R8C/56E Group

I^{2}C bus Single Master Control Program (Master Transmit/Receive)

R01AN1288EJ0100
Rev.1.00
Nov 30, 2012

Abstract

This document describes an I^{2}C bus single master control program (master transmit/receive) using I^{2}C bus interface.

Products

R8C/56E Group

When using this application note with other Renesas MCUs, careful evaluation is recommended after making modifications to comply with the alternate MCU.

Contents

1. Specifications

Transmit and receive data in master mode. The usage conditions are described below. The specifications of this application note conform to the I^{2}C bus communication protocol.

Usage conditions

- Slave address length: 7 bits
- Transfer rate: Approximately 357 kHz (standard mode and fast mode are supported)
- Single master communication (multi-master is not supported)
- Restart condition generation is not supported

Table 1.1 lists the Peripheral Function and Its Application and Figure 1.1 shows a Block Diagram.

Table 1.1 Peripheral Function and Its Application

Peripheral Function	Application
Clock synchronous serial interface (I^{2}C bus interface mode)	Transmit and receive data

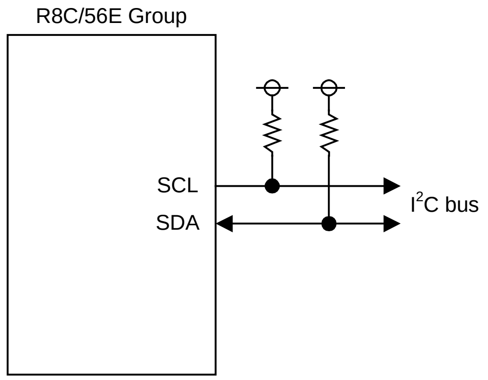


Figure 1.1 Block Diagram

2. Operation Confirmation Conditions

The sample code accompanying this application note has been run and confirmed under the conditions below.

Table 2.1 Operation Confirmation Conditions

Item	Contents
MCU used	R8C/56E Group
Operating frequencies	• XIN clock: 20 MHz • System clock: 20 MHz • CPU clock: 20 MHz
Operating voltage	5.0 V (2.7 to 5.5 V)
Integrated development environment	Renesas Electronics Corporation High-performance Embedded Workshop Version 4.09
C compiler	Renesas Electronics Corporation M16C Series, R8C Family C Complier V.5.45 Release 01
	Compile options -D__UART0__ -c -finfo -dir "$(CONFIGDIR)" -R8C (Default setting is used in the integrated development environment.)

3. Software

Transmit 3 bytes of data in master transmit mode and then receive 3 bytes of data in master receive mode. Repeat master transmit and master receive alternately.

Settings

- Use the P3_5/SCL pin for the serial clock I/O.
- Use the P3_7/SDA pin for the serial data I/O.
- Set the I^2C bus interface mode to master mode.
- Use channel I^2C_0.
- Set the transfer clock to f1/56 (set the transfer rate to approximately 357 kHz).
- Select no wait states (data and the acknowledge bit are transferred consecutively) for the wait insertion time.
- Use the MSB first for the transfer format.
- Set SDA digital delay value to 3 × f1 cycles.
- Use the receive acknowledge bit (ACKBR bit) to determine an acknowledge signal.
- Use the receive data full interrupt request.
- Use the transmit end interrupt request.
- Use the stop condition detection interrupt request.
- Do not use the transmit data empty interrupt request.
- Do not use the NACK receive interrupt request or arbitration lost/overrun error interrupt request.

Formula for calculating the transfer rate

Transfer rate = Settings for bits CKS3 to CKS0 in the SICR1_0 register

= 20 MHz (f1) ÷ 56

= 357.142 kHz

Notes on using this program

- When using the I^2C bus function, access the registers SITDR_0 and SIRDR_0 in 8-bit units.
- In the register definition file which is automatically generated by High-performance Embedded Workshop, data type of the above registers is defined as unsigned short. Therefore, those registers need to be configured as union byte_def or unsigned short in the program.

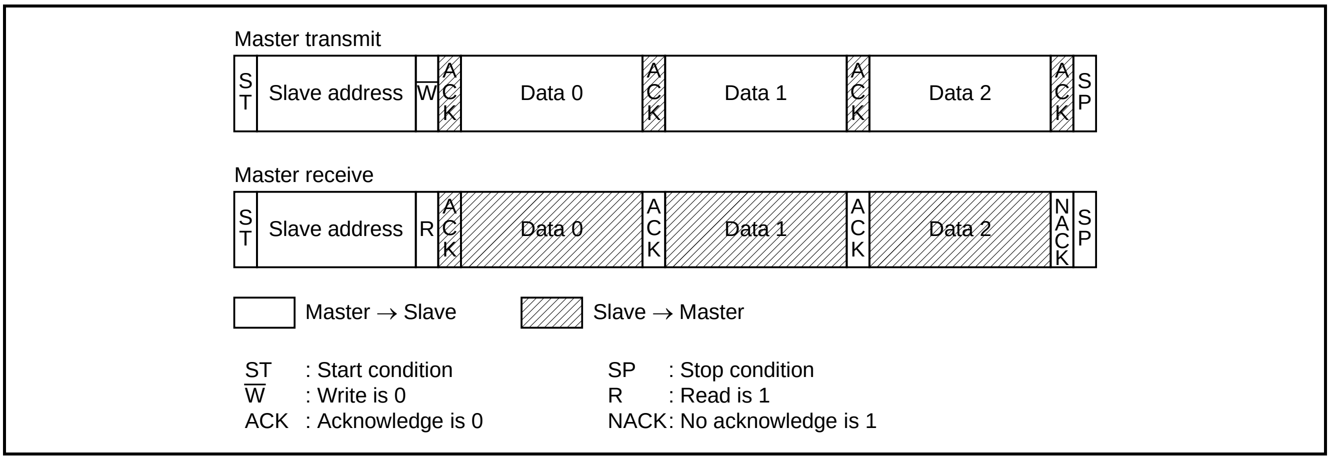


Figure 3.1 Transfer Format

3.1 Operation Overview

The processing outline is described below. Numbers in parenthesis correspond to the parenthesized numbers in Figures 3.2 to 3.5.

(1) Initial setting

Set the system clock and SFRs associated with I^2C bus interface, and initialize the variables used.

(2) Start master control

Generate a start condition. Enable the I^2C bus interface interrupt (transmit end interrupt request) to transmit the slave address.

(3) I^2C bus interface interrupt (transmit end interrupt request)

An interrupt is generated at the rising edge of the ninth bit of the SCL clock.

At master transmit:

- Determine ACK/NACK and set the next byte of transmit data.

At master receive:

- Determine ACK/NACK and set the next byte of transmit data if it is ACK.
- To complete communication, disable the transmit end interrupt request and receive data full interrupt request. Then, generate a stop condition and enable the stop condition detection interrupt request.

(4) I^2C bus interface interrupt (receive data full interrupt request)

At master receive, an interrupt is generated at the rising edge of the ninth bit of the SCL clock. Set the next byte ACK/NACK and read the receive data. To complete communication, disable the transmit end interrupt request and receive data full interrupt request. Then generate a stop condition and enable the stop condition detection interrupt request.

(5) I^2C bus interface interrupt (stop condition detection interrupt request)

An interrupt is generated when the stop condition is detected. Disable the stop condition detection interrupt request. Read the last receive data at master receive. Set to slave receive mode and disable the I^2C bus interface interrupt.

Figure 3.2 shows an Outline Flowchart and Figure 3.3 to Figure 3.5 show the Timing Diagrams.

