SH7058F Group

0.18-µm Process F-ZTAT Microcomputer On-Board Programming

Contents

1. Features

F-ZTAT*1 microcomputer series include an on-chip flash memory, which enables on-board (within the device) programming and erasing. Since the reprogramming is on-board, it is easy to revise software after product shipment, optimize the parameters of the embedded system as a whole, and carry out maintenance.

A 0.18-µm-process F-ZTAT microcomputer has a dedicated on-chip program for erasing and programming. When programming or erasing are to be carried out, the internal modules for these tasks are downloaded to the on-chip RAM and the prescribed parameters are set to suit the user system; programming or erasing is then automatically carried out by simply making the required subroutine call. Erasing is in one divided-block units, and one block is erased by a single call of the erasing program. Programming is in 128-byte units, and 128 bytes are programmed by a single call of the programming program.

*1: F-ZTAT (Flexible Turn Around Time) is the trademark of Renesas Technology Corp.

2. Differences from the 0.35-µm Process F-ZTAT Series

Figure 2.1 shows the procedure for erasing and programming of a 0.35-µm process F-ZTAT microcomputer and figure 2.2 shows that of a 0.18-µm process F-ZTAT microcomputer.

When erasing or programming a 0.35-µm process F-ZTAT system, an erasing program, a programming program, and a procedure program to control the other two must all be created. On the other hand, when a 0.18-µm process F-ZTAT microcomputer is used, only the procedure program need to be created, since the erasing program and the programming program are both on-chip.

When erasing or programming, the procedure program must be loaded to the RAM and executed. Processing by the procedure program includes downloading of the on-chip programs (for initialization, erasing, and programming) to the RAM and the execution of these programs by subroutine calls.

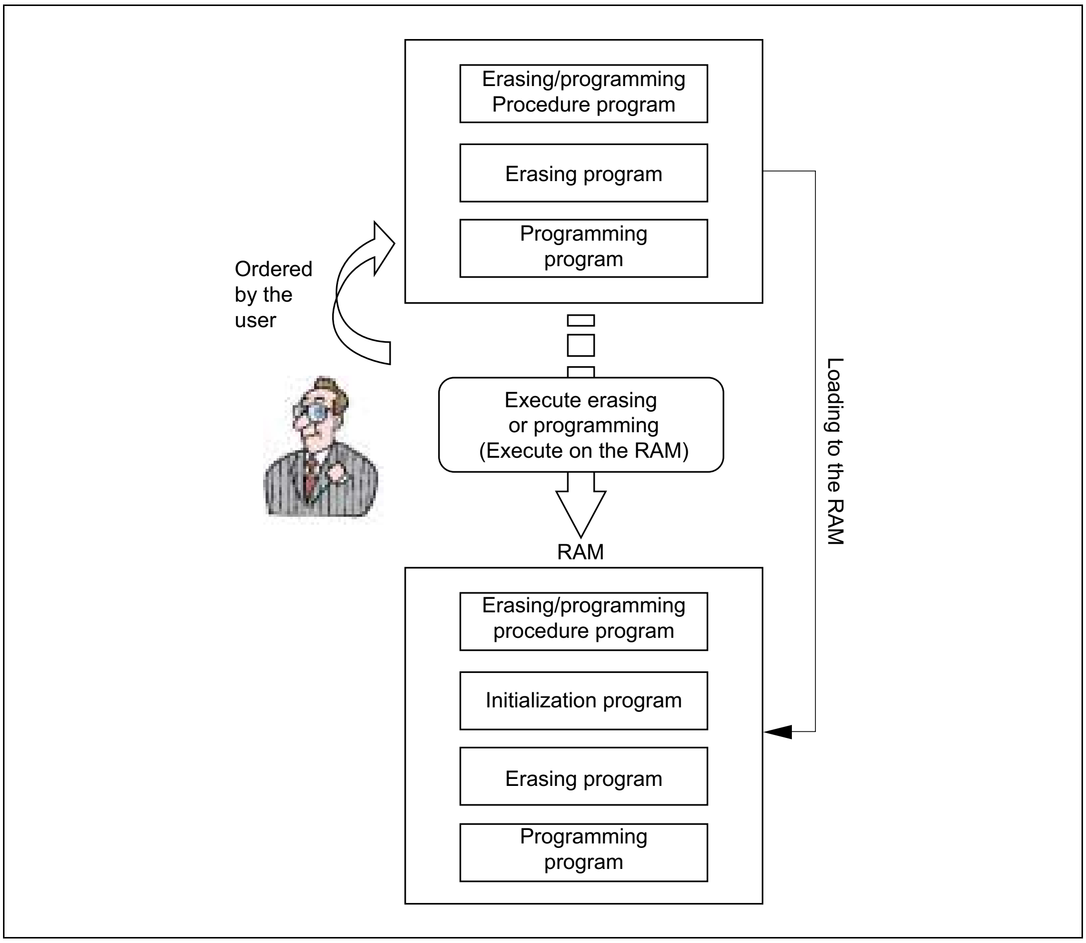

Figure 2.1 Procedure for Erasing and Programming on a 0.35-µm-Process F-ZTAT Microcomputer

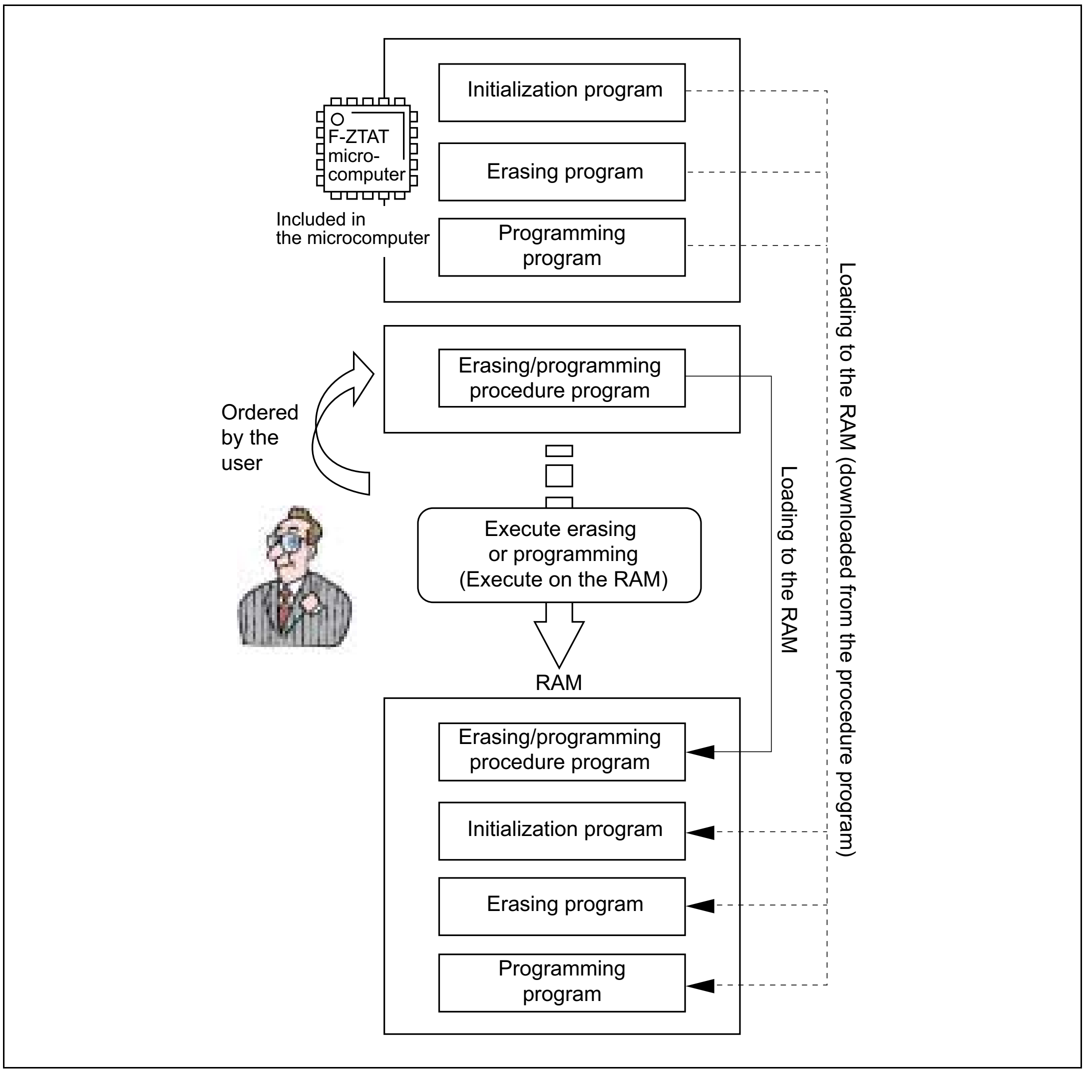
F-ZTAT micro-computer
Included in the microcomputer
Initialization program
Erasing program
Programming program
Ordered by the user
Erasing/programming procedure program
Execute erasing or programming (Execute on the RAM)
RAM
Erasing/programming procedure program
Initialization program
Erasing program
Programming program
Loading to the RAM
Loading to the RAM (downloaded from the procedure program)

Figure 2.2 Procedure for Erasing and Programming on a 0.18-µm-Process F-ZTAT Microcomputer

3. Outline of the Procedure Program

Figure 3.1 shows the sequence followed by the procedure program on a 0.18-µm process F-ZTAT microcomputer.

The sequences of initialization, erasing, and programming in the figure are executed by subroutine calls to the respective downloaded on-chip programs.

When an on-chip program is downloaded, either the erasing or programming program is selected according to the process which is to be executed. Note that it is not possible to carry out both erasing and programming with a single download of an on-chip program.

The process of erasing and programming the flash memory takes place in two rounds, one from the downloading of the erasing program to the end of erasing and the other from the downloading of the programming program to the end of programming.

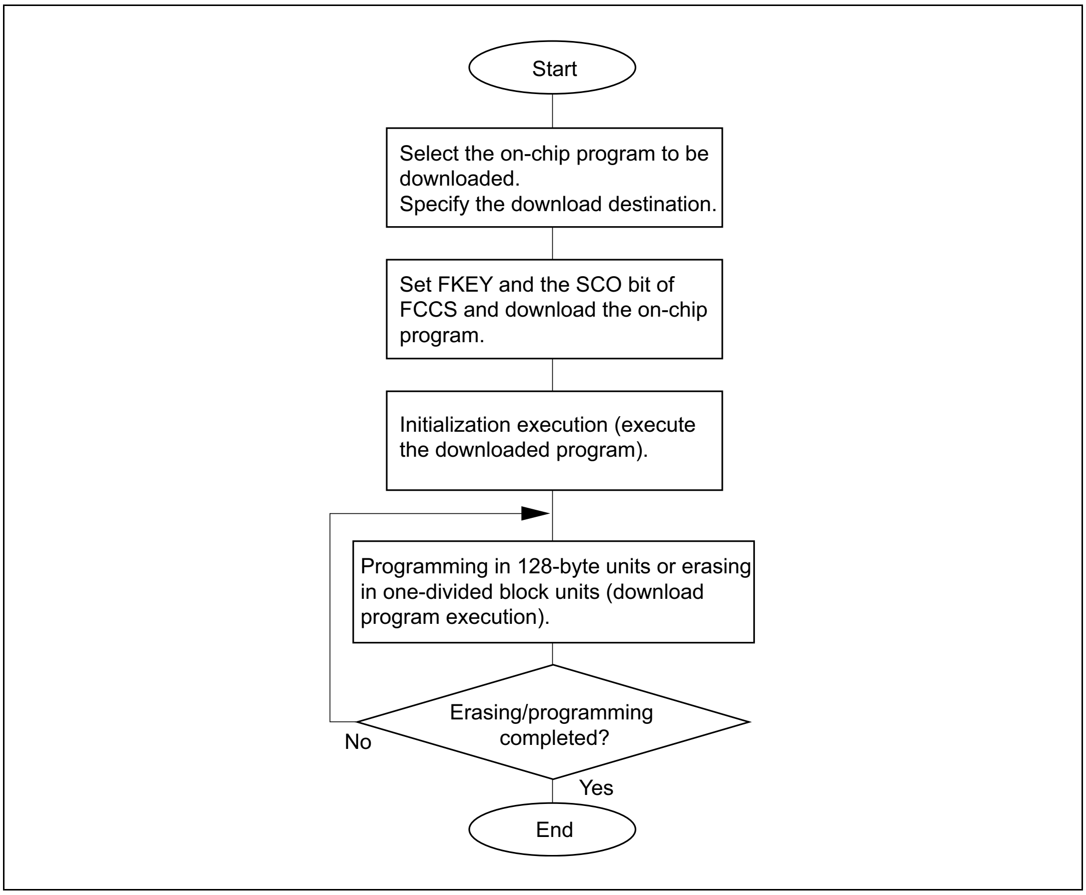
Start
Select the on-chip program to be downloaded.
Specify the download destination.
Set FKEY and the SCO bit of FCCS and download the on-chip program.
Initialization execution (execute the downloaded program).
Programming in 128-byte units or erasing in one-divided block units (download program execution).
Erasing/programming completed?
No
Yes
End

Figure 3.1 Flow of the Erasing/Programming Program

The outline of processing to be executed by the procedure program is described below. In creating the procedure program, these items must be included.

1. Specify the parameters for downloading.

   Select an address in RAM as the destination for downloading (FTDAR).

   Select the program to be downloaded (by setting the PPVS bit of FPCS for programming or the EPVB bit of FECS for erasing).

2. Download the on-chip program to RAM.

   Set H′A5 in the flash key code register (FKEY).

   Set the flash code control and status register (FCCS) and start downloading.

   => After downloading, check for errors by referring to the download pass/fail result parameter DPFR.

3. Initialization

   Make the subroutine call to the downloaded program that executes initialization. The entry address for the initialization program is (download start address set by FTDAR) + 32 bytes.

   Initialization sets these parameters and registers:

   — The CPU operating frequency (FPEFEQ); and

   — The user branch address (FUBRA): This is not supported by some microcomputers.

   => After initialization, check for errors by referring to the flash pass/fail result parameter (FPFR).

4. Erasing or programming

   Erasing or programming is performed by the subroutine call that executes the downloaded erasing or programming program. The entry address for the erasing or programming program is the address set in FTDAR + 16.

   The parameters and registers to be set during erasing and programming are given below.

   This setting is common to both programming and erasing:

   — H′5A in the flash key code register (FKEY).

   These parameters are set for programming:

   — Start address of the region of flash ROM to be programmed (FMPAR); and

   — Start address of the area where the data for programming is stored (FMPDR).

   This parameter is set for erasing:

   — Block-erase number (FEBS).

   => After erasing or writing, check for errors by referring to the flash pass/fail result parameter (FPFR)

4. On-Board Programming Mode (SH7058F)

This section describes the method of erasing and programming a 0.18-µm-process F-ZTAT microcomputer. Here, we take the SH7058F as the example and describe its features and three on-board programming modes.

4.1 Flash Memory on the SH7058F

The main features of the SH7058F are that it incorporates two flash-memory spaces and supports three on-board programming modes.

4.1.1 Two On-Chip Flash-Memory MATs

The flash memory on the 0.18-µm-process F-ZTAT systems is configured in two types of memory spaces, which are called memory MATs. As is shown in figure 4.1, the flash memory consists of the 1-Mbyte user MAT and 8-kbyte user boot MAT.

The application program, i.e., the program which is normally in use, is stored in the user MAT. The user boot MAT is used in user boot mode, which is described later, and the procedure program is stored in this area.

Since the user MAT and user boot MAT areas start at the same address, the MATs must be switched when execution from both MATs or access to data in both MATs is required. The user MAT is programmed in 128-byte units and erased in one divided-block units, as is shown in figure 4.2.

