FLASH download utility. (FLASH510.EXE)
- Fujitsu SKWIZARD. (PC terminal emulator)

For configuration and connection, please refer to the hardware manual of Flash-CAN100P Board and Figure 1 respectively. The procedure for running the sample code are listed below:

1. Set the jumper JP1 to provide VCC to each board of Flash-CAN-100P
2. Switch ON the Power supply to the boards.
3. Set all the switches of S3 on each board as follows:  
1 - ON, 2 - OFF, 3 - OFF, 4 - OFF, 5 - ON, 6 - OFF, 7 - ON, 8 - ON  
Note: ON stands for 0, OFF stands for 1
4. Select UART1, Pin 21 and Pin 24 by setting the jumpers JP10 and JP9 to connect pin 1 and 2 on each Board.
5. Run Flash510.exe (flash program utility) to download the bootloader program into each node. (Master Bootloader and Slave bootloader)  
Note: Please set the slave node ID in the sample code individually.
6. Connect the master node with PC using serial cable. Set the UART jumpers JP9 and JP10 to connect Pin 2 and pin 3 on the master node board as UART0, Pin18 and Pin20.
7. Connect the master node to the slave nodes through CAN cable. Set the CAN jumpers JP3 and JP4 to connect pin 2 and pin 3 on the boards to set CAN0 for all the nodes, Pin74 and Pin75.
8. Set JP12 jumper(HST) to connect pin 1 and pin 2 on the master board for the monitor reset.
9. Set all the switches of S3 on the board to OFF position, except switch 3, which should be ON to run the bootloader software on the boards.
10. Run SKWIZARD on the PC as PC terminal emulator.
11. Make the COM PORT Parameter Settings on the SKWIZARD as follows:  
Baud Rate: 38400, COM PORT1 or COM PORT2  
No Parity, 1 stop Bit, 8 data bits
12. Click on the 'CONNECT' of SKWIZARD.
13. Run the already loaded asynchronous boot loader software for all the nodes. The MASTER Node communicates with PC monitor and the screen appears as shown in Figure 2. Menu

appears on the screen to choose MASTER or SLAVE nodes.

14. If the user chooses MASTER by typing MASTER on the screen, then user will be able to download any application software (hex file) from PC terminal to Master node's FLASH memory.
15. If the user chooses the SLAVE by typing SLAVE on the screen, further the slave node ID should be entered, and thus the selected SLAVE node will establish Flash programming connection with master node and PC. Download any application program (hex file) to the SLAVE node's Flash memory.

Note: The node application program that the user wants to download should be linked to the proper address of microcontroller, therefore there is no conflict between node application program and node bootloader.

## Conclusion

This application demonstrates the ability to program the Fujitsu Flash Microcontroller via the CAN interface. As the name user *boot-loader* (master and slave) implies, users can modify it according to their application requirement.

## References

The following documents can be found on the Fujitsu MICROS CDRom, Version 3.1 or later:

1. Application Note on Fujitsu Flash Microcontroller 16LX family
2. Application Note on a CAN Bus Protocol Controller Macro
3. CAN Specification 2.0, Parts A and B
4. Sample CAN Management Code for MB90595
5. MB90F598 Hardware Manual
6. FLASH-CAN2 Board Manual
7. Sample Bootloader Code for MB90595

## FUJITSU MICROELECTRONICS AMERICA, INC.

Corporate Headquarters

1250 East Arques Avenue, Sunnyvale, California 94088-3470

Tel: (800) 866-8608 Fax: (408) 737-5999 E-Mail: [inquiry@fma.fujitsu.com](mailto:inquiry@fma.fujitsu.com) Internet: <http://www.fma.fujitsu.com>

© 2002 Fujitsu Microelectronics America, Inc.  
All rights reserved. All company and product  
names are trademarks or registered  
trademarks of their respective owners.

EC-AN-20866-2/2002