To our customers,

Old Company Name in Catalogs and Other Documents

On April 1st, 2010, NEC Electronics Corporation merged with Renesas Technology Corporation, and Renesas Electronics Corporation took over all the business of both companies. Therefore, although the old company name remains in this document, it is a valid Renesas Electronics document. We appreciate your understanding.

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

April 1st, 2010
Renesas Electronics Corporation

Issued by: Renesas Electronics Corporation (http://www.renesas.com)

Send any inquiries to http://www.renesas.com/inquiry.

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.

SH7211 Group

Reception by the I²C Bus Interface 3 Module in Single-Master Operation (EEPROM Reading)

Introduction

This application note describes reception by the I²C bus interface 3 module (IIC3) of the SH7211 in the case of a single master on the I²C bus.

Target Device

SH7211

Contents

1. Preface

1.1 Specifications

- Data are read from an EEPROM with the SH7211 as the master device and the EEPROM as a slave device.
- The transfer rate is set at 400 kHz.

Note: Please adjust settings as required to match the specifications of the EEPROM you are using.

1.2 Module Used

- I²C bus interface 3 (IIC3)

1.3 Applicable Conditions

- MCU: SH7211
- Operating frequency: Internal clock 160 MHz
  Bus clock 40 MHz
  Peripheral clock 40 MHz
- C compiler: SuperH RISC Engine Family C/C++ Compiler Package Ver.9.01 Release01
  from Renesas Technology
- Compiler options: -cpu = sh2a -include = "$(WORKSPDIR)\inc"
  -object = "$(CONFIGDIR)\$(FILELEAF).obj" -debug -gbr = auto -chgincpath
  -errorpath -global_volatile = 0 -opt_range = all -infinite_loop = 0 -del_vacant_loop = 0
  -struct_alloc = 1 -nologo
- EEPROM: HN58X24128FPIE (128 Kbits)
  from Renesas Technology

1.4 Related Application Note

The operation of the sample program in this application note was confirmed with the configuration specified in the application note “Example of Initial Configuration” of SH7211 (REJ06B0786). Please refer to that document when setting up this sample task.

2. Description of the Sample Application

In this sample program, the SH7211 (master device) receives data from the EEPROM (slave device) by using the I²C bus interface 3 (IIC3) module.

2.1 Operational Overview of Module Used

The I²C bus interface 3 (IIC3) module conforms to and provides a subset of the Philips I²C (Inter-IC) bus interface functions. However, the configuration of the registers that control the I²C bus differs in some respects from the register configuration implemented by Philips.

The features of the SH7211's I²C bus interface 3 (IIC3) are described below.

- I²C bus format and clock-synchronous serial format are selectable.
- Continuous transmission/reception
  Since the shift register, transmit data register, and receive data register are independent of each other, continuous data transfer is possible.

Table 1is a list of the features of the available formats, and figure 1 shows a block diagram of the IIC3 module.

Table 1 Features of the Formats

Format	Features
I²C bus format	• Start and stop conditions are generated automatically in master mode. • Acknowledge output levels are selectable in data reception. • Acknowledge bit is automatically loaded in data transmission • On-chip bit synchronization/wait function In master mode, the state of SCL is monitored per bit, and the timing is synchronized automatically. If transmission/reception is not yet possible, set the SCL to low until preparations are completed. • Six interrupt sources 1. Transmit data empty (including slave-address match) 2. Transmit end 3. Receive data full (including slave-address match) 4. Arbitration lost 5. NACK detection 6. Stop condition detection • Data transfer by the direct memory access controller (DMAC) can be activated by a transmit-data-empty or receive-data-full interrupt request. • Direct bus drive Two pins, SCL and SDA pins, function as NMOS open-drain outputs when the bus drive function is selected.
Clock-synchronous serial format	• Four interrupt sources 1. Transmit-data-empty 2. Transmit-end 3. Receive-data-full 4. Overrun error • Data transfer by the direct memory access controller (DMAC) can be activated by a transmit-data-empty or receive-data-full interrupt request.

Note: For details on IIC3, see the section on I2C Bus Interface 3 (IIC3) of the *SH7211 Group Hardware Manual*.