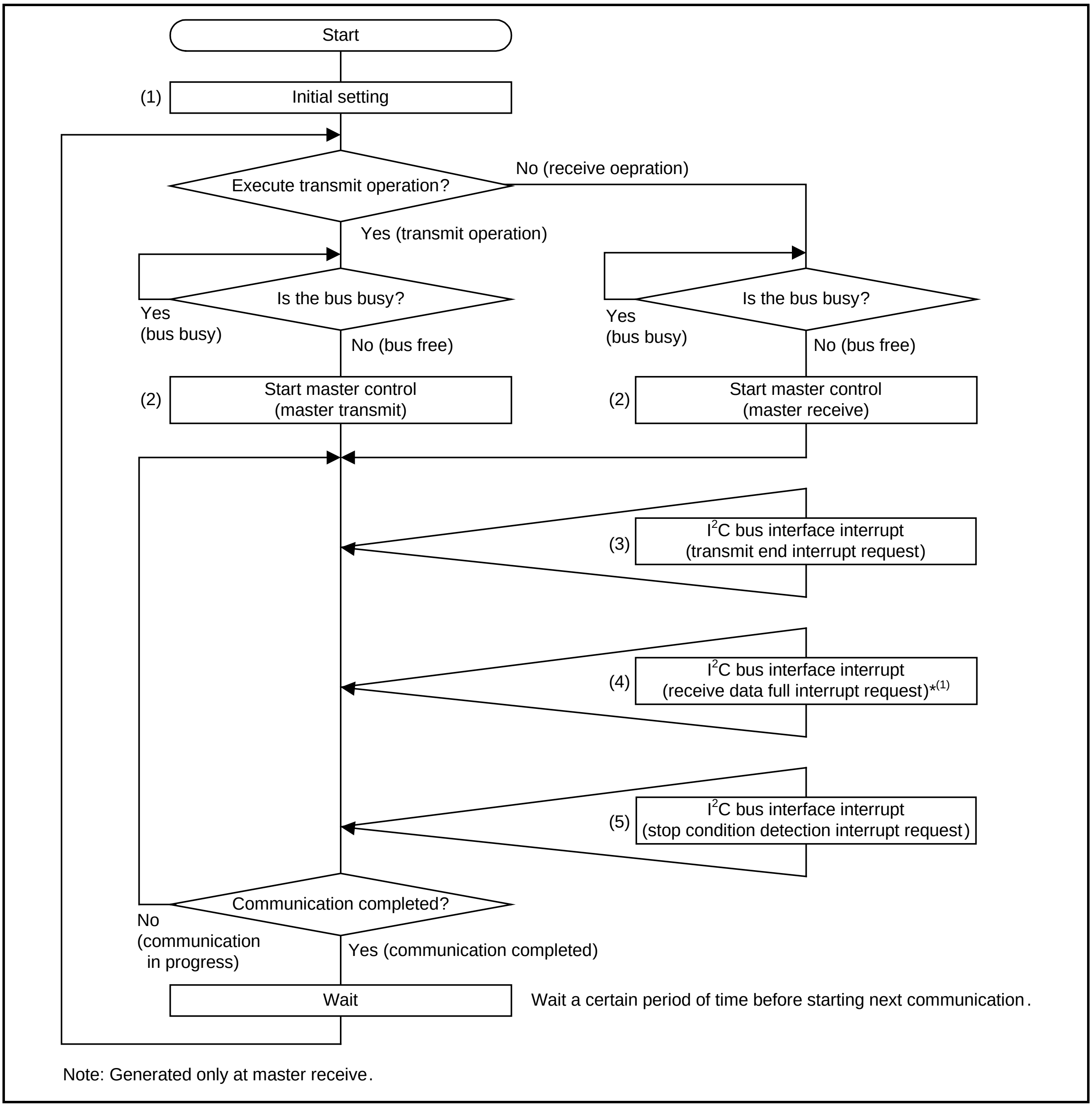


Figure 3.2 Outline Flowchart

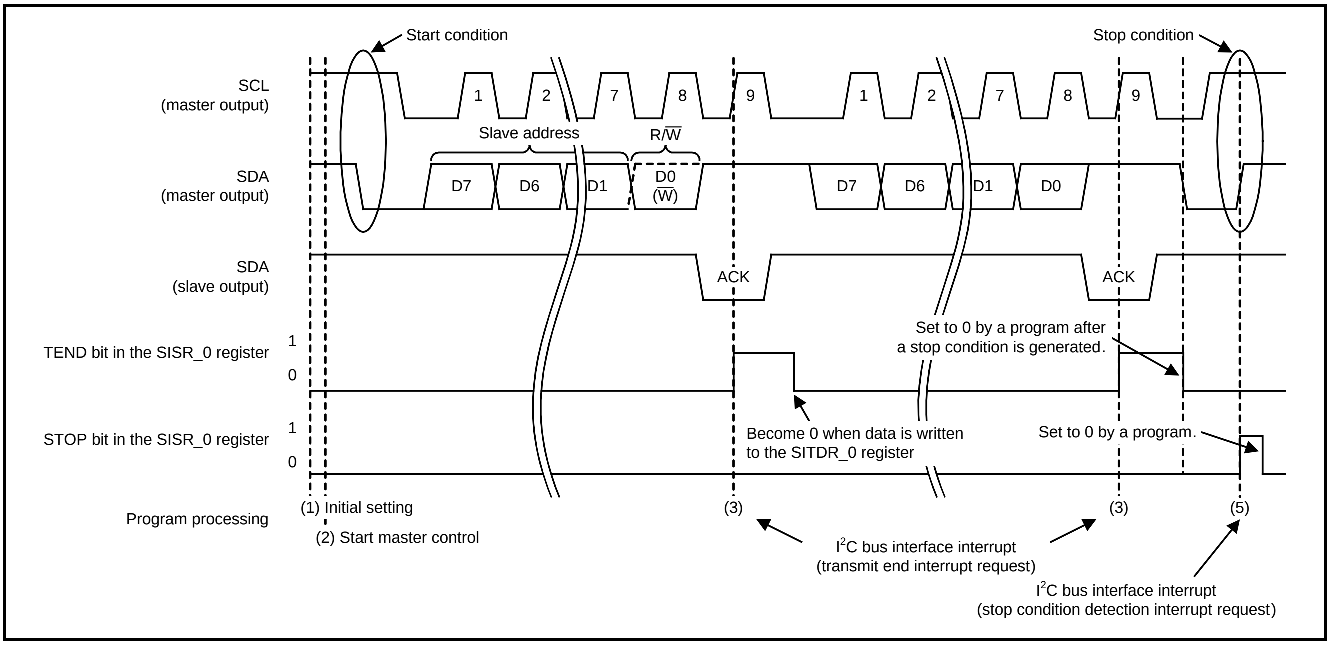


Figure 3.3 Operation Timing in Master Transmit Mode

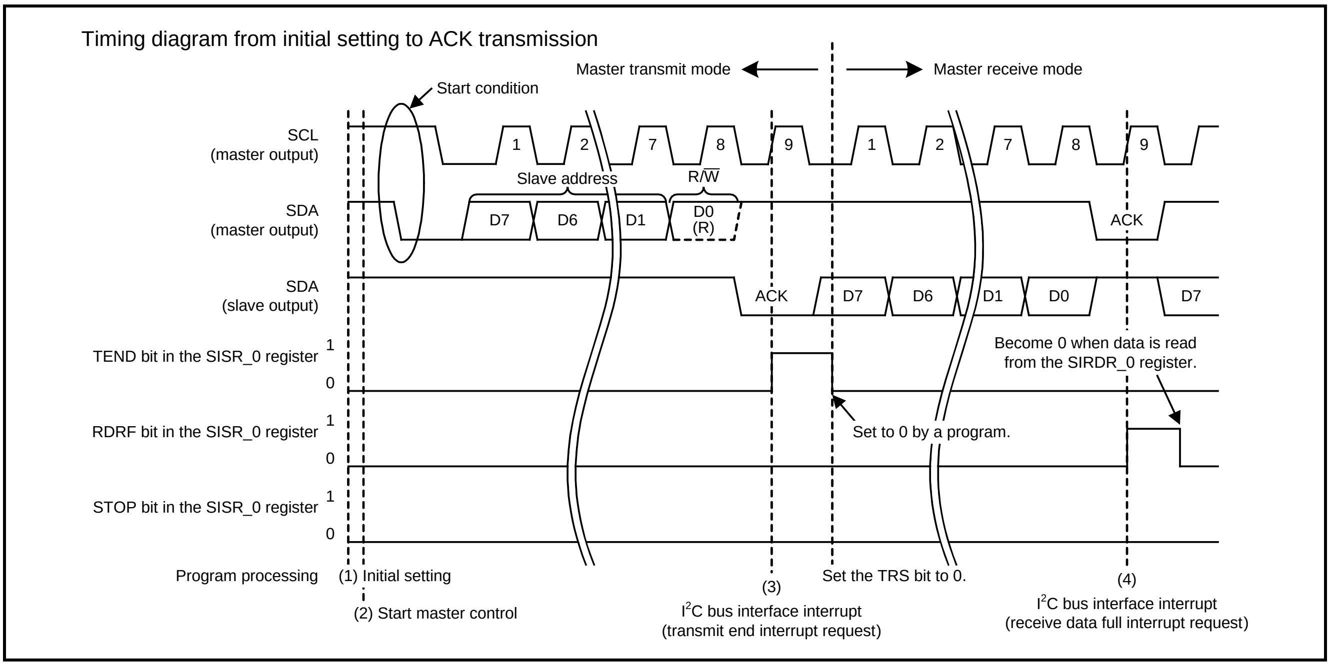


Figure 3.4 Operation Timing in Master Receive Mode (1/2)

