R8C/35C Group

Rewriting the Data Flash with DTC

REJ05B1242-0100
Rev.1.00
Apr. 21, 2010

# 1. Abstract

This document describes the setting method and an application example for rewriting the data flash using the DTC (chain transfer) and a flash memory ready interrupt in the R8C/35C Group.

# 2. Introduction

The application example described in this document applies to the following microcomputer (MCU):

- MCU: R8C/35C Group

This application note can be used with other R8C Family MCUs which have the same special function registers (SFRs) as the above group. Check the manual for any modifications to functions. Careful evaluation is recommended before using the program described in this application note.

# 3. Application Example

When rewriting the flash memory (rewrite or erase) in EW1 mode, the following show the differences depending on flash memory areas:

- Program ROM area: CPU is in a hold state (states of the I/O ports are retained prior to command execution).
- Data flash area: CPU is in operating due to a background operation (BGO).

When rewriting the data flash area, other processes can be performed during the write or erase operation. Use the flash memory ready interrupt to generate interrupts when a write operation is completed, an erase operation is completed, an error occurs, etc. In this application note, when executing the block erase, use the flash ready status interrupt to generate interrupts when an erase operation is completed. During interrupt handling, rewriting status check and data flash block is disabled, and CPU rewrite mode is disabled.

In this application note, write complete and block erase complete are detected by the following interrupts:

- Write complete or write error: Use the flash ready status interrupt or the erase/write error interrupt
- Erase complete: Use the flash ready status interrupt

Data over 1 byte can be written with the write operation is used in conjunction with the DTC. Selecting the flash ready status interrupt as the DTC activation source and enabling chain transfers for the first transfer (program command 40h) and second transfer (write data) enables two data transfers (write) by one activation source.

The first byte is written by the program. The DTC is activated by the flash ready status interrupt request generated when a write operation is completed, and from the second byte on, write operations are performed in succession.

In the interrupt handler that is generated by a write complete (DTC transfer complete), write error, or erase complete, settings such as status check, data flash block rewrite disable, and CPU rewrite mode disable are performed.

## 3.1 Program Outline

The switch (SW1) and LEDs (LED0 to LED3) on the Renesas Starter Kit for R8C/35C Group are used to direct write data to a record (1) and display the number of writes. After detecting that the switch has been pressed, data for one record is written to an empty record. (1) The same process is performed each time the switch is detected as being pressed and the number of writes are counted. However, during a write or erase operation, the pressed switch detection is ignored. The 4 lower bits of the number of writes are displayed on the LEDs. When each bit is 1, the LEDs are turned on. When each bit is 0, the LEDs are turned off.
The write complete, erase complete, write error, and block erase error are detected by the flash memory ready interrupt.

Note:
1. Details on records and the empty record search are described below.

Figure 3.1 shows an example of key and LED connections. Figure 3.1 lists the pins used and their functions.

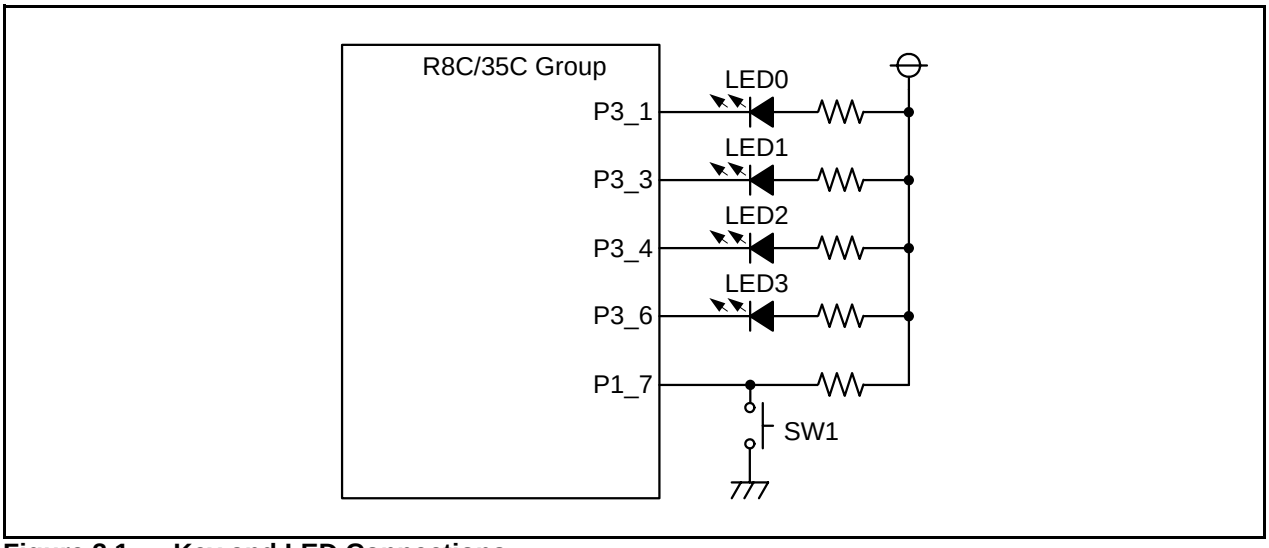



**Figure 3.1 Key and LED Connections**

**Table 3.1 Pins and Their Functions**

| Pin Name | I/O | Function |
|---|---|---|
| P1_7 | Input | Switch SW1 input |
| P3_1 | Output | LED0 output (bit 3 value of the number of writes) |
| P3_3 | Output | LED1 output (bit 2 value of the number of writes) |
| P3_4 | Output | LED2 output (bit 1 value of the number of writes) |
| P3_6 | Output | LED3 output (bit 0 value of the number of writes) |

## 3.2 Data Flash Area

In this application note, one record is 64 bytes and blocks are divided by 16. There is a total of 64 records in blocks A to D. Figure 3.2 shows the relationship between the data flash and the records.

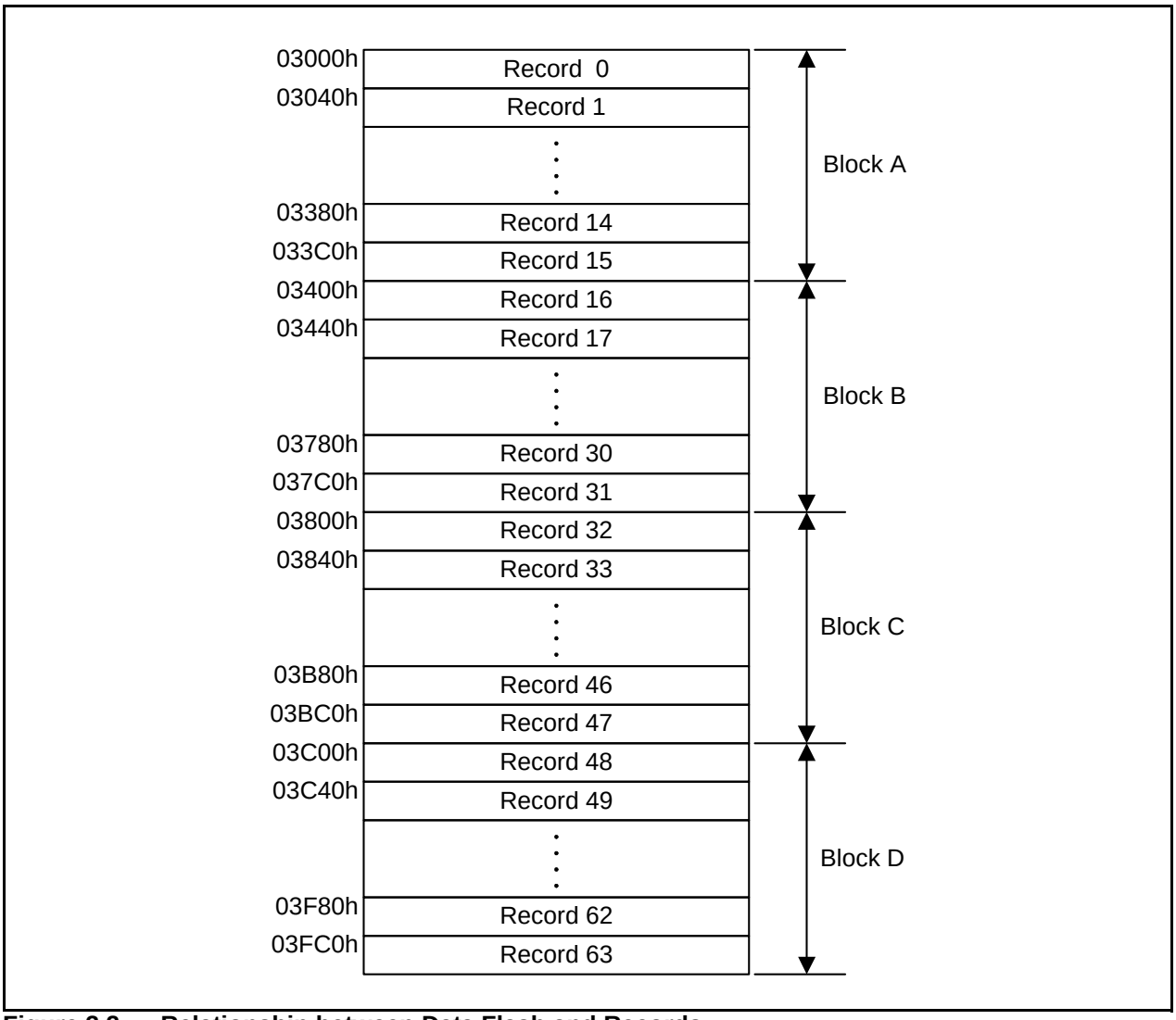



**Figure 3.2 Relationship between Data Flash and Records**

### 3.2.1 Empty Record Search (Data FFh Search)

Data written to the data flash is retained even if the power is turned off. Empty records in which all data are FFh are searched after a reset start. Empty record search the method are described below.

(1) Set the search pointer to the starting address in record 0.

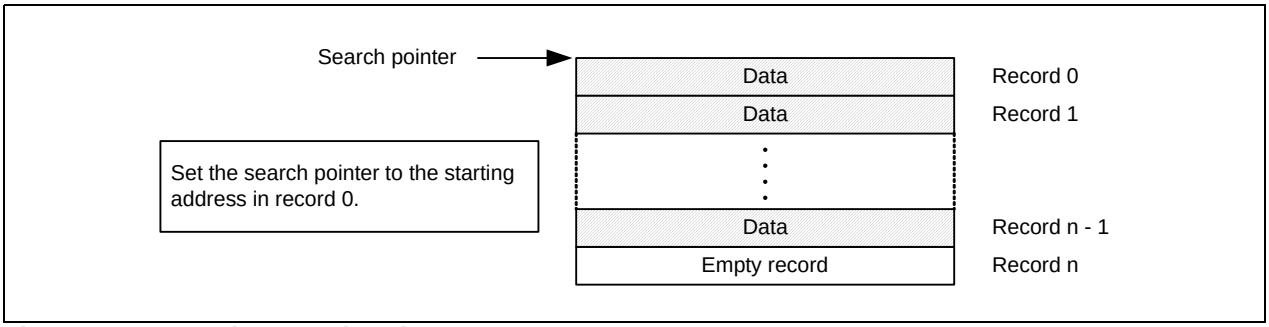



**Figure 3.3 Set the Search Pointer**

(2) Check to see that the record the search pointer indicates is an empty record.

(3) When the record is not empty, set the search pointer to the starting address in the next record.

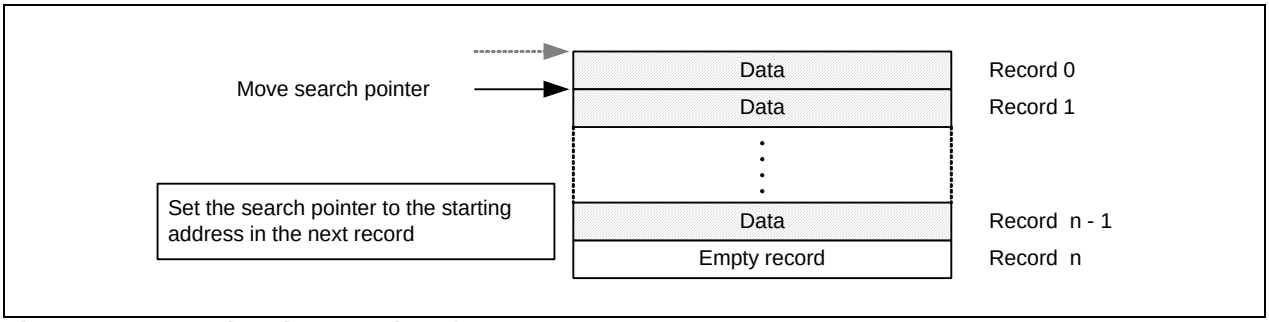



**Figure 3.4 Moving the Search Pointer**

(4) Repeat steps (2) and (3) until an empty record is found or all records are checked.

(5) When an empty record is found, set the starting address in the empty record to the data write address and set the block in which the empty record is stored as a used block.

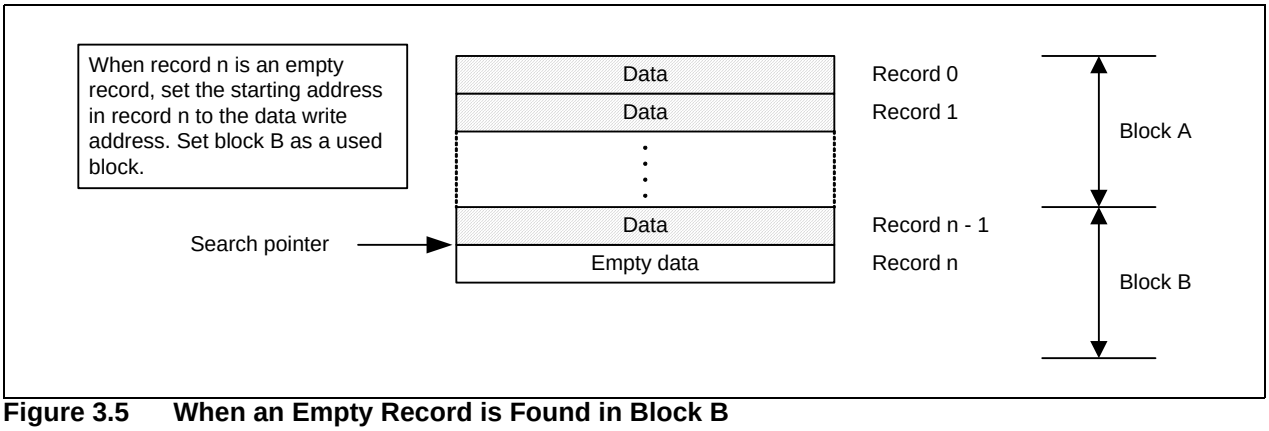



**Figure 3.5 When an Empty Record is Found in Block B**

(6) When an empty record is not found in any block, erase block A, set the starting address in record 0 to the data write address, and set block A as a used block.

### 3.2.2 Write Record and Erase Block

Write records sequentially based on the data write address and used blocks received by the empty record search. When writing data up to record 15, erase (block erase) all data in the next block (block B). When writing data to the record in the next step, start writing from record 16. When writing data up to the last record in each block, erase (block erase) all data in the next block. When writing data up to record 63, return to block A, erase (block erase) all data in block A, and write data from record 0 again.
When writing to a record, enable the flash ready status interrupt and erase/write error interrupt so that when writing is completed, generate the flash ready status interrupt. If a write error occurs during writing, an erase/write error interrupt is generated.
When erasing blocks, enable the flash ready status interrupt and generate it after erasure is completed.
The following are performed in the flash memory ready interrupt processes.

