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.

H8SX Family

Multi-Master Mode Communications Using I²C Bus Interface 2 (IIC2)

Introduction

This application note describes the usage of the I²C bus interface 2 (IIC2) module in the multi-master mode.

Target Device

H8SX/1663

Contents

1. Specification

- Figure 1 shows the connections for communications using the I²C bus interface 2 in multi-master mode. The slave addresses and settings for the SAR_0 registers of the individual devices are listed in table 1.
- The multi-master system in this sample task consists of two master devices and one slave device.
- The I²C bus transfer rate is 93.75 kbits/s (kHz).
- When communications from master 1 and master 2 are attempted simultaneously, the master that loses the arbitration will stop processing.

The following describes the procedures for the operation of this sample task.

1. The I²C bus interface multi-master transfer starts on the input of the low trigger to the $\overline{\text{IRQ0}}$ pin of the master side.
2. The master side transmits 128 bytes of data, which have been prepared in the on-chip ROM in advance, to the on-chip RAM on the slave side.
3. The slave device returns the 128 bytes of data received in step 2 from its on-chip RAM to the on-chip RAM on the master side.
4. The master side compares the received data in its on-chip RAM with the data transmitted from its on-chip ROM, and confirms whether the two match.
5. Based on the results of this comparison and the state of arbitration lost, the master side outputs levels on the P32 to P30 pins that indicate the result of operation.
6. From the value of the first byte of received data on the slave side, the slave judges whether the partner in communications is master 1 or master 2, and outputs levels on pins P35 and P34 that indicate the state of operation.

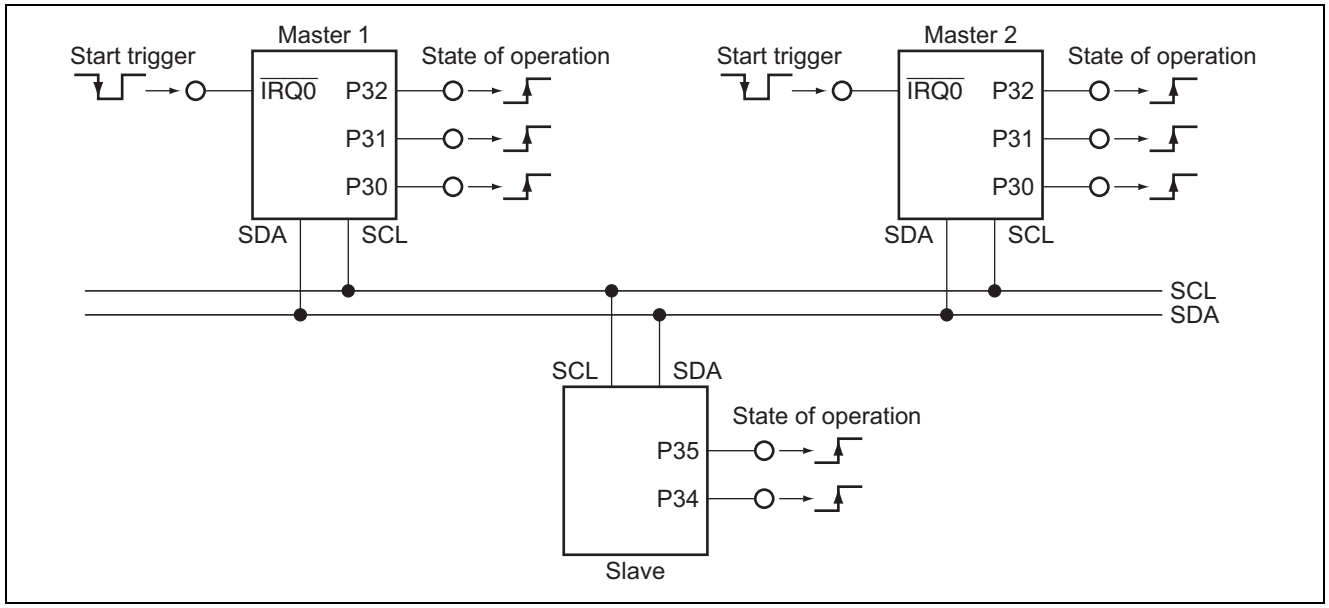


Figure 1 Connections for I²C Bus Interface 2 Multi-Master Mode Communication

Table 1 Slave Addresses

Device	Slave Address	SAR_0 Setting
Master 1	1	H'02
Master 2	2	H'04
Slave	3	H'06

2. Applicable Conditions

Table 2 Applicable Conditions

Items	Description	
Operating frequency	Input clock	: 12 MHz
	System clock (Iϕ)	: 24 MHz (input clock frequency × 2)
	Peripheral mode clock (Pϕ)	: 24 MHz (input clock frequency × 2)
	External bus clock (Bϕ)	: 24 MHz (input clock frequency × 2)
Mode of operation	Mode 7 (MD2 = 1, MD1 = 1, MD0 = 1)	

3. Description of Functions Used

3.1 Description of I²C Bus Interface 2 (IIC2)

An I²C bus interface 2 is used in multi-master operation to demonstrate bi-directional communications between in master mode and slave mode.

3.2 Watchdog Timer (WDT)

To make the I²C bus interface escape from hung states, the watchdog timer is used in the interval timer mode. Once the specified interval has elapsed, a WDT interrupt is generated, and error recovery processing for the I²C bus interface proceeds.

3.3 Master Side $\overline{\text{IRQ0}}$ Pin

The trigger to start master transmission and master reception is input to the $\overline{\text{IRQ0}}$ pin on the master side. $\overline{\text{IRQ0}}$ starts the processing of the I²C bus interface communications on the input of a falling edge on the $\overline{\text{IRQ0}}$ pin.

The master judges whether or not the $\overline{\text{IRQ0}}$ pin has received the start trigger by polling the IRQ status flag. The IRQ interrupt is not used.

3.4 Master Side P32 to P30 Pins

As indicated in table 3, the pins P32 to P30 on the master side indicate the state of I²C bus interface communications (reset state or result of operations).

Table 3 Output Values of Master Side Pins and State of Operations

P32	P31	P30	State of Operation
0	0	0	Reset
x	0	1	Data match
x	1	0	Data mismatch
1	x	x	Arbitration lost generated

3.5 Slave Side P35 and P34 Pins

As indicated in table 4, pins P35 and P34 on the slave side indicate the state of I²C bus interface communications (reset state or result of operations).

Table 4 Output Values of Slave Side Pins and State of Operations

P35	P34	State of Operation
0	0	Reset
0	1	Master 1 (The first byte of received data is H'81.)
1	0	Master 2 (The first byte of received data is H'82.)
1	1	Error (The first byte of received data is neither H'81 nor H'82.)

4. Principles of Operation

4.1 Timing of Operations in Master Transmit Mode

Figure 2 shows the timing of operations of the I²C bus interface 2 in master transmit mode. Table 5 describes processing by hardware and software at the numbered points in figure 2.

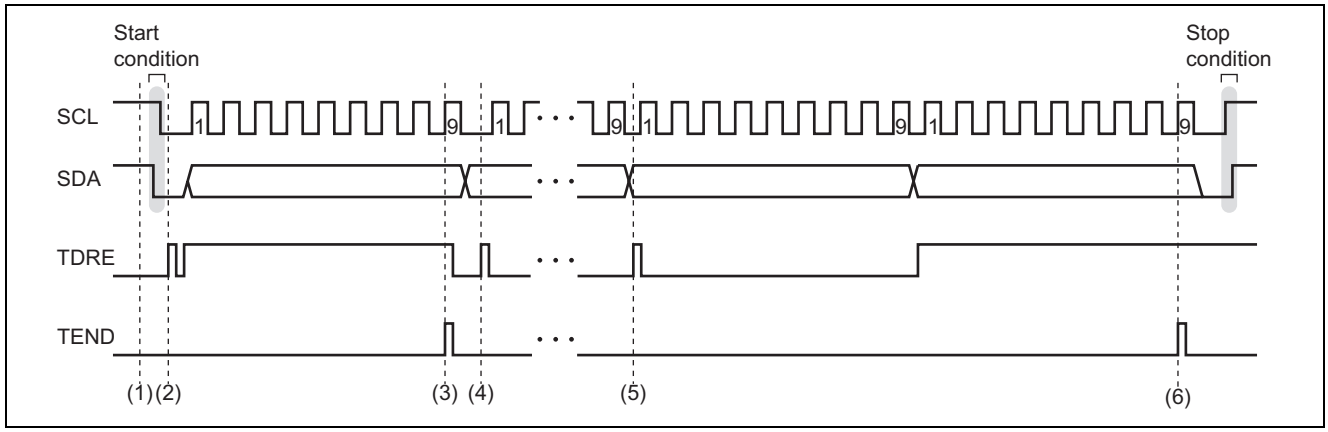


Figure 2 Timing of Operations in Master Transmit Mode

Table 5 Description of Processing

	Hardware Processing	Software Processing
(1)	No processing	a. Set the TIE bit to 1, enabling the data empty interrupt. Set the TDRE bit to 1 for interrupt generation. b. Issue the start condition.
(2)	a. Generation of transmit-data empty interrupt The start condition is detected and the TDRE is set to 1.	a. Write the slave-side address and data-direction bit (R/$\overline{W}$) to ICDRT, then transmit this data. Writing to ICDRT clears the TDRE and TEND flags. b. Set the TIE bit to 0, disabling the transmit data empty interrupts. c. Set the TEIE to 1, enabling the transmit-end interrupt. When the TEND bit is set to 1, the interrupt will be generated.
(3)	a. Generation of transmit-end interrupt On the rising edge of the ninth cycle of SCL, the TEND bit is set to 1.	a. Write the data for transmission to ICDRT and transmit the data. Writing to ICDRT clears the TDRE and TEND flags. b. Set the TIE bit to 1, enabling the data-empty interrupt. When the TDRE bit is set to 1, the interrupt will be generated. c. Set the TEIE bit to 0, disabling the transmit-end interrupt.
(4)	a. Generation of transmit-data empty interrupt Data are transferred from ICDRT to ICDRS, and ICDRT becomes empty. Then, the TDRE bit is set to 1.	a. Write the data for transmission to ICDRT, then transmit the data. Writing to ICDRT clears the TDRE and TEND flags.

	Hardware Processing	Software Processing
(5)	a. Generation of transmit-data empty interrupt Data are transferred from ICDRT to ICDRS, and ICDRT becomes empty. Then, the TDRE bit is set to 1.	a. Write the last transmit data to ICDRT, then transmit the data. Writing to ICDRT clears the TDRE and TEND flags. b. Set the TIE bit to 0, disabling the transmit-data empty interrupt. c. Set the TEIE bit to 1, enabling the transmit-end interrupt. When the TEND bit is set to 1, the interrupt will be generated.
(6)	a. Generation of transmit-end interrupt On the rising edge of the ninth cycle of SCL, the TEND bit is set to 1.	a. Clear the TEND flag. b. Confirm that the SCL signal is at the low level and then issue the stop condition.

4.2 Timing of Operations in Master Receive Mode

Figures 3 and 4 show the timing of operations of the I²C bus interface 2 in master receive mode. Tables 6 and 7 describe processing by hardware and software at the numbered points in figures 3 and 4.

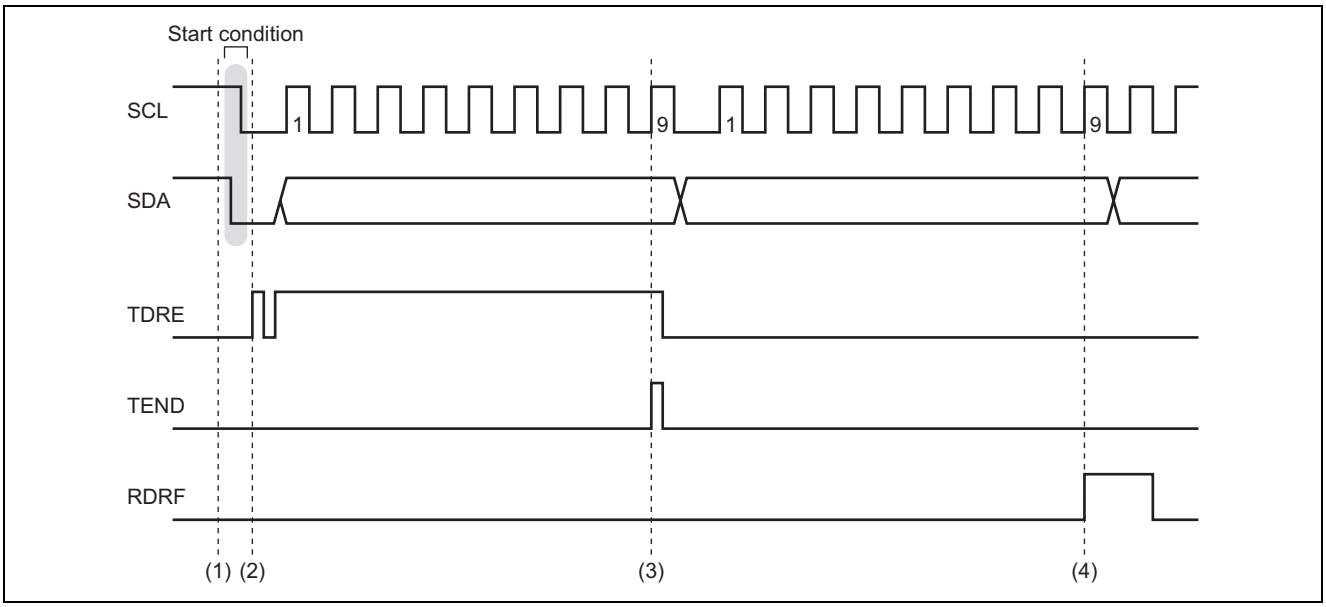


Figure 3 Timing of Operations in Master Receive Mode 1

Table 6 Description of Processing

	Hardware Processing	Software Processing
(1)	No processing	a. Set the TIE bit to 1, enabling the data empty interrupts. When the TDRE bit is set to 1, an interrupt is generated. b. Issue the start condition.
(2)	a. Generation of transmit-data empty interrupt Start condition is detected and the TDRE is set to 1.	a. Write the slave-side address and data-direction bit (R/$\overline{W}$) to ICDRT, then transmit this data. Writing to ICDRT clears the TDRE and TEND flags. b. Set the TIE bit to 0, disabling the transmit-data empty interrupt. c. Set the TEIE bit to 1, enabling the transmit-end interrupt. When the TEND bit is set to 1, the interrupt will be generated.
(3)	a. Generation of transmit-end interrupt On the rising edge of the ninth cycle of SCL, the TEND bit is set to 1.	a. Set the TEIE bit to 0, disabling the transmit-end interrupt. b. Set the RIE bit to 1, enabling the receive-data full interrupt. c. Set the RCVD bit to 1, disabling the next receive operation. d. Clear the TEND flag. e. Set the TRS bit to 0, selecting receive mode. f. Clear the TDRE flag. g. Set the ACKBT bit to 0, so that 0 is output at the timing of acknowledgement output. h. Execute a dummy read of ICDRR. Reading from ICDRR clears the RDRF bit.
(4)	a. Generation of receive-data interrupt Receiving frame data is completed. On the rising edge of the ninth cycle of SCL, the RDRF bit is set to 1.	a. First byte of data for reception is read from ICDRR and saved in RAM. Reading from ICDRR clears the RDRF bit.

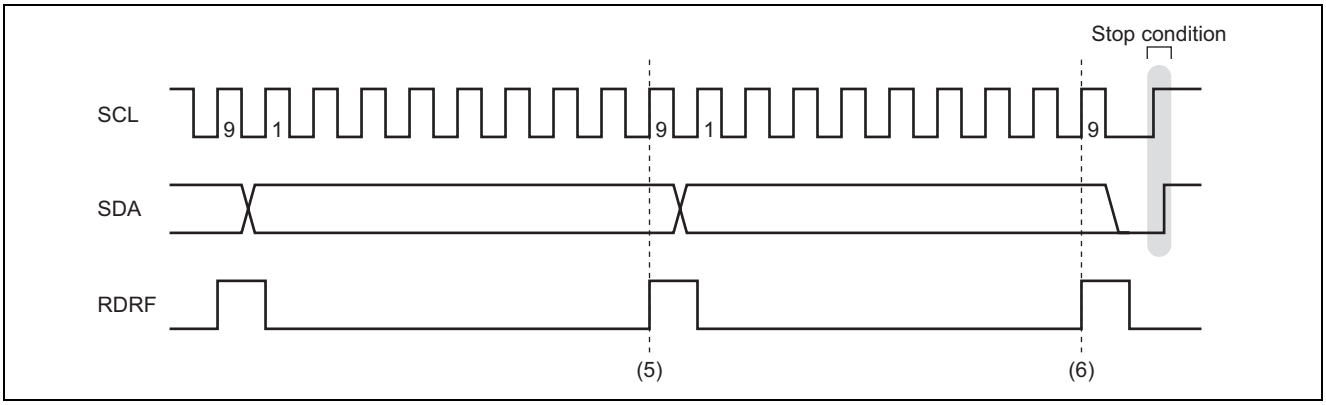


Figure 4 Timing of Operations in Master Receive Mode 2

Table 7 Description of Processing

	Hardware Processing	Software Processing
(5)	a. Generation of receive-data interrupt Receiving frame data is completed. On the rising edge of the ninth cycle of SCL, the RDRF bit is set to 1.	a. Set the ACKBT bit to 1, so that 1 is output at the timing of acknowledge output. b. Data for reception is read from ICDRR and saved in RAM. Reading from ICDRR clears the RDRF bit.
(6)	a. Generation of receive-data interrupt Receiving frame data is completed. On the rising edge of the ninth cycle of SCL, the RDRF bit is set to 1.	a. Last byte of data for reception is read from ICDRR and saved in RAM. Reading from ICDRR clears the RDRF bit. b. Set the RIE bit to 0, disabling the receive-data full interrupt. c. Confirm that the SCL signal is at the low level and then issue the stop condition.

4.3 Description of Bus Arbitration Operation

The I²C bus interface 2 in this LSI performs bus arbitration as illustrated in figures 5 and 6. Loss of arbitration by the LSI is detected in the following two cases.

- Loss of bus arbitration when the start condition is detected
  When the interface is in master mode, bus arbitration is lost if the SDA pin is at the high level when the start condition is detected.
- Loss of bus arbitration during data transmission
  When the interface is in master transmit mode, bus arbitration is lost in the case of a mismatch between the internal SDA signal and the level of the SDA pin on a rising edge of SCL. Each master device monitors the bus line on rising edges of SCL. When the master detects that the level of its internal SDA signal does not match the bus line's SCA level, it turns off its data-output stage.

4.3.1 Loss of Bus Arbitration when the Start Condition is Detected

Figure 5 illustrates an example of the loss of bus arbitration when the start condition is detected. The start condition from master 1 is output after that from master 2.

When the start condition is output from master 2, i.e. the level on the SDA0 pin of master 2 becomes low, the level on the SDA bus line also becomes low. In this case, the signal from master 2 and the signal on the bus line match, so master 2 takes the possession of bus.

When output of the start condition to the bus line sets the SDA signal to the low level, the SDA0 pin of master 1 will still be at the high level and thus will not output a start condition. That is, since the SDA of master 1 and the SDA of the bus line do not match, master 1 loses arbitration.

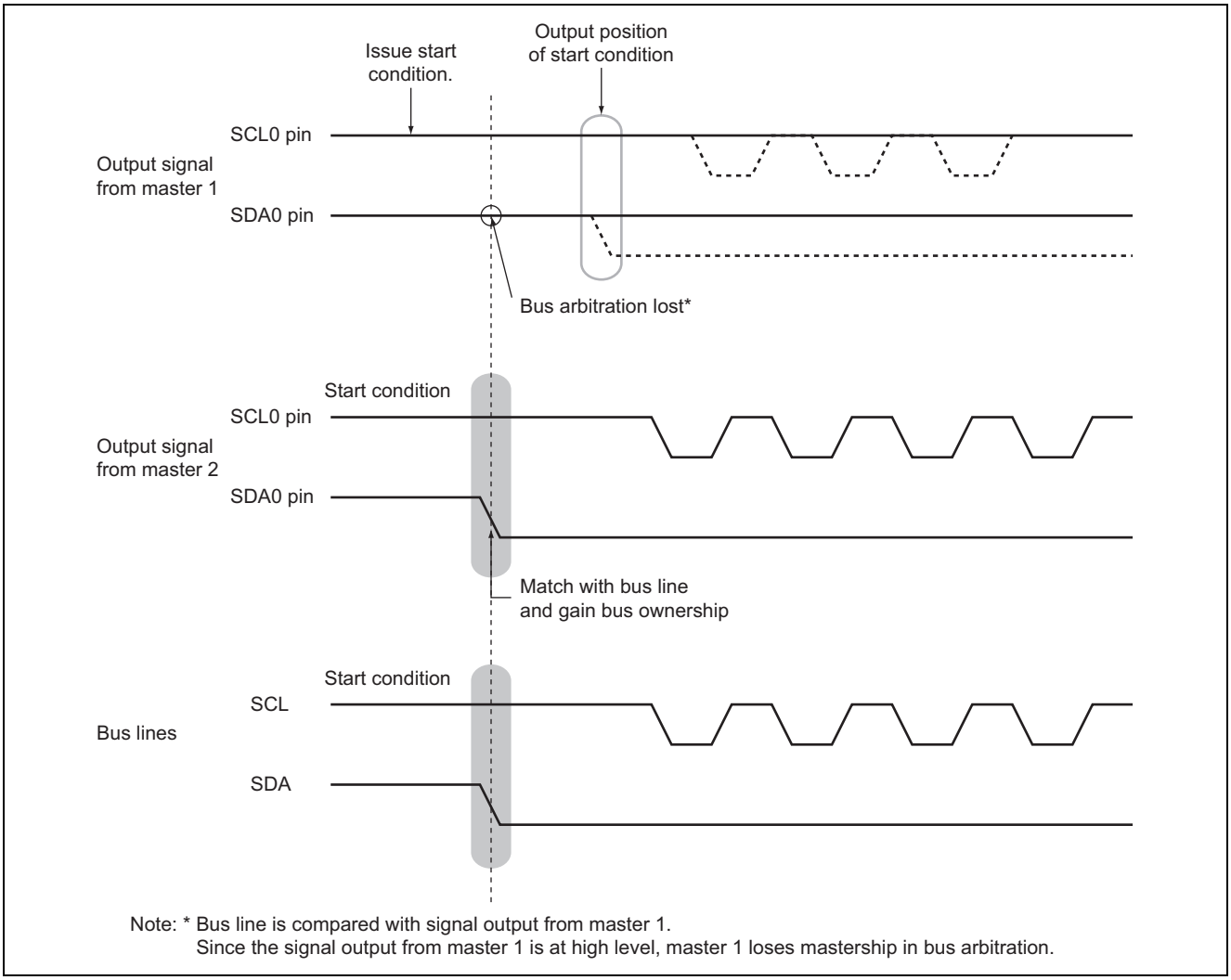


Figure 5 Loss of Bus Arbitration when the Start Condition is Detected

4.3.2 Bus Arbitration Lost when Data is in Transmission

When master 1 and master 2 start transmitting data simultaneously, the data are compared. When a collision is thus detected, master 1 gains bus mastership because it holds the data line (SDA) at the low level (by transmitting H'03) for longer than master 2 (which transmits H'05). As a result, master 1 gains bus mastership.