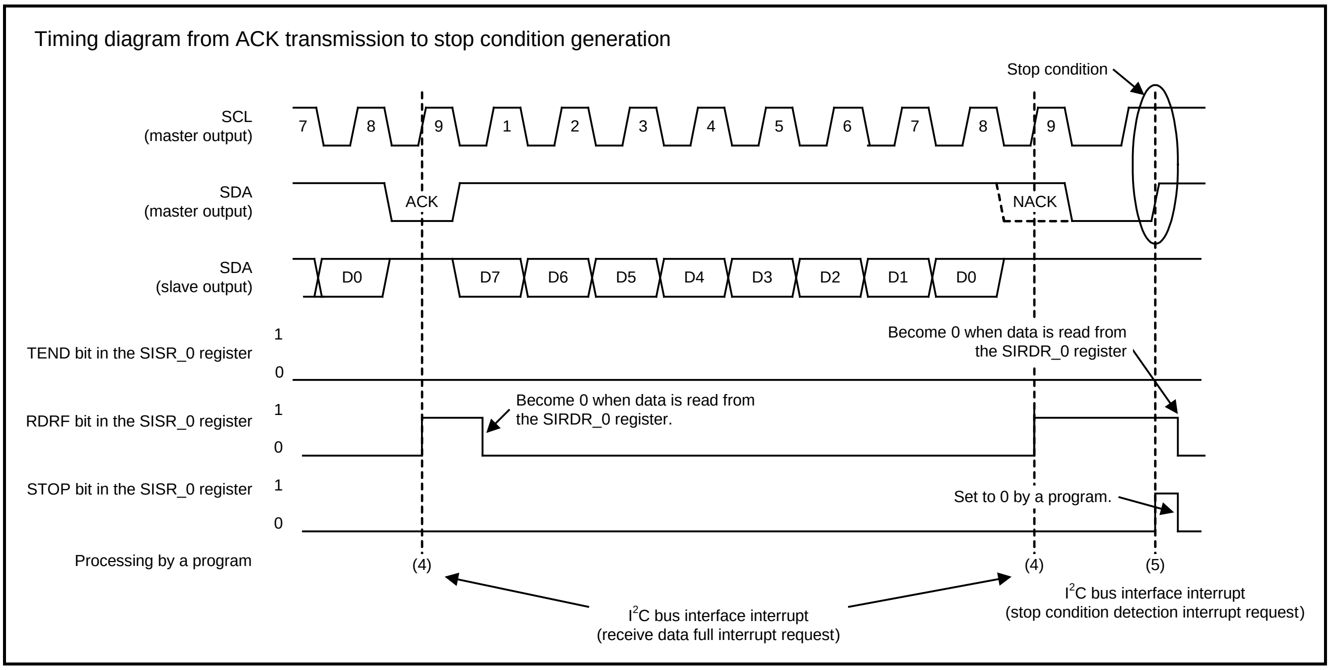


Figure 3.5 Operation Timing in Master Receive Mode (2/2)

3.2 Constants

Table 3.1 lists the Constants Used in the Sample Code.

Table 3.1 Constants Used in the Sample Code

Constant Name	Setting Value	Contents
DEVICE_ADDRESS	0b01010101	Slave address (7 bits from b6 to b0 are used)
READ	1	Master receive
WRITE	0	Master transmit
LENGTH	3	Number of transmit/receive data bytes
BUFSIZE	255	Number of transmit/receive data buffers
PD_IIC	0x2E7	Direction register address
PD_IIC_INIT	0b01011111	Direction register setting values PD3_5/SCL: input PD3_7/SDA: input
IIC_SP_ON	1	Stop condition generated
IIC_SP_OFF	0	Stop condition not generated

3.3 Structure List

Figure 3.6 shows the Structure Used in the Sample Code.

```
typedef union{
        struct{
                unsigned char b0:1;
                unsigned char b1:1;
                unsigned char b2:1;
                unsigned char b3:1;
                unsigned char b4:1;
                unsigned char b5:1;
                unsigned char b6:1;
                unsigned char b7:1;
        }bit;
        unsigned char all;
}byte_dt;

Static byte_dt             iic_str1;         /*-- Device address(b7-b1) + R/W(b0) --*/
#define iic_slave_addr     iic_str1.all      /* Slave Address */
#define iic_rw             iic_str1.bit.b0   /* 0: Write (Master Transmit)      1: Read (Master Receive) */

Static byte_dt             iic_str2;         /* -- Status -- */
#define iic_status         iic_str2.all      /* All status */
#define iic_starT          iic_str2.bit.b0   /* 1: During communication         0: Communication end */
#define iic_err_par        iic_str2.bit.b1   /* 1: Parameter error              0: not error */
#define iic_err_nack       iic_str2.bit.b2   /* 1: NACK detection error         0: not error */
#define iic_err_addr       iic_str2.bit.b3   /* 1: Address not match error      0: not error */

static unsigned char iic_length;             /* Transfer data length */
static unsigned char iic_index;              /* Index of transmit/receive byte number */
static unsigned char far *iic_pointer;       /* Pointer of buffer */
```

Figure 3.6 Structure Used in the Sample Code

3.4 Variables

Table 3.2 lists the Global Variables.

Table 3.2 Global Variables

Type	Variable Name	Contents	Function Used
unsigned char	iic_tx[BUFSIZE]	Transmit buffer	main
unsigned char	iic_rx[BUFSIZE]	Receive buffer	main

3.5 Functions

Table 3.3 lists the Functions.

Table 3.3 Functions

Function Name	Outline
mcu_init	System clock setting
iic_init	I^2C_0 initial setting
iic_master_start	Master control start processing
_ssuic	I^2C bus interface interrupt processing
stp_int	Stop condition detection processing
master_trn_int	Master transmission
master_rcv_int	Master reception
iic_master_end	Master control end processing

3.6 Function Specifications

The following tables list the sample code function specifications.

main	
Outline	Main processing
Header	iic.h
Declaration	void main(void)
Description	Perform the following processes. • Call the initialization functions for the system clock and I^{2}C bus interface. • After the initial setting, master transmission and master reception are repeated alternately. Call the iic_master_start function to start master control and call the iic_master_end function to wait for completion of master control.
Arguments	None
Returned Value	None

mcu_init	
Outline	System clock setting
Header	None
Declaration	void mcu_init(void)
Description	Set the system clock.
Arguments	None
Returned Value	None

iic_init		
Outline	I^{2}C_0 initial setting	
Header	None	
Declaration	void iic_init(unsigned char ini)	
Description	Perform initial setting for I^{2}C_0.	
Arguments	unsigned char ini	0: I^{2}C_0 module disabled 1: I^{2}C_0 module enabled
Returned Value	None	

iic_master_start

Outline	Master control start processing	
Header	None	
Declaration	unsigned char iic_master_start(unsigned char addr, unsigned char rw, unsigned char far *buf, unsigned char len)	
Description	Perform processing to start master control. Prior to executing this function, execute the iic_init function to enable the I^2C_0 module. Perform the following processes: • In the function header, all statuses are initialized and the argument parameters are read. If any parameter value is invalid, the parameter error flag is set to 1 and 0xFF is returned. Master control start processing is not performed when a parameter error is detected. • Next, read the bus status. When the bus is busy, the returned value is 0 and master control start processing is not performed. When the bus is free, the returned value is 1 and master control start processing is performed. After setting the communication-in-process flag to 1, a start condition is generated and a slave address is transmitted.	
Arguments	unsigned char addr	0x00 to 0x7F: Slave address
	unsigned char rw	0x00: Master transmit 0x01: Master receive
	unsigned char far *buf	Transmit/receive buffer pointer
	unsigned char len	0x01 to 0xFF: Transmit data length
Returned Value	unsigned char	0: Bus is busy 1: Bus is free 0xFF: Parameter error

