M16C/63, 64, 64A, 64C, 65, 65C, 6C, 5LD, 56D, 5L, 56, 5M, and 57 Groups

The Use Example of CPU Rewrite Mode (EW0 mode)

R01AN0172EJ0103
Rev.1.03
Jan. 31, 2011

1. Abstract

This document describes the use example of CPU rewrite mode (EW0 mode).

2. Introduction

The application example described in this document applies to the following microcomputers (MCUs):

- MCU: M16C/63 Group, M16C/64 Group, M16C/64A Group, M16C/64C Group, M16C/65 Group (only program ROM 1 is 512K byte or less), M16C/65C Group, M16C/6C Group, M16C/5LD Group, M16C/56D Group, M16C/5L Group, M16C/56 Group, M16C/5M Group, M16C/57 Group

This application note can be used with other M16C Family MCUs which have the same special function registers (SFRs) as the above groups. Check the user's manual for any modifications to functions. Careful evaluation is recommended before using the program described in this application note.

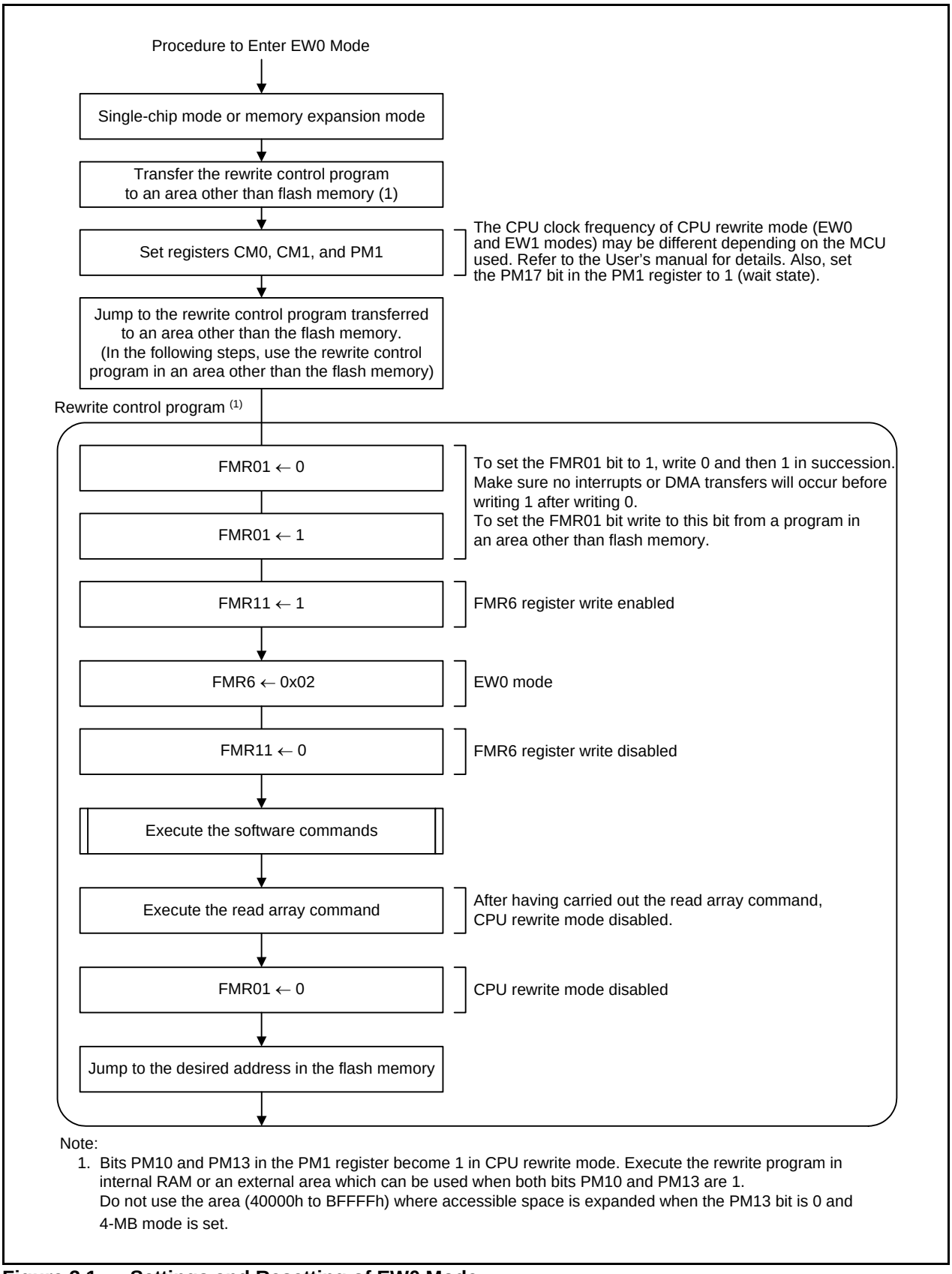
3. CPU Rewrite Mode

3.1 EW0 Mode Features

EW0 mode allows the user to rewrite the user ROM and the data areas by issuing program and erase commands generated from the CPU rewrite program already transferred to the RAM.
The CPU continues to operate during program and erase operations in EW0 mode. Peripheral function interrupts will be accepted during program and erase commands if the vector and the interrupting program are located in the RAM.

3.2 EW0 Mode Settings

The MCU enters CPU rewrite mode when the FMR01 bit in the FMR0 register is set to 1 (CPU rewrite mode enabled) and is ready to accept commands. EW0 mode is selected by setting the FMR60 bit in the FMR6 register to 0. Software commands control programming and erasing. The FMR0 register or status register indicates whether a program or erase operation is completed as expected or not.
Figure 3.1 shows Setting and Resetting of EW0 Mode.



Note:
1. Bits PM10 and PM13 in the PM1 register become 1 in CPU rewrite mode. Execute the rewrite program in internal RAM or an external area which can be used when both bits PM10 and PM13 are 1. Do not use the area (40000h to BFFFFh) where accessible space is expanded when the PM13 bit is 0 and 4-MB mode is set.

Figure 3.1 Settings and Resetting of EW0 Mode

3.3 Memory Map

The flash memory is used as ROM in this product. The flash memory comprises program ROM 1, program ROM 2, and data flash.
The flash memory is divided into several blocks, each of which can be protected (locked) from programming or erasing. The flash memory can be rewritten in CPU rewrite, standard serial I/O, and parallel I/O modes.
If the size of program ROM 1 is over 512 KB, blocks 8 to 11 can be used when the IRON bit in the PRG2C register is 1 (program ROM 1 addresses 40000h to 7FFFFh enabled).
Program ROM 2 can be used when the PRG2C0 bit in the PRG2C register is set to 0 (program ROM 2 enabled).
Program ROM 2 includes a user boot code area.
Data flash can be used when the PM10 bit in the PM1 register is set to 1 (0E000h to 0FFFFh: data flash).
Data flash is divided into block A and block B.
Figure 3.2 shows a Flash Memory Block Diagram for M16C/65.
Refer to the respective hardware manuals for other models.