In this case, master 2 has lost in bus arbitration and automatically enters the idle mode. In order to use master 2 in master transmit mode, master 2 needs to be set again, and the data that was not transmitted must again be written to ICDR.

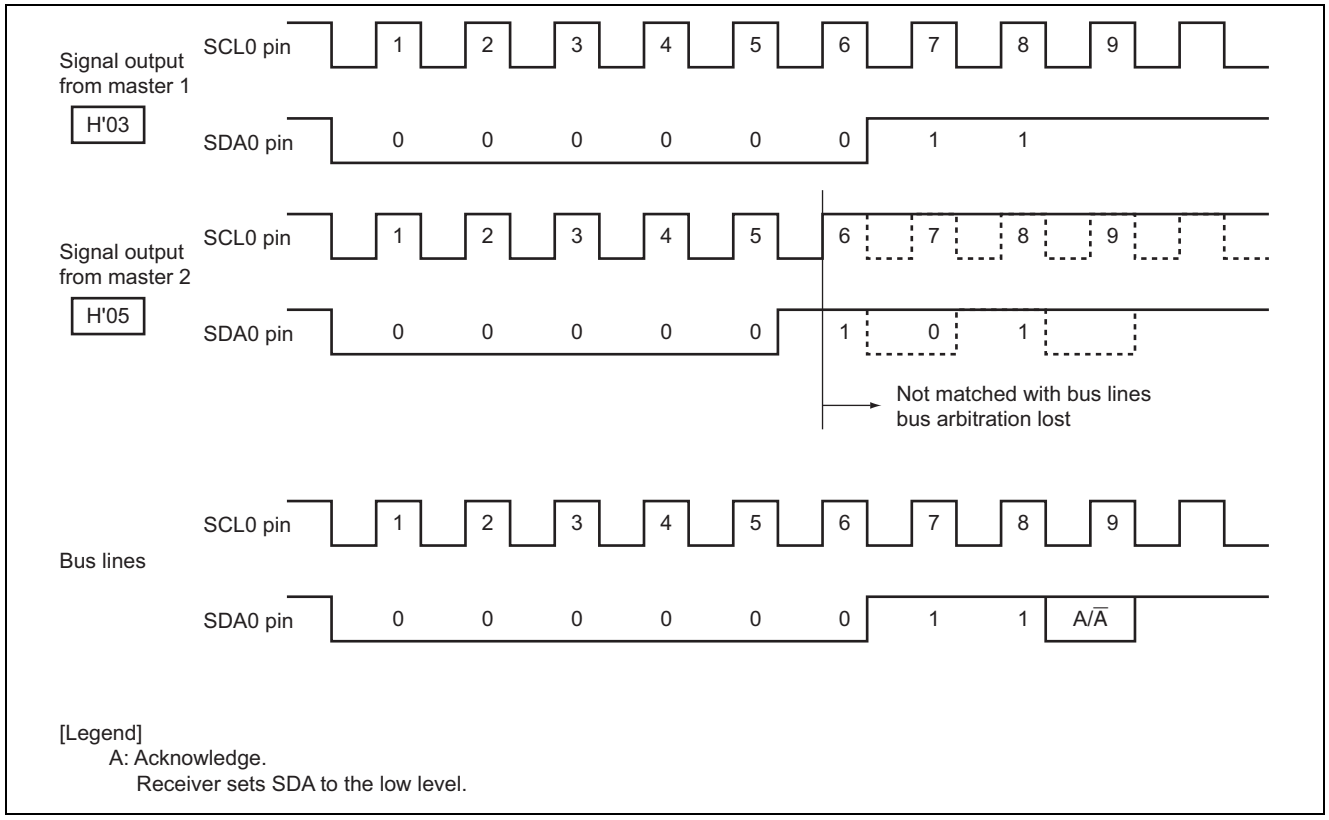


Figure 6 Loss of Bus Arbitration During Data Transmission

4.4 State Transition Diagram

Figure 7 is a state-transition diagram for this sample task. In this sample task, the idle mode is selected as the default.

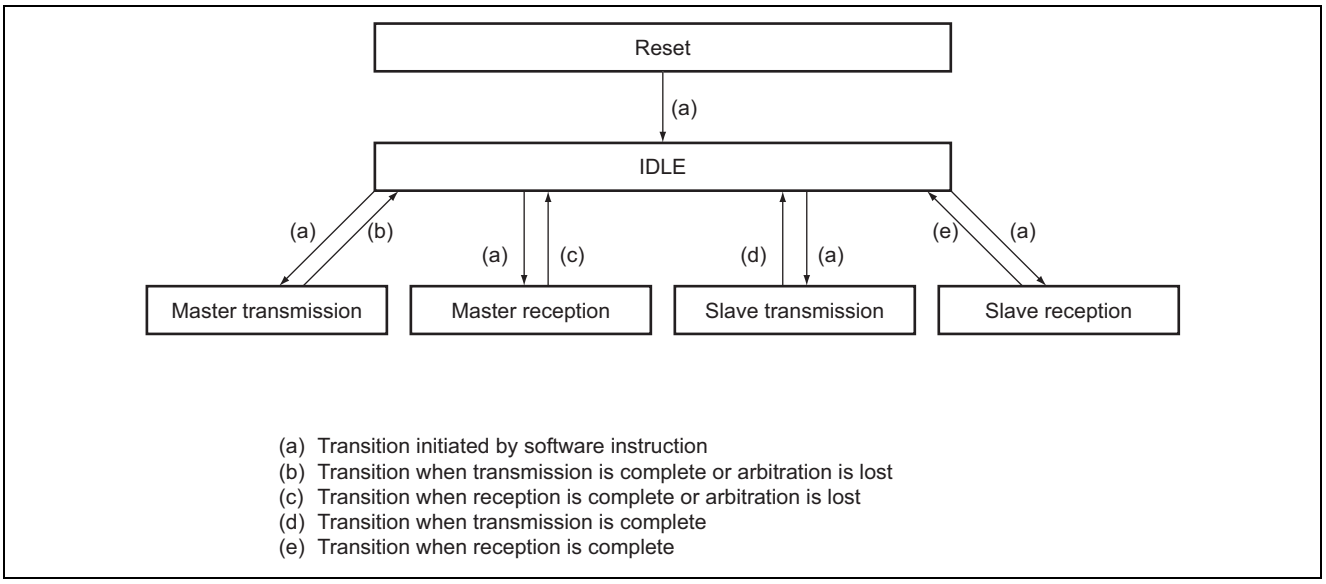


Figure 7 State Transition Diagram

5. Description of Software

5.1 Operating Environment

Table 8 Operating Environment

Item	Detail
Development tool	High-performance Embedded Workshop Ver.4.02.00
C/C++ compiler	H8S, H8/300 Series C/C++ Compiler Ver.6.01.02 (manufactured by Renesas Technology)
H8SX compiler options	-cpu = h8sxa:24:md, -code = machinecode, -optimize = 1, -regparam = 3 -speed = (register, shift, struct, expression)
Evaluation board	Master 1: H8SX/1663 evaluation board
	Master 2: H8SX/1663 evaluation board
	Slave: H8SX/1663 evaluation board

Table 9 Setting of Sections

Address	Section	Description
H'001000	P	Program area
	C	Data table
H'FF2000	B	Non-initialized data area (RAM area)

Table 10 Interrupt and Exception Handling Vector Table

Exception Handling Source	Vector Number	Vector Table Address	Exception Handling Routine
Task “Reset”	0	H'000000	init
WDT interrupt	81	H'000144	wovi_int
IICI0 interrupt	216	H'000360	iici0_int

5.2 List of Functions

Table 11 List of Functions: main.c File

Function Name	Description
init	Initialization routine Sets the CCR and configures the clocks, releases the required modules from module stop mode, and calls the main function.
main	Main routine • Defining macro for MASTER1 and MASTER2 Selects master mode operation, judges the state of the $\overline{\text{IRQ0}}$ pin, and handles master transmission/reception processing. • Defining macro for SLAVE Selects slave mode operation and handles slave transmission/reception processing.
wovi_int	WDT interval timer interrupt

Table 12 List of Functions: iic.c File

Function Name	Description
iic_init	I²C bus interface initialization routine
mtrs_start	Sets I²C bus interface master transmission. Issues the start condition.
mrcv_start	Sets I²C bus interface master reception. Issues the start condition.
iici0_int	Handler for I²C bus interface interrupts. According to the state of operations, the functions for receiving the stop condition, master transmission, master reception, slave transmission, and slave reception are called from this function.
receive_stop_condition	Detects the stop condition.
master_transfer	When the state of operation of this sample task is master transmission, this function for master-transmission processing is called from the I²C bus interface interrupt handler. One byte of data is transferred per call of this function. When arbitration is lost, this function is transited to idle mode operation.
master_receive	When the state of operation of this sample task is master reception, this function for master-reception processing is called from the I²C bus interface interrupt handler. One byte of data is received per call of this function. When arbitration is lost, this function is transited to idle mode operation.
slave_transfer	When the state of operation of this sample task is slave transmission, this function for slave-transmission processing is called from the I²C bus interface interrupt handler. One byte of data is transferred per call of this function.
slave_receive	When the state of operation of this sample task is slave reception, this function for slave-reception processing is called from the I²C bus interface interrupt handler. One byte of data is received per call of this function.

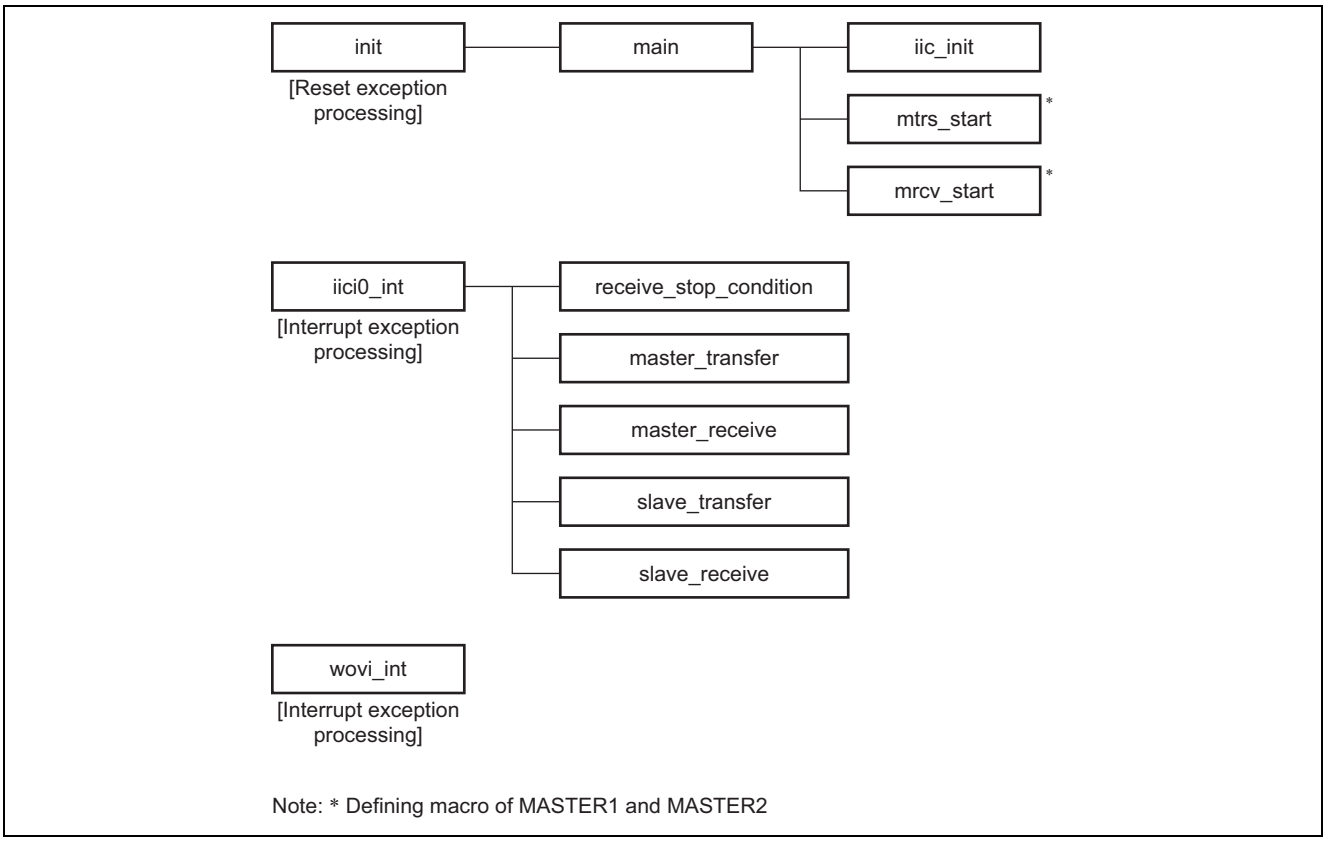


Figure 8 Hierarchy of Calls in the User Program

5.3 RAM Usage

Table 13 Description of RAM Usage

Type	Variable Name	Description	Used in
unsigned char	iic_mode	Sets state of processing by this sample task.	iic_init mtrs_start mrcv_start iici0_int receive_stop_condition master_transfer master_receive
unsigned short	mt_cnt	Counter used for master transmission	main iic_init mtrs_start master_transfer
unsigned short	mr_cnt	Counter used for master reception	main iic_init mrcv_start master_receive
unsigned short	st_cnt	Counter used for slave transmission	main iic_init slave_transfer
unsigned short	sr_cnt	Counter used for slave reception	main iic_init slave_receive
unsigned char	alcnt	Counter used for number of generation of arbitration lost	main master_transfer master_receive
unsigned short	mt_num	Number of bytes for master transmission	mtrs_start master_transfer
unsigned short	mr_num	Number of bytes for master reception	mrcv_start master_receive
unsigned short	st_num	Number of bytes for slave transmission	main slave_transfer
unsigned char	*mt_data	Pointer to data for transmission	mtrs_start master_transfer
unsigned char	*mr_data	Pointer to data for reception	mrcv_start master_receive
unsigned char	MRcv_dt[128]	Master-side receive area	main
unsigned char	SRcv_dt[128]	Slave-side receive area	main

5.4 Constants

Table 14 Constants

Type	Variable Name	Setting	Description	Used in
unsigned char	MTrs_dt[128]	H'81, H'01, H'02 … …… H'7E, H'7F	Data for master transmission 1 when defining macro of MASTER1	master_transfer
unsigned char	MTrs_dt[128]	H'82, H'01, H'02 … …… H'7E, H'7F	Data for master transmission 2 when defining macro of MASTER2	master_transfer

5.5 Macro Definition

Table 15 Macro Definition

Identifier	Description	Used in
MASTER1	Generates program of master 1.	main
MASTER2	Generates program of master 2.	main
SLAVE	Generates program of slave.	main

5.6 Macro Constants

Table 16 Macro Constants

Variable Name	Setting	Description	Used in
DTNUM	128	Number of data for transmission/reception	main
SLAVE_ADDR	Defining macro of MASTER1: H'02 Defining macro of MASTER2: H'04 Defining macro of SLAVE: H'06	Slave address	iic_init
MT_ID	H'06	Slave address +R/$\overline{W}$(0) bit for master transmission Slave-side slave address + 0 (transmission to the slave)	master_transfer
MR_ID	H'07	Slave address +R/$\overline{W}$(1) bit for master reception Slave-side slave address + 1 (reception from the slave)	master_receive
MODE_MT	4	State of processing of this sample task: Master transmission	mtrs_start iici0_int
MODE_MR	3	State of processing of this sample task: Master reception	mrcv_start iici0_int receive_stop_condition
MODE_ST	2	State of processing of this sample task: Slave transmission	iici0_int
MODE_SR	1	State of processing of this sample task: Slave reception	iici0_int
MODE_IDLE	0	State of processing of this sample task: Idle	main iic_init mtrs_start mrcv_start iici0_int receive_stop_condition master_transfer master_receive

5.7 Functions of File main.c

5.7.1 init Function

1. Functional overview
   Initialization routine which releases the required modules from module stop mode, makes clock settings, and calls the main function.
2. Argument
   None
3. Return value
   None
4. Description of internal registers used
   The internal registers used in this sample task are described below. The settings shown in these tables are the values used in this sample task and differ from the initial values.

- Mode control register (MDCR) Number of bits: 16 Address: H'FFFDC0

Bit	Bit Name	Setting	R/W	Description
15	MDS7	Undefined*	R	Indicates the value set by a mode pin (MD3). When MDCR is read, the input level on the MD3 pin is latched. This latching is released by a reset.
11	MDS3	Undefined*	R	Mode Select 3 to 0 These bits indicate the operating mode selected by the mode pins (MD2 to MD0) (see table 17). When MDCR is read, the signal levels input on pins MD2 to MD0 are latched into these bits. The latches are released by a reset.
10	MDS2	Undefined*	R	
9	MDS1	Undefined*	R	
8	MDS0	Undefined*	R	

Note: * Determined by pins MD3 to MD0.

Table 17 Settings of Bits MDS3 to MDS0

MCU Operating Mode	Pins			MDCR			
	MD2	MD1	MD0	MDS3	MDS2	MDS1	MDS0
2	0	1	0	1	1	0	0
4	1	0	0	0	0	1	0
5	1	0	1	0	0	0	1
6	1	1	0	0	1	0	1
7	1	1	1	0	1	0	0

- System clock control register (SCKCR) Number of bits: 16 Address: H'FFFDC4

Bit	Bit Name	Setting	R/W	Description
10	ICK2	0	R/W	System Clock (Iϕ) Select
9	ICK1	0	R/W	These bits select the frequency of the system clock signal, which is provided to the CPU, DMAC, and DTC
8	ICK0	1	R/W	001: Input clock × 2
6	PCK2	0	R/W	Peripheral Module Clock (Pϕ) Select
5	PCK1	0	R/W	These bits select the frequency of the peripheral module clock.
4	PCK0	1	R/W	001: Input clock × 2
2	BCK2	0	R/W	External Bus Clock (Bϕ) Select
1	BCK1	0	R/W	These bits select the frequency of the external bus clock.
0	BCK0	1	R/W	001: Input clock × 2

- MSTPCRA, MSTPCRB, and MSTPCRC control module stop mode. Setting a bit to 1 places the corresponding module in module stop mode, while clearing the bit to 0 releases the module from module stop mode.