_ssuic

Outline	I^2C bus interface interrupt processing
Header	None
Declaration	void _ssuic(void)
Description	An interrupt is generated at the rising edge of the ninth bit of the SCL clock or when a stop condition is detected. When a stop condition is detected, call the stp_int function. When a stop condition is not detected, call the master_trn_int function for master transmit and call the master_rcv_int function for master receive. To complete communication, generate a stop condition and enable the stop condition detection interrupt request.
Arguments	None
Returned Value	None

stp_int	
Outline	Stop condition detection processing
Header	None
Declaration	static void stp_int(void)
Description	This function is called from I^2C bus interface interrupt processing. The I^2C bus interface associated SFRs changed during communication are reset, and the communication-in-progress flag is set to 0.
Arguments	None
Returned Value	None

master_trn_int		
Outline	Master transmission	
Header	None	
Declaration	static unsigned char master_trn_int(void)	
Description	This function is called from the I^2C bus interface interrupt processing. IIC_SP_OFF is returned in the following case: • ACK is detected and not the last byte (starts next transmission). IIC_SP_ON is returned in the following case: • NACK is detected (NACK detection error flag is set to 1). • The last byte transmission is completed.	
Arguments	None	
Returned Value	unsigned char	IIC_SP_ON(0) : Stop condition generated IIC_SP_OFF(1) : Stop condition not generated

master_rcv_int		
Outline	Master reception	
Header	None	
Declaration	static unsigned char master_rcv_int(void)	
Description	This function is called from I^2C bus interface interrupt processing. After transmitting the first byte (slave address), set to master receive mode and enable the receive data full interrupt request. IIC_SP_OFF is returned in the following case: • The following data is not the last byte of data. IIC_SP_ON is returned in the following case: • NACK is detected (NACK detection error flag is set to 1). • The last byte has been received.	
Arguments	None	
Returned Value	unsigned char	IIC_SP_ON(0) : Stop condition generated IIC_SP_OFF(1) : Stop condition not generated

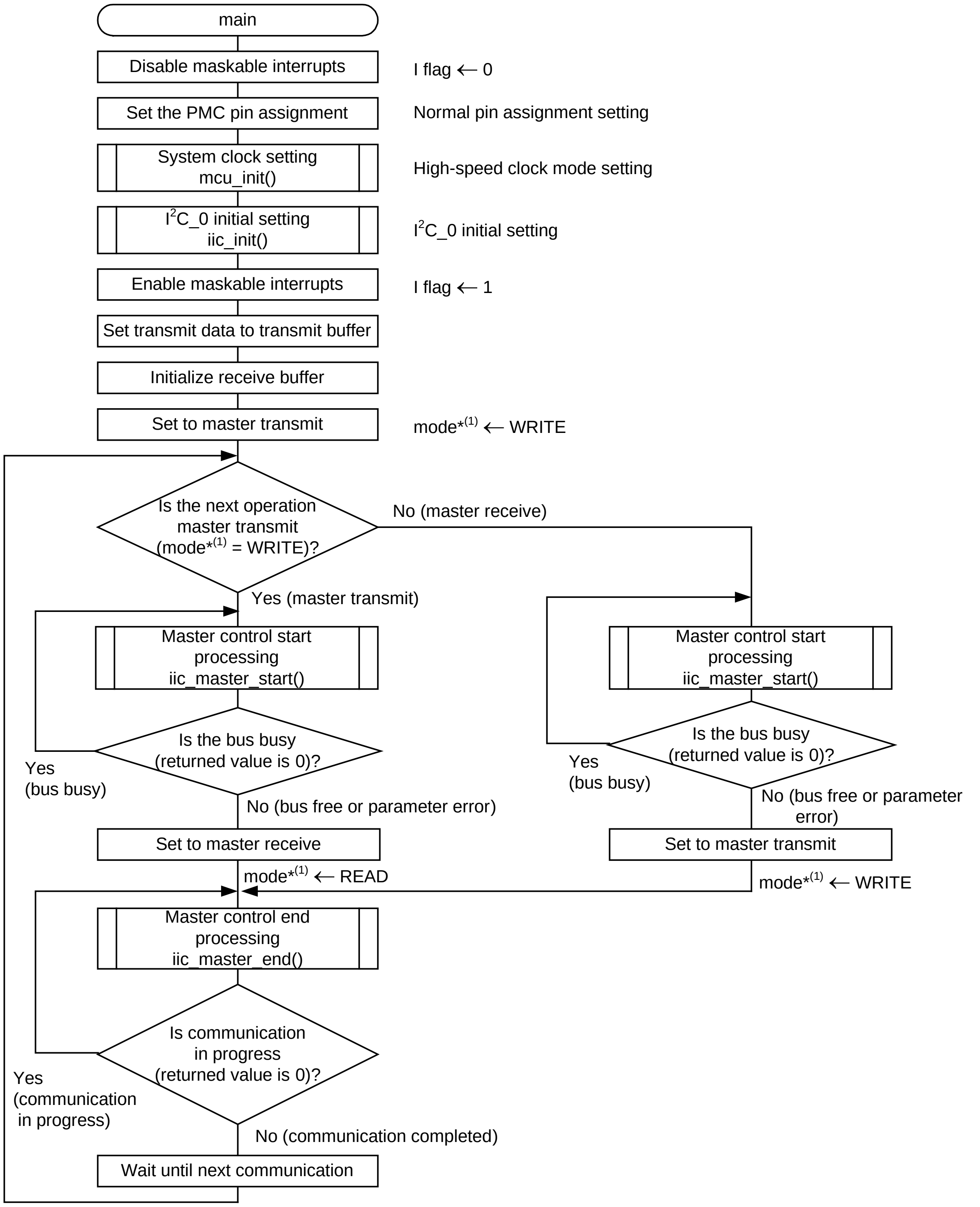
iic_master_end

Outline	Master control end processing	
Header	None	
Declaration	unsigned char iic_master_end(void)	
Description	This function is called from the main function. It informs the user of the master control state. During communication, this function returns 0. When communication is completed, this function returns 1. Add processing for communication completion as needed.	
Arguments	None	
Returned Value	unsigned char	0: Communication in progress 1: Communication completed

3.7 Flowcharts

3.7.1 Main Processing

Figure 3.7 shows the Main Processing.



Note: mode is a local variable used in the main function.

Figure 3.7 Main Processing

3.7.2 System Clock Setting

Figure 3.8 shows the System Clock Setting.

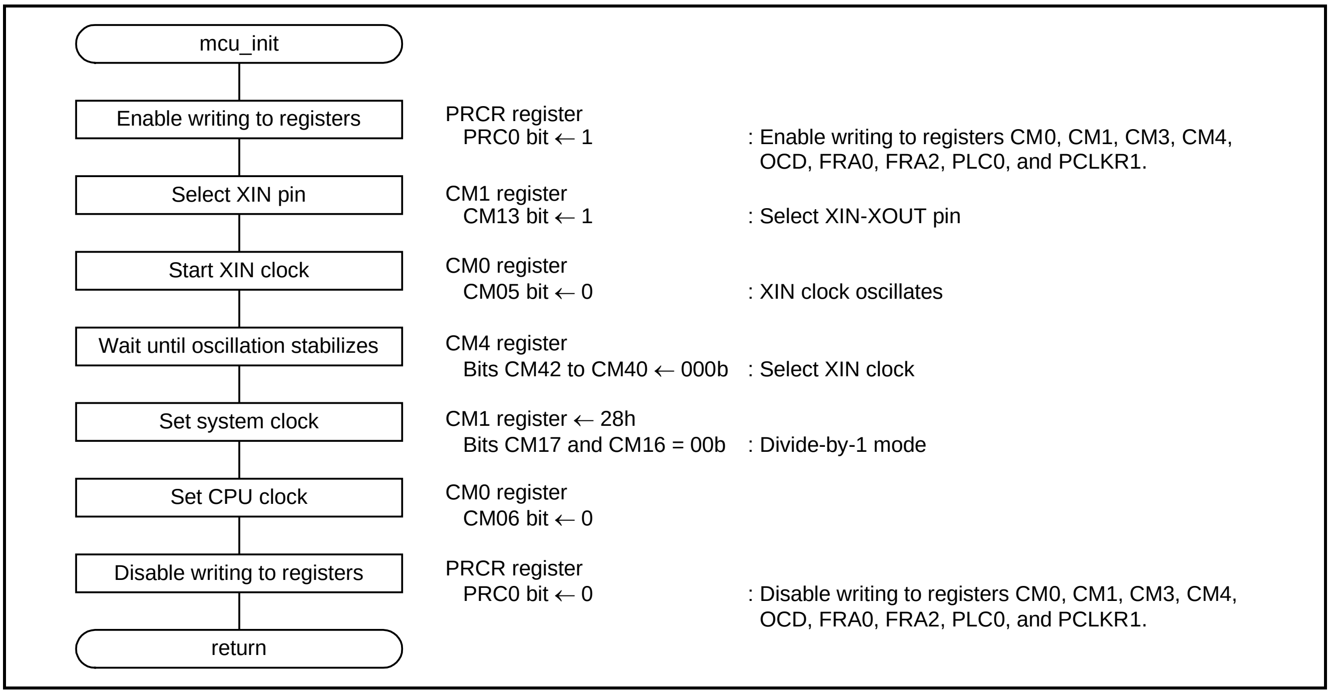


Figure 3.8 System Clock Setting

3.7.3 I²C_0 Initial Setting

Figure 3.9 and Figure 3.10 show the I²C_0 Initial Setting.

