SH7216 Group

Using User Program Mode

R01AN0388EJ0101
Rev. 1.01
Jun. 15, 2012

Summary

This application note describes an example to run the flash memory reprogramming program in SH7216 microcomputers (MCUs) user program mode. An external device which is connected to the SH7216 stores the data to write to the flash memory, and communicates with the flash memory using the Serial Communication Interface with FIFO.

The flash memory reprogramming program described in this application note is stored on the SH7216 user MAT. The simple flash API for SH2 and SH2A (Standard API) provided by the Renesas Electronics is used to reprogram the flash memory.

Target Device

SH7216 MCU

Contents

1. Introduction

1.1 Specifications

This application programs, erases, and reads the flash memory using user program mode. User program mode handles programming, erasing, and reading with a desired interface. This application uses the serial communication between the host computer and the SH7216 to handle these processing.

When the SH7216 receives the flash memory reprogramming/erasing command (user control command) from the host computer while executing the user application, the SH7216 programs or erases the flash memory. When it receives the flash memory reading command from the host computer, it reads the flash memory.

Figure 1 shows the system configuration of this application.

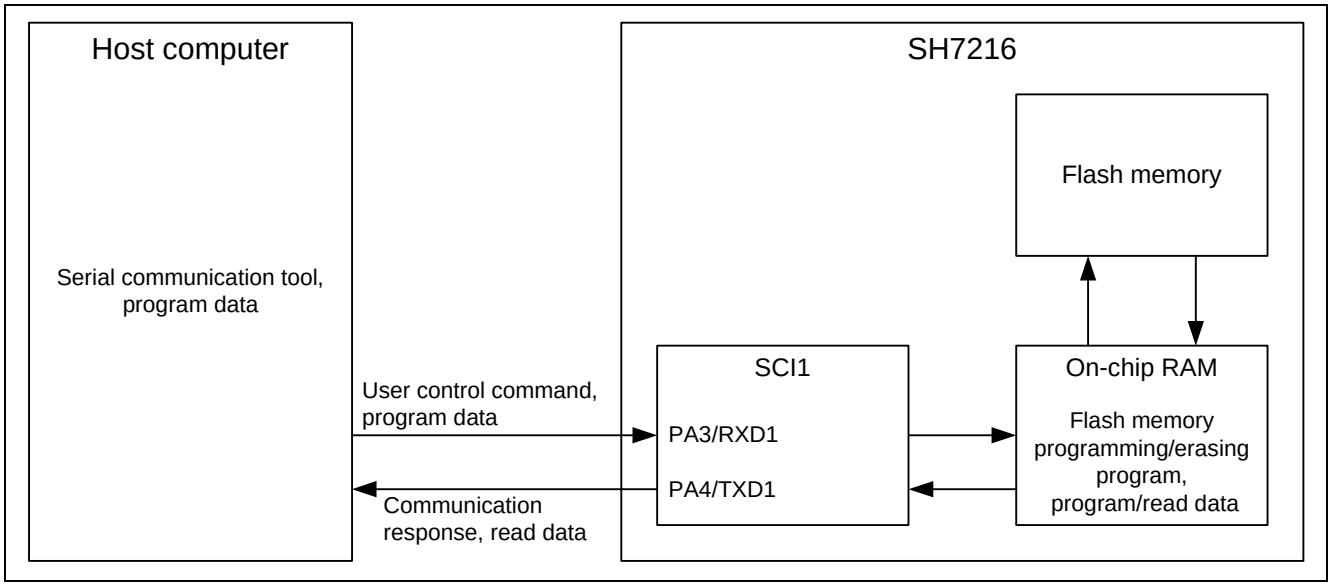


Figure 1 System Configuration

1.2 Modules Used

- Serial Communication Interface (SCI)
- On-chip Flash-dedicated Sequencer (FCU)

1.3 Applicable Conditions

MCU	SH7216 (R5F72167A: 1-MB flash memory version)
Operating Frequency	Internal clock: 200 MHz Bus clock: 50 MHz Peripheral clock: 50 MHz
Integrated Development Environment	Renesas Electronics Corporation High-performance Embedded Workshop Ver.4.07.00
C Compiler	Renesas Electronics SuperH RISC engine Family C/C++ compiler package Ver.9.03 Release 00
Compiler Options	Default setting in the High-performance Embedded Workshop (-cpu=sh2a -debug -gbr=auto -global_volatile=0 -opt_range=all -infinite_loop=0 -del_vacant_loop=0 -struct_alloc=1)

1.4 Related Application Notes

For more information, refer to the following application notes:

- SH7216 Group Example of Initialization
- SH Family Simple Flash API for SH2 and SH2A

2. Overview

This application uses the Serial Communication Interface (SCI) to connect the SH7216 with the external device. SH7216 uses a dedicated sequencer (FCU) to program or erase the on-chip flash memory.

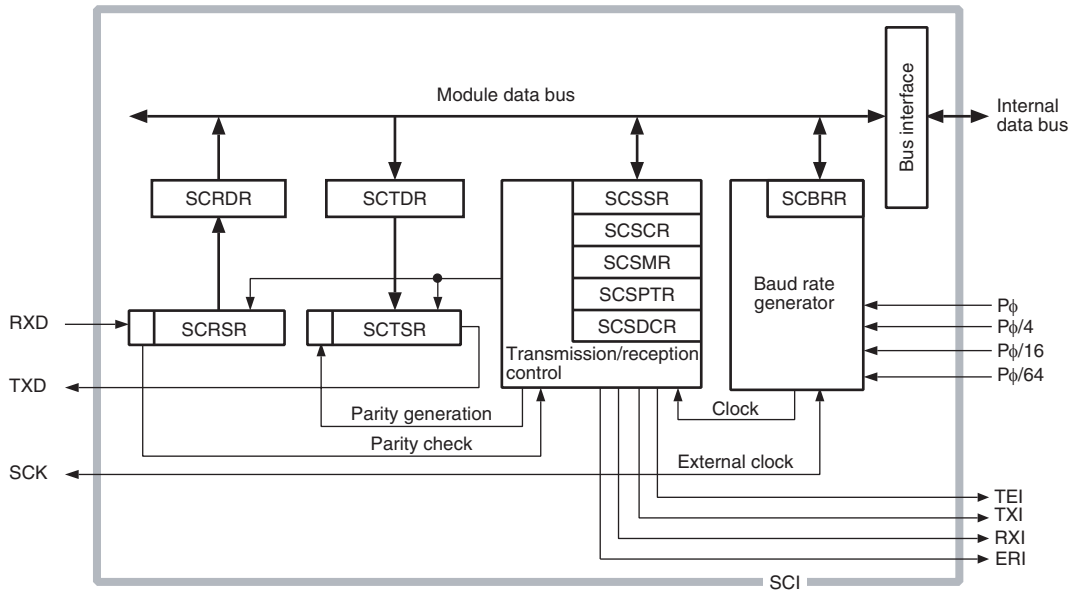
2.1 Overview of Modules

2.1.1 Serial Communication Interface (SCI)

SCI supports both asynchronous and clocked synchronous serial communication. It also supports full-duplex communication and allows double-buffering both at transmitter and receiver to transmit/receive the serial data continuously at high speed.

This application uses the SCI for the handshake between the SH7216 and an external device, and to transmit/receive the flash memory reprogram data.

Figure 2 shows the SCI block diagram.



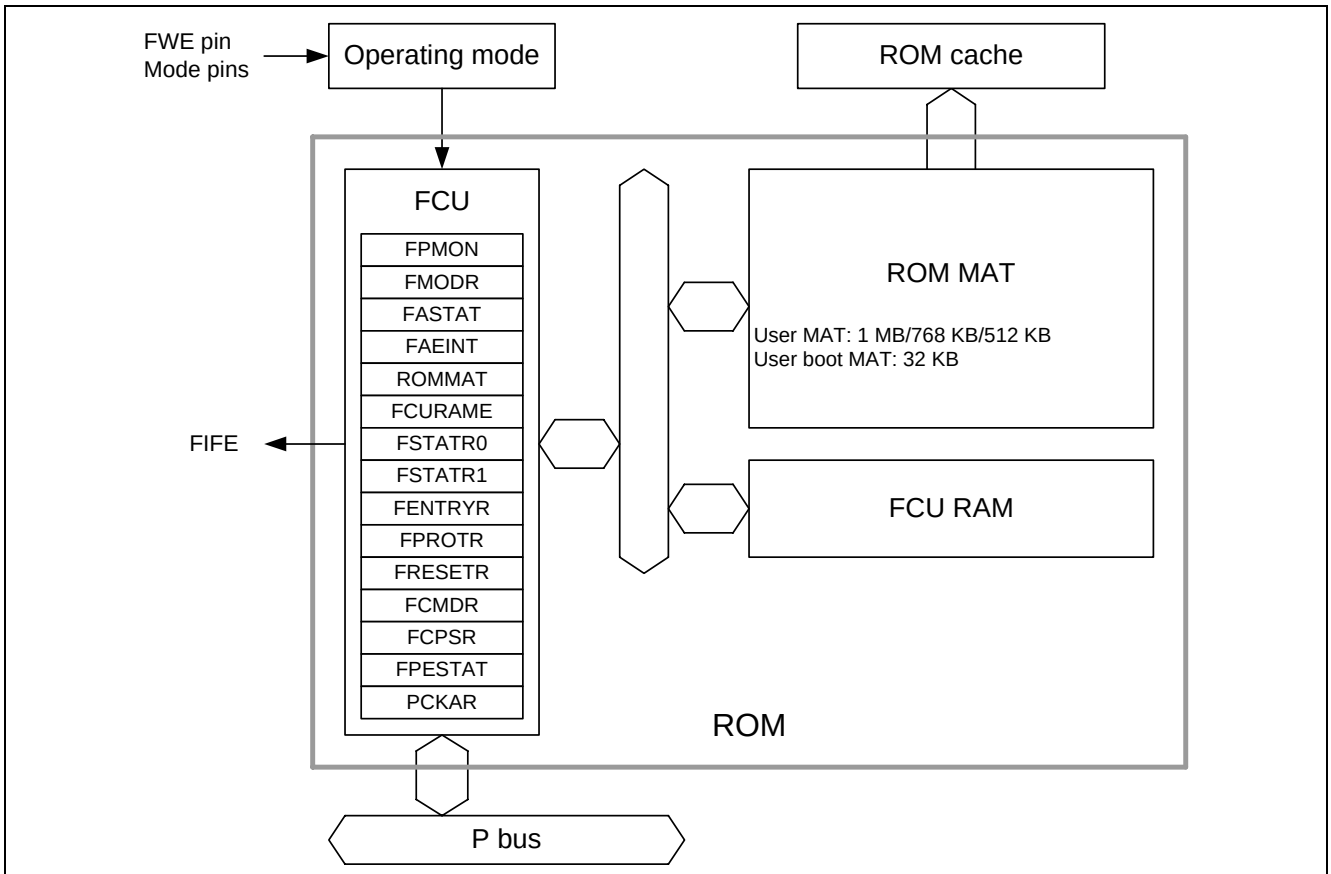
[Legend]
SCRSR: Receive shift register
SCRDR: Receive data register
SCTSR: Transmit shift register
SCTDR: Transmit data register
SCSMR: Serial mode register
SCSCR: Serial control register
SCSSR: Serial status register
SCBRR: Bit rate register
SCSPTR: Serial port register
SCSDCR: Serial direction control register

Figure 2 SCI Block Diagram

2.1.2 On-chip Flash-dedicated Sequencer (FCU)

The SH7216 group MCU programs or erases the flash memory using its FCU.