- Module stop control register A (MSTPCRA) Number of bits: 16 Adress: H'FFFDC8

Bit	Bit Name	Setting	R/W	Description
15	ACSE	0	R/W	All-Module-Clock-Stop Mode Enable This bit enables/disables all-module-clock-stop mode for reducing current consumption by stopping the bus controller and I/O port operation when the CPU executes the SLEEP instruction after module stop mode has been set for all of the on-chip peripheral modules controlled by MSTPCR. 0: All-module-clock-stop mode disabled 1: All-module-clock-stop mode enabled
13	MSTPA13	1	R/W	DMA controller (DMAC)
12	MSTPA12	1	R/W	Data transfer controller (DTC)
9	MSTPA9	1	R/W	8-bit timer unit (TMR_3, TMR_2)
8	MSTPA8	1	R/W	8-bit timer unit (TMR_1, TMR_0)
5	MSTPA5	1	R/W	D/A converter (channels 1 and 0)
3	MSTPA3	1	R/W	A/D converter (unit 0)
0	MSTPA0	1	R/W	16-bit timer pulse unit (TPU channels 5 to 0)

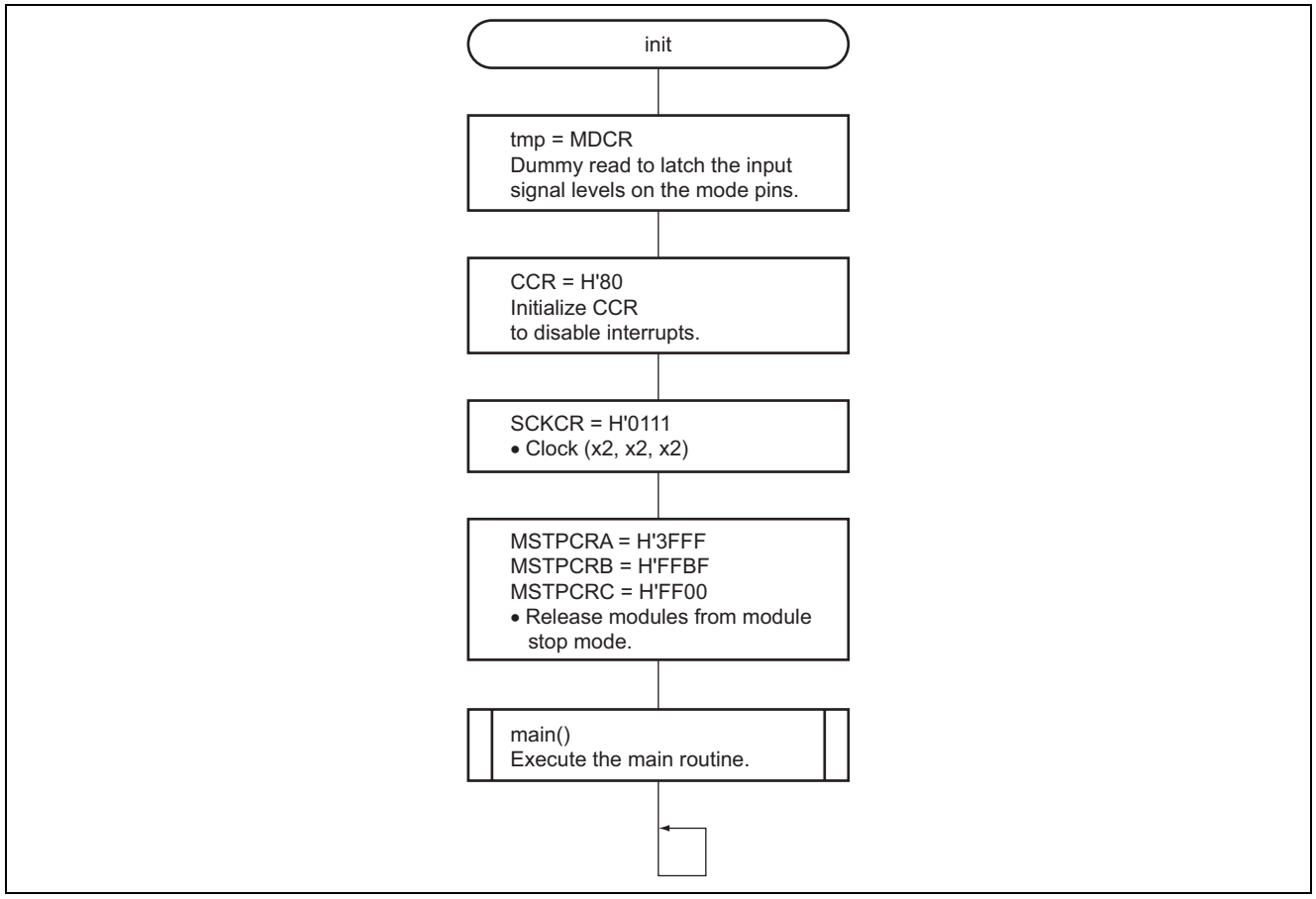
- Module stop control register B (MSTPCRB) Number of bits: 16 Adress: H'FFFDCA

Bit	Bit Name	Setting	R/W	Description
15	MSTPB15	1	R/W	Programmable pulse generator (PPG)
12	MSTPB12	1	R/W	Serial communications interface_4 (SCI_4)
10	MSTPB10	1	R/W	Serial communications interface_2 (SCI_2)
9	MSTPB9	1	R/W	Serial communications interface_1 (SCI_1)
8	MSTPB8	1	R/W	Serial communications interface_0 (SCI_0)
7	MSTPB7	1	R/W	I²C bus interface_1 (IIC_1)
6	MSTPB6	0	R/W	I²C bus interface_0 (IIC_0)

- Module stop control register C (MSTPCRC) Number of bits: 16 Address: H'FFFDCC

Bit	Bit Name	Setting	R/W	Description
15	MSTPC15	1	R/W	Serial communications interface_5 (SCI_5), (IrDA)
14	MSTPC14	1	R/W	Serial communications interface_6 (SCI_6)
13	MSTPC13	1	R/W	8-bit timer unit (TMR_4, TMR_5)
12	MSTPC12	1	R/W	8-bit timer unit (TMR_6, TMR_7)
11	MSTPC11	1	R/W	Universal serial bus interface (USB)
10	MSTPC10	1	R/W	Cyclic redundancy check module
4	MSTPC4	0	R/W	On-chip RAM_4 (H'FF2000 to H'FF3FFF)
3	MSTPC3	0	R/W	On-chip RAM_3 (H'FF4000 to H'FF5FFF)
2	MSTPC2	0	R/W	On-chip RAM_2 (H'FF6000 to H'FF7FFF)
1	MSTPC1	0	R/W	On-chip RAM_1 (H'FF8000 to H'FF9FFF)
0	MSTPC0	0	R/W	On-chip RAM_0 (H'FFA000 to H'FFBFFF)

5. Flowchart



5.7.2 main Function (Defining Macro of MASTER 1 and MASTER 2)

1. Functional overview

- On falling edges of the $\overline{\text{IRQ0}}$ signal, this function performs 128-byte master transmission and 128-byte master reception.
- Compares the master-transmission data with the master-reception data, and outputs an indicator of the results of comparison to pins P32 to P30.

2. Argument
   None

3. Return value
   None

4. Description of internal registers used
   The internal registers used in this sample task are described below. The settings shown in these tables are the values used in this sample task and differ from the initial values.

- I²C bus control register A_0 (ICCRA_0) Number of bits: 8 Address: H'FFFEB0

Bit	Bit Name	Setting	R/W	Description
7	ICE	0	R/W	I²C Bus Interface Enable 0: Disables the IIC2 module. 1: Enables transfer via the IIC2 module (pins SCL and SDA are driving the bus).
5	MST	0	R/W	Master/Slave Select
4	TRS	0	R/W	00: Slave receive mode

- I²C bus control register B_0 (ICCRB_0) Number of bits: 8 Address: H'FFFEB1

Bit	Bit Name	Setting	R/W	Description
5	SDAO	0/1	R	Monitors the level of signal output from SDA. When the SDAO bit is set to 1 in reading, the signal output from SDA is at high level. When the SDAO bit is set to 0 in reading, the signal output from SDA is at low level.
3	SCLO	0	R	Monitors level of signal output from SCL. When the SCLO bit is set to 1 in reading, the signal output from SCL is at high level. When the SCLO bit is set to 0 in reading, the signal output from SCL is at low level.

- IRQ sense control register L (ISCRL) Number of bits: 16 Address: H'FFFD6A

Bit	Bit Name	Setting	R/W	Description
1	IRQ0SR	0	R/W	IRQ0 Sense Control Rise
0	IRQ0SF	1	R/W	IRQ0 Sense Control Fall 01: Generation of an interrupt request at the falling edge of $\overline{\text{IRQ0}}$ input.

- Port 3 data direction register (P3DDR) Number of bits: 8 Address: H'FFFB82

Bit	Bit Name	Setting	R/W	Description
2	P32DDR	1	R/W	0: Sets the P32 pin as an input pin. 1: Sets the P32 pin as an output pin.
1	P31DDR	1	R/W	0: Sets the P31 pin as an input pin. 1: Sets the P31 pin as an output pin.
0	P30DDR	1	R/W	0: Sets the P30 pin as an input pin. 1: Sets the P30 pin as an output pin.

- IRQ status register (ISR) Number of bits: 16 Address: H'FFFF36

Bit	Bit Name	Setting	R/W	Description
0	IRQ0F	0	R/(W)*	IRQ0 Enable 0: No generation of an IRQ0 interrupt 1: Generation of an IRQ0 interrupt

Note: * Only 0 can be written here, to clear the flag.

- Port 3 data register (P3DR) Number of bits: 8 Address: H'FFFF52

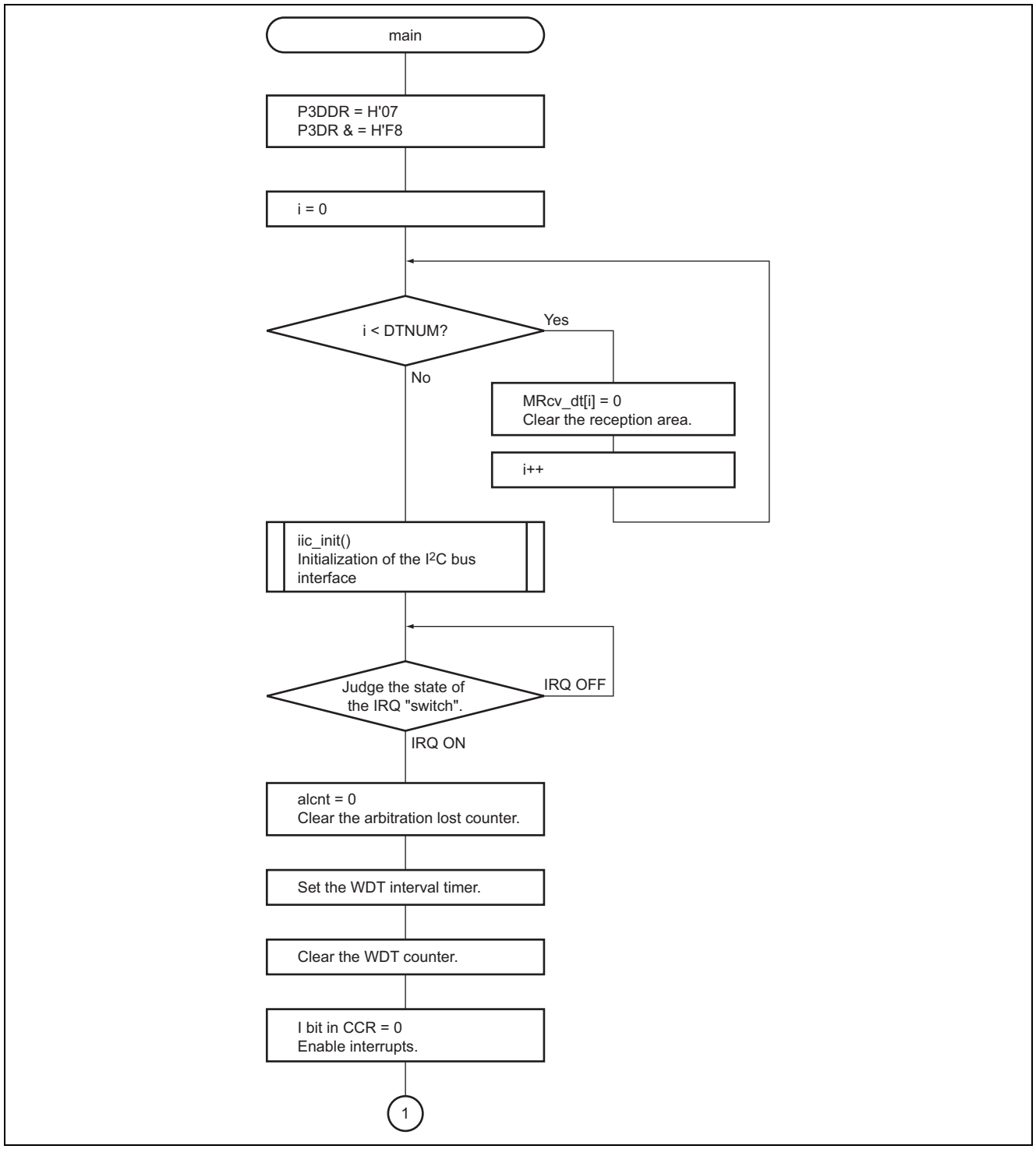
Bit	Bit Name	Setting	R/W	Description
2	P32DR	0/1	R/W	0: P32 pin is set to the low level. 1: P32 pin is set to the high level.
1	P31DR	0/1	R/W	0: P31 pin is set to the low level. 1: P31 pin is set to the high level.
0	P30DR	0/1	R/W	0: P30 pin is set to the low level. 1: P30 pin is set to the high level.

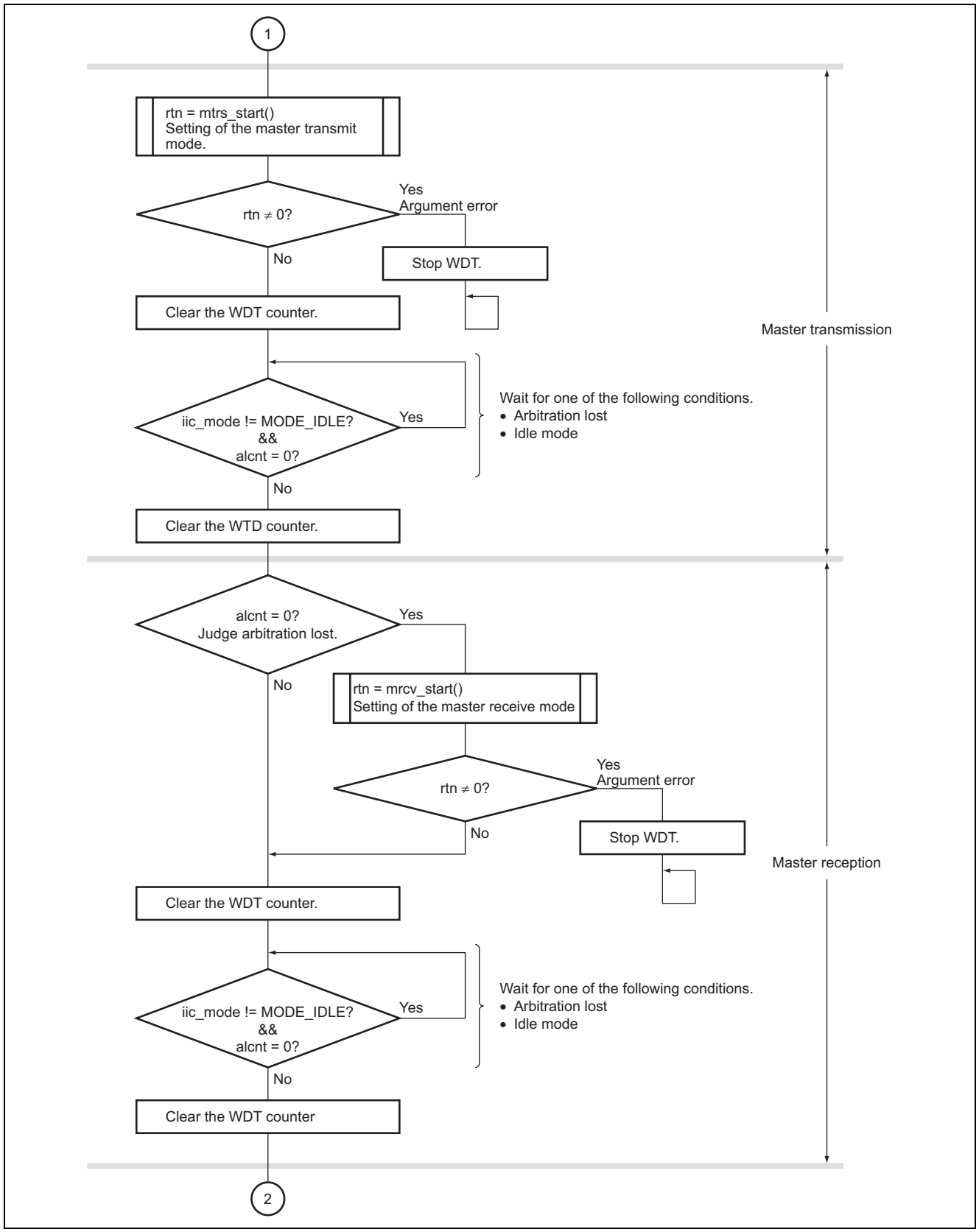
- Timer control/status register (TCSR) Number of bits: 8 Address: H'FFFFA4

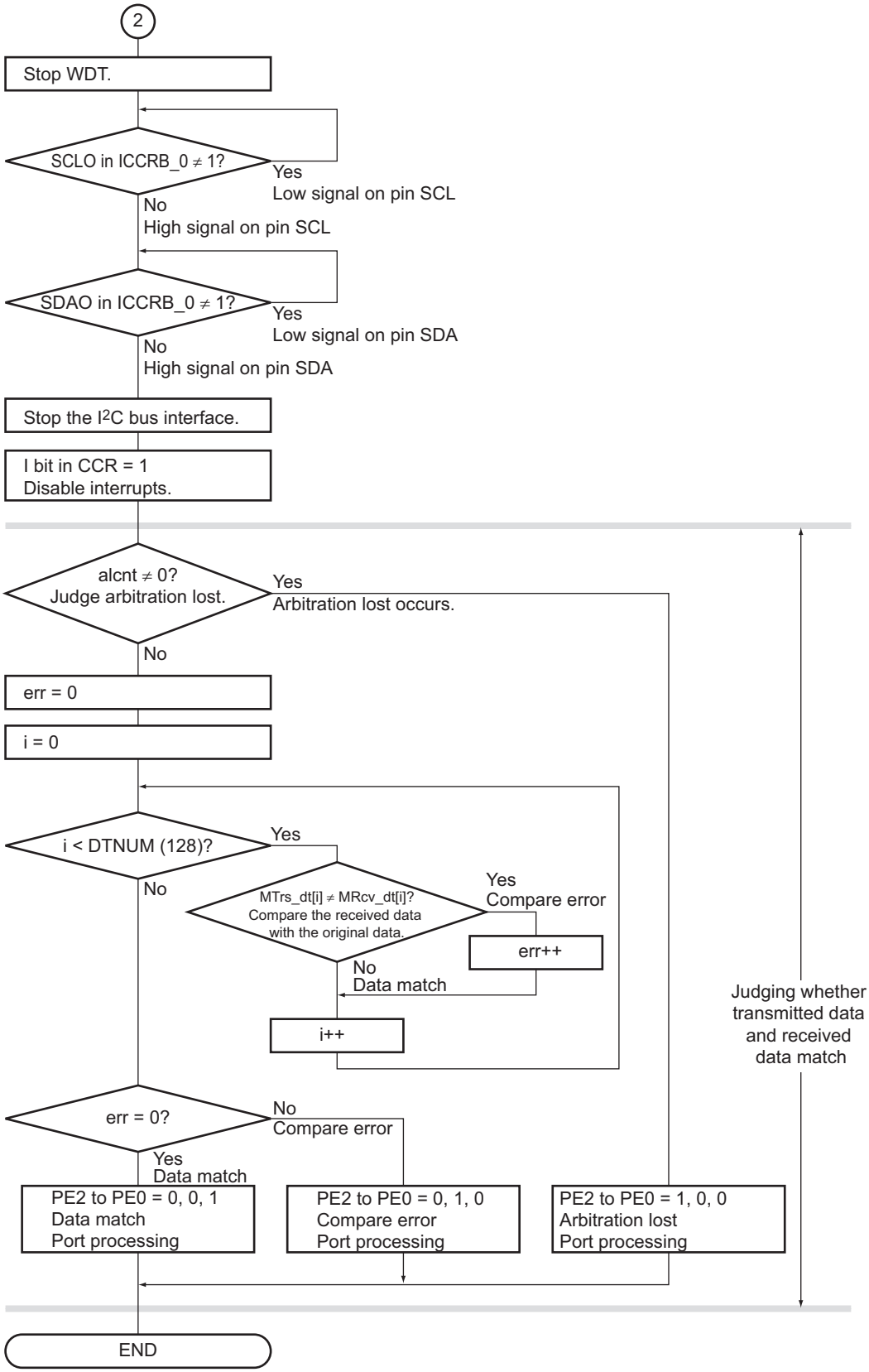
Bit	Bit Name	Setting	R/W	Description
6	WT/$\overline{IT}$	0	R/W	Timer Mode Select 0: Used as interval timer mode. 1: Used as watchdog timer mode.
5	TME	1	R/W	Timer Enable 0: TCNT stops counting and is initialized to H'00. 1: TCNT starts counting.
2	CKS2	1	R/W	Clock Select 2 to 0
1	CKS1	1	R/W	Select clocks for input to TCNT.
0	CKS0	0	R/W	110: Clock Pϕ/32768 When Pϕ is 24 MHz, overflow period is 184.8 ms.

- Timer counter (TCNT) Number of bits: 8 Address: H'FFFFA4 (in writing) and H'FFFFA5 (in reading)
  This bit is an 8-bit readable and writable up-counter.
  Setting: H'00

5. Flowchart


main
P3DDR = H'07
P3DR & = H'F8
i = 0
i < DTNUM?
Yes
No
MRcv_dt[i] = 0
Clear the reception area.
i++
iic_init()
Initialization of the I2C bus interface
Judge the state of the IRQ "switch".
IRQ OFF
IRQ ON
alcnt = 0
Clear the arbitration lost counter.
Set the WDT interval timer.
Clear the WDT counter.
I bit in CCR = 0
Enable interrupts.
1


1
rtn = mtrs_start()
Setting of the master transmit mode.
rtn ≠ 0?
Yes
Argument error
Stop WDT.
No
Clear the WDT counter.
Master transmission
iic_mode != MODE_IDLE?
&&
alcnt = 0?
Yes
Wait for one of the following conditions.
• Arbitration lost
• Idle mode
No
Clear the WTD counter.
alcnt = 0?
Judge arbitration lost.
Yes
No
rtn = mrcv_start()
Setting of the master receive mode
rtn ≠ 0?
Yes
Argument error
No
Stop WDT.
Master reception
Clear the WDT counter.
iic_mode != MODE_IDLE?
&&
alcnt = 0?
Yes
Wait for one of the following conditions.
• Arbitration lost
• Idle mode
No
Clear the WDT counter
2


2
Stop WDT.
SCLO in ICCRB_0 ≠ 1?
Yes
Low signal on pin SCL
No
High signal on pin SCL
SDAO in ICCRB_0 ≠ 1?
Yes
Low signal on pin SDA
No
High signal on pin SDA
Stop the I2C bus interface.
I bit in CCR = 1
Disable interrupts.
alcnt ≠ 0?
Judge arbitration lost.
Yes
Arbitration lost occurs.
No
err = 0
i = 0
i < DTNUM (128)?
Yes
No
MTrs_dt[i] ≠ MRcv_dt[i]?
Compare the received data
with the original data.
Yes
Compare error
err++
No
Data match
i++
Judging whether
transmitted data
and received
data match
err = 0?
No
Compare error
Yes
Data match
PE2 to PE0 = 0, 0, 1
Data match
Port processing
PE2 to PE0 = 0, 1, 0
Compare error
Port processing
PE2 to PE0 = 1, 0, 0
Arbitration lost
Port processing
END

5.7.3 main Function (Defining Macro of SLAVE)

1. Functional overview

- Receives 128 bytes of data from master side and transmits the 128-byte received data to master side.
- Judges the first byte of received data. When the address is H'81, outputs P34 = 1. When the address is H'82, outputs P35 = 1.

2. Argument
   None

3. Return value
   None

4. Description of internal registers used
   The internal registers used in this sample task are described below. The settings shown in these tables are the values used in this sample task and differ from the initial values.