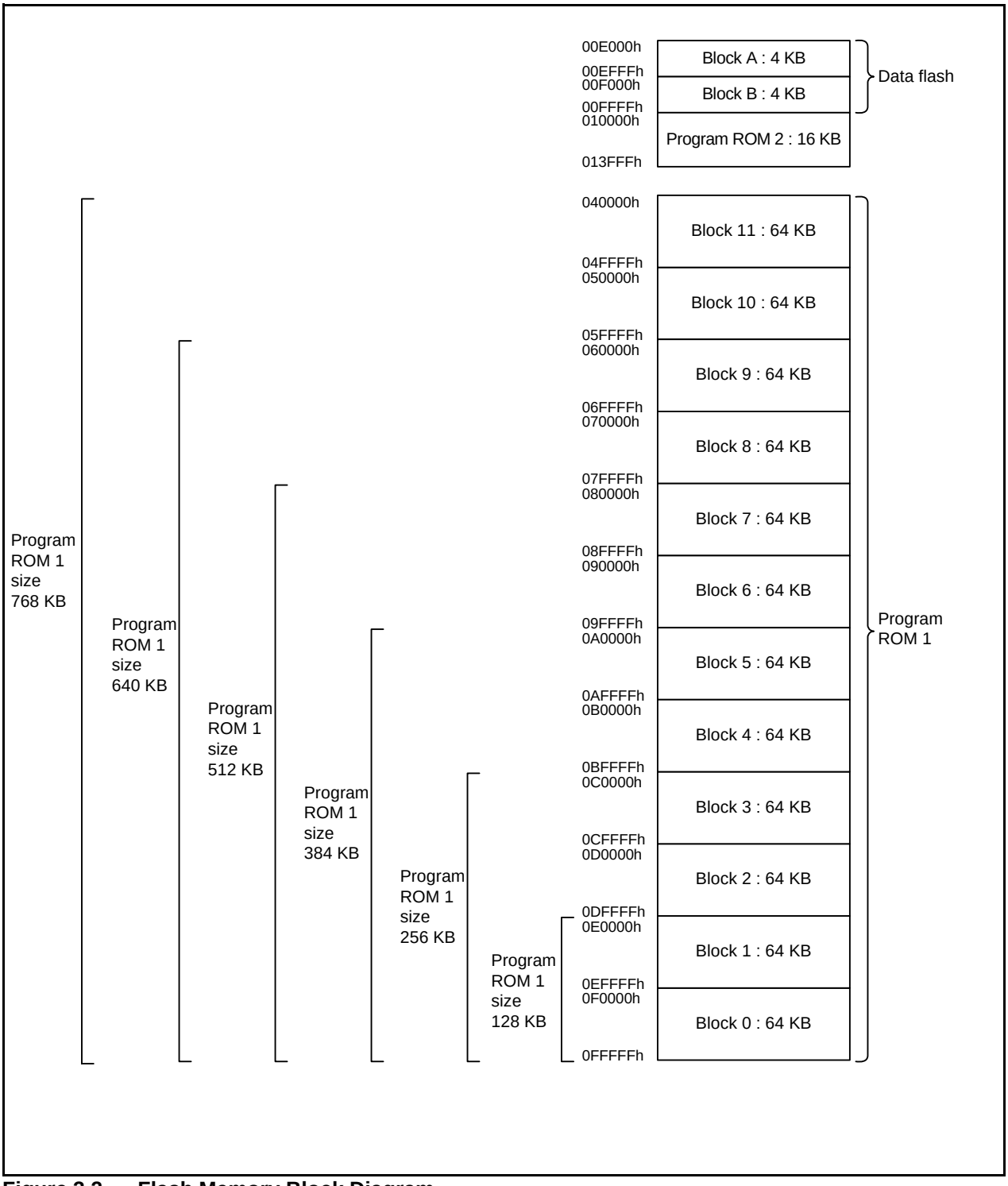


Figure 3.2 Flash Memory Block Diagram

4. Description of the Application Example

This application note provides a monitor program example where the sample program is received from the master device and the sample program execute and program ROM 2 area rewrite commands are executed.

Figure 4.1 shows the system structure.

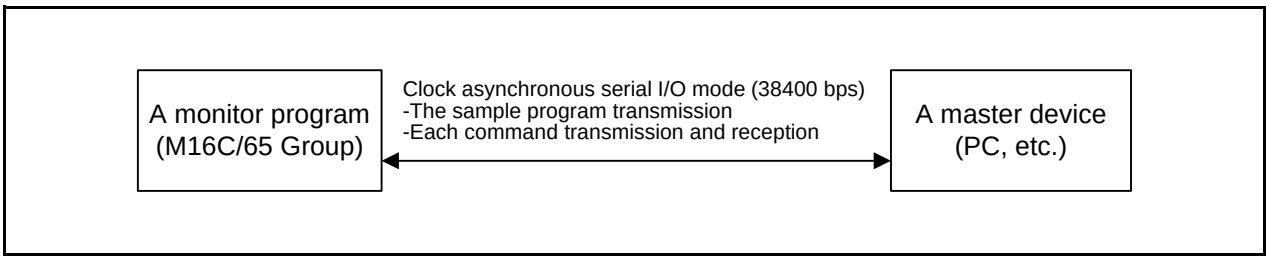


Figure 4.1 System Structure Diagram

The control commands used in this application note are as follows.

Table 4.1 Control Commands

Control Command Name	Command Explanation	1st - 3rd bytes	4th byte	5th byte	6th byte and beyond		
Program (write) command	Writes the sample program	"prg"	Size (2 bytes)		Data (max. 256 bytes)	Sum Value (2 bytes)	Results (1)
Erase command	Erases the program ROM 2 area	"ers"	Results (1)		Data and sum value transmission, and results receipt are repeated up to the size of the program.		
Sample program execute command	Executes the written sample program	"run"					

Master device → Transferred to the monitor program

Monitor Program → Transferred to the master device

Note:

1. When the program and erase have successfully completed, 6Fh ('o') is returned. If an error occurs, 65h ('e') is returned.

UART0 clock asynchronous serial I/O mode is used in communications with the master device. The UART0 settings are as follows.

Mode:	Clock asynchronous serial I/O mode
Communication bit rate:	38400 bps
CTS/RTS:	N/A
Stop bit:	1 stop bit
Parity:	None
Data bit length:	8 bits

The following explains the operations of this application note.

(1) The monitor program waits to receive the control command.
(2) When the received command is "prg"
  (2-1) Receive sample size (2-byte data).
  (2-2) Receive one packet (maximum 256 bytes) of program data.
  (2-3) Receive packet data sum value (2-byte data).
  (2-4) Compare received packet data sum value and the received sum value.
    (2-4-1) If no match, send error code to master device.
    (2-4-2) If they match, the CPU clock is set to 10 MHz or lower and one packet of data is written in the program ROM 2 area before returning the CPU clock to its original setting.
      (2-4-2-1) When the data has been successfully written, the write complete code is sent to the master device.
      (2-4-2-2) If a write error occurs, an error code is sent to the master device and program data receipt is stopped.
    (2-5) If an error does not occur, steps (2-2) through (2-4) are repeated until receipt of the sample program is completed.
(3) When the received command is "ers"
  (3-1) The CPU clock is set to 10 MHz or lower and the program ROM 2 area is erased before returning the CPU clock to its original setting.
    (3-2-1) When successfully erased, the erase complete code is sent to the master device.
    (3-2-2) If an erase error occurs, an error code is sent to the master device.
(4) When the received command is "run"
  (4-1) The sample program written in the program ROM 2 area is executed.

Figure 4.2. shows an example of the monitor program operation.

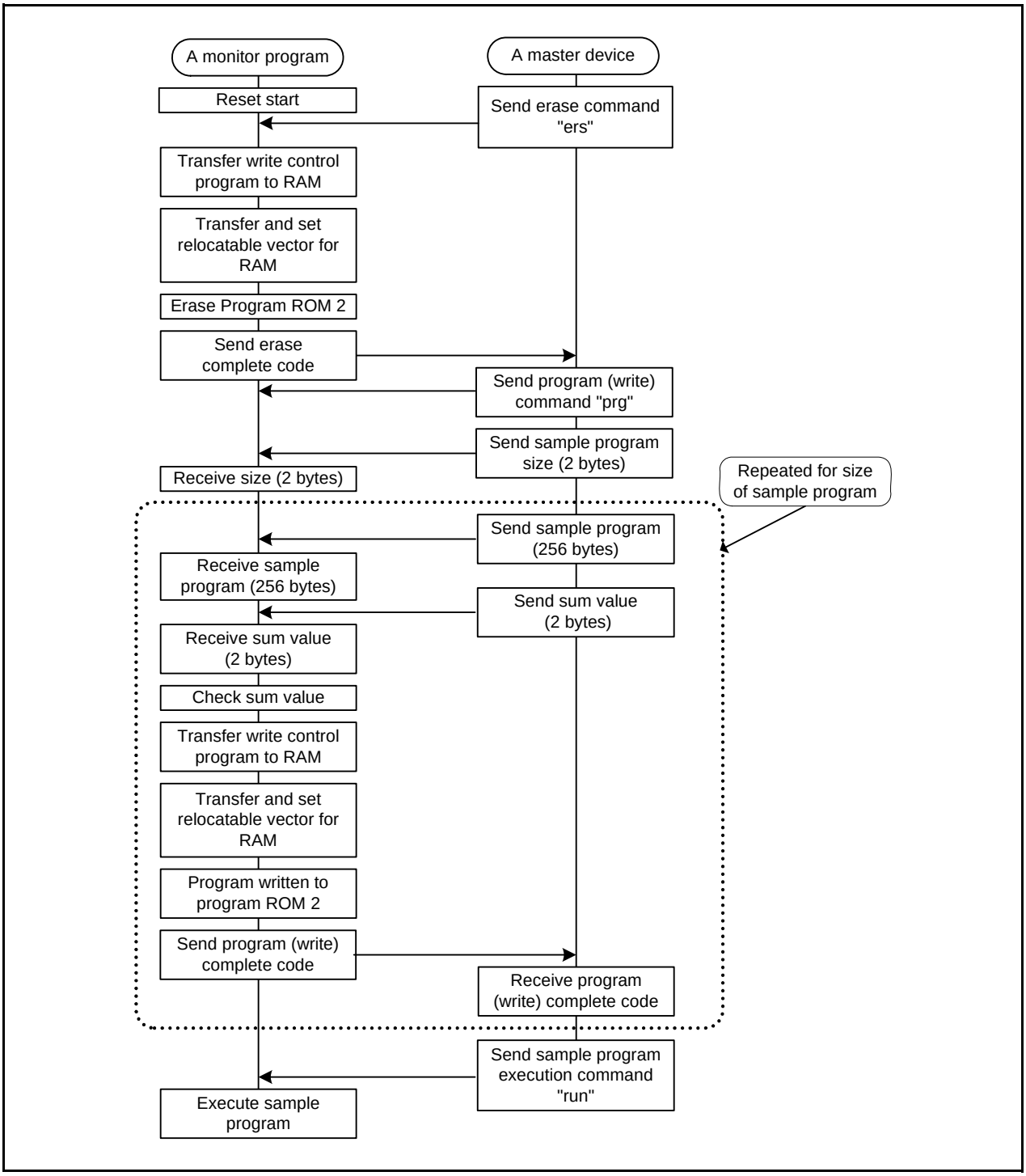


Figure 4.2 Monitor Program Operation Example

5. Structure

Declaration	typedef struct buff{ unsigned char prg_data[RECORD_SIZE]; unsigned short rev_sum; }REV_BUFF;	
Variable	unsigned char prg_data[RECORD_SIZE]	RECORD_SIZE (256) byte sample program storage array
	unsigned short rev_sum	Sum value storage variable
Function	Structure that stores the received sample program (256 bytes) and the sum value.	