Figure 3 shows the on-chip flash memory block diagram.



[Legend]
FPMON: Flash pin monitor register
FMODR: Flash mode register
FASTAT: Flash access status register
FAEINT: Flash access error interrupt enable register
ROMMAT: ROM MAT select register
FCURAME: FCU RAM enable register
FSTATR0, 1: Flash status registers 0, 1
FENTRYR: Flash P/E mode entry register
FPROTR: Flash protect register
FRESETR: Flash reset register
FCMDR: FCU command register
FCPSR: FCU processing switch register
FPESTAT: Flash P/E status register
PCKAR: Peripheral clock notification register
FIFE: Flash interface error interrupt

Figure 3 Flash Memory Block Diagram

2.2 Programming/Erasing the On-chip Flash Memory

SH7216 uses its FCU to program and erase the on-chip flash memory. This section describes how to reprogram the on-chip flash memory. For more information, refer to the SH7214 Group, SH7216 Group User's Hardware Manual. This application uses the Standard API for programming and erasing the on-chip flash memory. For more information about the API, refer to the related application note.

2.2.1 Preparing to Program/Erase the On-chip Flash Memory

To use the FCU, the user must store the firmware for the FCU (FCU firmware) in the FCURAM. After transferring the FCU firmware, issue the FCU command to allow the FCU to program or erase the on-chip flash memory.

As the FCU firmware is stored in the FCU firmware area on the MCU, user must transfer the FCU firmware to the FCURAM when the MCU is activated. Make sure to enable accessing the FCURAM by the register setting, because accessing the FCURAM is not allowed when the MCU is activated.

Figure 4 shows the flow chart for preparing to use the FCU command.

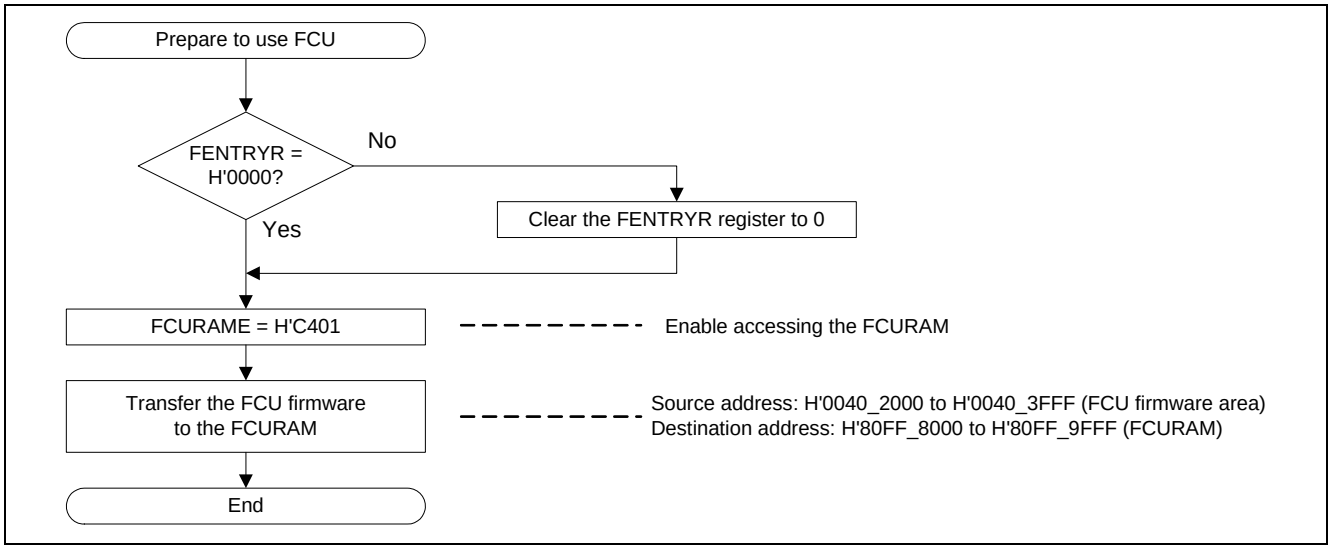


Figure 4 Flow Chart for Preparing to Use the FCU Command

2.2.2 Erasing the On-chip Flash Memory

SH7216 on-chip flash memory is divided into multiple blocks to be erased in blocks. After transferring the FCU firmware, program the erase command (1) and execute command to the erase target block address, and FCU erases blocks.

Figure 5 shows the block division of the SH7216. Table 1 lists each block and address. Figure 6 shows the flow chart (2) for erasing the on-chip flash memory.

Notes: 1. Erase command can be used at any program/erase address in on-chip flash memory.
2. The flow chart in Figure 6 does not follow the standard API.

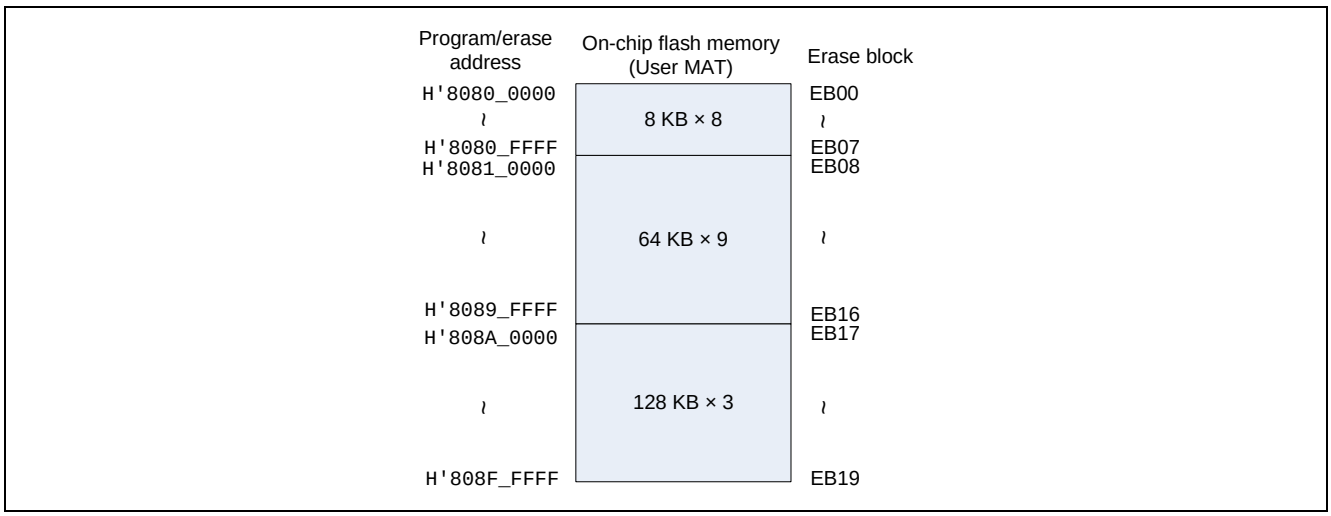


Figure 5 Dividing the On-chip Flash Memory Erase Block

Table 1 Erase Block and Address

Erase Block	Actual Address	Program/Erase Address	Capacity
EB00	H'0000_0000 to H'0000_1FFF	H'8080_0000 to H'8080_1FFF	8 KB
EB01	H'0000_2000 to H'0000_3FFF	H'8080_2000 to H'8080_3FFF	
EB02	H'0000_4000 to H'0000_5FFF	H'8080_4000 to H'8080_5FFF	
EB03	H'0000_6000 to H'0000_7FFF	H'8080_6000 to H'8080_7FFF	
EB04	H'0000_8000 to H'0000_9FFF	H'8080_8000 to H'8080_9FFF	
EB05	H'0000_A000 to H'0000_BFFF	H'8080_A000 to H'8080_BFFF	
EB06	H'0000_C000 to H'0000_DFFF	H'8080_C000 to H'8080_DFFF	
EB07	H'0000_E000 to H'0000_FFFF	H'8080_E000 to H'8080_FFFF	
EB08	H'0001_0000 to H'0001_FFFF	H'8081_0000 to H'8081_FFFF	64 KB
EB09	H'0002_0000 to H'0002_FFFF	H'8082_0000 to H'8082_FFFF	
EB10	H'0003_0000 to H'0003_FFFF	H'8083_0000 to H'8083_FFFF	
EB11	H'0004_0000 to H'0004_FFFF	H'8084_0000 to H'8084_FFFF	
EB12	H'0005_0000 to H'0005_FFFF	H'8085_0000 to H'8085_FFFF	
EB13	H'0006_0000 to H'0006_FFFF	H'8086_0000 to H'8086_FFFF	
EB14	H'0007_0000 to H'0007_FFFF	H'8087_0000 to H'8087_FFFF	
EB15	H'0008_0000 to H'0008_FFFF	H'8088_0000 to H'8088_FFFF	
EB16	H'0009_0000 to H'0009_FFFF	H'8089_0000 to H'8089_FFFF	
EB17	H'000A_0000 to H'000B_FFFF	H'808A_0000 to H'808B_FFFF	128 KB
EB18	H'000C_0000 to H'000D_FFFF	H'808C_0000 to H'808D_FFFF	
EB19	H'000E_0000 to H'000F_FFFF	H'808E_0000 to H'808F_FFFF	

