R8C/LA8A Group

Power Control Using Power-Off 0 Mode

R01AN0093EJ0101
Rev.1.01
Sep. 30, 2011

1. Abstract

This document describes a setting method and an application example for power control using power-off 0 mode in the R8C/LA8A Group.

2. Introduction

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

- MCU: R8C/LA8A 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

3.1 Program Outline

The number of resets caused by exiting power-off 0 mode is counted. After writing the count data to the data flash, the MCU enters power-off 0 mode while the $\overline{\text{WKUP0}}$ pin is high. When a low-level pulse is input to the $\overline{\text{WKUP0}}$ pin, the MCU exits power-off 0 mode and a reset operation is performed. When the $\overline{\text{WKUP0}}$ pin is low while the enter processing to power-off 0 mode is being performed, the MCU does not enter power-off 0 mode and a software reset is performed. After reset, the same processing as the processing performed after exiting power-off 0 mode is performed.

Settings

- Use power-off 0 mode in which all peripheral functions stop (oscillation stops).
- Input a low-level pulse for 10 µs or more to the $\overline{\text{WKUP0}}$ pin to exit power-off 0 mode. Before entering power-off 0 mode, input high to the $\overline{\text{WKUP0}}$ pin.
- After executing processing such as counting the number of resets caused by exiting power-off 0 mode, or writing to the data flash, enter power-off 0 mode again. (Since the RAM value becomes undefined when exiting power-off 0 mode, in the same way as when turning the power on, write the count data to the data flash.)
- Search the data written to the data flash and the write address for the updated data at the timings below. The search method is described in 3.2 Data Flash Area.

(1) When turning the power on (cold start-up)
(2) At voltage monitor 0 reset
(3) When exiting power-off 0 mode
(4) At software reset

- Write the count data at the following timing:

(1) When exiting power-off 0 mode
(2) At software reset

Figure 3.1 shows an Operation.

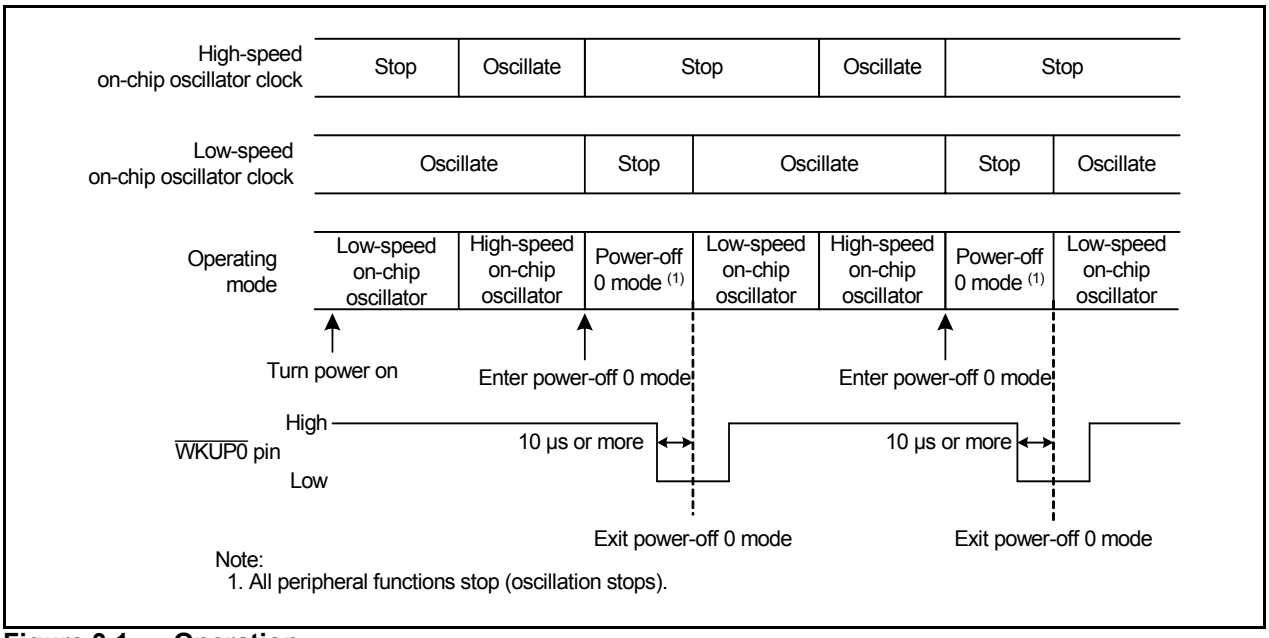


Figure 3.1 Operation

3.2 Data Flash Area

In this application note, the data flash area has a total of 512 4-byte records. These records are divided into two blocks of 256 records each (block A and block B). Figure 3.2 shows the Relationship between Data Flash and Records.

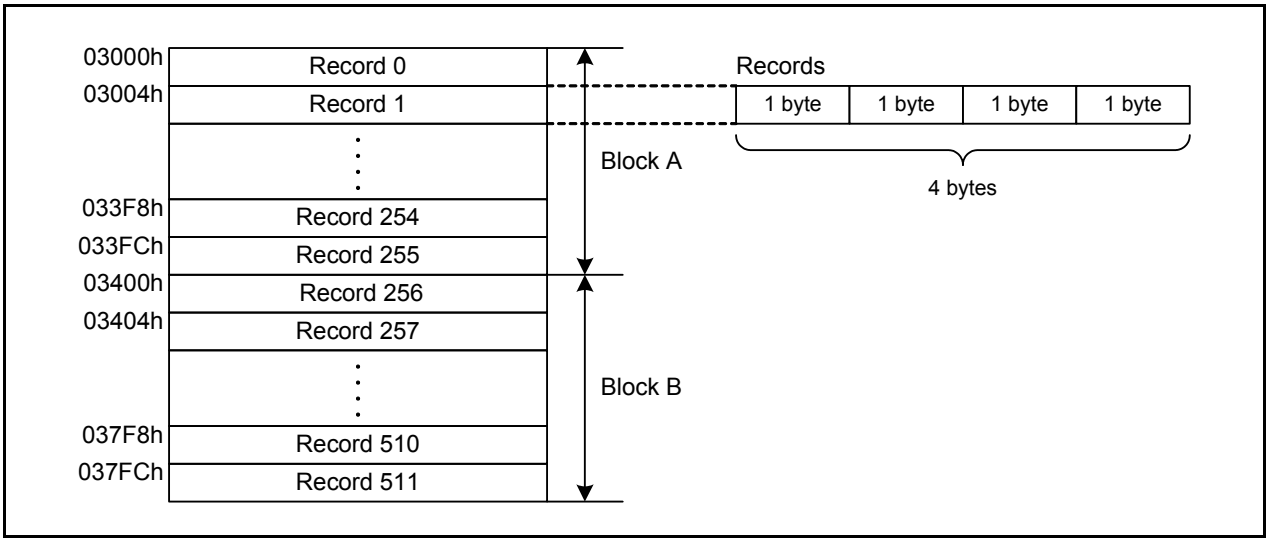


Figure 3.2 Relationship between Data Flash and Records

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. The method of searching for empty records is described below.

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

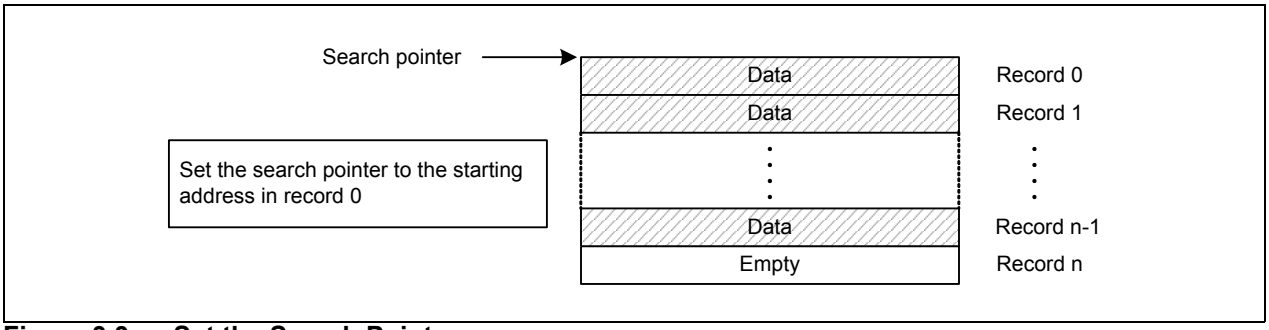


Figure 3.3 Set the Search Pointer

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

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

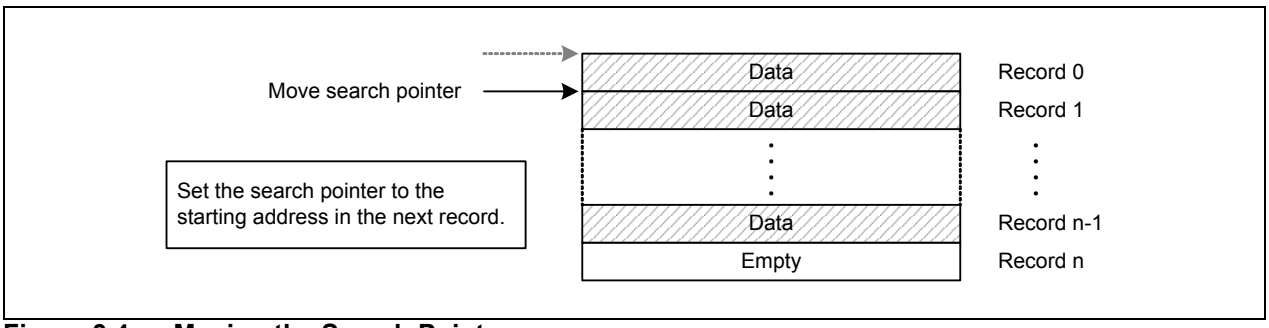


Figure 3.4 Moving the Search Pointer

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

(5) Execute steps (1) to (4) for block B (from record 256).

(6) Set the block used and write address based on the searched results to block A and block B. Table 3.1 lists Searched Results and Block Used/Write Address.