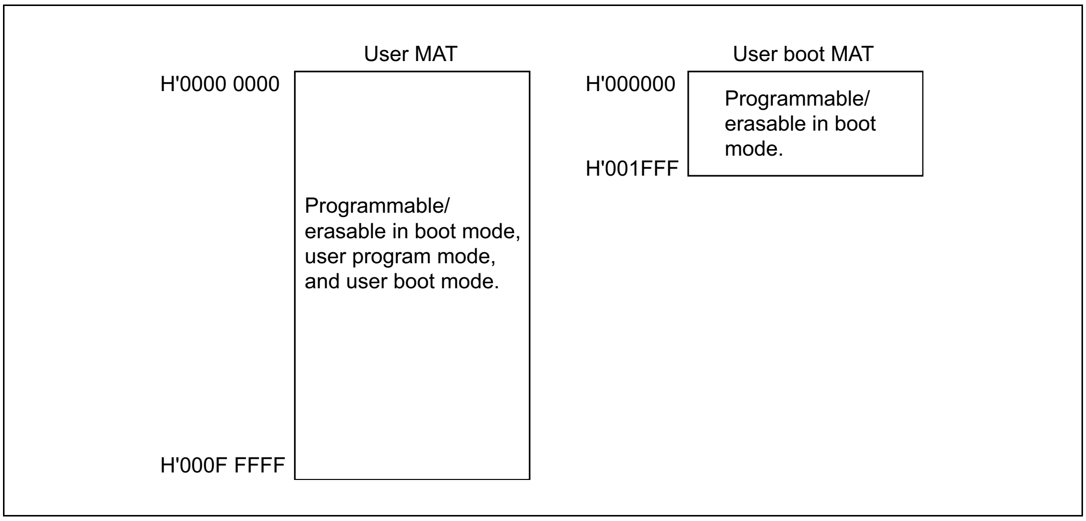
User MAT
H'0000 0000
Programmable/ erasable in boot mode, user program mode, and user boot mode.
H'000F FFFF
User boot MAT
H'000000
Programmable/ erasable in boot mode.
H'001FFF

Figure 4.1 Configuration of the Flash Memory

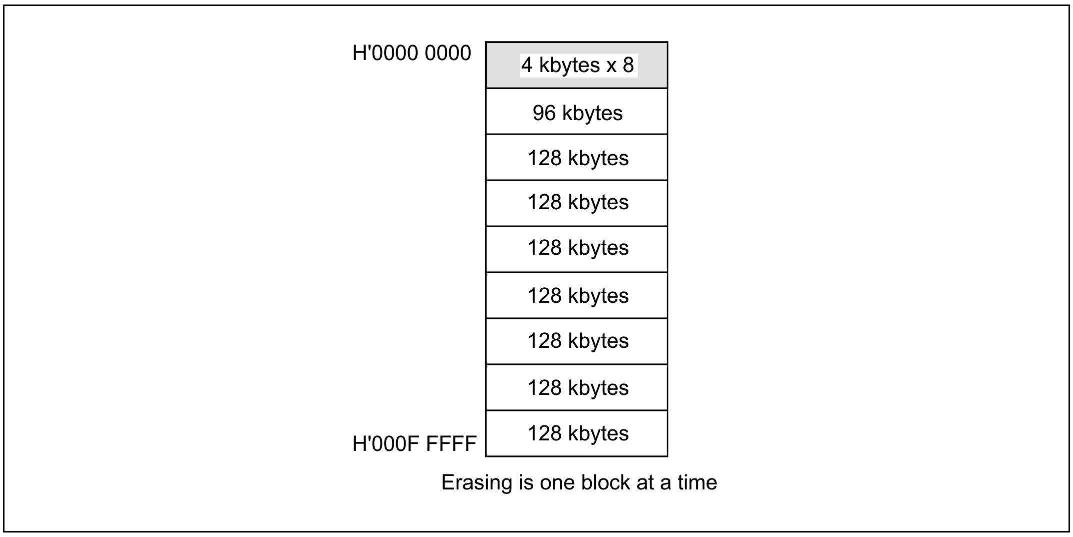

Figure 4.2 Block Structure in Erasing

4.1.2 Three On-Board Programming Modes

The following three on-board programming modes provide support for the erasing and programming of the flash memory. Select the mode by applying the corresponding pin setting (see table 4.1, Mode Selection Pin Settings).

Boot Mode: The on-chip SCI interface is used. Commands and data are transferred between the host and the F-ZTAT microcomputer's boot program which is incorporated in the F-ZTAT microcomputer. Programming and erasing of the user MAT and the user boot MAT may be carried out in this mode.

User Program Mode: This mode allows programming of the user MAT through the desired interface.

User Boot Mode: This mode allows the use of a user created boot program in programming the user MAT through the desired interface. Since this mode starts up with the user boot MAT in the memory map, the MATs must be switched before the user MAT is erased or programmed.

Table 4.1 Mode Selection Pin Settings

Pins	Reset State	Programming Mode			Application Execution State
		User Program Mode	User Boot Mode	Boot Mode	
RES	0	1	1	1	1
FWE	0/1	1	1	1	0
MD0	0/1	1	1	1	0/1
MD1	0/1	0/1	0/1	0/1	0/1
MD2	0/1	1	0	0	0/1
NMI	0/1	0/1	0	1	0/1

4.2 Boot Mode

In boot mode, the user MAT and the user boot MAT are erased or programmed through the receiving of control commands and data for programming from the host and the sending of control commands to the host via the SCI.

The system configuration diagram in boot mode is shown in figure 4.3.

When the boot mode is selected by the corresponding pin settings shown in table 4.1, the boot program starts up when system operation resumes after a reset. The boot program automatically adjusts the SCI bit rate, accepts the control commands that are sent from the host, and replies with responses to the host. If the received command involves a programming or erasing process, that process is executed.

The tool for sending the control commands and the data for programming must be prepared on the host.

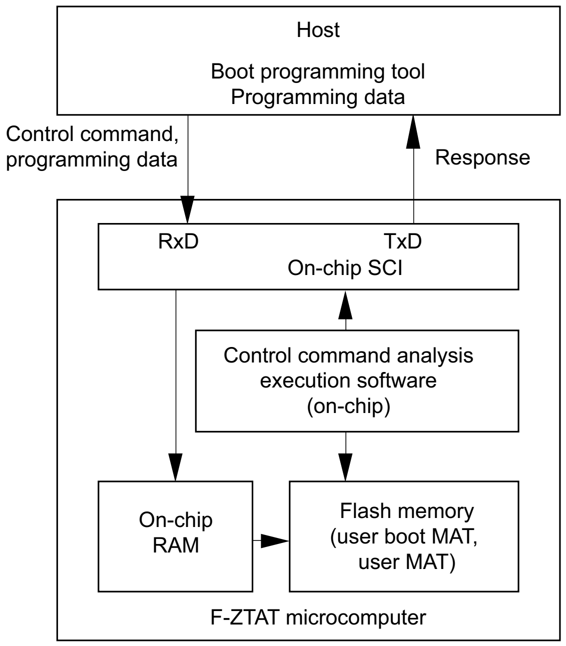
Host
Boot programming tool
Programming data
Control command, programming data
Response
RxD
TxD
On-chip SCI
Control command analysis execution software (on-chip)
On-chip RAM
Flash memory (user boot MAT, user MAT)
F-ZTAT microcomputer

Figure 4.3 System Configuration in Boot Mode

There are three states in the transfer of commands between the host and the boot program; the states become effect in order from one to three.

1. Bit rate-adjustment state
2. Inquiry and selection states
3. Programming/erasing state

For details on the state transitions, see figure 4.4, Overview of State Transitions in Boot Mode.

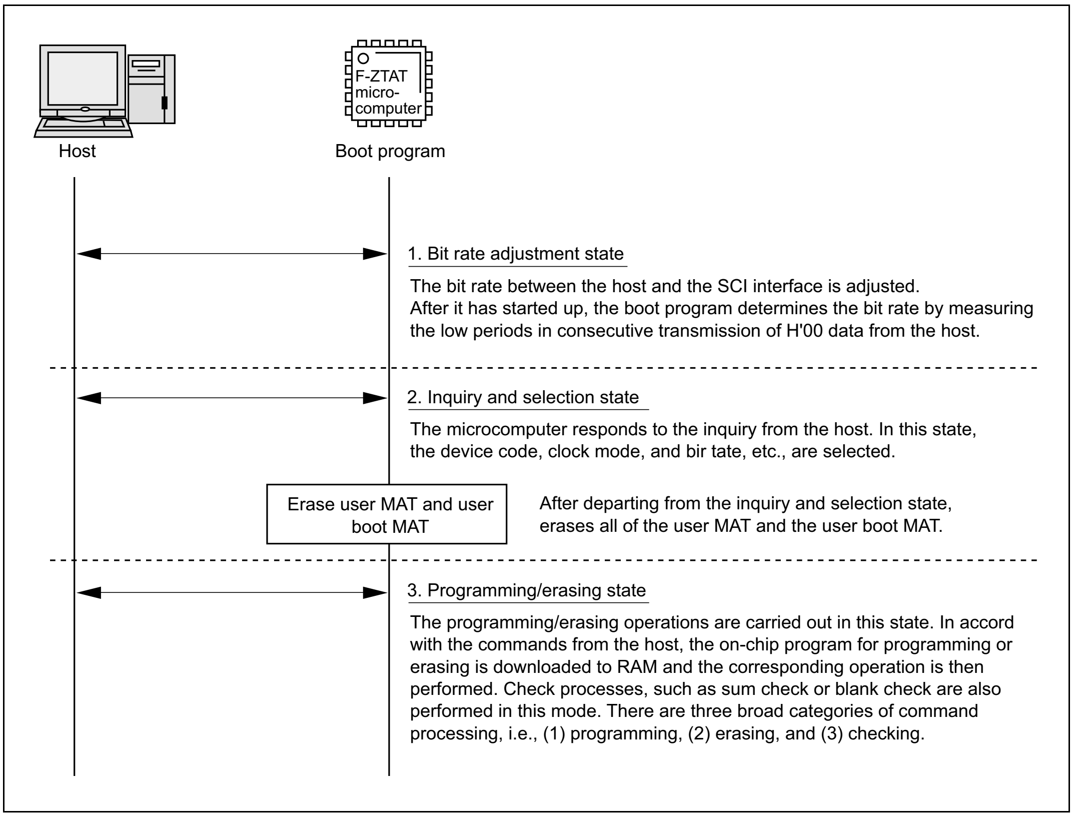
F-ZTAT micro-computer
Host
Boot program
1. Bit rate adjustment state
The bit rate between the host and the SCI interface is adjusted.
After it has started up, the boot program determines the bit rate by measuring the low periods in consecutive transmission of H'00 data from the host.
2. Inquiry and selection state
The microcomputer responds to the inquiry from the host. In this state, the device code, clock mode, and bir tate, etc., are selected.
Erase user MAT and user boot MAT
After departing from the inquiry and selection state, erases all of the user MAT and the user boot MAT.
3. Programming/erasing state
The programming/erasing operations are carried out in this state. In accord with the commands from the host, the on-chip program for programming or erasing is downloaded to RAM and the corresponding operation is then performed. Check processes, such as sum check or blank check are also performed in this mode. There are three broad categories of command processing, i.e., (1) programming, (2) erasing, and (3) checking.

Figure 4.4 Overview of State Transitions in Boot Mode

4.3 User Program Mode

The user program mode allows the programming and erasing of the user MAT. This is done by downloading the programming or erasing program which is included on the microcomputer's chip. The procedure program that requests downloading of the on-chip program, the control of programming or erasing, and judgement of the results is created in the user MAT. The created procedure program is loaded to RAM and executed to carry out programming or erasing.

The flow of processing in user program mode is shown in figures 4.5.

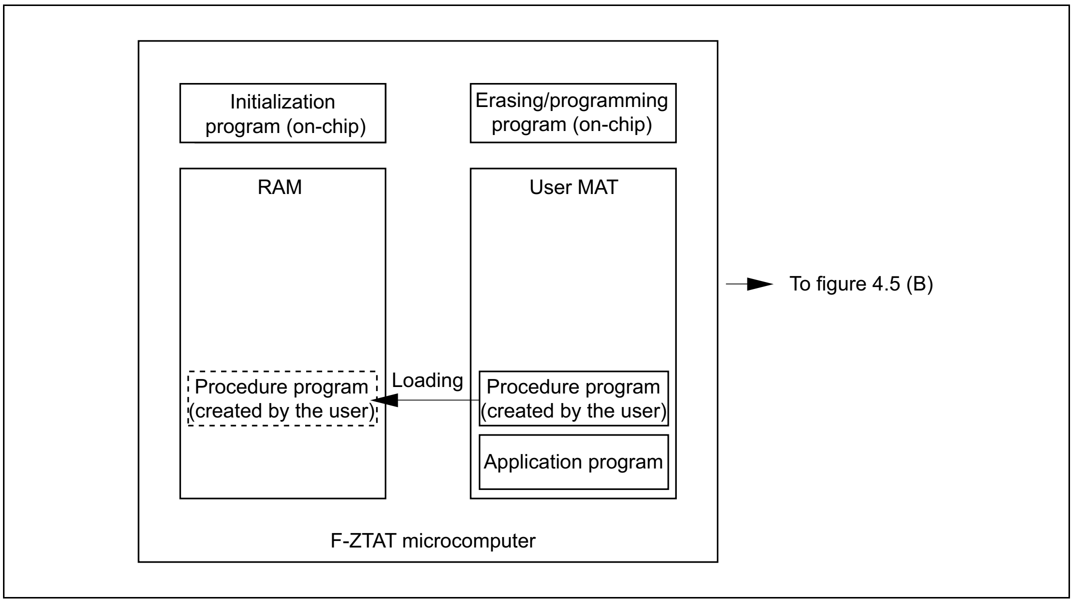
Initialization program (on-chip)
Erasing/programming program (on-chip)
RAM
User MAT
To figure 4.5 (B)
Procedure program (created by the user)
Loading
Procedure program (created by the user)
Application program
F-ZTAT microcomputer

Figure 4.5 (A) Loading the Procedure Program to the RAM

Loading the Procedure Program to the RAM: The procedure program is loaded from the user MAT to the RAM. Control from that point on will be by the procedure program in RAM.

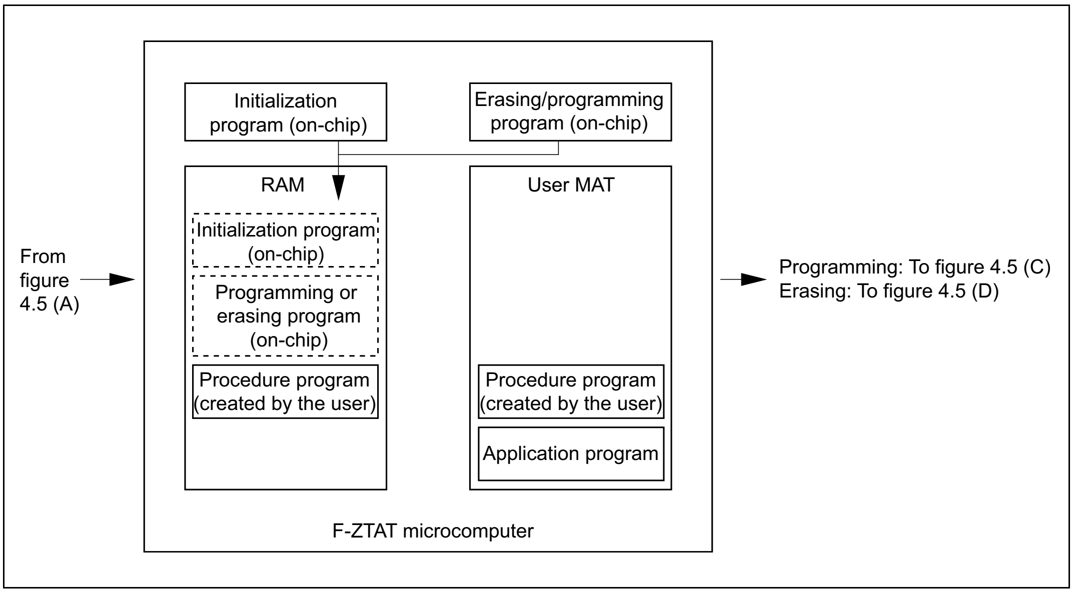
Initialization program (on-chip)
Erasing/programming program (on-chip)
RAM
User MAT
Initialization program (on-chip)
Programming or erasing program (on-chip)
Procedure program (created by the user)
Procedure program (created by the user)
Application program
From figure 4.5 (A)
Programming: To figure 4.5 (C)
Erasing: To figure 4.5 (D)
F-ZTAT microcomputer

Figure 4.5 (B) Downloading the On-Chip Program

Downloading the On-Chip Program: When erasing, the erasing program is downloaded to the RAM and when programming, the programming program is downloaded to the RAM; in either case, the initialization program is also downloaded to the RAM at the same time.

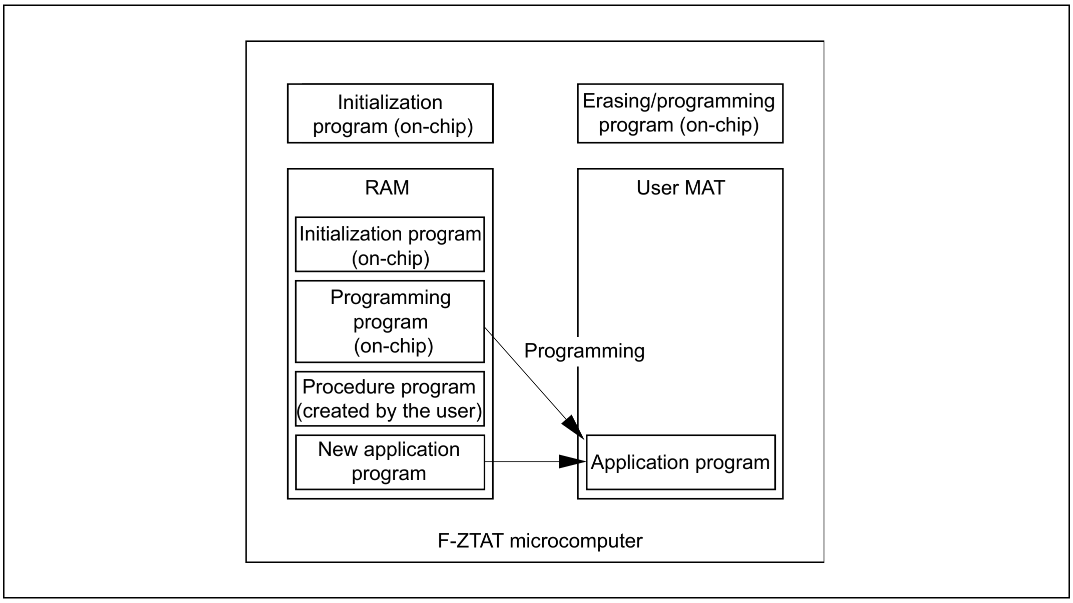
Initialization program (on-chip)
Erasing/programming program (on-chip)
RAM
User MAT
Initialization program (on-chip)
Programming program (on-chip)
Programming
Procedure program (created by the user)
New application program
Application program
F-ZTAT microcomputer

Figure 4.5 (C) Programming the User MAT

Programming the User MAT: A subroutine call makes the initialization program which has been downloaded to the RAM perform initialization which is required for programming. After initialization has been completed, the user MAT is programmed in response to a subroutine call to the programming program which has been downloaded to the RAM. The user MAT is programmed in 128-byte units, so the programming program must be called for every 128-byte unit of data to be programmed until all of the programming data (the application program) is in the MAT.