When writing:

(1) The flash memory ready interrupt is disabled.
(2) The flash ready status interrupt is disabled.
(3) The flash ready status interrupt request flag is cleared.
(4) The erase/write error interrupt is disabled.
(5) A full status check is performed.
(The number of writes (write_cnt) after writing one record successfully is updated.)
(6) The rewrite disable bit for a used block is set.
(7) CPU rewrite mode is disabled.
(8) The data write address is updated.
(9) The last record of the block is determined.
(When the record written is the last record, set an erase request for the next block, and the block used to execute the block erase in the main process is updated.)

When erasing:

(1) The flash memory ready interrupt is disabled.
(2) The flash ready status interrupt is disabled.
(3) The flash ready status interrupt request flag is cleared.
(4) The erase/write error interrupt is disabled.
(5) The full status check is performed.
(6) The rewrite disable bit for the used block is set.
(7) CPU rewrite mode is disabled.

## 3.3 DTC Activation by Flash Ready Status Interrupt

When writing one record, write command 40h and write data are written to the first byte of the written record by a program. From the second byte on, DTC is activated by the flash ready status interrupt request generated by the first byte writing being completed then data is written.
When DTC is activated, it reads control data 0 and writes 40h to the address in write record area. After that, as the chain transfer enable bit in DTC control register 0 is set to 1 (enabled), DTC reads control data 1 allocated next to control data 0 and writes the data to the address in write record area. As the chain transfer enable bit in DTC control register 1 is set to 0 (disabled), chain transfer is completed.
These processes are repeated the number of times set in DTC transfer count register 0.

Table 3.2 shows the Control Data 0 Settings, Table 3.3 shows the Control Data 1 Settings and Figure 3.6 shows the DTC activation and chain transfer operation by flash ready status interrupt.

**Table 3.2 Control Data 0 Settings**

| | |
|---|---|
| Control data address | Control data 0 (addresses 2C40 to 2C47) |
| Transfer mode | Normal mode |
| Source address control | Fixed |
| Destination address control | Incremented |
| Chain transfer | Enabled (DTC activation of control data 1 after executing DTC of control data 0) |
| Size of data block to be transferred per activation | 1 byte |
| Number of DTC data transfers | 63 (64 bytes (1 record) - 1) |
| Source address | Address of the variable where write command 40h is stored (&command_data) |
| Destination address | Data write address + 1 (write_addr+1) |

**Table 3.3 Control Data 1 Settings**

| | |
|---|---|
| Control data address | Control data 1 (addresses 2C48 to 2C4F) |
| Transfer mode | Normal mode |
| Source address control | Incremented |
| Destination address control | Incremented |
| Chain transfer | Disabled |
| Size of data block to be transferred per activation | 1 byte |
| Number of DTC data transfers | 63 (1) |
| Source address | Address of the variable where write data to the second byte is stored (&record_data[1]) |
| Destination address | Data write address + 1 (write_addr+1) |

Note:
1. The number of data transfers does not need to be set.

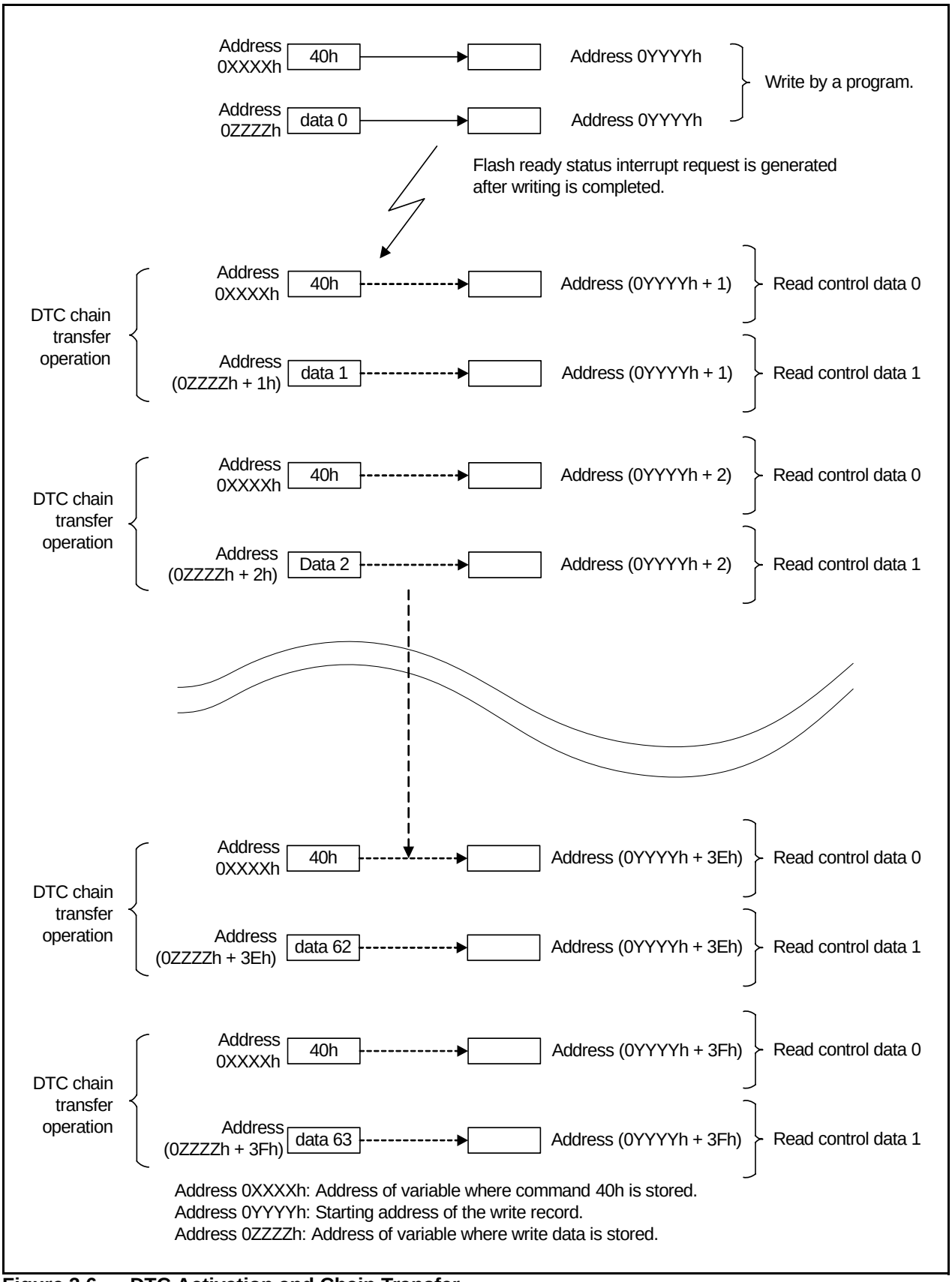



**Figure 3.6 DTC Activation and Chain Transfer**

## 3.4 Advantages of Using DTC

Figure 3.7 shows the data flash write operations when writing to the data flash with the DTC and when writing to the data flash without the DTC.

When writing to the data flash without the DTC, 64-byte data is written while waiting for the write completion (FST7 = 1) by a program. The CPU is occupied by the data flash write during (1) in the figure below. When writing to the data flash with the DTC, the DTC transfer is performed only when necessary during (2). The CPU performs other processes while writing to the data flash (FST7 = 0 (busy)) and the CPU is not occupied by writing to the data flash.

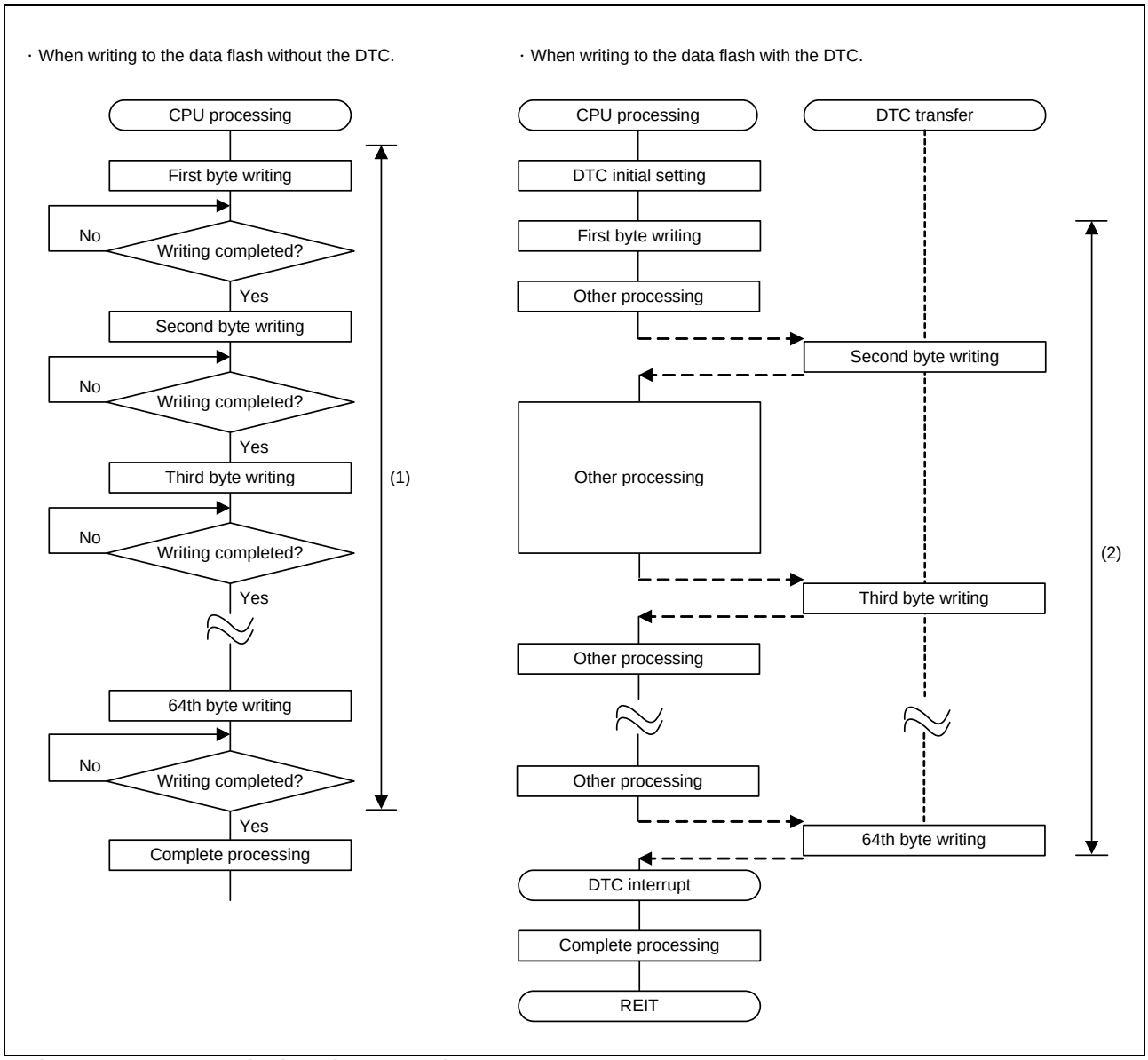



**Figure 3.7 Data Flash Write Operations**

## 3.5 Memory

**Table 3.4 Memory**

| Memory | Size | Remarks |
|---|---|---|
| ROM | 1143 bytes | In the rej05b1242_src.c module |
| RAM | 75 bytes | In the rej05b1242_src.c module |
| Maximum user stack | 16 bytes | |
| Maximum interrupt stack | 25 bytes | |

Memory size varies depending on the C compiler version and compile options.
The above applies to the following conditions:
C compiler: M16C/60, 30, 20, 10, and Tiny and R8C/Tiny Series Compiler V.5.45 Release 00
Compile option: -c -finfo -dir “$(CONFIGDIR)” -R8C

# 4. Software

This section shows the initial setting procedures and values to set the example described in section **3. Application Example**. Refer to the latest **R8C/35C Group hardware user's manual** for details on individual registers.

The × in the register's Setting Value represents bits not used in this application, blank spaces represent bits that do not change, and the dash represents reserved bits or bits that have nothing assigned.

## 4.1 Function Tables

| | | | |
|---|---|---|---|
| Declaration | void main(void) | | |
| Outline | Main processing | | |
| Argument | Argument name | | Meaning |
| | None | | — |
| Variable (global) | Variable name | | Contents |
| | unsigned char erase_req_mode | | Erase request mode |
| | unsigned char block_select | | Block selected |
| | unsigned char status | | Full status check result |
| Returned value | Type | Value | Meaning |
| | None | — | — |
| Function | When the switch is pressed and CPU rewrite mode is disabled, the data write operation is controlled. When an erase operation is requested, execute the block erase. | | |

| | | | |
|---|---|---|---|
| Declaration | void mcu_init(void) | | |
| Outline | System clock setting | | |
| Argument | Argument name | | Meaning |
| | None | | — |
| Variable (global) | Variable name | | Contents |
| | None | | — |
| Returned value | Type | Value | Meaning |
| | None | — | — |
| Function | Set the system clock (high-speed on-chip oscillator). | | |

| | | | |
|---|---|---|---|
| Declaration | void write_address_init(void) | | |
| Outline | Record write address initial setting | | |
| Argument | Argument name | | Meaning |
| | None | | — |
| Variable (global) | Variable name | | Contents |
| | unsigned char *write_addr | | Write address |
| | unsigned char block_select | | Block selected |
| Returned value | Type | Value | Meaning |
| | None | — | — |
| Function | Search for an empty record, set a block which has an empty record as a used block (block_select), and set the starting address in the empty record as the write address (write_addr). | | |

| | | | |
|---|---|---|---|
| Declaration | void write_control(void) | | |
| Outline | Data write control | | |
| Argument | Argument name | | Meaning |
| | None | | — |
| Variable (global) | Variable name | | Contents |
| | unsigned char record_data[RECORD_SIZE] | | Write data |
| Returned value | Type | Value | Meaning |
| | None | — | — |
| Function | Set the write data in the write data generation process. Execute the write operation after initializing the DTC. | | |

| | | | |
|---|---|---|---|
| Declaration | void set_data(unsigned char *data) | | |
| Outline | Write data generation | | |
| Argument | Argument name | | Meaning |
| | unsigned char *data | | Write data starting address |
| Variable (global) | Variable name | | Contents |
| | None | | — |
| Returned value | Type | Value | Meaning |
| | None | — | — |
| Function | Generate the record data written to the data flash. No processing is performed in this application note. Add processing based on the user system. | | |

| | | | |
|---|---|---|---|
| Declaration | void dtc_init(void) | | |
| Outline | DTC initial setting | | |
| Argument | Argument name | | Meaning |
| | None | | — |
| Variable (global) | Variable name | | Contents |
| | unsigned char command_data | | Command data |
| | unsigned char *write_addr | | Write address |
| | unsigned char record_data[RECORD_SIZE] | | Write data |
| Returned value | Type | Value | Meaning |
| | None | — | — |
| Function | Set DTC activation source. Perform initial setting before starting chain transfer or writing record. | | |

| | | | |
|---|---|---|---|
| Declaration | void data_write(unsigned char *data) | | |
| Outline | Writing | | |
| Argument | Argument name | | Meaning |
| | unsigned char *data | | Write data starting address |
| Variable (global) | Variable name | | Contents |
| | unsigned char block_select | | Block selected |
| | unsigned char *write_addr | | Write address |
| Returned value | Type | Value | Meaning |
| | None | — | — |
| Function | Write the first byte of data of the write address (write_addr) in CPU rewrite mode (EW1 mode). The flash ready status interrupt request is generated after the first byte of data is written | | |

| | | | |
|---|---|---|---|
| Declaration | unsigned char block_erase(unsigned char block_no) | | |
| Outline | Block erase | | |
| Argument | Argument name | | Meaning |
| | unsigned char block_no | | Number for erase blocks |
| Variable (global) | Variable name | | Contents |
| | unsigned char *ers_addr | | Erase address |
| | unsigned char flash_mode | | Flash mode |
| Returned value | Type | Value | Meaning |
| | unsigned char | ERASE_EXE | Erase executed |
| | | ERASE_UNEXE | Erase not executed |
| Function | Erase the specified block in CPU rewrite mode (EW1 mode). This process only triggers erasure and does not wait for the erase operation to be completed. | | |

| | | | |
|---|---|---|---|
| Declaration | unsigned char full_sts_chk(unsigned *chk_adr) | | |
| Outline | Full status check | | |
| Argument | Argument name | | Meaning |
| | unsigned *chk_adr | | Address where erase command or program command data is written |
| Variable (global) | Variable name | | Contents |
| | None | | — |
| Returned value | Type | Value | Meaning |
| | unsigned char | NORMAL | Completed normally |
| | | CMD_SEQ_ERROR | Command sequence error |
| | | ERS_BLK_CHK_ERROR | Erase/blank check error |
| | | PROGRAM_ERROR | Program error |
| Function | Perform full status check. | | |