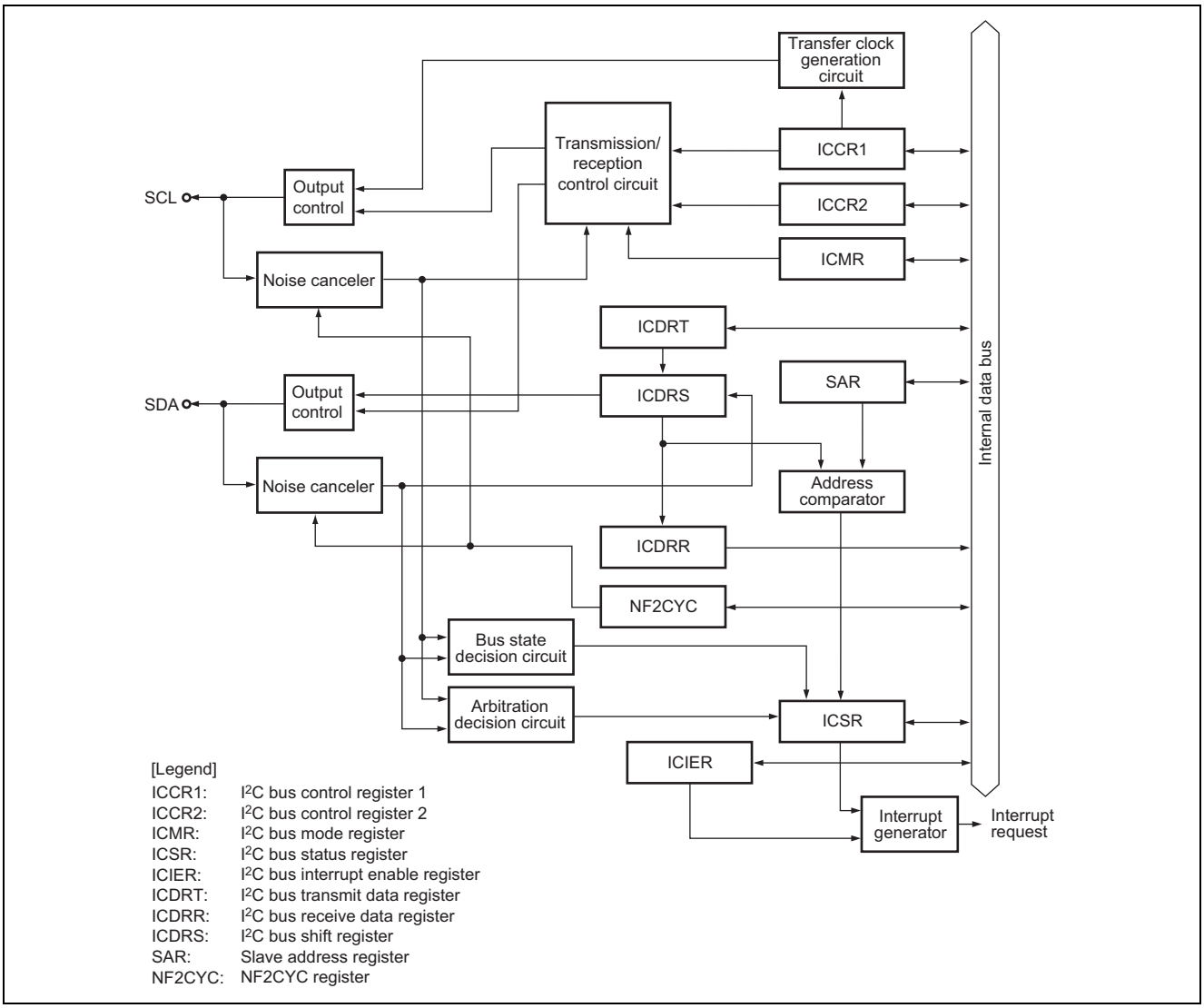


Figure 1 Overview of I²C Bus Interface 3

2.2 Procedure for Setting the Module Used

This section describes the procedure for making initial settings for IIC3. The transfer rate must be set to meet the external specification. In this sample program, Pϕ /100 is specified as the transfer rate. Figure 2 shows an example of the initialization sequence for IIC3. For details on the settings of individual registers, see the *SH7211 Group Hardware Manual*.

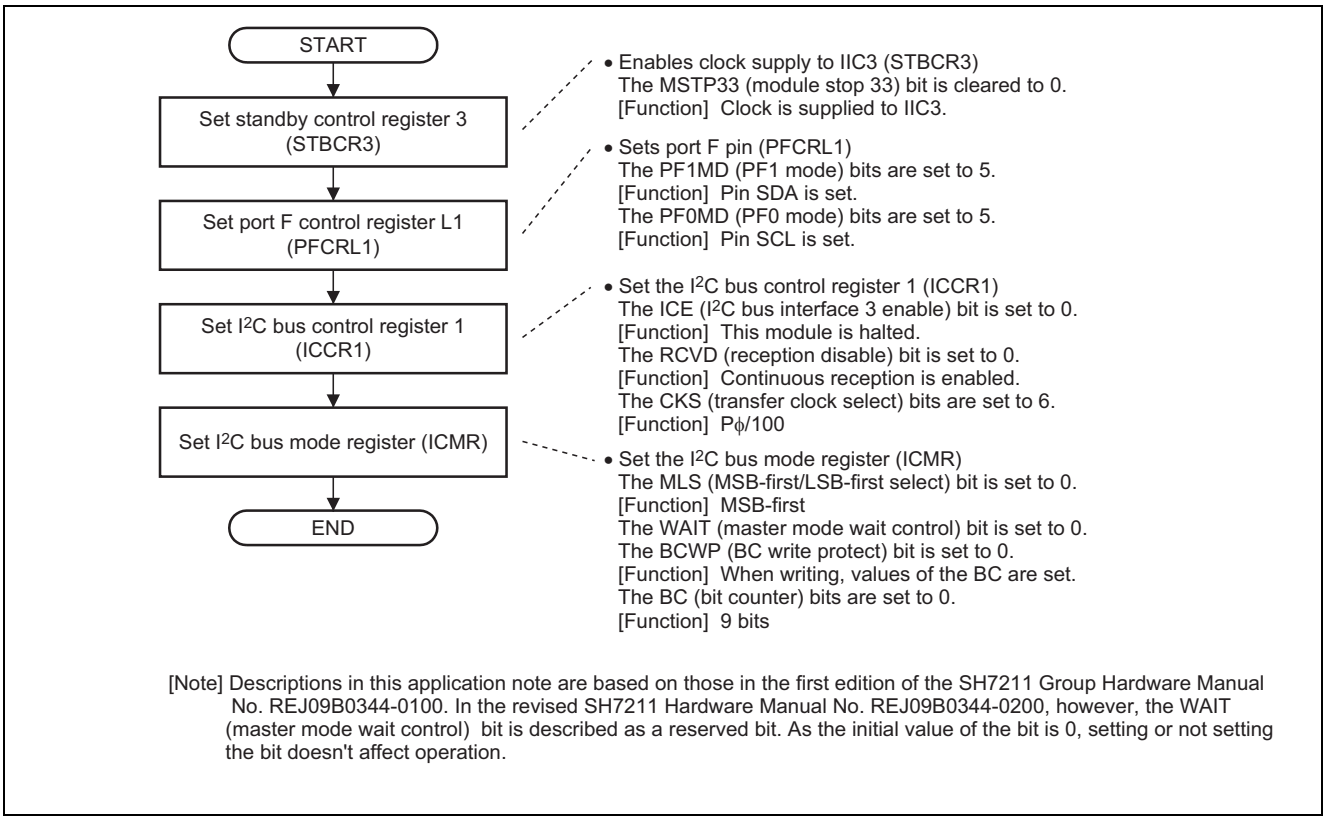


Figure 2 Example of the Initialization Sequence for I²C Bus Interface 3

2.3 Operation of the Sample Program

In this sample program, IIC3 is placed in master transmit mode, and reads out 10 bytes of data in sequence from the EEPROM.

The device code employed in this sample program is “B’1010”. Consult the datasheet of the EEPROM you are using for its device code.

The device address employed in this sample program is “B’000”. Consult the datasheet of the EEPROM you are using for its device address.

The memory address indicates the point where writing of EEPROM starts. Each time EEPROM is read, the address is incremented.

Figure 3 shows the operations for sequential reading, and figure 4 shows the operating environment of this sample program.