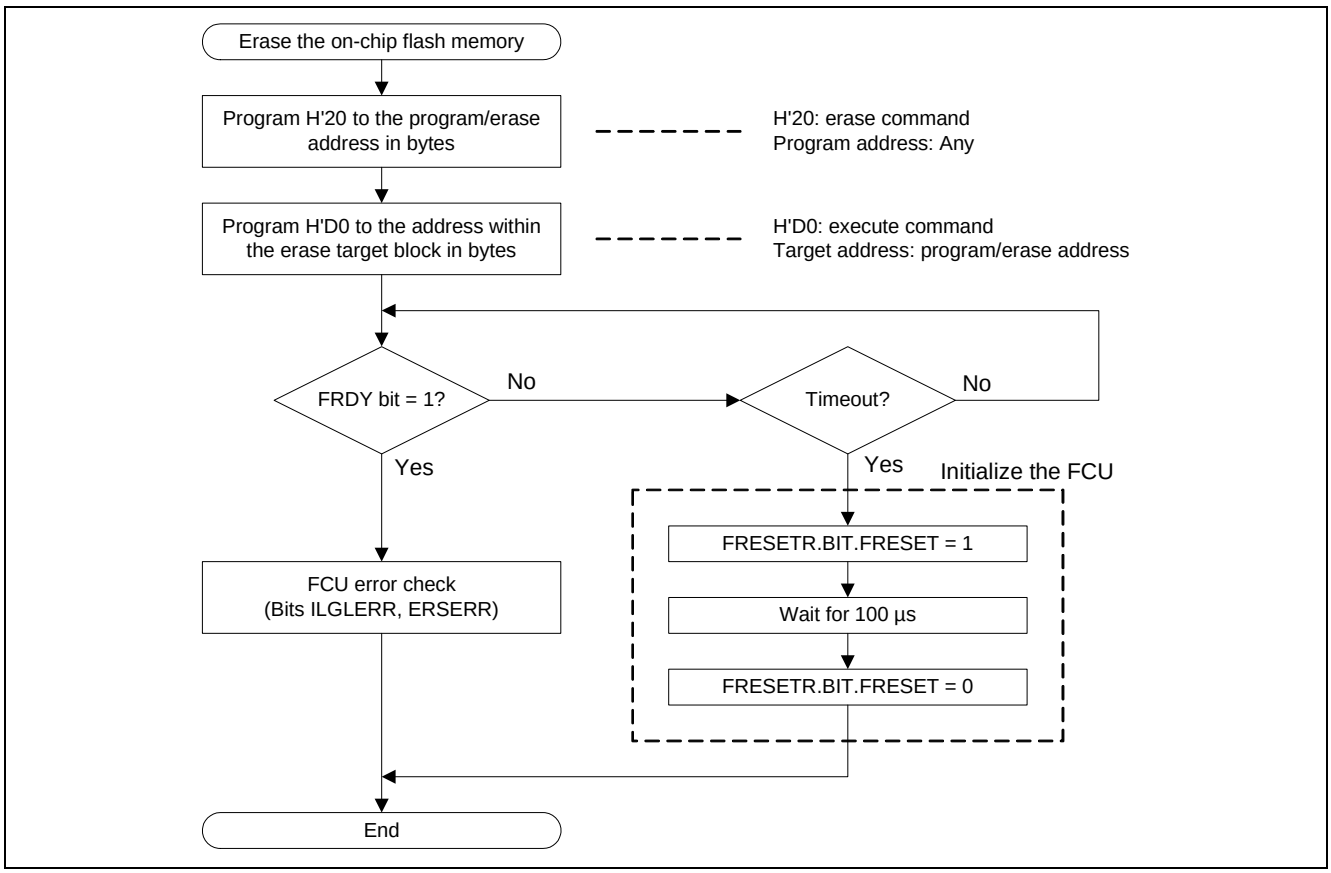


Figure 6 Flow Chart for Erasing the On-chip Flash Memory

2.2.3 Programming the On-chip Flash Memory

Programming data in the on-chip flash memory can be allowed only on the erased area, and program data to the user MAT in units of 256-byte at a time. FCU programs the on-chip flash memory by issuing the command to the FCU, as same as erasing. Issue the program command and program size (1) to the program/erase address, and set (2) the program data (256 bytes) to the program target address (3).

Figure 7 shows the flow chart for programming the flash memory (4).

Notes: 1. Programming data to the user MAT and user boot MAT must be in units of 256-byte (Issue the command H'80).
2. Program the program data to the program/erase address in words.
3. Program target address is the program address plus H'8080_0000 (program/erase address).
4. Flow chart in Figure 7 does not follow the standard API.

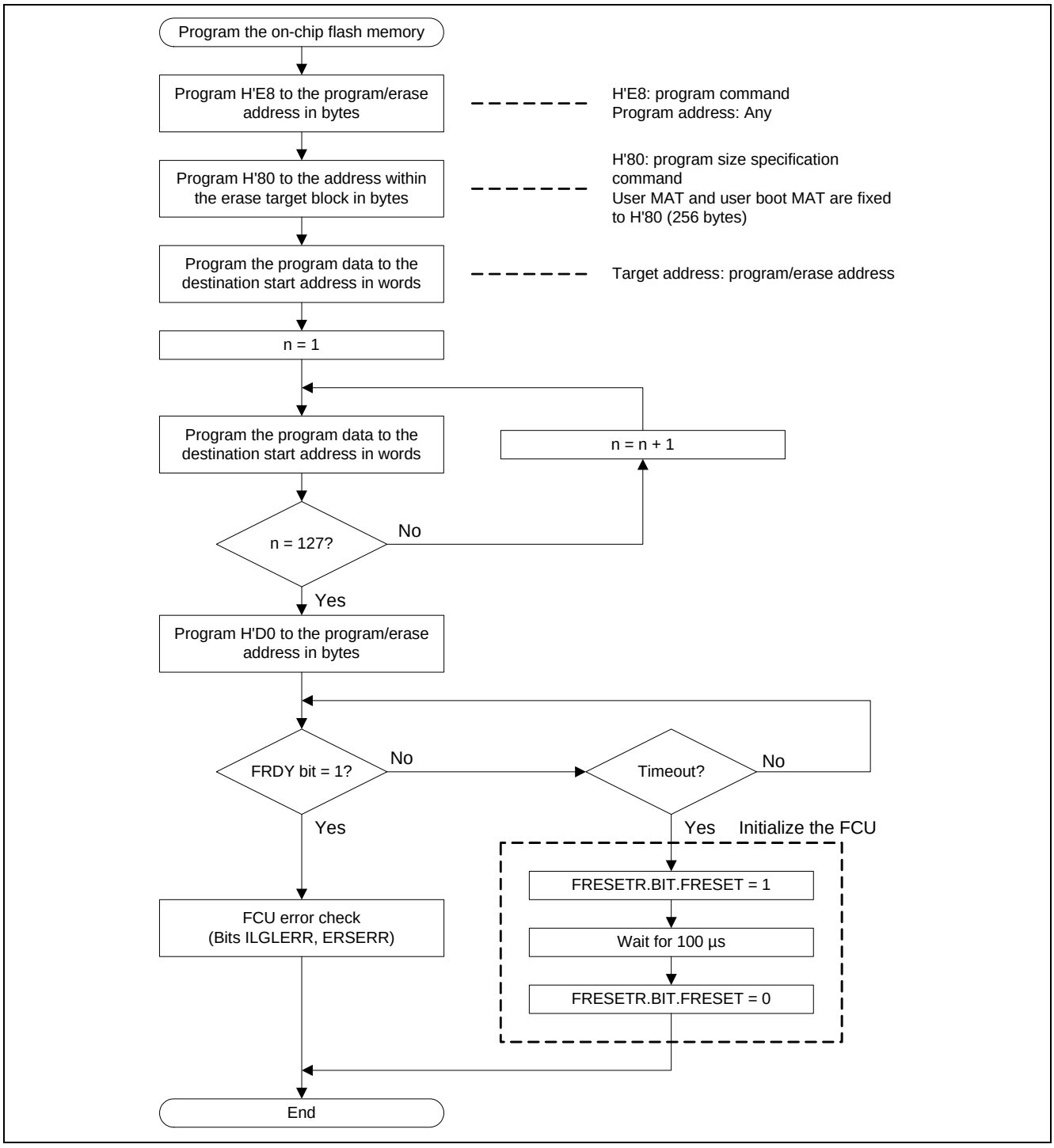


Figure 7 Programming the Flash Memory

2.3 Flash Program Data Buffer

This application has the buffer area to hold the program data in the SH7216 on-chip RAM. The capacity of the buffer area is 256 bytes, which is equivalent to a flash programming.

Figure 8 shows the operation image of the buffer. Table 2 lists the data buffer area address (note).

Note: Data buffer area is divided into sections. Change the section allocation address to set the desired buffer area address. Make sure not to use the same area as the on-chip program in on-chip RAM.

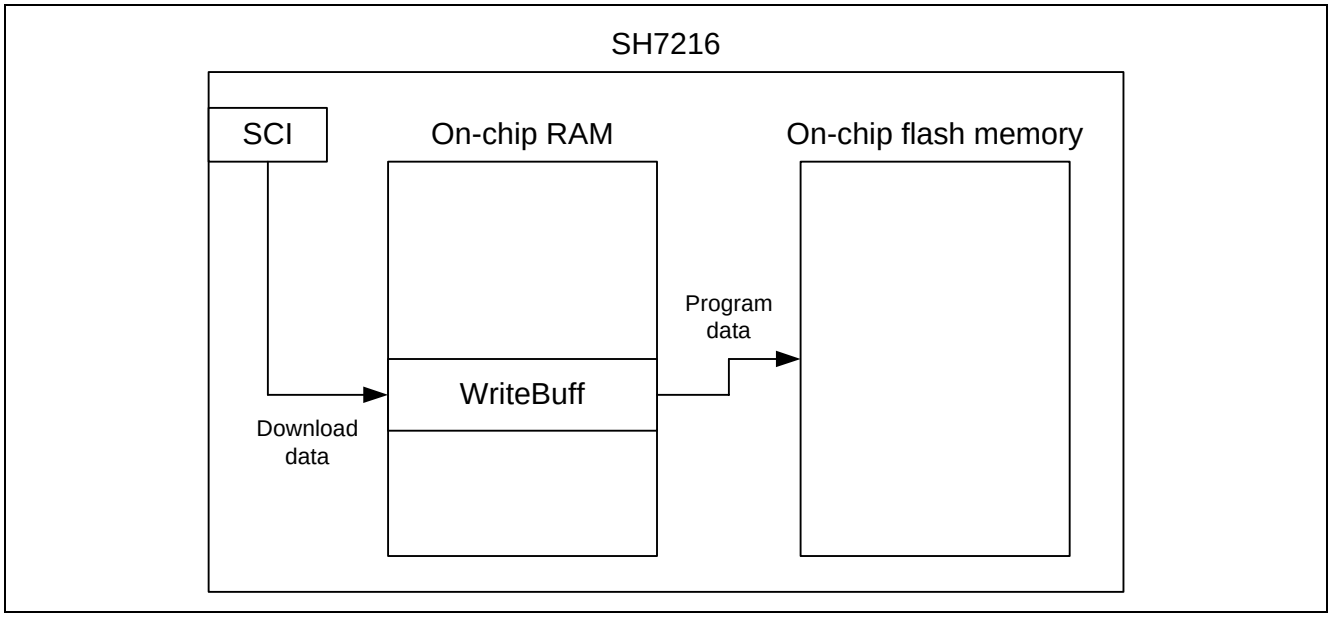


Figure 8 Buffer Operating Image

Table 2 Data Buffer Area Address

Buffer Name	Address	Capacity
WriteBuff	H'FFF8_1000 to H'FFF8_10FF	256 bytes

3. Sample Program External Specifications

This application allocates the flash memory reprogramming sample program including main function (sample program) in EB00 block in the user MAT (address: H'0000 0000 to H'0000 1FFF). Sample program consists of the user application (main function), serial communication program, flash memory reprogramming program, and Standard API.