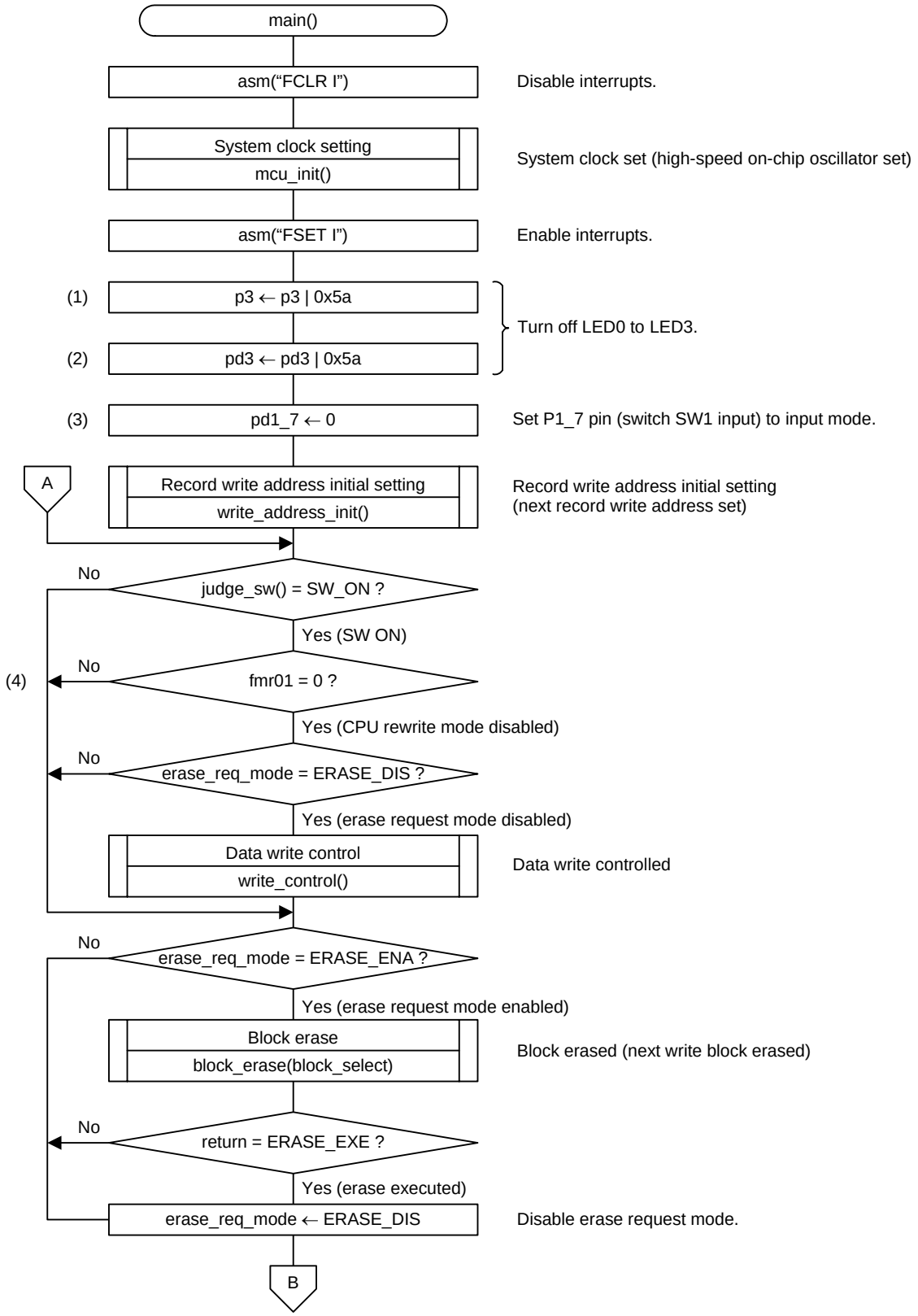
| | | | |
|---|---|---|---|
| Declaration | void _flash_memory_ready(void) | | |
| Outline | Flash memory ready interrupt | | |
| Argument | Argument name | | Meaning |
| | None | | — |
| Variable (global) | Variable name | | Contents |
| | unsigned char flash_mode | | Flash mode |
| | unsigned char *ers_addr | | Erase address |
| | unsigned char *write_addr | | Write address |
| | unsigned char status | | Full status check result |
| | unsigned char write_cnt | | Number of writes |
| | unsigned char block_select | | Block selected |
| | unsigned char erase_req_mode | | Erase request mode |
| Returned value | Type | Value | Meaning |
| | None | — | — |
| Function | An interrupt is generated when an erase operation is completed, writing by DTC is completed, or an error occurs during writing. Settings such as disabling data flash block rewrite and CPU write can be made here.<br>After writing data to the last record in each block, set the erase request of the next block to be written. Also, set the next block as the used block (block_select), and the starting address of this block as the write address (write_addr). | | |

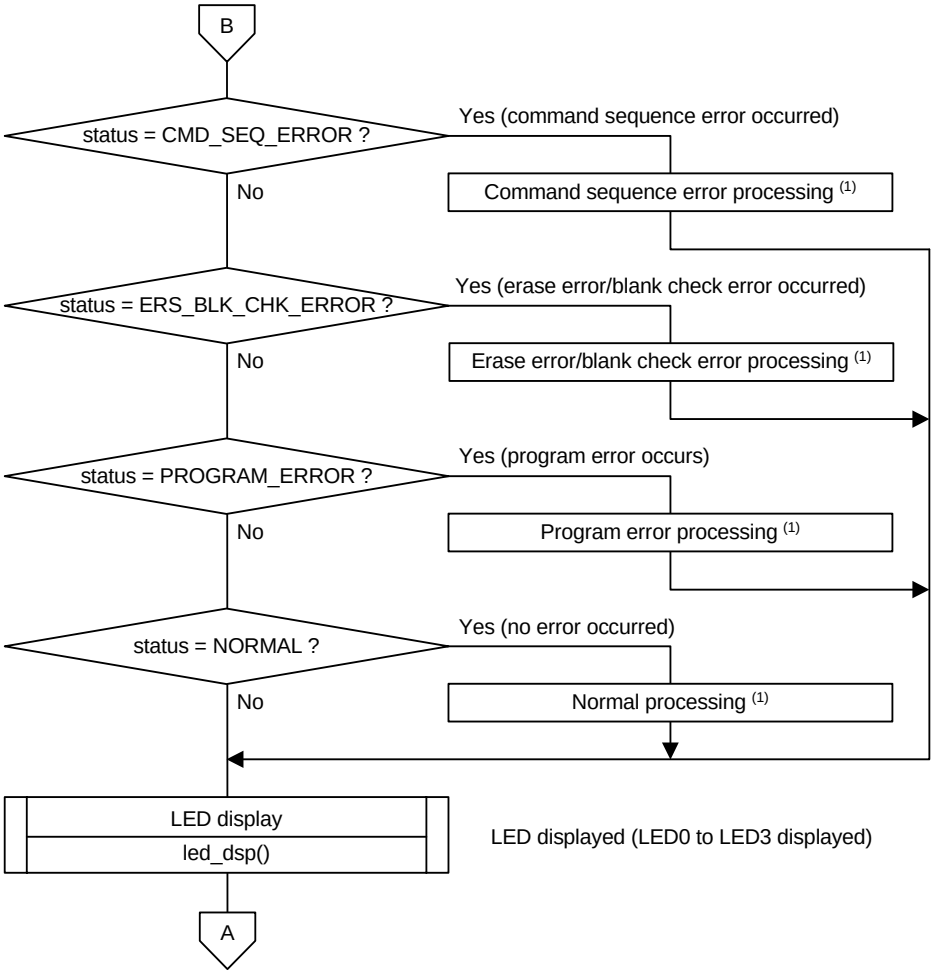
| | | | |
|---|---|---|---|
| Declaration | unsigned char judge_sw(void) | | |
| Outline | SW input judgment | | |
| Argument | Argument name | | Meaning |
| | None | | — |
| Variable (global) | Variable name | | Contents |
| | None | | — |
| Returned value | Type | Value | Meaning |
| | unsigned char | SW_ON | SW input |
| | | SW_OFF | No SW input |
| Function | Judge switch input and return the result. | | |

| | | | |
|---|---|---|---|
| Declaration | void led_dsp(void) | | |
| Outline | LED display | | |
| Argument | Argument name | | Meaning |
| | None | | — |
| Variable (global) | Variable name | | Contents |
| | unsigned char write_cnt | | Number of writes |
| Returned value | Type | Value | Meaning |
| | None | — | — |
| Function | The lower 4 bits of the number of data writes (write_cnt) are displayed on LED0 to LED3.<br>When bit 0 is 0, LED3 is off (high level). When bit 0 is 1, LED3 is on (low level).<br>When bit 1 is 0, LED2 is off (high level). When bit 1 is 1, LED2 is on (low level).<br>When bit 2 is 0, LED1 is off (high level). When bit 2 is 1, LED1 is on (low level).<br>When bit 3 is 0, LED0 is off (high level). When bit 3 is 1, LED0 is on (low level). | | |

## 4.2 Main Function

- Flowchart






Note:
1. These processes are not performed in this application note. Perform these processes based on the user system.

• Register settings

(1) Set P3_1 (LED0 output), P3_3 (LED1 output), P3_4 (LED2 output), and P3_6 (LED3 output) to high.

Port P3 Register (P3)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | x | 1 | x | 1 | 1 | x | 1 | x |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b1 | P3_1 | Port P3_1 bit | 1: “H” level | R/W |
| b3 | P3_3 | Port P3_3 bit | 1: “H” level | R/W |
| b4 | P3_4 | Port P3_4 bit | 1: “H” level | R/W |
| b6 | P3_6 | Port P3_6 bit | 1: “H” level | R/W |

(2) Set P3_1 (LED0 output), P3_3 (LED1 output), P3_4 (LED2 output), and P3_6 (LED3 output) as output ports.

Port P3 Direction Register (PD3)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | x | 1 | x | 1 | 1 | x | 1 | x |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b1 | PD3_1 | Port P3_1 direction bit | 1: Output mode (functions as output port) | R/W |
| b3 | PD3_3 | Port P3_3 direction bit | 1: Output mode (functions as output port) | R/W |
| b4 | PD3_4 | Port P3_4 direction bit | 1: Output mode (functions as output port) | R/W |
| b6 | PD3_6 | Port P3_6 direction bit | 1: Output mode (functions as output port) | R/W |

(3) Set P1_7 (switch SW1 input) as an input port.

Port P1 Direction Register (PD1)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | 0 | x | x | x | x | x | x | x |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b7 | PD1_7 | Port P1_7 direction bit | 0: Input mode (functions as input port) | R/W |

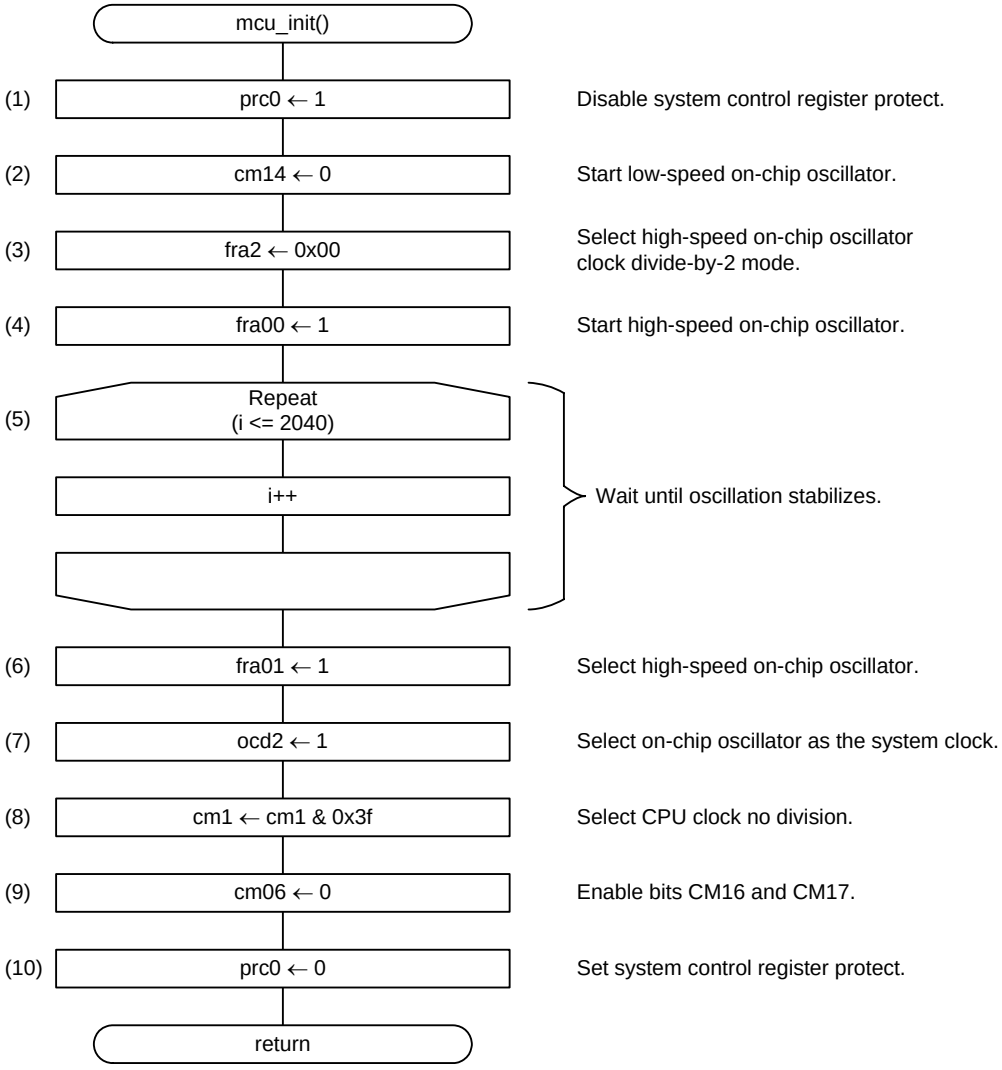
(4) Confirm that CPU rewrite mode is disabled (record write or block erase is completed).

Flash Memory Control Register 0 (FMR0)

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b1 | FMR01 | CPU rewrite mode select bit | 0: CPU rewrite mode disabled<br>1: CPU rewrite mode enabled | R/W |

## 4.3 System Clock Setting

- Flowchart



• Register settings

(1) Enable writing to registers CM0, CM1, CM3, OCD, FRA0, FRA1, FRA2, and FRA3.