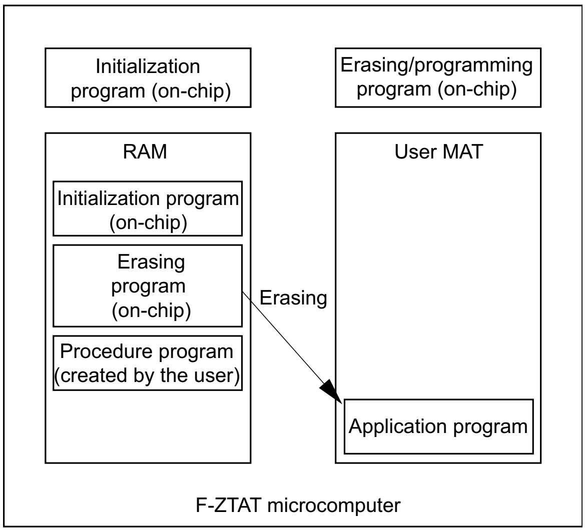
Initialization program (on-chip)
Erasing/programming program (on-chip)
RAM
User MAT
Initialization program (on-chip)
Erasing program (on-chip)
Erasing
Procedure program (created by the user)
Application program
F-ZTAT microcomputer

Figure 4.5 (D) Erasing the User MAT

Erasing the User MAT: A subroutine call makes the initialization program which has been downloaded to the RAM perform initialization which is required for erasing. After initialization has been completed, the user MAT is erased in response to a subroutine call to the erasing program which has been downloaded to the RAM. The user MAT is erased in one divided-block units, so the erasing program must be called for each block which is to be erased.

4.4 User Boot Mode

User boot mode realizes a boot mode for programming via interfaces other than the on-chip SCI. For initiation of user boot mode, refer to table 4.1, Mode Selection Pin Settings. When programming or erasing in user boot mode, the procedure program must be created beforehand in the user boot MAT. On initiation of the user boot mode, the user boot MAT is selected. Accordingly, when the user MAT is to be programmed or erased, the user MAT must be switched to replace the user boot MAT.

The flow of processing in user program mode is shown in figures 4.6.

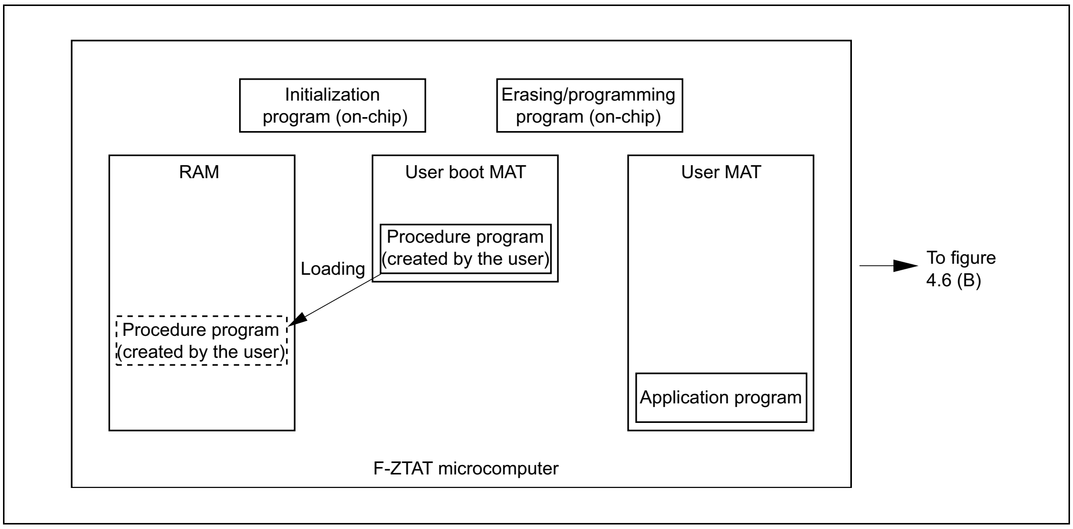
Initialization program (on-chip)
Erasing/programming program (on-chip)
RAM
User boot MAT
User MAT
Procedure program (created by the user)
Loading
Procedure program (created by the user)
Application program
F-ZTAT microcomputer
To figure 4.6 (B)

Figure 4.6 (A) Loading the Procedure Program to the RAM

Loading the Procedure Program to the RAM: The procedure program is loaded to the RAM. Control from that point on will be by the procedure program in RAM.

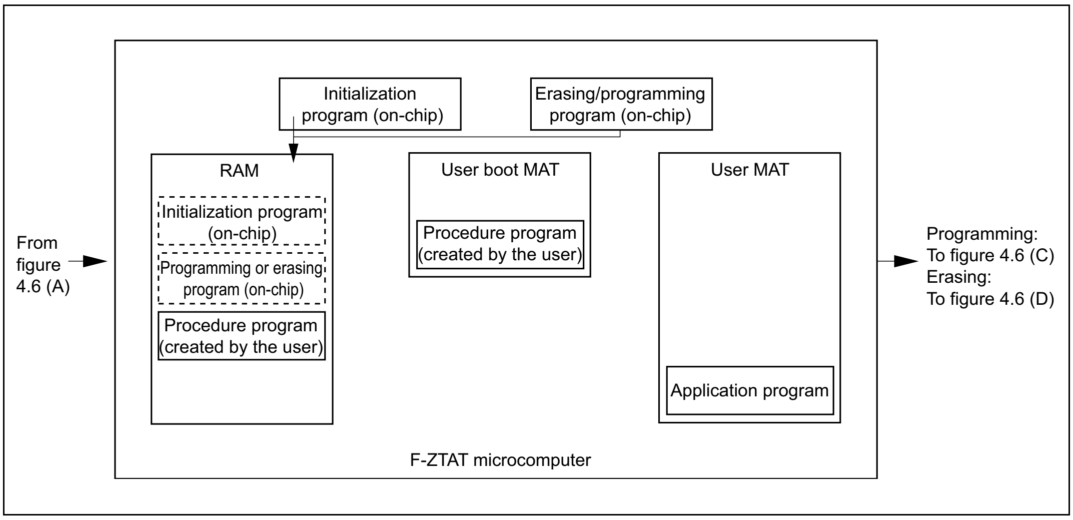
From figure 4.6 (A)
Initialization program (on-chip)
Erasing/programming program (on-chip)
RAM
Initialization program (on-chip)
Programming or erasing program (on-chip)
Procedure program (created by the user)
User boot MAT
Procedure program (created by the user)
User MAT
Application program
F-ZTAT microcomputer
Programming: To figure 4.6 (C)
Erasing: To figure 4.6 (D)

Figure 4.6 (B) Downloading the On-Chip Programs

Downloading the On-Chip Programs: When erasing, the erasing program is downloaded to the RAM, and when programming, the programming program is downloaded to the RAM; in either case, the initialization program is also downloaded to the RAM at the same time.

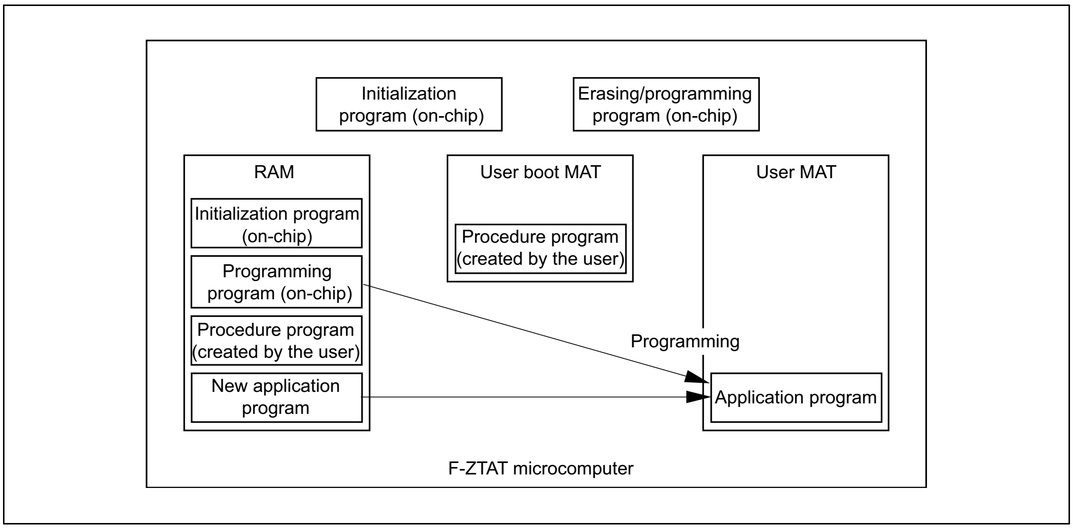
Initialization program (on-chip)
Erasing/programming program (on-chip)
RAM
Initialization program (on-chip)
Programming program (on-chip)
Procedure program (created by the user)
New application program
User boot MAT
Procedure program (created by the user)
User MAT
Programming
Application program
F-ZTAT microcomputer

Figure 4.6 (C) Programming the User MAT

Programming the User MAT: A subroutine call makes the initialization program which has been downloaded to the RAM perform initialization which is required for programming. After initialization has been completed, the user MAT is programmed in response to a subroutine call to the programming program which has been downloaded to the RAM. Programming of the user MAT is in 128-kbyte units, and these units are programmed until all of the programmed data (the application program) has been programmed. Note that when the user MAT is to be programmed, it must be switched to replace the user boot MAT.

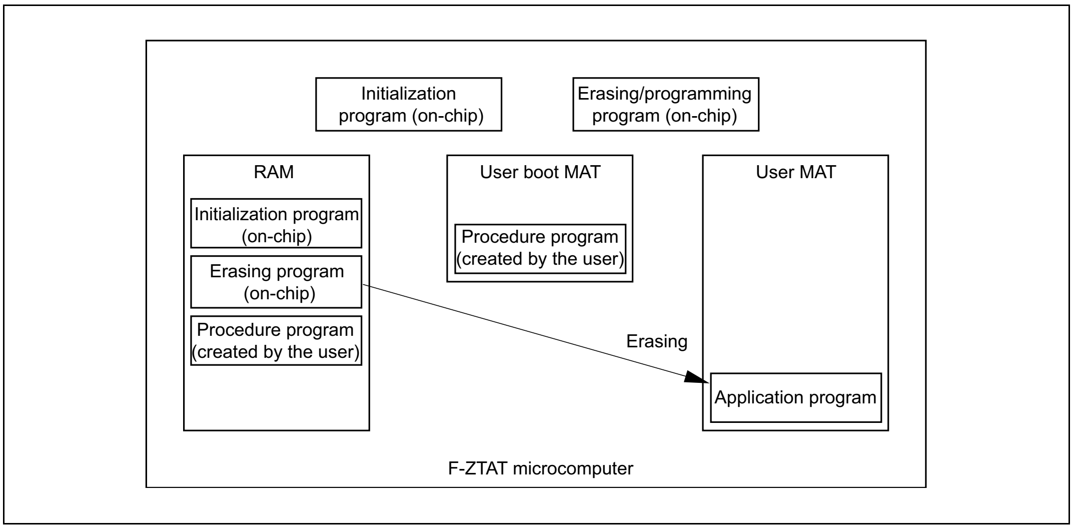
Initialization program (on-chip)
Erasing/programming program (on-chip)
RAM
Initialization program (on-chip)
Erasing program (on-chip)
Procedure program (created by the user)
User boot MAT
Procedure program (created by the user)
User MAT
Erasing
Application program
F-ZTAT microcomputer

Figure 4.6 (D) Erasing the User MAT

Erasing the User MAT: A subroutine call makes the initialization program which has been downloaded to the RAM perform initialization which is required for erasing. After initialization has been completed, the user MAT is erased in response to a subroutine call to the erasing program which has been downloaded to the RAM. The user MAT is erased in one divided-block unit, and these units continue to be erased until all of the blocks have been erased. Note that, if an area on the user MAT is to be erased, this MAT must be switched to replace the user boot MAT.

5. Flash Programming Procedure (SH7058F)

The SH7058F is taken as an example in describing the flash programming procedure for the 0.18-µm-process F-ZTAT microcomputer for each of the three on-board programming modes.

5.1 Programming Procedure in Boot Mode

After a reset and start-up in boot mode, the boot program has the following three states. For an outline of the states, refer to section 4.2, Boot Mode.

The interface between the host and the boot program is described below for each of the states.

5.1.1 Bit Rate Inquiry State

In the bit rate inquiry state, the host continually sends H′00 over the SCI in asynchronous mode. The low period of the sent data is measured by the boot program. The boot program calculates the bit rate at which the host is sending and then returns the end-of-adjustment status byte, H′00, to the host. The host confirms that the H′00 byte has been received normally, and then sends H′55 to the boot program.

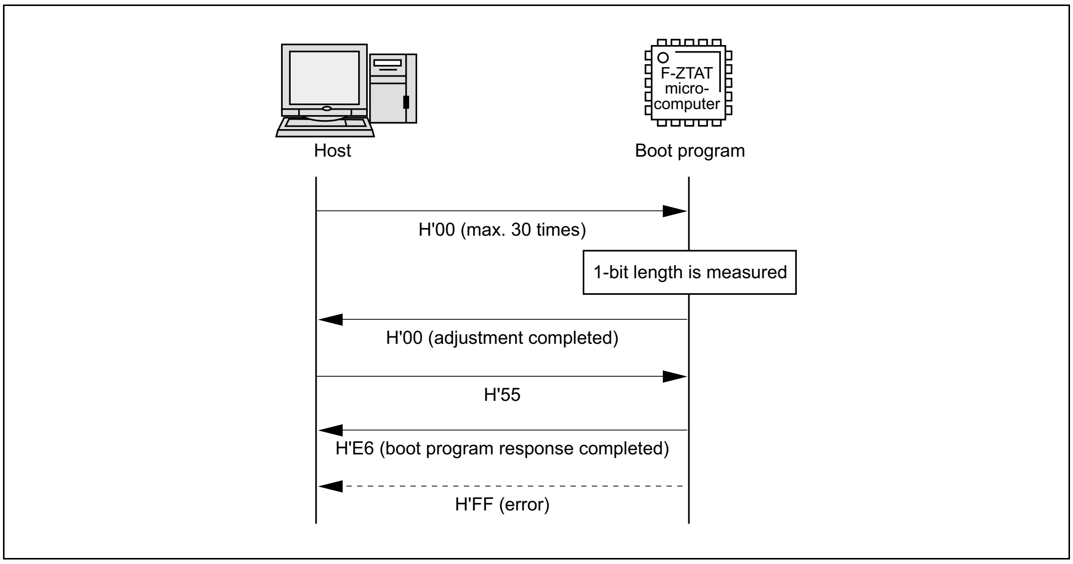

Figure 5.1 Bit Rate Inquiry Sequence

5.1.2 Inquiry and Selection States

In the inquiry and selection states, the boot program responds to inquiry commands from the host with flash ROM information, and responds to selection commands by selecting device code, clock mode selections and bit rate settings.

The host checks the response to the inquiry. For more details on the methods of checking the input frequency, multiplication ratio, operating frequency, and bit rate, refer to the hardware manual.

To make the transition to the programming/erasing state, issue the device selection (H′20), clock mode selection (H′11), and new bit rate selection (H′3F) bytes, in that order, followed by the programming/erasing state transition (H′40) byte, from the host.

After the boot program has automatically erased the data in the user MAT and user boot MAT and returned the ACK (H′06) byte to the host, the transition to the programming/erasing state takes place.

A list of the inquiry and selection commands is given below (table 5.1, Inquiry and Selection Commands). For further details on the respective commands, refer to the hardware manual.