Table 3.1 Searched Results and Block Used/Write Address

Searched Result		Referred Figure	Block Used	Write Address	Erase Executed Block
Block A	Block B				
All records empty	All records empty	—	Block A	Block A starting address	Not executed
Written records found	All records empty	Figure 3.5	Block A	Block A search result	Not executed
All records written (no empty records)	All records empty	—	Block B	Block B starting address	Not executed
All records empty	Written records found	Figure 3.6	Block B	Block B search result	Not executed
All records empty	All records written (no empty records)	—	Block A	Block A starting address	Not executed
All records written (no empty records)	Written records found	Figure 3.7	Block B	Block B search result	Block A
Written records found	All records written (no empty records)	Figure 3.8	Block A	Block A search result	Block B
All records written (no empty records)	All records written (no empty records)	Figure 3.9	Block A	Block A starting address	Block A Block B
Written records found	Written records found	Figure 3.10	Block A	Block A starting address	Block A Block B

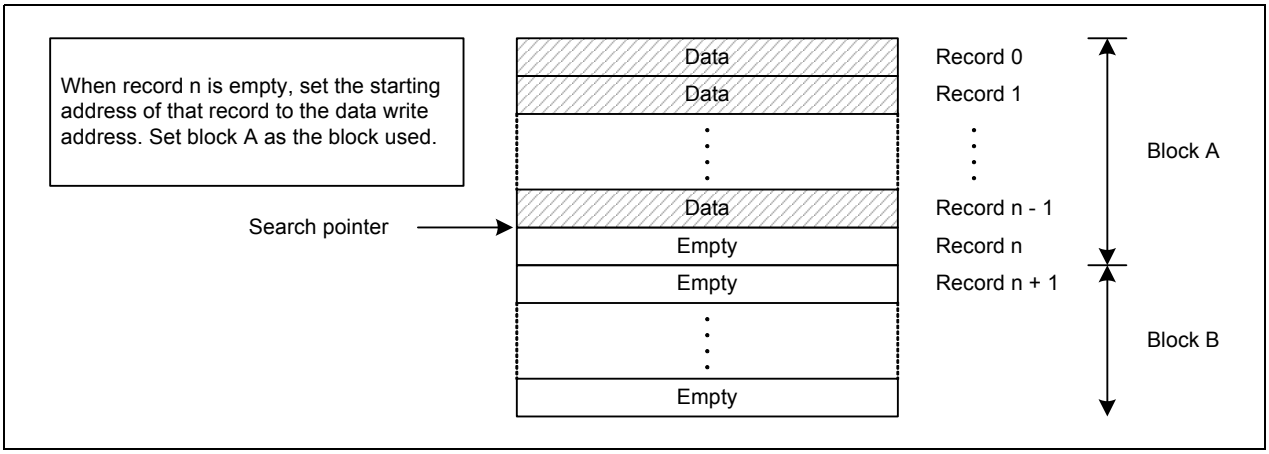


Figure 3.5 Block A: Written Records Found; Block B: All Records Empty

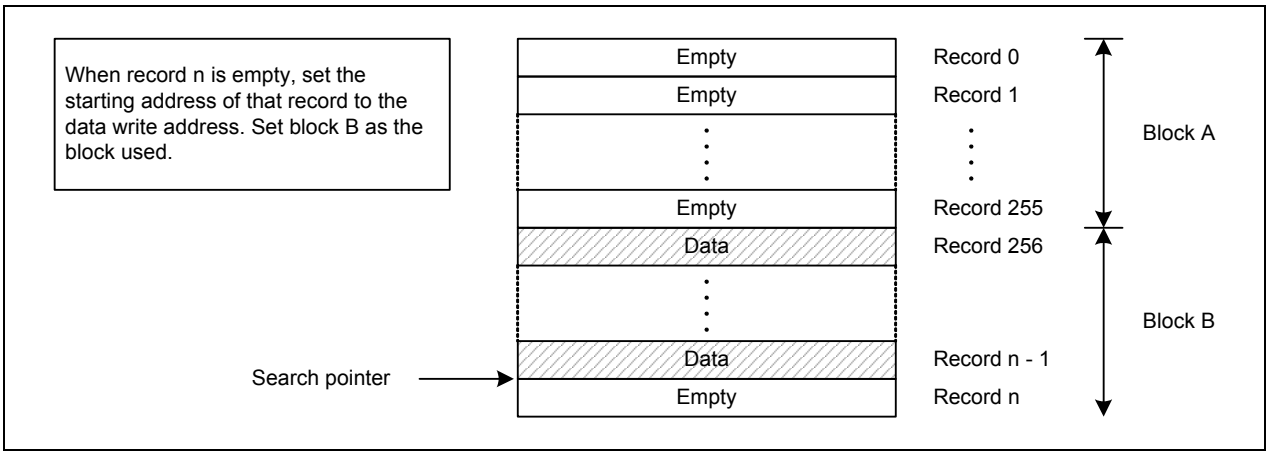


Figure 3.6 Block A: All Records Empty; Block B: Written Records Found

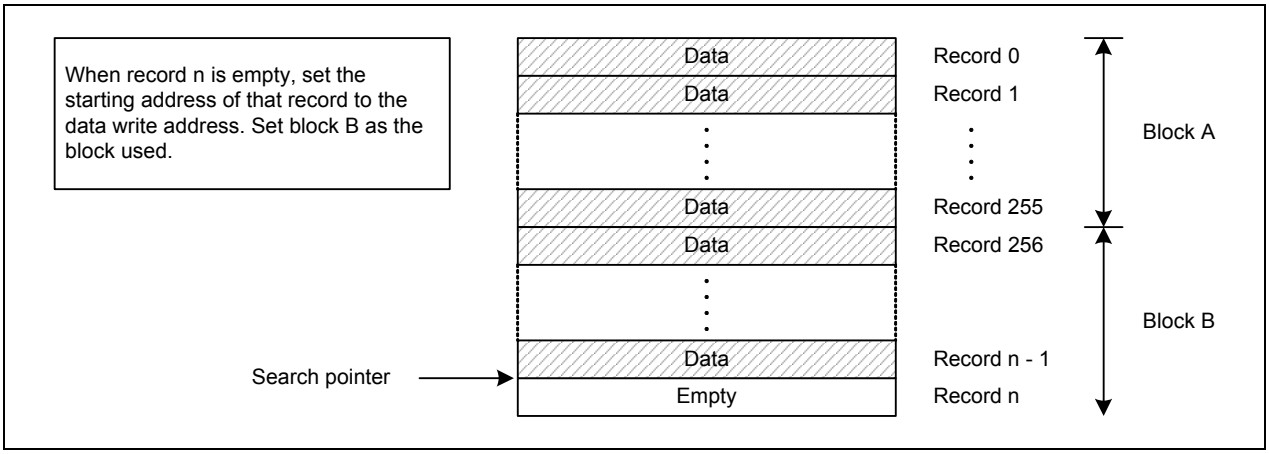


Figure 3.7 Block A: All Records Written; Block B: Written Records Found

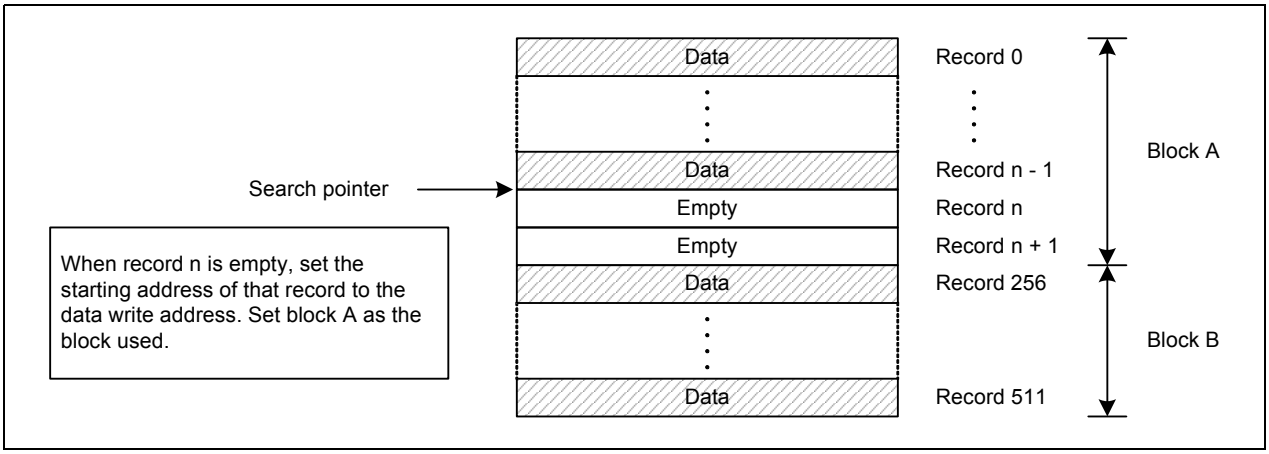


Figure 3.8 Block A: Written Records Found; Block B: All Records Written

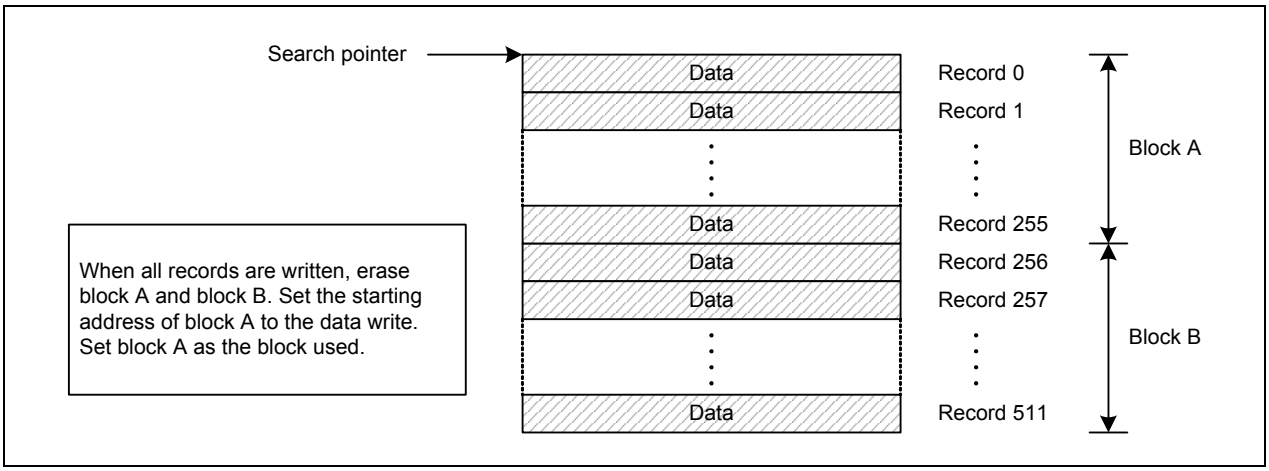


Figure 3.9 Block A: All Records Written; Block B: All Records Written

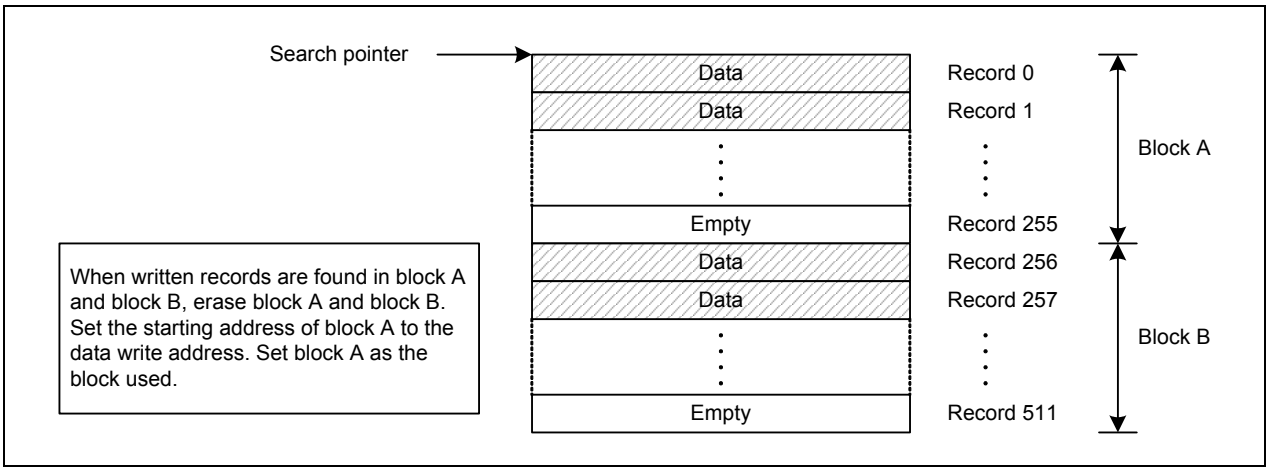


Figure 3.10 Block A: Written Records Found; Block B: Written Records Found

3.3 Memory

Table 3.2 Memory

Memory	Size	Remarks
ROM	1109 bytes	In the r01an0093_src.c module
RAM	14 bytes	In the r01an0093_src.c module
Maximum user stack	33 bytes	
Maximum interrupt stack	0 bytes	

Memory size varies depending on the C compiler version and compile options.
The above applies to the following conditions:
C compiler: M16C Series, R8C Family C Compiler V.5.45 Release 01
Compile options: -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/LA8A 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 function		
Argument	Argument name		Meaning
	None		—
Variable (global)	Variable name		Contents
	unsigned char write_req		Write request
Returned value	Type	Value	Meaning
	None	—	—
Function	Determine the reset source, set the system clock, read and write the data flash, and perform user program processing. The MCU enters power-off 0 mode after the processing is completed.		