6. Function Table

Declaration	void peripheral_init(void)
Outline	Peripheral function initial setting function
Argument	None
Variable(global)	None
Returned value	None
Function	Sets the UART0 send and receive setting and the timer A0 10 ms timer setting.

Declaration	void cpu_slow(void)
Outline	CPU slowdown process function
Argument	None
Variable(global)	None
Returned value	None
Function	Sets the main clock divider to CM06 = 0, CM17 - CM16 = 01 (divide-by-2 mode), PM17 = 1 (1 wait).

Declaration	void cpu_fast(void)
Outline	CPU speed up process function
Argument	None
Variable(global)	None
Returned value	None
Function	Sets the main clock divider to CM06 = 0, CM17-CM16 = 00 (no division mode), PM17 = 0 (no wait).

Declaration	unsigned char rev_byte(unsigned char *rev_data)		
Outline	Command 1-byte receive function		
Argument	Argument Type		Meaning
	unsigned char *rev_data		Receive command storage array address
Variable(global)	None		
Returned value	Return Value Type	Value	Meaning
	unsigned char	COMPLETE	Successfully completed
		TIMEOUT	Timeout
		RECEIVE_ERROR	Receive error
Function	Stores the 1 byte received data in the array.		

Declaration	unsigned char rev_cmd_check(unsigned char *cmd_buff)		
Outline	Command check function		
Argument	Argument Type		Meaning
	unsigned char *cmd_buff		Start address of the received command storage array
Variable(global)	None		
Returned value	Return Value Type	Value	Meaning
	unsigned char	REV_ERASE	Erase command received
		REV_PROGRAM	Program command received
		REV_RUN	Sample program execution command received
		REV_ERROR	Receive error
Function	Determines the received character string and returns the appropriate command.		

Declaration	void erase(void)
Outline	Flash erase function
Argument	None
Variable(global)	None
Returned value	None
Function	Executes the block erase function located on the RAM, determines if the erase was successful and sends a message.

Declaration	void receive_program(void)	
Outline	Flash write function	
Argument	None	
Variable(global)	Variable Name	Contents
	REV_BUFF rb	Structure that stores the received program data and the sum value
Returned value	None	
Function	Receives the sample program size, program data and sum value sent from the master device. Executes the program function located on the RAM and writes the received program data. Determines if it the write was received correctly and sends a message.	

Declaration	unsigned short rev_size(void)		
Outline	Sample program size receive function		
Argument	None		
Variable(global)	None		
Returned value	Return Value Type	Value	Meaning
	unsigned short	rev_size	Received size data
Function	Receives the sample program size sent from the master device.		

Declaration	unsigned char rev_data(REV_BUFF *buff,unsigned short *size)		
Outline	Sample program data receive function		
Argument	Argument Type		Meaning
	REV_BUFF *buff		Start address of the receive data storage structure
	unsigned short *size		Address of the variable where the size is stored
Variable(global)	Variable Name		Content
	REV_BUFF rb		Stores sum value
Returned value	Return Value Type	Value	Meaning
	unsigned char	COMPLETE	Receive successful
		FAIL	Receive failed
Function	Receives 256 bytes sample program data and the sum value. Compares the received packet data sum value and the received sum value. When the received data is below the record size, the remaining space is filled with 0xFFh.		

Declaration	void note_program_start(void)
Outline	Sample program execute function
Argument	None
Variable(global)	None
Returned value	None
Function	Executes the sample program written in the 10000h address.

Declaration	void send_message(const unsigned char *mess)	
Outline	Message send function	
Argument	Argument Type	Meaning
	const unsigned char mess*	Start address of the send message array
Variable(global)	None	
Returned value	None	
Function	Sends a message.	

Declaration	void send_to_ram(void)
Outline	Write control program transfer function
Argument	None
Variable(global)	None
Returned value	None
Function	Transfers the erase function, program function and the full status check function to the RAM.

Declaration	void send_to_ram_vector(void)
Outline	Interrupt handler for RAM transfer function
Argument	None
Variable(global)	None
Returned value	None
Function	Transfers the interrupt handler to use on the RAM.

Declaration	void renewal_of_ram_vector(void)		
Outline	Relocatable vector table for RAM create function		
Argument	None		
Variable(global)	None		
Returned value	None		
Function	Create the relocatable vector table for the RAM.		

Declaration	unsigned char block_erase_ew0(unsigned short far* ers_addr)		
Outline	Block erase function		
Argument	Argument Type		Meaning
	unsigned short far* ers_addr		Address of block to be erased
Variable(global)	None		
Returned value	Return Value Type	Value	Meaning
	unsigned char	COMPLETE	Successfully completed
		CMD_SEQ_ERR	Command sequence error
		PROGRAM_ERR	Program write error
		ERASE_ERR	Erase error
Function	Erases the specified block in EW0 mode and executes a full status check. Returns the appropriate error message when an error occurs.		

Declaration	unsigned char program_write_ew0 (unsigned short far* write_addr,unsigned short *buff)		
Outline	Program function		
Argument	Argument Type		Meaning
	unsigned short far* write_addr		Start address of write destination
	unsigned short *buff		Data to be written (256 bytes)
Variable(global)	None		
Returned value	Return Value Type	Value	Meaning
	unsigned char	COMPLETE	Successfully completed
		CMD_SEQ_ERR	Command sequence error
		PROGRAM_ERR	Program write error
		ERASE_ERR	Erase error
Function	Writes 256 bytes of data from the specified address in EW0 mode.		

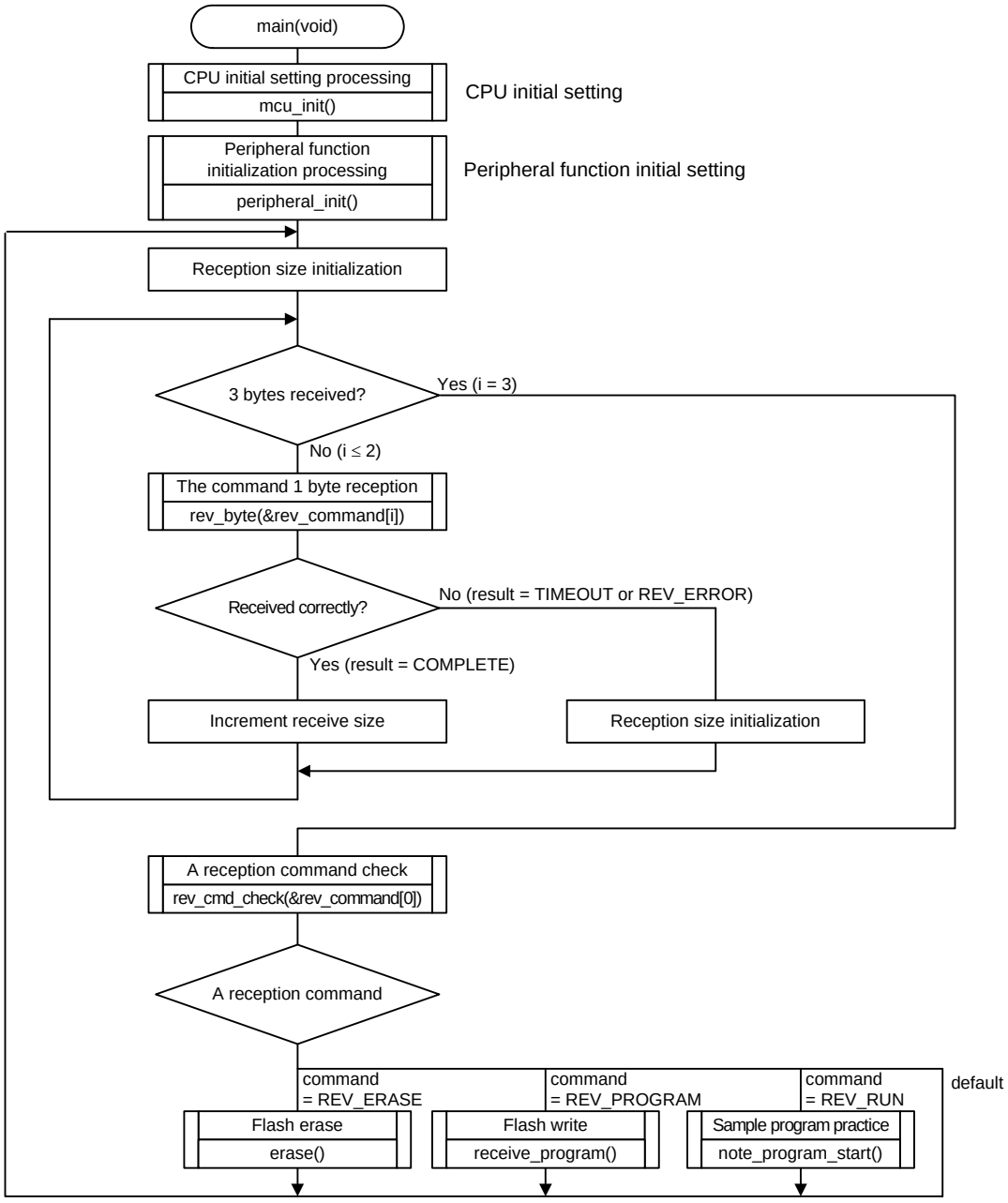
Declaration	unsigned char full_status_check(void)		
Outline	Full status check function		
Argument	None		
Variable(global)	None		
Returned value	Return Value Type	Value	Meaning
	unsigned char	COMPLETE	Successfully completed
		CMD_SEQ_ERR	Command sequence error
		PROGRAM_ERR	Program write error
		ERASE_ERR	Erase error
Function	Executes a full status check and returns the results.		