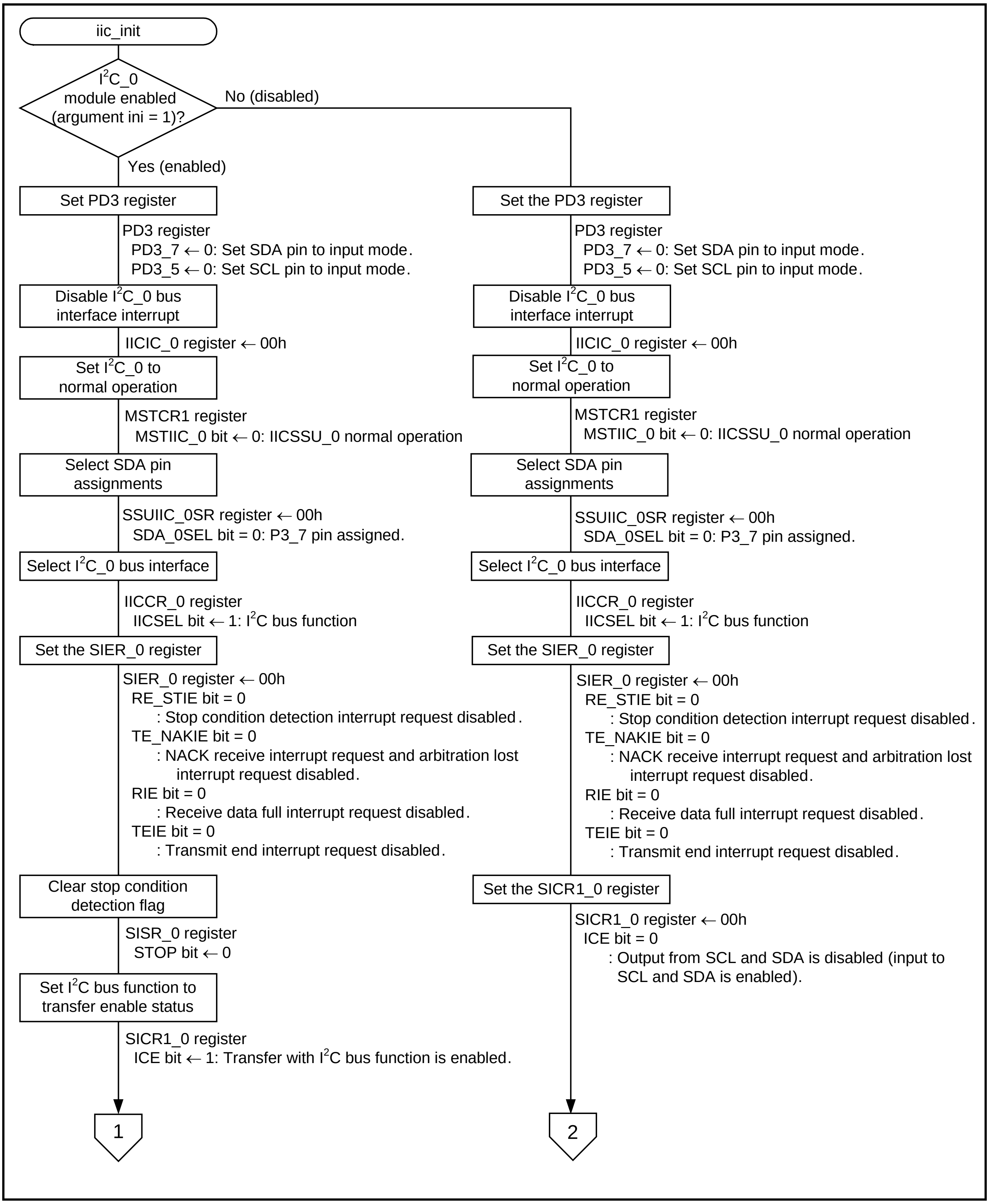


Figure 3.9 I²C_0 Initial Setting (1/2)

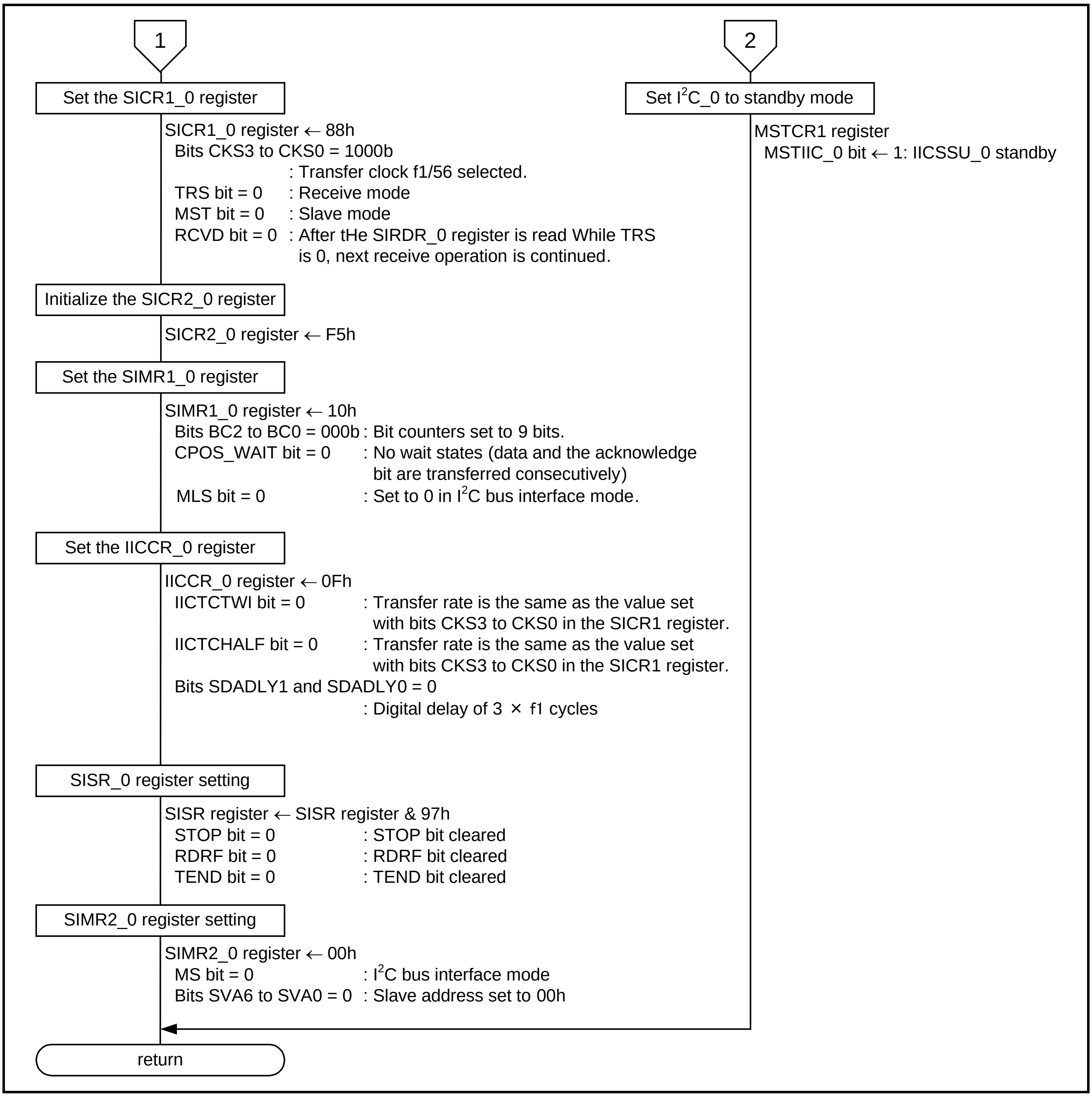


Figure 3.10 I^{2}C_0 Initial Setting (2/2)

3.7.4 Master Control Start Processing

Figure 3.11 and Figure 3.12 show the Master Control Start Processing.