Table 5.1 Inquiry and Selection Commands

Command	Command Name	Description
H'20	Support device inquiry	Inquiry regarding device codes and product names of F-ZTAT
H'10	Device selection	Selection of device code
H'21	Clock mode inquiry	Inquiry regarding numbers of clock modes and values of each mode
H'11	Clock mode selection	Indication of the selected clock mode
H'22	Multiplication ratio inquiry	Inquiry regarding the number of multiplication ratios, and the values of each multiple
H'23	Operating clock frequency inquiry	Inquiry regarding maximum and minimum values of the main clock and peripheral clocks
H'24	User boot MAT information inquiry	Inquiry regarding the number of user boot MATs and start and last addresses of each block
H'25	User MAT information inquiry	Inquiry regarding the number of user MATs and start and last addresses of each block
H'26	Block for erasing information inquiry	Inquiry regarding the number of blocks and start and last addresses of each block
H'27	Programming unit inquiry	Inquiry regarding the unit of programming data
H'3F	New bit rate selection	Selection of new bit rate
H'40	Transition to program/erase state	Erasing of user boot MAT and the user MAT, and entry to programming/erasing state
H'4F	Boot program status inquiry	Inquiry into the operating status of the boot program

5.1.3 Programming/Erasing State

Programming: Firstly, the program selection command is issued from the host; programming and the MAT to be programmed are then selected. The 128-byte programming command (H′40) is then issued to program the data.

Erasing: Firstly, the erase command is issued from the host. The block erase command (H′58) is then issued to erase the specified blocks. See table 5.2, Programming/Erasing Commands, for the programming and erase commands. For further details on the commands, refer to the hardware manual.

Table 5.2 Programming/Erasing Commands

Command	Command Name	Description
H'42	User boot MAT programming selection	Transfers the user boot MAT programming program
H'43	User MAT programming selection	Transfers the user MAT programming program
H'50	128-byte programming	Programs 128 bytes of data
H'48	Erase selection	Transfer the erasing program
H'58	Block erasing	Erases a block of data
H'52	Memory read	Read the contents of memory
H'4A	User boot MAT sum check	Checks the checksum of user boot MAT
H'4B	User MAT sum check	Checks the checksum of user MAT
H'4C	User boot MAT blank check	Checks whether the contents of the user boot MAT are blank
H'4D	User MAT blank check	Checks whether the contents of the user MAT are blank
H'4F	Boot program status inquiry	Inquires into the boot program’s status

5.2 Procedure for Programming in User Program Mode

The program that controls the programming/erasing of the user MAT in user program mode is created by the user. The programming and erasing operations are executed from the RAM.

The following programs must be created by the user:

1. A programming/erasing procedure program for execution from the RAM; and
2. A program to execute the above program on the RAM.

The programming program for programming in 128-byte units and the erasing program for erasing in one divided-block units are included on the microcomputer chip. The user must create the procedure program that downloads the programs to RAM and calls the on-chip programs. The initialization program is also downloaded when the on-chip program is downloaded.

The area to be programmed must be erased in advance when programming flash memory.

Figure 5.2 gives an outline of programming and erasing of the user MAT of the flash memory.

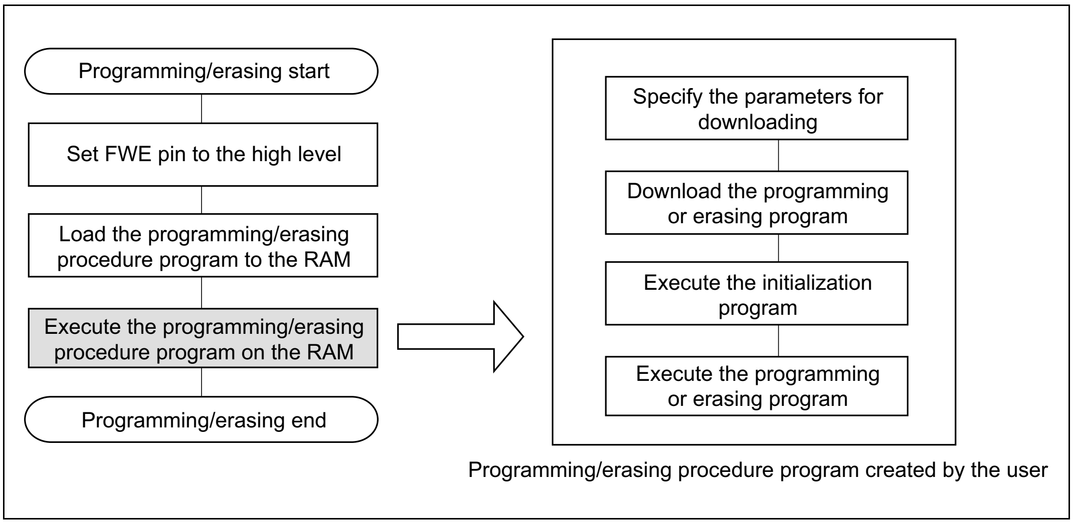
Programming/erasing start
Set FWE pin to the high level
Load the programming/erasing procedure program to the RAM
Execute the programming/erasing procedure program on the RAM
Programming/erasing end
Specify the parameters for downloading
Download the programming or erasing program
Execute the initialization program
Execute the programming or erasing program
Programming/erasing procedure program created by the user

Figure 5.2 Programming/Erasing Processes in User Program Mode

The registers to be set by the programming/erasing procedure program are described in sections 5.2.1 to 5.2.4. The order is the same as that of the four processes in the frame at right in figure 5.2, i.e., (1) specifying the parameters for downloading, (2) downloading, (3) initialization, and (4) executing the erasing or programming program.

5.2.1 Specifying the Parameters for Downloading

The on-chip program must be downloaded to the RAM before programming or erasing. The state of the RAM after downloading is shown in figure 5.3, RAM Map after Downloading of the On-Chip Program. Registers that require settings before downloading are as follows:

1. Specifying the on-chip RAM address for downloading:

   Flash transfer destination address register (FTDAR)

   Before downloading the on-chip program and the related information, an area of on-chip RAM must be specified as the destination for downloading. The start address for downloading is selected from among the specifiable addresses by this register setting.

2. Selecting the program (programming or erasing) for downloading:

   Flash program code select register (FPCS)

   This register is used to select downloading of the programming program. This selection is made by setting the PPVS bit (bit 0) of the register to 1.

   Flash erase code select register (FECS)

   This register is used to select downloading of the erasing program. This selection is made by setting the EPVB bit (bit 0) of this register to 1.

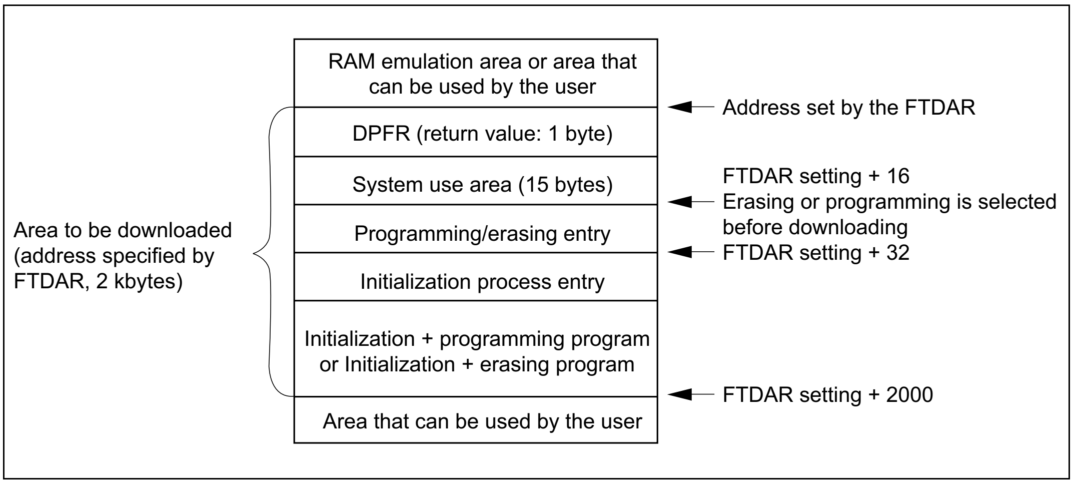
RAM emulation area or area that can be used by the user
DPFR (return value: 1 byte)
System use area (15 bytes)
Programming/erasing entry
Initialization process entry
Initialization + programming program or Initialization + erasing program
Area that can be used by the user
Area to be downloaded (address specified by FTDAR, 2 kbytes)
Address set by the FTDAR
FTDAR setting + 16
Erasing or programming is selected before downloading
FTDAR setting + 32
FTDAR setting + 2000

Figure 5.3 RAM Map after Downloading of the On-Chip Program

5.2.2 Downloading

The following registers must be set before downloading.

Flash key code register (FKEY)

When downloading, H′A5 must be set in this register. If any other value is set here, software protection mode is entered and it is not possible to write to the SCO bit of the FCCS. After downloading, H′00 must be set in this register to select software protection mode.

Flash code control and status register (FCCS)

The specified program is downloaded to the specified area of on-chip RAM by writing 1 to the SCO bit (bit 0) in this register.

5.2.3 Initialization

Initialization is performed before erasing or programming by executing the downloaded initialization program (i.e., calling the subroutine). The entry address for the initialization program is the address stored in FTDAR + 32. A program is given for reference as figure 5.4, Example of Executing the Initialization Program. The parameter settings for initialization are as follows:

Flash programming/erasing frequency control parameter (FPEFEQ: R4):

Setting of CPU operating frequency.

Example: For operation at 40 MHz, 40.00 (any third digit is rounded) × 100 = 4000 is set in R4.

Flash user branch address set parameter (FUBRA: R5):

During erasing or programming, the selected user program is executed with each corresponding predefined processing unit.

Flash pass/fail result parameter (FPFR: R0)

The result of initialization is stored in FPFR (R0). A value in R0 other than H′00 indicates an error.

When consecutive erasing or programming operations are to be carried out, the initialization is only performed once, i.e., before the first erasing or programming.

```
Initialization: The entry address is specified by the FTDAR register.
MOV.L    ENTRY+32,R0      ; Specify the address where the initialization program has been stored
JSR      @R0              ; Execute the initialization program
```

Figure 5.4 Example of Executing the Initialization Program

5.2.4 Erasing or Programming

Erasing/programming is performed by executing the downloaded erasing or programming program (i.e., calling the subroutine). The entry address for the erasing or programming program is stored at the (download start address set by FTDAR) + 16. The reference program for erasing and programming is given in figure 5.5, Example of Executing the Erasing or Programming Program.

The downloaded erasing program erases in one divided-block units, and the programming program programs in 128-byte units. Accordingly, consecutive erasing or programming operations must be performed to erase multiple blocks or to write more than 128 bytes, respectively.

1. The parameters that must be set for erasing or programming are:

   Flash key code register (FKEY):

   When erasing or programming, H′5A must be set in this register. If a value other than H′5A is set here, software protection mode is entered and erasing or programming is not performed. After erasing or programming, H′00 must be set in this register to select software protection mode.

   Flash MAT select register (FMATS):

   This register is used to switch between the user MAT and the user boot MAT. To place the user MAT in the memory map, set H′00 here; to place the user boot MAT in memory map, set H′AA.

2. The parameters that must be set for programming are:

   Flash multipurpose address area parameter (FMPAR: R5):

   Specifies the start address of the destination area in the user MAT.

   Flash multipurpose data destination parameter (FMPDR: R4):

   Specifies the start address of the area where the data to be programmed to the user MAT is stored.

   Flash pass/fail result parameter (FPFR: R0):

   An indicator of the result of programming is stored in FPFR (R0). When R0 contains any value other than H′00, an error has occurred.

3. The parameters that must be set for erasing are:

   Flash erase block select parameter (FEBS: R4):

   Specifies the block number to be erased.

   Flash pass/fail result parameter (FPFR: R0):

   An indicator of the result of erasing is stored in FPFR (R0). When R0 contains any value other than H′00, an error has occurred.

```
Erasing or programming: Entry address is specified by the FTDAR register.
MOV.L    ENTRY+16,R0      ; Specify the address for storing the erasing or programming program
JSR      @R0              ; Execute the erasing or programming program
```

Figure 5.5 Example of Executing the Erasing or Programming Program

The procedure for programming in user program mode is given below. Figure 5.6 is a flowchart for the procedure program for programming in user program mode.

1. Set the FWE signal to the high level.
2. Transfer the procedure program to the on-chip RAM and execute it.
3. Specify the destination address for downloading to RAM in FTDAR.
4. Select an on-chip programming program to be downloaded. Set the PPVS bit in the FPCS register to 1 to select the programming program.
5. Write H′A5 to the FKEY register.
6. Set the SCO bit in the FCCS register to 1. The downloading operation starts. After the SCO bit is written to, the FKEY register is cleared to H′00 for protection.
7. Check the result of downloading by confirming that the SS, FK, and SF bits of the DFPR parameter are all 0. When any of these bits is set to 1, error processing is performed.
8. Set the parameters for initialization; set the CPU clock frequency as the FPEFEQ parameter (R4) and set the user branch start address as the FUBRA parameter (R5).
9. Initialization; the initialization program is downloaded along with the programming program. The entry point for the initialization program is the (download start address set by FTDAR) + 32 bytes. Initialization is performed in response to a subroutine call to this address.
10. Check the result of initialization by confirming that the BR, FQ, and SF bits of the FPFR parameter are all 0. When any of these bits is set to 1, error processing is performed.
11. Disable all interrupts and the operation as bus master of anything except the CPU.
12. Write H′5A to the FKEY register.
13. Set the start address of the destination area for programming of the user MAT as the FMPAR parameter (R5) and set the start address of the area where the data to be written to the user MAT is stored as the FMPDR parameter (R4).
14. Perform the programming operation; the entry point of the programming program is at the (download start address set by FTDAR) + 16 bytes. Perform a subroutine call to this address to program the selected 128 bytes of data.
15. Check the result of programming by confirming that the MD, EE, FK, WD, WA, and SF bits of the FPFR parameter are all 0. If any of these bits is set to 1, error processing is performed.
16. If the all of the data to be written has not yet been written, return to step 13 and repeat the procedure from there.
17. Clear the FKEY register to H′00.

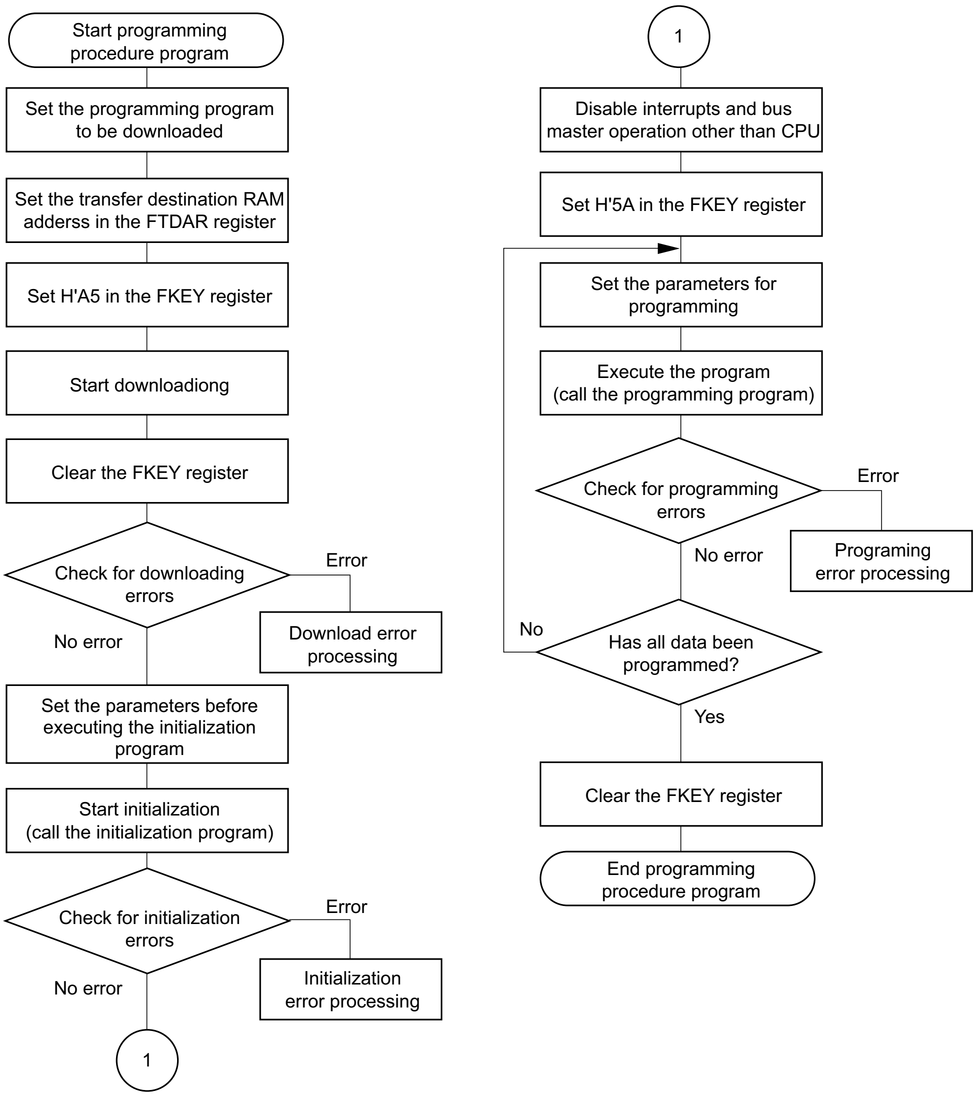
Start programming procedure program
Set the programming program to be downloaded
Set the transfer destination RAM adderss in the FTDAR register
Set H'A5 in the FKEY register
Start downloadiong
Clear the FKEY register
Check for downloading errors
Error
Download error processing
No error
Set the parameters before executing the initialization program
Start initialization (call the initialization program)
Check for initialization errors
Error
Initialization error processing
No error
1
1
Disable interrupts and bus master operation other than CPU
Set H'5A in the FKEY register
Set the parameters for programming
Execute the program (call the programming program)
Check for programming errors
Error
Programing error processing
No error
No
Has all data been programmed?
Yes
Clear the FKEY register
End programming procedure program

Figure 5.6 Flowchart of the Procedure Program for Programming in User Program Mode

The procedure for erasing in user program mode is described below. Figure 5.7 shows the erasing procedure program flowchart in user program mode.