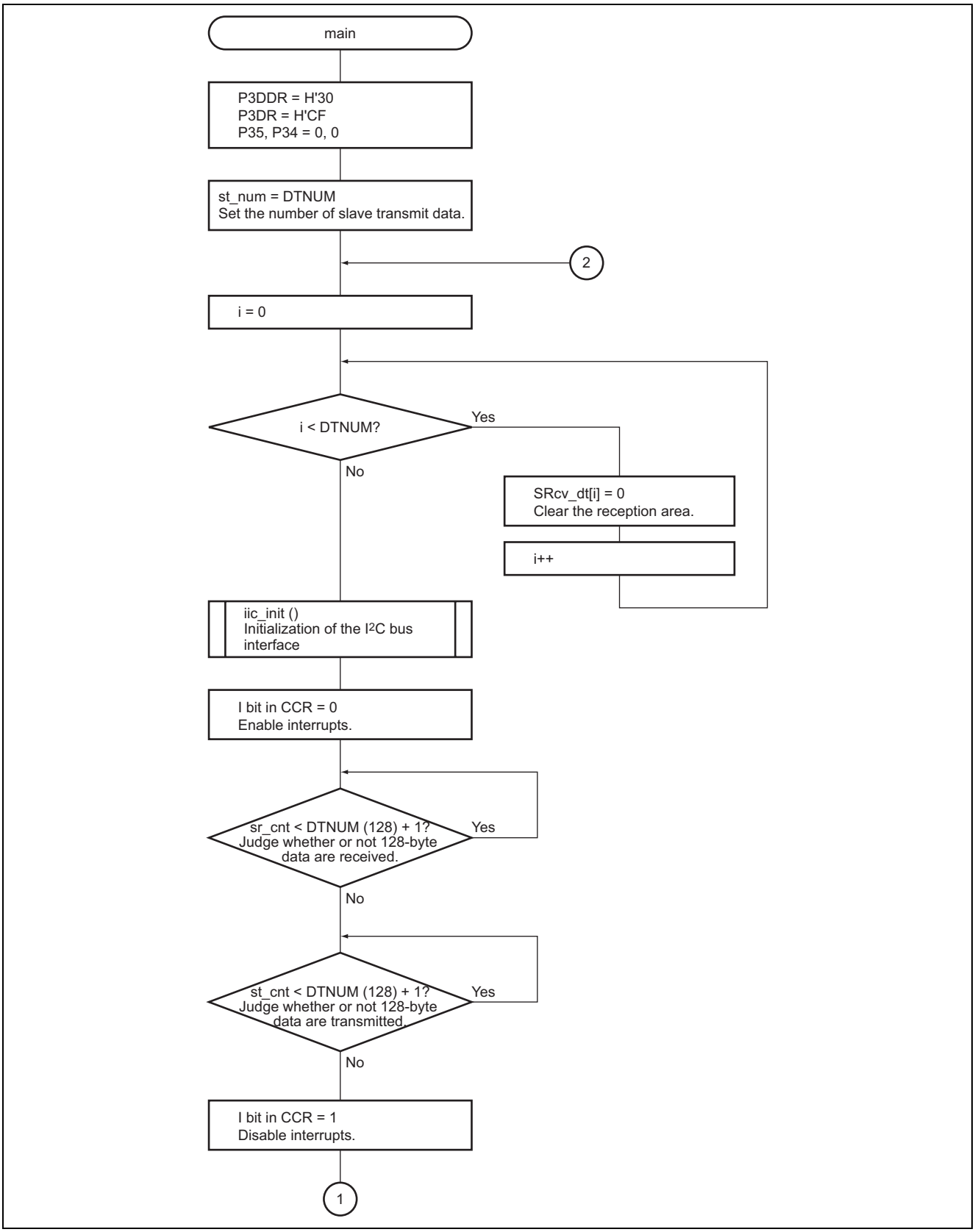
- Port 3 data direction register (P3DDR) Number of bits: 8 Address: H'FFFB82

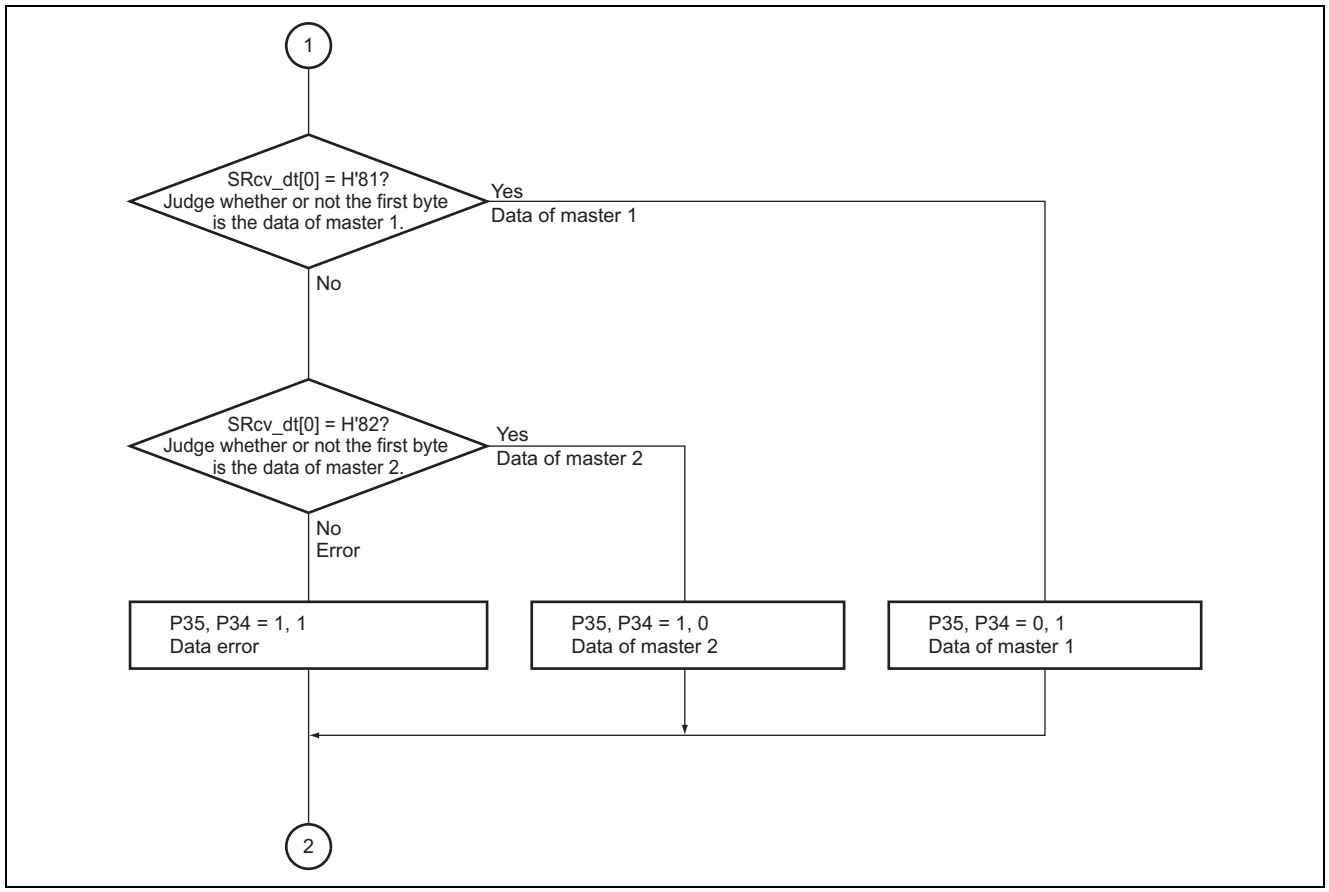
Bit	Bit Name	Setting	R/W	Description
5	P35DDR	1	R/W	0: Sets the P35 pin as an input pin. 1: Sets the P35 pin as an output pin.
4	P34DDR	1	R/W	0: Sets the P34 pin as an input pin. 1: Sets the P34 pin as an output pin.

- Port 3 data register (P3DR) Number of bits: 8 Address: H'FFFF52

Bit	Bit Name	Setting	R/W	Description
5	P35DR	0/1	R/W	0: P35 pin is set to the low level. 1: P35 pin is set to the high level.
4	P34DR	0/1	R/W	0: P34 pin is set to the low level. 1: P34 pin is set to the high level.

5. Flowchart


main
P3DDR = H'30
P3DR = H'CF
P35, P34 = 0, 0
st_num = DTNUM
Set the number of slave transmit data.
2
i = 0
i < DTNUM?
Yes
No
SRcv_dt[i] = 0
Clear the reception area.
i++
iic_init ()
Initialization of the I2C bus interface
I bit in CCR = 0
Enable interrupts.
sr_cnt < DTNUM (128) + 1?
Judge whether or not 128-byte data are received.
Yes
No
st_cnt < DTNUM (128) + 1?
Judge whether or not 128-byte data are transmitted.
Yes
No
I bit in CCR = 1
Disable interrupts.
1


1
SRcv_dt[0] = H'81?
Judge whether or not the first byte is the data of master 1.
Yes
Data of master 1
No
SRcv_dt[0] = H'82?
Judge whether or not the first byte is the data of master 2.
Yes
Data of master 2
No
Error
P35, P34 = 1, 1
Data error
P35, P34 = 1, 0
Data of master 2
P35, P34 = 0, 1
Data of master 1
2

5.7.4 wovi_int Function

1. Functional overview
   This is the handler for the WDT interval timer overflow interrupt. When the I²C bus interface hangs because of noise or some other factor, the WDT counter will overflow, generating the interrupt. The handler executes recovery processing for the I²C bus interface.
2. Argument
   None
3. Return value
   None
4. Description of internal registers used
   The internal registers used in this sample task are described below. The settings shown in these tables are the values used in this sample task and differ from the initial values.

- I²C bus control register A_0 (ICCRA_0) Number of bits: 8 Address: H'FFFEB0

Bit	Bit Name	Setting	R/W	Description
7	ICE	1	R/W	I²C Bus Interface Enable 0: Disables the IIC2 module. 1: Enables transfer via the IIC2 module (pins SCL and SDA are driving the bus).
5	MST	1	R/W	Master/Slave Select
4	TRS	1	R/W	11: Master transmit mode

- I²C bus control register B_0 (ICCRB_0) Number of bits: 8 Address: H'FFFEB1

Bit	Bit Name	Setting	R/W	Description
7	BBSY	0/1	R/W	Bus Busy The BBSY flag can be read to check whether the I²C bus (SCL, SDA) is busy or free. In master mode, this bit is also used to issue start and stop conditions. A high-to-low transition of SDA while SCL is high is recognized as a start condition, setting BBSY to 1. A low-to-high transition of SDA while SCL is high is recognized as a stop condition, clearing BBSY to 0. Using a MOV instruction to write 1 in BBSY and 0 in SCP issues a start condition. A retransmit start condition is issued in the same way. Using a MOV instruction to write 0 in BBSY and SCP issues a stop condition.
6	SCP	0	R/W	Start Condition/Stop Condition Prohibit Bit Controls the issuing of start and stop conditions in master mode. Writing 1 in BBSY and 0 in SCP issues a start condition. A retransmit start condition is issued in the same way. Writing 0 in BBSY and SCP issues a stop condition. This bit is always read as 1. Writing of 1 has no effect.
1	IICRST	1	R/W	I²C Bus Interface Control Part Reset This bit resets control parts except for I²C registers. If this bit is set to 1 when hang-up is occurred because of communication failure during I²C operation, I²C control part can be reset without setting ports and initializing registers.

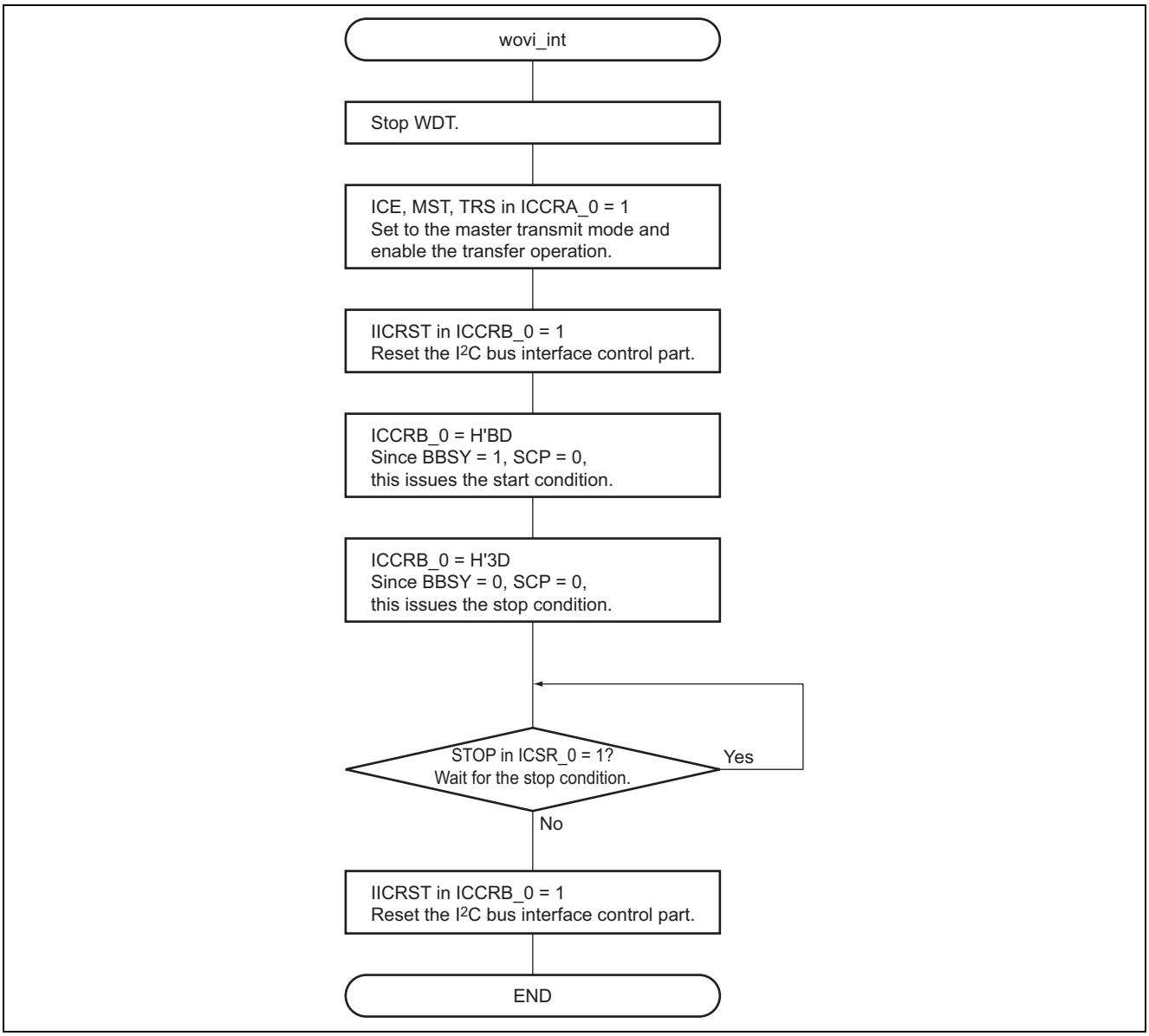
- I²C bus status register_0 (ICSR_0) Number of bits: 8 Address: H'FFFEB4

Bit	Bit Name	Setting	R/W	Description
3	STOP	Undefined	R/W	Stop Condition Detection Flag [Setting conditions] • Detection of a stop condition after completion of frame transfer in master mode • Detection of a stop condition after match of the first-byte address after general call and detection of start condition and address set in SAR in slave mode [Clearing condition] • Writing 0 in STOP after reading STOP as 1

- Timer control/status register (TCSR) Number of bits: 8 Address: H'FFFFA4

Bit	Bit Name	Setting	R/W	Description
5	TME	0	R/W	Timer Enable 0: TCNT stops counting and is initialized to H'00. 1: TCNT starts counting.

5. Flowchart


wovi_int
Stop WDT.
ICE, MST, TRS in ICCRA_0 = 1
Set to the master transmit mode and
enable the transfer operation.
IICRST in ICCRB_0 = 1
Reset the I2C bus interface control part.
ICCRB_0 = H'BD
Since BBSY = 1, SCP = 0,
this issues the start condition.
ICCRB_0 = H'3D
Since BBSY = 0, SCP = 0,
this issues the stop condition.
STOP in ICSR_0 = 1?
Wait for the stop condition.
Yes
No
IICRST in ICCRB_0 = 1
Reset the I2C bus interface control part.
END

5.8 Functions of File iic.c

5.8.1 iic_init Function

1. Functional overview
   I²C bus interface initialization routine
2. Argument
   None
3. Return value
   None
4. Description of internal registers used
   The internal registers used in this sample task are described below. The settings shown in these tables are the values used in this sample task and differ from the initial values.

- Port 1 input buffer control register (P1ICR) Number of bits: 8 Address: H'FFFB90

Bit	Bit Name	Setting	R/W	Description
7	P17ICR	1	R/W	0: Disables the input buffer of P17 (SCL0) pin. 1: Enables the input buffer of P17 (SCL0) pin.
6	P16ICR	1	R/W	0: Disables the input buffer of P16 (SDA0) pin. 1: Enables the input buffer of P16 (SDA0) pin.

- I²C bus control register A_0 (ICCRA_0) Number of bits: 8 Address: H'FFFEB0

Bit	Bit Name	Setting	R/W	Description
7	ICE	1	R/W	I²C Bus Interface Enable 0: Disables the IIC2 module. 1: Enables transfer via the IIC2 module (pins SCL and SDA are driving the bus).
6	RCVD	0	R/W	Reception Disable Enables or disables the next operation when TRS is 0 and ICDRR is read. 0: Enables the next reception. 1: Disables the next reception.
5	MST	0	R/W	Master/Slave Select
4	TRS	0	R/W	00: Slave receive mode
3	CKS3	1	R/W	Transfer Clock Select 3 to 0
2	CKS2	1	R/W	1111: Transfer rate is 93.75 kbits/s with Pϕ = 24 MHz.
1	CKS1	1	R/W	
0	CKS0	1	R/W	

- I²C bus mode register_0 (ICMR_0) Number of bits: 8 Address: H'FFFEB2

Bit	Bit Name	Setting	R/W	Description
6	WAIT	0	R/W	Wait Insertion Bit This bit selects whether to insert a wait on completion of data transfer other than the acknowledge bit. When WAIT is set to 1, after the falling edge of the clock cycle for the final data bit, the low period is extended for two cycles of the transfer clock. When WAIT is cleared to 0, data and acknowledge bits are transferred consecutively with no wait inserted. The setting of this bit is invalid in slave mode. Note that in usage with WAIT set to 1, when the slave device pulls SCL to the low level after the low period of SDA in the eighth and ninth clock cycles has extended for at least two cycles of the transfer clock, the high period of the ninth cycle of the transfer clock may be shortened. In such cases, the WAIT setting in this situation should be changed to 0. Except under this condition, however, there is no problem with usage.

- I²C bus interrupt enable register_0 (ICIER_0) Number of bits: 8 Address: H'FFFEB3

Bit	Bit Name	Setting	R/W	Description
7	TIE	1	R/W	Transmit Interrupt Enable Enables or disables the transmit data empty interrupt (TXI) when the TDRE bit in ICSR is set to 1. 0: Disables the transmit data empty interrupt request (TXI). 1: Enables the transmit data empty interrupt request (TXI).
6	TEIE	0	R/W	Transmit End Interrupt Enable Enables or disables the transmit end interrupt (TEI) at the rising of the ninth clock while the TDRE bit in ICSR is 1. TEI can be canceled by clearing the TEND bit or the TEIE bit to 0. 0: Disables the transmit end interrupt request (TEI). 1: Enables the transmit end interrupt request (TEI).
5	RIE	1	R/W	Receive Interrupt Enable Enables or disables the receive data full interrupt request (RXI) when a received data is transferred from ICDRS to ICDRR and the RDRF bit in ICSR is set to 1. RXI can be canceled by clearing the RDRF or RIE bit to 0. 0: Disables the receive data full interrupt request (RXI). 1: Enables the receive data full interrupt request (RXI).
4	NAKIE	1	R/W	NACK Receive Interrupt Enable Enables or disables the NACK receive interrupt request (NAKI) when the NACKF and AL bits in ICSR are set to 1. NAKI can be canceled by clearing the NACKF, AL, or NAKIE bit to 0. 0: Disables the NACK receive interrupt request (NAKI). 1: Enables the NACK receive interrupt request (NAKI).
3	STIE	1	R/W	Stop Condition Detection Interrupt Enable 0: Disables the stop condition detection interrupt request (STPI). 1: Enables the stop condition detection interrupt request (STPI).
2	ACKE	1	R/W	Acknowledge Bit Judgment Select 0: The value of the acknowledge bit is ignored, and continuous transfer is performed. 1: When the acknowledge bit is 1, continuous transfer is interrupted.
0	ACKBT	0	R/W	Transmit Acknowledge Specifies the bit to be sent at the acknowledge timing in receive mode. 0: 0 is output at acknowledge timing. 1: 1 is output at acknowledge timing.