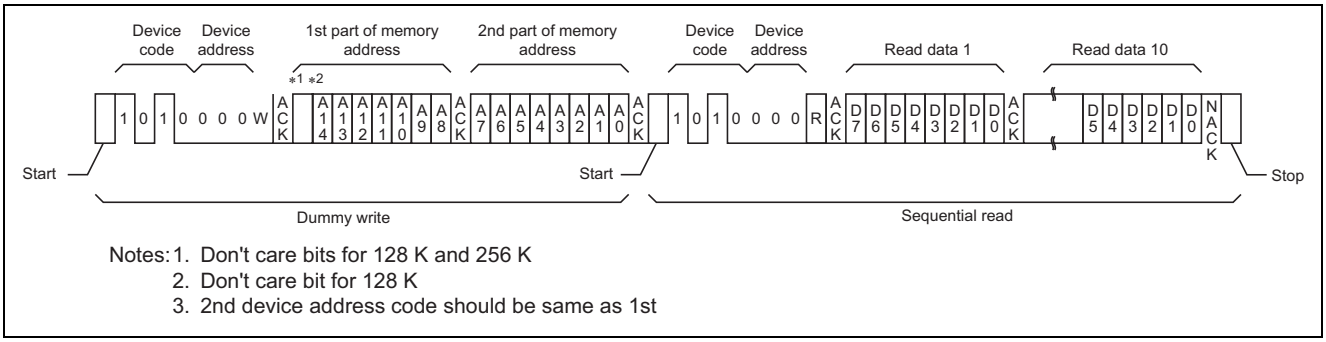


Figure 3 Operations for Sequential Reading

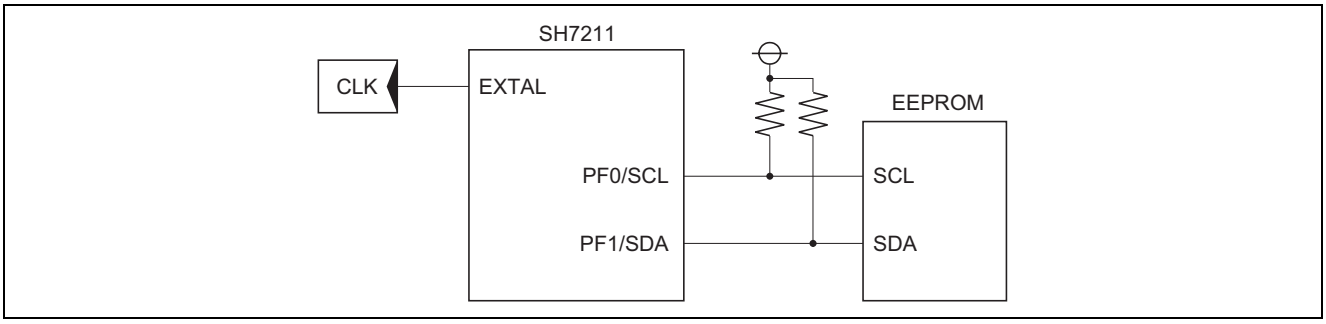


Figure 4 Operating Environment of the Sample Program

2.4 Sequence of Processing by the Sample Program

Table 2 gives the register settings in the sample program. Table 3 shows macro definitions in the sample program. Figures 5 to 10 show the flow of processing by the sample program.

Table 2 Register Settings Used in Sample Program

Register Name	Address	Setting Value	Description
Standby control register 3 (STBCR3)	H'FFFE 0408	H'00	MSTP33 = "0": IIC3 operates.
I²C bus control register 1 (ICCR1)	H'FFFE E000	H'E6	ICE = "1": SCL and SDA pins are placed in the bus-drive state. RCVD = "1": Continuous reception is disabled. MST = "1", TRS = "0": Master receive mode CKS = "B'0110": transfer rate Pϕ/100
I²C bus mode register (ICMR)	H'FFFE E002	H'30	MLS = "0": MSB-first BCWP = "0": Allows the writing of values to the BC bits. BC = "B'000":9 bits

Table 3 Macro Definitions Used in Sample Program

Macro Definition	Setting Value	Description
EEPROM_MEM_ADDR	H'0000	EEPROM start address
DEVICE_CODE	H'A0	Device code
DEVICE_ADDR	H'00	Device address
IIC_DATA_WR	H'00	Write code
IIC_DATA_RD	H'01	Read code
IIC3_DATA	10	Data transfer size

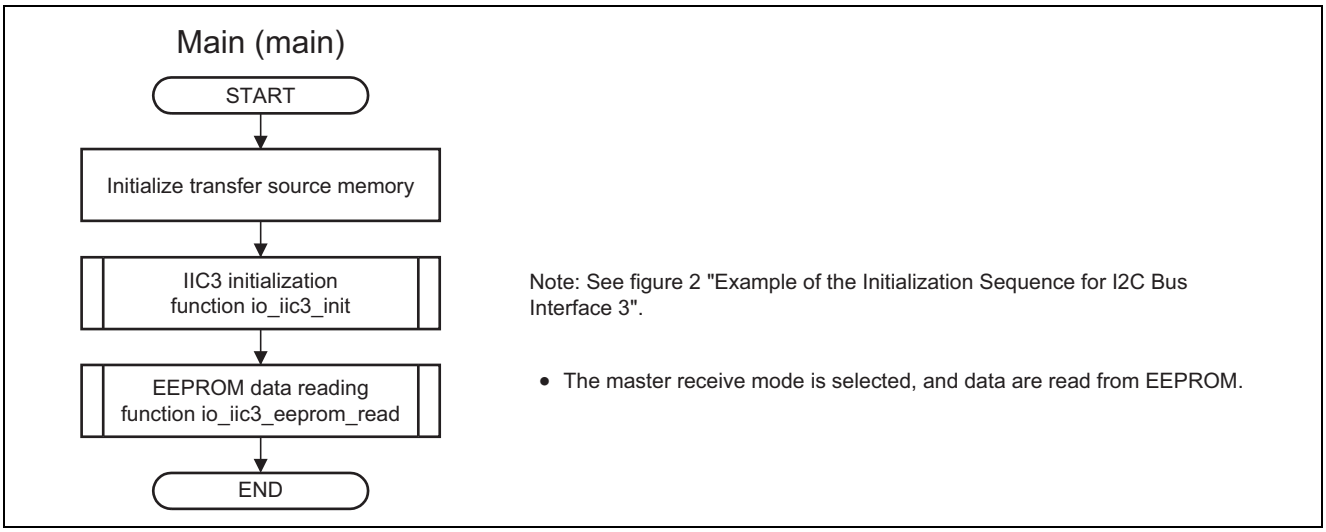


Figure 5 Flow of Processing by the Sample Program (1)