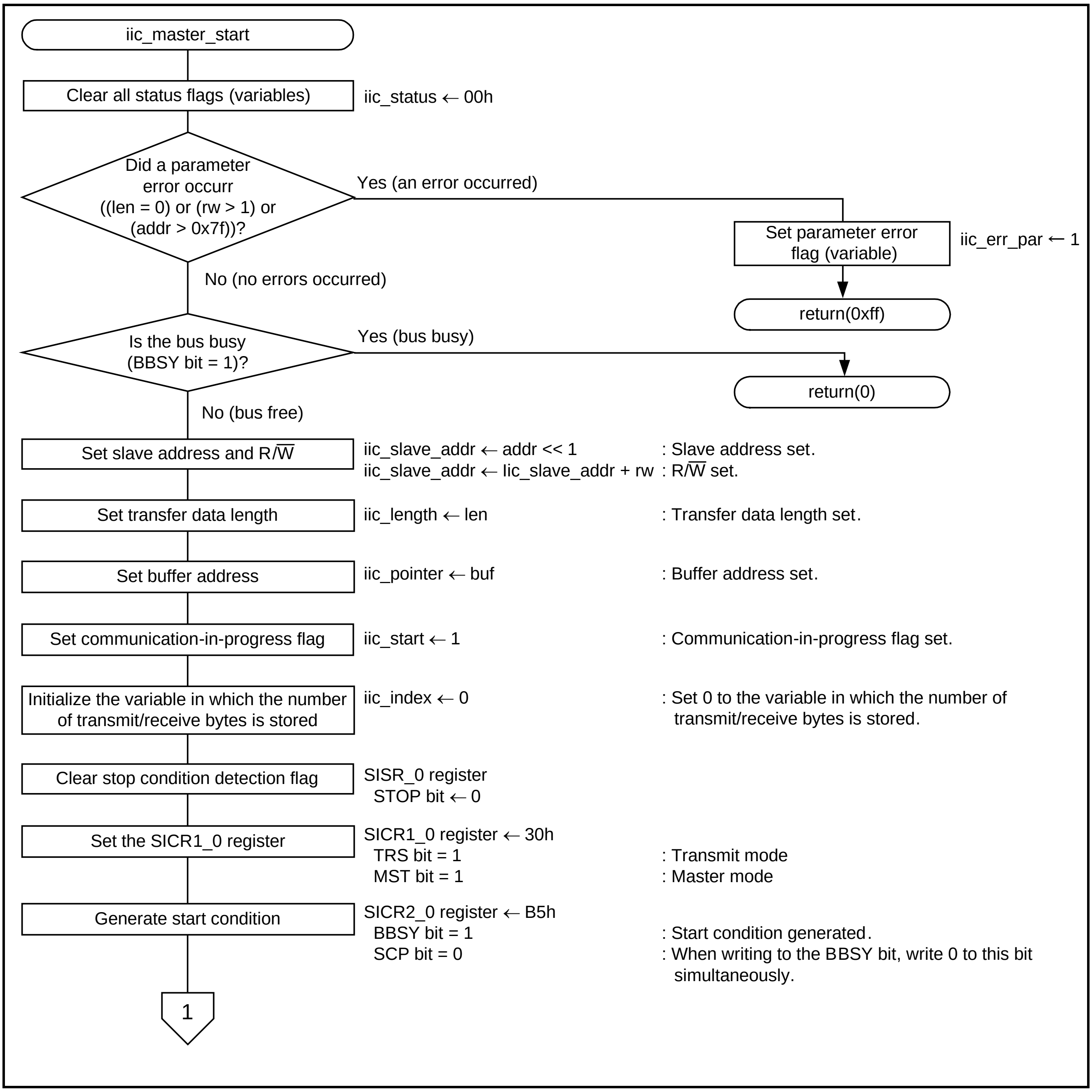


Figure 3.11 Master Control Start Processing (1/2)

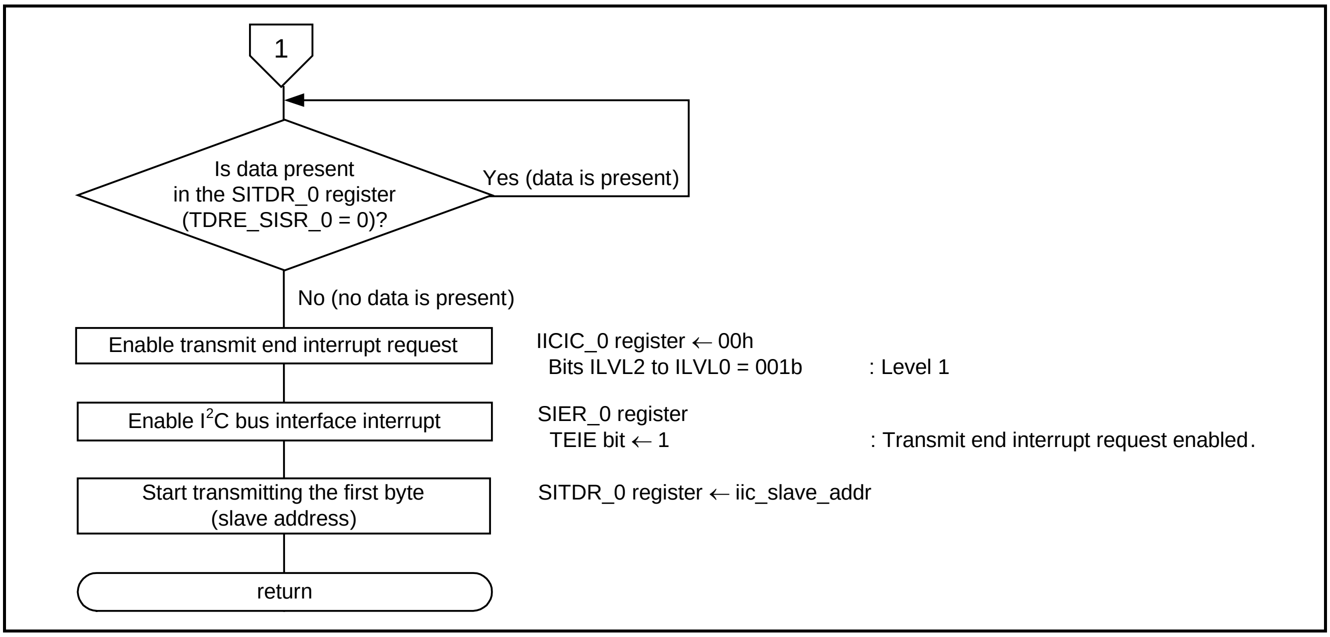


Figure 3.12 Master Control Start Processing (2/2)

3.7.5 I²C bus Interface Interrupt Processing

Figure 3.13 shows the I²C bus Interface Interrupt Processing.