Declaration	unsigned char reset_check (void)		
Outline	Reset source determination		
Argument	Argument name		Meaning
	None		—
Variable (global)	Variable name		Contents
	None		—
Returned value	Type	Value	Meaning
	unsigned char	COLD_HARDWARE_RST	Hardware reset (cold start-up)
		LVD_RST	Voltage monitor 0 reset
		SOFTWARE_RST	Software reset
		WDT_RST	Watchdog timer reset
		WARM_HARDWARE_RST	Hardware reset (warm start-up)
		POWER_OFF_MODE	Exit power-off 0 mode
Function	Determine the reset source and return the result.		

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 clock).		

Declaration	void user_program (unsigned char reset_result)		
Outline	User program processing		
Argument	Argument name		Meaning
	unsigned char reset_result		Reset source determination result
Variable (global)	Variable name		Contents
	unsigned char write_req		Write request
	unsigned long DataStr.all		Count data
Returned value	Type	Value	Meaning
	None	—	—
Function	Perform user program processing. Add processing based on the user system. In this application note, count the number of resets caused by exiting power-off 0 mode and set the data flash write request (write_req) to 1.		

Declaration	void power_control (void)		
Outline	Power control processing		
Argument	Argument name		Meaning
	None		—
Variable (global)	Variable name		Contents
	None		—
Returned value	Type	Value	Meaning
	None	—	—
Function	Enter power-off 0 mode. When the MCU cannot enter power-off 0 mode, perform a software reset.		

Declaration	void write_address_init (void)		
Outline	Initial setting of record write address		
Argument	Argument name		Meaning
	None		—
Variable (global)	Variable name		Contents
	unsigned char block_select		Block selected
	unsigned char *read_addr		Read address
	unsigned char *write_addr		Write address
	struct { unsigned char lower unsigned char middle_lower unsigned char middle_upper unsigned char upper } DataStr.byte		Count data
Returned value	Type	Value	Meaning
	None	—	—
Function	Search for empty records in each block, select the block used depending on block use, and set the starting address in the empty record to the write address (write_addr). Read the last write record and set the data as the count data.		

Declaration	unsigned char write_control(void)		
Outline	Data write control		
Argument	Argument name		Meaning
	None		—
Variable (global)	Variable name		Contents
	unsigned char record_data[RECORD_SIZE]		Write record
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	Set the write data in the write data setting processing. Write the record data.		

Declaration	void set_data(unsigned char *data)		
Outline	Write data setting		
Argument	Argument name		Meaning
	unsigned char *data		Write data starting address
Variable (global)	Variable name		Contents
	struct { unsigned char lower unsigned char middle_lower unsigned char middle_upper unsigned char upper } DataStr.byte		Count data
Returned value	Type	Value	Meaning
	None	—	—
Function	Set the record data (count data) written to the data flash.		

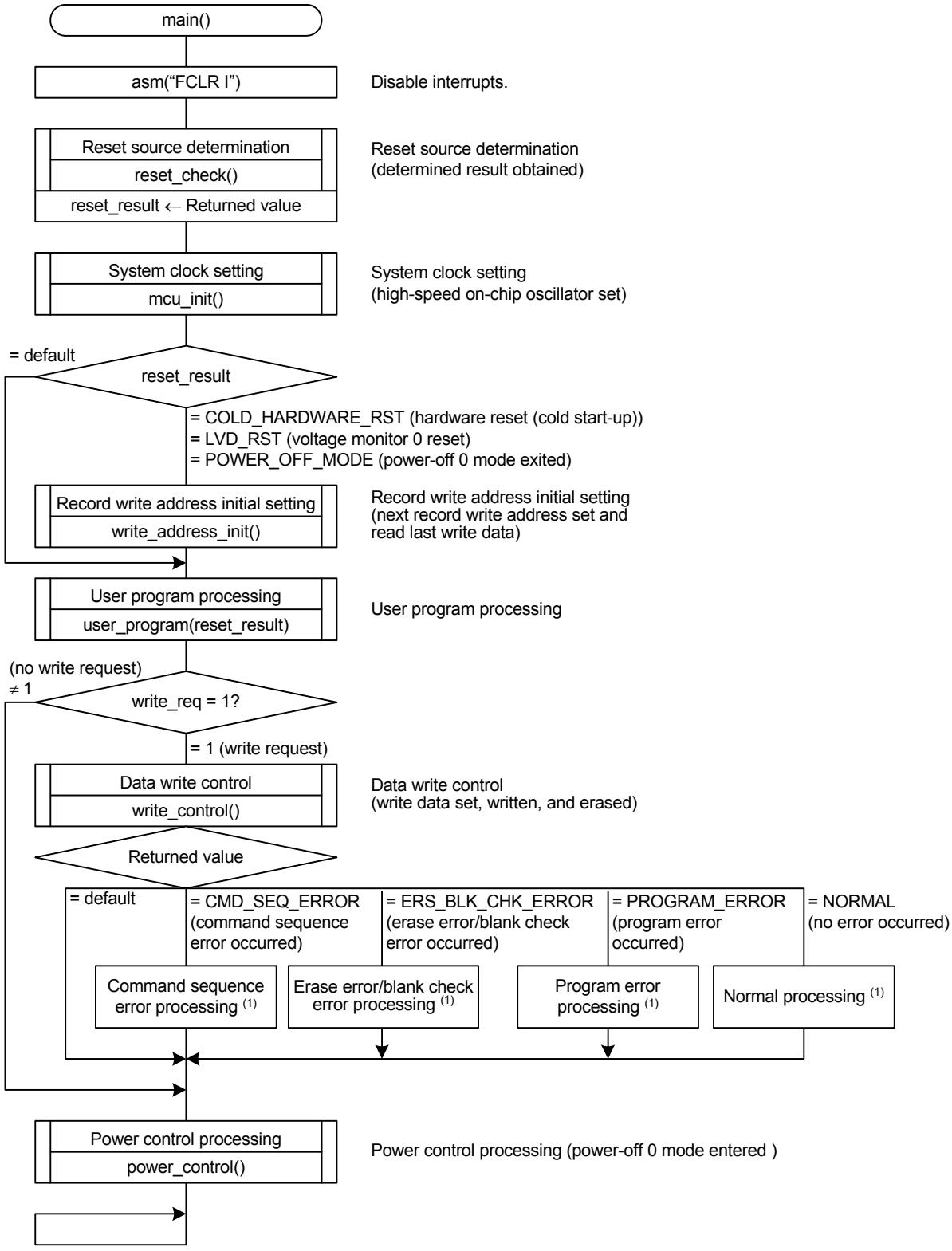
Declaration	unsigned char block_erase(unsigned char block_no)		
Outline	Block erase processing		
Argument	Argument name		Meaning
	unsigned char block_no		Block number to erase
Variable (global)	Variable name		Contents
	None		—
Returned value	Type	Value	Meaning
	unsigned char	NORMAL	Completed normally
		CMD_SEQ_ERROR	Command sequence error
		ERS_BLK_CHK_ERROR	Erase/blank check error
		PROGRAM_ERROR	Program error
Function	Erase the specified block in CPU rewrite mode (EW1 mode). Perform full status check and return the result after erasing is completed.		

Declaration	unsigned char data_write (unsigned char *data)		
Outline	Writing		
Argument	Argument name		Meaning
	unsigned char *data		Starting address of write data
Variable (global)	Variable name		Contents
	unsigned char block_select		Block selected
	unsigned char *write_addr		Write address
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	Write one record data from the write address (write_addr) in CPU rewrite mode (EW1 mode). Perform a full status check each time 1 byte is written and return the result.		

Declaration	unsigned char full_sts_chk(unsigned char *chk_adr)		
Outline	Full status check		
Argument	Argument name		Meaning
	unsigned char *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 a full status check and return the result.		

4.2 Main Function

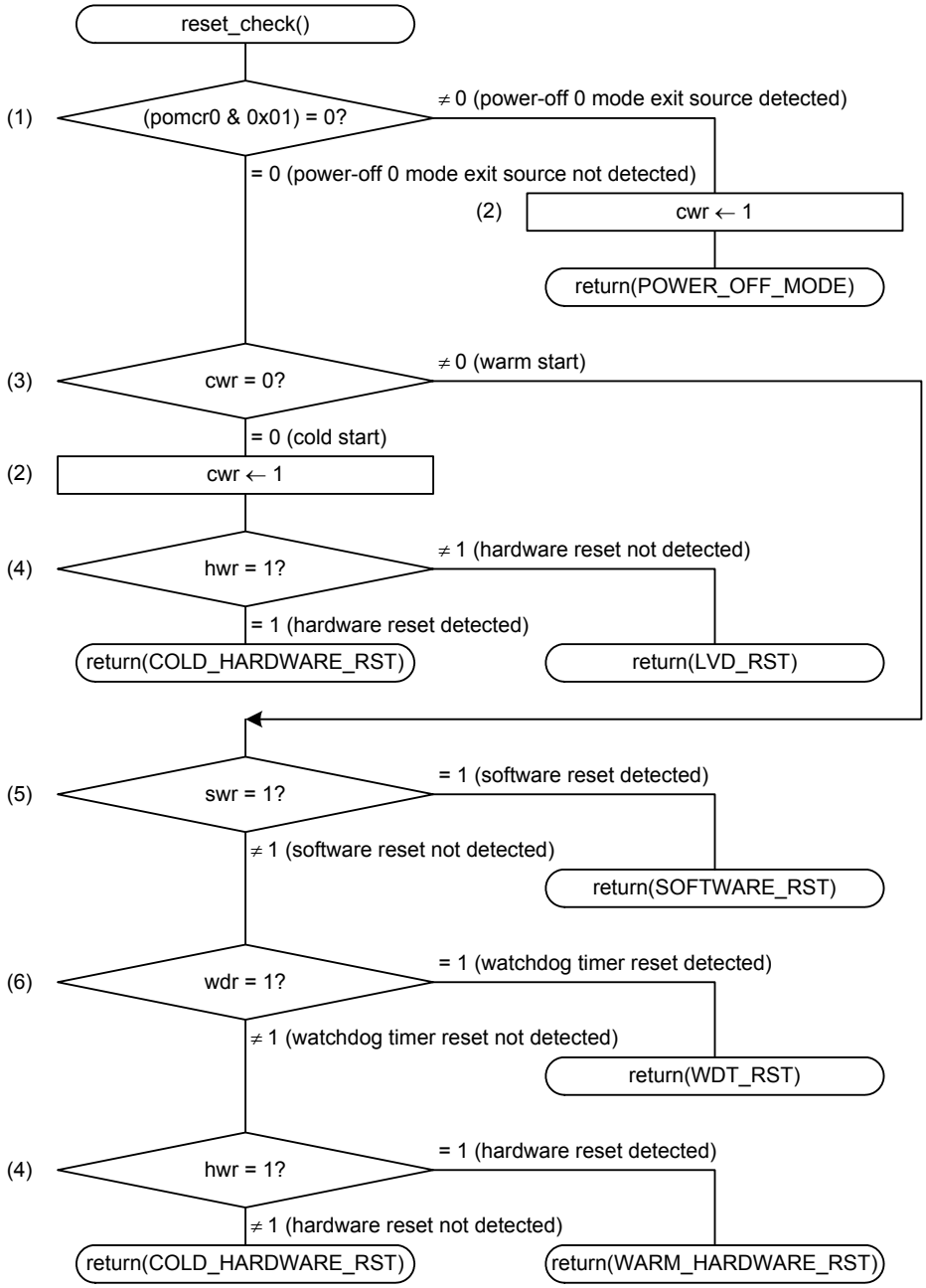
• Flowchart



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

4.3 Reset Source Determination

• Flowchart



• Register settings

(1) Determine whether the power-off 0 mode exit source is detected or not detected.