Protect Register (PRCR)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | — | — | — | — | x | x | x | 1 |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b0 | PRC0 | Protect bit 0 | Enables writing to registers CM0, CM1, CM3, OCD, FRA0, FRA1, FRA2, and FRA3.<br>1: Write enabled | R/W |

(2) Start the low-speed on-chip oscillator.

System Clock Control Register 1 (CM1)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | | | — | 0 | x | x | x | x |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b4 | CM14 | Low-speed on-chip oscillator stop bit | 0: Low-speed on-chip oscillator on | R/W |

(3) Set the division ratio for the high-speed on-chip oscillator.

High-Speed On-Chip Oscillator Control Register 2 (FRA2)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | — | — | — | — | — | 0 | 0 | 0 |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b0 | FRA20 | High-speed on-chip oscillator frequency switching bit | Division selection<br>These bits select the division ratio for the high-speed on-chip oscillator clock.<br>b2 b1 b0<br>0 0 0: Divide-by-2 mode | R/W |
| b1 | FRA21 | | | R/W |
| b2 | FRA22 | | | R/W |

(4) Start the high-speed on-chip oscillator.

High-Speed On-Chip Oscillator Control Register 0 (FRA0)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | — | — | — | — | x | — | | 1 |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b0 | FRA00 | High-speed on-chip oscillator enable bit | 1: High-speed on-chip oscillator on | R/W |

(5) Wait until oscillation stabilizes.

(6) Select the high-speed on-chip oscillator.

High-Speed On-Chip Oscillator Control Register 0 (FRA0)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | — | — | — | — | x | — | 1 | |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b1 | FRA01 | High-speed on-chip oscillator select bit | 1: High-speed on-chip oscillator selected | R/W |

(7) Select the on-chip oscillator clock as the system clock.

Oscillation Stop Detection Register (OCD)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | — | — | — | — | x | 1 | x | x |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b2 | OCD2 | System clock select bit | 1: On-chip oscillator clock selected | R/W |

(8) Set CPU clock division select bit 1.

System Clock Control Register 1 (CM1)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | 0 | 0 | — | | x | x | x | x |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b6 | CM16 | CPU clock division select bit 1 | b7 b6<br>0 0: No division mode | R/W |
| b7 | CM17 | | | R/W |

(9) Set CPU clock division select bit 0.

System Clock Control Register 0 (CM0)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | x | 0 | x | x | x | x | — | — |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b6 | CM06 | CPU clock division select bit 0 | 0: Bits CM16 and CM17 in CM1 register enabled | R/W |

(10) Disable writing to registers CM0, CM1, CM3, OCD, FRA0, FRA1, FRA2, and FRA3.

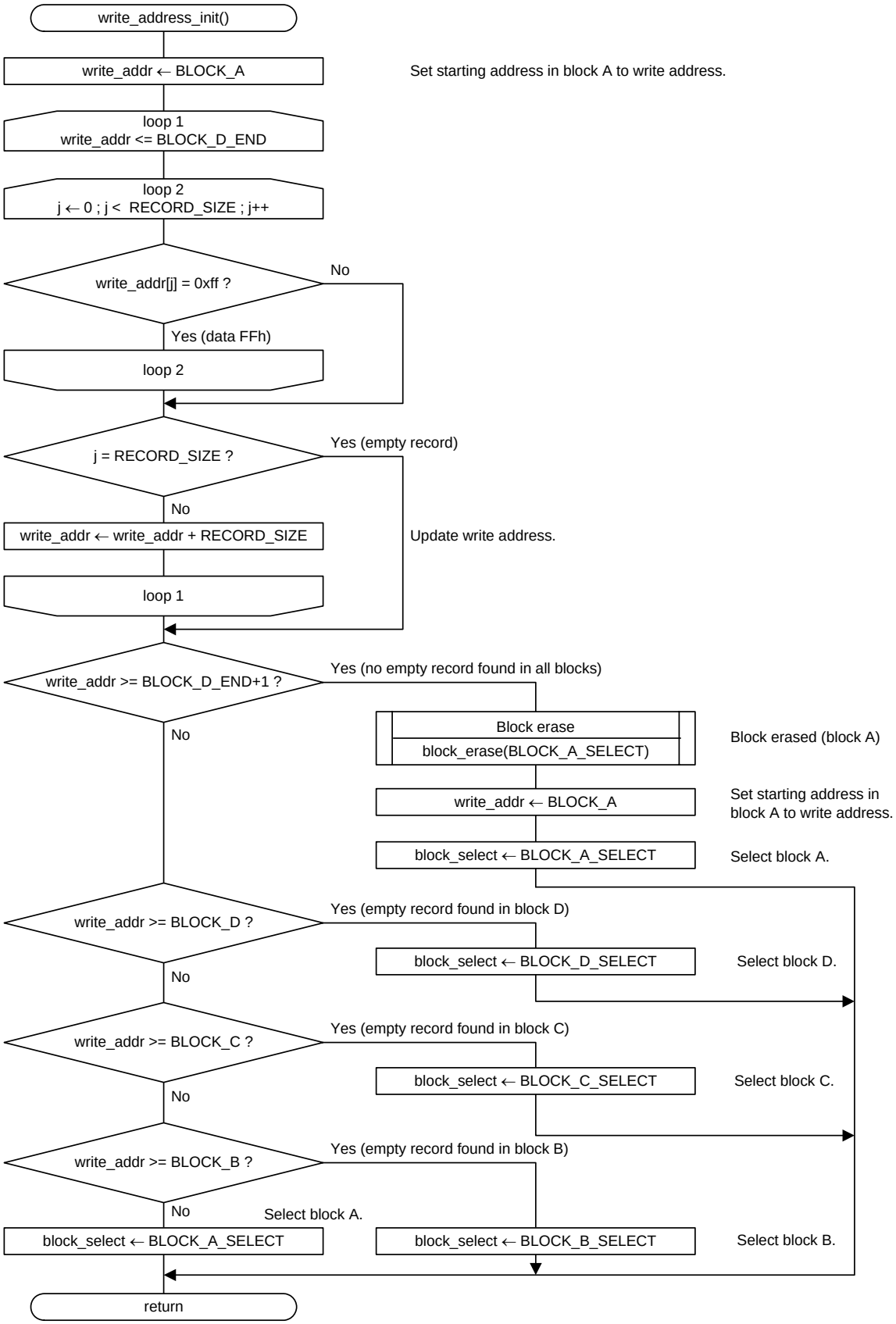
Protect Register (PRCR)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | — | — | — | — | x | x | x | 0 |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b0 | PRC0 | Protect bit 0 | Enables writing to registers CM0, CM1, CM3, OCD, FRA0, FRA1, FRA2, and FRA3.<br>0: Write disabled | R/W |

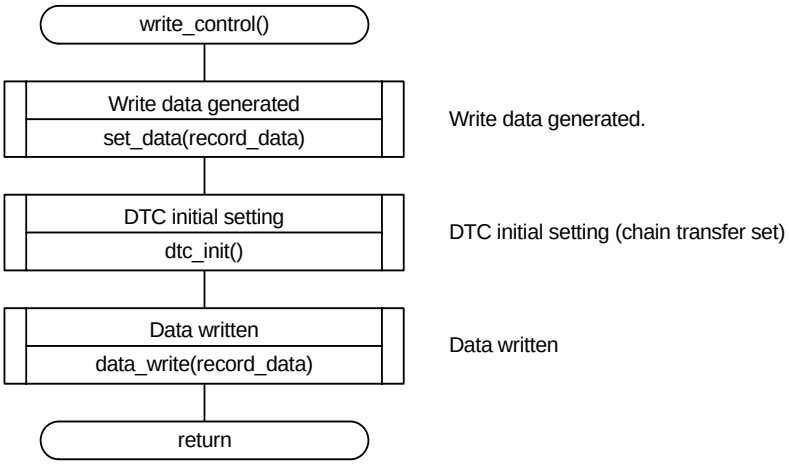
## 4.4 Record Write Address Initial Setting

- Flowchart



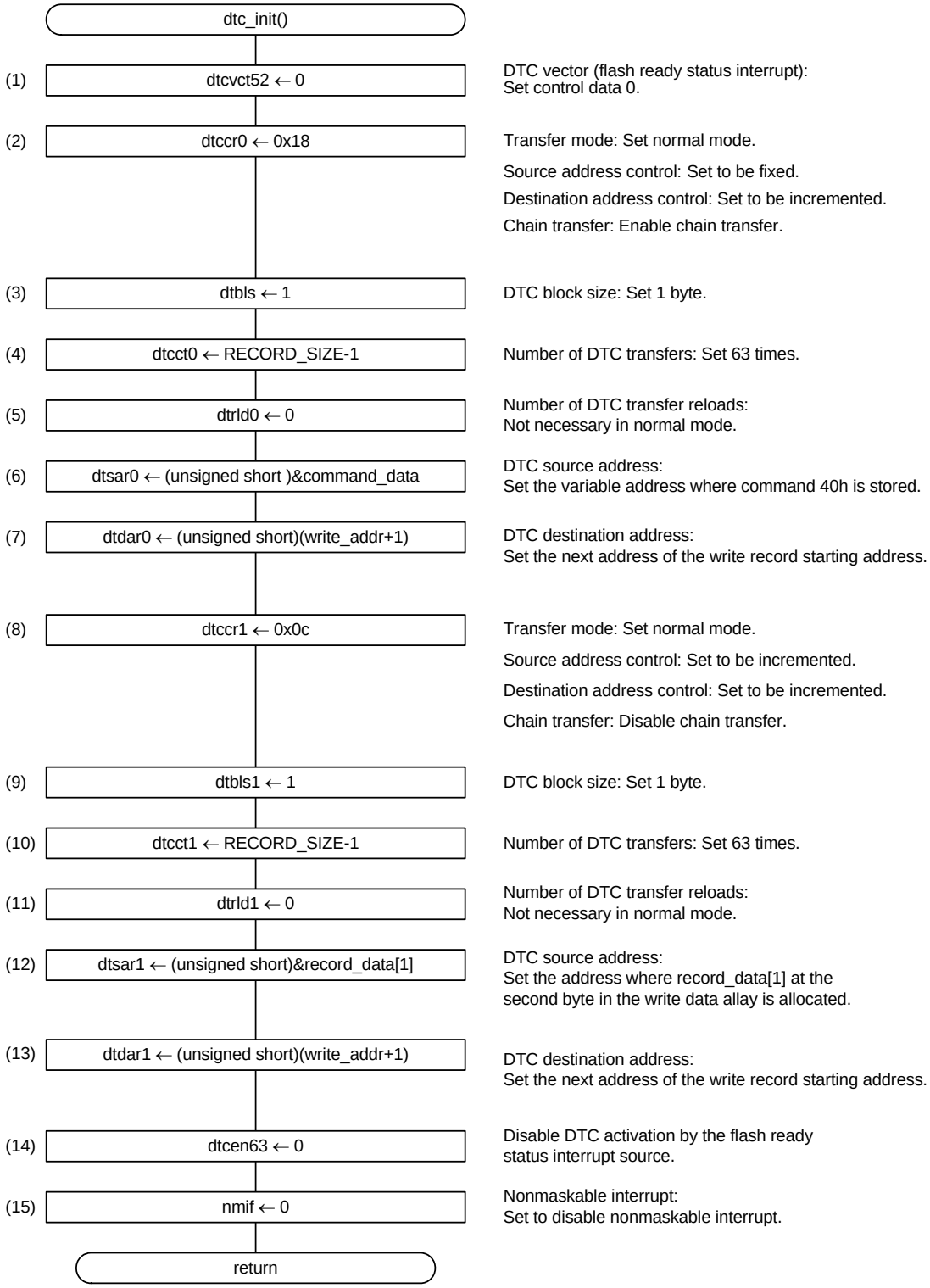
## 4.5 Data Write Control

- Flowchart



## 4.6 DTC Initial Setting

- Flowchart



• Register settings

(1) Set the DTC control data number to the DTC vector address (address 2C34h) where the flash ready status interrupt is allocated. In this program, set address 2C34h to 0 to use control data 0.

DTC Vector Address of Flash Ready Status Interrupt

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Function | Setting Range | R/W |
|---|---|---|---|
| b7 to b0 | These bits store the data from 00000000b to 00010111b. Select one of the 24 control data sets. | 00h to 17h | — |

(2) Set DTC control register 0 for control data 0. Set the transfer mode to normal mode, source address to fixed, destination address to increment, and chain transfer to enabled.

DTC Control Register 0 (DTCCR0)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | — | — | x | 1 | 1 | 0 | x | 0 |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b0 | MODE | Transfer mode select bit | 0: Normal mode | R/W |
| b2 | SAMOD | Source address control bit | 0: Fixed | R/W |
| b3 | DAMOD | Destination address control bit | 1: Incremented | R/W |
| b4 | CHNE | Chain transfer enable bit | 1: Chain transfers enabled | R/W |

(3) Set DTC block size register 0 for control data 0. In this program, set this register to 1 (01h) to transfer 1-byte data 63 times.

DTC Block Size Register 0 (DTBLS0)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

| Bit | Function | Setting Range | R/W |
|---|---|---|---|
| b7 to b0 | These bits specify the size of the data block to be transferred by one activation. | 00h to FFh | R/W |

(4) Set DTC transfer count register 0 for control data 0. In this program, set this register to 63 (3Fh) to transfer 1-byte data 63 times.

DTC Transfer Count Register 0 (DTCCT0)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | Function | Setting Range | R/W |
|---|---|---|---|
| b7 to b0 | These bits specify the number of times of DTC data transfers. | 00h to FFh | R/W |

(5) Set DTC transfer count reload register 0 for control data 0. As this register is not used in normal mode, set it to 0 (00h).

DTC Transfer Count Reload Register 0 (DTRLD0)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Function | Setting Range | R/W |
|---|---|---|---|
| b7 to b0 | This register value is reloaded to the DTCCT register in repeat mode. | 00h to FFh | R/W |

(6) Set DTC source address register 0 for control data 0. In this program, set the variable address where command 40h is stored.

DTC Source Address Register 0 (DTSAR0)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 |

| Bit | b15 | b14 | b13 | b12 | b11 | b10 | b9 | b8 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 |

| Bit | Function | Setting Range | R/W |
|---|---|---|---|
| b15 to b0 | These bits specify a transfer source address for data transfer. | 0000h to FFFFh | R/W |

(7) Set destination register 0 for control data 0. In this program, set the next address of the write record starting address.

DTC Destination Address Register 0 (DTDAR0)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 |

| Bit | b15 | b14 | b13 | b12 | b11 | b10 | b9 | b8 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 |

| Bit | Function | Setting Range | R/W |
|---|---|---|---|
| b15 to b0 | These bits specify a transfer destination address for data transfer. | 0000h to FFFFh | R/W |

(8) Set DTC control register 1 for control data 1. Set the transfer mode to normal mode, source address to increment, destination address to increment, and chain transfer to disabled.

DTC Control Register 1 (DTCCR1)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | — | — | x | 0 | 1 | 1 | x | 0 |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b0 | MODE | Transfer mode select bit | 0: Normal mode | R/W |
| b2 | SAMOD | Source address control bit | 1: Incremented | R/W |
| b3 | DAMOD | Destination address control bit | 1: Incremented | R/W |
| b4 | CHNE | Chain transfer enable bit | 1: Chain transfers disabled | R/W |

(9) Set DTC block size register 1 for control data 1. In this program, set this register to 1 (01h) to transfer 1-byte data 63 times.