Declaration	void asm_smovf(void _far *_source, void _near *_dest, unsigned int _size)	
Outline	RAM transfer process function	
Argument	Argument Type	Meaning
	void_far *_source	Source address (program)
	void_near *_dest	Destination address (RAM area)
	unsigned int _size	Transfer size
Variable(global)	None	
Returned value	None	
Function	Transfers the specified area to the RAM area.	

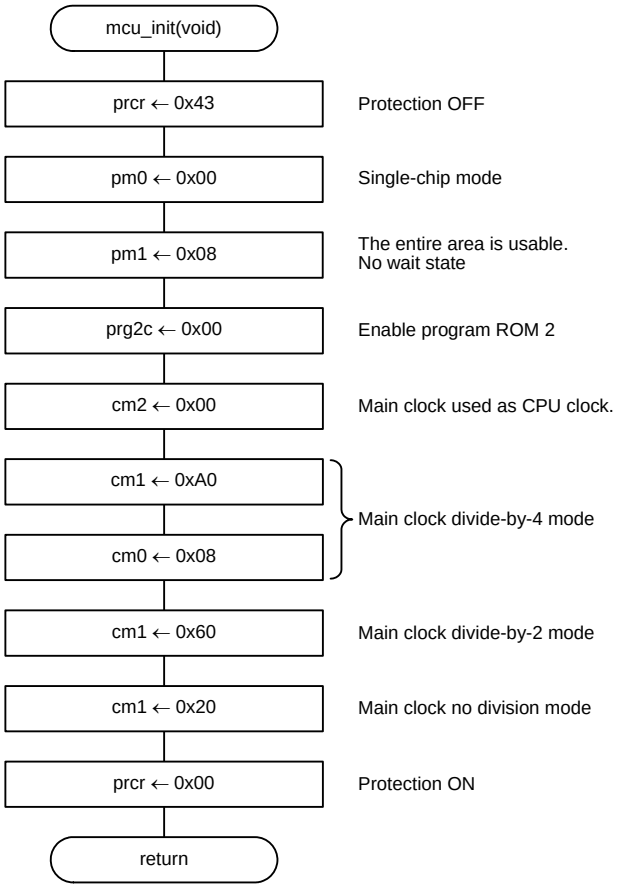
Declaration	void ram_int_dummy(void)
Outline	Interrupt handler for RAM of dummy function
Argument	None
Variable(global)	None
Returned value	None
Function	Dummy function for the RAM. Add program if needed.

7. Flowchart

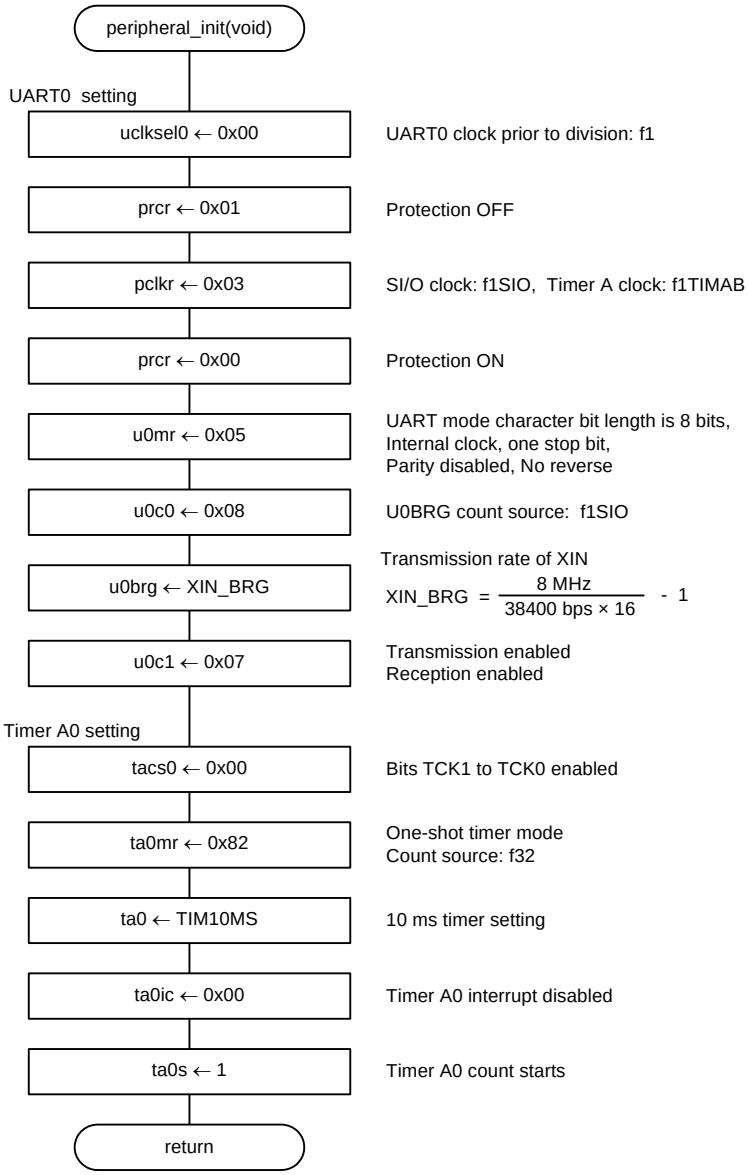
7.1 Main Function



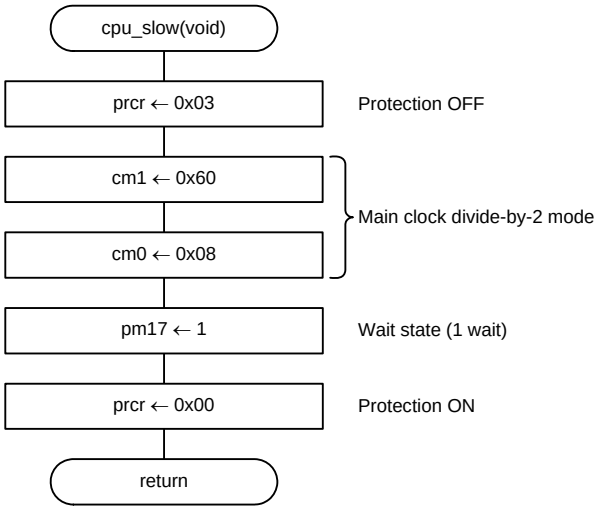
7.2 CPU Initial Setting Function



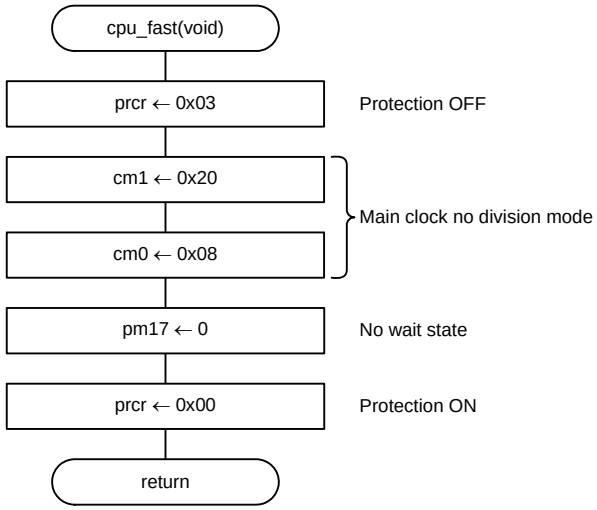
7.3 Peripheral Function Initial Setting Function