Figure 9 shows the sample program configuration.

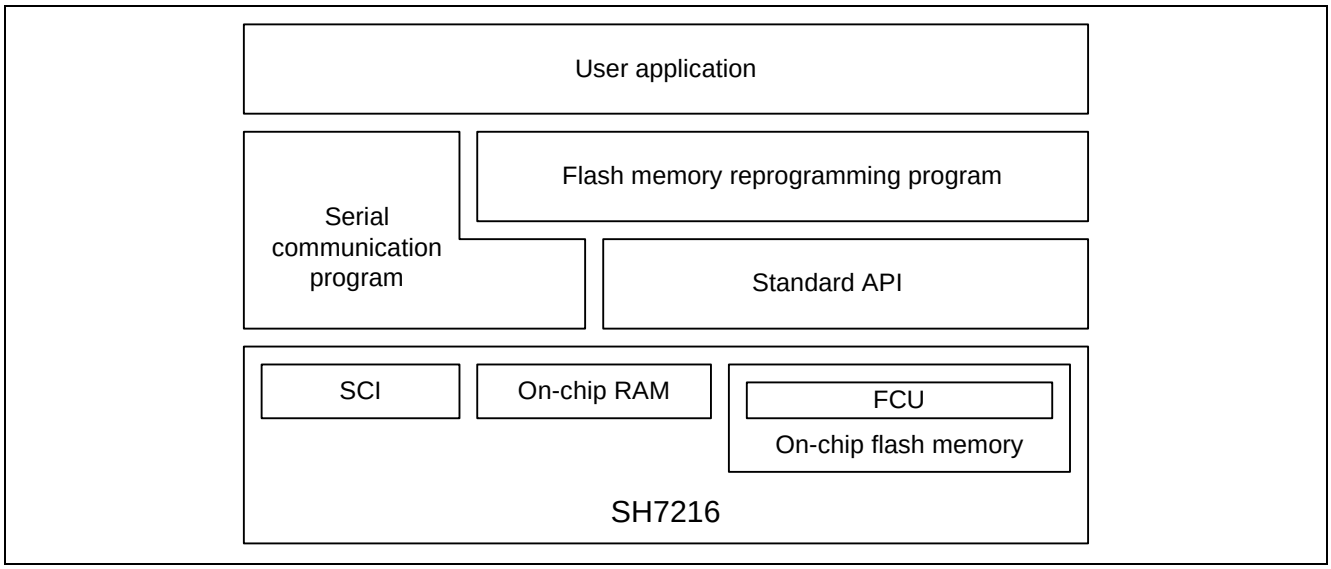


Figure 9 Sample Program Configuration

3.1 On-chip Flash Memory Programming/Erasing Specifications

The target area to program or erase in the on-chip flash memory is the user MAT (EB01 to EB19 block address: H'8080 2000 to H'808F FFFF) other than EB00 block where the sample program is allocated.

When the sample program receives the flash memory programming/erasing command from the host computer, it erases blocks in the program/erase target area in the on-chip flash memory, and programs the specified size of data in the on-chip flash memory from the destination start address which is specified by the host computer.

Figure 10 shows the image of programming and erasing the flash memory by the sample program.

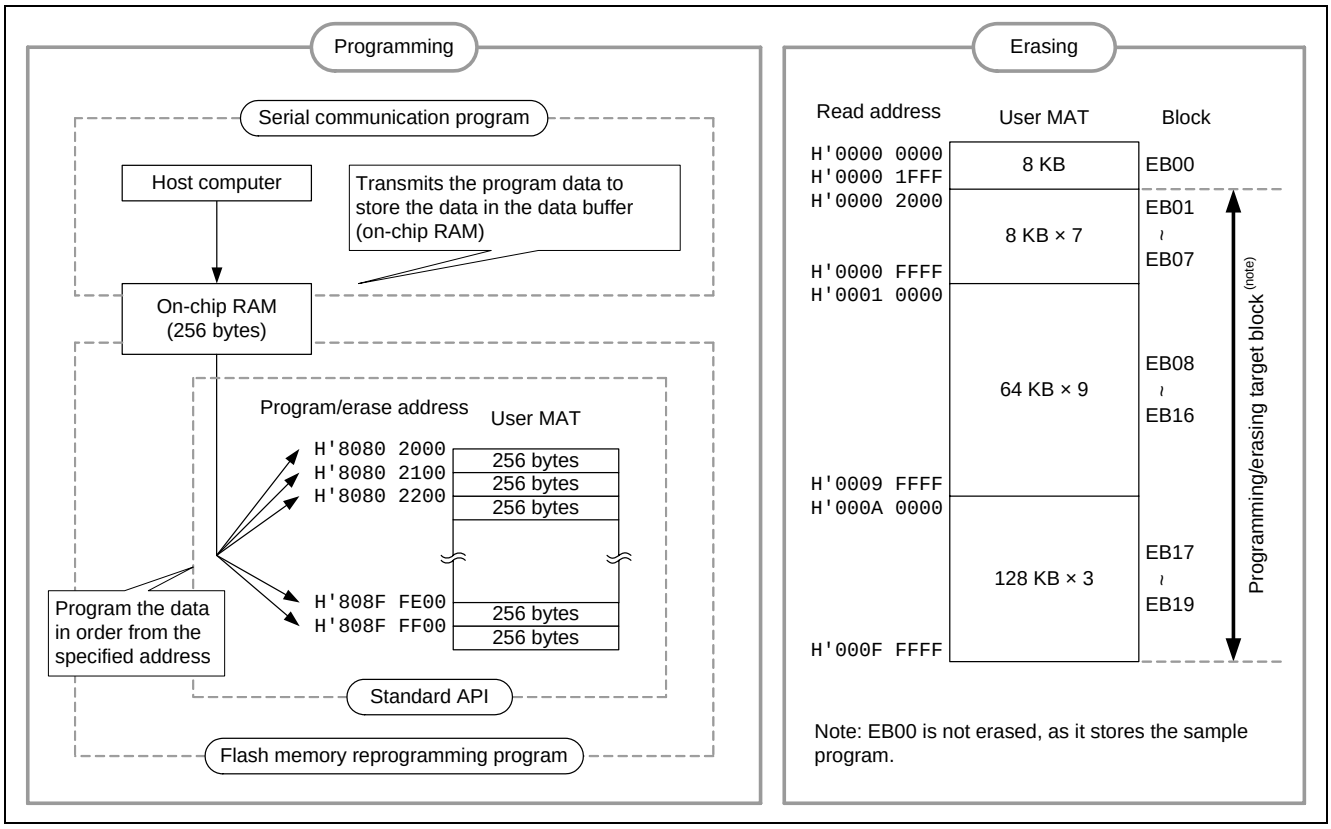


Figure 10 Programming and Erasing the Flash Memory

3.2 Sample Program Operation

This application executes the serial communication with the host computer and transmits/receives the user control commands for communication and data to program, erase and read the flash memory. It uses SCI channel 1 (SCI1) for the serial communication. The sample program these processing to control the flash memory in on-chip RAM.

The sample program checks whether the flash memory is program-/erase-enabled or not. When the flash memory is program-/erase-enabled, the sample program requests the host computer to issue the user control command for communication; otherwise, the sample program polls the FWE bit until the flash memory is program-/erase-enabled.

Figure 11 shows the main processing flow chart.

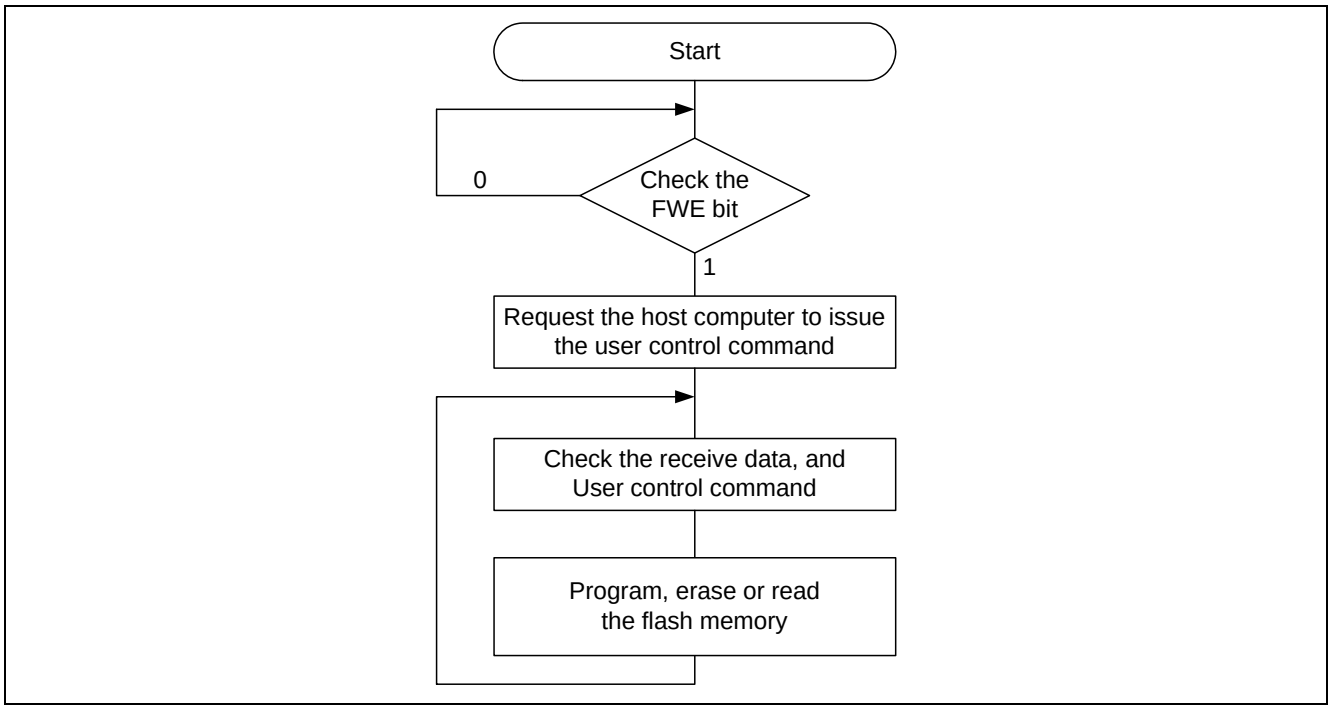


Figure 11 Main Processing Flow Chart

Table 3 lists the user control commands for communication from the host computer. Table 4 lists the notification from the SH7216.

When an error occurs while programming or erasing the on-chip flash memory, the sample program notifies the error end (RET_NG) to the host computer and enters an infinite loop. Add the error processing as appropriate.