- I²C bus status register_0 (ICSR_0) Number of bits: 8 Address: H'FFFEB4

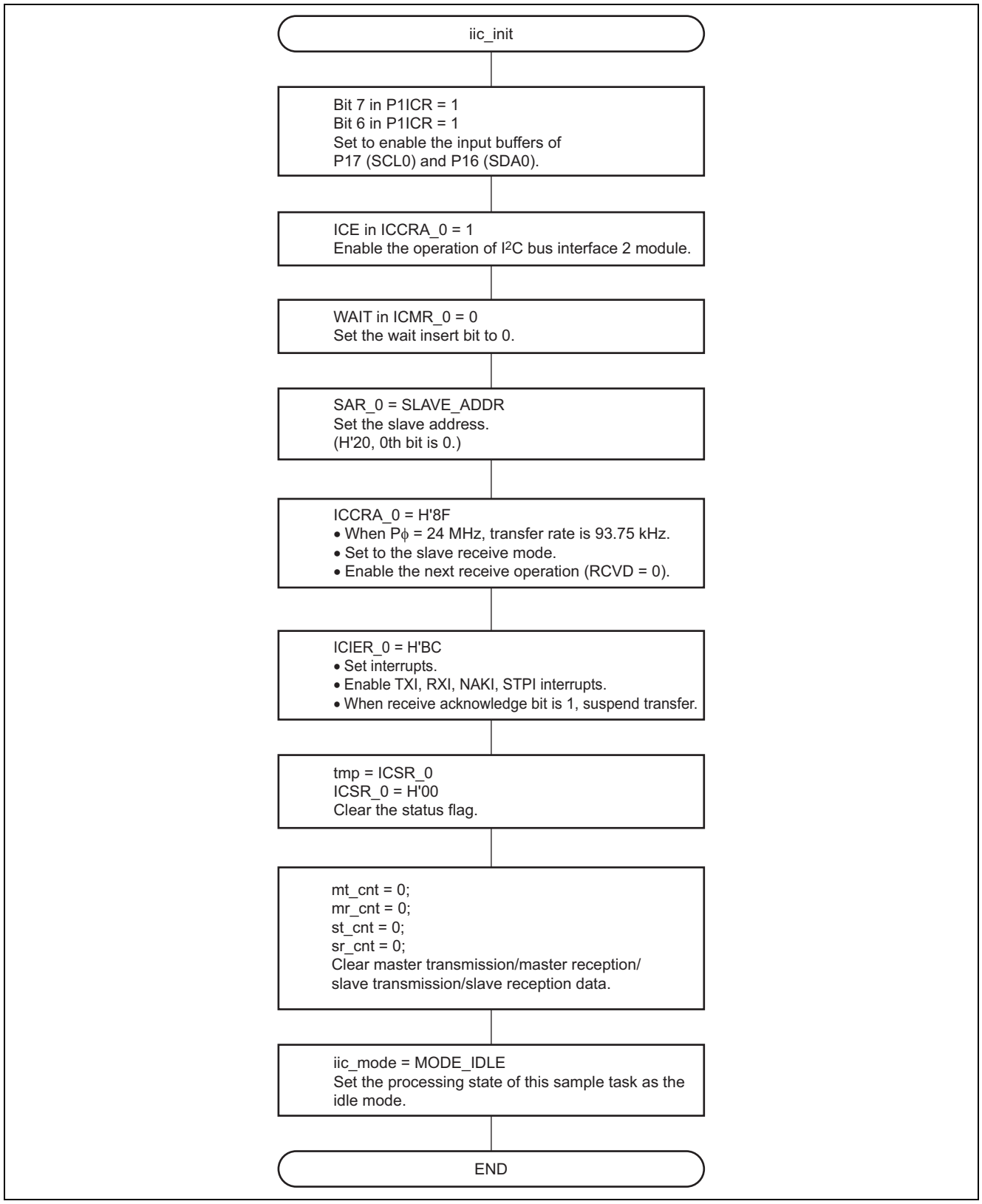
Bit	Bit Name	Setting	R/W	Description
7	TDRE	0	R/W	Transmit Data Register Empty [Setting conditions] • Transferring of data from ICDRT to ICDRS and having ICDRT empty • Setting of TRS • Issuing of a start condition (including retransmission) • Transition from the receive mode to the transmit mode has been made in the slave mode [Clearing conditions] • Writing of 0 in TDRE after reading TDRE as 1 • Writing of data in ICDRT
6	TEND	0	R/W	Transmit End [Setting condition] • Rising of the ninth clock of SCL while the TDRE flag is 1 [Clearing conditions] • Writing 0 in TEND after reading TEND as 1 • Writing of data in ICDRT
5	RDRF	0	R/W	Receive Data Register Full [Setting condition] • Transferring of received data from ICDRS to ICDRR [Clearing conditions] • Writing of 0 in RDRF after reading RDRF as 1 • Reading of data from ICDRR
4	NACKF	0	R/W	No Acknowledge Detection Flag [Setting condition] • Detection of no acknowledge from the receive device in transmission while the ACKE bit in ICIER is 1 [Clearing condition] • Writing of 0 in NACKF after reading NACKF as 1
3	STOP	0	R/W	Stop Condition Detection Flag [Setting conditions] • Detection of a stop condition after completion of frame transfer in master mode • Detection of a stop condition after match of the first-byte address after general call and detection of start condition and address set in SAR in slave mode [Clearing condition] • Writing of 0 in STOP after reading STOP as 1

Bit	Bit Name	Setting	R/W	Description
2	AL	0	R/W	Arbitration Lost Flag Indicates that arbitration was lost in the master mode. When two or more master devices attempt to seize the bus at nearly the same time, if the I²C bus interface detects data differing from the data it sent, it sets AL to 1 to indicate that the bus has been taken by another master. [Setting conditions] • Different values for the internal SDA signal and SDA pin on a rising edge of SCL in master transmit mode • The SDA pin being at the high level in master mode while a start condition is detected [Clearing condition] • Writing of 0 in AL after reading AL as 1
1	AAS	0	R/W	Slave Address Recognition Flag In slave receive mode, this flag is set to 1 when the first frame following a start condition matches bits SVA6 to SVA0 in SAR. [Setting conditions] • Detection of the slave address in slave receive mode • Detection of the general call address in the slave receive mode [Clearing condition] • Writing of 0 in AAS after reading AAS as 1

- Slave address register_0 (SAR_0) Number of bits: 8 Address: H'FFFEB5
  The slave address is set in the SAR bits. An interface in slave mode responds as the slave device when the 7 higher-order bits of SAR match the 7 higher-order bits of the first frame received after a start condition.

Bit	Bit Name	Setting	R/W	Description
7 to1	SVA6 to SVA0	SLAVE_ADDR	R/W	Slave Address 6 to 0 Unique address setting (address differing from the addresses of other slave devices connected to the I²C bus) for the device
0	–		R/W	Reserved This bit is readable/writable. 0 should be written in writing.

5. Flowchart


iic_init
Bit 7 in P1ICR = 1
Bit 6 in P1ICR = 1
Set to enable the input buffers of
P17 (SCL0) and P16 (SDA0).
ICE in ICCRA_0 = 1
Enable the operation of I²C bus interface 2 module.
WAIT in ICMR_0 = 0
Set the wait insert bit to 0.
SAR_0 = SLAVE_ADDR
Set the slave address.
(H'20, 0th bit is 0.)
ICCRA_0 = H'8F
• When Pϕ = 24 MHz, transfer rate is 93.75 kHz.
• Set to the slave receive mode.
• Enable the next receive operation (RCVD = 0).
ICIER_0 = H'BC
• Set interrupts.
• Enable TXI, RXI, NAKI, STPI interrupts.
• When receive acknowledge bit is 1, suspend transfer.
tmp = ICSR_0
ICSR_0 = H'00
Clear the status flag.
mt_cnt = 0;
mr_cnt = 0;
st_cnt = 0;
sr_cnt = 0;
Clear master transmission/master reception/
slave transmission/slave reception data.
iic_mode = MODE_IDLE
Set the processing state of this sample task as the
idle mode.
END

5.8.2 mtrs_start Function

1. Functional overview

   This function sets up the task for I²C bus interface master transmission and issues the start condition.

2. Argument

Type	Name of Variable	Description
const unsigned char	*dtadd	First address of data for transmission
unsigned short	dtnum	Number of data to be transmitted

3. Return value

Type	Description
unsigned char	0: Starts normal transmission. 1: Transmission in progress 2: Bus busy 3: Argument error

4. Description of internal registers used

   The internal registers used in this sample task are described below. The settings shown in these tables are the values used in this sample task and differ from the initial values.

- I²C bus control register A_0 (ICCRA_0) Number of bits: 8 Address: H'FFFEB0

Bit	Bit Name	Setting	R/W	Description
5	MST	1	R/W	Master/ Slave Select
4	TRS	1	R/W	11: Master transmit mode

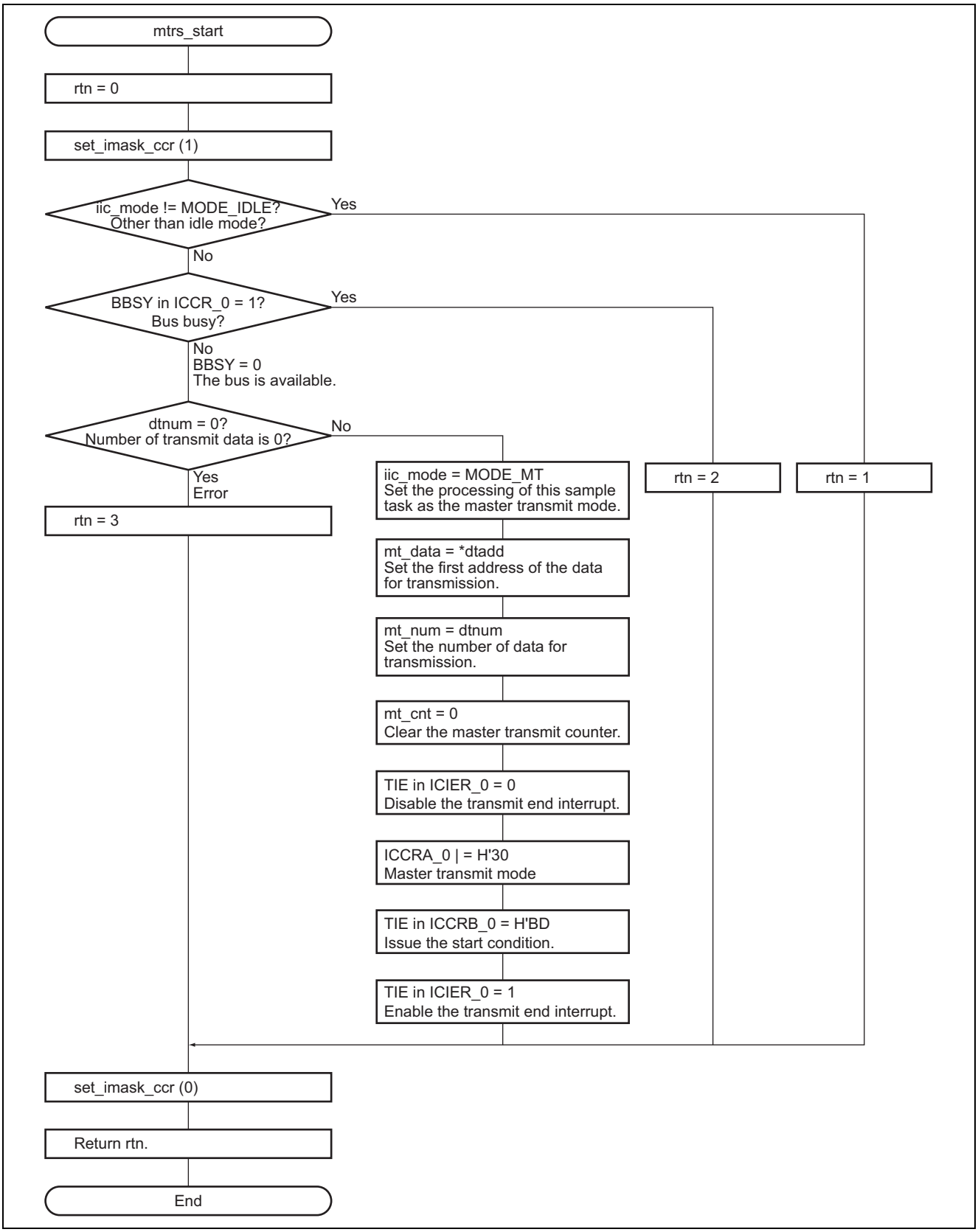
- I²C bus control register B_0 (ICCRB_0) Number of bits: 8 Address: H'FFFEB1

Bit	Bit Name	Setting	R/W	Description
7	BBSY	1	R/W	Bus Busy The BBSY flag can be read to check whether the I²C bus (SCL, SDA) is busy or free. In master mode, this bit is also used to issue start and stop conditions. A high-to-low transition of SDA while SCL is high is recognized as a start condition, setting BBSY to 1. A low-to-high transition of SDA while SCL is high is recognized as a stop condition, clearing BBSY to 0. Using a MOV instruction to write 1 in BBSY and 0 in SCP issues a start condition. A retransmit start condition is issued in the same way. Using a MOV instruction to write 0 in BBSY and SCP issues a stop condition.
6	SCP	0	R/W	Start Condition/Stop Condition Prohibit Bit Controls the issuing of start and stop conditions in master mode. Writing 1 in BBSY and 0 in SCP issues a start condition. A retransmit start condition is issued in the same way. Writing 0 in BBSY and SCP issues a stop condition. This bit is always read as 1. Writing of 1 has no effect.

- I²C bus interrupt enable register_0 (ICIER_0) Number of bits: 8 Address: H'FFFEB3

Bit	Bit Name	Setting	R/W	Description
7	TIE	1	R/W	Transmit Interrupt Enable Enables or disables the transmit data empty interrupt (TXI) when the TDRE bit in ICSR is set to 1. 0: Disables the transmit data empty interrupt request (TXI). 1: Enables the transmit data empty interrupt request (TXI).

5. Flowchart


mtrs_start
rtn = 0
set_imask_ccr (1)
iic_mode != MODE_IDLE?
Other than idle mode?
Yes
No
BBSY in ICCR_0 = 1?
Bus busy?
Yes
No
BBSY = 0
The bus is available.
dtnum = 0?
Number of transmit data is 0?
No
Yes
Error
rtn = 3
iic_mode = MODE_MT
Set the processing of this sample task as the master transmit mode.
mt_data = *dtadd
Set the first address of the data for transmission.
mt_num = dtnum
Set the number of data for transmission.
mt_cnt = 0
Clear the master transmit counter.
TIE in ICIER_0 = 0
Disable the transmit end interrupt.
ICCRA_0 | = H'30
Master transmit mode
TIE in ICCRB_0 = H'BD
Issue the start condition.
TIE in ICIER_0 = 1
Enable the transmit end interrupt.
rtn = 2
rtn = 1
set_imask_ccr (0)
Return rtn.
End

5.8.3 mrcv_start Function

1. Functional overview

This function sets up the task for I²C bus interface master reception and issues the start condition.

2. Argument

Type	Name of Variable	Description
const unsigned char	*dtadd	First address of received data
unsigned short	dtnum	Number of data to be received

3. Return value

Type	Description
unsigned char	0: Starts normal transmission. 1: Transmission in progress 2: Bus busy 3: Argument error

4. Description of internal registers used

The internal registers used in this sample task are described below. The settings shown in these tables are the values used in this sample task and differ from the initial values.

- I²C bus control register A_0 (ICCRA_0) Number of bits: 8 Address: H'FFFEB0

Bit	Bit Name	Setting	R/W	Description
5	MST	1	R/W	Master/Slave Select
4	TRS	1	R/W	11: Master transmit mode

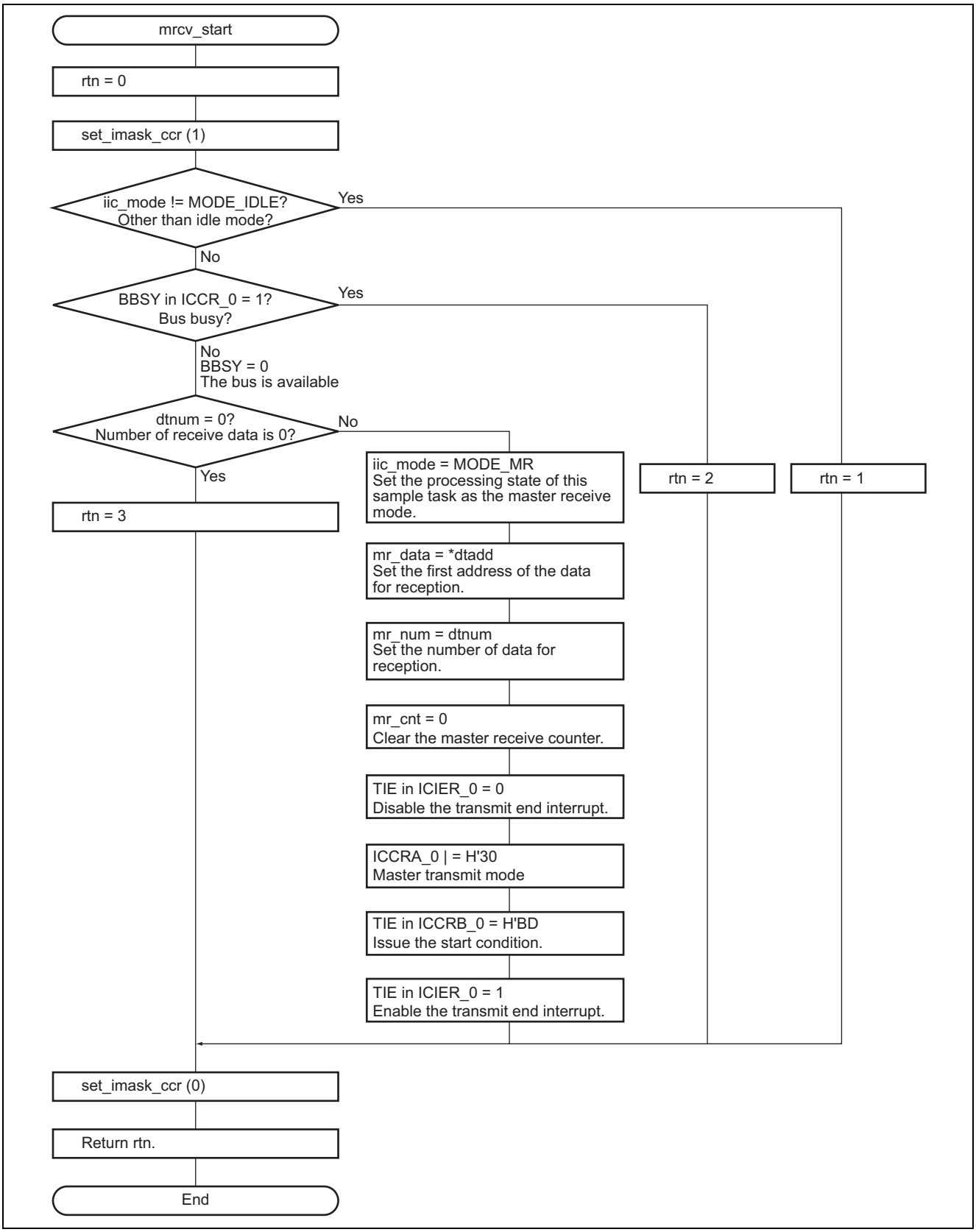
- I²C bus control register B_0 (ICCRB_0) Number of bits: 8 Address: H'FFFEB1

Bit	Bit Name	Setting	R/W	Description
7	BBSY	1	R/W	Bus Busy The BBSY flag can be read to check whether the I²C bus (SCL, SDA) is busy or free. In master mode, this bit is also used to issue start and stop conditions. A high-to-low transition of SDA while SCL is high is recognized as a start condition, setting BBSY to 1. A low-to-high transition of SDA while SCL is high is recognized as a stop condition, clearing BBSY to 0. Using a MOV instruction to write 1 in BBSY and 0 in SCP issues a start condition. A retransmit start condition is issued in the same way. Using a MOV instruction to write 0 in BBSY and SCP issues a stop condition.
6	SCP	0	R/W	Start Condition/Stop Condition Prohibit Bit Controls the issuing of start and stop conditions in master mode. Writing 1 in BBSY and 0 in SCP issues a start condition. A retransmit start condition is issued in the same way. Writing 0 in BBSY and SCP issues a stop condition. This bit is always read as 1. Writing of 1 has no effect.

- I²C bus interrupt enable register_0 (ICIER_0) Number of bits: 8 Address: H'FFFEB3

Bit	Bit Name	Setting	R/W	Description
7	TIE	1	R/W	Transmit Interrupt Enable Enables or disables the transmit data empty interrupt (TXI) when the TDRE bit in ICSR is set to 1. 0: Disables the transmit data empty interrupt request (TXI). 1: Enables the transmit data empty interrupt request (TXI).

5. Flowchart


mrcv_start
rtn = 0
set_imask_ccr (1)
iic_mode != MODE_IDLE?
Other than idle mode?
Yes
No
BBSY in ICCR_0 = 1?
Bus busy?
Yes
No
BBSY = 0
The bus is available
dtnum = 0?
Number of receive data is 0?
No
Yes
rtn = 3
iic_mode = MODE_MR
Set the processing state of this sample task as the master receive mode.
mr_data = *dtadd
Set the first address of the data for reception.
mr_num = dtnum
Set the number of data for reception.
mr_cnt = 0
Clear the master receive counter.
TIE in ICIER_0 = 0
Disable the transmit end interrupt.
ICCRA_0 | = H'30
Master transmit mode
TIE in ICCRB_0 = H'BD
Issue the start condition.
TIE in ICIER_0 = 1
Enable the transmit end interrupt.
rtn = 2
rtn = 1
set_imask_ccr (0)
Return rtn.
End

5.8.4 iici0_int Function

1. Functional overview
   Handler for I²C bus interface interrupts. According to the state of operations, this function calls the functions for receiving the stop condition, master transmission, and master reception, slave transmission, and slave reception.
2. Argument
   None
3. Return value
   None
4. Description of internal registers used
   The internal registers used in this sample task are described below. The settings shown in these tables are the values used in this sample task and differ from the initial values.