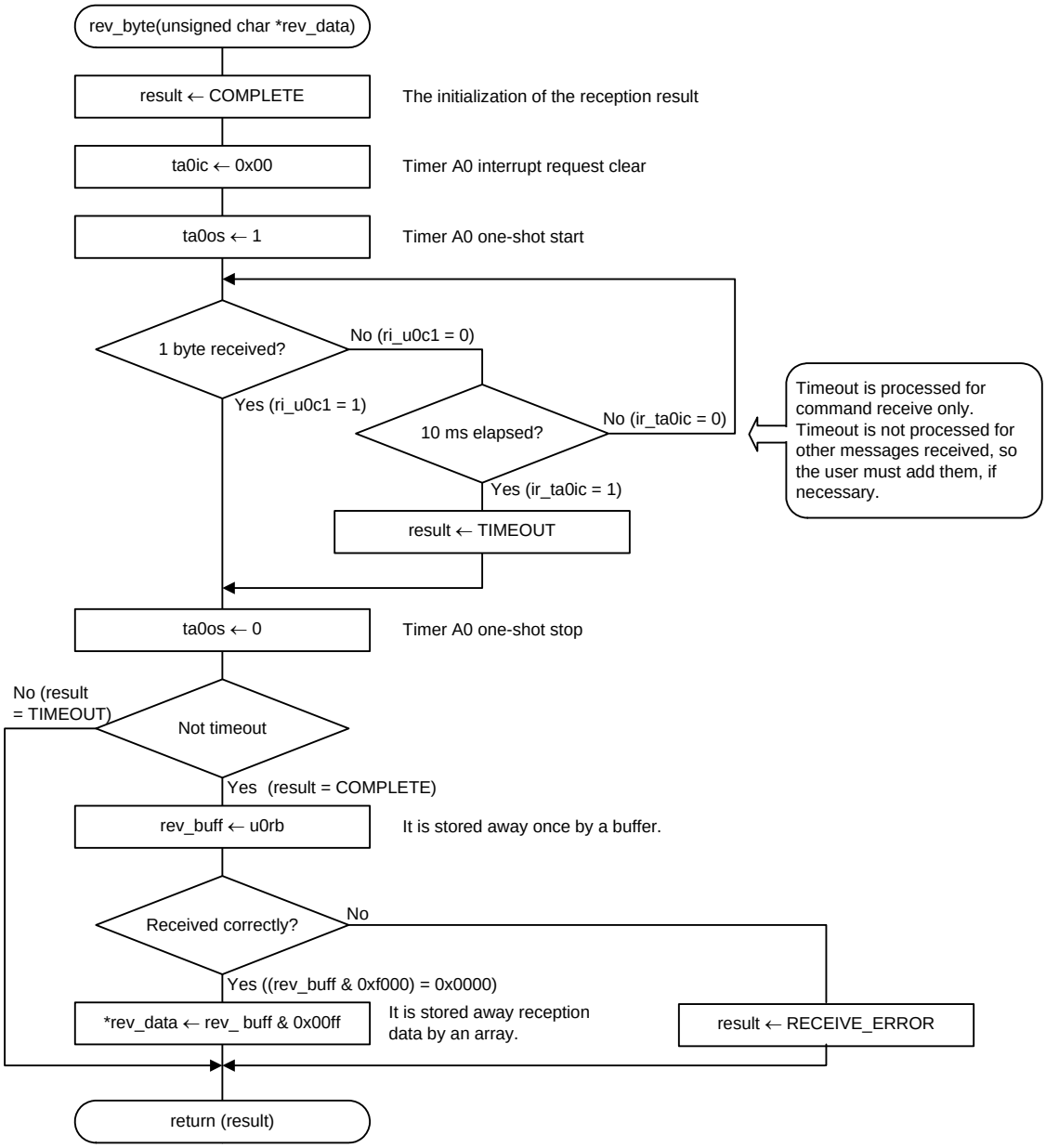
7.4 CPU Slowdown Process Function



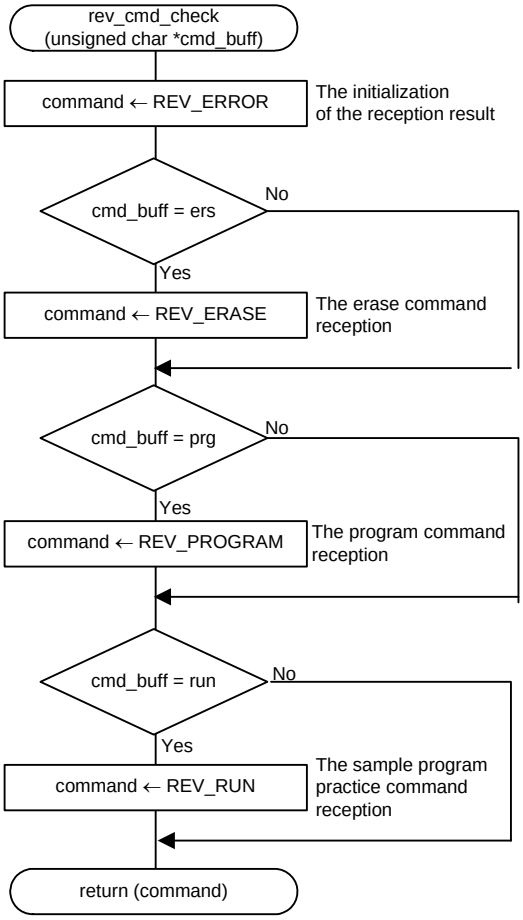
7.5 CPU Speed Up Process Function



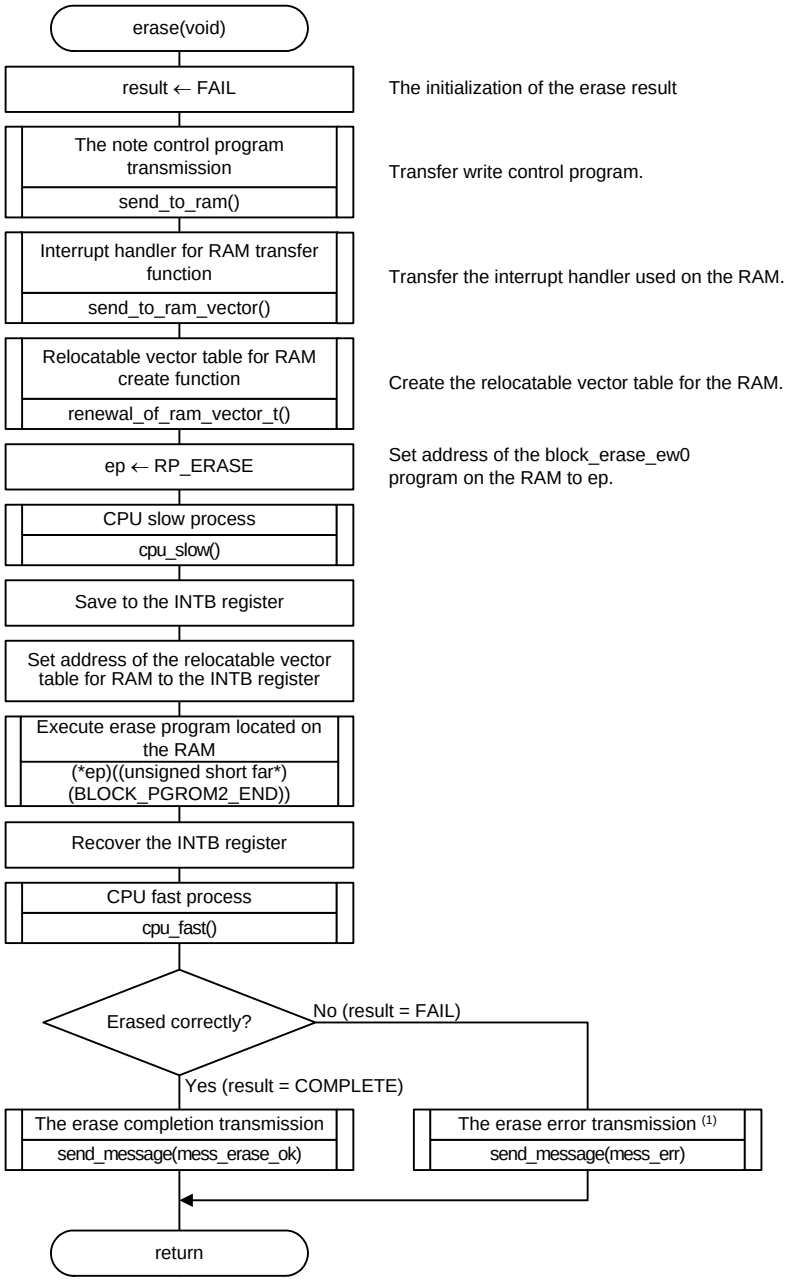
7.6 Command 1-byte Receive Function



7.7 Command Check Function

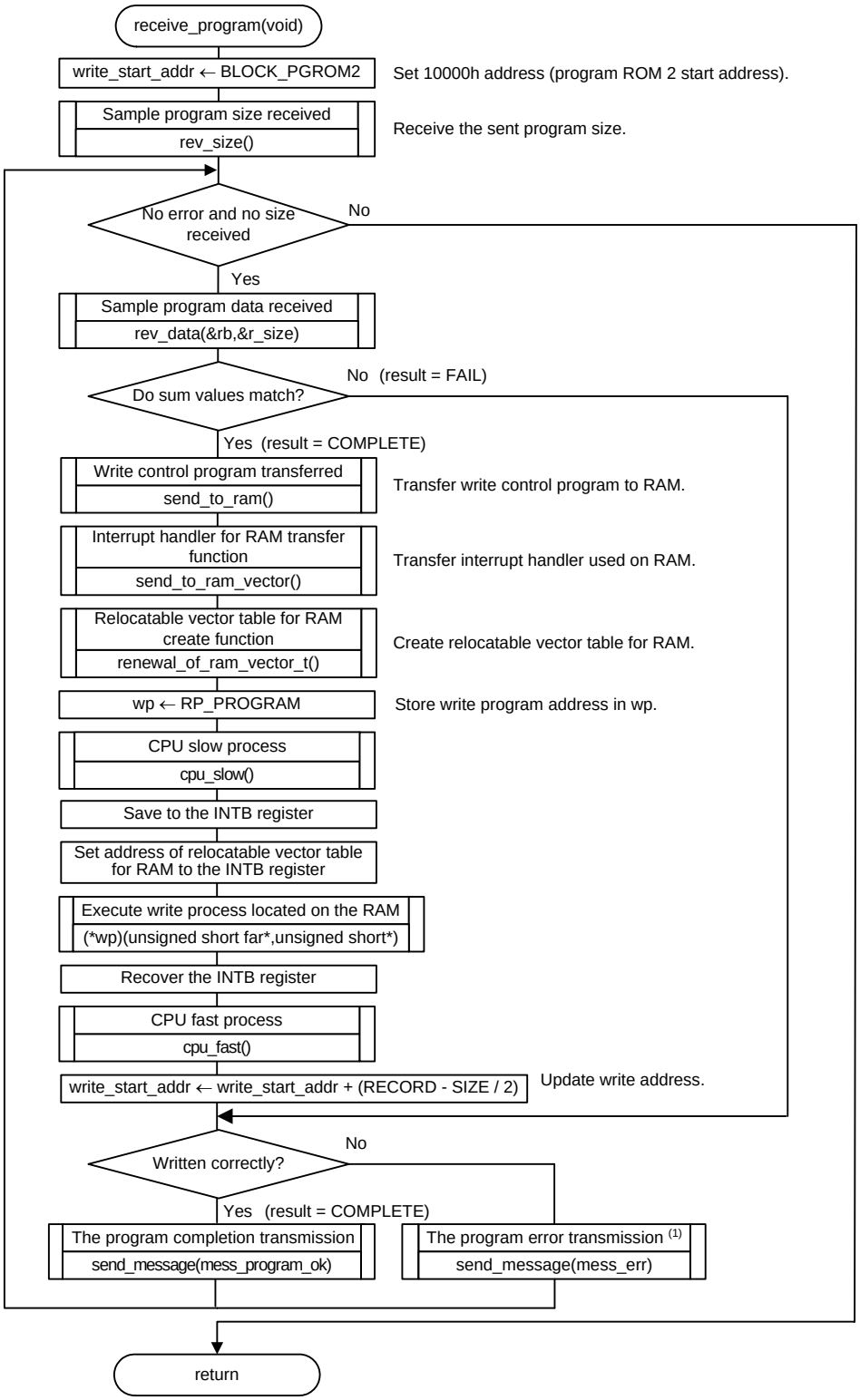


7.8 Flash Erase Function



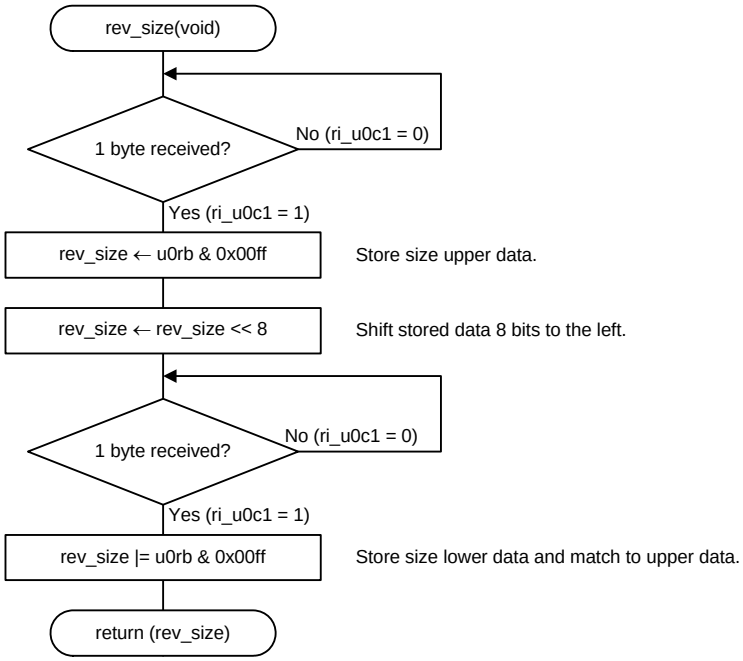
Note:
1. This application note only sends a message when an error occurs.