DTC Block Size Register 1 (DTBLS1)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

| Bit | Function | Setting Range | R/W |
|---|---|---|---|
| b7 to b0 | These bits specify the size of the data block to be transferred by one activation. | 00h to FFh | R/W |

(10) Set DTC transfer count register 1 for control data 1. In this program, set this register to 63 (3Fh) to transfer 1-byte data 63 times.

DTC Transfer Count Register 1 (DTCCT1)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | Function | Setting Range | R/W |
|---|---|---|---|
| b7 to b0 | These bits specify the number of times of DTC data transfers. | 00h to FFh | R/W |

(11) Set DTC transfer count reload register 1 for control data 1. As this register is not used in normal mode, set it to 0 (00h).

DTC Transfer Count Reload Register 1 (DTRLD1)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Function | Setting Range | R/W |
|---|---|---|---|
| b7 to b0 | This register value is reloaded to the DTCCT register in repeat mode. | 00h to FFh | R/W |

(12) Set source address register 1 for control data 1. In this program, set the address where record_data[1] at the second byte in the write data array is allocated.

DTC Source Address Register 1 (DTSAR1)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 |
| Bit | b15 | b14 | b13 | b12 | b11 | b10 | b9 | b8 |
| Setting Value | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 |

| Bit | Function | Setting Range | R/W |
|---|---|---|---|
| b15 to b0 | These bits specify a transfer source address for data transfer. | 0000h to FFFFh | R/W |

(13) Set destination register 1 for control data 1. In this program, set the next address of the write record starting address.

DTC Destination Register 1 (DTDAR1)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 |
| Bit | b15 | b14 | b13 | b12 | b11 | b10 | b9 | b8 |
| Setting Value | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 |

| Bit | Function | Setting Range | R/W |
|---|---|---|---|
| b15 to b0 | These bits specify a transfer destination address for data transfer. | 0000h to FFFFh | R/W |

(14) Disable DTC activation by the flash ready status interrupt source.

DTC Activation Enable Register 6 (DTCEN6)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | — | x | — | x | 0 | — | — | — |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b3 | DTCEN63 | DTC activation enable bit by flash ready status interrupt source | 0: Activation disabled | R/W |

(15) Set the non-maskable interrupt generation bit to 0 (non-maskable interrupts not generated).

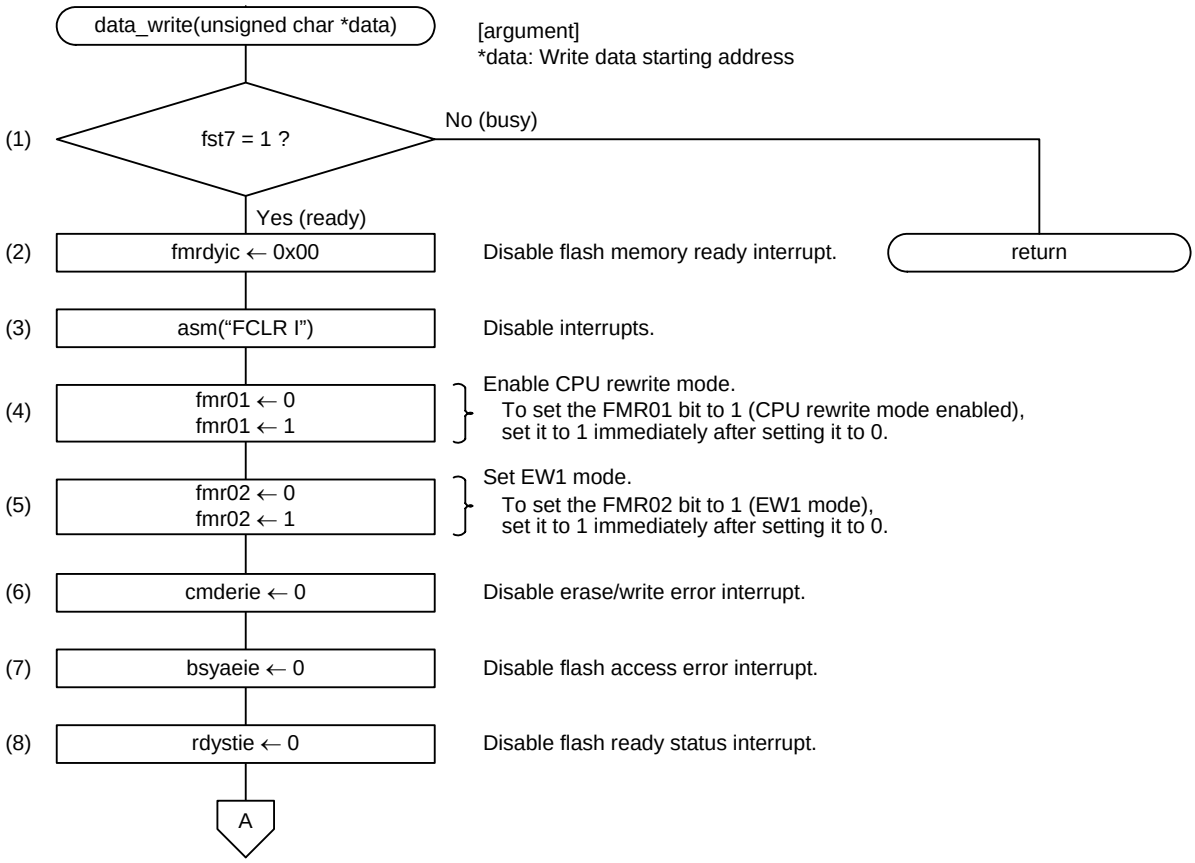
DTC Activation Control Register (DTCTL)

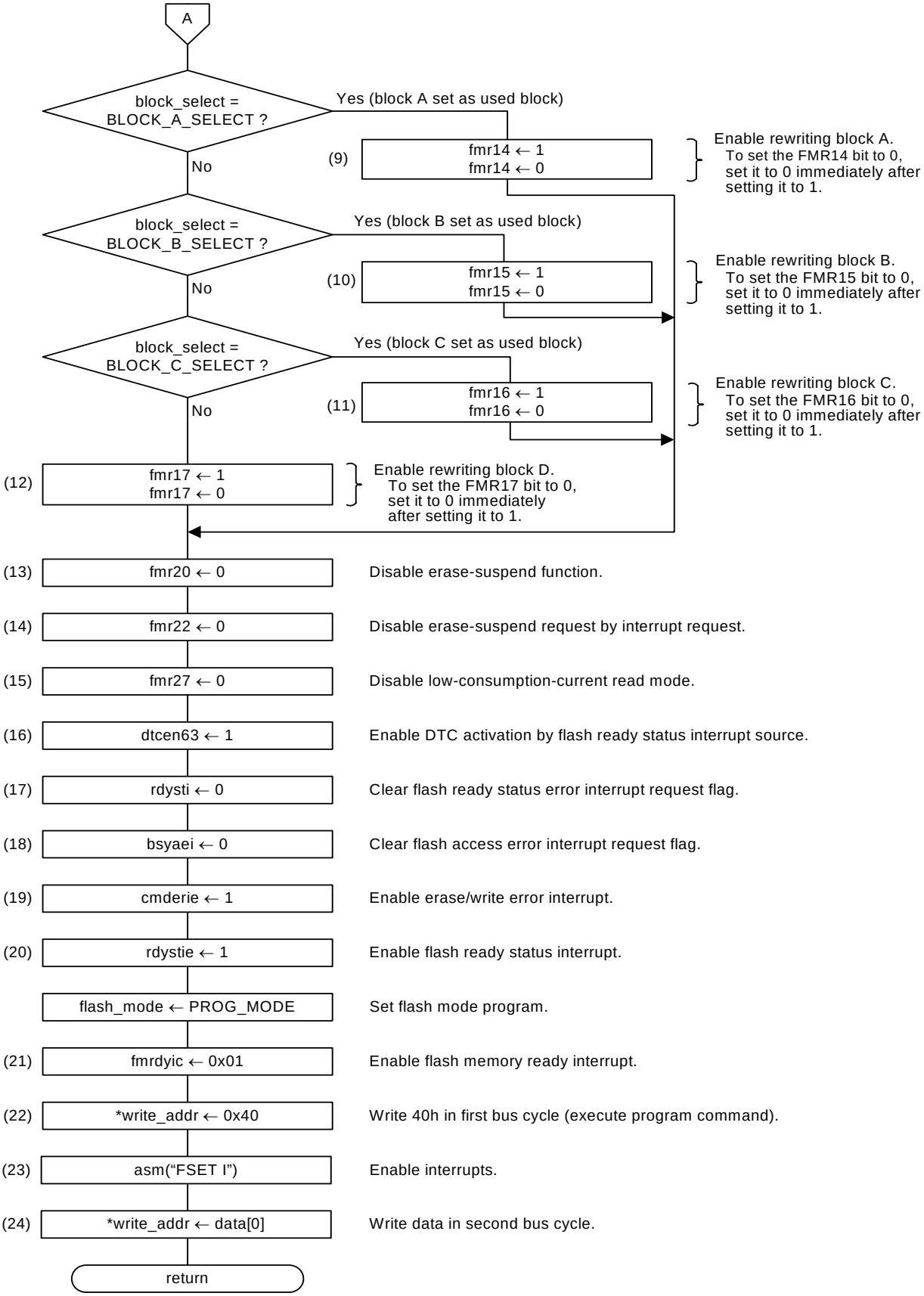
| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | — | — | — | — | — | — | 0 | — |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b1 | NMIF | Non-maskable interrupt generation bit | 0: Non-maskable interrupts not generated | R/W |

## 4.7 Data Written

• Flowchart




A
block_select = BLOCK_A_SELECT ?
Yes (block A set as used block)
No
(9)
fmr14 ← 1
fmr14 ← 0
Enable rewriting block A.
To set the FMR14 bit to 0, set it to 0 immediately after setting it to 1.
block_select = BLOCK_B_SELECT ?
Yes (block B set as used block)
No
(10)
fmr15 ← 1
fmr15 ← 0
Enable rewriting block B.
To set the FMR15 bit to 0, set it to 0 immediately after setting it to 1.
block_select = BLOCK_C_SELECT ?
Yes (block C set as used block)
No
(11)
fmr16 ← 1
fmr16 ← 0
Enable rewriting block C.
To set the FMR16 bit to 0, set it to 0 immediately after setting it to 1.
(12)
fmr17 ← 1
fmr17 ← 0
Enable rewriting block D.
To set the FMR17 bit to 0, set it to 0 immediately after setting it to 1.
(13)
fmr20 ← 0
Disable erase-suspend function.
(14)
fmr22 ← 0
Disable erase-suspend request by interrupt request.
(15)
fmr27 ← 0
Disable low-consumption-current read mode.
(16)
dtcen63 ← 1
Enable DTC activation by flash ready status interrupt source.
(17)
rdysti ← 0
Clear flash ready status error interrupt request flag.
(18)
bsyaei ← 0
Clear flash access error interrupt request flag.
(19)
cmderie ← 1
Enable erase/write error interrupt.
(20)
rdystie ← 1
Enable flash ready status interrupt.
flash_mode ← PROG_MODE
Set flash mode program.
(21)
fmrdyic ← 0x01
Enable flash memory ready interrupt.
(22)
*write_addr ← 0x40
Write 40h in first bus cycle (execute program command).
(23)
asm("FSET I")
Enable interrupts.
(24)
*write_addr ← data[0]
Write data in second bus cycle.
return

• Register settings

(1) Confirm that the write or erase operation is completed.

Flash Memory Status Register (FST)

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b7 | FST7 | Ready/busy status flag | 0: Busy<br>1: Ready | R |

(2) Disable the flash memory ready interrupt.

Interrupt Control Register (FMRDYIC)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | — | — | — | — | | 0 | 0 | 0 |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b0 | ILVL0 | Interrupt priority level select bit | b2 b1 b0<br>0 0 0: Level 0 (interrupt disabled) | R/W |
| b1 | ILVL1 | | | R/W |
| b2 | ILVL2 | | | R/W |
| b3 | IR | Interrupt request bit | 0: No interrupt requested<br>1: Interrupt requested | R |

(3) Clear the I flag to disable the interrupts.

(4) Enable CPU rewrite mode. When setting the FMR01 bit to 1, set it to 1 immediately after setting it to 0.

Flash Memory Control Register 0 (FMR0)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | | | | x | x | | 1 | — |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b1 | FMR01 | CPU rewrite mode select bit | 1: CPU rewrite mode enabled | R/W |

(5) Set EW1 mode. When setting the FMR02 bit to 1, set it to 1 immediately after setting it to 0.

Flash Memory Control Register 0 (FMR0)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | | | | x | x | 1 | | — |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b2 | FMR02 | EW1 mode select bit | 1: EW1 mode | R/W |

(6) Disable the erase/write error interrupt.

Flash Memory Control Register 0 (FMR0)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | | | 0 | x | x | | | — |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b5 | CMDERIE | Erase/write error interrupt enable bit | 0: Erase/write error interrupt disabled | R/W |

(7) Disable the flash access error interrupt.

Flash Memory Control Register 0 (FMR0)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | | 0 | | x | x | | | — |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b6 | BSYAEIE | Flash access error interrupt enable bit | 0: Flash access error interrupt disabled | R/W |

(8) Disable the flash ready status interrupt.

Flash Memory Control Register 0 (FMR0)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | 0 | | | x | x | | | — |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b7 | RDYSTIE | Flash ready status interrupt enable bit | 0: Flash ready status interrupt disabled | R/W |

(9) Enable rewriting of data flash block A when rewriting block A. When setting the FMR14 bit, set it to 0 immediately after setting it to 1.

Flash Memory Control Register 1 (FMR1)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | | | | 0 | x | — | — | — |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b4 | FMR14 | Data flash block A rewrite disable bit | 0: Rewrite enabled (software command acceptable) | R/W |

(10) Enable rewriting of data flash block B when rewriting block B. When setting the FMR15 bit, set it to 0 immediately after setting it to 1.

Flash Memory Control Register 1 (FMR1)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | | | 0 | | x | — | — | — |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b5 | FMR15 | Data flash block B rewrite disable bit | 0: Rewrite enabled (software command acceptable) | R/W |

(11) Enable rewriting of data flash block C when rewriting block C. When setting the FMR16 bit, set it to 0 immediately after setting it to 1.

Flash Memory Control Register 1 (FMR1)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | | 0 | | | x | — | — | — |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b6 | FMR16 | Data flash block C rewrite disable bit | 0: Rewrite enabled (software command acceptable) | R/W |

(12) Enable rewriting of data flash block D when rewriting block D. When setting the FMR17 bit, set it to 0 immediately after setting it to 1.

Flash Memory Control Register 1 (FMR1)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | 0 | | | | x | — | — | — |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b7 | FMR17 | Data flash block D rewrite disable bit | 0: Rewrite enabled (software command acceptable) | R/W |

(13) Disable the erase-suspend function.

Flash Memory Control Register 2 (FMR2)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | | — | — | — | — | | x | 0 |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b0 | FMR20 | Erase-suspend enable bit | 0: Erase-suspend disabled | R/W |

(14) Disable the erase-suspend request by an interrupt request.

Flash Memory Control Register 2 (FMR2)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | | — | — | — | — | 0 | x | |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b2 | FMR22 | Interrupt request suspend request enable bit | 0: Erase-suspend request disabled by interrupt request | R/W |

(15) Disable the low-consumption-current read mode.

Flash Memory Control Register 2 (FMR2)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | 0 | — | — | — | — | | x | |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b7 | FMR27 | Low-consumption-current read mode enable bit | 0: Low-consumption-current read mode disabled | R/W |

(16) Enable the DTC activation triggered by the flash ready status interrupt source.

DTC Activation Enable Register 6 (DTCEN6)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | x | x | x | x | 1 | x | x | x |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b3 | DTCEN63 | DTC activation by flash ready status interrupt source | 1: Activation enabled | R/W |

(17) Set no flash ready status interrupt request.