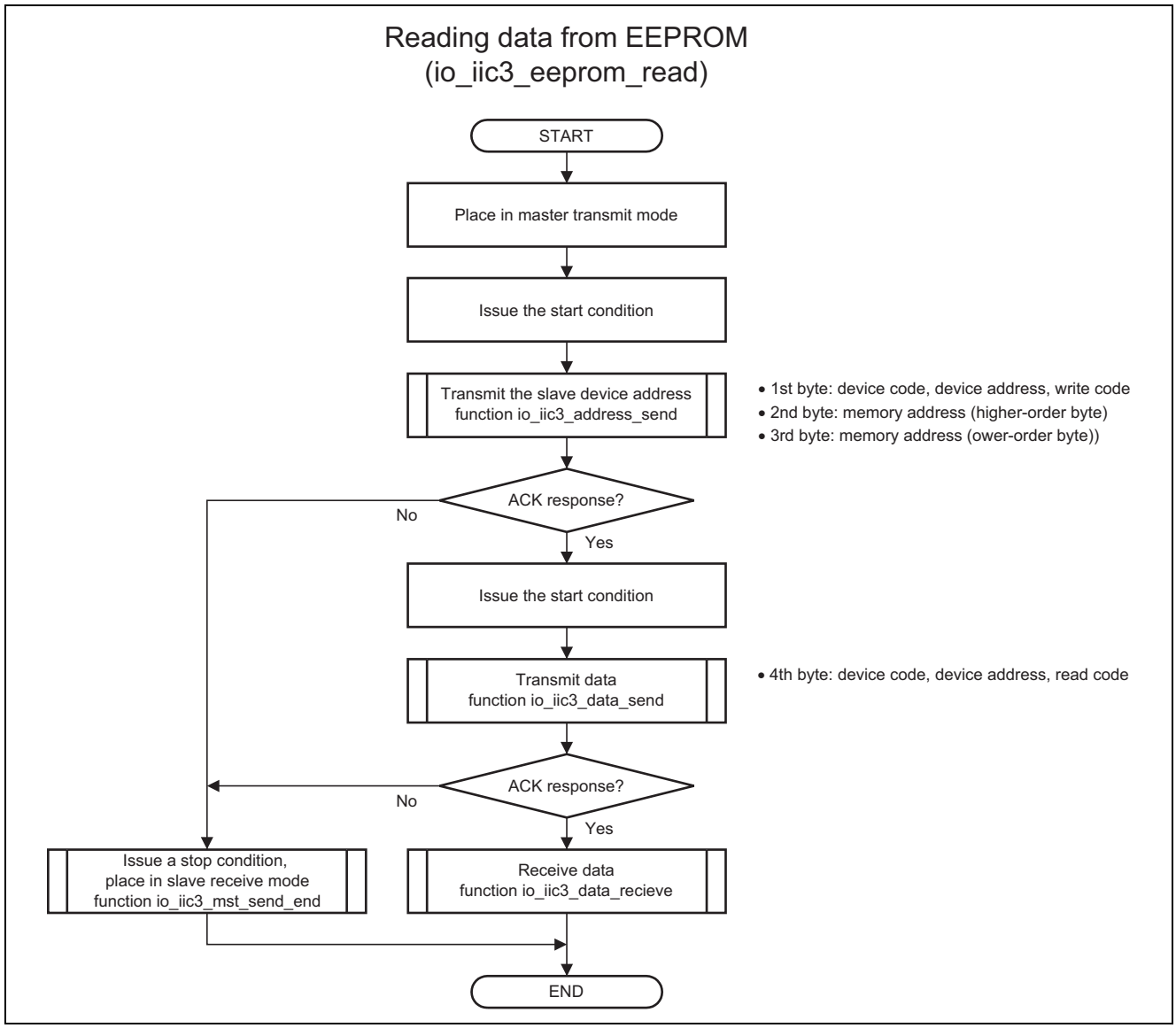


Figure 6 Flow of Processing by the Sample Program (2)

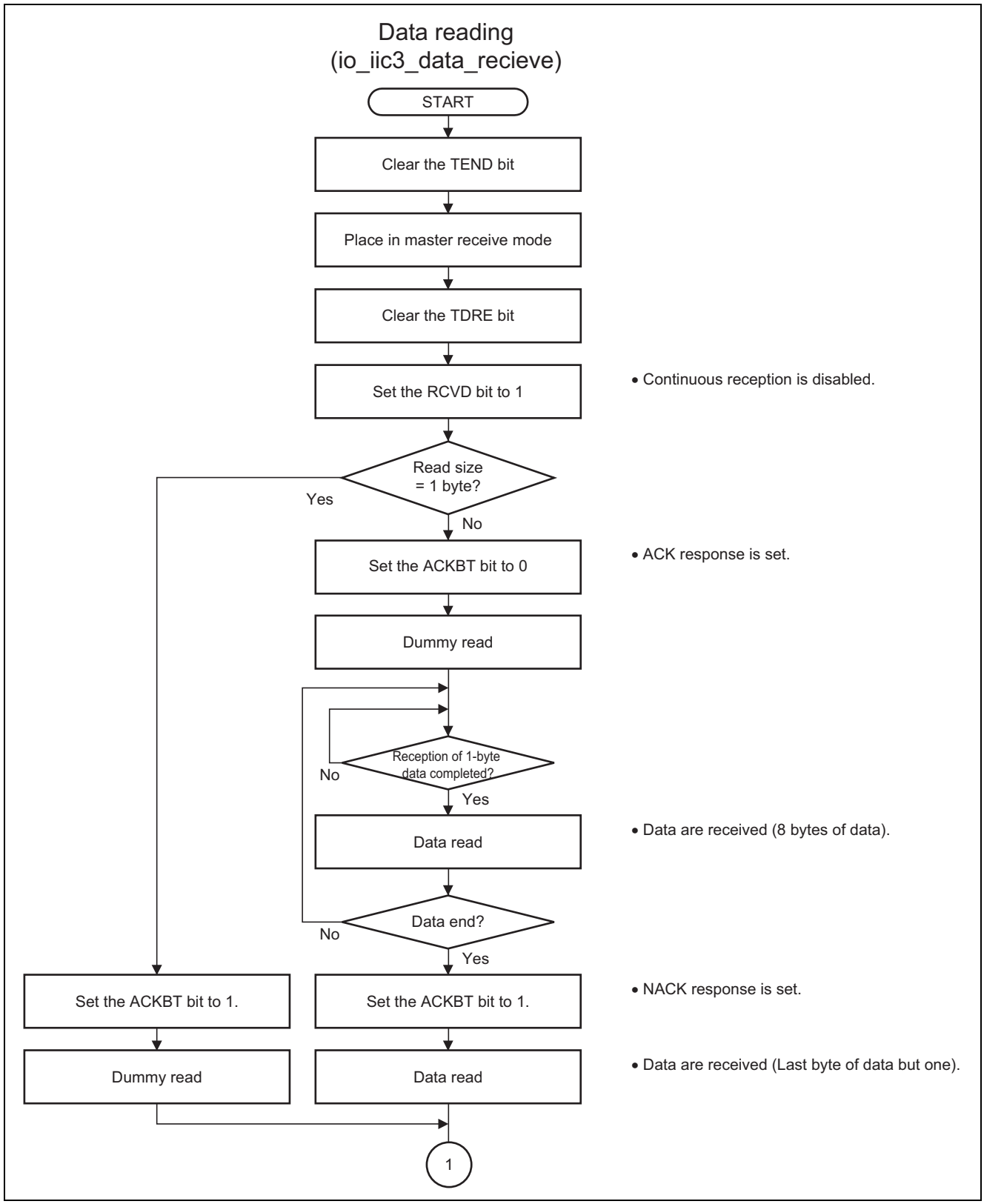
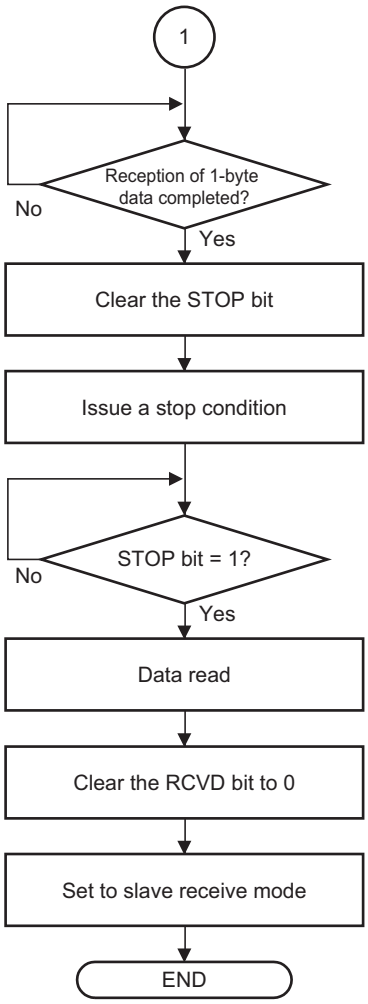


Figure 7 Flow of Processing by the Sample Program (3)



• Data are received (last byte of data).