   Add error processes as necessary.

7.9 Flash Write Function

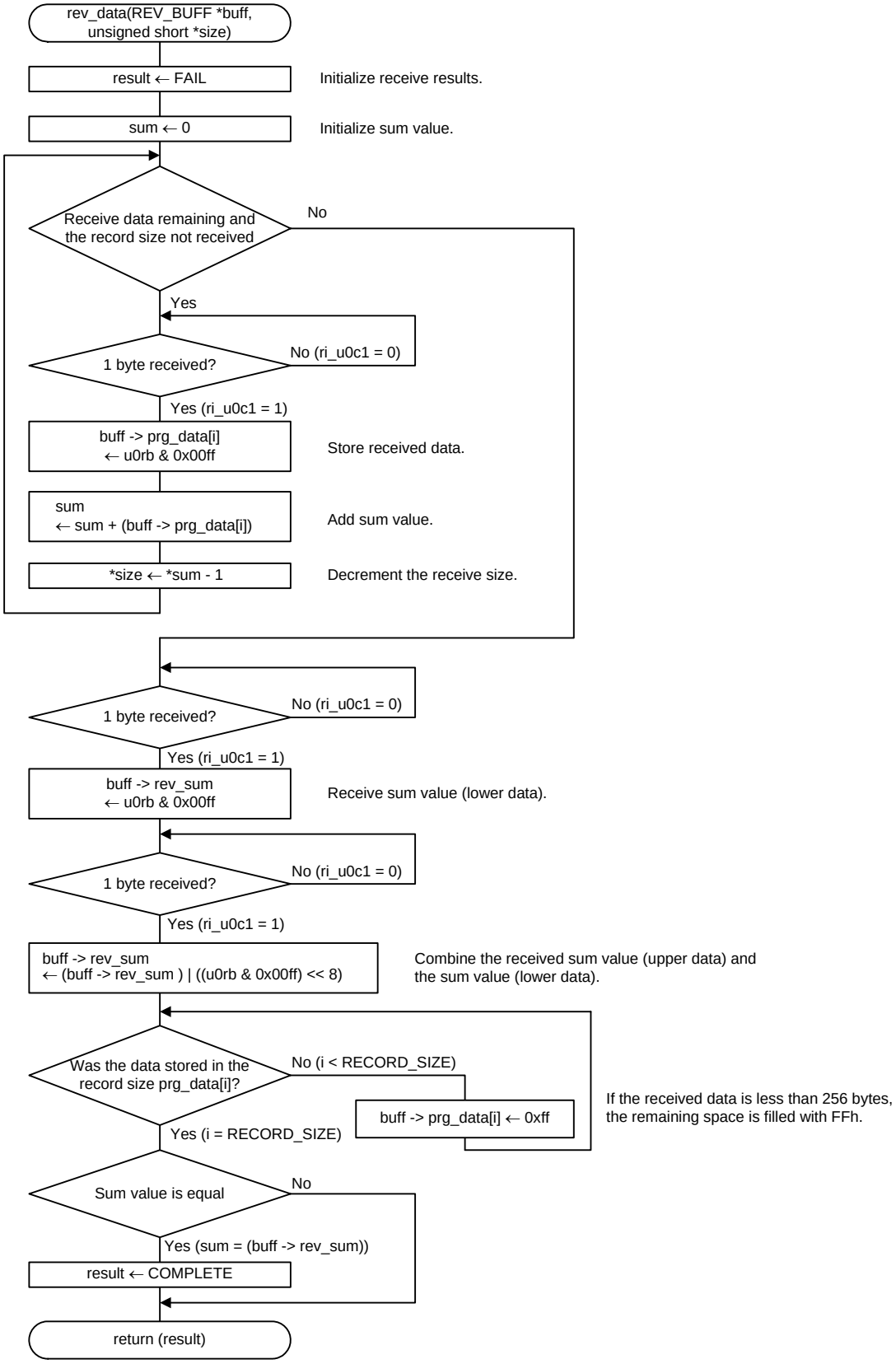


Note:
1. This application note only sends a message when an error occurs.
   Add error processes as necessary.

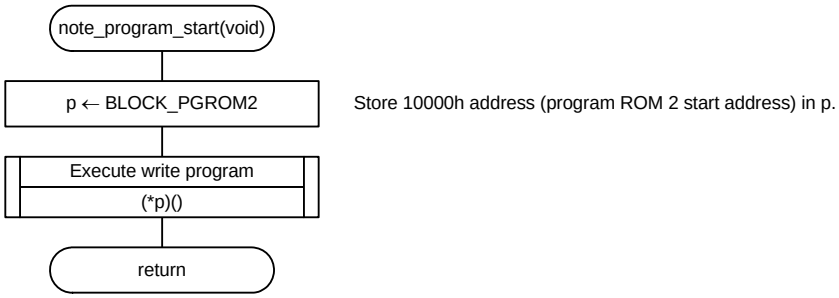
7.10 Sample Program Size Receive Function



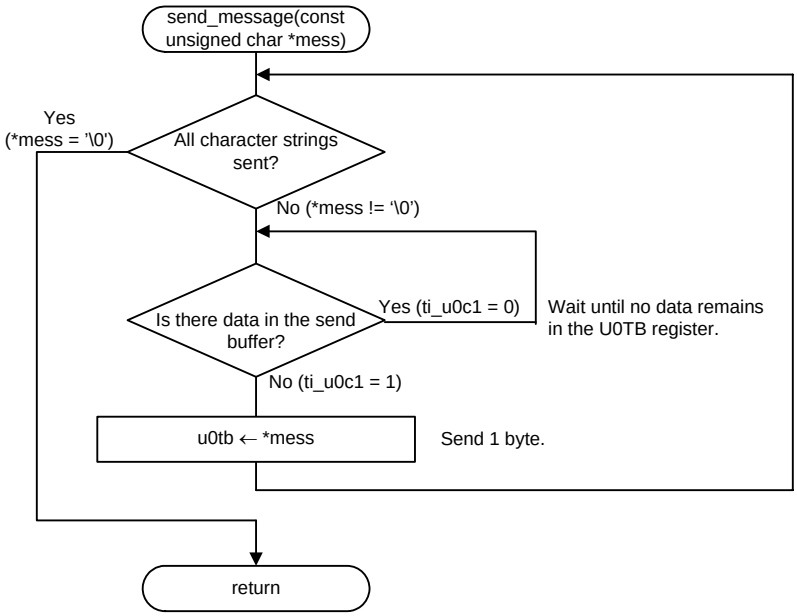
7.11 Sample Program Data Receive Function