Power-Off Mode Control Register 0 (POMCR0)

Bit	Symbol	Bit Name	Function	R/W
b0	POM00	$\overline{\text{WKUP0}}$ source power-off 0 exit flag	0: Undetected 1: Detected	R

(2) Set the cold start-up/warm start-up determine flag to 1.

Reset Source Determination Register (RSTFR)

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

Bit	Symbol	Bit Name	Function	R/W
b0	CWR	Cold start-up/warm start-up determine flag	1: Warm start-up	R/W

(3) Determine cold start-up or warm start-up.

Reset Source Determination Register (RSTFR)

Bit	Symbol	Bit Name	Function	R/W
b0	CWR	Cold start-up/warm start-up determine flag	0: Cold start-up 1: Warm start-up	R/W

(4) Determine whether a hardware reset is detected or not detected.

Reset Source Determination Register (RSTFR)

Bit	Symbol	Bit Name	Function	R/W
b1	HWR	Hardware reset detect flag	0: Not detected 1: Detected	R

(5) Determine whether a software reset is detected or not detected.

Reset Source Determination Register (RSTFR)

Bit	Symbol	Bit Name	Function	R/W
b2	SWR	Software reset detect flag	0: Not detected 1: Detected	R

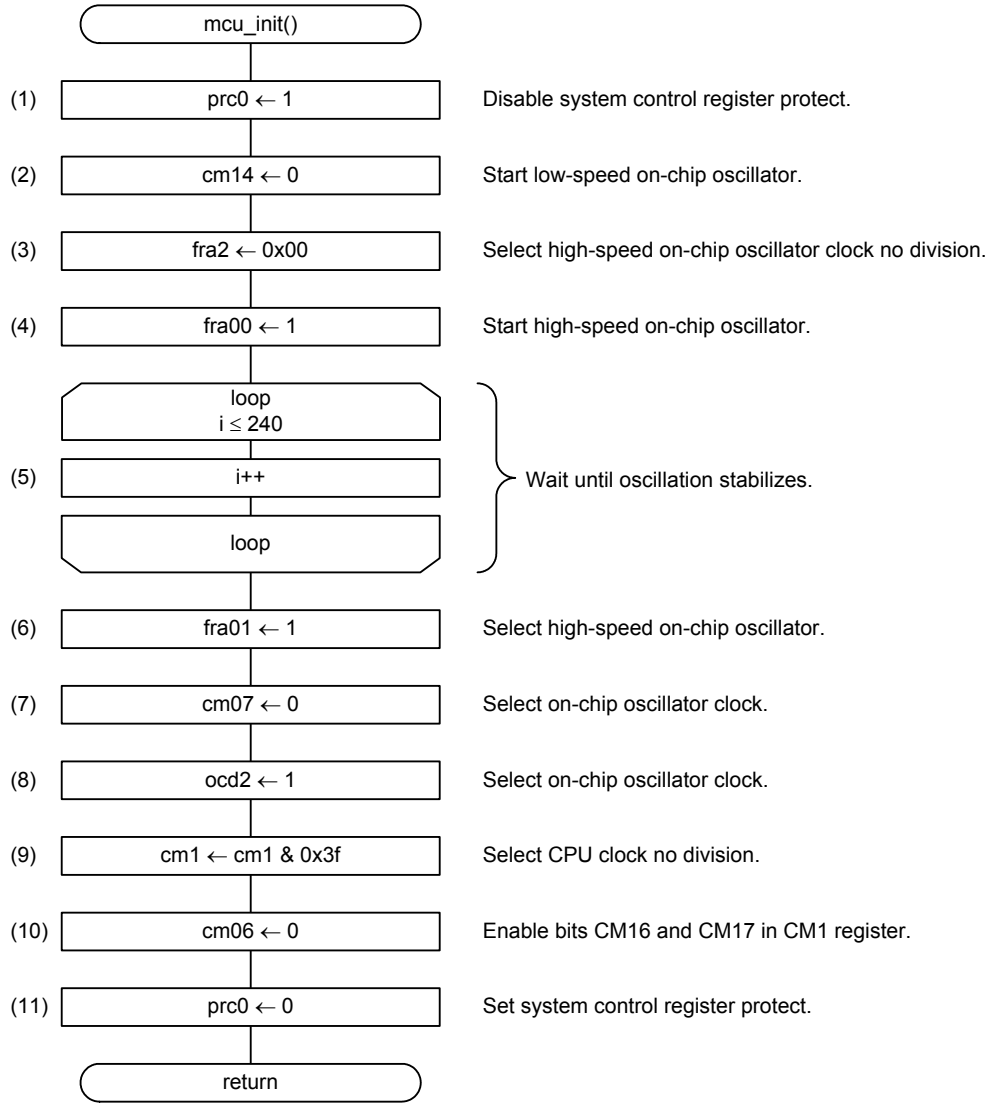
(6) Determine whether a watchdog timer reset is detected or not detected.

Reset Source Determination Register (RSTFR)

Bit	Symbol	Bit Name	Function	R/W
b3	WDR	Watchdog timer reset detect flag	0: Not detected 1: Detected	R

4.4 System Clock Setting

• Flowchart



• Register settings

(1) Enable writing to registers CM0, CM1, CM3, OCD, FRA0, FRC0, FRA2, and FRC1.

Protect Register (PRCR)

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

Bit	Symbol	Bit Name	Function	R/W
b0	PRC0	Protect bit 0	Enables writing to registers CM0, CM1, CM3, OCD, FRA0, FRC0, FRA2, and FRC1. 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 high-speed on-chip oscillator frequency switch bit.

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 switch bit	Division ratio selection These bits select the division ratio for the high-speed on-chip oscillator clock. b2 b1 b0 0 0 0: Divide-by-1 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 as the on-chip oscillator clock.

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.

System Clock Control Register 0 (CM0)

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

Bit	Symbol	Bit Name	Function	R/W
b7	CM07	System clock select bit	0: XIN clock or on-chip oscillator clock	R/W

(8) 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	On-chip oscillator clock select bit	1: On-chip oscillator clock selected	R/W

(9) 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 0 0: No division mode	R/W
b7	CM17			R/W

(10) Set CPU clock division select bit 0.

System Clock Control Register 0 (CM0)

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

(11) Disable writing to registers CM0, CM1, CM3, OCD, FRA0, FRC0, FRA2, and FRC1.

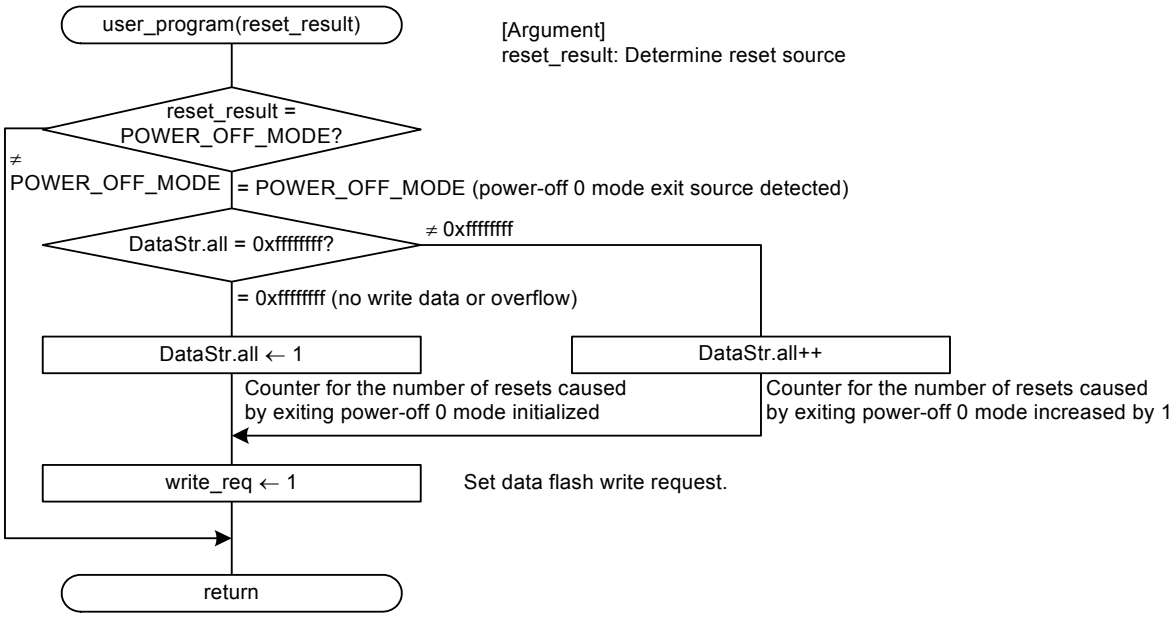
Protect Register (PRCR)

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

Bit	Symbol	Bit Name	Function	R/W
b0	PRC0	Protect bit 0	Enables writing to registers CM0, CM1, CM3, OCD, FRA0, FRC0, FRA2, and FRC1. 0: Write disabled	R/W

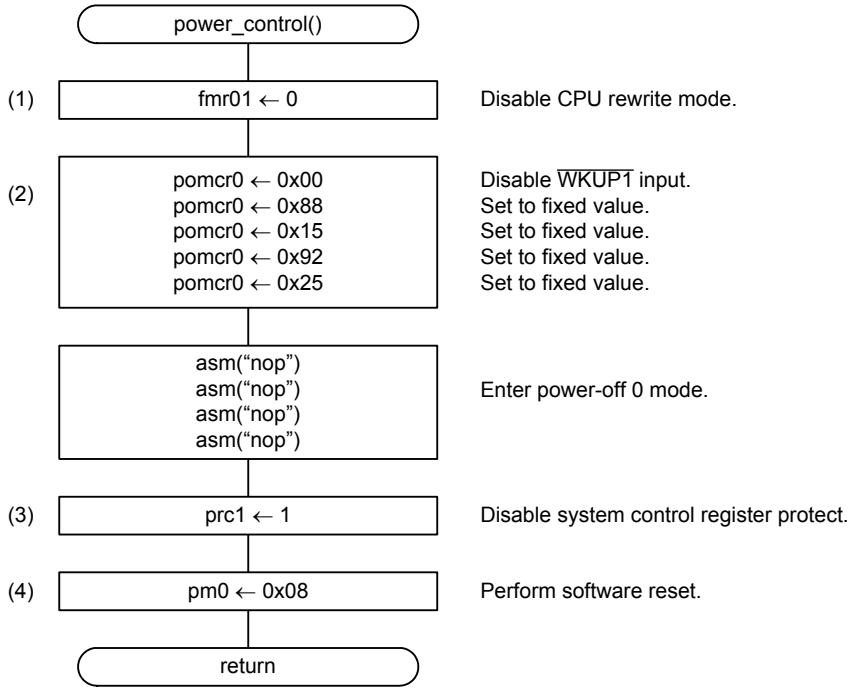
4.5 User Program Processing

• Flowchart



4.6 Power Control Processing

• Flowchart



• Register settings

(1) Disable 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

(2) Disable $\overline{\text{WKUP1}}$ input for initial write. Then write 88h, 15h, 92h, and 25h successively.