1. Set the FWE signal to the high level.
2. Transfer the programming/erasing procedure program to the on-chip RAM and execute it.
3. Specify the destination address for downloading to RAM to FTDAR.
4. Select an on-chip program to be downloaded. Set the EPVB bit of the FECS register to 1 to select the erasing program.
5. Write H′A5 to the FKEY register.
6. Set the SCO bit in the FCCS register to 1. The downloading operation starts. After the SCO bit is written to, the FKEY register is cleared to H′00 for protection.
7. Check the result of downloading by confirming that the SS, FK, and SF bits of the DFPR parameter are all 0. When any of these bits is set to 1, error processing is required.
8. Set the parameters for initialization; set the CPU clock frequency as the FPEFEQ parameter (R4) and set the user branch start address as the FUBRA parameter (R5).
9. Initialization; the initialization program is downloaded along with the erasing program. The entry point for the initialization program is the (download start address set by FTDAR) + 32 bytes. Initialization is performed in response to a subroutine call to this address.
10. Check the result of initialization by confirming that the BR, FQ, and SF bits of the FPFR parameter are all 0. When any of these bits is set to 1, error processing is performed.
11. Disable all interrupts and the operation as bus master of anything except the CPU.
12. Write H′5A to the FKEY register.
13. Set the block number to be erased in the user MAT as the FEBS parameter (R4).
14. Perform the erase operation; the entry point of the erasing program is at the address given by (download start address set by FTDAR) + 16 bytes. Perform a subroutine call to this address to erase the selected block of data.
15. Check the result of erasing by confirming that the MD, EE, FK, EB, and SF bits of the FPFR parameter are all 0. If any of these bits is set to 1, error processing is performed.
16. If all of the blocks to be erased have not yet been erased, return to step 13 and repeat the procedure from there.
17. Clear the FKEY register to H′00.

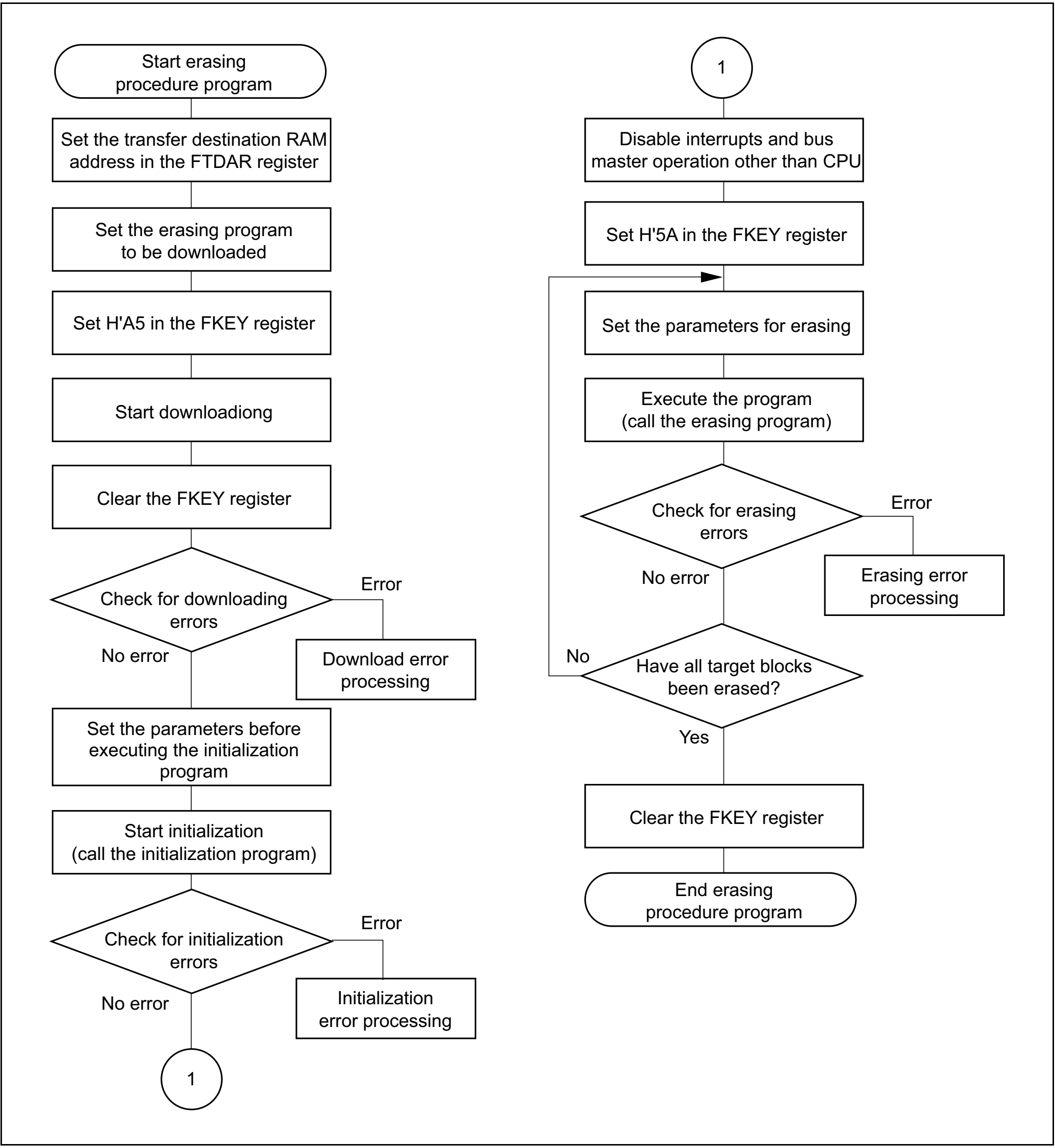
Start erasing procedure program
Set the transfer destination RAM address in the FTDAR register
Set the erasing program to be downloaded
Set H'A5 in the FKEY register
Start downloadiong
Clear the FKEY register
Check for downloading errors
Error
Download error processing
No error
Set the parameters before executing the initialization program
Start initialization (call the initialization program)
Check for initialization errors
Error
Initialization error processing
No error
1
1
Disable interrupts and bus master operation other than CPU
Set H'5A in the FKEY register
Set the parameters for erasing
Execute the program (call the erasing program)
Check for erasing errors
Error
Erasing error processing
No error
No
Have all target blocks been erased?
Yes
Clear the FKEY register
End erasing procedure program

Figure 5.7 Flowchart of the Procedure Program for Erasing in User Program Mode

5.3 Programming Procedure in User Boot Mode

Figure 5.8 shows the flow of erasing/programming processing in user boot mode.

In the same way as for operations in user program mode, a procedure program must be created by the user.

The difference in the programs for the respective modes lies in whether or not the program has to include switching of the MAT. The user boot MAT is selected on start-up in user boot mode after a reset. To program the user MAT, the current MAT must be switched from the user boot MAT to the user MAT. After programming, the current MAT must be switched back from the user MAT to the user boot MAT.

The processing for this switchover must be included in the procedure program created by the user.

The switchover from the user boot MAT to the user MAT takes place before the start of programming or erasing (i.e., before H′5A is set in the FKEY register). The switch from the user MAT back to the user boot MAT takes place after the end of programming or erasing (after clearing of the FKEY register).

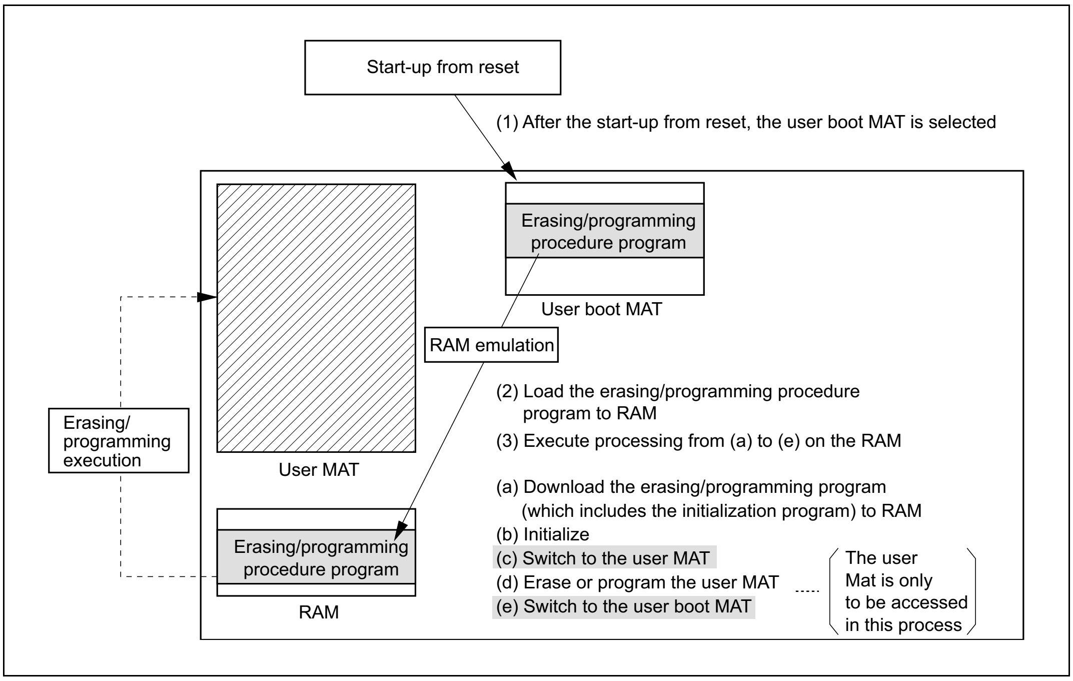
Start-up from reset
(1) After the start-up from reset, the user boot MAT is selected
Erasing/programming procedure program
User boot MAT
RAM emulation
Erasing/ programming execution
User MAT
(2) Load the erasing/programming procedure program to RAM
(3) Execute processing from (a) to (e) on the RAM
(a) Download the erasing/programming program (which includes the initialization program) to RAM
(b) Initialize
(c) Switch to the user MAT
(d) Erase or program the user MAT
(e) Switch to the user boot MAT
The user Mat is only to be accessed in this process
Erasing/programming procedure program
RAM

Figure 5.8 Programming Procedure for Processing in User Boot Mode

Figure 5.9 is the flowchart for the procedure program for programming in user boot mode. As stated above, the difference from user program mode is the switching of the MAT, which is indicated by the gray boxes.

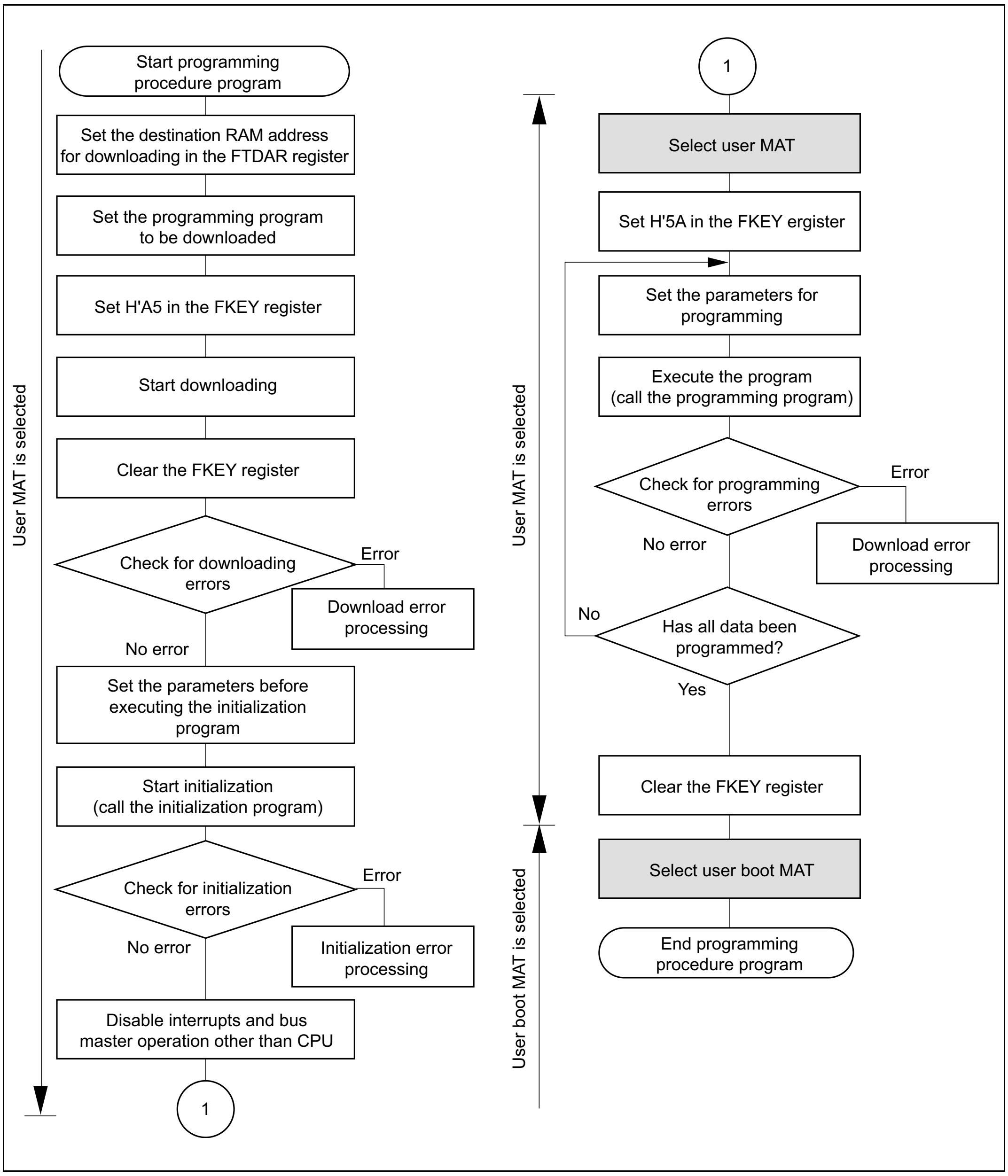
User MAT is selected
Start programming procedure program
Set the destination RAM address for downloading in the FTDAR register
Set the programming program to be downloaded
Set H'A5 in the FKEY register
Start downloading
Clear the FKEY register
Check for downloading errors
Error
Download error processing
No error
Set the parameters before executing the initialization program
Start initialization (call the initialization program)
Check for initialization errors
Error
Initialization error processing
No error
Disable interrupts and bus master operation other than CPU
1
1
User MAT is selected
User boot MAT is selected
Select user MAT
Set H'5A in the FKEY ergister
Set the parameters for programming
Execute the program (call the programming program)
Check for programming errors
Error
Download error processing
No error
Has all data been programmed?
No
Yes
Clear the FKEY register
Select user boot MAT
End programming procedure program

Figure 5.9 Flowchart of the Procedure Program for Programming in User Boot Mode

Figure 5.10 is the flowchart for the procedure program for erasing in user boot mode. The difference from user program mode is again the switching of the MAT, which is indicated by the gray boxes.