Table 3 User Control Commands from the Host Computer to SH7216

Command Name	Value	Description
Program/erase the flash memory (CMD_WRITE)	H'50	Erases blocks (EB01 to EB19), and then programs the specified bytes of data from the specified address
Read the flash memory (CMD_READ)	H'52	Reads the specified bytes of data from the specified address in the on-chip flash memory

Table 4 Notifications from the SH7216 to the Host Computer

Notification Name	Value	Description
Normal end (RET_OK)	H'00	Notifies the host computer that the command handling ends successfully
Error end (RET_NG)	H'01	Notifies the host computer that the command handling ends in error
Transmit request (RET_REQ)	H'11	Notifies the host computer that the sample program is requesting to transmit the user control command or the program data

3.2.1 Programming or Erasing the On-chip Flash Memory

The sample program erases blocks and programs the specified bytes of data from the destination start address in the on-chip flash memory by the flash memory programming/erasing command (CMD_WRITE).

When the sample program receives the flash memory programming/erasing command (CMD_WRITE) from the host computer, it erases blocks to EB01 to EB19. After erasing blocks, the sample program notifies the transmission request (RET_REQ) to the host computer. Then, the sample program receives the destination start address (in units of 4-byte) and program data size (in units of 4-byte) from the host computer (8 bytes in total), and it notifies the transmission request (RET_REQ) of the program data to the host computer, and transitions to programming.

Specify the read address (H'0000 2000 to H'000F FFFF) (note) within blocks EB01 to EB19 as the program destination start address. Otherwise, the sample program notifies the error end (RET_NG) to the host computer to enter an infinite loop. As the sample program does not include the error check when the specified address is not on the user MAT, do not specify the address that is out of bounds.

Note: Specify the read address as the program destination start address to use the Standard API. The Standard API converts the read address into the program/erase address internally.

After the sample program transitions to programming, it notifies the transmission request (RET_REQ) of the program data at every 256-byte data is received (a flash programming). The host computer must transmit 256-byte data for each transmission request (RET_REQ).

The sample program programs the flash memory at every 256-byte data is received. When the last program data size is less than 256-byte, the sample program pads the remaining data to H'FF, to be in units of 256-byte.

When the total number of programming the flash memory reaches the program data size, the sample program notifies the normal end (RET_OK) to the host computer.

Figure 12 shows the communication command sequence when programming or erasing the flash memory by the sample program.

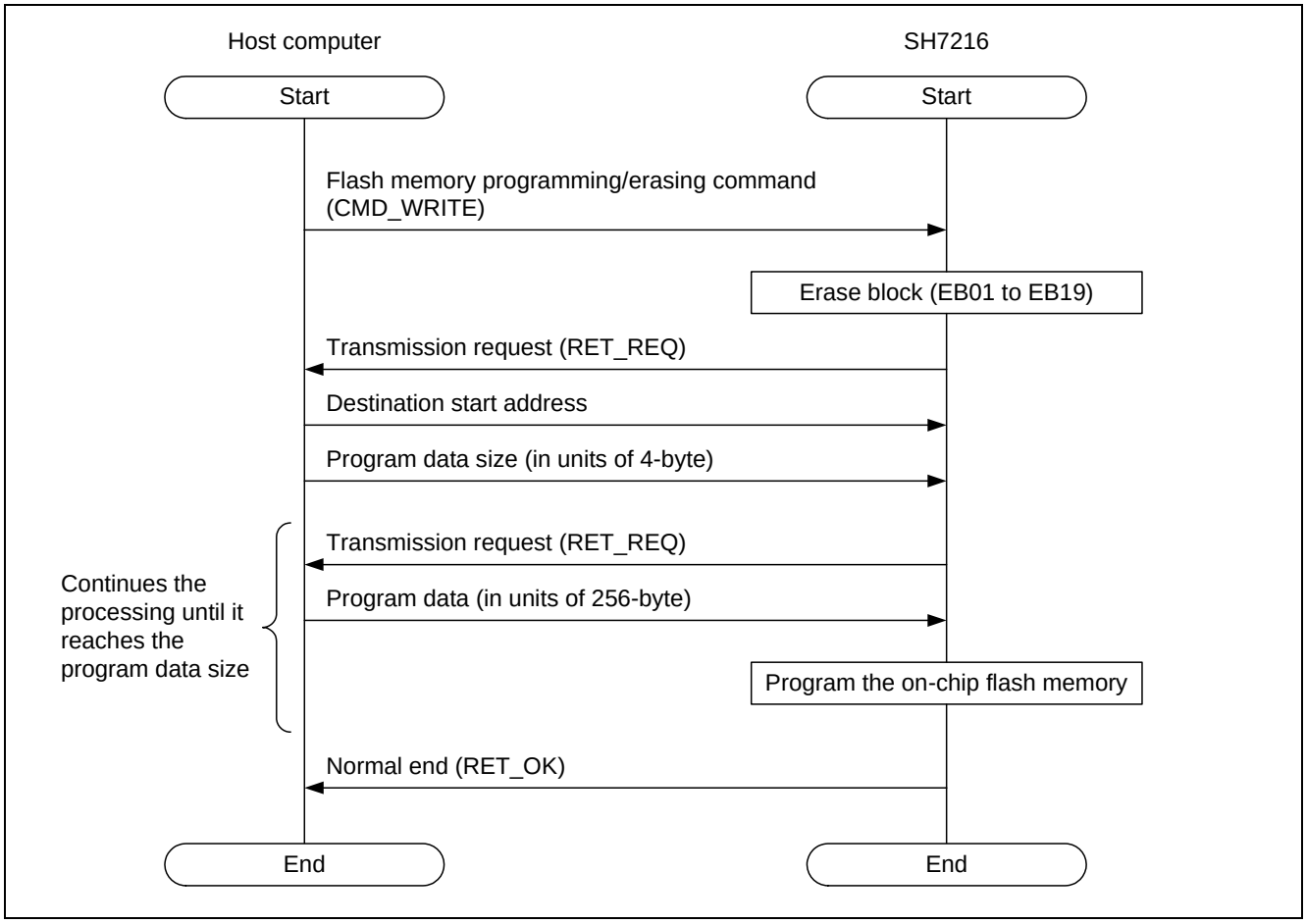


Figure 12 Communication Command Sequence When Programming/Erasing the Flash Memory

3.2.2 Reading the On-chip Flash Memory

The sample program reads the specified bytes of data from the destination start address in the on-chip flash memory and transmits the read data to the host computer by the flash memory reading command (CMD_READ).

When the sample program receives the flash memory reading command (CMD_READ), it notifies the transmission request (RET_REQ) to the host computer. Then, the sample program receives the destination start address (in units of 4-byte) and read data size (in units of 4-byte) from the host computer (8 bytes in total), and it reads the specified size of data from the destination address, and transmits the data to the host computer.

Specify the read address (H'0000 0000 to H'000F FFFF) within blocks EB00 to EB19 (user MAT) as the read destination start address. Otherwise, the sample program does not read the flash memory, notifies the error end (RET_NG) to the host computer to enter an infinite loop. As the sample program does not include the error check when the specified address is not on the user MAT, do not specify the address that is out of bounds.

Figure 13 shows the communication command sequence when reading the flash memory.

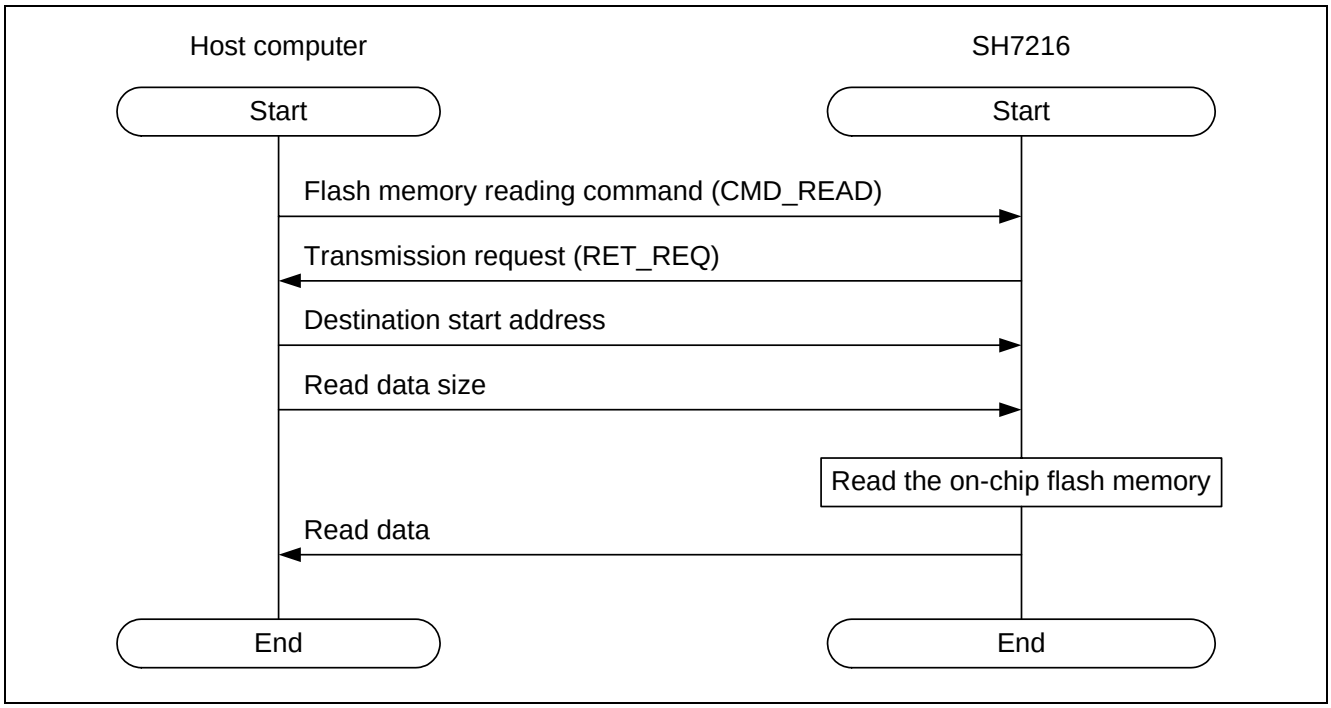


Figure 13 Communication Command Sequence When Reading the Flash Memory

4. Sample Program Internal Specifications

4.1 Modules

Table 5 lists the specifications of sample program modules.