Power-Off Mode Control Register 0 (POMCR0)

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

Initial write: Setting the input pin to exit the mode in power-off 0 mode

Bit	Symbol	Bit Name	Function	R/W
b1	POM01	$\overline{\text{WKUP1}}$ input enable bit	0: Input disabled	W

Second to fifth writes: Entering power-off 0 mode

Bit	Function	R/W
b7 to b0	Write 88h, 15h, 92h, and 25h successively.	W

(3) Enable writing to registers PM0 and PM1.

Protect Register (PRCR)

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

Bit	Symbol	Bit Name	Function	R/W
b1	PRC1	Protect bit 1	Enables writing to registers PM0 and PM1. 1: Write enabled	R/W

(4) Perform a software reset.

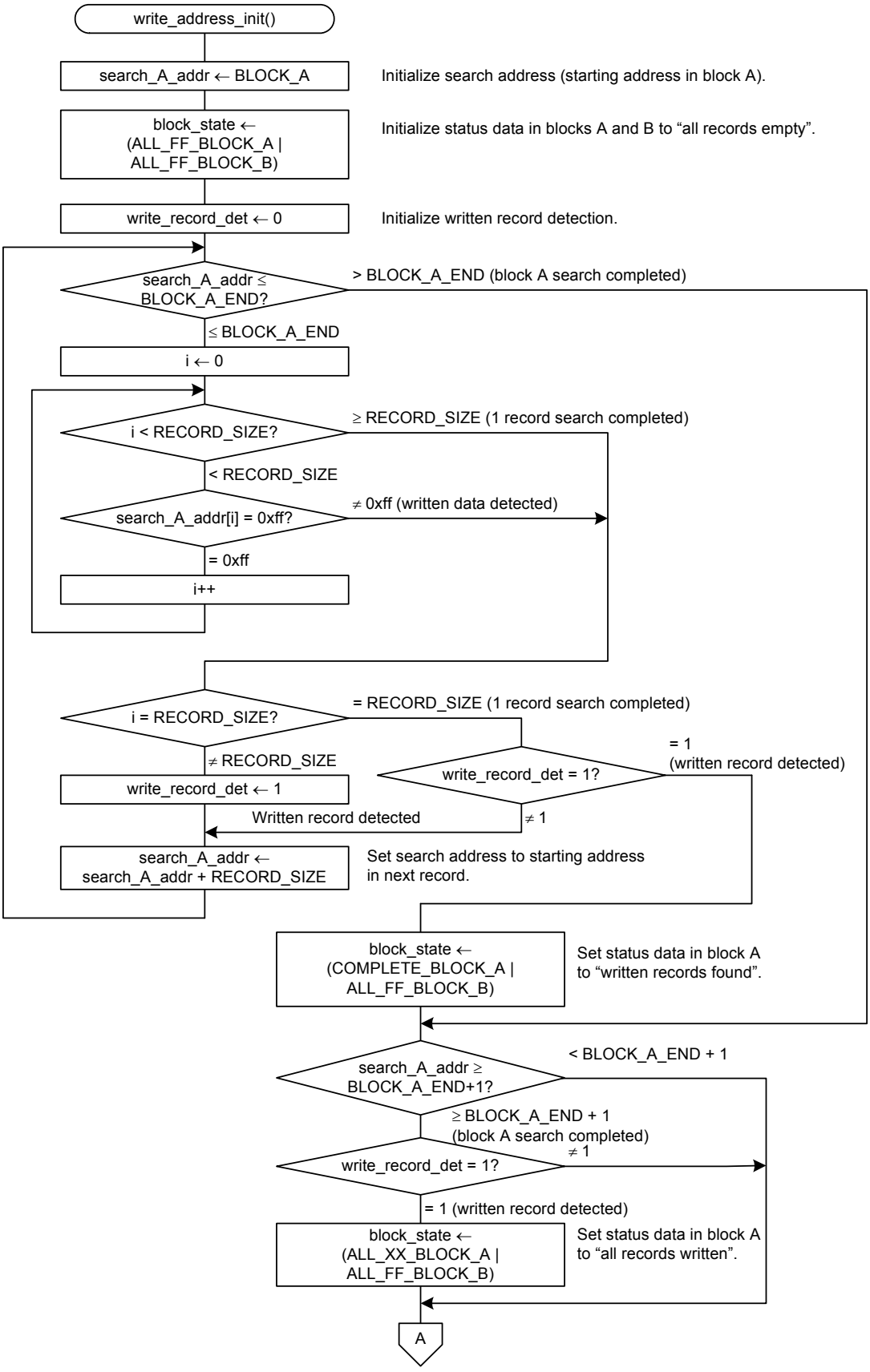
Processor Mode Register 0 (PM0)

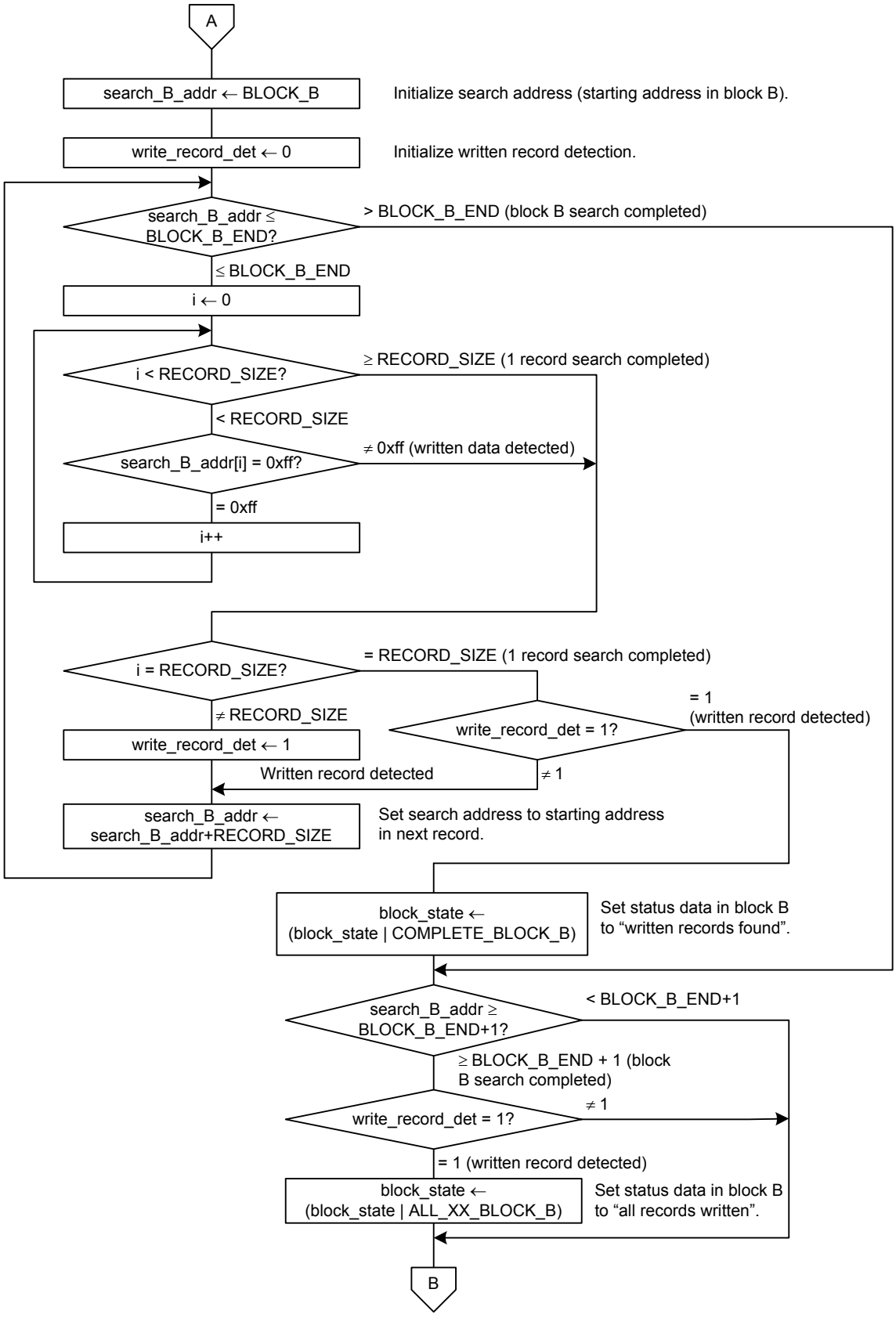
Bit	b7	b6	b5	b4	b3	b2	b1	b0
Setting Value	—	—	—	—	1	—	—	—