Flash Memory Status Register (FST)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | | x | | | — | x | | 0 |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b0 | RDYSTI | Flash ready status interrupt request flag | 0: No flash ready status interrupt request | R/W |

(18) Set no flash access error interrupt request.

Flash Memory Status Register (FST)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | | x | | | — | x | 0 | |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b1 | BSYAEI | Flash access error interrupt request flag | 0: No flash access error interrupt request | R/W |

(19) Enable the erase/write error interrupt.

Flash Memory Control Register 0 (FMR0)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | | | 1 | x | x | | | — |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b5 | CMDERIE | Erase/write error interrupt enable bit | 1: Erase/write error interrupt enabled | R/W |

(20) Enable the flash ready status interrupt.

Flash Memory Control Register 0 (FMR0)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | 1 | | | x | x | | | — |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b7 | RDYSTIE | Flash ready status interrupt enable bit | 1: Flash ready status interrupt enabled | R/W |

(21) Enable the flash memory ready interrupt.

Interrupt Control Register (FMRDYIC)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | — | — | — | — | | 0 | 0 | 1 |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b0 | ILVL0 | Interrupt priority level select bit | b2 b1 b0<br>0 0 1: Level 1 | R/W |
| b1 | ILVL1 | | | R/W |
| b2 | ILVL2 | | | R/W |
| b3 | IR | Interrupt request bit | 0: No interrupt requested<br>1: Interrupt requested | R |

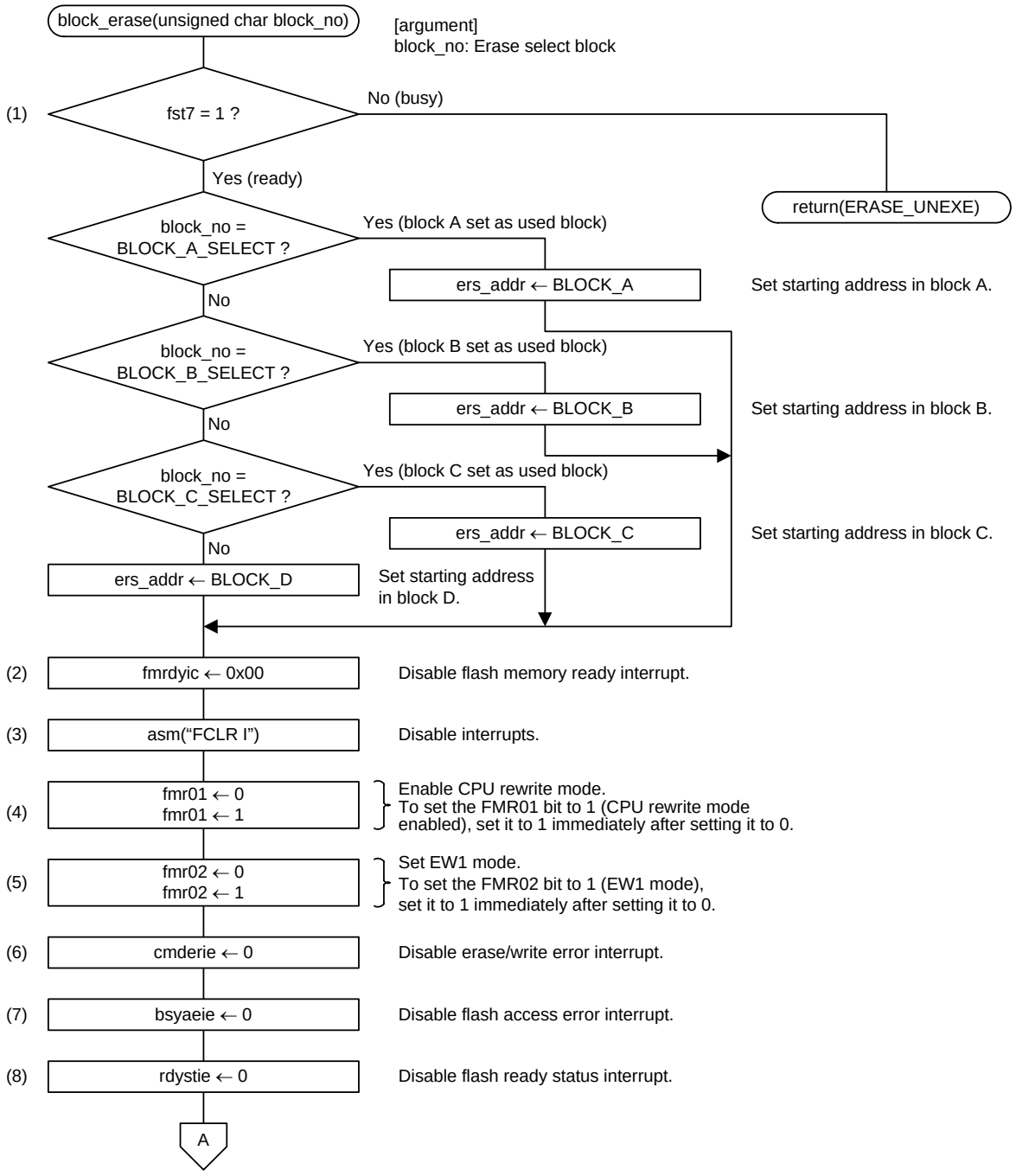
(22) Write program command 40h in the first bus cycle to the write address.

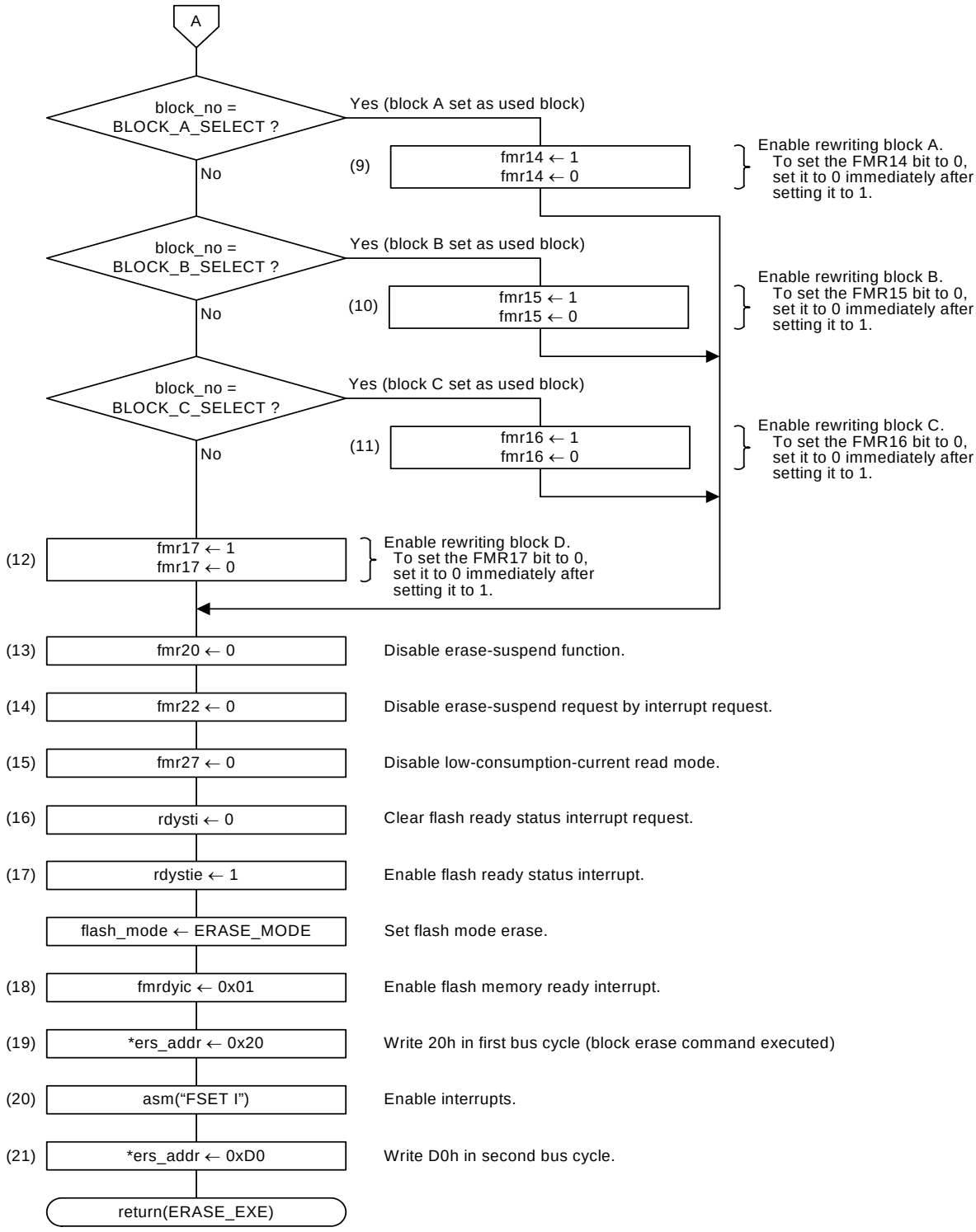
(23) Set the I flag to enable the interrupts.

(24) Writing (data written and verified) starts by writing data in the second bus cycle. Set the same address value in the second bus cycle as the address value specified in the first bus cycle.

## 4.8 Block Erase

• Flowchart




A
block_no = BLOCK_A_SELECT ?
Yes (block A set as used block)
No
(9)
fmr14 ← 1
fmr14 ← 0
Enable rewriting block A. To set the FMR14 bit to 0, set it to 0 immediately after setting it to 1.
block_no = BLOCK_B_SELECT ?
Yes (block B set as used block)
No
(10)
fmr15 ← 1
fmr15 ← 0
Enable rewriting block B. To set the FMR15 bit to 0, set it to 0 immediately after setting it to 1.
block_no = BLOCK_C_SELECT ?
Yes (block C set as used block)
No
(11)
fmr16 ← 1
fmr16 ← 0
Enable rewriting block C. To set the FMR16 bit to 0, set it to 0 immediately after setting it to 1.
(12)
fmr17 ← 1
fmr17 ← 0
Enable rewriting block D. To set the FMR17 bit to 0, set it to 0 immediately after setting it to 1.
(13)
fmr20 ← 0
Disable erase-suspend function.
(14)
fmr22 ← 0
Disable erase-suspend request by interrupt request.
(15)
fmr27 ← 0
Disable low-consumption-current read mode.
(16)
rdysti ← 0
Clear flash ready status interrupt request.
(17)
rdystie ← 1
Enable flash ready status interrupt.
flash_mode ← ERASE_MODE
Set flash mode erase.
(18)
fmrdyic ← 0x01
Enable flash memory ready interrupt.
(19)
*ers_addr ← 0x20
Write 20h in first bus cycle (block erase command executed)
(20)
asm(“FSET I”)
Enable interrupts.
(21)
*ers_addr ← 0xD0
Write D0h in second bus cycle.
return(ERASE_EXE)

• Register settings

(1) Confirm that the write or erase operation is completed.

Flash Memory Status Register (FST)

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b7 | FST7 | Ready/busy status flag | 0: Busy<br>1: Ready | R |

(2) Disable the flash memory ready interrupt.

Interrupt Control Register (FMRDYIC)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | — | — | — | — | | 0 | 0 | 0 |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b0 | ILVL0 | Interrupt priority level select bit | b2 b1 b0<br>0 0 0: Level 0 (interrupt disabled) | R/W |
| b1 | ILVL1 | | | R/W |
| b2 | ILVL2 | | | R/W |
| b3 | IR | Interrupt request bit | 0: No interrupt requested<br>1: Interrupt requested | R |

(3) Clear the I flag to disable the interrupts.

(4) Enable CPU rewrite mode. When setting the FMR01 bit to 1, set it to 1 immediately after setting it to 0.

Flash Memory Control Register 0 (FMR0)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | | | | x | x | | 1 | — |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b1 | FMR01 | CPU rewrite mode select bit | 1: CPU rewrite mode enabled | R/W |

(5) Set EW1 mode. When setting the FMR02 bit to 1, set it to 1 immediately after setting it to 0.

Flash Memory Control Register 0 (FMR0)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | | | | x | x | 1 | | — |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b2 | FMR02 | EW1 mode select bit | 1: EW1 mode | R/W |

(6) Disable the erase/write error interrupt.

Flash Memory Control Register 0 (FMR0)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | | | 0 | x | x | | | — |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b5 | CMDERIE | Erase/write error interrupt enable bit | 0: Erase/write error interrupt disabled | R/W |

(7) Disable the flash access error interrupt.

Flash Memory Control Register 0 (FMR0)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | | 0 | | x | x | | | — |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b6 | BSYAEIE | Flash access error interrupt enable bit | 0: Flash access error interrupt disabled | R/W |

(8) Disable the flash ready status interrupt.

Flash Memory Control Register 0 (FMR0)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | 0 | | | x | x | | | — |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b7 | RDYSTIE | Flash ready status interrupt enable bit | 0: Flash ready status interrupt disabled | R/W |

(9) Enable rewriting of data flash block A when erasing block A. When setting the FMR14 bit, set it to 0 immediately after setting it to 1.

Flash Memory Control Register 1 (FMR1)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | | | | 0 | x | — | — | — |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b4 | FMR14 | Data flash block A rewrite disable bit | 0: Rewrite enabled (software command acceptable) | R/W |

(10) Enable rewriting of data flash block B when erasing block B. When setting the FMR15 bit, set it to 0 immediately after setting it to 1.

Flash Memory Control Register 1 (FMR1)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | | | 0 | | x | — | — | — |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b5 | FMR15 | Data flash block B rewrite disable bit | 0: Rewrite enabled (software command acceptable) | R/W |

(11) Enable rewriting of data flash block C when erasing block C. When setting the FMR16 bit, set it to 0 immediately after setting it to 1.

Flash Memory Control Register 1 (FMR1)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | | 0 | | | x | — | — | — |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b6 | FMR16 | Data flash block C rewrite disable bit | 0: Rewrite enabled (software command acceptable) | R/W |

(12) Enable rewriting of data flash block D when erasing block D. When setting the FMR17 bit, set it to 0 immediately after setting it to 1.

Flash Memory Control Register 1 (FMR1)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | 0 | | | | x | — | — | — |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b7 | FMR17 | Data flash block D rewrite disable bit | 0: Rewrite enabled (software command acceptable) | R/W |

(13) Disable the erase-suspend function.

Flash Memory Control Register 2 (FMR2)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | | — | — | — | — | | x | 0 |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b0 | FMR20 | Erase-suspend enable bit | 0: Erase-suspend disabled | R/W |

(14) Disable the interrupt request suspend request.

Flash Memory Control Register 2 (FMR2)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | | — | — | — | — | 0 | x | |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b2 | FMR22 | Interrupt request suspend request enable bit | 0: Erase-suspend request disabled by interrupt request | R/W |

(15) Disable low-consumption-current read mode.

Flash Memory Control Register 2 (FMR2)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | 0 | — | — | — | — | | x | |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b7 | FMR27 | Low-consumption-current read mode enable bit | 0: Low-consumption-current read mode disabled | R/W |

(16) Set no flash ready status interrupt request.

Flash Memory Status Register (FST)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Symbol | | x | | | — | x | | 0 |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b0 | RDYSTI | Flash ready status interrupt request flag | 0: No flash ready status interrupt request | R/W |

(17) Enable the flash ready status interrupt.

Flash Memory Control Register 0 (FMR0)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | 1 | | | x | x | | | — |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b7 | RDYSTIE | Flash ready status interrupt enable bit | 1: Flash ready status interrupt enabled | R/W |

(18) Enable the flash memory ready interrupt.