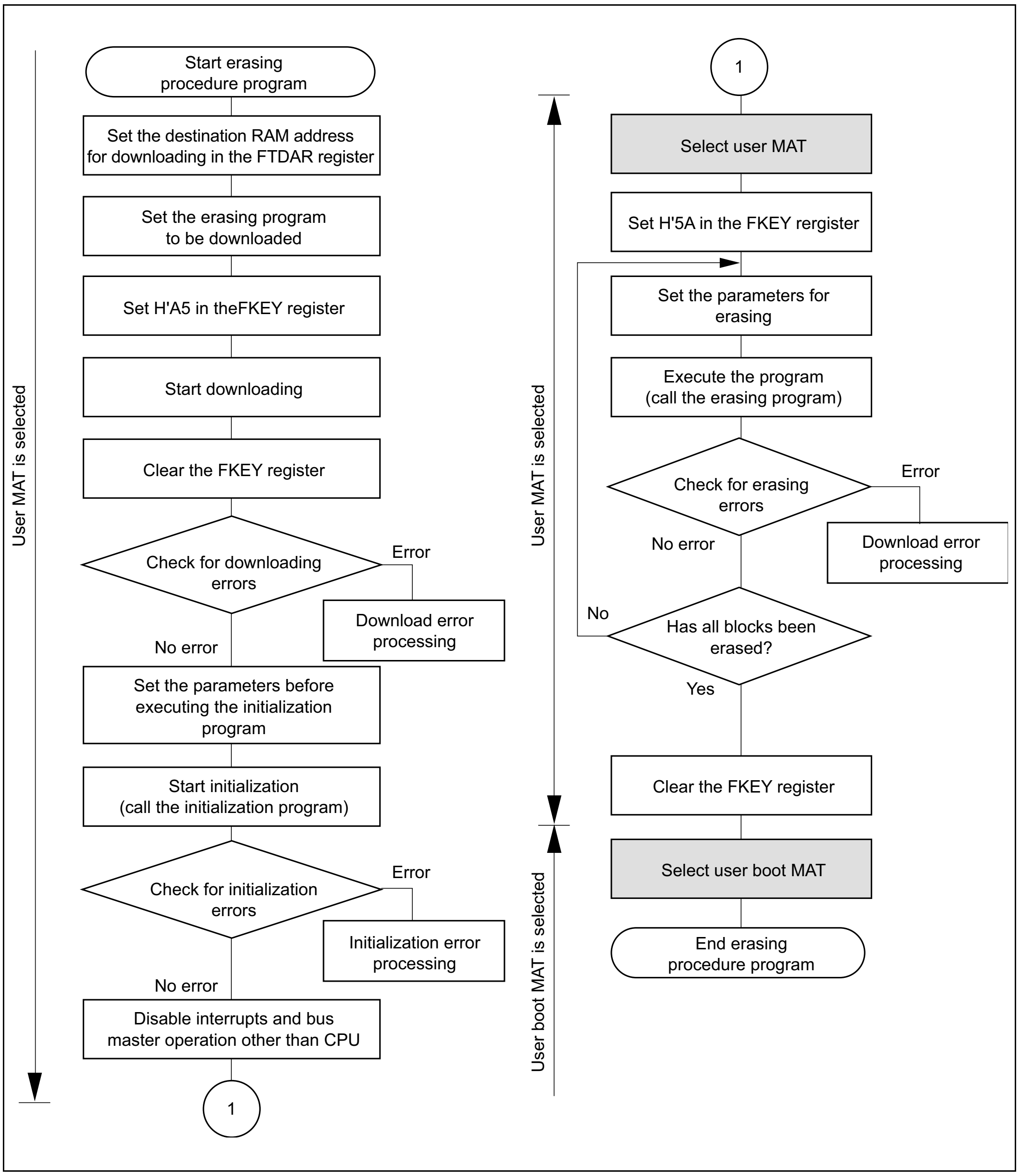
User MAT is selected
Start erasing procedure program
Set the destination RAM address for downloading in the FTDAR register
Set the erasing program to be downloaded
Set H'A5 in theFKEY register
Start downloading
Clear the FKEY register
Check for downloading errors
Error
Download error processing
No error
Set the parameters before executing the initialization program
Start initialization (call the initialization program)
Check for initialization errors
Error
Initialization error processing
No error
Disable interrupts and bus master operation other than CPU
1
User MAT is selected
User boot MAT is selected
1
Select user MAT
Set H'5A in the FKEY rergister
Set the parameters for erasing
Execute the program (call the erasing program)
Check for erasing errors
Error
Download error processing
No error
No
Has all blocks been erased?
Yes
Clear the FKEY register
Select user boot MAT
End erasing procedure program

Figure 5.10 Flowchart of the Procedure Program for Erasing in User Boot Mode

6. Example of User Created Program

6.1 Example of Procedure Program for Erasing/Programming Flash in User Program Mode

6.1.1 Overview

In this sample program, the flash reprogramming command is received by means of the external advanced user debugger (AUD) while the user application program is executed in user program mode, so that the procedure program is executed.

Therefore to execute the procedure program, reading the command area periodically and checking whether or not the command in the area is flash reprogramming command are should be performed by the user application program. When the command is the flash reprogramming command, the procedure program is executed after all interrupts are disabled and the procedure program is loaded to the on-chip RAM.

In this sample application program, not only flash reprogramming command but erased block information, programming command, and transferring programming data are controlled by external AUD.

Figure 6.1 shows the flow of procedure in this sample program.

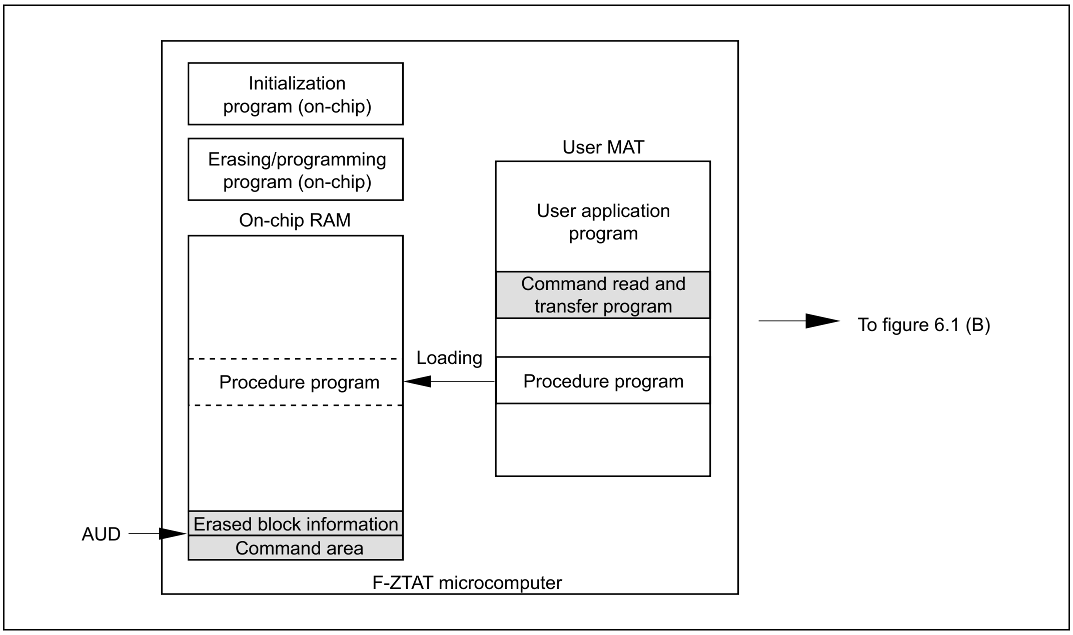

Figure 6.1 (A) Loading the Procedure Program to the RAM

Loading the Procedure Program to the RAM: Information on the block to be cleared and reprogramming command are written by the external AUD. After checking whether or not the programming command is written in the command area by the user application program and disabling all interrupts, the procedure program is loaded from the user MAT to the RAM. Control from that point on will be by the procedure program in RAM.

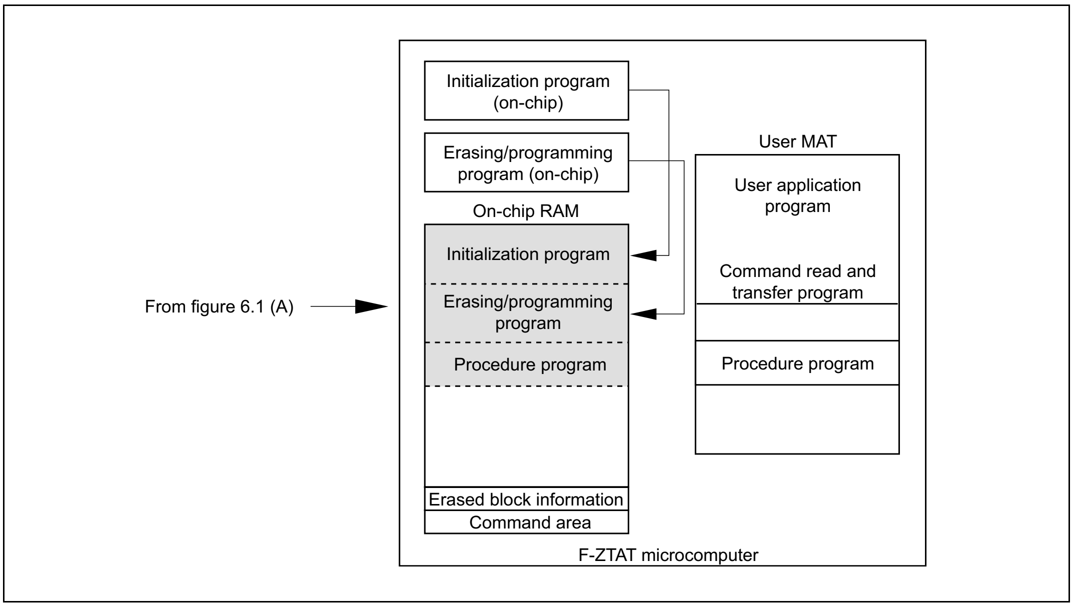

Figure 6.1 (B) Downloading the On-Chip Programs

Downloading the On-Chip Program: When erasing, the erasing program is downloaded to the RAM and when programming, the programming program is downloaded to the RAM; in either case, the initialization program is also downloaded to the RAM at the same time.

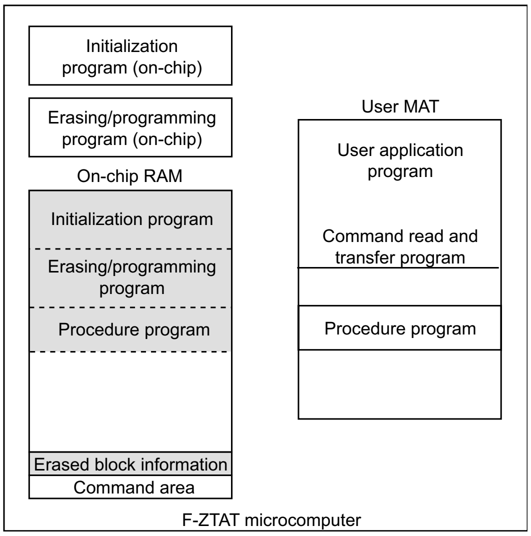

Figure 6.1 (C) Erasing the User MAT

Erasing the User MAT: A subroutine call makes the initialization program which has been downloaded to the RAM perform initialization which is required for erasing. After initialization has been completed, the user MAT is erased in response to a subroutine call to the erasing program which has been downloaded to the RAM from information on the block to be erased. The user MAT is erased in one divided-block units, so the erasing program must be called for each block which is to be erased.

Note that the area where the procedure program is stored and the blocks for the transfer program and command reading must not be erased.

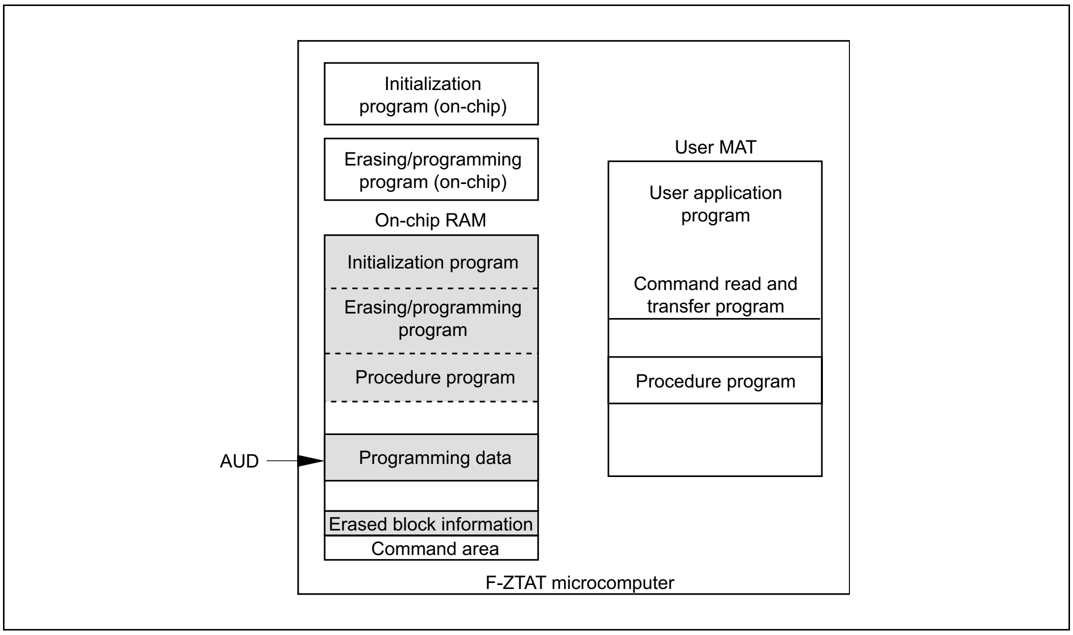

Figure 6.1 (D) Programming the User MAT

Programming the User MAT: A subroutine call makes the initialization program which has been downloaded to the RAM perform initialization which is required for programming.

After initialization has been completed, the user MAT is programmed the programming data which has been stored in the erased block of the on-chip RAM by the external AUD in response to a subroutine call to the programming program which has been downloaded to the RAM. The programming data is stored in the order from the start address of the erased block.

The user MAT is programmed in 128-byte units, so the programming data is stored in the on-chip RAM in every128-byte units by external AUD and the programming program is called. This process is repeated until all of the programming data (the application program) is programmed in the MAT.

Note that in this procedure program, data is programmed in order from the start address of the erased block. Data to be stored on the RAM by AUD is prepared in the order of address.

6.1.2 Details of Specification

Specification of User Application Program:

The command area is checked within the compare match processing in every 2 ms. When the flash reprogramming command is in the command area, the procedure program is executed after the program has been loaded to the on-chip RAM.

Specification of Procedure Program for Erasing/Programming Flash in User Program Mode:

The procedure program is loaded to the RAM and executed using the user application program.

Execution procedure:

1. Erase blocks from the erased block information after the erasing procedure program is issued.
2. After blocks have been erased, write the erase end command in the command area and require the data to be programmed.
3. Write the programming data which is prepared in on-chip RAM in the erased block using the programming procedure program. The data should be written in from the first address of the erased block.
4. Write the programming end command every time when 128-byte programming has been completed and require the subsequent programming data.

Specification of Command:

Command area is H′FFFFB000 to H′FFFFB003.

Command is rewritten by external AUD control or by CPU using the procedure program.

Table 6.1 gives command specifications.

Table 6.1 Specification of Command

Command (H′FFFFB000 to H′FFFFB003)

Command	Command Name	Description
H’464C5752	FLWR	Reprogramming flash memory (rewriting by AUD)
H’45444552	EDER	Flash memory erasing program download error (rewriting by CPU)
H’45494552	EIER	Flash memory erase initial setting error (rewriting by CPU)
H’45524552	ERER	Flash memory block erase error(rewriting by CPU)
H’45454E44	EEND	Flash memory erase successful end (rewriting by CPU)
H’57444552	WDER	Flash memory programming program download error (rewriting by CPU)
H’57494552	WIER	Flash memory programming initial setting error (rewriting by CPU)
H’57454E44	WEND	Flash memory page (128 bytes) programming successful end (rewriting by CPU)
H’57524552	WRER	Flash memory page (128 bytes) programming error (rewriting by CPU)
H’42554631	BUF1	Completion of preparing data to be written in area BUFF1 (rewriting by AUD)
H’42554632	BUF2	Completion of preparing data to be written in area BUFF2 (rewriting by AUD)

Erased Block Information:

Erased block information is H′FFFFB004 to H′FFFFB007.

Erased block information is rewritten by the external AUD just before the flash reprogramming command (FLWR) is rewritten. Information is specified in bits 0 to 15 and the bit 0 and bit 15 correspond the block EB0 and block EB15, respectively.

Figure 6.2 shows the command control sequence followed by the procedure program for erasing/programming flash in user program mode.