Table 5 Sample Program Modules

Type	Module Name	Function Name	Description	Flow Chart
User application	Main processing	main	Executes the user application	See Figure 14
Flash memory reprogramming	Flash memory programming/ erasing	ocf_write	Programs or erasing the flash memory	See Figure 15 and Figure 16
	Flash memory reading	ocf_read	Reads the flash memory	See Figure 17
	Flash memory program-/erase-enabled check	ocf_pe_chk	Checks that the flash memory is program-/erase-enabled	See Figure 18
Serial communication control	SCI configuration	io_sci_init	Configures the SCI (channel 1)	–
	SCI receive data existence check	io_sci_chk_rcv	Checks if the receive data is stored in the SCRDR register	–
	SCI transmit	io_sci_snd	Transmits one-byte data	–
	SCI receive	io_sci_rcv	Receives the specified bytes of data	–
	SCI module stop	io_sci_stop	Stop supplying the clock to the SCI (channel 1)	–
Standard API	Block erase	R_FlashErase	Erases the data in the specified block	–
	Flash memory programming	R_FlashWrite	Programs the data in the specified address	–

4.2 Variable Used

Table 6 lists a variable used in the sample program.

Table 6 Variable

Variable Label Name	Description	Module to Use
unsigned char WriteBuff[256]	Stores the program data	ocf_write

4.3 Register Settings

Table 7 lists the register settings for the peripherals.

Table 7 SCI (Channel 1) Register Settings in the Sample Program

Register Name	Address	Setting	Description
Serial mode register_1 (SCSMR_1)	H'FFFF 8000	H'00	• C/A# = "0": Asynchronous mode • CHR = "0": 8-bit data • PE = "0": Disables to add and check the parity bit • STOP = "0" 1 stop bit • MP = "0": Disables the multiprocessor mode • CKS [1:0] = "B'00": Peripheral clock
Bit rate register_1 (SCBRR_1)	H'FFFF 8802	D'162	Bit rate = 9600 bps (Peripheral clock = 50 MHz)
Serial control register_1 (SCSCR_1)	H'FFFF 8804	H'30	• TE = "1": Enables the transmitter • RE = "1": Enables the receiver
Port A control register L2 (PACRL2)	H'FFFE 3814	H'0006	• PA4MD [3:0] = "B'110": Outputs TXD1 (SCI)
Port A control register L1 (PACRL1)	H'FFFE 3816	H'6000	• PA3MD [2:0] = "B'110": Inputs RXD1 (SCI)

4.4 Flow Charts

This section describes the flow charts of the sample program.

4.4.1 Main Flow Chart

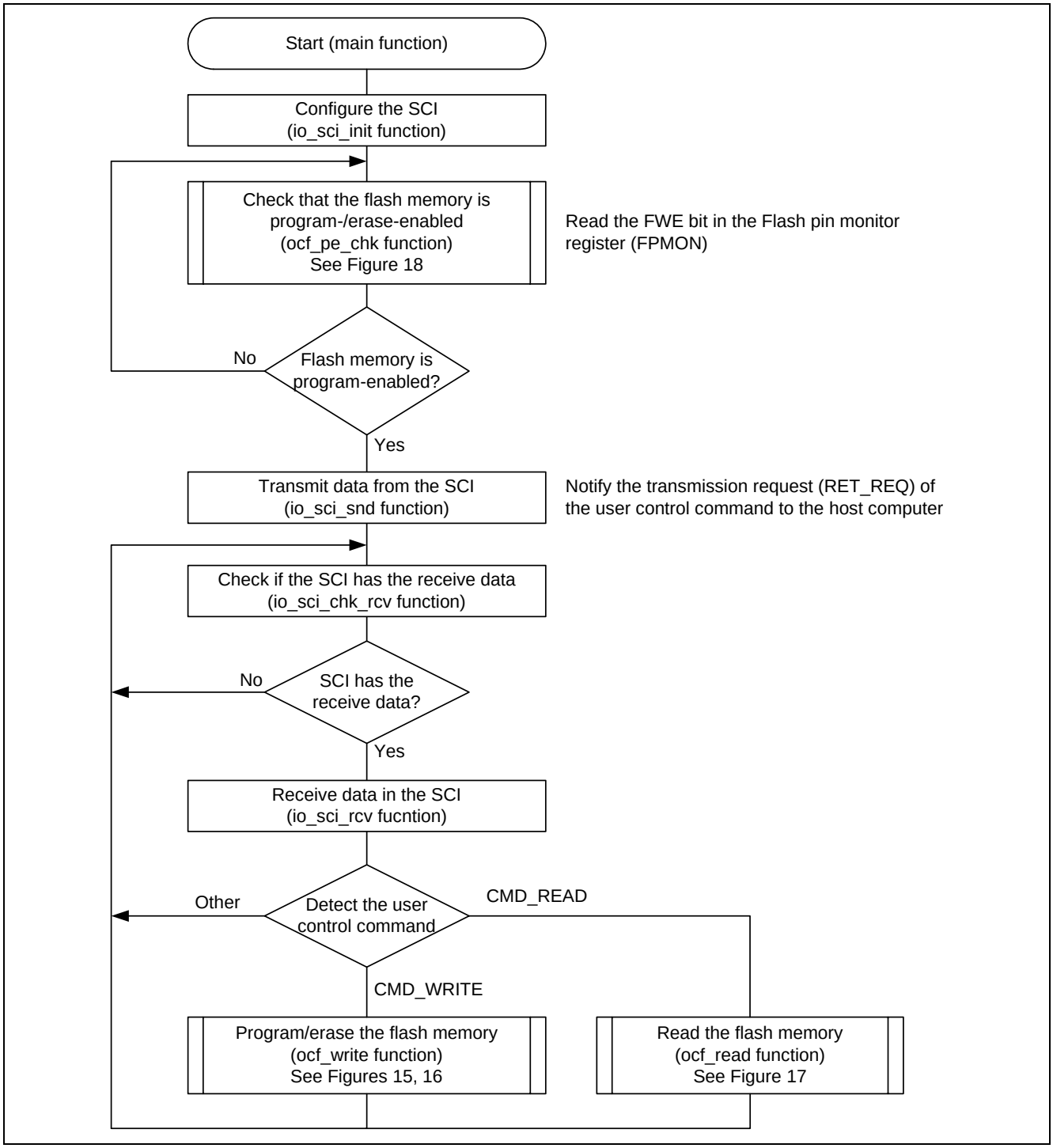


Figure 14 Main Processing Flow Chart

4.4.2 Programming/Erasing the Flash Memory

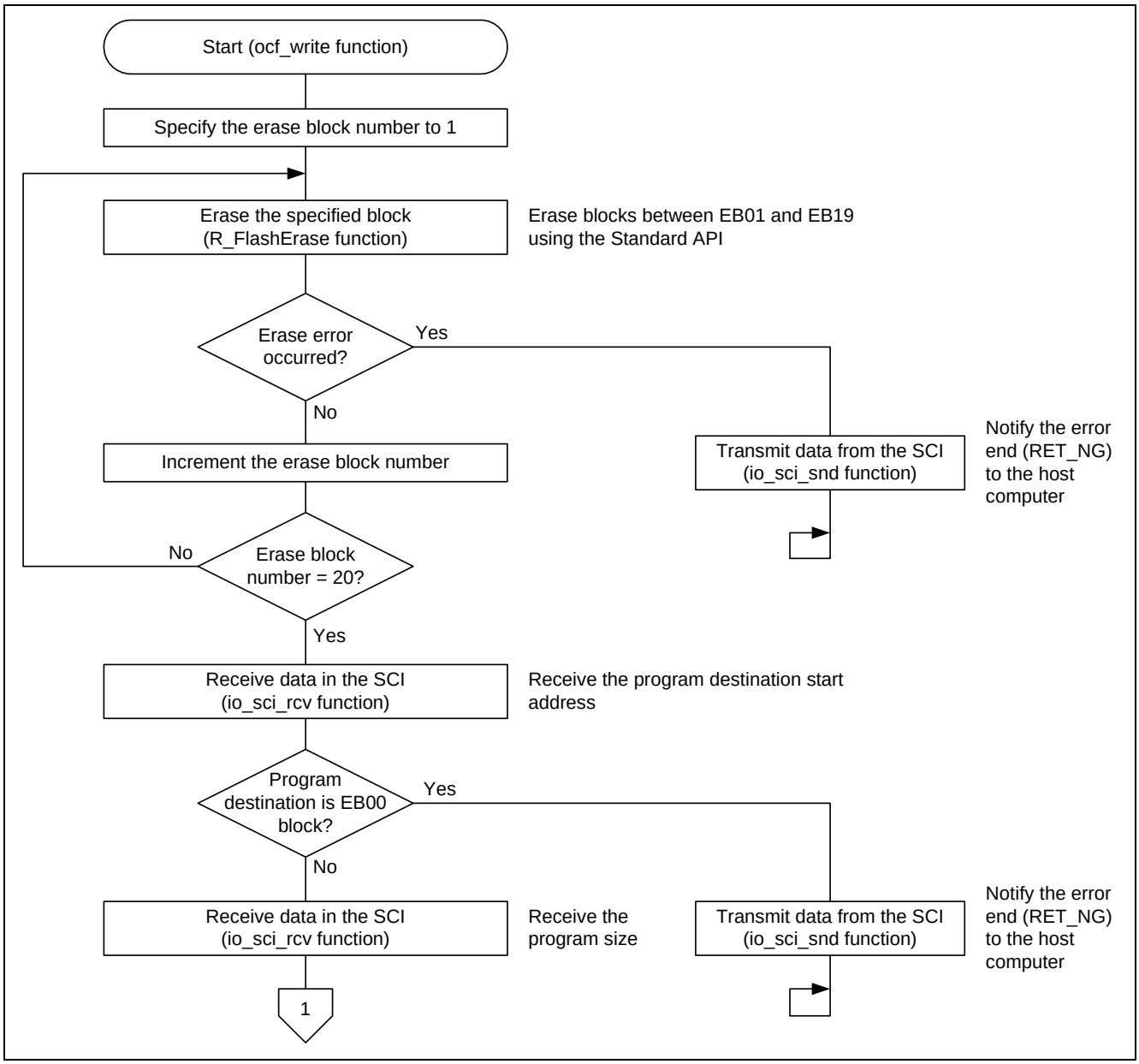


Figure 15 Programming/Erasing the Flash Memory (1/2)