Figure 8 Flow of Processing by the Sample Program (4)

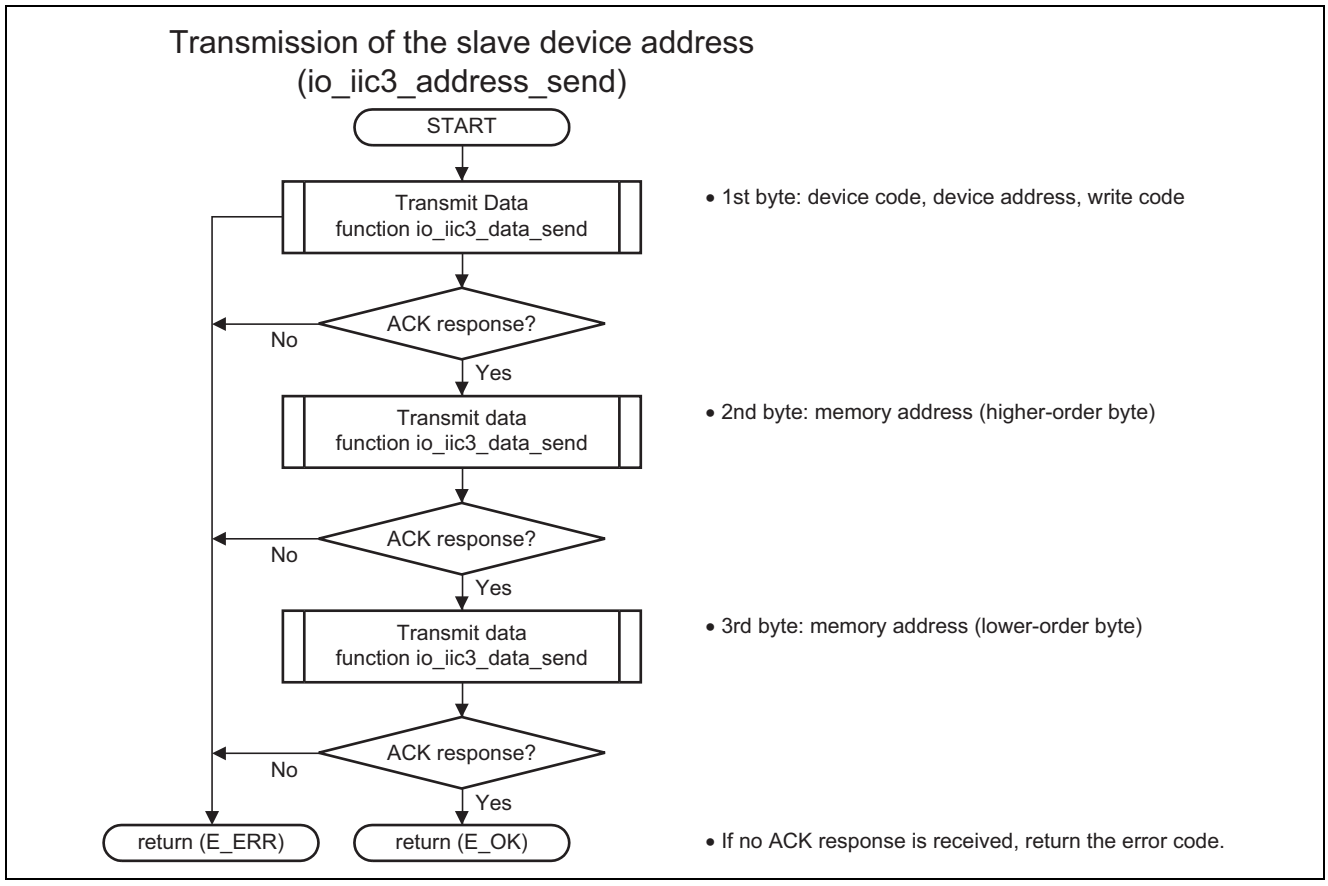


Figure 9 Flow of Processing by the Sample Program (5)

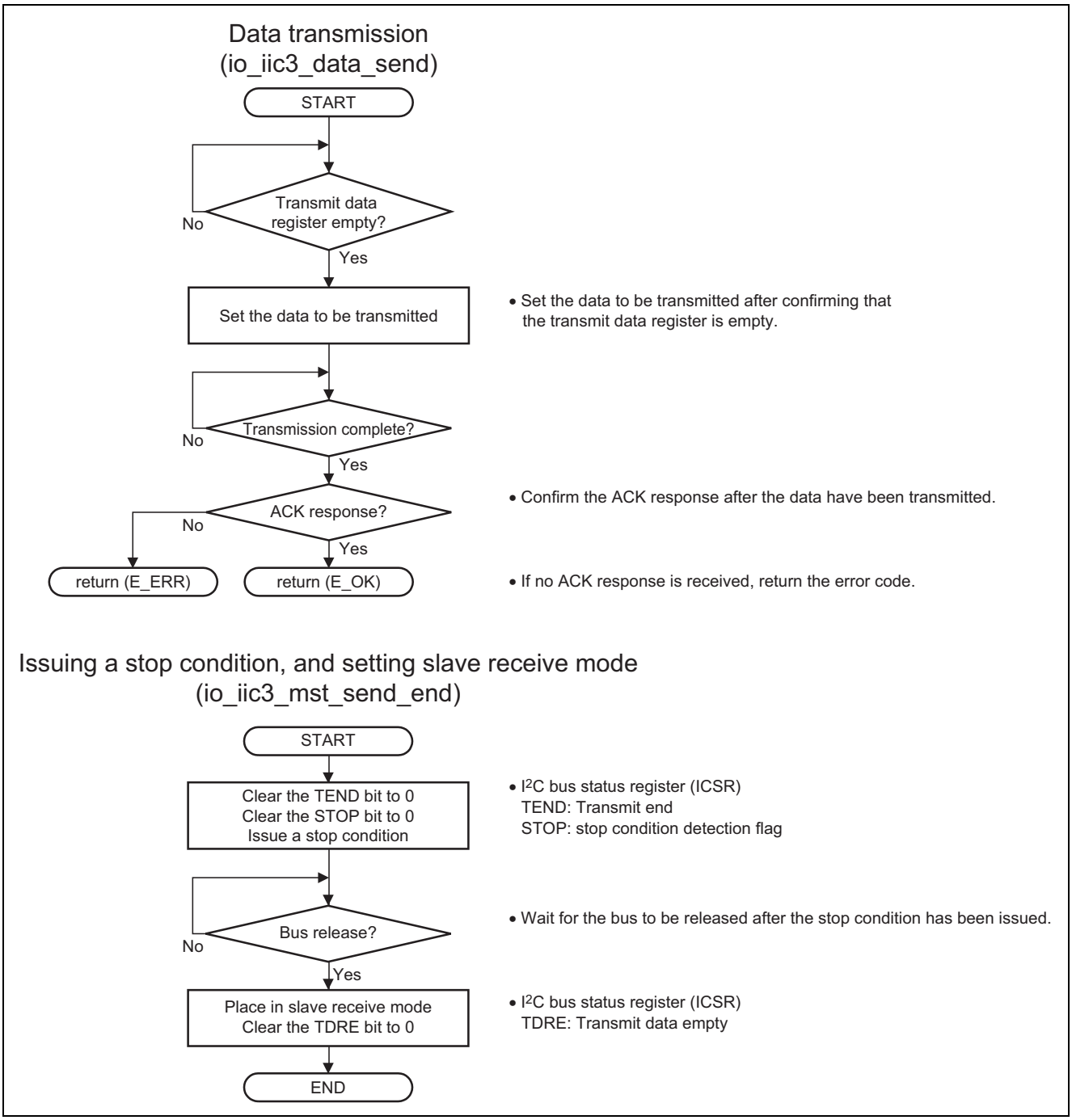


Figure 10 Flow of Processing by the Sample Program (6)

2.5 Note on Master Receive Mode

If the I²C bus receive data register (ICDRR) is read near the falling edge of the 8th clock cycle, the data will not be received in some cases. In addition, if the reception disable (RCVD) bit in the I²C bus control register is set to 1 near the falling edge of the 8th clock cycle while the receive buffer is full, a stop condition cannot be issued in some cases. To prevent these errors, one of the following two methods should be selected.
In this sample program, the RCVD bit is set to 1 to select data reception in byte units.