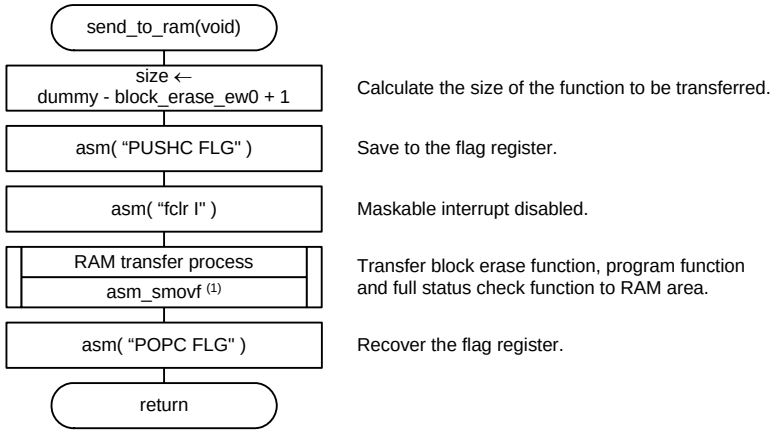
7.12 Sample Program Execute Function



7.13 Message Send Function

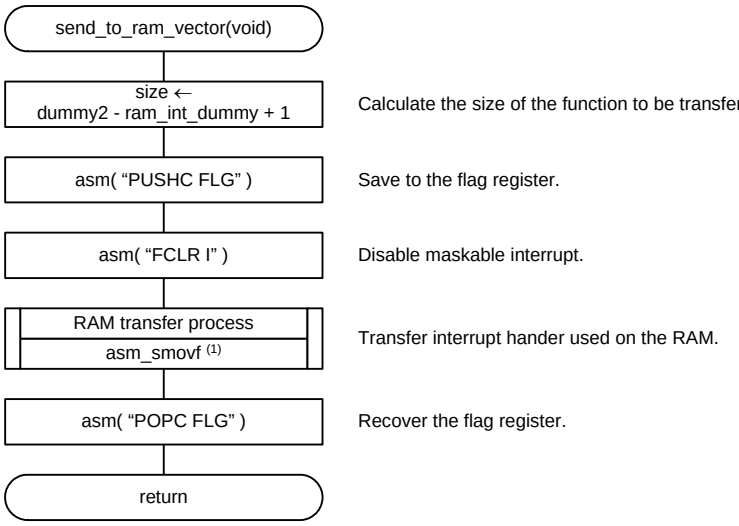


7.14 Write Control Program Transfer Function



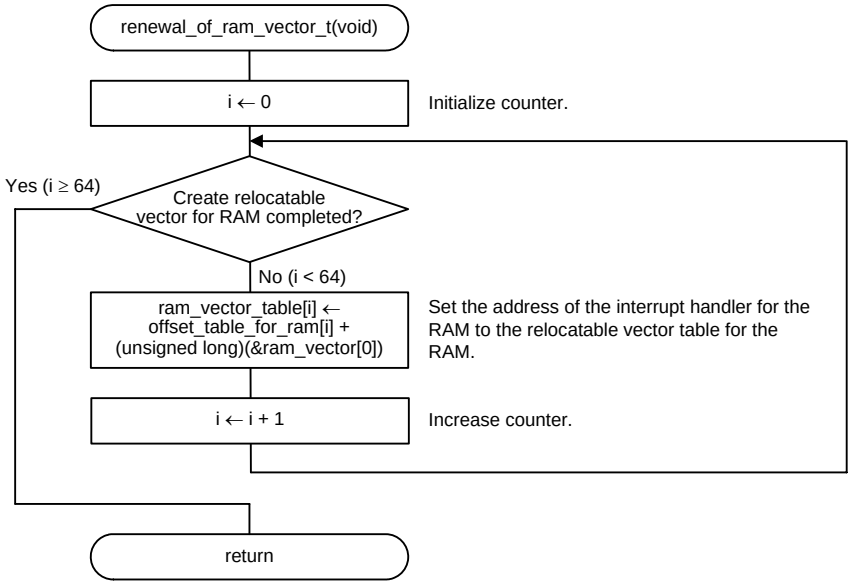
Note:
1. ((void far *) block_erase_ew0, (void near *)ram_p, ((unsigned short) size / 2))

7.15 Interrupt Handler for RAM Transfer Function

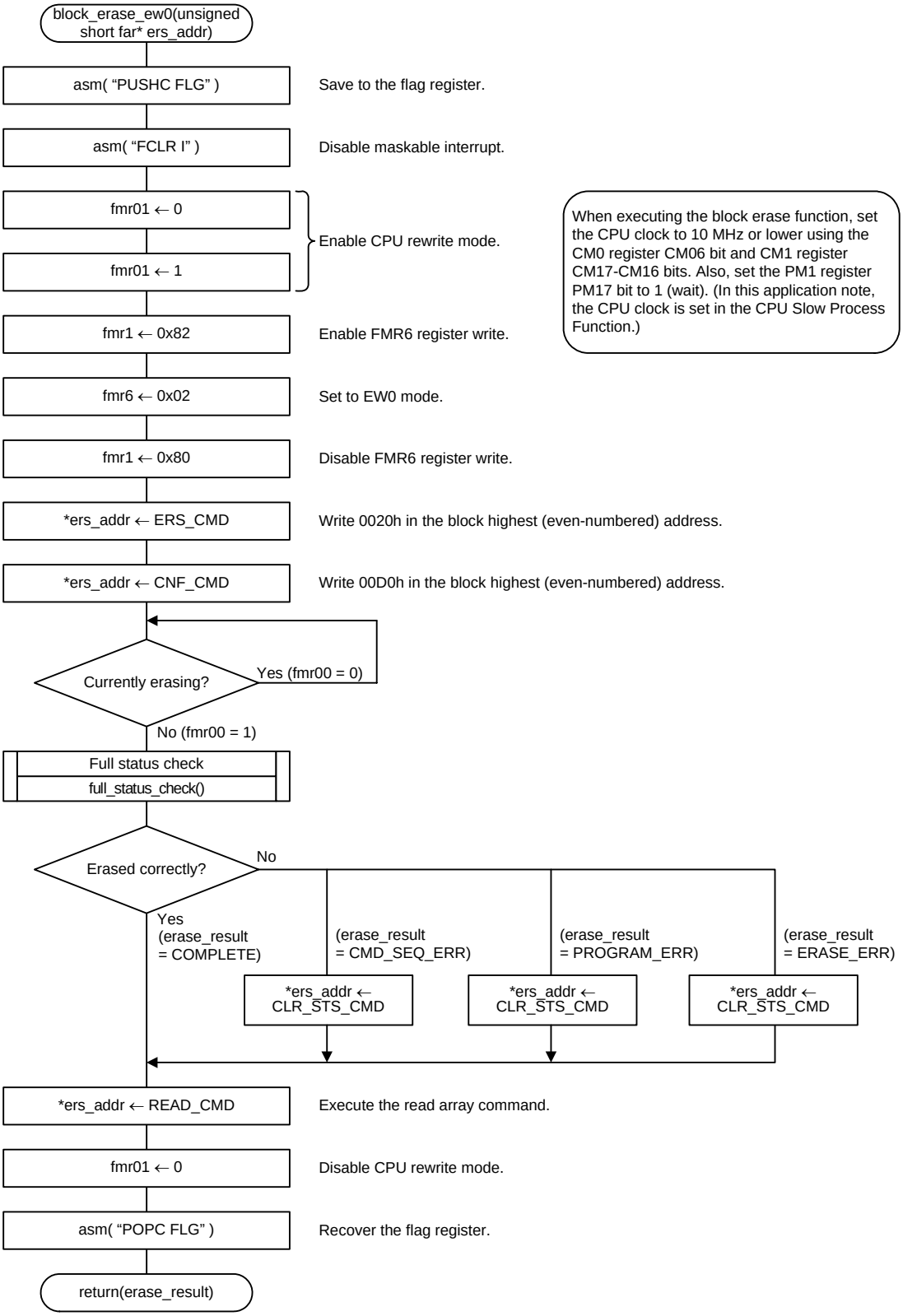


Note:
1. ((void far *)ram_int_dummy, (void near *)ram_vector, ((unsigned short)size / 2))