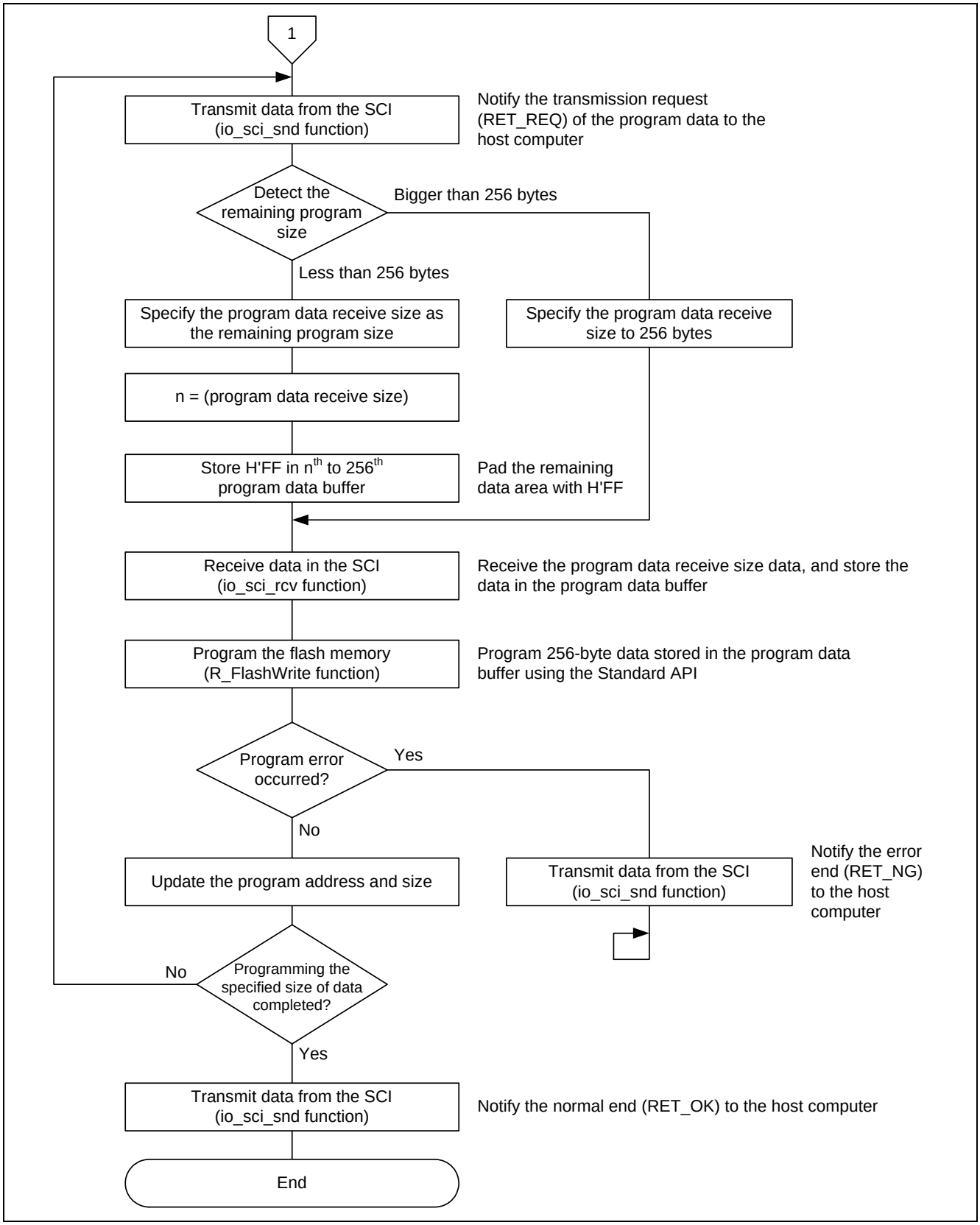


Figure 16 Programming/Erasing the Flash Memory (2/2)

4.4.3 Reading the Flash Memory

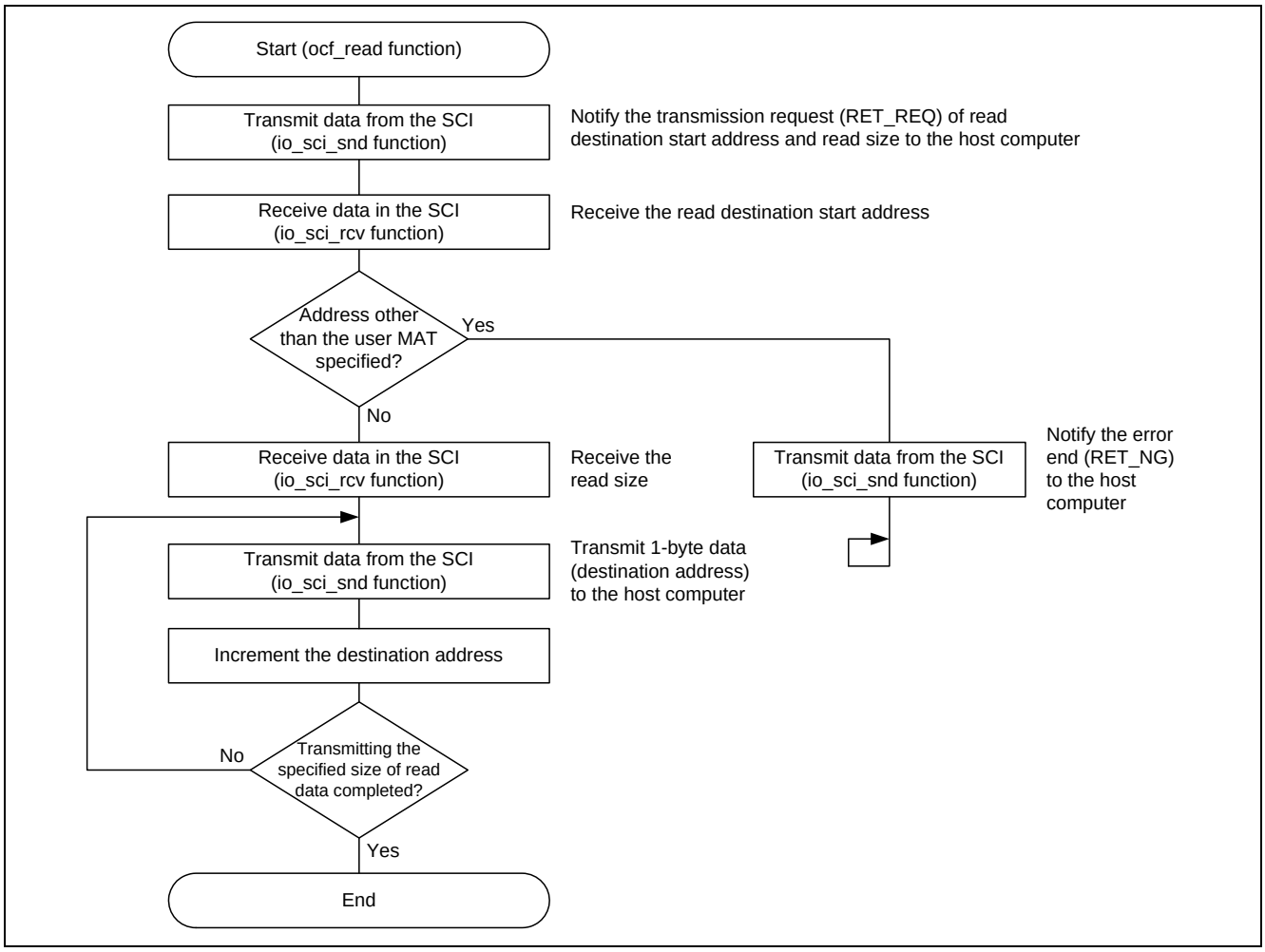


Figure 17 Reading the Flash Memory

4.4.4 Checking the Flash Memory is Program-/Erase-enabled

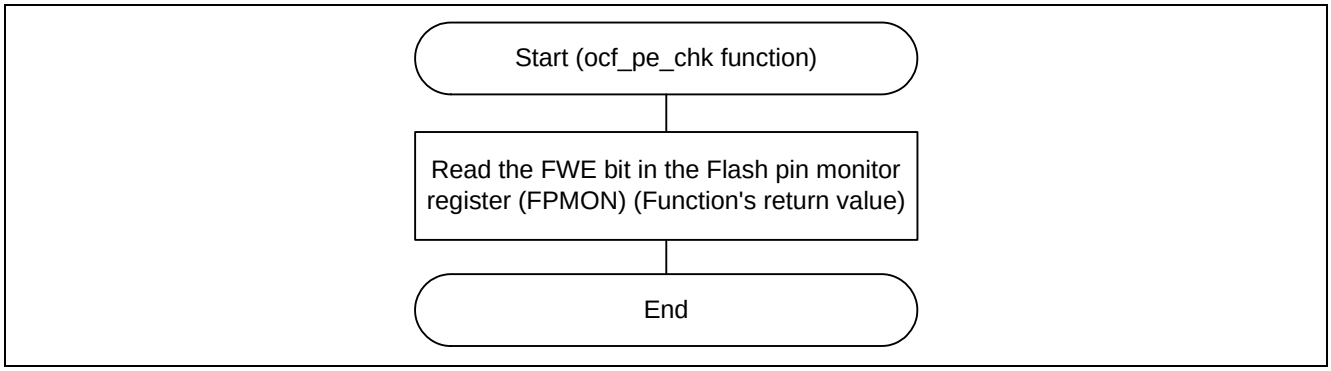


Figure 18 Checking the Flash Memory is Program-/Erase-enabled

5. Sample Program Listing

5.1 Sample Program Listing "main.c" (1/6)

```
/***********************************************************************
*   DISCLAIMER
*
*   This software is supplied by Renesas Electronics Corp. and is only
*   intended for use with Renesas products.  No other uses are authorized.
*
*   This software is owned by Renesas Electronics Corp. and is protected under
*   all applicable laws, including copyright laws.
*
*   THIS SOFTWARE IS PROVIDED "AS IS" AND RENESAS MAKES NO WARRANTIES
*   REGARDING THIS SOFTWARE, WHETHER EXPRESS, IMPLIED OR STATUTORY,
*   INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
*   PARTICULAR PURPOSE AND NON-INFRINGEMENT.  ALL SUCH WARRANTIES ARE EXPRESSLY
*   DISCLAIMED.
*
*   TO THE MAXIMUM EXTENT PERMITTED NOT PROHIBITED BY LAW, NEITHER RENESAS
*   ELECTRONICS CORP. NOR ANY OF ITS AFFILIATED COMPANIES SHALL BE LIABLE
*   FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES
*   FOR ANY REASON RELATED TO THIS SOFTWARE, EVEN IF RENESAS OR ITS
*   AFFILIATES HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.
*
*   Renesas reserves the right, without notice, to make changes to this
*   software and to discontinue the availability of this software.
*   By using this software, you agree to the additional terms and
*   conditions found by accessing the following link:
*   http://www.renesas.com/disclaimer
***********************************************************************
* Copyright (C) 2010 Renesas Electronics Corporation. All rights reserved.
***********************************************************************/
/*""FILE COMMENT""*********** Technical reference data ************************
*   System Name : SH7216 Sample Program
*   File Name   : main.c
*   Abstract    : Using user program mode
*   Version     : 1.00.00
*   Device      : SH7216
*   Tool-Chain  : High-performance Embedded Workshop (Ver.4.07.00).
*               : C/C++ compiler package for the SuperH RISC engine family
*               :                             (Ver.9.03 Release00).
*   OS          : None
*   H/W Platform: R0K572167 (CPU board)
*   Description :
***********************************************************************
*   History     : Aug.20,2010 Ver.1.00.00
*""FILE COMMENT END""*****************************************************/
#include "iodefine.h"
#include "Flash_API_SH7216.h"