Interrupt Control Register (FMRDYIC)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | — | — | — | — | | 0 | 0 | 1 |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b0 | ILVL0 | Interrupt priority level select bit | b2 b1 b0<br>0 0 1: Level 1 | R/W |
| b1 | ILVL1 | | | R/W |
| b2 | ILVL2 | | | R/W |
| b3 | IR | Interrupt request bit | 0: No interrupt requested<br>1: Interrupt requested | R |

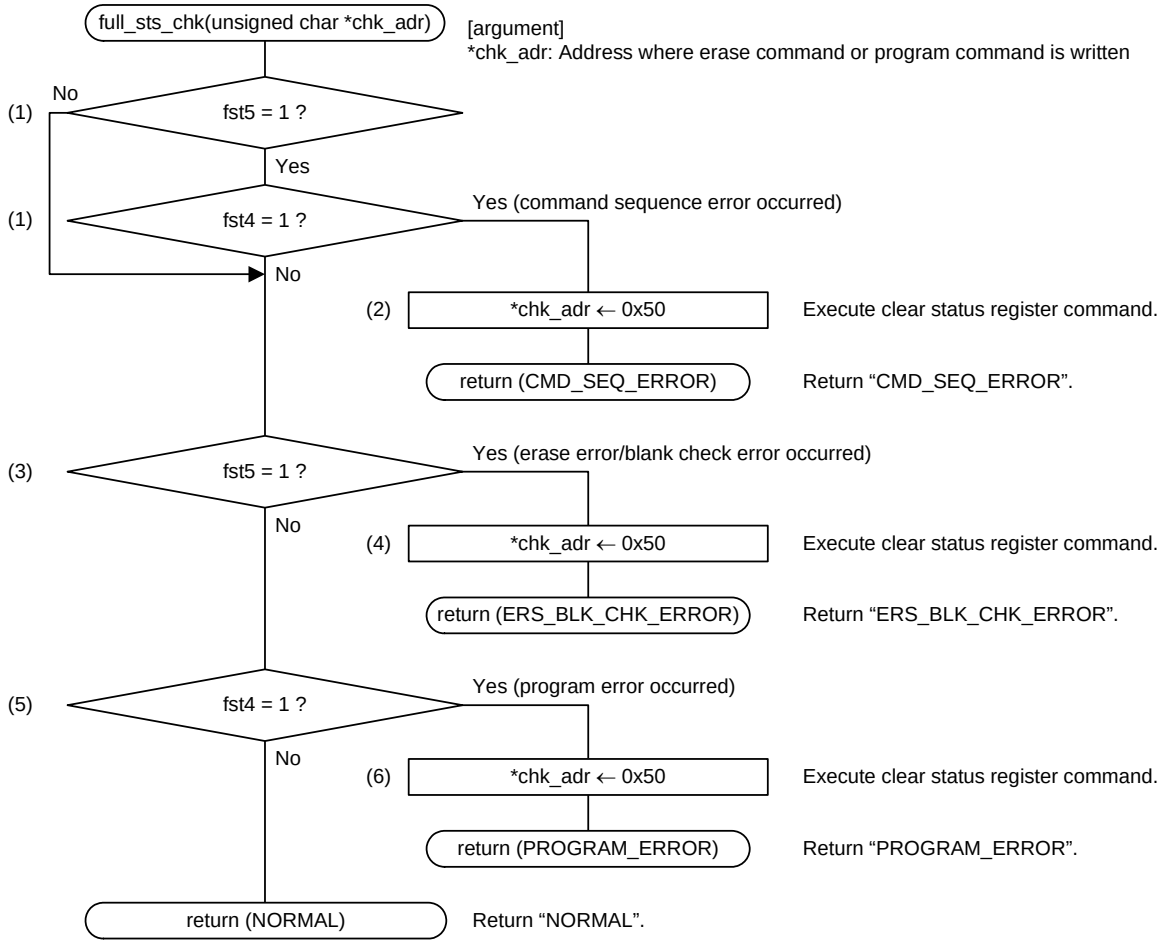
(19) Write block erase command 20h to a given address in the block to be erased in the first bus cycle.

(20) Set the I flag to enable the interrupts.

(21) Erasure (erase and erase verify) starts by writing confirmation command D0h in the second bus cycle.

## 4.9 Full Status Check

• Flowchart



• Register settings

(1) Confirm that a command sequence error occurs by reading bits FST4 and FST5 in the FST register.

Flash Memory Status Register (FST)

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b4 | FST4 | Program error flag | 0: No program error<br>1: Program error | R |
| b5 | FST5 | Erase error/blank check error flag | 0: No erase error/blank check error<br>1: Erase error/blank check error | R |

(2) Write clear status register command 50h to the address where erase command 20h or program command 40h was written when a program error (FST4 = 1) and an erase error (FST5 = 1) occur.

(3) Confirm that an erase error/blank check error occurs by reading the FST5 bit in the FST register.

Flash Memory Status Register (FST)

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b5 | FST5 | Erase error/blank check error flag | 0: No erase error/blank check error<br>1: Erase error/blank check error | R |

(4) Write clear status register command 50h to the address where erase command 20h was written when an erase error (FST5 = 1) occurs.

(5) Confirm that a program error occurs by reading the FST4 bit in the FST register.

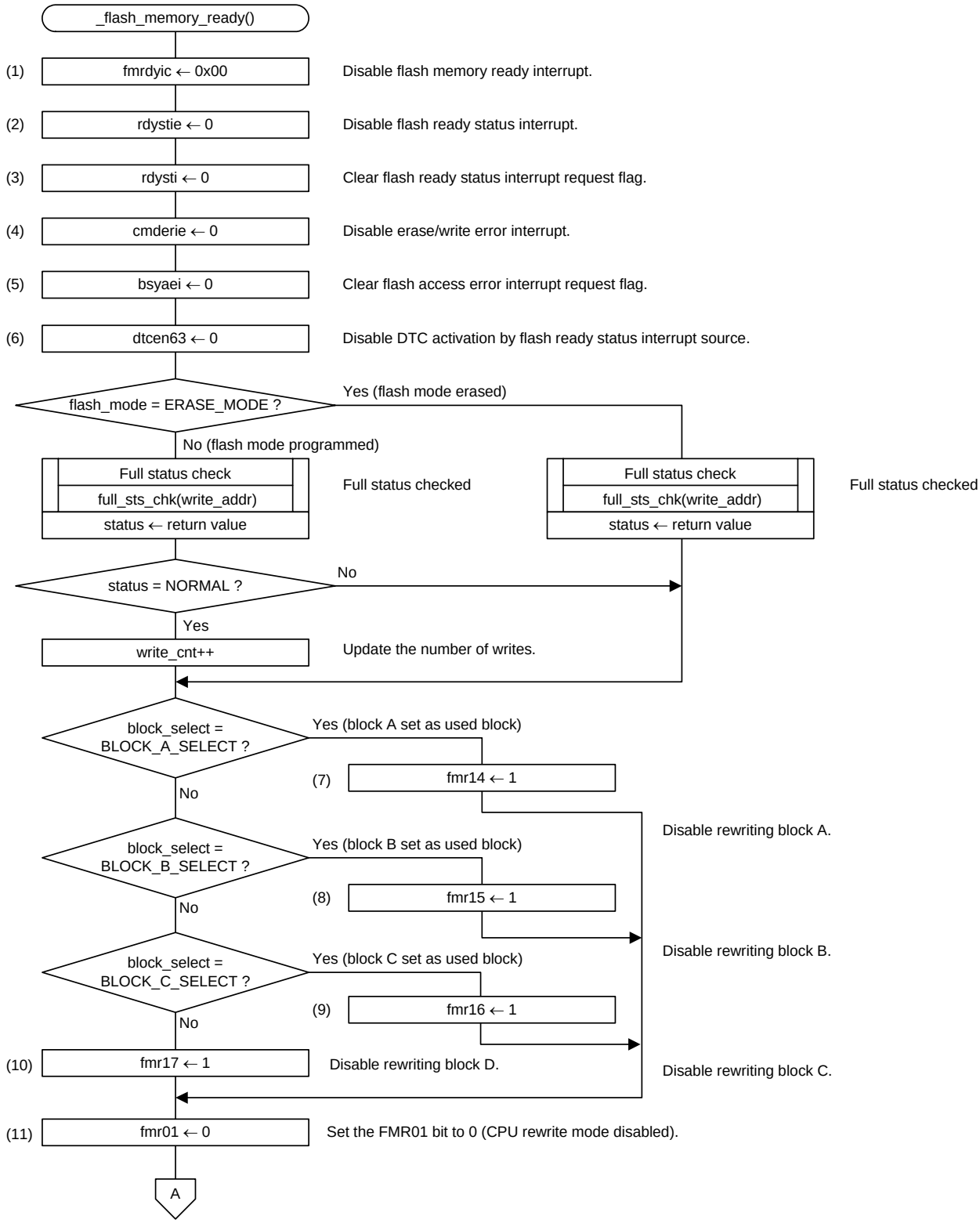
Flash Memory Status Register (FST)

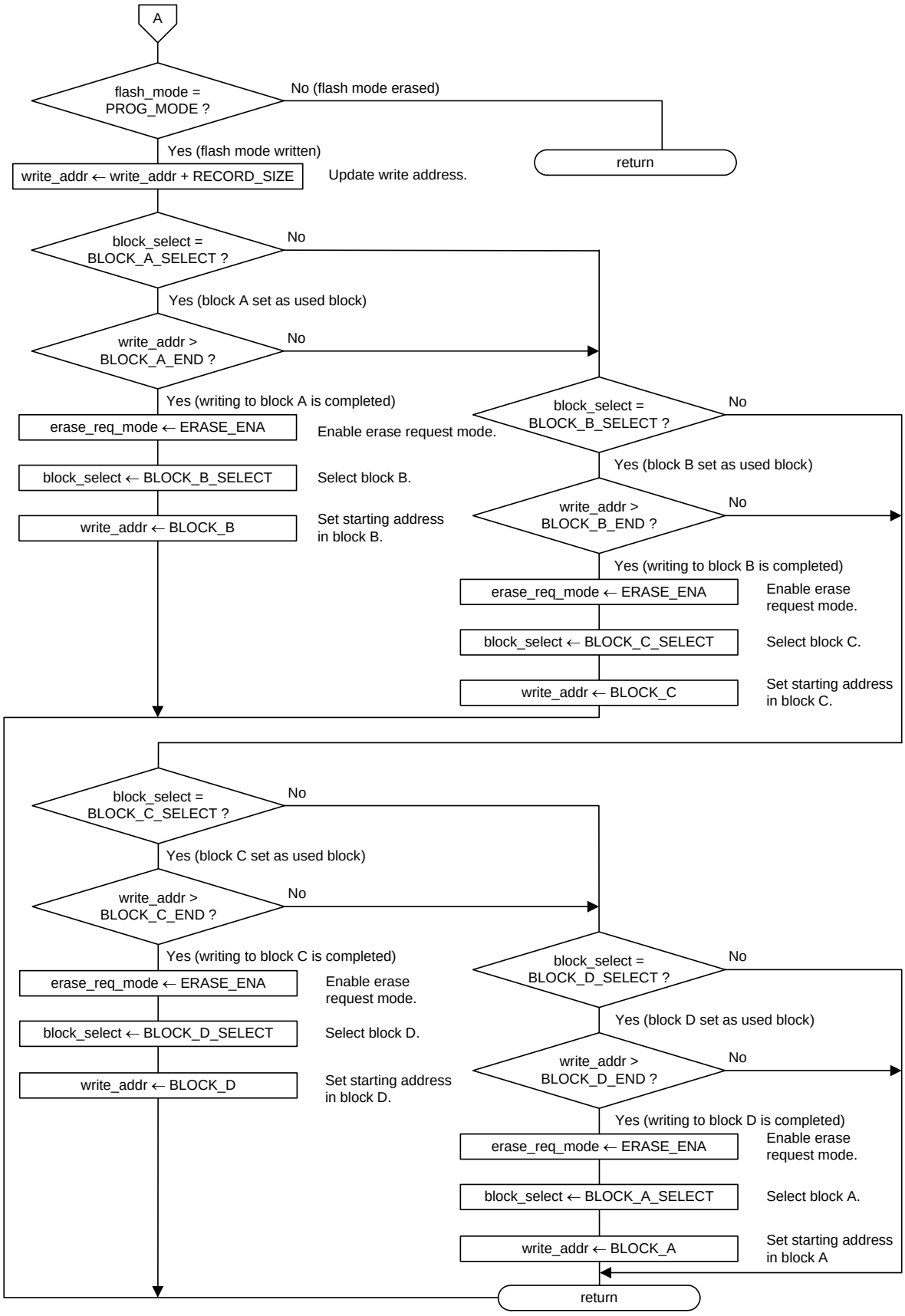
| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b4 | FST4 | Program error flag | 0: No program error<br>1: Program error | R |

(6) Write clear status register command 50h to the address where program command 40h was written when a program error (FST4 = 1) occurs.

## 4.10 Flash Memory Ready Interrupt

• Flowchart




A
flash_mode = PROG_MODE ?
No (flash mode erased)
return
Yes (flash mode written)
write_addr ← write_addr + RECORD_SIZE
Update write address.
block_select = BLOCK_A_SELECT ?
No
Yes (block A set as used block)
write_addr > BLOCK_A_END ?
No
Yes (writing to block A is completed)
erase_req_mode ← ERASE_ENA
Enable erase request mode.
block_select ← BLOCK_B_SELECT
Select block B.
write_addr ← BLOCK_B
Set starting address in block B.
block_select = BLOCK_B_SELECT ?
No
Yes (block B set as used block)
write_addr > BLOCK_B_END ?
No
Yes (writing to block B is completed)
erase_req_mode ← ERASE_ENA
Enable erase request mode.
block_select ← BLOCK_C_SELECT
Select block C.
write_addr ← BLOCK_C
Set starting address in block C.
block_select = BLOCK_C_SELECT ?
No
Yes (block C set as used block)
write_addr > BLOCK_C_END ?
No
Yes (writing to block C is completed)
erase_req_mode ← ERASE_ENA
Enable erase request mode.
block_select ← BLOCK_D_SELECT
Select block D.
write_addr ← BLOCK_D
Set starting address in block D.
block_select = BLOCK_D_SELECT ?
No
Yes (block D set as used block)
write_addr > BLOCK_D_END ?
No
Yes (writing to block D is completed)
erase_req_mode ← ERASE_ENA
Enable erase request mode.
block_select ← BLOCK_A_SELECT
Select block A.
write_addr ← BLOCK_A
Set starting address in block A
return

• Register settings

(1) Disable the flash memory ready interrupt.

Interrupt Control Register (FMRDYIC)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | — | — | — | — | 0 | 0 | 0 | 0 |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b0 | ILVL0 | Interrupt priority level select bit | b2 b1 b0<br>0 0 0: Level 0 (interrupt disabled) | R/W |
| b1 | ILVL1 | | | R/W |
| b2 | ILVL2 | | | R/W |
| b3 | IR | Interrupt request bit | 0: No interrupt requested | R |

(2) Disable the flash ready status interrupt.

Flash Memory Control Register 0 (FMR0)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | 0 | | | x | x | | | — |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b7 | RDYSTIE | Flash ready status interrupt enable bit | 0: Flash ready status interrupt disabled | R/W |

(3) Set no flash ready status interrupt request.

Flash Memory Status Register (FST)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Symbol | | x | | | — | x | | 0 |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b0 | RDYSTI | Flash ready status interrupt request flag | 0: No flash ready status interrupt request | R/W |

(4) Disable the erase/write error interrupt.