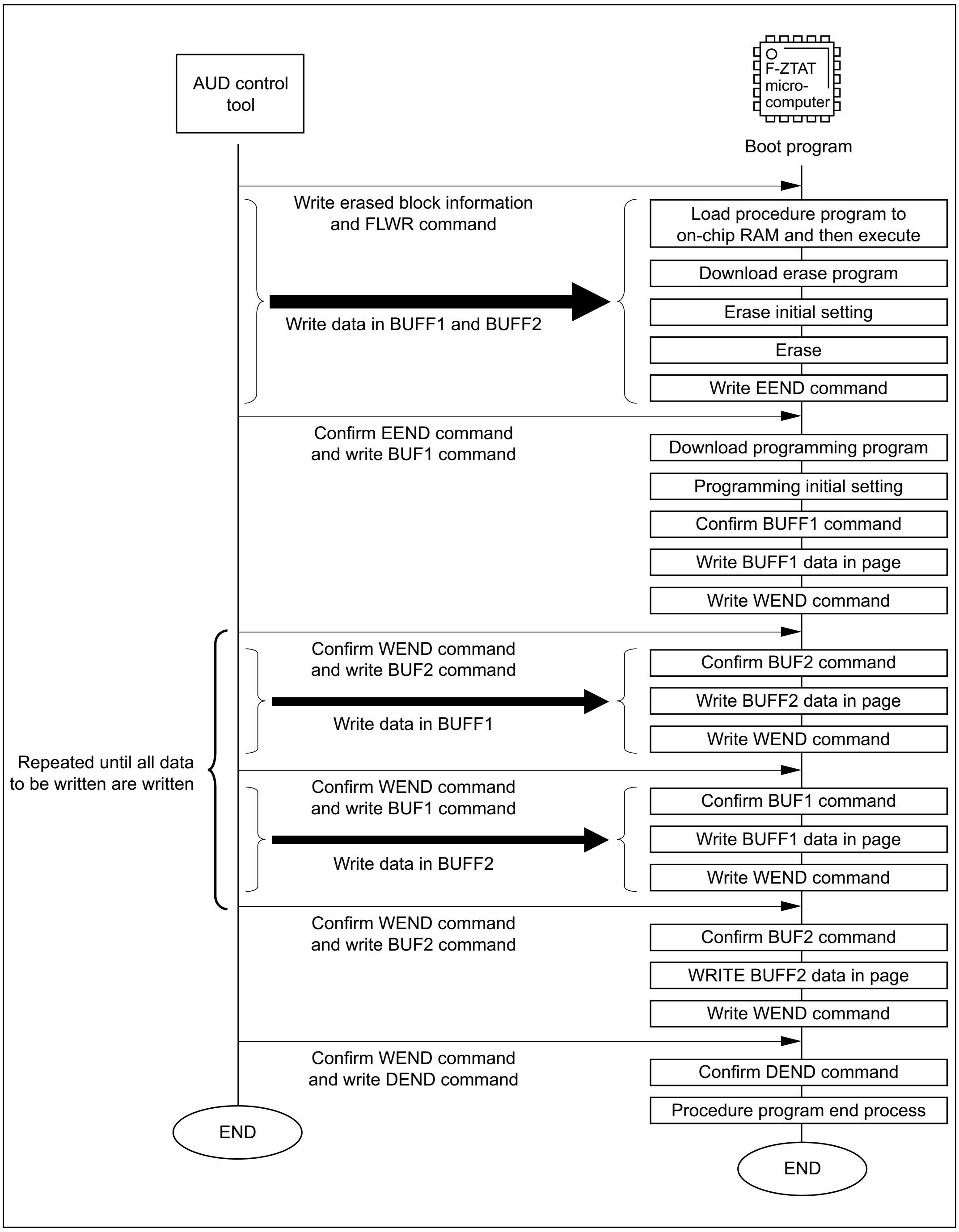
AUD control tool
F-ZTAT micro-computer
Boot program
Write erased block information and FLWR command
Write data in BUFF1 and BUFF2
Load procedure program to on-chip RAM and then execute
Download erase program
Erase initial setting
Erase
Write EEND command
Confirm EEND command and write BUF1 command
Download programming program
Programming initial setting
Confirm BUFF1 command
Write BUFF1 data in page
Write WEND command
Repeated until all data to be written are written
Confirm WEND command and write BUF2 command
Write data in BUFF1
Confirm BUF2 command
Write BUFF2 data in page
Write WEND command
Confirm WEND command and write BUF1 command
Write data in BUFF2
Confirm BUF1 command
Write BUFF1 data in page
Write WEND command
Confirm WEND command and write BUF2 command
Confirm BUF2 command
WRITE BUFF2 data in page
Write WEND command
Confirm WEND command and write DEND command
Confirm DEND command
Procedure program end process
END
END

Figure 6.2 Command Control Sequence

6.1.3 Memory Map

Table 6.2 gives the memory map of area for storing the user application program and procedure program in the user MAT, and area for executing the procedure program and erasing/programming program in the on-chip RAM.

Table 6.2 Memory Map of User MAT and On-Chip RAM

User MAT	
H′00000000 to H′0000039F	Vector table
H′00000600 to H′00000FFF	Compare match processing (analyzing command)
H′00001000 to H′000011CF	Procedure program store area
H′00002000 to H′0000FFFF	Area to be reprogrammed in flash
On-Chip RAM	
H′FFFF0000 to H′FFFF07FF	Erasing program download area
H′FFFF0000	DPFR (return value: 1 byte)
H′FFFF0001 to H′FFFF000F	System area (15 bytes)
H′FFFF0010 to H′FFFF001F	Erase process entry
H′FFFF0020 to H′FFFF002F	Erase initialization entry
H′FFFF0030 to H′FFFF07FF	Erase initialization and erasing programs
H′FFFF0800 to H′FFFF0FFF	Programming program download area
H′FFFF0800	DPFR (return value: 1 byte)
H′FFFF0801 to H′FFFF080F	System area (15 bytes)
H′FFFF0810 to H′FFFF081F	Programming process entry
H′FFFF0820 to H′FFFF082F	Programming initialization entry
H′FFFF0830 to H′FFFF0FFF	Programming initialization and programming programs
H′FFFF1000 to H′FFFF107F	BUFF1 (programming data store area 1)
H′FFFF1080 to H′FFFF10FF	BUFF2 (programming data store area 2)
H′FFFF1100 to H′FFFF12CF	Procedure program execution area
H′FFFFB000 to H′FFFFB003	Command area (4 bytes): command
H′FFFFB004 to H′FFFFB005	Erased block information (2 bytes): ERASE_block

List 1 and List 2 in Appendix show the compare match processing (analyzing command) program list and the procedure program list, respectively.

6.2 Example of Procedure Program for Erasing/Programming Flash of in User Boot Mode

6.2.1 Overview

This sample program is issued in user boot mode, then the flash are erased and programmed after the procedure program which is stored in the user boot MAT has been loaded to the on-chip RAM by the user boot program.

In this sample procedure program, the programming data is transferred by the external AUD.

User boot mode differs from user program mode; the procedure program is stored in the user boot MAT and MATs should be switched when erasing/programming the flash.

Figure 6.3 shows the flow of procedure in this sample application.

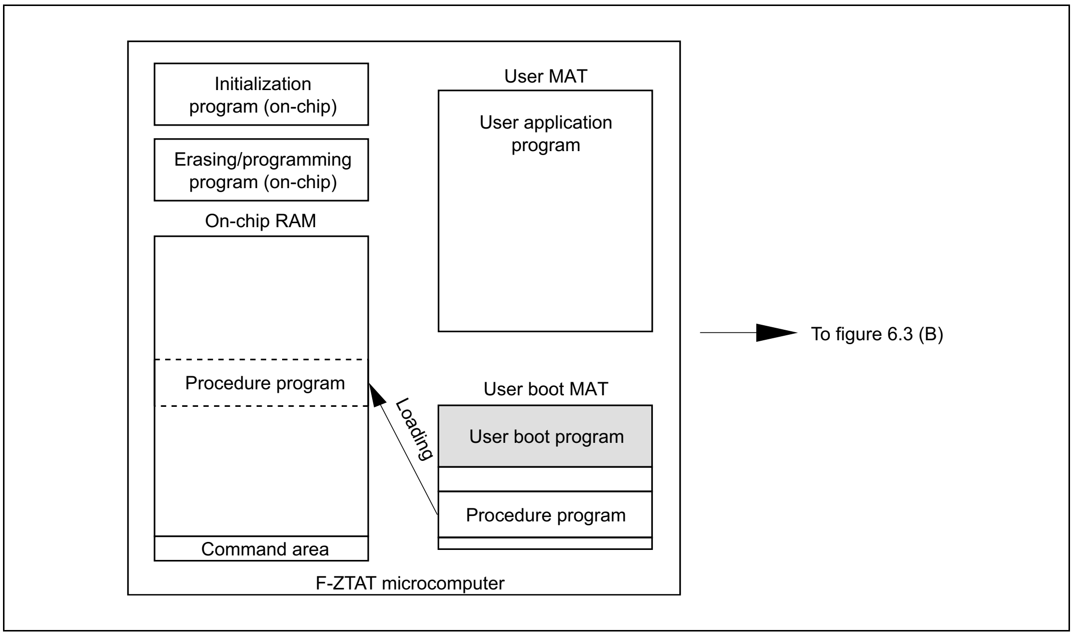

Figure 6.3 (A) Loading the Procedure Program to the RAM

Loading the Procedure Program to the RAM: The procedure program is loaded to the RAM by the user boot program after user boot mode is started up. Control from that point on will be by the procedure program in RAM.

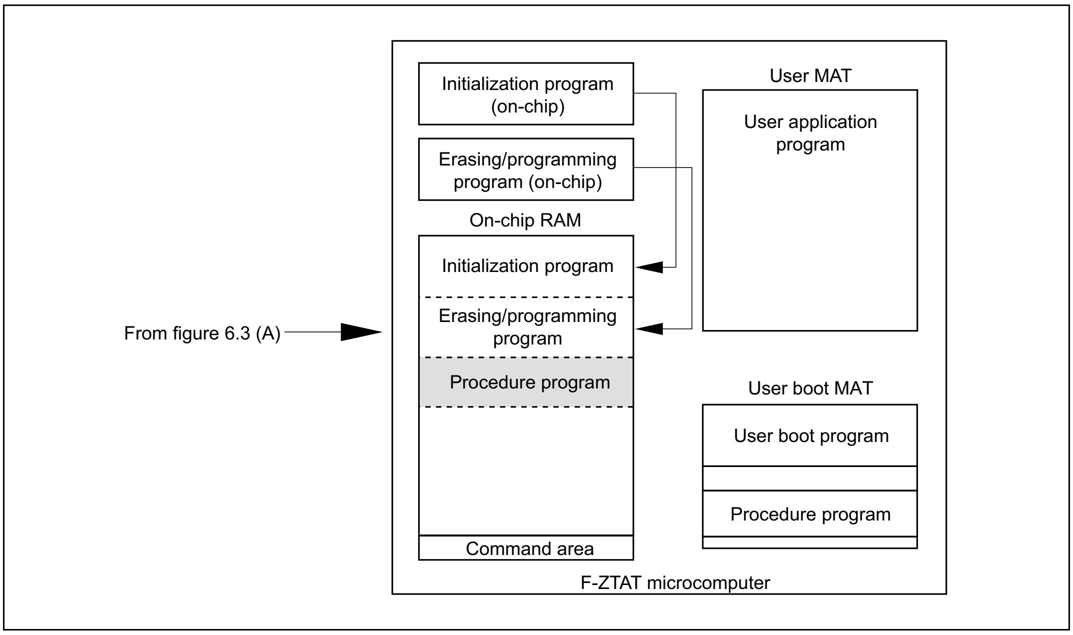

Figure 6.3 (B) Downloading the On-Chip Programs

Downloading the On-Chip Program: When erasing, the erasing program is downloaded to the RAM and when programming, the programming program is downloaded to the RAM; in either case, the initialization program is also downloaded to the RAM at the same time.

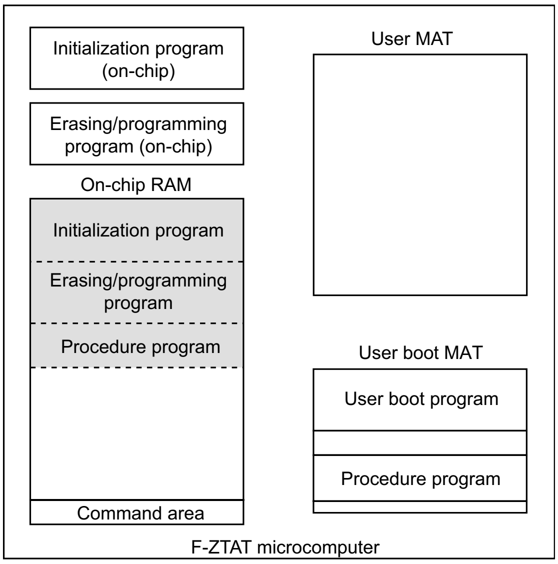

Figure 6.3 (C) Erasing the User MAT

Erasing the User MAT: A subroutine call makes the initialization program which has been downloaded to the RAM perform initialization which is required for erasing in user boot MAT. After initialization has been completed, the user MAT is erased in response to a subroutine call to the erasing program which has been downloaded to the RAM from information on the block to be erased. In this sample program, all data in the user MAT is erased.

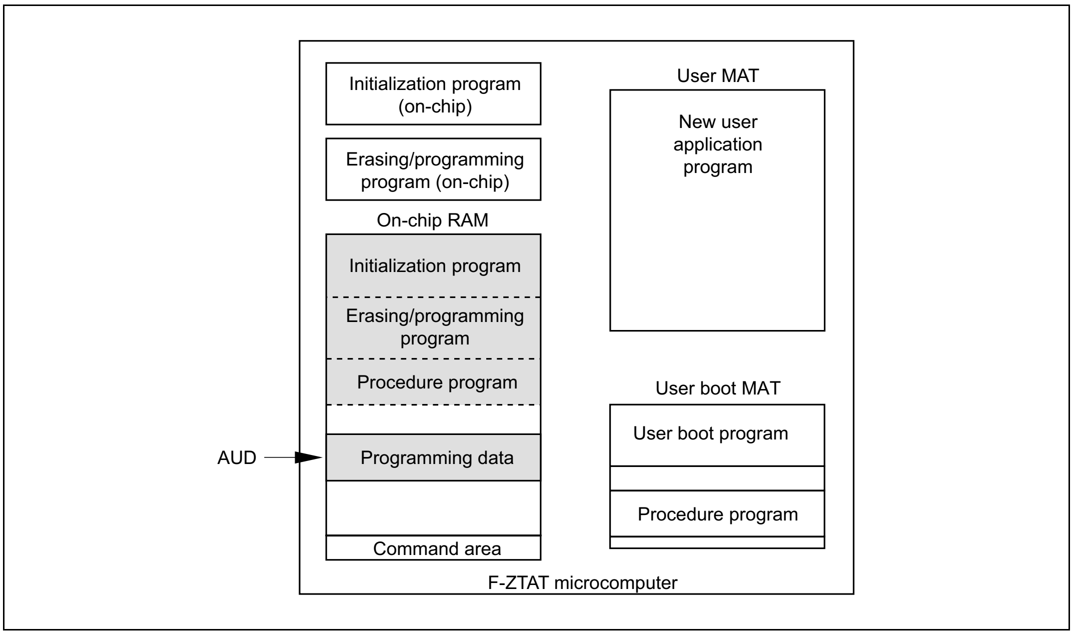

Figure 6.3 (D) Programming the User MAT

Programming the User MAT: A subroutine call makes the initialization program which has been downloaded to the RAM perform initialization which is required for programming.

After initialization has been completed, the user MAT is programmed in response to a subroutine call to the programming program which has been downloaded to the RAM in the order from the start address.

The user MAT is programmed from the start address to 1 Mbyte larger than the start address. Data is stored to the RAM in the order of address by external AUD.

6.2.2 Details of Specification

Specification of User Boot Program:

After staring up, the procedure program is loaded to the on-chip RAM, and then the user boot program is executed.

Specification of Procedure Program for Erasing/Programming Flash in User Boot Mode:

The procedure program is loaded to the RAM and executed using the user boot program.

Execution procedure:

1. Erase blocks after the erasing procedure program is issued and switching to the user MAT is made.
2. After blocks have been erased, write the erase end command in the command area and require the data to be programmed in user boot MAT.
3. Write the programming data which is prepared in on-chip RAM in the erased block after the programming procedure program is issued and switching to the user MAT is made. The data should be written in from the first address of the erased block.
4. Write the programming end command every time when 128-byte programming has been completed and require subsequent programming data.

Specification of Command:

Command area is H′FFFFB000 to H′FFFFB003.

Command is rewritten by external AUD control or by CPU using the procedure program.

Table 6.3 gives command specifications.

Table 6.3 Specifications of Command

Command (H′FFFFB000 to H′FFFFB003)

Command	Command Name	Description
H′45444552	EDER	Flash memory erasing program download error (rewriting by CPU)
H′45494552	EIER	Flash memory erase initial setting error (rewriting by CPU)
H′45524552	ERER	Flash memory block erase error(rewriting by CPU)
H′45454E44	EEND	Flash memory erase successful end (rewriting by CPU)
H′57444552	WDER	Flash memory programming program download error (rewriting by CPU)
H′57494552	WIER	Flash memory programming initial setting error (rewriting by CPU)
H′57454E44	WEND	Flash memory page (128 bytes) programming successful end (rewriting by CPU)
H′57524552	WRER	Flash memory page (128 bytes) programming error (rewriting by CPU)
H′42554631	BUF1	Completion of preparing data to be written in area BUFF1 (rewriting by AUD)
H′42554632	BUF2	Completion of preparing data to be written in area BUFF2 (rewriting by AUD)

Figure 6.4 shows the command control sequence followed by the procedure program for erasing/programming flash in user boot mode.