```

5.2 Sample Program Listing "main.c" (2/6)

```
48  /* ==== Macro definition ==== */
49  #define FLASH_PE_ENABLE       1       /* Flash program/erase enabled       */
50  #define FLASH_PE_DISABLE  0       /* Flash program/erase disabled      */
51  #define CMD_WRITE         0x50    /* Flash memory erasing/programming command */
52  #define CMD_READ          0x52    /* Flash memory reading command */
53  #define RET_OK            0x00    /* Normal end */
54  #define RET_NG            0x01    /* Error end */
55  #define RET_REQ           0x11    /* Transmission request */
56
57  /* ==== Prototype declaration ==== */
58  void main(void);
59  int ocf_pe_chk(void);
60  void ocf_write(void);
61  void ocf_read(void);
62  /* ---- External reference ---- */
63  extern void io_sci_init(void);
64  extern int io_sci_chk_rcv(void);
65  extern void io_sci_snd(unsigned char data);
66  extern void io_sci_rcv(unsigned char *data, unsigned long num);
67
68  /* ==== Global variable ==== */
69  #pragma section WriteDATA  /* Program data buffer area */
70  unsigned char WriteBuff[PROGRAM_SIZE_ROM];
71
72
73  #pragma section FRAM
74  /*""FUNC COMMENT""*****************************************************************
75   * ID           :
76   * Outline      : Sample program main
77   *-------------------------------------------------------------------------------
78   * Include      : "iodefine.h" and "flash.h"
79   *-------------------------------------------------------------------------------
80   * Declaration  : void main(void);
81   *-------------------------------------------------------------------------------
82   * Function     :
83   *-------------------------------------------------------------------------------
84   * Argument     : void
85   *-------------------------------------------------------------------------------
86   * Return Value : void
87   *-------------------------------------------------------------------------------
88   * Note         : None
89   *""FUNC COMMENT END""*************************************************************/
90  void main(void)
91  {
92    unsigned char RcvData;
93    int pe_ok;
94
95    /* ==== Configures the SCI ==== */
96    io_sci_init();
97
```

5.3 Sample Program Listing "main.c" (3/6)

```
98      /* ==== Checks the flash memory is program-/erase-enabled ==== */
99      do{
100       pe_ok = ocf_pe_chk();      /* FWE pin = High ? */
101     }while(pe_ok != FLASH_PE_ENABLE);
102
103     /* ==== Notifies the transmission request to the host computer ==== */
104     io_sci_snd(RET_REQ);
105
106     /* ==== Programs/erases the flash memory or reads the flash memory ==== */
107     while(1){
108       /* ---- Checks the user control command ---- */
109       if(io_sci_chk_rcv() != 0){
110           io_sci_rcv(&RcvData, 1);
111           if(RcvData == CMD_WRITE){
112               ocf_write();          /* Erases or programs the flash memory */
113           }
114           else if(RcvData == CMD_READ){
115               ocf_read();           /* Reads the flash memory */
116           }
117       }
118     }
119   }
120
121   /*""FUNC COMMENT""*************************************************************
122    * ID           :
123    * Outline      : Flash memory program-/erase-enabled state check
124    *-----------------------------------------------------------------------------
125    * Include      : "iodefine.h"
126    *-----------------------------------------------------------------------------
127    * Declaration  : int ocf_pe_chk(void);
128    *-----------------------------------------------------------------------------
129    * Description  : Reads the FWE bit in the Flash pin monitor register (FPMON) and
130    *              : returns the value.
131    *-----------------------------------------------------------------------------
132    * Argument     : void
133    *-----------------------------------------------------------------------------
134    * Return Value : 0 ; Flash memory is program-/erase-disabled
135    *              : 1 ; Flash memory is program-/erase-enabled
136    *-----------------------------------------------------------------------------
137    * Note         : None
138    *""FUNC COMMENT END""*********************************************************/
139   int ocf_pe_chk(void)
140   {
141     return FLD.FPMON.BIT.FWE;
142   }
143
```

5.4 Sample Program Listing "main.c" (4/6)

```
/*""FUNC COMMENT""*****************************************************************
 * ID           :
 * Outline      : Programming/erasing the flash memory
 *-----------------------------------------------------------------------------
 * Include      : "iodefine.h" and "flash.h"
 *-----------------------------------------------------------------------------
 * Declaration  : void ocf_write(void);
 *-----------------------------------------------------------------------------
 * Description  : Erases blocks between EB01 to EB19, and programs the specified
 *              : bytes of data from the destination start address which is
 *              : specified by the host computer.
 *-----------------------------------------------------------------------------
 * Argument     : void
 *-----------------------------------------------------------------------------
 * Return Value : void
 *-----------------------------------------------------------------------------
 * Note         : None
 *""FUNC COMMENT END""*************************************************************/
void ocf_write(void)
{
  unsigned char error;        /* Function return value */
  unsigned char RcvData;          /* Receive data */
  unsigned char EraseBlkNum;      /* Erase block number */
  unsigned long i;            /* Loop counter */
  unsigned long WriteAddr;    /* Start address to be programmed */
  unsigned long WriteSize;    /* Data size to be programmed */
  unsigned long RcvSize;          /* Receiving size for data to be programmed */

  /* ==== Erases blocks ==== */
  for(EraseBlkNum = BLOCK_1; EraseBlkNum <= BLOCK_19; EraseBlkNum++){
    /* Erases EB01 to EB19 */
    error = R_FlashErase((uint8_t)EraseBlkNum);
    if(error != RET_OK){              /* Erase error? */
        io_sci_snd(RET_NG);               /* Error end */
        while(1){
        }
    }
  }

  /* ==== Transmission request ==== */
  io_sci_snd(RET_REQ);

  /* ==== Receives the program destination start address ==== */
  io_sci_rcv((unsigned char *)&WriteAddr, 4);
  if( (WriteAddr >= 0x00000000) && (WriteAddr < 0x00002000) ){
    /* EB00 is specified as the destination? */
    io_sci_snd(RET_NG);                   /* Error end */
    while(1){
    }
  }

```

5.5 Sample Program Listing "main.c" (5/6)

```
195     /* ==== Receives the program data size ==== */
196     io_sci_rcv((unsigned char *)&WriteSize, 4);
197
198     /* ==== Programs the flash memory ==== */
199     while(WriteSize > 0){
200       io_sci_snd(RET_REQ);              /* Transmission request */
201
202       if(WriteSize > PROGRAM_SIZE_ROM){
203           RcvSize = PROGRAM_SIZE_ROM;
204       }
205       else{
206           RcvSize = WriteSize;
207           for(i = RcvSize; i < PROGRAM_SIZE_ROM; i++){
208               WriteBuff[i] = 0xff;
209           }
210       }
211
212       /* ---- Receives the program data ---- */
213       io_sci_rcv(WriteBuff, RcvSize);
214
215       /* ---- Programs the flash memory ---- */
216       error = R_FlashWrite((uint32_t)WriteAddr, (uint32_t)WriteBuff, PROGRAM_SIZE_ROM);
217       if(error != 0){                   /* Program error? */
218           io_sci_snd(RET_NG);               /* Error end */
219           while(1){
220           }
221       }
222
223       WriteAddr += PROGRAM_SIZE_ROM;
224       WriteSize -= RcvSize;
225     }
226
227     io_sci_snd(RET_OK);             /* Normal end */
228   }
229
```

5.6 Sample Program Listing "main.c" (6/6)

```
/*""FUNC COMMENT""*****************************************************************
 * ID           :
 * Outline      : Reading the flash memory
 *-----------------------------------------------------------------------------
 * Include      : "flash.h"
 *-----------------------------------------------------------------------------
 * Declaration  : void ocf_read(void);
 *-----------------------------------------------------------------------------
 * Description  : Reads the specified size of data from the read destination
 *              : start address and transmits the data to the host computer.
 *-----------------------------------------------------------------------------
 * Argument     : void
 *-----------------------------------------------------------------------------
 * Return Value : void
 *-----------------------------------------------------------------------------
 * Note         : None
 *""FUNC COMMENT END""*************************************************************/
void ocf_read(void)
{
  unsigned char *ReadData;/* Pointer for readout data */
  unsigned long ReadAddr;     /* Start address to be read */
  unsigned long ReadSize;     /* Reading size */
  unsigned long i;        /* Loop counter */

  /* ==== Transmission request ==== */
  io_sci_snd(RET_REQ);

  /* ==== Receives the read destination start address ==== */
  io_sci_rcv((unsigned char *)&ReadAddr, 4);
  if(ReadAddr >= 0x00100000){
    /* Specified the address other than the read address? */
    io_sci_snd(RET_NG);               /* Error end */
    while(1){
    }
  }

  /* ==== Receives the read data size ==== */
  io_sci_rcv((unsigned char *)&ReadSize, 4);

  /* ==== Transmits the data which is read from ROM ==== */
  ReadData = (unsigned char *)ReadAddr;
  for(i = 0; i < ReadSize; i++){
    io_sci_snd(*ReadData++);
  }
}

/* End of File */
```

6. References

- Hardware Manual
  SH7214 Group, SH7216 Group User's Manual: Hardware Rev. 3.00
  The latest version of the hardware manual can be downloaded from the Renesas Electronics website.

- Software Manual
  SH-2A SH2A-FPU Software Manual Rev. 3.00
  The latest version of the software manual can be downloaded from the Renesas Electronics website.

Website and Support

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

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

Revision Record

Rev.	Date	Description	
		Page	Summary
1.00	Dec.10.10	–	First edition issued
1.01	Jun.15.12	–	Sample code (simple flash API) 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 type number, confirm that the change will not lead to problems.
   — The characteristics of MPU/MCU in the same group but having different type numbers may differ because of the differences in internal memory capacity and layout pattern. When changing to products of different type numbers, implement a system-evaluation test for each of the products.

Notice

1. 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.
2. 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.
3. 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.
4. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from such alteration, modification, copy or otherwise misappropriation of Renesas Electronics product.
5. Renesas Electronics products are classified according to the following two quality grades: "Standard" and "High Quality". The recommended applications for each Renesas Electronics product depends on the product's quality grade, as indicated below.
   "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 etc.
   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; and safety equipment etc.
   Renesas Electronics products are neither intended nor authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems, surgical implantations etc.), or may cause serious property damages (nuclear reactor control systems, military equipment etc.). 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 for which it is not intended. 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 which the product is not intended by Renesas Electronics.
6. 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.
7. 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 systems manufactured by you.
8. 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.
9. 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. You should not use Renesas Electronics products or 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. When exporting the Renesas Electronics 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.
10. It is the responsibility of the buyer or distributor of Renesas Electronics products, who distributes, disposes of, or otherwise places the product with a third party, to notify such third party in advance of the contents and conditions set forth in this document, Renesas Electronics assumes no responsibility for any losses incurred by you or third parties as a result of unauthorized use of Renesas Electronics products.
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 2.0