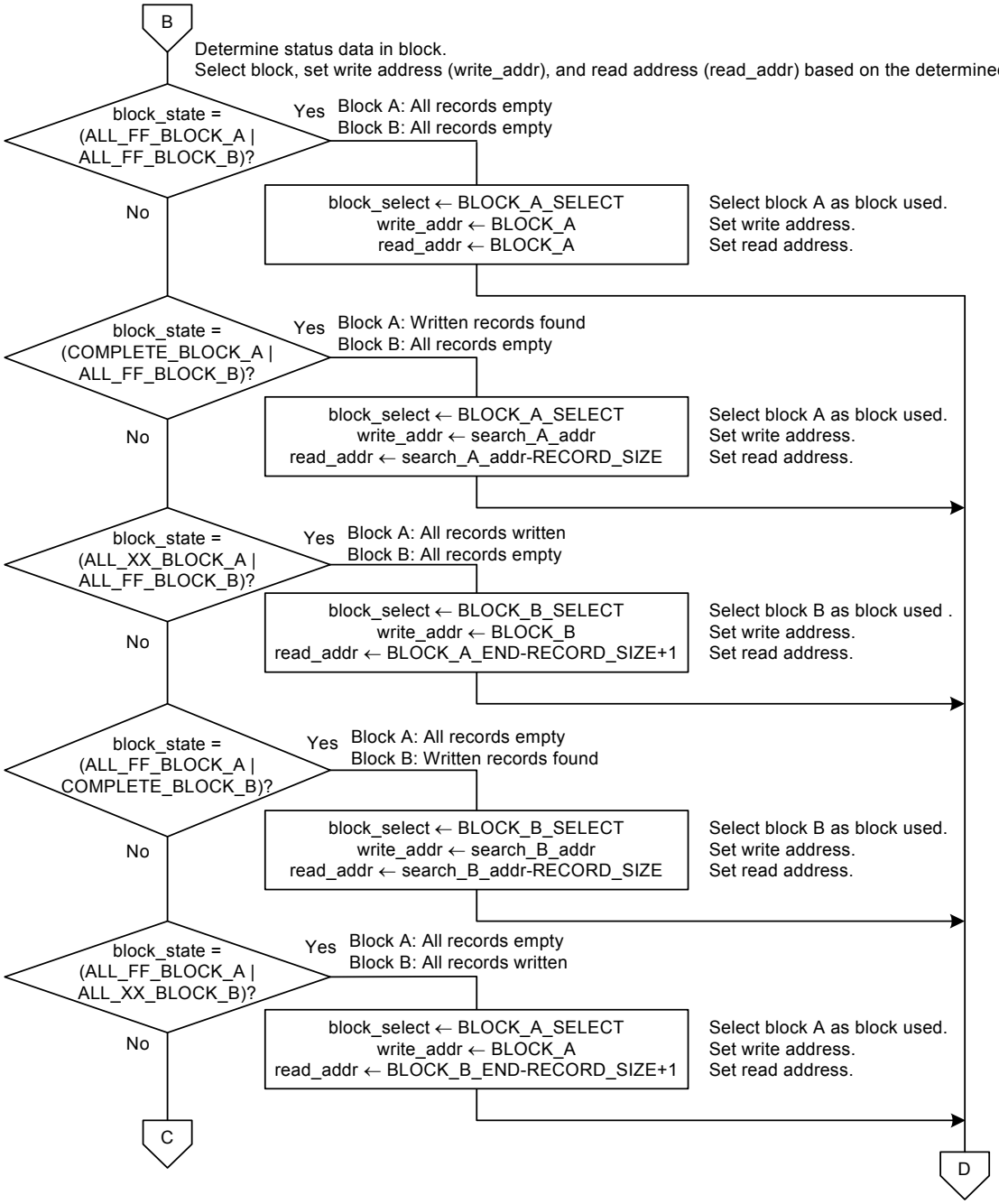
Bit	Symbol	Bit Name	Function	R/W
b3	PM03	Software reset bit	Setting this bit to 1 resets the MCU. When read, the content is 0.	R/W

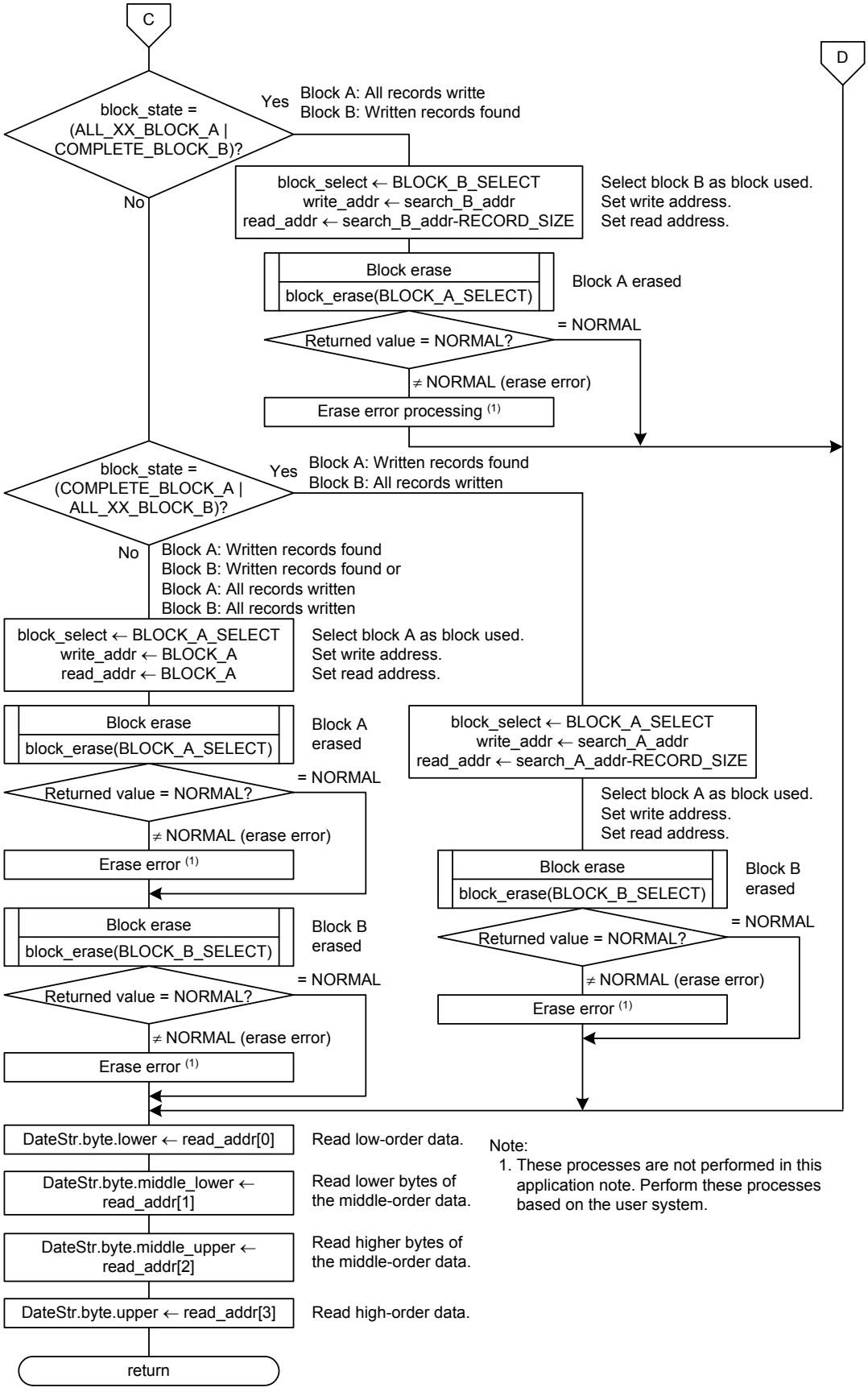
4.7 Initial Setting of Record Write Address

- Flowchart



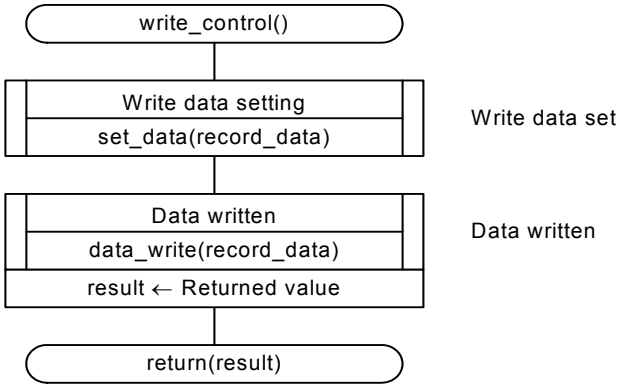

A
search_B_addr ← BLOCK_B
Initialize search address (starting address in block B).
write_record_det ← 0
Initialize written record detection.
search_B_addr ≤ BLOCK_B_END?
> BLOCK_B_END (block B search completed)
≤ BLOCK_B_END
i ← 0
i < RECORD_SIZE?
≥ RECORD_SIZE (1 record search completed)
< RECORD_SIZE
search_B_addr[i] = 0xff?
≠ 0xff (written data detected)
= 0xff
i++
i = RECORD_SIZE?
= RECORD_SIZE (1 record search completed)
≠ RECORD_SIZE
write_record_det = 1?
= 1
(written record detected)
write_record_det ← 1
Written record detected
≠ 1
search_B_addr ← search_B_addr+RECORD_SIZE
Set search address to starting address in next record.
block_state ← (block_state | COMPLETE_BLOCK_B)
Set status data in block B to “written records found”.
search_B_addr ≥ BLOCK_B_END+1?
< BLOCK_B_END+1
≥ BLOCK_B_END + 1 (block B search completed)
write_record_det = 1?
≠ 1
= 1 (written record detected)
block_state ← (block_state | ALL_XX_BLOCK_B)
Set status data in block B to “all records written”.
B


B
Determine status data in block.
Select block, set write address (write_addr), and read address (read_addr) based on the determined result.
block_state = (ALL_FF_BLOCK_A | ALL_FF_BLOCK_B)?
Yes
Block A: All records empty
Block B: All records empty
block_select ← BLOCK_A_SELECT
write_addr ← BLOCK_A
read_addr ← BLOCK_A
Select block A as block used.
Set write address.
Set read address.
No
block_state = (COMPLETE_BLOCK_A | ALL_FF_BLOCK_B)?
Yes
Block A: Written records found
Block B: All records empty
block_select ← BLOCK_A_SELECT
write_addr ← search_A_addr
read_addr ← search_A_addr-RECORD_SIZE
Select block A as block used.
Set write address.
Set read address.
No
block_state = (ALL_XX_BLOCK_A | ALL_FF_BLOCK_B)?
Yes
Block A: All records written
Block B: All records empty
block_select ← BLOCK_B_SELECT
write_addr ← BLOCK_B
read_addr ← BLOCK_A_END-RECORD_SIZE+1
Select block B as block used .
Set write address.
Set read address.
No
block_state = (ALL_FF_BLOCK_A | COMPLETE_BLOCK_B)?
Yes
Block A: All records empty
Block B: Written records found
block_select ← BLOCK_B_SELECT
write_addr ← search_B_addr
read_addr ← search_B_addr-RECORD_SIZE
Select block B as block used.
Set write address.
Set read address.
No
block_state = (ALL_FF_BLOCK_A | ALL_XX_BLOCK_B)?
Yes
Block A: All records empty
Block B: All records written
block_select ← BLOCK_A_SELECT
write_addr ← BLOCK_A
read_addr ← BLOCK_B_END-RECORD_SIZE+1
Select block A as block used.
Set write address.
Set read address.
No
C
D


C
D
block_state = (ALL_XX_BLOCK_A | COMPLETE_BLOCK_B)?
Yes
Block A: All records writte
Block B: Written records found
No
block_select ← BLOCK_B_SELECT
write_addr ← search_B_addr
read_addr ← search_B_addr-RECORD_SIZE
Select block B as block used.
Set write address.
Set read address.
Block erase
block_erase(BLOCK_A_SELECT)
Block A erased
Returned value = NORMAL?
= NORMAL
≠ NORMAL (erase error)
Erase error processing (1)
block_state = (COMPLETE_BLOCK_A | ALL_XX_BLOCK_B)?
Yes
Block A: Written records found
Block B: All records written
No
Block A: Written records found
Block B: Written records found or
Block A: All records written
Block B: All records written
block_select ← BLOCK_A_SELECT
write_addr ← BLOCK_A
read_addr ← BLOCK_A
Select block A as block used.
Set write address.
Set read address.
Block erase
block_erase(BLOCK_A_SELECT)
Block A erased
Returned value = NORMAL?
= NORMAL
≠ NORMAL (erase error)
Erase error (1)
block_select ← BLOCK_A_SELECT
write_addr ← search_A_addr
read_addr ← search_A_addr-RECORD_SIZE
Select block A as block used.
Set write address.
Set read address.
Block erase
block_erase(BLOCK_B_SELECT)
Block B erased
Returned value = NORMAL?
= NORMAL
≠ NORMAL (erase error)
Erase error (1)
Block erase
block_erase(BLOCK_B_SELECT)
Block B erased
Returned value = NORMAL?
= NORMAL
≠ NORMAL (erase error)
Erase error (1)
DateStr.byte.lower ← read_addr[0]
Read low-order data.
DateStr.byte.middle_lower ← read_addr[1]
Read lower bytes of the middle-order data.
DateStr.byte.middle_upper ← read_addr[2]
Read higher bytes of the middle-order data.
DateStr.byte.upper ← read_addr[3]
Read high-order data.
return
Note:
1. These processes are not performed in this application note. Perform these processes based on the user system.