1. In master receive mode, reading the ICDRR must proceed before the falling edge of the 8th clock cycle.
2. In master receive mode, the RCVD bit should be set to 1 and the processing should be performed in byte units.

2.6 Note on Master Receive Mode with ACKBT Setting

In master receive mode operation, the ACKBT bit must be set before the falling edge of the 8th clock cycle on pin SCL falls in the final data transfer of consecutive data transfer. Otherwise, an overrun may occur on the slave device.

In the sample program, the RCVD bit is set to 1 to select data transfer in byte units; therefore, this note does not apply to this sample program.

3. Listing of the Sample Program

1. Sample Program Listing: main.c (1)

```
1           /*""FILE COMMENT""*****************************************************************
2           *
3           * System Name  : SH7211 Sample Program
4           * File Name    : main.c
5           * Contents     : Sample program for reception by IIC3 in master receive mode
6           * Version      : 1.00.00
7           * Model        : M3A-HS11
8           * CPU          : SH7211
9           * Compiler     : SHC9.1.1.0
10          * note         : Data are received from EEPROM
11          *                using IIC3 in master receive mode.
12          *
13          * The information described here may contain technical inaccuracies or
14          * typographical errors. Renesas Technology Corporation and Renesas Solutions
15          * assume no responsibility for any damage, liability, or other loss rising
16          * from these inaccuracies or errors.
17          *
18          * Copyright (C) 2008 Renesas Technology Corp. All Rights Reserved
19          * AND Renesas Solutions Corp. All Rights Reserved
20          *
21          * history : 2008.04.24 ver.1.00.00
22          *""FILE COMMENT END""*************************************************************/
23          #include <machine.h>
24          #include "iodefine.h"     /* SH7211 iodefine */
25
26          /* ==== symbol definition ==== */
27          #define EEPROM_MEM_ADDR 0x0000
28          #define DEVICE_CODE 0xA0   /* EEPROM device code :b'1010  */
29          #define DEVICE_ADDR 0x00   /* EEPROM device address:b'000 */
30          #define IIC_DATA_WR 0x00   /* Data write code :b'0        */
31          #define IIC_DATA_RD 0x01   /* Data read code :b'1         */
32          #define IIC3_DATA 10
33
34          #define E_OK 0
35          #define E_ERR -1
36          #define HIGH 1
37          #define LOW 0
38
39          /* ==== RAM allocation variable declaration ==== */
40          unsigned char ReadData[IIC3_DATA];
41
42          /* ==== Prototype declaration ==== */
43          void main(void);
44          void io_iic3_mst_send_end(void);
45          int io_iic3_init(void);
46          int io_iic3_eeprom_read(unsigned char d_code,unsigned char d_adr,unsigned short r_adr,
47                                  unsigned int r_size,unsigned char* r_buf);
48          int io_iic3_data_recieve(unsigned char* r_buf,unsigned int r_size);
49          int io_iic3_data_send(unsigned char data);
50          int io_iic3_address_send(unsigned char* data);
```

2. Sample Program Listing: main.c (2)

```
51          /*""FUNC COMMENT""*************************************************************
52          * Outline      : Sample program main
53          *-----------------------------------------------------------------------------
54          * Include      :
55          *-----------------------------------------------------------------------------
56          * Declaration  : void main(void);
57          *-----------------------------------------------------------------------------
58          * Function     : Sample program main
59          *-----------------------------------------------------------------------------
60          * Argument     : void
61          *-----------------------------------------------------------------------------
62          * Return Value : void
63          *-----------------------------------------------------------------------------
64          * Notice       :
65          *""FUNC COMMENT END""*********************************************************/
66          void main(void)
67          {
68              int i;
69              /* ==== Clear the data storage location ==== */
70              for(i=0;i<IIC3_DATA;i++){
71                      ReadData[i] = 0x00;
72              }
73              /* ==== IIC3 initialization setting ==== */
74              io_iic3_init();
75
76              /* ==== Reception by IIC3 in master receive mode ==== */
77              io_iic3_eeprom_read(   DEVICE_CODE,       /* Device code */
78                                     DEVICE_ADDR,       /* Device address */
79                                     0x0000,            /* Start address for reading out data */
80                                     sizeof(ReadData),  /* Read data size */
81                                     ReadData);         /* Data storage location */
82              while(1){
83                          /* Loop */
84              }
85          }
86
```

3. Sample Program Listing: main.c (3)

```
87          /*""FUNC COMMENT""*************************************************************
88          * Outline      : IIC3 module initialization
89          *-------------------------------------------------------------------------
90          * Include      : #include "iodefine.h"
91          *-------------------------------------------------------------------------
92          * Declaration  : int io_iic3_init(void);
93          *-------------------------------------------------------------------------
94          * Function     : IIC3 module initialization
95          *-------------------------------------------------------------------------
96          * Argument     : void
97          *-------------------------------------------------------------------------
98          * Return Value : E_OK
99          *-------------------------------------------------------------------------
100         * Notice       :
101         *""FUNC COMMENT END""*************************************************/
102         int io_iic3_init(void)
103         {
104             STB.CR3.BIT._IIC3 = 0x00;         /* Clear the IIC3 module standby mode */
105
106             /* ---- PFC setting ---- */
107             PFC.PFCRL1.BIT.PF0MD = 0x05;      /* SCL selection */
108             PFC.PFCRL1.BIT.PF1MD = 0x05;      /* SDA selection */
109             /* ----IIC31 module operation disabled ---- */
110             IIC3.ICCR1.BIT.ICE = 0x00;        /* IIC transfer disabled state */
111             IIC3.ICCR1.BIT.ICE = 0x01;        /* IIC3 module operation is enabled */
112             IIC3.ICCR1.BIT.RCVD = 0x00;       /* Continuous reception is to proceed */
113             IIC3.ICCR1.BIT.CKS = 0x06;        /* Transfer rate: Pϕ/100(400 kHz) */
114
115         /* ---IIC bus mode register (ICMR) setting --- */
116             IIC3.ICMR.BYTE = 0x30;
117                         /*
118                                     bit7    : MLS:0 --------------------  MSB first
119                                     bit6    : WAIT:0 -------------------  No WAIT insertion
120                                     bit5-4  : Reserve:1 ----------------  Reserve bit
121                                     bit3    : BCWP:0--------------------  Unsetting
122                                     bit2-0  : BC0:0, BC1:0,BC0:0--------  IIC format 9-bit
123                         */
124             return(E_OK);
125         }
126
```