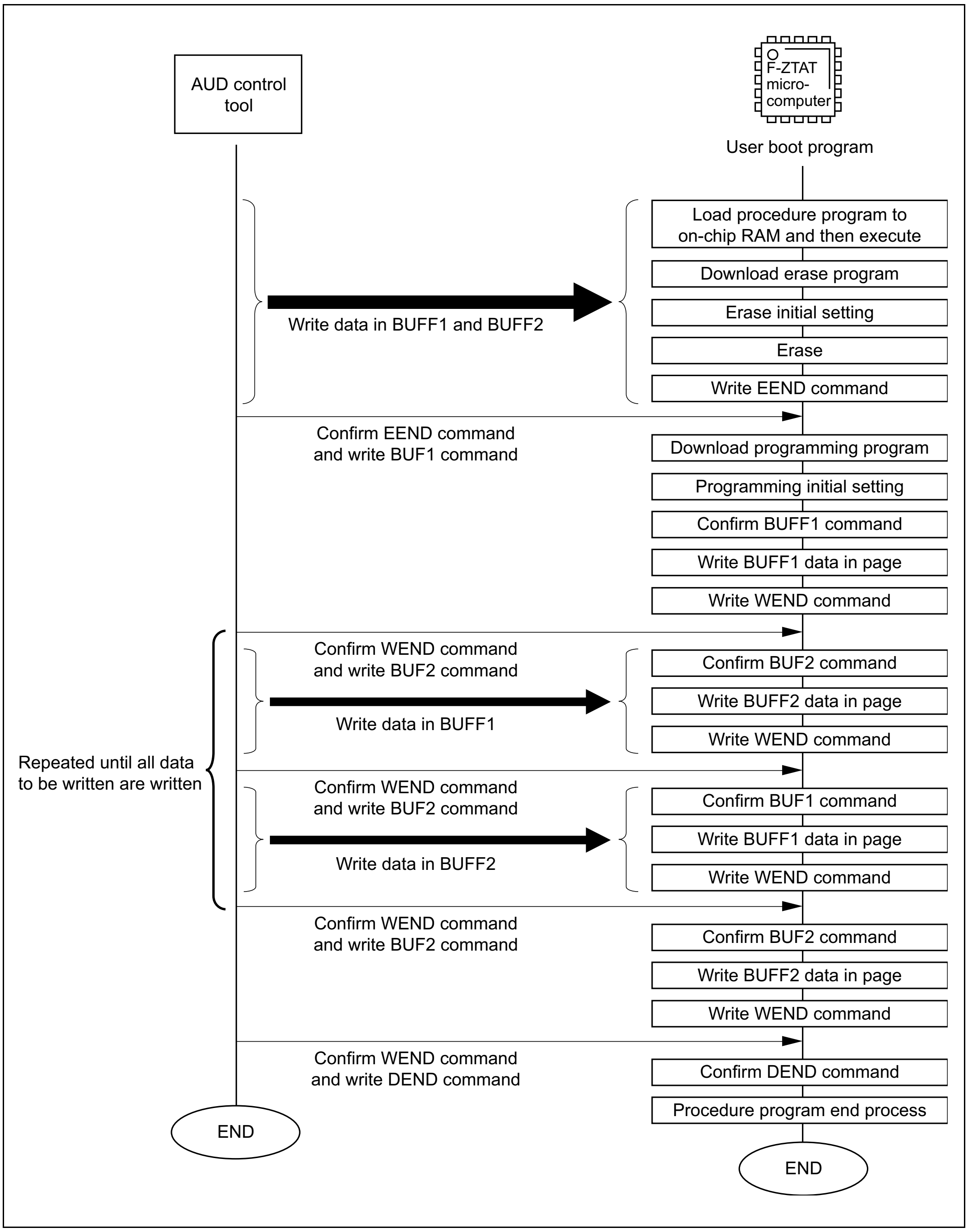

Figure 6.4 Command Control Sequence

6.2.3 Memory Map

Table 6.4 gives the memory map of area for storing the user application program and procedure program in the user boot MAT, and area for executing the procedure program and erasing/programming program in the on-chip RAM.

Table 6.4 Memory Map of User Boot MAT and On-Chip RAM

User Boot MAT	
H′00000000 to H′0000039F	Vector table
H′00000600 to H′00000FFF	Procedure program transfer program
H′00001000 to H′000011F4	Procedure program store area
On-Chip RAM	
H′FFFF0000 to H′FFFF07FF	Erasing program download area
H′FFFF0000	DPFR (return value: 1 byte)
H′FFFF0001 to H′FFFF000F	System area (15 bytes)
H′FFFF0010 to H′FFFF001F	Erase process entry
H′FFFF0020 to H′FFFF002F	Erase initialization entry
H′FFFF0030 to H′FFFF07FF	Erase initialization and erasing programs
H′FFFF0800 to H′FFFF0FFF	Programming program download area
H′FFFF0800	DPFR (return value: 1 byte)
H′FFFF0801 to H′FFFF080F	System area (15 bytes)
H′FFFF0810 to H′FFFF081F	Programming process entry
H′FFFF0820 to H′FFFF082F	Programming initialization entry
H′FFFF0830 to H′FFFF0FFF	Programming initialization and programming programs
H′FFFF1000 to H′FFFF107F	BUFF1 (programming data store area 1)
H′FFFF1080 to H′FFFF10FF	BUFF2 (programming data store area 2)
H′FFFF1100 to H′FFFF12F4	Procedure program execution area
H′FFFFB000 to H′FFFFB003	Command area (4 bytes): command

List 3 and List 4 in Appendix show the procedure program transferring program list and the procedure program list, respectively.

Appendix Program Listing

List 1 Compare Match Processing (Command Analyzing) Program

```
#include        <stdio.h>
#include        <machine.h>
#include        "SH7058.h"
#include        "reg.h"
#pragma inline_asm(jmp_RAM1100)
void  jmp_RAM1100(void){                 /* Jump processing after transfer                                */
      MOV.L #H'FFFF1100,R0
      JMP       @R0
      NOP
}
extern unsigned char *_FWRT_BGN;         /* Start address for storing the procedure program               */
extern unsigned char *_FWRT_END;         /* End address for storing the procedure program                 */
extern unsigned char *_FWRT_RAM;         /* Destination address for transferring the procedure program    */

#pragma interrupt(CMT00)
void CMT00(void){
      unsigned char *src, *dst;
      CMT0.CMCSR &= 0x0041;              /* Clear the compare match interrupt flag                        */
      if(command == FWRT){               /* Analyze the command                                           */
           set_imask(0xF);               /* Interrupt mask                                                */
/* Transfer the procedure program to the on-chip RAM        */
           for(src=_FWRT_BGN, dst=_FWRT_RAM; src<=_FWRT_END; src++, dst++){
               *dst = *src;
           }
           jmp_RAM1100();                /* Jump to the on-chip RAM                                       */
      }
}
```

List 2 Procedure Program in User Program Mode

```
#include        <machine.h>
#include        <stdio.h>
#include        "SH7058.h"
#include        "reg.h"

#pragma inline_asm(NOP4)
static void NOP4(void){
   NOP
   NOP
   NOP
   NOP
}

void FWRT_main(void){                  /* 0X00001000 → 0X0FFFF1100                                    */
   unsigned long ERR_CHECK;
   unsigned long FPEFEQ;
   unsigned long FUBRA;
   unsigned long FPFR;
   unsigned long FEBS;
   unsigned long FMPAR;
   unsigned long FMPDR;

/* Download the erasing program   */
   FLASH.FTDAR = RAM00;               /* Set the download destination address (from H'FFFF0000)       */
   RAM00_TOP = 0xFF;                  /* Clear the register for confirming download processing error  */
   FLASH.FECS = 0x01;                 /* Select the erasing program (set EPVB)                        */
   FLASH.FKEY = 0xA5;                 /* Enable downloading the erasing program                       */
   FLASH.FCCS |= 0x01;                /* Require downloading (set SCO)                                */
   NOP4();                            /* NOP instruction × 4                                          */
   FLASH.FKEY = 0x00;                 /* FKEY clear                                                   */
   if(RAM00_TOP != 0x00){             /* Download error?                                              */
        command = EDER;               /* Download error processing 0x45444552
   */
        while(1);
   }
```

```
/*    Erase initial setting          */
      FPEFEQ = 0x00000FA0;                    /* 40 MHz    H'0FA0 => D'4000                              */
      FUBRA = 0x00000000;
      ERR_CHECK = FLASH_INIT(FPEFEQ,FUBRA);   /* FPEFEQ-->R4,FUBRA-->R5,ERR_CHECK-->R0                   */
      if(ERR_CHECK != 0x00000000){            /* Erase initial setting error?                            */
           command = EIER;                    /* Erase initial setting error processing  0x45494552      */
           while(1);
      }
/*    Erase processing     */
      FLASH.FKEY = 0x5A;                      /* Set FKEY                                                */
      FEBS = 0x00000000;
      while(ERASE_block != 0x0000){           /* Erase block                                             */
           if((ERASE_block & 0x0001)==0x0001){
               ERR_CHECK = FLASH_ERASE(FEBS); /* FEBS-->R4,ERR_CHECK-->R0                                */
               if(ERR_CHECK != 0x00000000){   /* Erase error?                                            */
                    command = ERER;           /* Erase error processing  0x45524552                      */
                    while(1);
               }
           }
           ERASE_block = (ERASE_block>>1);
           FEBS++;
      }
      FLASH.FKEY = 0x00;                      /* Clear FKEY                                              */
      command = EEND;                         /* Successful completion of erasing 0x45454E44             */
/*    Download the programming program    */
      FLASH.FTDAR = RAM08;                    /* Set the download destination address (H'FFFF0800)       */
      RAM08_TOP = 0xFF;                       /* Clear the register for confirming download processing error  */
      FLASH.FPCS = 0x01;                      /* Select the programming program (set PPVS)               */
      FLASH.FKEY = 0xA5;                      /* Enable downloading the programming program              */
      FLASH.FCCS |= 0x01;                     /* Require downloading (set SCO)                           */
      NOP4();                                 /* NOP instruction × 4                                     */
      FLASH.FKEY = 0x00;                      /* Clear FKEY                                              */
      if(RAM08_TOP != 0x00){                  /* Download error?                               */
           command = WDER;                    /* Download error processing 0x57444552                    */
           while(1);
      }
```

```
/*    Programming initial setting    */
      FPEFEQ = 0x00000FA0;                               /* 40 MHz H'0FA0 => D'4000                          */
      FUBRA = 0x00000000;
      ERR_CHECK = FLASHwr_INIT(FPEFEQ,FUBRA);            /* FPEFEQ-->R4,FUBRA-->R5,ERR_CHECK-->R0            */
      if(ERR_CHECK != 0x00000000){                       /* Initial setting error?                           */
           command = WIER;                               /* Initial setting error processing  0x57494552     */
           while(1);
      }
      WRITE_block = 0x00010000;
/*    Programming processing         */
      FLASH.FKEY = 0x5A;                                 /* Set FKEY                                         */
      FMPAR = 0x00002000;                                /* Program 0x00002000 to 0x000FFFFF                 */
      while(command != DEND){                            /* Completion of programming?                       */
           while(command != BUF1 && command != BUF2);    /* Wait for command input 0x42554631                */
           if(command == BUF1){                          /* Write data in BUFF1                              */
               FMPDR = 0xFFFF1000;
           }
           else{
               FMPDR = 0xFFFF1080;
           }
           ERR_CHECK = FLASH_WRITE(FMPDR,FMPAR);         /* FMPDR-->R4,FMPAR-->R5,ERR_CHECK-->R0             */
           if(ERR_CHECK != 0x00000000){                  /* Programming error?                               */
               command = WRER;                           /* Programming error processing  0x57524552         */
               while(1);
           }
           FMPAR += 0x80;
           command = WEND;                               /* Completion of writing in page  0x57454E44        */
      }
      FLASH.FKEY = 0x00;                                 /* Clear FKEY                                       */
      while(1);
}
```

List 3 Procedure Program Transferring Program

```
/*    Main program in user boot MAT       */
#include        <machine.h>
#include        <stdio.h>
#include        "SH7058FCC.h"
#include        "reg.h"
#pragma inline_asm(jmp_RAM)
static voidjmp_RAM(void){                 /* Jump processing after transfer                              */
      MOV.L #H'FFFF1100,R0
      JMP       @R0
      NOP
}
extern unsigned char *_FWRT_BGN;          /* Start address for storing the procedure program             */
extern unsigned char *_FWRT_END;          /* End address for storing the procedure program               */
extern unsigned char *_FWRT_RAM;          /* Destination address for transferring the procedure program    */

void uboot_main(void){                    /*    0x0001000                                                 */
      unsigned char *src, *dst;
/*    Copy from the user boot MAT to the on-chip RAM */
      for(src=_FWRT_BGN, dst=_FWRT_RAM; src<=_FWRT_END; src++, dst++){
            *dst = *src;
      }
      jmp_RAM();
      while(1);
}
```

List 4 Procedure Program in User Boot Mode

```
#include        <machine.h>
#include        <stdio.h>
#include        "SH7058.h"
#include        "reg.h"
#pragma inline_asm(NOP4)
static void NOP4(void){
      NOP
      NOP
      NOP
      NOP
}

void e_main(void){                       /* 0xFFFF2000                                              */
      unsigned long ERR_CHECK,FPEFEQ,FUBRA,FPFR,FEBS,FMPAR,FMPDR;
      /*   Erasing program download processing */
      FLASH.FTDAR = RAM00;               /* Set the download destination address (H'FFFF0000)       */
      RAM00_TOP = 0xFF;                  /* Clear the register for confirming download processing error  */
      FLASH.FECS = 0x01;                 /* Select the erasing program (set EPVB)                   */
      FLASH.FKEY = 0xA5;                 /* Enable downloading the erasing program                  */
      FLASH.FCCS |= 0x01;                /* Require downloading (set SCO)                           */
      NOP4();                            /* NOP instruction × 4                                     */
      FLASH.FKEY = 0x00;                 /* Clear FKEY                                              */
      if(RAM00_TOP != 0x00){             /* Download error?                                         */
            command = EDER;              /* Download error processing 0x45444552                    */
            while(1);
      }
/*    Erase initial setting          */
      FPEFEQ = 0x00000FA0;               /* 40 MHz     H'0FA0 => D'4000                             */
      FUBRA = 0x00000000;
      ERR_CHECK = FLASH_INIT(FPEFEQ,FUBRA);/* FPEFEQ-->R4,FUBRA-->R5,ERR_CHECK-->R0                 */
      if(ERR_CHECK != 0x00000000){       /* Initial setting error?                                  */
            command = EIER;              /* Initial setting error processing 0x45494552             */
            while(1);
      }
      FLASH.FMATS = 0x55;                /* Switch to the user MAT                                  */
      NOP4();                                 /* NOP instruction × 4                                */
/*    Erase processing     */
      FLASH.FKEY = 0x5A;                      /* Set FKEY                                           */
      FEBS = 0x00000000;
      ERASE_block = 0xFFFF;                   /* all erase                                          */
      while(ERASE_block != 0x0000){           /* Block erase processing                             */
            if((ERASE_block & 0x0001)==0x0001){
                ERR_CHECK = FLASH_ERASE(FEBS);  /* FEBS-->R4,ERR_CHECK-->R0                          */
                if(ERR_CHECK != 0x00000000){    /* Erase error?                                      */
                      command = ERER;           /* Erase error processing 0x45524552                 */
                      while(1);
                }
            }
            ERASE_block = (ERASE_block>>1);
            FEBS++;
      }
      FLASH.FKEY = 0x00;                      /* Clear FKEY                                         */
      FLASH.FMATS = 0xAA;                     /* Switch to the user boot MAT                        */
      NOP4();                                 /* NOP instruction × 4                                */
```

```
     command = EEND;                        /* Successful completion of erasing 0x45454E44          */
     while(command == EEND);
/*   Programming program download processing   */
     FLASH.FTDAR = RAM08;                   /* Set the download destination address (H'FFFF0800)    */
     RAM08_TOP = 0xFF;                      /* Clear the register for confirming download processing error  */
     FLASH.FPCS = 0x01;                     /* Select programming program (set PPVS)                */
     FLASH.FKEY = 0xA5;                     /* Enable downloading the programming program           */
     FLASH.FCCS |= 0x01;                    /* Require downloading (set SCO)                        */
     NOP4();                                /* NOP instruction × 4                                  */
     FLASH.FKEY = 0x00;                     /* Clear FKEY                                           */
     if(RAM08_TOP != 0x00){                 /* Download error                                       */
          command = WDER;                   /* Download error processing 0x57444552                 */
          while(1);
     }
/*   Programming initial setting   */
     FPEFEQ = 0x00000FA0;                           /* 40 MHz H'0FA0 => D'4000                      */
     FUBRA = 0x00000000;
     ERR_CHECK = FLASHwr_INIT(FPEFEQ,FUBRA);        /* FPEFEQ-->R4,FUBRA-->R5,ERR_CHECK-->R0        */
     if(ERR_CHECK != 0x00000000){                   /* Initial setting error?                       */
          command = WIER;                           /* Initial setting error 0x57494552             */
          while(1);
     }
     FLASH.FMATS = 0x55;                            /* Switch to the user MAT                       */
     NOP4();                                        /* NOP instruction × 4                          */
/* Programming processing     */
     FLASH.FKEY = 0x5A;                             /* Set FKEY                                     */
     FMPAR=0x00000000;
     while(FMPAR < 0x00100000){                     /* Completion of programming                    */
          while(command != BUF1 && command != BUF2);  /* Wait for command input 0x42554631          */
          if(command == BUF1){                      /* Command 0x42554631                           */
              FMPDR = 0xFFFF1000;
          }
          else{
              FMPDR = 0xFFFF1080;
          }
          ERR_CHECK = FLASH_WRITE(FMPDR,FMPAR);     /* FMPDR-->R4,FMPAR-->R5,ERR_CHECK-->R0         */
          if(ERR_CHECK != 0x00000000){              /* Programming error?                           */
              command = WRER;                       /* Programming error processing  0x57524552     */
              while(1);
          }
          FMPAR += 0x80;
          command = WROK;                           /*                                              */
     }
     FLASH.FKEY = 0x00;                             /* Clear FKEY                                   */
     FLASH.FMATS = 0xAA;                            /* Switch to the user boot MAT                  */
     NOP4();                                        /* NOP instruction × 4                          */
     command = WEND;;                               /* Message of programming completion 0x57454E44 */
     while(1);
}
```