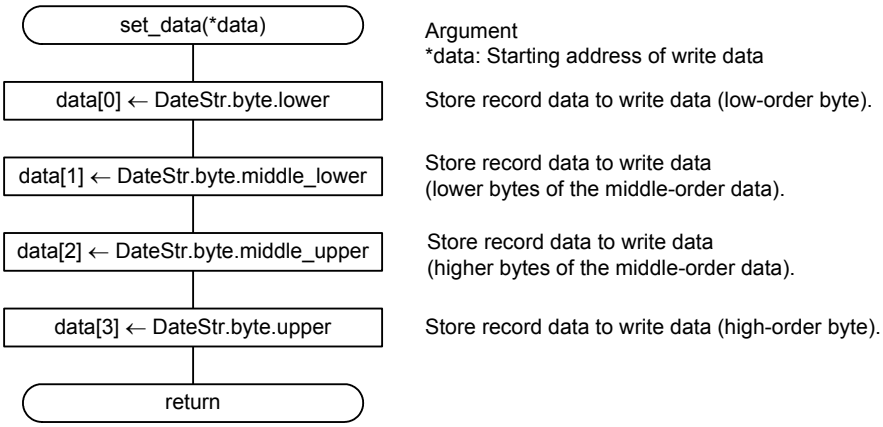
4.8 Data Write Control

- Flowchart



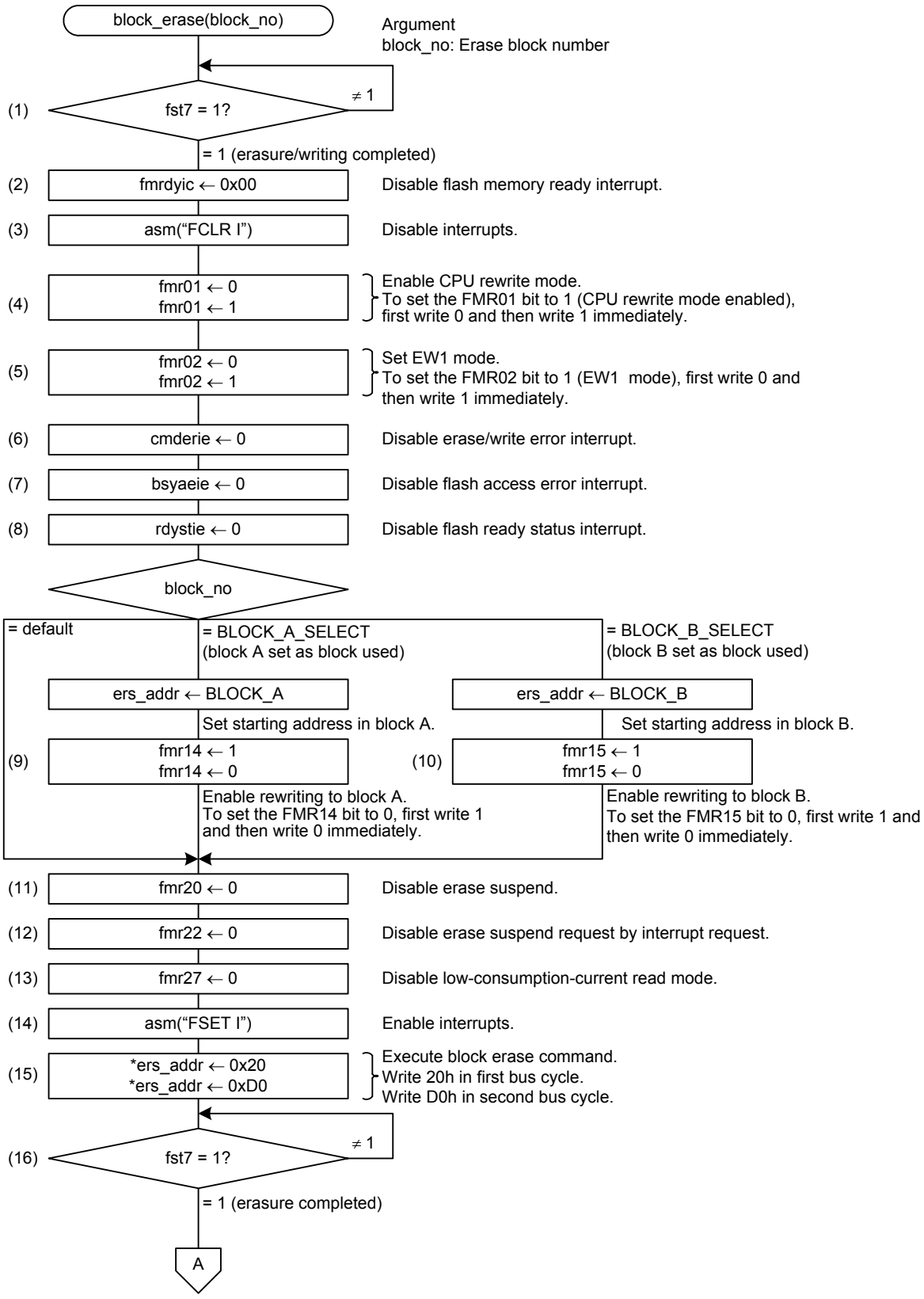
4.9 Write Data Setting

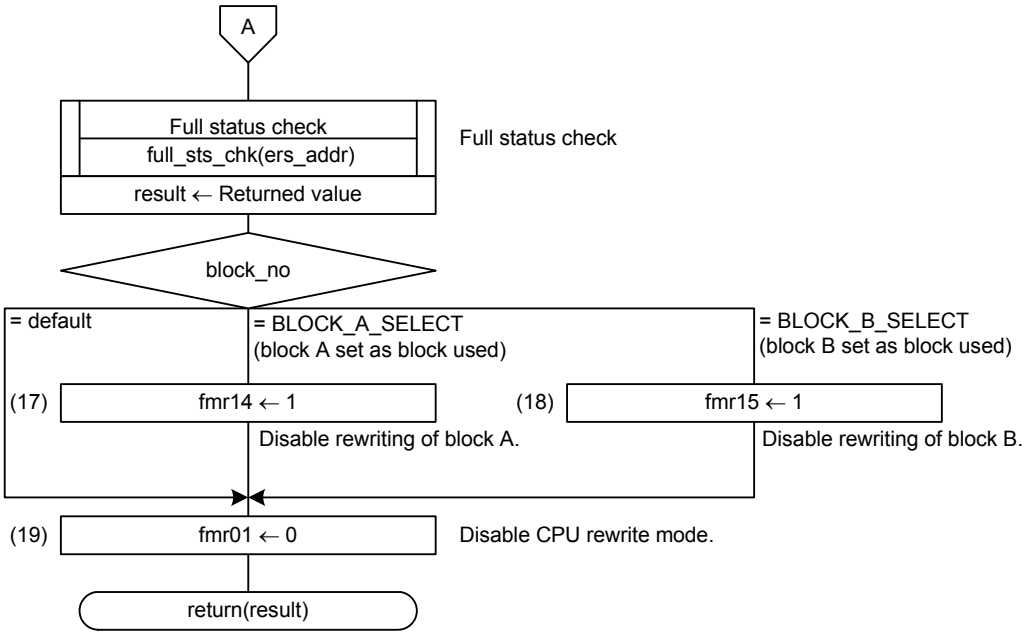
- Flowchart



4.10 Block Erase Processing

• Flowchart




A
Full status check
full_sts_chk(ers_addr)
result ← Returned value
Full status check
block_no
= default
= BLOCK_A_SELECT
(block A set as block used)
= BLOCK_B_SELECT
(block B set as block used)
(17)
fmr14 ← 1
(18)
fmr15 ← 1
Disable rewriting of block A.
Disable rewriting of block B.
(19)
fmr01 ← 0
Disable CPU rewrite mode.
return(result)

• Register settings

(1) Wait until the flash memory becomes ready.

Flash Memory Status Register (FST)

Bit	Symbol	Bit Name	Function	R/W
b7	FST7	Ready/busy status flag	0: Busy 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	0

Bit	Symbol	Bit Name	Function	R/W
b0	ILVL0	Interrupt priority level select bit	b2 b1 b0 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

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

(4) Enable CPU rewrite mode. To set the FMR01 bit to 1, first write 0 and then write 1 immediately.

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. To set the FMR02 bit to 1, first write 0 and then write 1 immediately.

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. To set the FMR14 bit to 1, first write 1 and then write 0 immediately.

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. To set the FMR15 bit to 0, first write 1 and then write 0 immediately.

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) 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	Suspend enable bit	0: Suspend disabled	R/W

(12) 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: Suspend request disabled by interrupt request	R/W

(13) 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

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

(15) In the first bus cycle, write block erase command 20h to a given address in the block to be erased. Erasure (erase and erase verify) starts by writing confirmation command D0h in the second bus cycle.

(16) Wait until erasure is completed.

Flash Memory Status Register (FST)

Bit	Symbol	Bit Name	Function	R/W
b7	FST7	Ready/busy status flag	0: Busy 1: Ready	R

(17) Disable rewriting of data flash block A when erasing block A is completed.

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

(18) Disable rewriting of data flash block B when erasing block B is completed.

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

(19) Disable CPU rewrite mode.