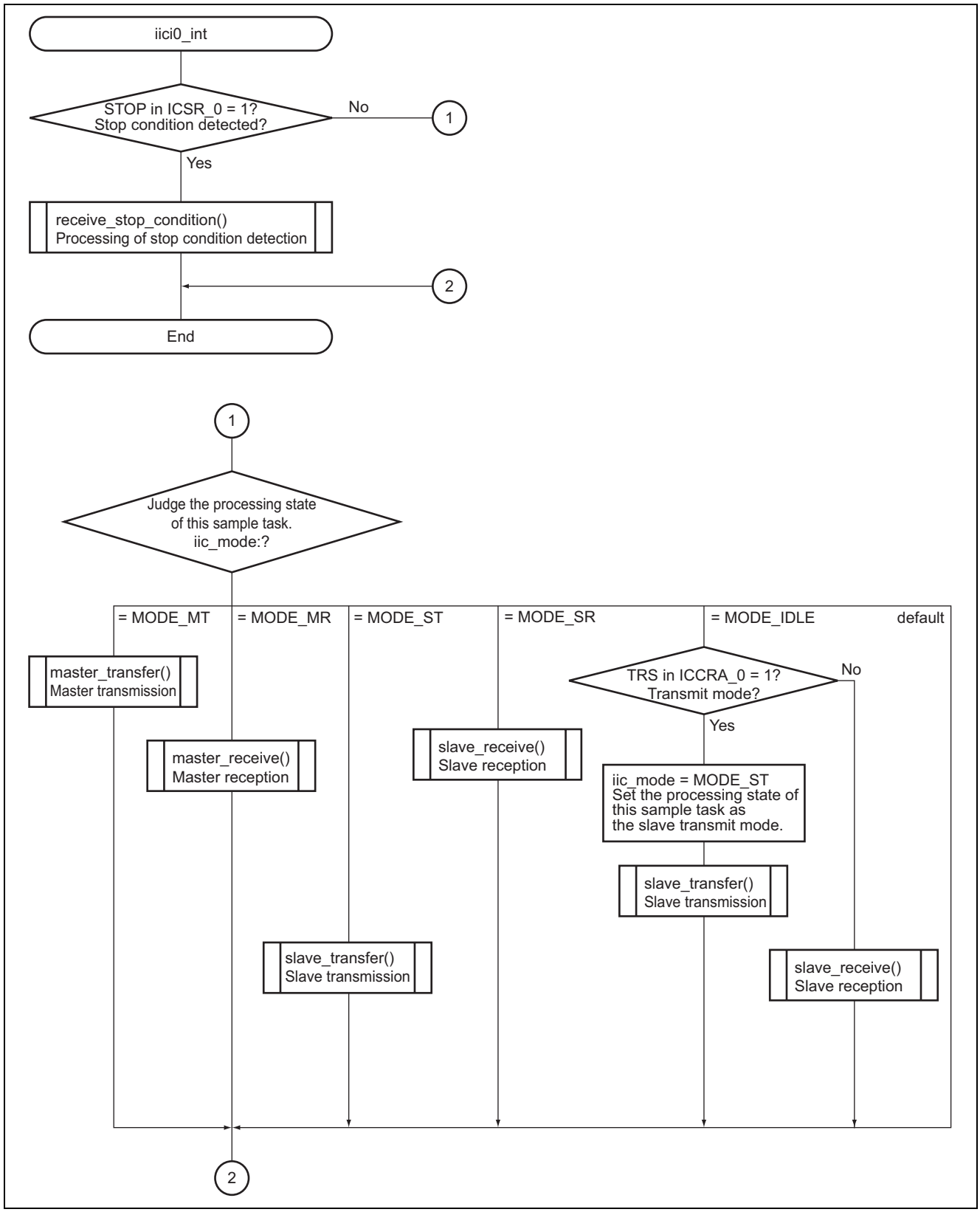
- I²C bus control register A_0 (ICCRA_0) Number of bits: 8 Address: H'FFFEB0

Bit	Bit Name	Setting	R/W	Description
4	TRS	Undefined	R/W	Master/Slave Select 0: Receive mode 1: Transmit mode

- I²C bus status register_0 (ICSR_0) Number of bits: 8 Address: H'FFFEB4

Bit	Bit Name	Setting	R/W	Description
3	STOP	Undefined	R/W	Stop Condition Detection Flag [Setting conditions] • Detection of a stop condition after completion of frame transfer in master mode • Detection of a stop condition after match of the first-byte address after general call and detection of start condition and address set in SAR in slave mode [Clearing condition] • Writing 0 in STOP after reading STOP as 1

5. Flowchart


iici0_int
STOP in ICSR_0 = 1?
Stop condition detected?
No
1
Yes
receive_stop_condition()
Processing of stop condition detection
2
End
1
Judge the processing state
of this sample task.
iic_mode:?
= MODE_MT
= MODE_MR
= MODE_ST
= MODE_SR
= MODE_IDLE
default
master_transfer()
Master transmission
master_receive()
Master reception
slave_transfer()
Slave transmission
slave_receive()
Slave reception
TRS in ICCRA_0 = 1?
Transmit mode?
No
Yes
iic_mode = MODE_ST
Set the processing state of
this sample task as
the slave transmit mode.
slave_transfer()
Slave transmission
slave_receive()
Slave reception
2

5.8.5 receive_stop_condition Function

1. Functional overview
   This function handles processing on detection of the stop condition.
2. Argument
   None
3. Return value
   None
4. Description of internal registers used
   The internal registers used in this sample task are described below. The settings shown in these tables are the values used in this sample task and differ from the initial values.

- I²C bus control register A_0 (ICCRA_0) Number of bits: 8 Address: H'FFFEB0

Bit	Bit Name	Setting	R/W	Description
6	RCVD	0	R/W	Reception Disable Enables or disables the next operation when TRS is 0 and ICDRR is read. 0: Enables the next reception. 1: Disables the next reception.
5	MST	0	R/W	Master/Slave Select
4	TRS	0	R/W	00: Slave receive mode

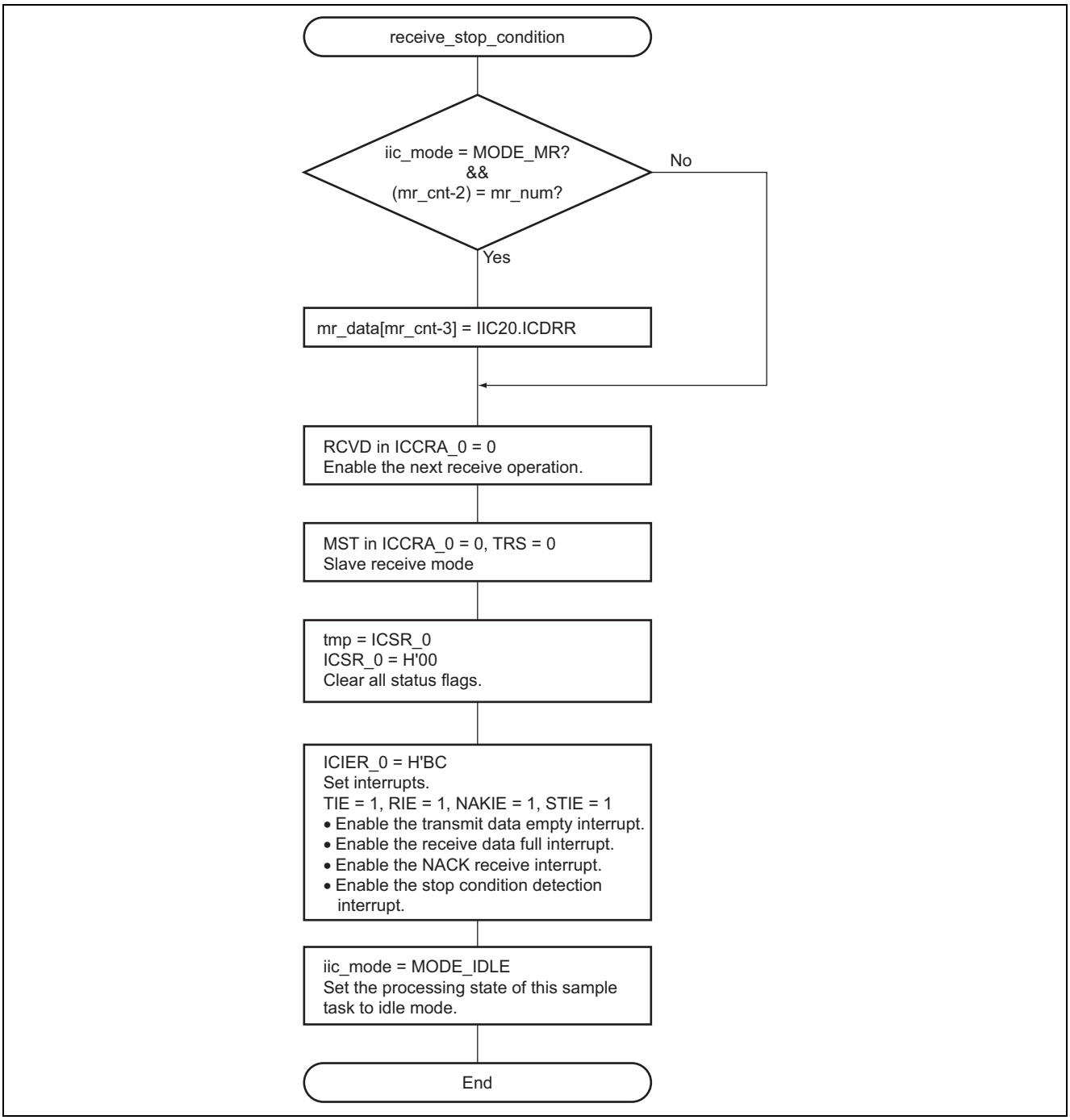
- I²C bus interrupt enable register_0 (ICIER_0) Number of bits: 8 Address: H'FFFEB3

Bit	Bit Name	Setting	R/W	Description
7	TIE	1	R/W	Transmit Interrupt Enable Enables or disables the transmit data empty interrupt (TXI) when the TDRE bit in ICSR is set to 1. 0: Disables the transmit data empty interrupt request (TXI). 1: Enables the transmit data empty interrupt request (TXI).
6	TEIE	0	R/W	Transmit End Interrupt Enable Enables or disables the transmit end interrupt (TEI) at the rising of the ninth clock while the TDRE bit in ICSR is 1. TEI can be canceled by clearing the TEND bit or the TEIE bit to 0. 0: Disables the transmit end interrupt request (TEI). 1: Enables the transmit end interrupt request (TEI).
5	RIE	1	R/W	Receive Interrupt Enable Enables or disables the receive data full interrupt request (RXI) when a received data is transferred from ICDRS to ICDRR and the RDRF bit in ICSR is set to 1. RXI can be canceled by clearing the RDRF or RIE bit to 0. 0: Disables the receive data full interrupt request (RXI). 1: Enables the receive data full interrupt request (RXI).
4	NAKIE	1	R/W	NACK Receive Interrupt Enable Enables or disables the NACK receive interrupt request (NAKI) when the NACKF and AL bits in ICSR are set to 1. NAKI can be canceled by clearing the NACKF, AL, or NAKIE bit to 0. 0: Disables the NACK receive interrupt request. 1: Enables the NACK receive interrupt request.
3	STIE	1	R/W	Stop Condition Detection Interrupt Enable 0: Disables the stop condition detection interrupt request (STPI). 1: Enables the stop condition detection interrupt request (STPI).
2	ACKE	1	R/W	Acknowledge Bit Judgment Select 0: The value of the acknowledge bit is ignored, and continuous transfer is performed. 1: When the acknowledge bit is 1, continuous transfer is interrupted.
0	ACKBT	0	R/W	Transmit Acknowledge Specifies the bit to be sent at the acknowledge timing in receive mode. 0: 0 is output at acknowledge timing. 1: 1 is output at acknowledge timing.

- I²C bus status register_0 (ICSR_0) Number of bits: 8 Address: H'FFFEB4

Bit	Bit Name	Setting	R/W	Description
7	TDRE	0	R/W	Transmit Data Register Empty [Setting conditions] • Transferring of data from ICDRT to ICDRS and having ICDRT empty • Setting of TRS • Issuing of a start condition (including retransmission) • Transition from the receive mode to the transmit mode has been made in the slave mode [Clearing conditions] • Writing of 0 in TDRE after reading it as 1 • Writing of data in ICDRT
4	NACKF	0	R/W	No Acknowledge Detection Flag [Setting condition] • Detection of no acknowledge from the receive device in transmission while the ACKE bit in ICIER is 1 [Clearing condition] • Writing of 0 in NACKF after reading it as 1
1	AAS	0	R/W	Slave Address Recognition Flag In slave receive mode, this flag is set to 1 when the first frame following a start condition matches bits SVA6 to SVA0 in SAR. [Setting conditions] • Detection of the slave address in slave receive mode • Detection of the general call address in the slave receive mode [Clearing condition] • Writing of 0 in AAS after reading AAS as 1

5. Flowchart


receive_stop_condition
iic_mode = MODE_MR?
&&
(mr_cnt-2) = mr_num?
No
Yes
mr_data[mr_cnt-3] = IIC20.ICDRR
RCVD in ICCRA_0 = 0
Enable the next receive operation.
MST in ICCRA_0 = 0, TRS = 0
Slave receive mode
tmp = ICSR_0
ICSR_0 = H'00
Clear all status flags.
ICIER_0 = H'BC
Set interrupts.
TIE = 1, RIE = 1, NAKIE = 1, STIE = 1
• Enable the transmit data empty interrupt.
• Enable the receive data full interrupt.
• Enable the NACK receive interrupt.
• Enable the stop condition detection interrupt.
iic_mode = MODE_IDLE
Set the processing state of this sample task to idle mode.
End

5.8.6 master_transfer Function

1. Functional overview
   Master-transmission processing which is called from the I²C bus interface interrupt handler. In this case, the interrupt source will be the transmit data empty interrupt for each byte of transmitted data. When arbitration is lost, it places the interface in slave receive mode.
2. Argument
   None
3. Return value
   None
4. Description of internal registers used
   The internal registers used in this sample task are described below. The settings shown in these tables are the values used in this sample task and differ from the initial values.

- I²C bus control register B_0 (ICCRB_0) Number of bits: 8 Address: H'FFFEB1

Bit	Bit Name	Setting	R/W	Description
7	BBSY	0	R/W	Bus Busy The BBSY flag can be read to check whether the I²C bus (SCL, SDA) is busy or free. In master mode, this bit is also used to issue start and stop conditions. A high-to-low transition of SDA while SCL is high is recognized as a start condition, setting BBSY to 1. A low-to-high transition of SDA while SCL is high is recognized as a stop condition, clearing BBSY to 0. Using a MOV instruction to write 1 in BBSY and 0 in SCP issues a start condition. A retransmit start condition is issued in the same way. Using a MOV instruction to write 0 in BBSY and SCP issues a stop condition.
6	SCP	0	R/W	Start Condition/Stop Condition Prohibit Bit Controls the issuing of start and stop conditions in master mode. Writing 1 in BBSY and 0 in SCP issues a start condition. A retransmit start condition is issued in the same way. Writing 0 in BBSY and SCP issues a stop condition. This bit is always read as 1. Writing of 1 has no effect.
3	SCLO	Undefined	R	Monitors level of signal output from SCL. When the SCLO bit is set to 1 in reading, the signal output from SCL is at high level. When the SCLO bit is set to 0 in reading, the signal output from SCL is at low level.

- I²C bus interrupt enable register_0 (ICIER_0) Number of bits: 8 Address: H'FFFEB3

Bit	Bit Name	Setting	R/W	Description
7	TIE	0/1	R/W	Transmit Interrupt Enable Enables or disables the transmit data empty interrupt (TXI) when the TDRE bit in ICSR is set to 1. 0: Disables the transmit data empty interrupt request (TXI). 1: Enables the transmit data empty interrupt request (TXI).
6	TEIE	0/1	R/W	Transmit End Interrupt Enable Enables or disables the transmit end interrupt (TEI) at the rising of the ninth clock while the TDRE bit in ICSR is 1. TEI can be canceled by clearing the TEND bit or the TEIE bit to 0. 0: Disables the transmit end interrupt request (TEI). 1: Enables the transmit end interrupt request (TEI).

- I²C bus status register_0 (ICSR_0) Number of bits: 8 Address: H'FFFEB4

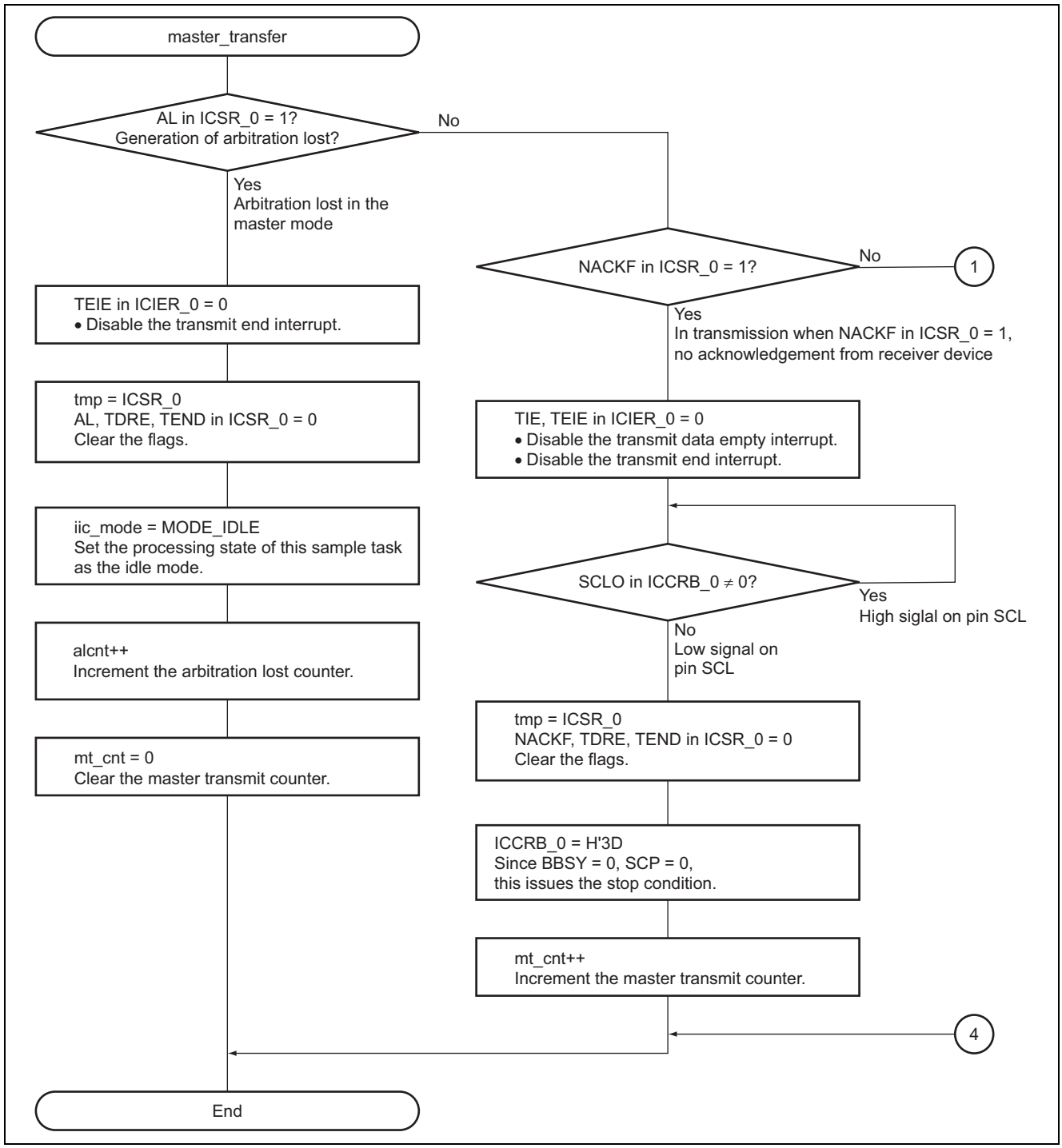
Bit	Bit Name	Setting	R/W	Description
7	TDRE	0	R/W	Transmit Data Register Empty [Setting conditions] • Transferring of data from ICDRT to ICDRS and having ICDRT empty • Setting of TRS • Issuing of a start condition (including retransmission) • Transition from the receive mode to the transmit mode has been made in the slave mode [Clearing conditions] • Writing of 0 in TDRE after reading it as 1 • Writing of data in ICDRT
6	TEND	0	R/W	Transmit End [Setting condition] • Rising of the ninth clock of SCL while the TDRE flag is 1 [Clearing conditions] • Writing 0 in TEND after reading it as 1 • Writing of data in ICDRT
4	NACKF	0	R/W	No Acknowledge Detection Flag [Setting condition] • Detection of no acknowledge from the receive device in transmission while the ACKE bit in ICIER is 1 [Clearing condition] • Writing of 0 in NACKF after reading it as 1
2	AL	0	R/W	Arbitration Lost Flag Indicates that arbitration was lost in the master mode. When two or more master devices attempt to seize the bus at nearly the same time, if the I²C bus interface detects data differing from the data it sent, it sets AL to 1 to indicate that the bus has been taken by another master. [Setting conditions] • Different values for the internal SDA signal and SDA pin on a rising edge of SCL in master transmit mode • The SDA pin being at the high level in master mode while a start condition is detected [Clearing condition] • Writing of 0 in AL after reading AL as 1

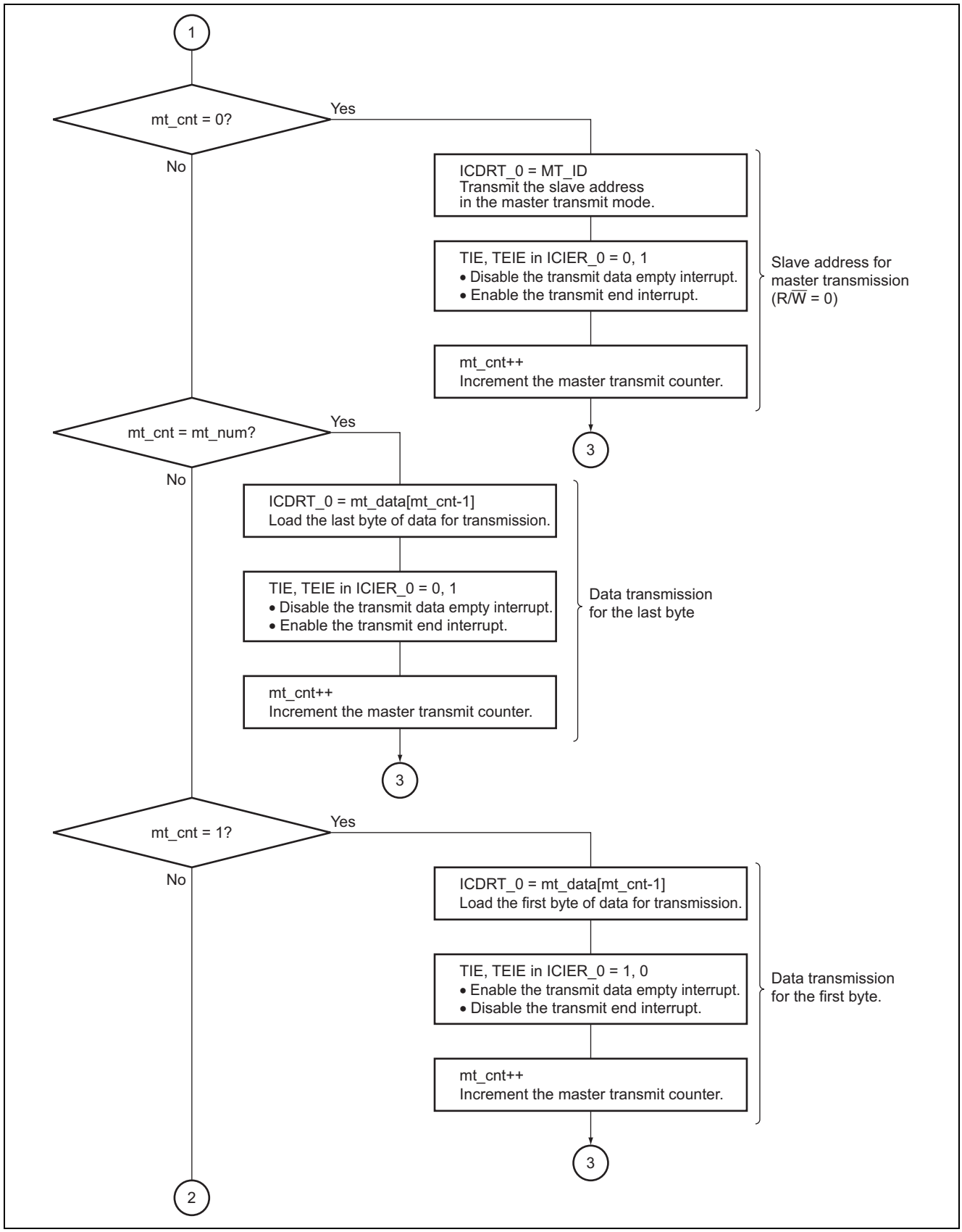
- I²C bus transmit data register_0 (ICDRT_0) Number of bits: 8 Address: H'FFFEB6