7.16 Relocatable Vector Table for RAM Create Function

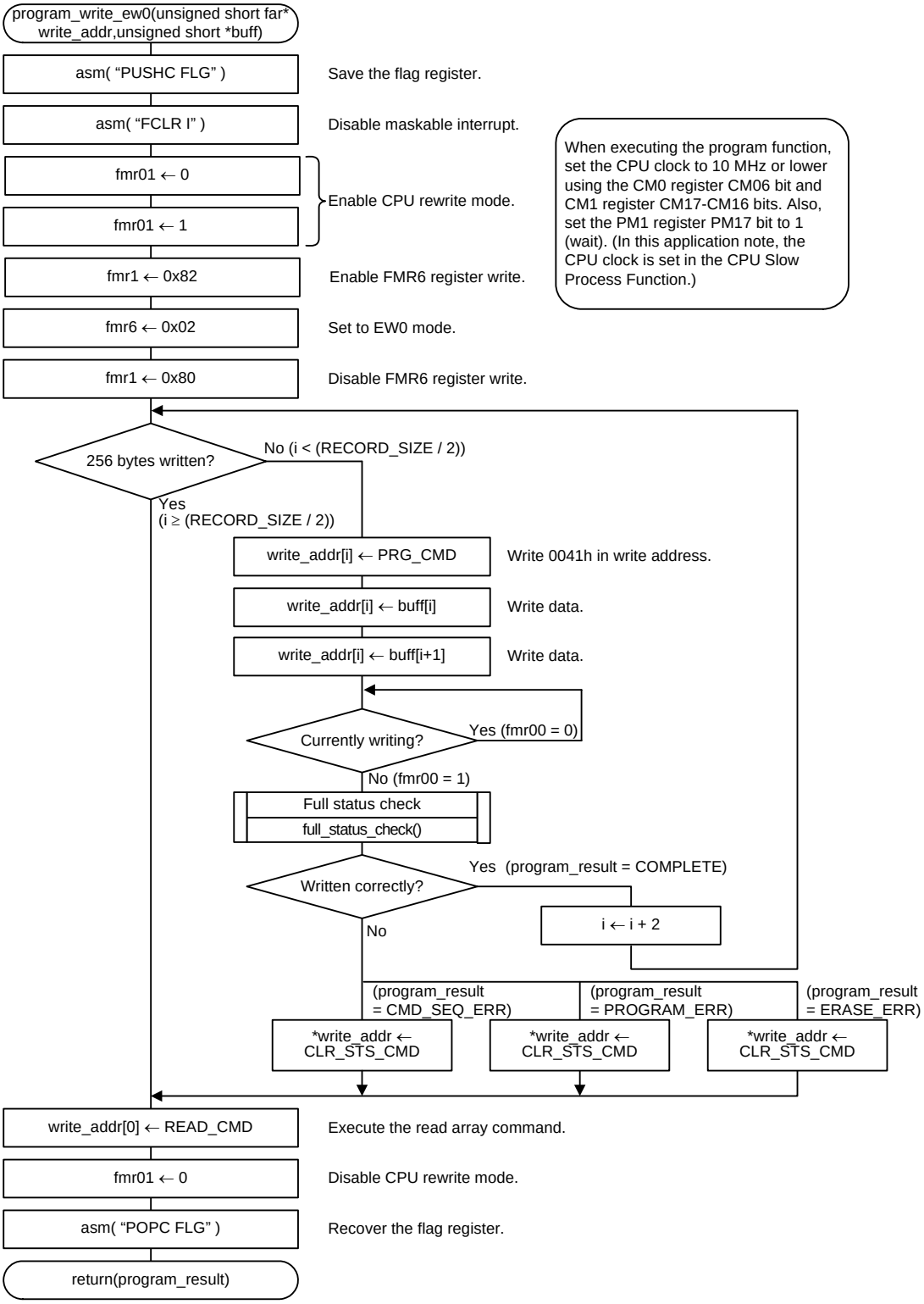


7.17 Block Erase Function



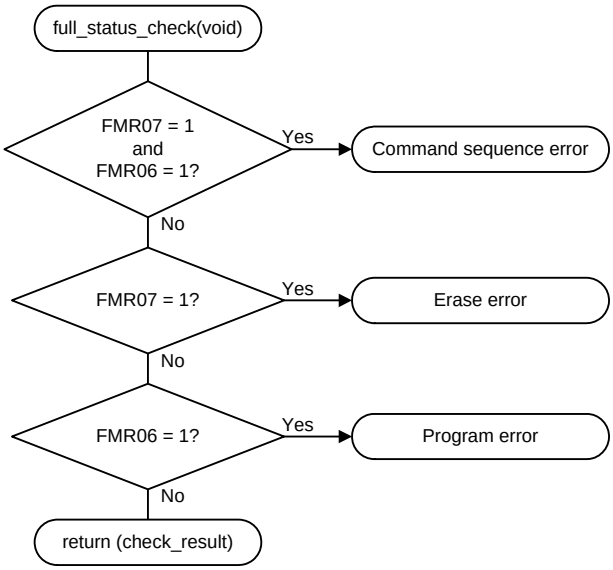
Note:
1. CPU clock frequencies that can be used in CPU rewrite mode (EW0 and EW1 modes) differ for each product. Refer to the user’s manual for details.

7.18 Program Function

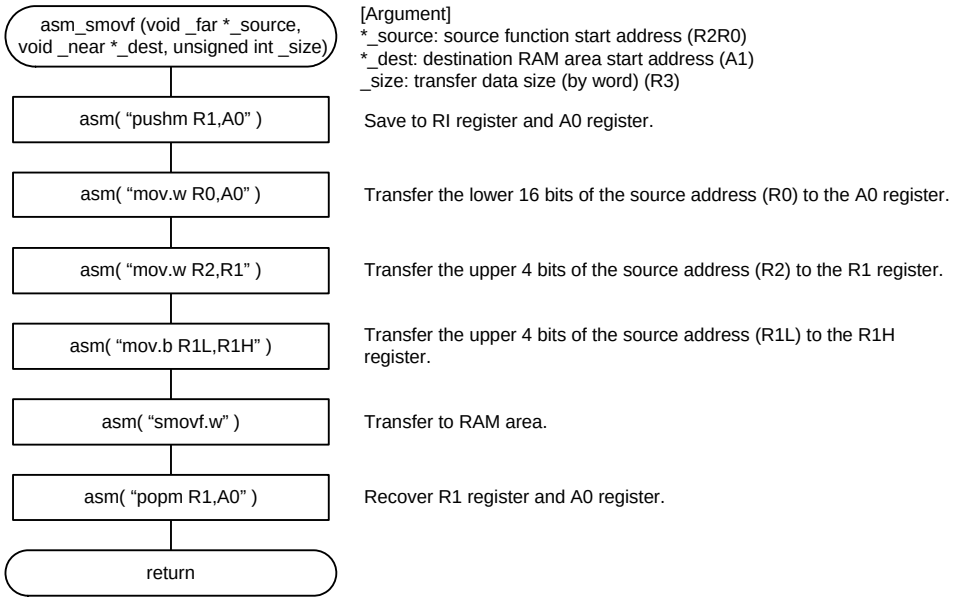


Note:
1. The frequency of the CPU clock used for the CPU rewrite mode (EW0 and EW1 modes) may be different depending on MCUs. Refer to the User’s manual for details.

7.19 Full Status Check Function



7.20 RAM Transfer Process Function



8. 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 M16C Family page.

9. Reference Documents

M16C/63 Group User’s Manual: Hardware Rev.1.00
M16C/64 Group User’s Manual: HardwareRev.1.05
M16C/64A Group User’s Manual: Hardware Rev.1.10
M16C/64C Group User’s Manual: Hardware Rev.0.10
M16C/65 Group User’s Manual: Hardware Rev.1.10
M16C/65C Group User’s Manual: Hardware Rev.0.10
M16C/6C Group User’s Manual: Hardware Rev.1.00
M16C/5LD Group, M16C/56D Group User’s Manual: Hardware Rev.1.10
M16C/5L Group, M16C/56 Group User’s Manual: Hardware Rev.1.00
M16C/5M Group, M16C/57 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.

C Compiler Manual
M16C Series, R8C Family C Compiler Package V.5.45
C Compiler User’s Manual Rev.2.00
The latest version 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	M16C/63, 64, 64A, 64C, 65, 65C, 6C, 5LD, 56D, 5L, 56, 5M, and 57 Groups The Use Example of CPU Rewrite Mode (EW0 mode)

Rev.	Date	Description	
		Page	Summary
1.00	Sep. 02, 2009	—	First edition issued
1.01	Oct. 14, 2009	27, 28	The processing of the read array command execution is added
1.02	Jan. 29, 2010	26	Updated the flowchart of "7.14 Write Control Program Transfer Function"
1.03	Jan. 31, 2011	—	M16C/5L, M16C/5LD, M16C/5M, M16C/64C, M16C/65C, M16C/56D, M16C/56, and M16C/57 Groups are added
		—	Partially modified
		1	The condition of M16C/65 Group can be used is added in “2. Introduction”
		3	The explanation in “Figure 3.1 Settings and Resetting of EW0 Mode” is devised
		6	The explanation in “Figure 4.1 System Structure Diagram” is added
		8	The processing of “Transfer and set relocatable vector for RAM” command is added
		12	Function table of Interrupt handler for RAM transfer function is added
		13	Function table of Relocatable vector table for RAM create function is added
		14	Function table of Interrupt handler for RAM of dummy function is added
		21	The processing of Interrupt handler for RAM transfer function, Relocatable vector table for RAM create function, and Save to the INTB register commands are added
		22	The processing of Interrupt handler for RAM transfer function, Relocatable vector table for RAM create function, and Save to the INTB register commands are added
		25	In “7.14 Write Control Program Transfer Function”: The processing of Save to the flag register and Recover to the flag register commands are added The handler of Interrupt enabled command is deleted
		26	“7.15 Interrupt Handler for RAM Transfer Function” and “7.16 Relocatable Vector Table for RAM Create Function” are added
		27	Note 1 is added in “7.17 Block Erase Function” The processing of Save to the flag register, Disable maskable interrupt, and Recover to the flag register commands are added Devised from “fmr0 ←0x00” to “fmr01 ← 0 Devised from “fmr0 ←0x02” to “fmr01 ← 1
		28	Note 1 is added in “7.18 Program Function” The processing of Save to the flag register, Disable maskable interrupt, and Recover to the flag register commands are added Devised from “fmr0 ← 0x00” to “fmr01 ← 0 Devised from “fmr0 ← 0x02” to “fmr01 ← 1

All trademarks and registered trademarks are the property of their respective owners.

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

© 2011 Renesas Electronics Corporation. All rights reserved.

Colophon 1.0