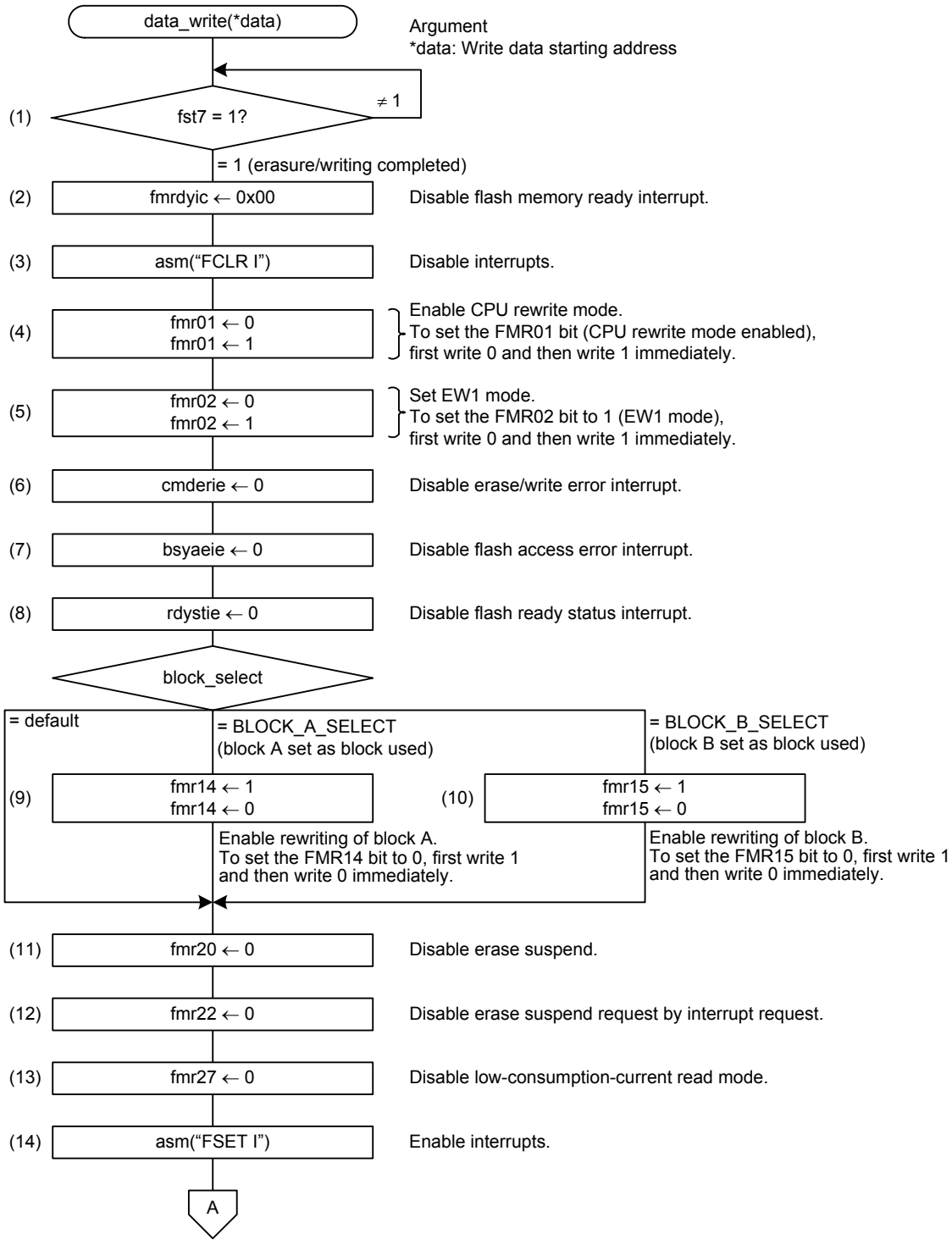
Flash Memory Control Register 0 (FMR0)

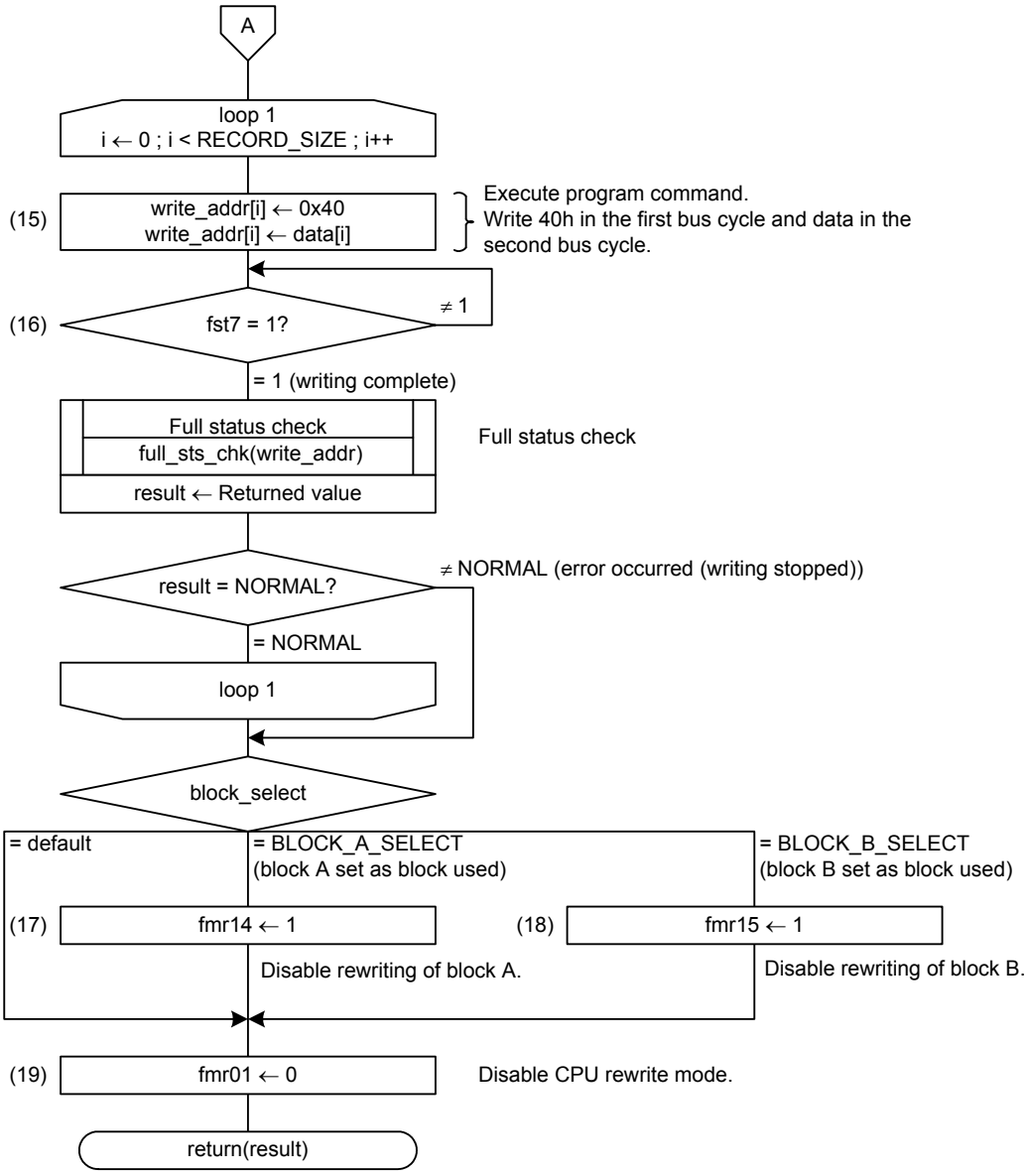
Bit	b7	b6	b5	b4	b3	b2	b1	b0
Setting Value				x	x		0	—

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

4.11 Writing

• Flowchart





• Register settings

(1) Wait until the flash memory becomes ready.

Flash Memory Status Register (FST)

Bit	Symbol	Bit Name	Function	R/W
b7	FST7	Ready/busy status flag	0: Busy 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	0

Bit	Symbol	Bit Name	Function	R/W
b0	ILVL0	Interrupt priority level select bit	b2 b1 b0 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

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

(4) Enable CPU rewrite mode. To set the FMR01 bit to 1, first write 0 and then write 1 immediately.

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. To set the FMR02 bit to 1, first write 0 and then write 1 immediately.

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 of block A. To set the FMR14 bit to 1, first write 1 and then write 0 immediately.

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 of block B. To set the FMR15 bit to 1, first write 1 and then write 0 immediately.

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) 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	Suspend enable bit	0: Suspend disabled	R/W

(12) 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: Suspend request disabled by interrupt request	R/W

(13) 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

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

(15) Write program command 40h to the write address in the first bus cycle. Writing (data write and verification) 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.

(16) Wait until writing is completed.

Flash Memory Status Register (FST)

Bit	Symbol	Bit Name	Function	R/W
b7	FST7	Ready/busy status flag	0: Busy 1: Ready	R

(17) Disable rewriting of data flash block A when rewriting of block A is completed.

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

(18) Disable rewriting of data flash block B when rewriting of block B is completed.

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

(19) Disable CPU rewrite mode.

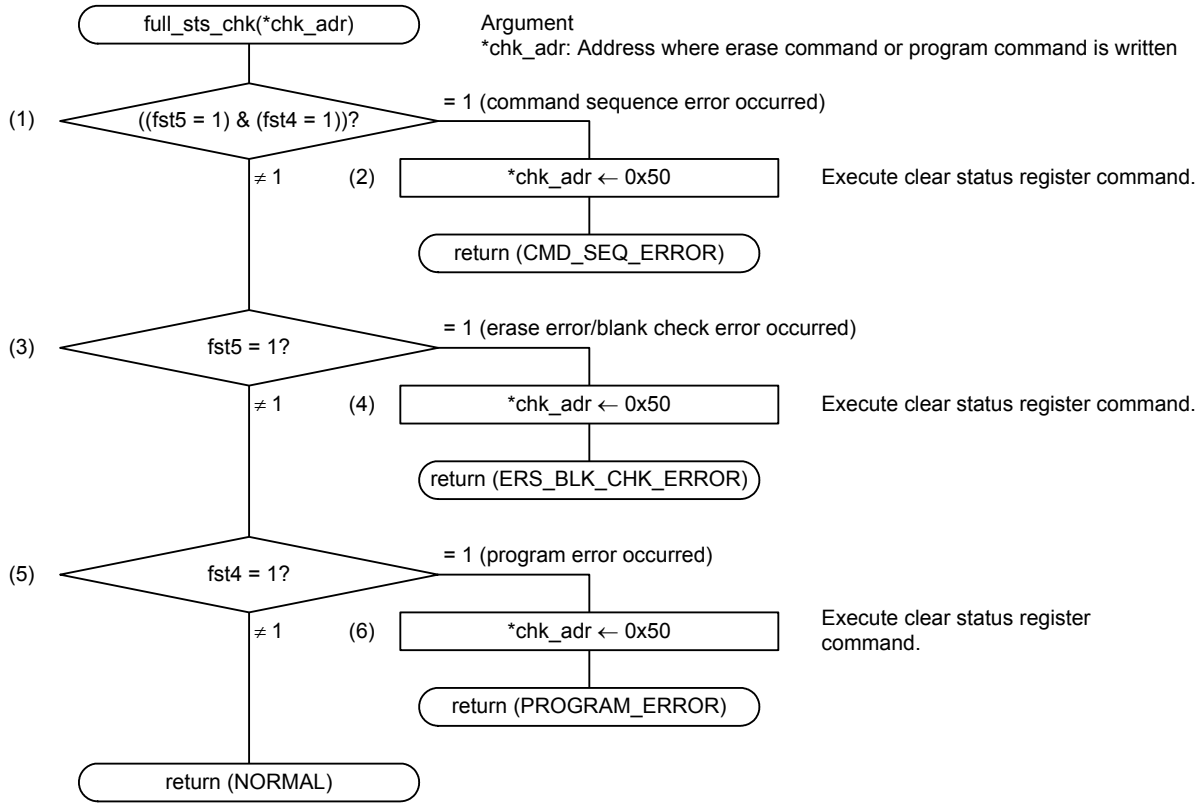
Flash Memory Control Register 0 (FMR0)

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

4.12 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 1: Program error	R
b5	FST5	Erase error/blank check error flag	0: No erase error/blank check error 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.

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 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.

Flash Memory Status Register (FST)

Bit	Symbol	Bit Name	Function	R/W
b4	FST4	Program error flag	0: No program error 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.

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 Family page.

6. Reference Documents

R8C/LA8A Group User’s Manual: Hardware Rev.1.01
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/LA8A Group Power Control Using Power-Off 0 Mode

Rev.	Date	Description	
		Page	Summary
1.00	May 16, 2011	—	First edition issued
1.01	Sep. 30, 2011	—	Sample program union revised

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.
13F, 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 1.1