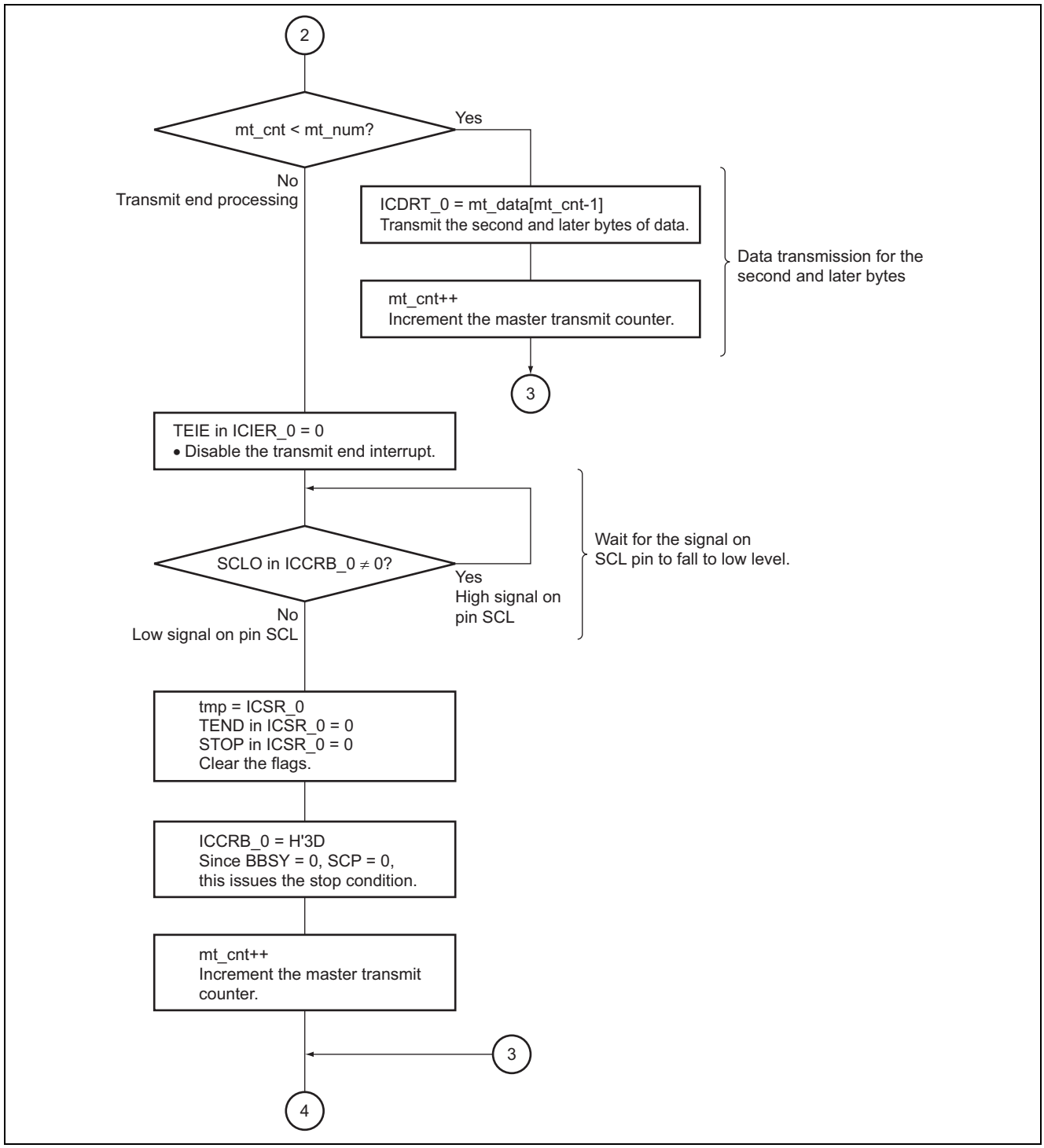
Function: ICDRT is an 8-bit readable/writable register that stores the transmit data. When ICDRT detects the space in the I²C bus shift register (ICDRS), it transfers the transmit data which is written in ICDRT to ICDRS and starts transferring data. When the next transfer data is written to ICDRT during transferring data of ICDRS, continuous transfer is possible. The initial value of ICDRT is H'FF.

Setting: MT_ID, mt_data[mt_cnt-1]

5. Flowchart


master_transfer
AL in ICSR_0 = 1?
Generation of arbitration lost?
No
Yes
Arbitration lost in the master mode
TEIE in ICIER_0 = 0
• Disable the transmit end interrupt.
tmp = ICSR_0
AL, TDRE, TEND in ICSR_0 = 0
Clear the flags.
iic_mode = MODE_IDLE
Set the processing state of this sample task as the idle mode.
alcnt++
Increment the arbitration lost counter.
mt_cnt = 0
Clear the master transmit counter.
NACKF in ICSR_0 = 1?
No
1
Yes
In transmission when NACKF in ICSR_0 = 1, no acknowledgement from receiver device
TIE, TEIE in ICIER_0 = 0
• Disable the transmit data empty interrupt.
• Disable the transmit end interrupt.
SCLO in ICCRB_0 ≠ 0?
Yes
High siglal on pin SCL
No
Low signal on pin SCL
tmp = ICSR_0
NACKF, TDRE, TEND in ICSR_0 = 0
Clear the flags.
ICCRB_0 = H'3D
Since BBSY = 0, SCP = 0,
this issues the stop condition.
mt_cnt++
Increment the master transmit counter.
4
End


1
mt_cnt = 0?
Yes
No
ICDRT_0 = MT_ID
Transmit the slave address
in the master transmit mode.
TIE, TEIE in ICIER_0 = 0, 1
• Disable the transmit data empty interrupt.
• Enable the transmit end interrupt.
mt_cnt++
Increment the master transmit counter.
Slave address for
master transmission
(R/W = 0)
3
mt_cnt = mt_num?
Yes
No
ICDRT_0 = mt_data[mt_cnt-1]
Load the last byte of data for transmission.
TIE, TEIE in ICIER_0 = 0, 1
• Disable the transmit data empty interrupt.
• Enable the transmit end interrupt.
mt_cnt++
Increment the master transmit counter.
Data transmission
for the last byte
3
mt_cnt = 1?
Yes
No
ICDRT_0 = mt_data[mt_cnt-1]
Load the first byte of data for transmission.
TIE, TEIE in ICIER_0 = 1, 0
• Enable the transmit data empty interrupt.
• Disable the transmit end interrupt.
mt_cnt++
Increment the master transmit counter.
Data transmission
for the first byte.
3
2


2
mt_cnt < mt_num?
Yes
No
Transmit end processing
ICDRT_0 = mt_data[mt_cnt-1]
Transmit the second and later bytes of data.
mt_cnt++
Increment the master transmit counter.
Data transmission for the second and later bytes
3
TEIE in ICIER_0 = 0
• Disable the transmit end interrupt.
SCLO in ICCRB_0 ≠ 0?
Yes
High signal on pin SCL
No
Low signal on pin SCL
Wait for the signal on SCL pin to fall to low level.
tmp = ICSR_0
TEND in ICSR_0 = 0
STOP in ICSR_0 = 0
Clear the flags.
ICCRB_0 = H'3D
Since BBSY = 0, SCP = 0,
this issues the stop condition.
mt_cnt++
Increment the master transmit counter.
3
4

5.8.7 master_receive Function

1. Functional overview
   Master-reception processing which is called by the I²C bus interface interrupt handler. In this case, the interrupt source will be the receive data full interrupt for each byte of received data.
2. Argument
   None
3. Return value
   None
4. Description of internal registers used
   The internal registers used in this sample task are described below. The settings shown in these tables are the values used in this sample task and differ from the initial values.

- I²C bus control register A_0 (ICCRA_0) Number of bits: 8 Address: H'FFFEB0

Bit	Bit Name	Setting	R/W	Description
6	RCVD	1	R/W	Reception Disable Enables or disables the next operation when TRS is 0 and ICDRR is read. 0: Enables the next reception. 1: Disables the next reception.
4	TRS	0	R/W	Master/Slave Select 0: Receive mode 1: Transmit mode

- I²C bus control register B_0 (ICCRB_0) Number of bits: 8 Address: H'FFFEB1

Bit	Bit Name	Setting	R/W	Description
7	BBSY	0	R/W	Bus Busy The BBSY flag can be read to check whether the I²C bus (SCL, SDA) is busy or free. In master mode, this bit is also used to issue start and stop conditions. A high-to-low transition of SDA while SCL is high is recognized as a start condition, setting BBSY to 1. A low-to-high transition of SDA while SCL is high is recognized as a stop condition, clearing BBSY to 0. To issue a start condition, use a MOV instruction to write 1 in BBSY and 0 in SCP. A retransmit start condition is issued in the same way. To issue a stop condition, use a MOV instruction to write 0 in BBSY and 0 in SCP.
6	SCP	0	R/W	Start Condition/Stop Condition Prohibit Bit Controls the issuing of start and stop conditions in master mode. Writing 1 in BBSY and 0 in SCP issues a start condition. A retransmit start condition is issued in the same way. Writing 0 in BBSY and SCP issues a stop condition. This bit is always read as 1. Writing of 1 has no effect.
3	SCLO	Undefined	R	Monitors level of signal output from SCL. When the SCLO bit is set to 1 in reading, the signal output from SCL is at high level. When the SCLO bit is set to 0 in reading, the signal output from SCL is at low level.

- I²C bus interrupt enable register_0 (ICIER_0) Number of bits: 8 Address: H'FFFEB3

Bit	Bit Name	Setting	R/W	Description
7	TIE	0	R/W	Transmit Interrupt Enable Enables or disables the transmit data empty interrupt (TXI) when the TDRE bit in ICSR is set to 1 0: Disables the transmit data empty interrupt request (TXI). 1: Enables the transmit data empty interrupt request (TXI).
6	TEIE	0/1	R/W	Transmit End Interrupt Enable Enables or disables the transmit end interrupt (TEI) at the rising of the ninth clock while the TDRE bit in ICSR is 1. TEI can be canceled by clearing the TEND bit or the TEIE bit to 0. 0: Disables the transmit end interrupt request (TEI). 1: Enables the transmit end interrupt request (TEI).
5	RIE	0/1	R/W	Receive Interrupt Enable Enables or disables the receive data full interrupt request (RXI) when a received data is transferred from ICDRS to ICDRR and the RDRF bit in ICSR is set to 1. RXI can be canceled by clearing the RDRF or RIE bit to 0. 0: Disables the receive data full interrupt request (RXI). 1: Enables the receive data full interrupt request (RXI).
0	ACKBT	0/1	R/W	Transmit Acknowledge Specifies the bit to be sent at the acknowledge timing in receive mode. 0: 0 is output at acknowledge timing. 1: 1 is output at acknowledge timing.

- I²C bus status register_0 (ICSR_0) Number of bits: 8 Address: H'FFFEB4

Bit	Bit Name	Setting	R/W	Description
7	TDRE	0	R/W	Transmit Data Register Empty [Setting conditions] • Transferring of data from ICDRT to ICDRS and having ICDRT empty • Setting of TRS • Issuing of a start condition (including retransmission) • Transition from the receive mode to the transmit mode has been made in the slave mode [Clearing conditions] • Writing of 0 in TDRE after reading it as 1 • Writing of data in ICDRT
6	TEND	0	R/W	Transmit End [Setting condition] • Rising of the ninth clock of SCL while the TDRE flag is 1 [Clearing conditions] • Writing of 0 in TEND after reading it as 1 • Writing of data in ICDRT
4	NACKF	0	R/W	No Acknowledge Detection Flag [Setting condition] • Detection of no acknowledge from the receive device in transmission while the ACKE bit in ICIER is 1 [Clearing condition] • Writing of 0 in NACKF after reading it as 1
2	AL	0	R/W	Arbitration Lost Flag Indicates that arbitration was lost in the master mode. When two or more master devices attempt to seize the bus at nearly the same time, when the I²C bus interface detects data differing from the data it sent, it sets AL to 1 to indicate that the bus has been taken by another master. [Setting conditions] • Different values for the internal SDA signal and SDA pin on a rising edge of SCL in master transmit mode • The SDA pin being at the high level in master mode while a start condition is detected [Clearing condition] • Writing of 0 in AL after reading AL as 1

- I²C bus transmit data register_0 (ICDRT_0) Number of bits: 8 Address: H'FFFEB6

Function: ICDRT is an 8-bit readable/writable register that stores the transmit data. When ICDRT detects the space in the I²C bus shift register (ICDRS), it transfers the transmit data which is written in ICDRT to ICDRS and starts transferring data. When the next transfer data is written to ICDRT during transferring data of ICDRS, continuous transfer is possible. The initial value of ICDRT is H'FF.

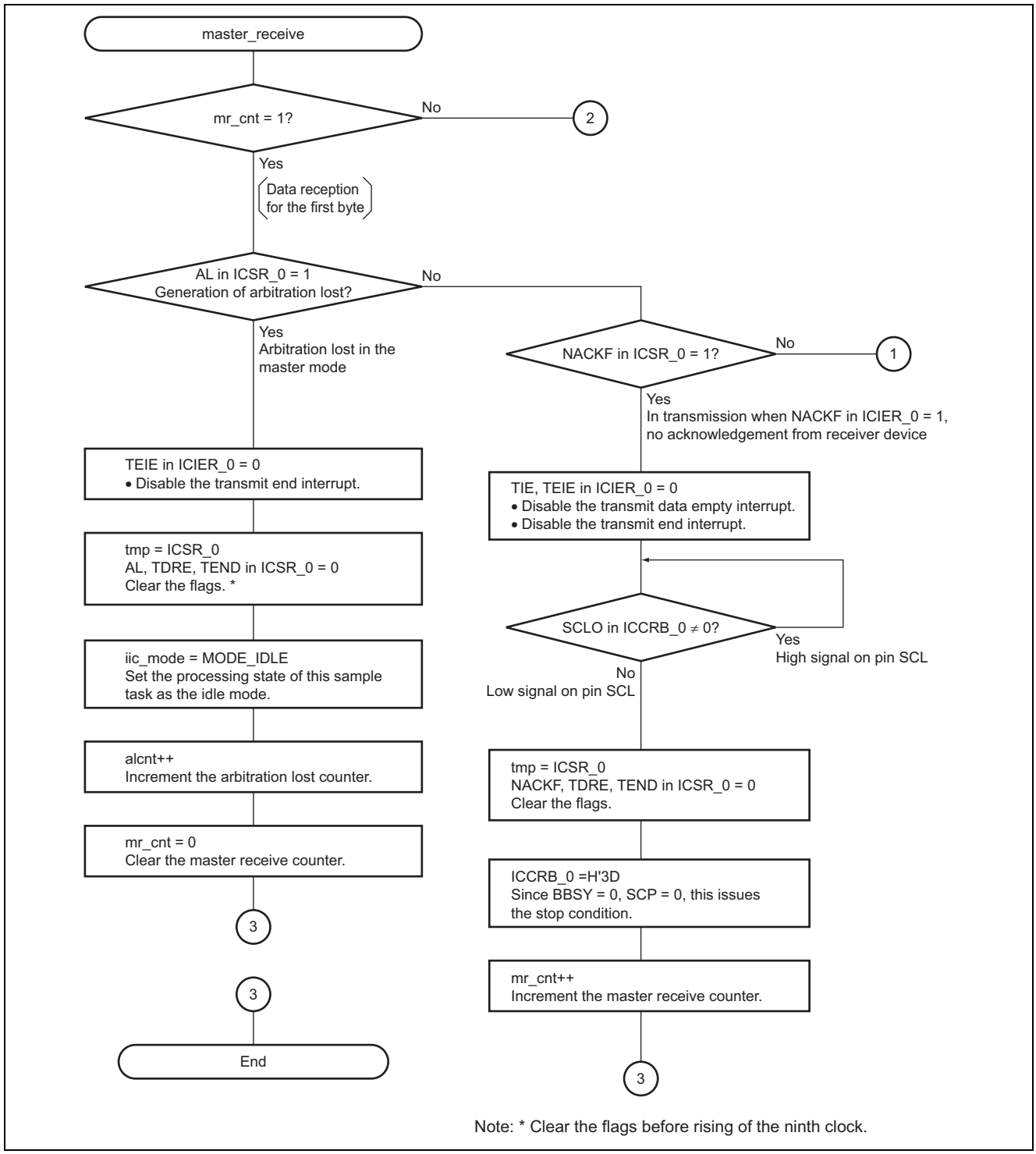
Setting: MR_ID

- I²C bus receive data register_0 (ICDRR_0) Number of bits: 8 Address: H'FFFEB7

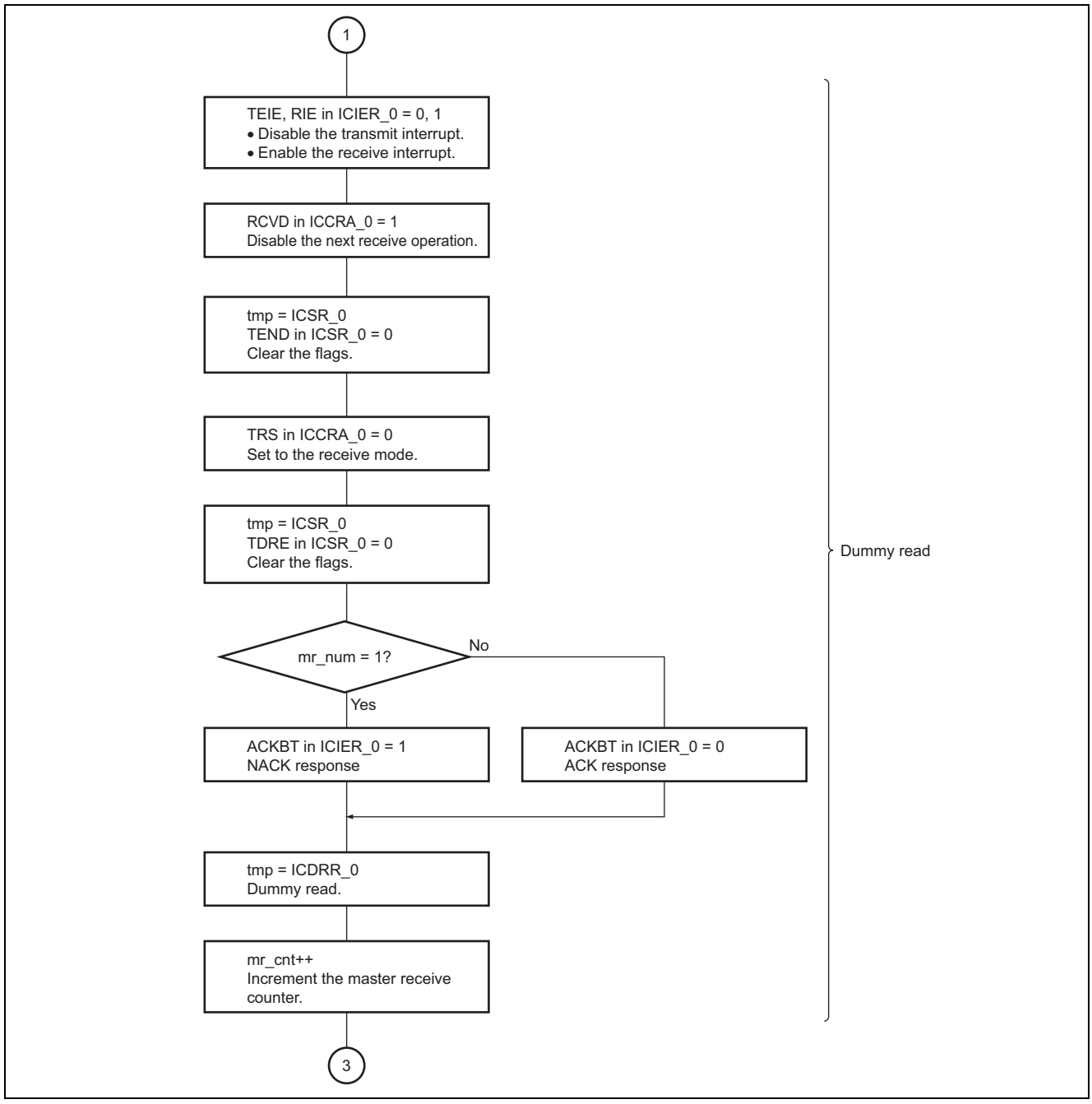
Function: ICDRR is an 8-bit register that stores the receive data. When data of one byte is received, ICDRR transfers the received data from ICDRS to ICDRR and the next data can be received. ICDRR is a receive-only register, therefore the CPU cannot be written to this register. The initial value of ICDRR is H'FF.