4. Sample Program Listing: main.c (4)

```
/*""FUNC COMMENT""*************************************************
* Outline      : EEPROM data read
*-----------------------------------------------------------------
* Include      : #include "iodefine.h"
*-----------------------------------------------------------------
* Declaration  : int io_iic3_eeprom_read( unsigned char d_code,
*              :                          unsigned char d_adr,
*              :                          unsigned short r_adr,
*              :                          unsigned int r_size,
*              :                          unsigned char* r_buf);
*-----------------------------------------------------------------
* Function     : The amount of data specified by “r_size” are read out from the
*              : EEPROM specified by device code “d_code” and device address “d_adr”,
*              : and are stored in the area specified by “r_buf”.
*              : Memory addresses within the EEPROM are specified by “r_adr”.
*-----------------------------------------------------------------
* Argument     : unsigned char d_code     : Device code
*              : unsigned char d_adr      : Device address
*              : unsigned short r_adr     : Address where data to be read in starts
*              : unsigned int r_size      : Amount of data to be read-in
*              : unsigned char* r_buf     : Location of data to be read-in
*-----------------------------------------------------------------
* Return Value : With ACK response        : E_OK
*              : With no ACK response     : E_ERR
*-----------------------------------------------------------------
* Notice :
*""FUNC COMMENT END""*************************************************/
int io_iic3_eeprom_read(unsigned char d_code,unsigned char d_adr,unsigned short r_adr,
                        unsigned int r_size,unsigned char* r_buf)
{
    int ack = E_OK;
    unsigned char send[4];

    send[0] = (unsigned char)(d_code|((d_adr & 0x7)<<1)|IIC_DATA_WR);
    send[1] = (unsigned char)((r_adr>>8) & 0x00ff);
    send[2] = (unsigned char)(r_adr & 0x00ff);
    send[3] = (unsigned char)(d_code|((d_adr & 0x7)<<1)|IIC_DATA_RD);

    while(IIC3.ICCR2.BIT.BBSY == 0x01){
              /* Waiting for bus release */
    }
    IIC3.ICCR1.BYTE |= 0x30;                    /* Set to master transmission mode */
    IIC3.ICCR2.BYTE = ((IIC3.ICCR2.BYTE & 0xbf)|0x80);      /* Issue the start condition */

    ack = io_iic3_address_send(send);
                          /* Transmit the first, second, and third bytes of data */

    if(ack == E_OK){
           /* ACK response is received from the specified device */
           IIC3.ICCR2.BYTE=((IIC3.ICCR2.BYTE & 0xbf) | 0x80);
                                                    /* Issue a start condition */
           ack = io_iic3_data_send(send[3]);     /* Transmit the fourth byte of data */
           if(ack == E_OK){
                     io_iic3_data_recieve(r_buf,r_size);   /* Data reception */
           }
           else{
                     io_iic3_mst_send_end();
           }
    }
    else{
           /* ACK response is not received from the specified device */
           io_iic3_mst_send_end();
    }
    return(ack);
```

5. Sample Program Listing: main.c (5)

```
191       }
192       /*""FUNC COMMENT""*************************************************
193       * Outline        : Master receive mode
190       *-----------------------------------------------------------------
191       * Include        : #include "iodefine.h"
192       *-----------------------------------------------------------------
193       * Declaration    : int io_iic3_data_recieve(unsigned char* r_buf,
194       *                :                          unsigned int r_size);
195       *-----------------------------------------------------------------
196       * Function       : The amount of data specified by “r_size” are received
197       *                : in master receive mode and are stored in the area specified by
198       *                : “r_buf”. After the specified number of data have been received,
199       *                : slave receive mode is selected.
200       *-----------------------------------------------------------------
201       * Argument       : unsigned char* r_buf     : Location of data to be read in
202       *                : unsigned int r_size      : Amount of data to be read in
203       *-----------------------------------------------------------------
204       * Return Value   : Always E_OK
205       *-----------------------------------------------------------------
206       * Notice     :
207       *""FUNC COMMENT END""*********************************************/
208       int io_iic3_data_recieve(unsigned char* r_buf,unsigned int r_size)
209       {
210            int i;
211            unsigned char dummy;
212            IIC3.ICSR.BIT.TEND = 0x00;                  /* Clear bit TEND */
213            IIC3.ICCR1.BIT.TRS = 0x00;                  /* Master receive mode */
214            IIC3.ICSR.BIT.TDRE = 0x00;                  /* Clear bit TDRE */
215            IIC3.ICCR1.BIT.RCVD = 0x01;                 /* Disable continuous reception */
216            if(r_size == 1){                            /* When one byte of data is received */
217                       IIC3.ICIER.BIT.ACKBT = 0x01;     /* Acknowledge setting “H” */
218                       dummy = IIC3.ICDRR;              /* Dummy read */
219            }
220            else{
221                       IIC3.ICIER.BIT.ACKBT = 0x00;     /* Acknowledge setting “L”  */
222                       dummy = IIC3.ICDRR;              /* Dummy read */
223                       for(i=0;i<r_size - 2;i++){
224                              while(IIC3.ICSR.BIT.RDRF == 0x00){
225                                        /* Waiting for reception of one-byte data */
226                              }
227                              *r_buf++ = IIC3.ICDRR;    /* Data read */
228                       }
229                       while(IIC3.ICSR.BIT.RDRF == 0x00){
230                              /* Waiting for reception of one byte of data */
231                     }
232
233                       IIC3.ICIER.BIT.ACKBT = 0x01;     /* Acknowledge setting “H” */
234                       *r_buf++ = IIC3.ICDRR;           /* Next to last byte of data */
235            }
236            while(IIC3.ICSR.BIT.RDRF == 0x00){
237                   /* Waiting for reception of one byte of data */
238            }
239            IIC3.ICSR.BIT.STOP = 0x00;                  /* Clear the STOP flag */
240            IIC3.ICCR2.BYTE &= 0x3f;                    /* Issue a stop condition */
241            while(IIC3.ICSR.BIT.STOP == 0x00){
242                   /* Waiting for generation of a stop condition */
243            }
244
245            *r_buf = IIC3.ICDRR;                        /* Last byte of data */
246            IIC3.ICCR1.BIT.RCVD = 0x00;                 /* Clear bit RCVD */
247            IIC3.ICCR1.BYTE &= 0xcf;                    /* Slave receive mode */
252
253            return(E_OK);
254       }
```