Flash Memory Control Register 0 (FMR0)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | | | 0 | x | x | | | — |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b5 | CMDERIE | Erase/write error interrupt enable bit | 0: Erase/write error interrupt disabled | R/W |

(5) Set no flash access error interrupt request.

Flash Memory Status Register (FST)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Symbol | | x | | | — | x | 0 | |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b1 | BSYAEI | Flash access error interrupt request flag | 0: No flash access error interrupt request | R/W |

(6) Disable the DTC activation by the flash ready status interrupt source.

DTC Activation Enable Register (DTCEN6)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | x | x | x | x | 0 | x | x | x |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b3 | DTCEN63 | DTC activation enable bit by flash ready status interrupt source | 0: Activation disabled | R/W |

(7) Disable rewriting data flash block A.

Flash Memory Control Register 1 (FMR1)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | | | | 1 | x | — | — | — |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b4 | FMR14 | Data flash block A rewrite disable bit | 1: Rewrite disabled (software command not acceptable, no error occurred) | R/W |

(8) Disable rewriting data flash block B.

Flash Memory Control Register 1 (FMR1)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | | | 1 | | x | — | — | — |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b5 | FMR15 | Data flash block B rewrite disable bit | 1: Rewrite disabled (software command not acceptable, no error occurred) | R/W |

(9) Disable rewriting data flash block C.

Flash Memory Control Register 1 (FMR1)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | | 1 | | | x | — | — | — |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b6 | FMR16 | Data flash block C rewrite disable bit | 1: Rewrite disabled (software command not acceptable, no error occurred) | R/W |

(10) Disable rewriting data flash block D.

Flash Memory Control Register 1 (FMR1)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | 1 | | | | x | — | — | — |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b7 | FMR17 | Data flash block D rewrite disable bit | 1: Rewrite disabled (software command not acceptable, no error occurred) | R/W |

(11) Disable the CPU rewrite mode.

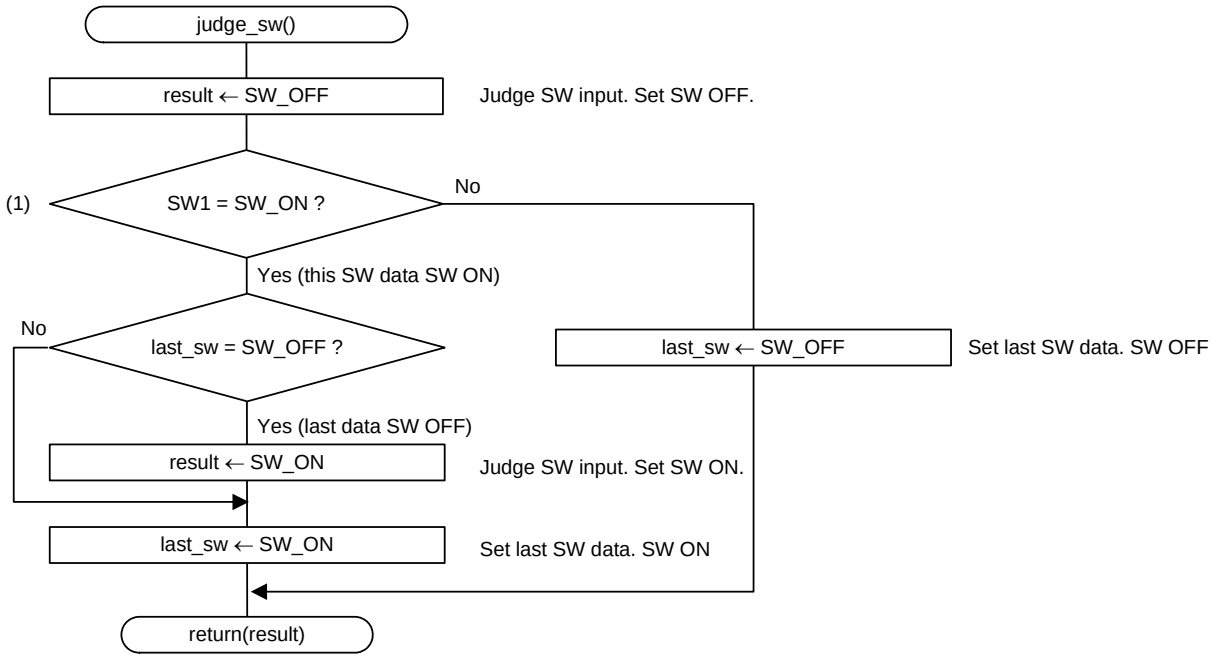
Flash Memory Control Register 0 (FMR0)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | | | | x | x | | 0 | — |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b1 | FMR01 | CPU rewrite mode select bit | 0: CPU rewrite mode disabled | R/W |

## 4.11 SW Input Judgment

• Flowchart




• Register settings

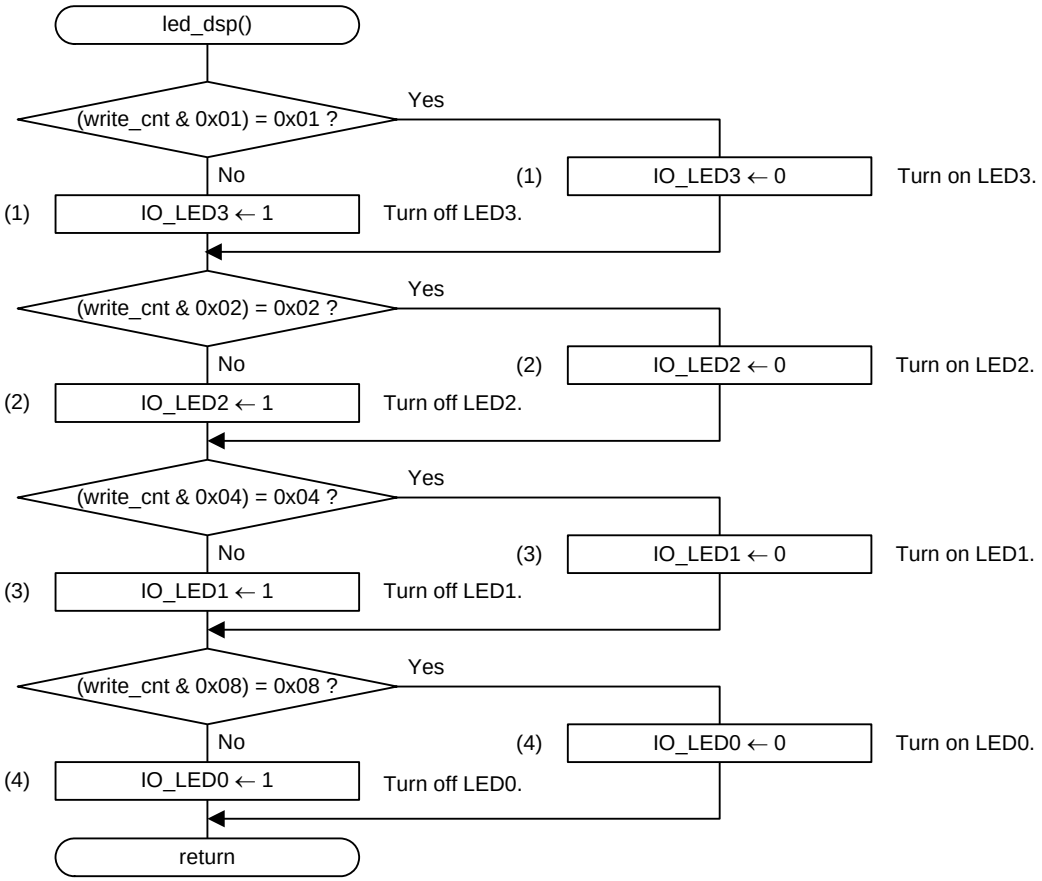
(1) Confirm that P1_7 is low.

Port P1 Register (P1)

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b7 | P1_7 | Port P1_7 bit | 0: “L” level<br>1: “H” level | R/W |

## 4.12 LED Display

• Flowchart




• Register settings

(1) Set P3_6 (LED3 output) to high or low depending on the bit 0 value of the number of writes (write_cnt).

Port P3 Register (P3)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | x | 0/1 | x | | | x | | x |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b6 | P3_6 | Port P3_6 bit | 0: “L” level<br>1: “H” level | R/W |

(2) Set P3_4 (LED2 output) to high or low depending on the bit 1 value of the number of writes (write_cnt).

Port P3 Register (P3)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | x | | x | 0/1 | | x | | x |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b4 | P3_4 | Port P3_4 bit | 0: “L” level<br>1: “H” level | R/W |

(3) Set P3_3 (LED1 output) to high or low depending on the bit 2 value of the number of writes (write_cnt).

Port P3 Register (P3)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | x | | x | | 0/1 | x | | x |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b3 | P3_3 | Port P3_3 bit | 0: “L” level<br>1: “H” level | R/W |

(4) Set P3_1 (LED0 output) to high or low depending on the bit 3 value of the number of writes (write_cnt).

Port P3 Register (P3)

| Bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| Setting Value | x | | x | | | x | 0/1 | x |

| Bit | Symbol | Bit Name | Function | R/W |
|---|---|---|---|---|
| b1 | P3_1 | Port P3_1 bit | 0: “L” level<br>1: “H” level | R/W |

# 5. Sample Program

A sample program can be downloaded from the Renesas Electronics website.
To download, click “Application Notes” in the left-hand side menu of the R8C/Tiny Family page.

# 6. Reference Documents

Hardware Manual
R8C/35C Group User’s Manual: HardwareRev.1.00
The latest version can be downloaded from the Renesas Electronics website.

Technical Update/Technical News
The latest information can be downloaded from the Renesas Electronics website.

# Website and Support

Renesas Electronics website
http://www.renesas.com/

Inquiries
http://www.renesas.com/inquiry

| Revision History | R8C/35C Group<br>Rewriting the Data Flash with DTC |
|---|---|

| Rev. | Date | Description | |
|---|---|---|---|
| | | Page | Summary |
| 1.00 | Apr. 21, 2010 | – | First Edition issued |
| | | | |

# General Precautions in the Handling of MPU/MCU Products

The following usage notes are applicable to all MPU/MCU products from Renesas. For detailed usage notes on the products covered by this manual, refer to the relevant sections of the manual. If the descriptions under General Precautions in the Handling of MPU/MCU Products and in the body of the manual differ from each other, the description in the body of the manual takes precedence.

1. Handling of Unused Pins

   Handle unused pins in accord with the directions given under Handling of Unused Pins in the manual.

   — The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of LSI, an associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible. Unused pins should be handled as described under Handling of Unused Pins in the manual.

2. Processing at Power-on

   The state of the product is undefined at the moment when power is supplied.

   — The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the moment when power is supplied.

   In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the moment when power is supplied until the reset process is completed.

   In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the moment when power is supplied until the power reaches the level at which resetting has been specified.

3. Prohibition of Access to Reserved Addresses

   Access to reserved addresses is prohibited.

   — The reserved addresses are provided for the possible future expansion of functions. Do not access these addresses; the correct operation of LSI is not guaranteed if they are accessed.

4. Clock Signals

   After applying a reset, only release the reset line after the operating clock signal has become stable. When switching the clock signal during program execution, wait until the target clock signal has stabilized.

   — When the clock signal is generated with an external resonator (or from an external oscillator) during a reset, ensure that the reset line is only released after full stabilization of the clock signal. Moreover, when switching to a clock signal produced with an external resonator (or by an external oscillator) while program execution is in progress, wait until the target clock signal is stable.

5. Differences between Products

   Before changing from one product to another, i.e. to one with a different part number, confirm that the change will not lead to problems.

   — The characteristics of MPU/MCU in the same group but having different part numbers may differ because of the differences in internal memory capacity and layout pattern. When changing to products of different part numbers, implement a system-evaluation test for each of the products.

# Notice

1. All information included in this document is current as of the date this document is issued. Such information, however, is subject to change without any prior notice. Before purchasing or using any Renesas Electronics products listed herein, please confirm the latest product information with a Renesas Electronics sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas Electronics such as that disclosed through our website.
2. Renesas Electronics does not assume any liability for infringement of patents, copyrights, or other intellectual property rights of third parties by or arising from the use of Renesas Electronics products or technical information described in this document. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
3. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part.
4. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
5. When exporting the products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations. You should not use Renesas Electronics products or the technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. Renesas Electronics products and technology may not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations.
6. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
7. Renesas Electronics products are classified according to the following three quality grades: "Standard", "High Quality", and "Specific". The recommended applications for each Renesas Electronics product depends on the product's quality grade, as indicated below. You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application categorized as "Specific" without the prior written consent of Renesas Electronics. Further, you may not use any Renesas Electronics product for any application for which it is not intended without the prior written consent of Renesas Electronics. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for an application categorized as "Specific" or for which the product is not intended where you have failed to obtain the prior written consent of Renesas Electronics. The quality grade of each Renesas Electronics product is "Standard" unless otherwise expressly specified in a Renesas Electronics data sheets or data books, etc.
   - "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; and industrial robots.
   - "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; safety equipment; and medical equipment not specifically designed for life support.
   - "Specific": Aircraft; aerospace equipment; submersible repeaters; nuclear reactor control systems; medical equipment or systems for life support (e.g. artificial life support devices or systems), surgical implantations, or healthcare intervention (e.g. excision, etc.), and any other applications or purposes that pose a direct threat to human life.
8. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or system manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.

(Note 1) "Renesas Electronics" as used in this document means Renesas Electronics Corporation and also includes its majority-owned subsidiaries.

(Note 2) "Renesas Electronics product(s)" means any product developed or manufactured by or for Renesas Electronics.

RENESAS

**SALES OFFICES** Renesas Electronics Corporation http://www.renesas.com

Refer to "http://www.renesas.com/" for the latest and detailed information.


**Renesas Electronics America Inc.**
2880 Scott Boulevard Santa Clara, CA 95050-2554, U.S.A.
Tel: +1-408-588-6000, Fax: +1-408-588-6130

**Renesas Electronics Canada Limited**
1101 Nicholson Road, Newmarket, Ontario L3Y 9C3, Canada
Tel: +1-905-898-5441, Fax: +1-905-898-3220

**Renesas Electronics Europe Limited**
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K
Tel: +44-1628-585-100, Fax: +44-1628-585-900

**Renesas Electronics Europe GmbH**
Arcadiastrasse 10, 40472 Düsseldorf, Germany
Tel: +49-211-65030, Fax: +49-211-6503-1327

**Renesas Electronics (China) Co., Ltd.**
7th Floor, Quantum Plaza, No.27 ZhiChunLu Haidian District, Beijing 100083, P.R.China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

**Renesas Electronics (Shanghai) Co., Ltd.**
Unit 204, 205, AZIA Center, No.1233 Lujiazui Ring Rd., Pudong District, Shanghai 200120, China
Tel: +86-21-5877-1818, Fax: +86-21-6887-7858 / -7898

**Renesas Electronics Hong Kong Limited**
Unit 1601-1613, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2886-9318, Fax: +852 2886-9022/9044

**Renesas Electronics Taiwan Co., Ltd.**
7F, No. 363 Fu Shing North Road Taipei, Taiwan
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670

**Renesas Electronics Singapore Pte. Ltd.**
1 harbourFront Avenue, #06-10, keppel Bay Tower, Singapore 098632
Tel: +65-6213-0200, Fax: +65-6278-8001

**Renesas Electronics Malaysia Sdn.Bhd.**
Unit 906, Block B, Menara Amcorp, Amcorp Trade Centre, No. 18, Jln Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: +60-3-7955-9390, Fax: +60-3-7955-9510

**Renesas Electronics Korea Co., Ltd.**
11F., Samik Lavied' or Bldg., 720-2 Yeoksam-Dong, Kangnam-Ku, Seoul 135-080, Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5141



Colophon 1.0