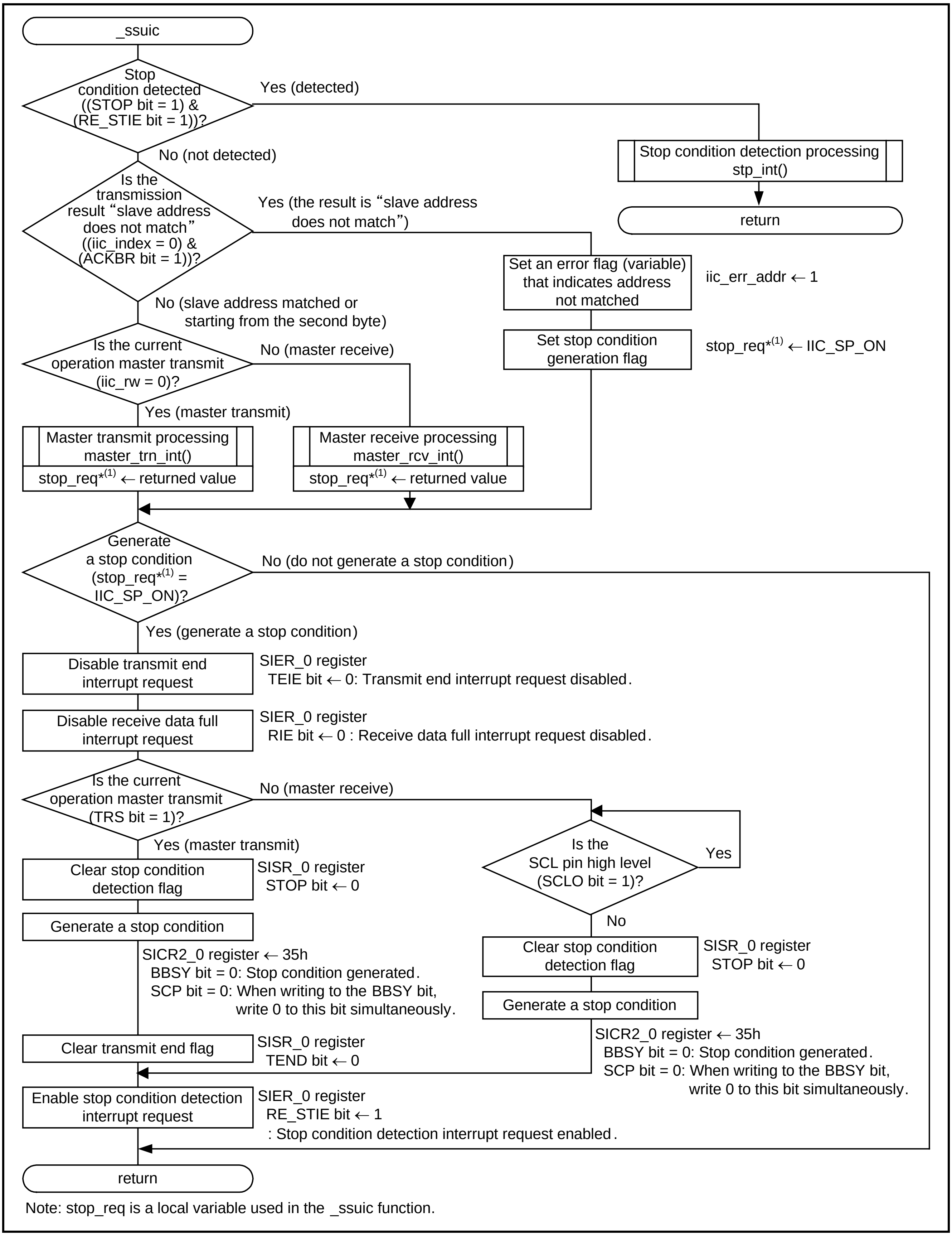


Figure 3.13 I²C bus Interface Interrupt Processing

3.7.6 Stop Condition Detection

Figure 3.14 shows the Stop Condition Detection.

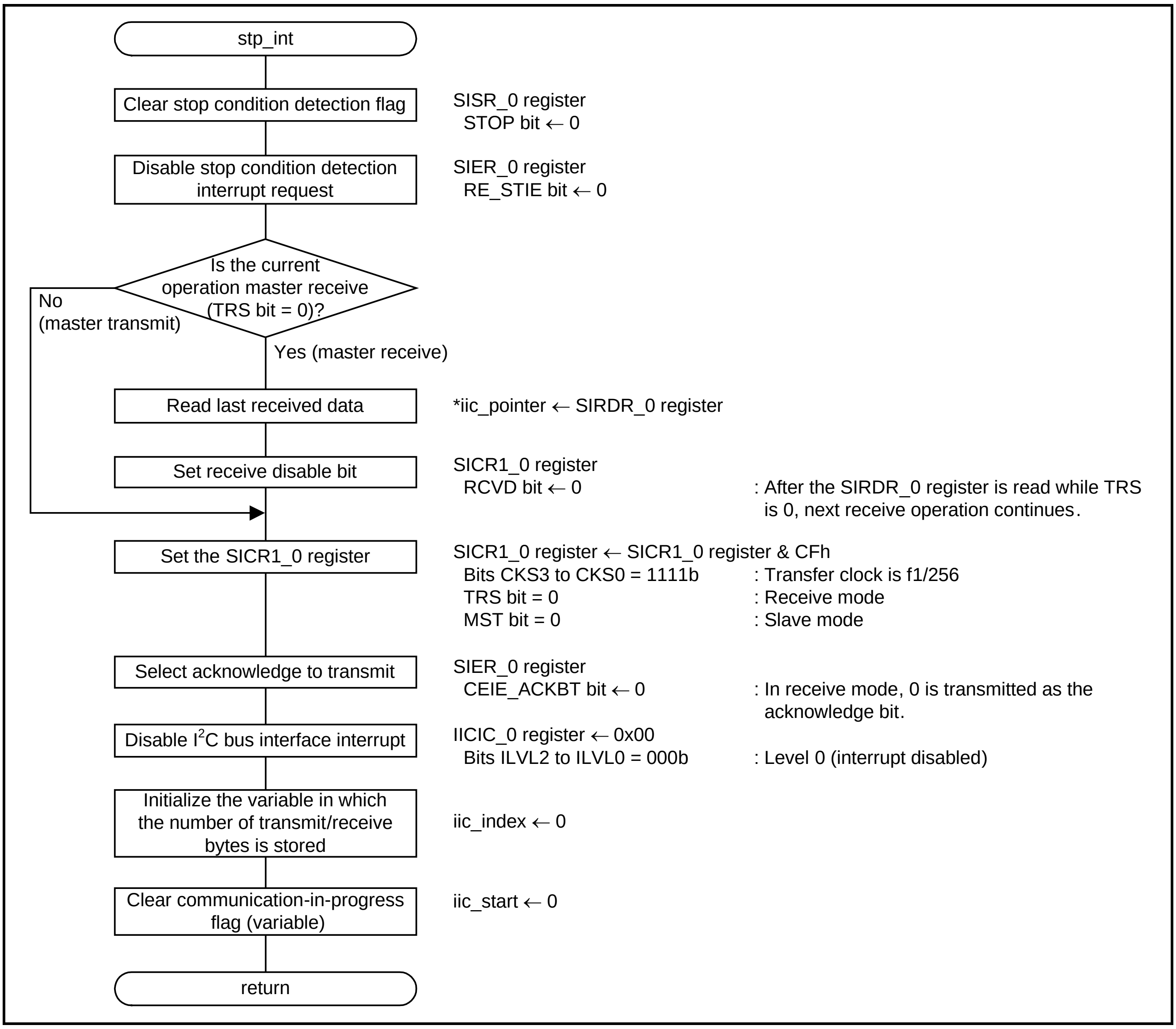


Figure 3.14 Stop Condition Detection

3.7.7 Master Transmission

Figure 3.15 shows the Master Transmission.