6. Sample Program Listing: main.c (6)

```
248       /*""FUNC COMMENT""************************************************
249       * Outline        : Transmission of the slave device address
250       *-----------------------------------------------------------------
251       * Include        :
252       *-----------------------------------------------------------------
253       * Declaration    : int io_iic3_address_send(unsigned char* data);
254       *-----------------------------------------------------------------
255       * Function       : Transmission of the slave device address specified by “data” (one byte)
256       *                : and the memory address (two bytes)
257       *-----------------------------------------------------------------
258       * Argument       : unsigned char* data  : Transmit data
259       *-----------------------------------------------------------------
260       * Return Value   : With ACK response    : E_OK
261       *                : With no ACK response     : E_ERR
262       *-----------------------------------------------------------------
263       * Notice     :
264       *""FUNC COMMENT END""*********************************************/
265       int io_iic3_address_send(unsigned char* data)
266       {
267            int ack;
268
269            ack = io_iic3_data_send(*data++);         /* Slave device address */
270            if(ack == E_ERR){
271                      return(ack);
272            }
273            ack = io_iic3_data_send(*data++);         /* 1st part of memory address */
274            if(ack == E_ERR){
275                      return(ack);
276            }
277            ack = io_iic3_data_send(*data);           /* 2nd part of memory address */
278            if(ack == E_ERR){
279                      return(ack);
280            }
281            return(ack);
282       }
283       /*""FUNC COMMENT""************************************************
284       * Outline        : Transmission of one byte of data
285       *-----------------------------------------------------------------
286       * Include        : #include "iodefine.h"
287       *-----------------------------------------------------------------
288       * Declaration    : int io_iic3_data_send(unsigned char data);
289       *-----------------------------------------------------------------
290       * Function       : Data are transmitted according to the following procedure.
291       *                : 1.Wait for ICDRT to become empty.
292       *                : 2.Set the data to be transmitted.
293       *                : 3.Check completion of data transmission.
294       *                : 4.Check the ACK response.
295       *-----------------------------------------------------------------
296       * Argument       : unsigned char data   : Data for transmission
297       *-----------------------------------------------------------------
298       * Return Value   : With ACK response    : E_OK
299       *                : With no ACK response     : E_ERR
307       *-----------------------------------------------------------------
308       * Notice     :
309       *""FUNC COMMENT END""*********************************************/
310
```

7. Sample Program Listing: main.c (7)

```
300         int io_iic3_data_send(unsigned char data)
301         {
302             int ack;
303
304             while(IIC3.ICSR.BIT.TDRE == 0x0){
305                         /* Wait for ICDRT to become empty. */
306             }
307             IIC3.ICDRT = data;
308             while(IIC3.ICSR.BIT.TEND == 0x00){
309                         /* Wait for completion of data transmission */
310             }
311             if(IIC3.ICIER.BIT.ACKBR == 0){
312                         ack = E_OK;
313             }
314             else{
315                         ack = E_ERR;
316             }
317             return(ack);
318         }
319         /*""FUNC COMMENT""*****************************************************
320         * Outline      : Issuing a stop condition
321         *---------------------------------------------------------------------
322         * Include      : #include "iodefine.h"
323         *---------------------------------------------------------------------
324         * Declaration  : void io_iic3_mst_send_end(void);
325         *---------------------------------------------------------------------
326         * Function     : A stop condition is issued and slave receive mode is set.
327         *---------------------------------------------------------------------
328         * Argument     : void
329         *---------------------------------------------------------------------
330         * Return Value : void
331         *---------------------------------------------------------------------
332         * Notice       :
333         *""FUNC COMMENT END""*************************************************/
334         void io_iic3_mst_send_end(void)
335         {
336             IIC3.ICSR.BIT.TEND = 0x00;        /* Clear bit TEND */
337             IIC3.ICSR.BIT.STOP = 0x00;        /* Clear the STOP flag */
338             IIC3.ICCR2.BYTE &= 0x3f;          /* Issue a stop condition */
339
340             while(IIC3.ICSR.BIT.STOP == 0x00){
341                         /* Wait for bus release */
342             }
343
344             IIC3.ICCR1.BYTE &= 0xcf;          /* Slave receive mode */
345             IIC3.ICSR.BIT.TDRE = 0x00;        /* Clear bit TDRE */
346         }
347         /* End of File */
```

4. Documents for Reference

- Software Manual
  SH-2A, SH2A-FPU Software Manual (REJ09B0051)
  The most up-to-date version of this document is available on the Renesas Technology Website.
- Hardware Manual
  SH7211 Group Hardware Manual (REJ09B0344)
  The most up-to-date version of this document is available on the Renesas Technology Website.

Website and Support

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

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

Revision Record

		Description	
Rev.	**Date**	**Page**	**Summary**
1.00	Nov.19.08	—	First edition issued

Notes regarding these materials

1. This document is provided for reference purposes only so that Renesas customers may select the appropriate Renesas products for their use. Renesas neither makes warranties or representations with respect to the accuracy or completeness of the information contained in this document nor grants any license to any intellectual property rights or any other rights of Renesas or any third party with respect to the information in this document.
2. Renesas shall have no liability for damages or infringement of any intellectual property or other rights arising out of the use of any information in this document, including, but not limited to, product data, diagrams, charts, programs, algorithms, and application circuit examples.
3. You should not use the products or the technology described in this document for the purpose of military applications such as the development of weapons of mass destruction or for the purpose of any other military use. When exporting the products or technology described herein, you should follow the applicable export control laws and regulations, and procedures required by such laws and regulations.
4. All information included in this document such as product data, diagrams, charts, programs, algorithms, and application circuit examples, 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 products listed in this document, please confirm the latest product information with a Renesas sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas such as that disclosed through our website. (http://www.renesas.com)
5. Renesas has used reasonable care in compiling the information included in this document, but Renesas assumes no liability whatsoever for any damages incurred as a result of errors or omissions in the information included in this document.
6. When using or otherwise relying on the information in this document, you should evaluate the information in light of the total system before deciding about the applicability of such information to the intended application. Renesas makes no representations, warranties or guaranties regarding the suitability of its products for any particular application and specifically disclaims any liability arising out of the application and use of the information in this document or Renesas products.
7. With the exception of products specified by Renesas as suitable for automobile applications, Renesas products are not designed, manufactured or tested for applications or otherwise in systems the failure or malfunction of which may cause a direct threat to human life or create a risk of human injury or which require especially high quality and reliability such as safety systems, or equipment or systems for transportation and traffic, healthcare, combustion control, aerospace and aeronautics, nuclear power, or undersea communication transmission. If you are considering the use of our products for such purposes, please contact a Renesas sales office beforehand. Renesas shall have no liability for damages arising out of the uses set forth above.
8. Notwithstanding the preceding paragraph, you should not use Renesas products for the purposes listed below:
   (1) artificial life support devices or systems
   (2) surgical implantations
   (3) healthcare intervention (e.g., excision, administration of medication, etc.)
   (4) any other purposes that pose a direct threat to human life

   Renesas shall have no liability for damages arising out of the uses set forth in the above and purchasers who elect to use Renesas products in any of the foregoing applications shall indemnify and hold harmless Renesas Technology Corp., its affiliated companies and their officers, directors, and employees against any and all damages arising out of such applications.
9. You should use the products described herein within the range specified by Renesas, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas shall have no liability for malfunctions or damages arising out of the use of Renesas products beyond such specified ranges.
10. Although Renesas endeavors to improve the quality and reliability of its products, IC products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Please be sure to implement safety measures to guard against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas 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 applicable measures. Among others, since the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or system manufactured by you.
11. In case Renesas products listed in this document are detached from the products to which the Renesas products are attached or affixed, the risk of accident such as swallowing by infants and small children is very high. You should implement safety measures so that Renesas products may not be easily detached from your products. Renesas shall have no liability for damages arising out of such detachment.
12. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written approval from Renesas.
13. Please contact a Renesas sales office if you have any questions regarding the information contained in this document, Renesas semiconductor products, or if you have any other inquiries.