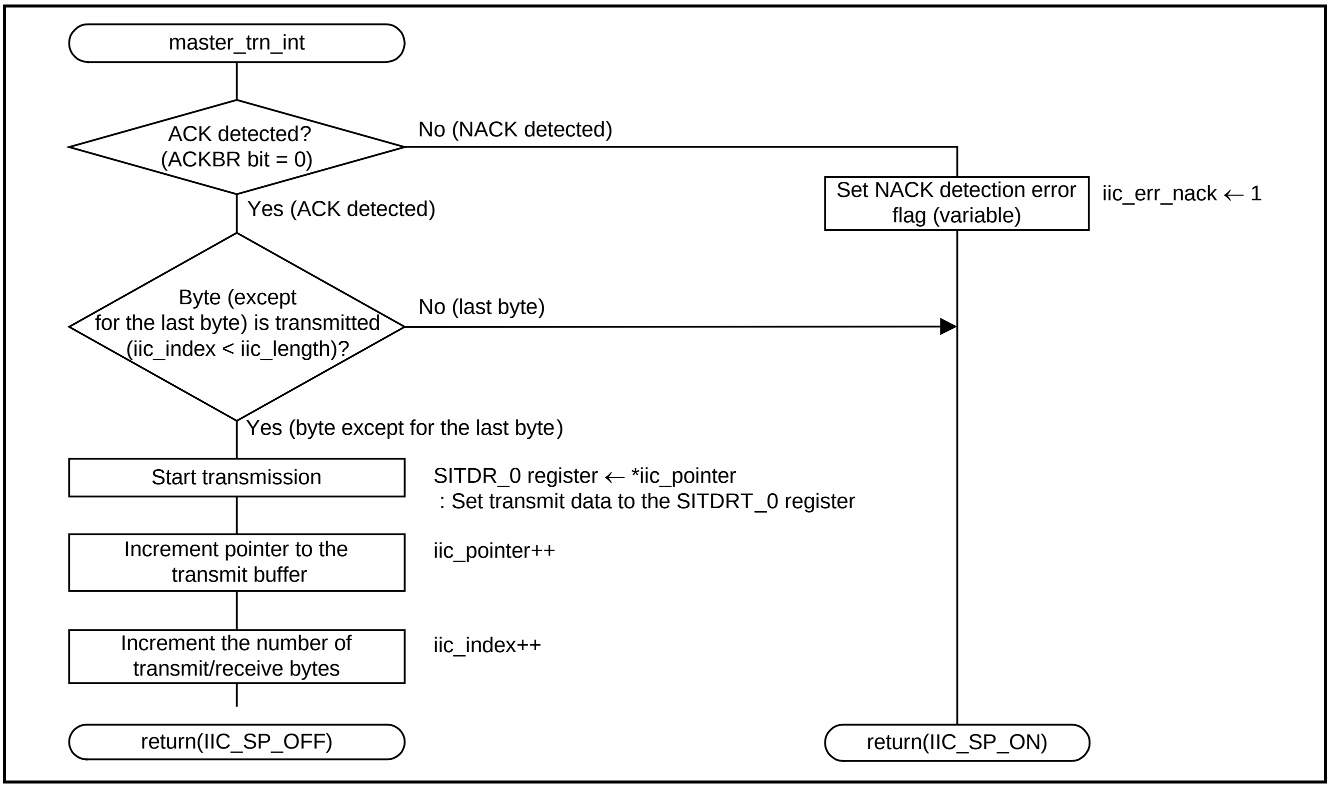
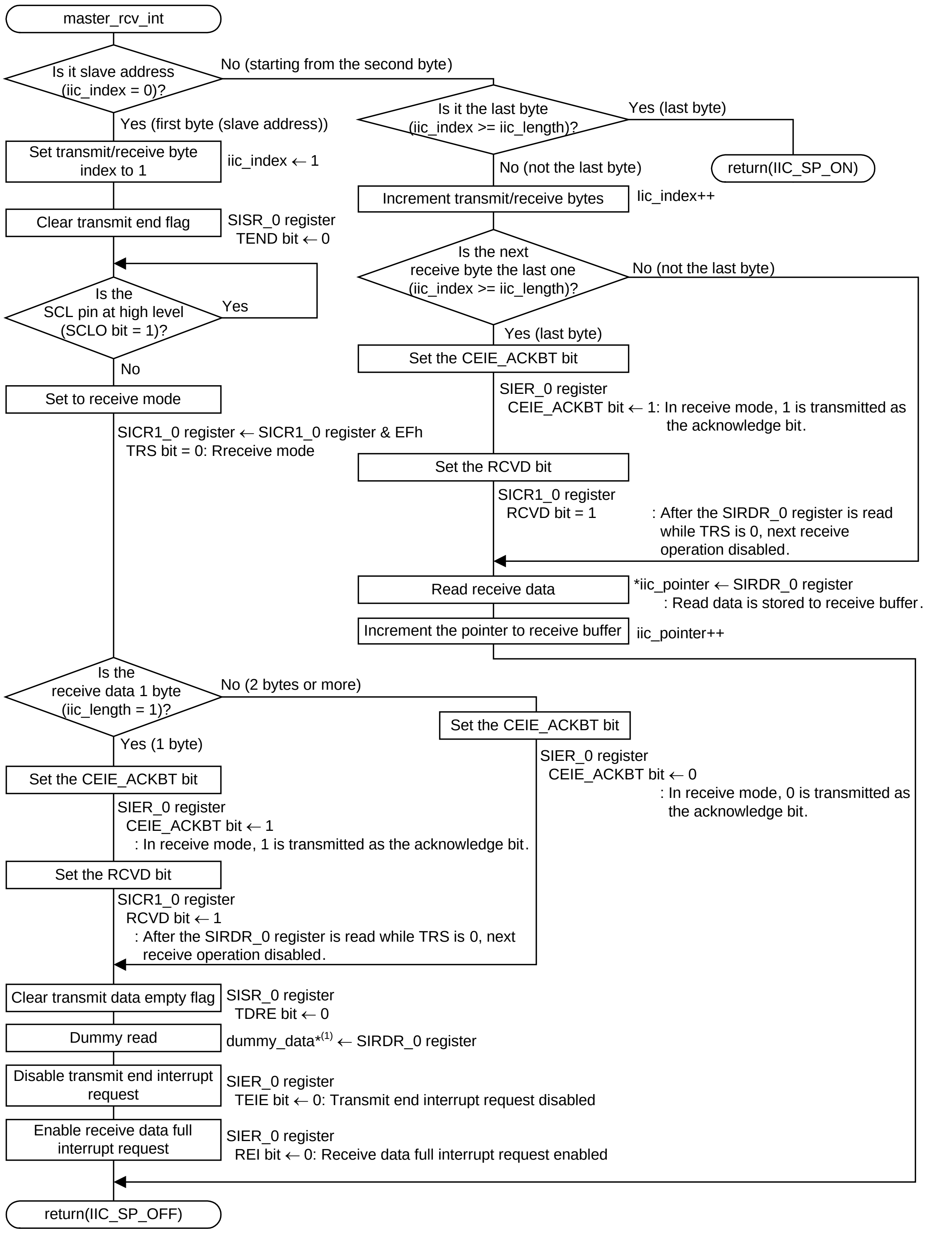


Figure 3.15 Master Transmission

3.7.8 Master Reception

Figure 3.16 shows the Master Reception.

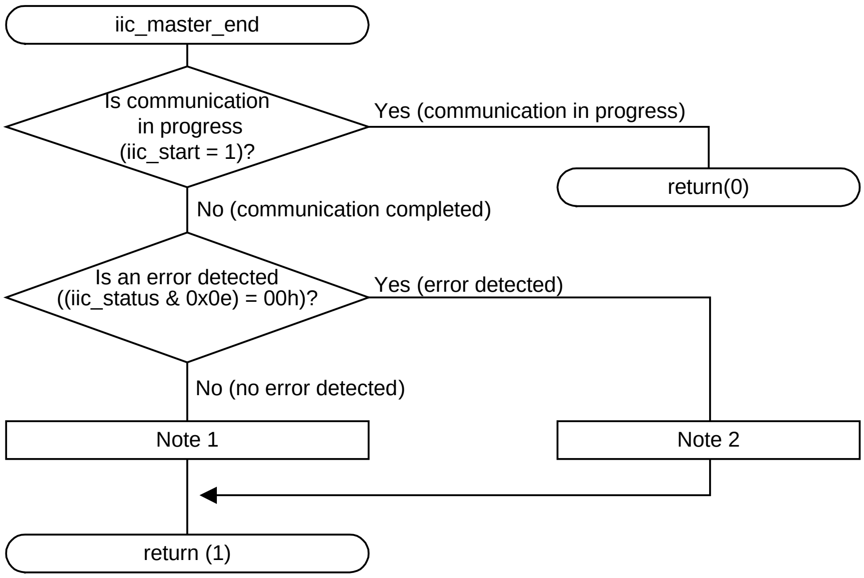


Note: dummy_data is a local variable used in the master_rcv_int function.

Figure 3.16 Master Reception

3.7.9 Master Control End Processing

Figure 3.17 shows the Master Control End Processing.



Notes: 1. Add processing for communication normal completion as needed.
2. Add processing for communication error completion as needed.

Figure 3.17 Master Control End Processing

4. Sample Code

Sample code can be downloaded from the Renesas Electronics website.

5. Reference Documents

User's Manual: Hardware

R8C/56E Group User's Manual: Hardware Rev.2.00

The latest version can be downloaded from the Renesas Electronics website.

Technical Update/Technical News

The latest information can be downloaded from the Renesas Electronics website.

Website and Support

Renesas Electronics website

http://www.renesas.com

Inquiries

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

REVISION HISTORY	R8C/56E Group Application Note I^2C bus Single Master Control Program (Master Transmit/Receive)

Rev.	Date	Description	
		Page	Summary
1.00	**Nov 30, 2012**	—	First edition issued

General Precautions in the Handling of MPU/MCU Products

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

1. Handling of Unused Pins

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

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

2. Processing at Power-on

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

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

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

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

3. Prohibition of Access to Reserved Addresses

   Access to reserved addresses is prohibited.

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

4. Clock Signals

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

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

5. Differences between Products

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

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

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: +44-1628-651-700, Fax: +44-1628-651-804

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.
80 Bendemeer Road, Unit #06-02 Hyflux Innovation Centre Singapore 339949
Tel: +65-6213-0200, Fax: +65-6213-0300

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 2.2