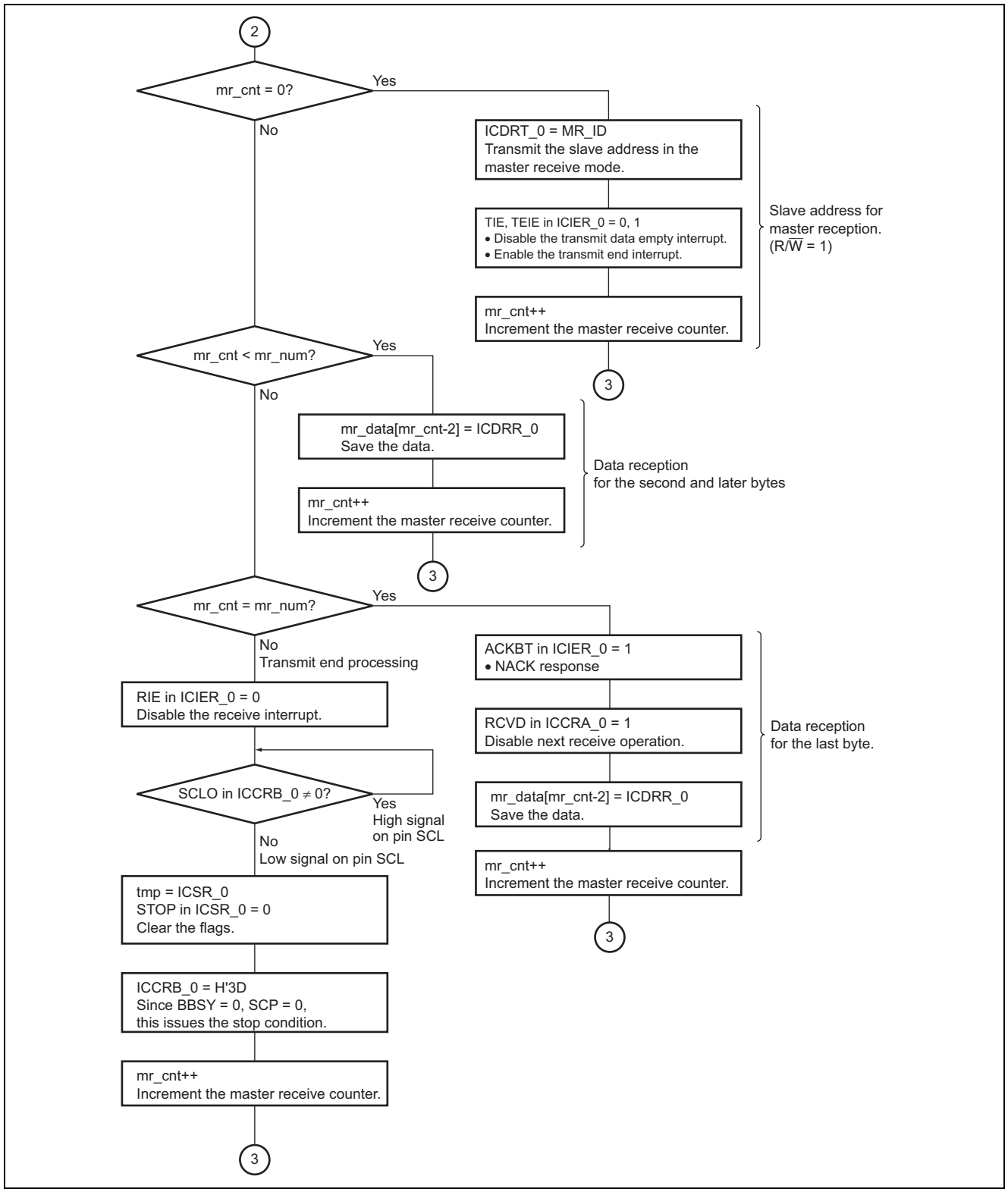
Setting: Undefined

5. Flowchart



Note: * Clear the flags before rising of the ninth clock.


1
TEIE, RIE in ICIER_0 = 0, 1
• Disable the transmit interrupt.
• Enable the receive interrupt.
RCVD in ICCRA_0 = 1
Disable the next receive operation.
tmp = ICSR_0
TEND in ICSR_0 = 0
Clear the flags.
TRS in ICCRA_0 = 0
Set to the receive mode.
tmp = ICSR_0
TDRE in ICSR_0 = 0
Clear the flags.
mr_num = 1?
No
Yes
ACKBT in ICIER_0 = 1
NACK response
ACKBT in ICIER_0 = 0
ACK response
tmp = ICDRR_0
Dummy read.
mr_cnt++
Increment the master receive counter.
3
Dummy read


2
mr_cnt = 0?
Yes
No
ICDRT_0 = MR_ID
Transmit the slave address in the master receive mode.
TIE, TEIE in ICIER_0 = 0, 1
• Disable the transmit data empty interrupt.
• Enable the transmit end interrupt.
mr_cnt++
Increment the master receive counter.
Slave address for master reception. (R/W = 1)
3
mr_cnt < mr_num?
Yes
No
mr_data[mr_cnt-2] = ICDRR_0
Save the data.
mr_cnt++
Increment the master receive counter.
Data reception for the second and later bytes
3
mr_cnt = mr_num?
Yes
No
Transmit end processing
ACKBT in ICIER_0 = 1
• NACK response
RCVD in ICCRA_0 = 1
Disable next receive operation.
mr_data[mr_cnt-2] = ICDRR_0
Save the data.
mr_cnt++
Increment the master receive counter.
Data reception for the last byte.
3
RIE in ICIER_0 = 0
Disable the receive interrupt.
SCLO in ICCRB_0 ≠ 0?
Yes
High signal on pin SCL
No
Low signal on pin SCL
tmp = ICSR_0
STOP in ICSR_0 = 0
Clear the flags.
ICCRB_0 = H'3D
Since BBSY = 0, SCP = 0, this issues the stop condition.
mr_cnt++
Increment the master receive counter.
3

5.8.8 slave_transfer Function

1. Functional overview
   Slave-transmission processing which is called from the I2C bus interface handler. In this case, the interrupt source will be the transmit data empty interrupt for each byte of transmitted data.
2. Argument
   None
3. Return value
   None
4. Description of internal registers used
   The internal registers used in this sample task are described below. The settings shown in these tables are the values used in this sample task and differ from the initial values.

- I2C bus control register A_0 (ICCRA_0) Number of bits: 8 Address: H'FFFEB0

Bit	Bit Name	Setting	R/W	Description
4	TRS	0	R/W	Transmit/Receive Select 0: Receive mode 1: Transmit mode

- I2C bus interrupt enable register_0 (ICIER_0) Number of bits: 8 Address: H'FFFEB3

Bit	Bit Name	Setting	R/W	Description
7	TIE	1	R/W	Transmit Interrupt Enable Enables or disables the transmit data empty interrupt (TXI) when the TDRE bit in ICSR is set to 1. 0: Disables the transmit data empty interrupt request (TXI). 1: Enables the transmit data empty interrupt request (TXI).
6	TEIE	0	R/W	Transmit End Interrupt Enable Enables or disables the transmit end interrupt (TEI) at the rising of the ninth clock while the TDRE bit in ICSR is 1. TEI can be canceled by clearing the TEND bit or the TEIE bit to 0. 0: Disables the transmit end interrupt request (TEI). 1: Enables the transmit end interrupt request (TEI).

- I²C bus status register_0 (ICSR_0) Number of bits: 8 Address: H'FFFEB4

Bit	Bit Name	Setting	R/W	Description
7	TDRE	0	R/W	Transmit Data Register Empty [Setting conditions] • Transferring of data from ICDRT to ICDRS and having ICDRT empty • Setting of TRS • Issuing of a start condition (including retransmission) • Transition from the receive mode to the transmit mode has been made in the slave mode [Clearing conditions] • Writing of 0 in TDRE after reading it as 1 • Writing of data in ICDRT
6	TEND	0	R/W	Transmit End [Setting condition] • Rising of the ninth clock of SCL while the TDRE flag is 1 [Clearing conditions] • Writing 0 in TEND after reading it as 1 • Writing of data in ICDRT
4	NACKF	0	R/W	No Acknowledge Detection Flag [Setting condition] • Detection of no acknowledge from the receive device in transmission while the ACKE bit in ICIER is 1 [Clearing condition] • Writing of 0 in NACKF after reading it as 1

- I²C bus transmit data register_0 (ICDRT_0) Number of bits: 8 Address: H'FFFEB6

Function: ICDRT is an 8-bit readable/writable register that stores the transmit data. When ICDRT detects the space in the I²C bus shift register (ICDRS), it transfers the transmit data which is written in ICDRT to ICDRS and starts transferring data. When the next transfer data is written to ICDRT during transferring data of ICDRS, continuous transfer is possible. The initial value of ICDRT is H'FF.

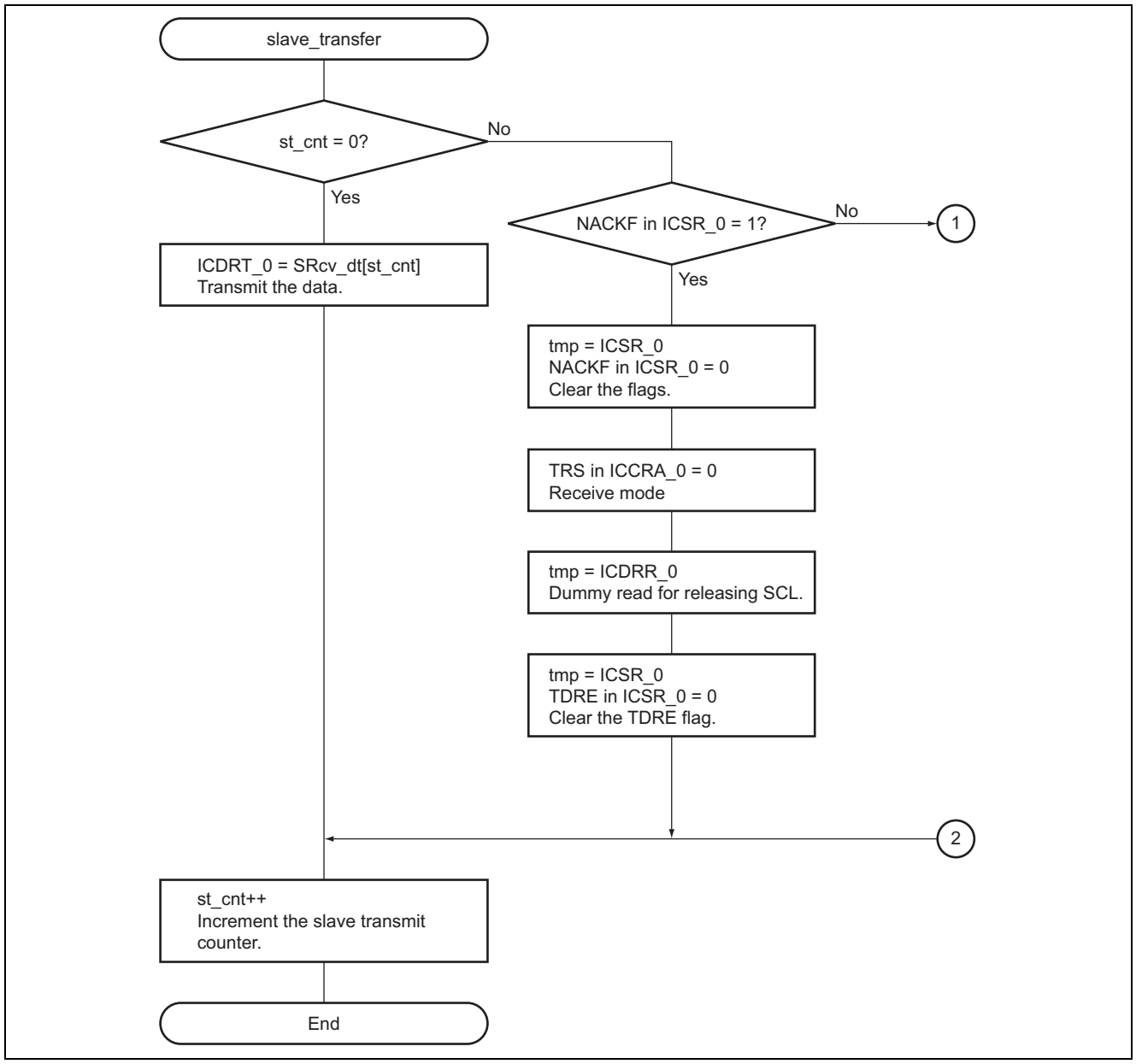
Setting: SRcv_dt[st_cnt]

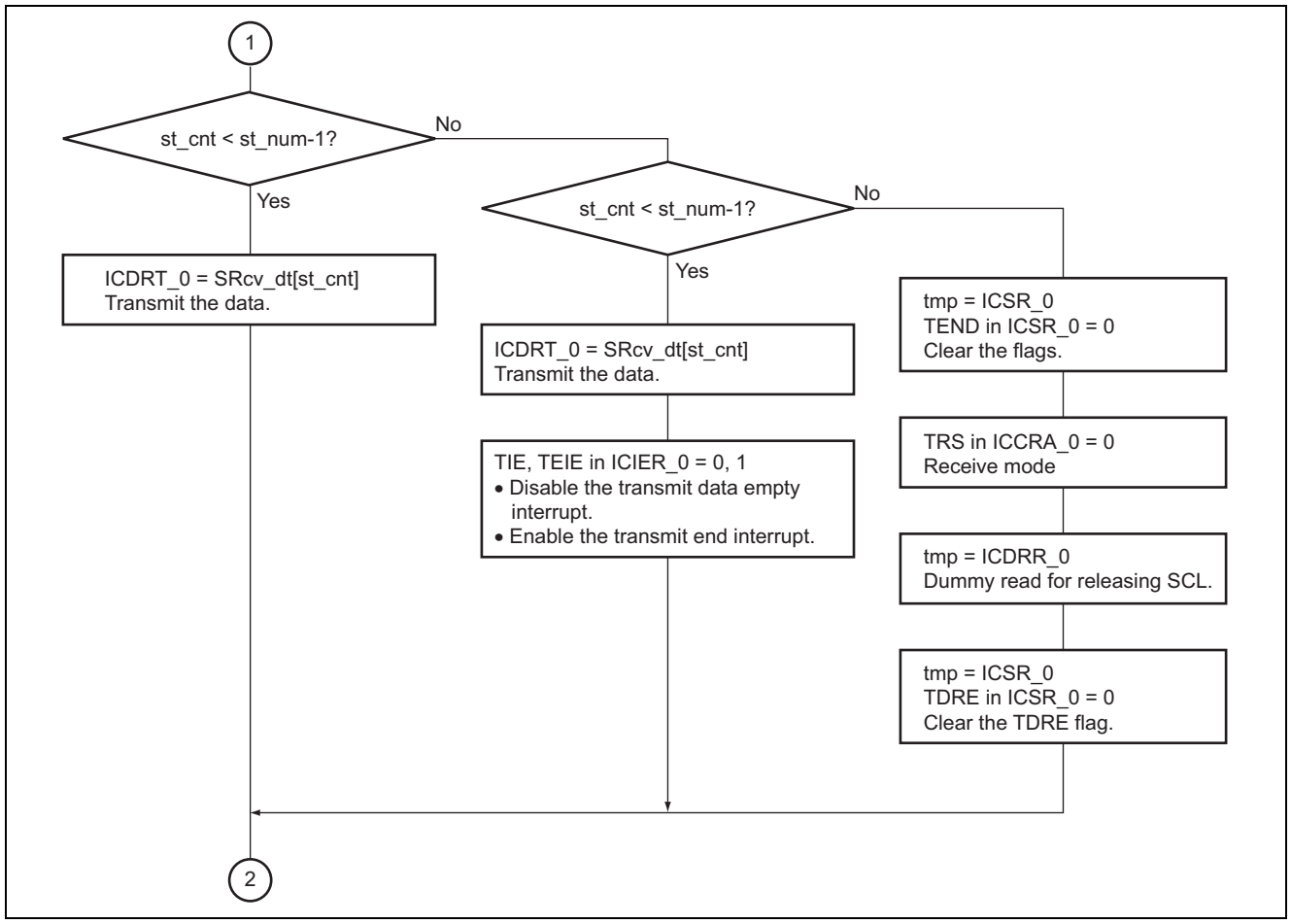
- I²C bus receive data register_0 (ICDRR_0) Number of bits: 8 Address: H'FFFEB7

Function: ICDRR is an 8-bit register that stores the receive data. When data of one byte is received, ICDRR transfers the received data from ICDRS to ICDRR and the next data can be received. ICDRR is a receive-only register, therefore the CPU cannot be written to this register. The initial value of ICDRR is H'FF.

Setting: Undefined

5. Flowchart


slave_transfer
st_cnt = 0?
No
Yes
NACKF in ICSR_0 = 1?
No
1
Yes
ICDRT_0 = SRcv_dt[st_cnt]
Transmit the data.
tmp = ICSR_0
NACKF in ICSR_0 = 0
Clear the flags.
TRS in ICCRA_0 = 0
Receive mode
tmp = ICDRR_0
Dummy read for releasing SCL.
tmp = ICSR_0
TDRE in ICSR_0 = 0
Clear the TDRE flag.
2
st_cnt++
Increment the slave transmit counter.
End


1
st_cnt < st_num-1?
No
Yes
ICDRT_0 = SRcv_dt[st_cnt]
Transmit the data.
st_cnt < st_num-1?
No
Yes
ICDRT_0 = SRcv_dt[st_cnt]
Transmit the data.
TIE, TEIE in ICIER_0 = 0, 1
• Disable the transmit data empty interrupt.
• Enable the transmit end interrupt.
tmp = ICSR_0
TEND in ICSR_0 = 0
Clear the flags.
TRS in ICCRA_0 = 0
Receive mode
tmp = ICDRR_0
Dummy read for releasing SCL.
tmp = ICSR_0
TDRE in ICSR_0 = 0
Clear the TDRE flag.
2

5.8.9 slave_receive Function

1. Functional overview
   Slave-reception processing which is called by the I²C bus interface interrupt handler. In this case, the interrupt source will be the receive data full interrupt for each byte of received data.
2. Argument
   None
3. Return value
   None
4. Description of internal registers used
   The internal registers used in this sample task are described below. The settings shown in these tables are the values used in this sample task and differ from the initial values.

- I²C interrupt enable tegister_0 (ICIER_0) Number of bits: 8 Address: H'FFFEB3

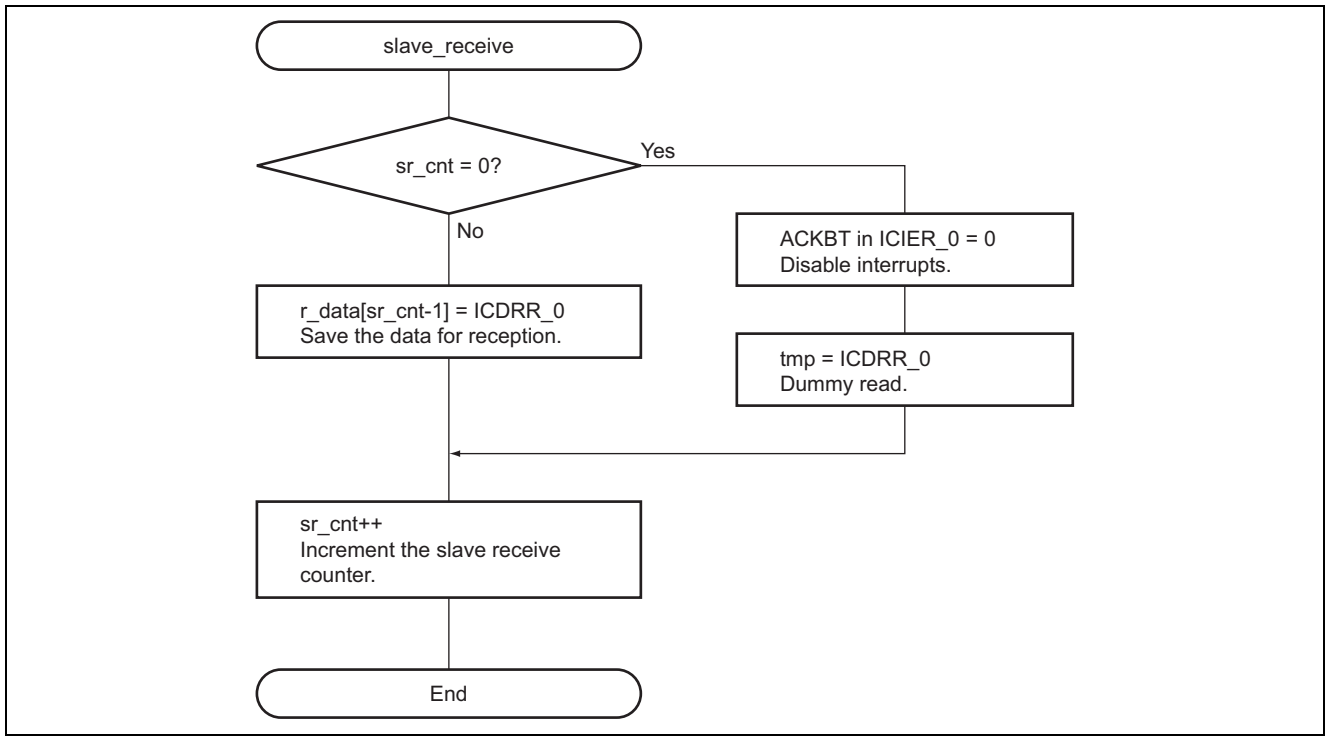
Bit	Bit Name	Setting	R/W	Description
0	ACKBT	0	R/W	Transmit Acknowledge Specifies the bit to be sent at the acknowledge timing in receive mode. 0: 0 is output at acknowledge timing. 1: 1 is output at acknowledge timing.

- I²C bus receive data register_0 (ICDRR_0) Number of bits: 8 Address: H'FFFEB7

Function: ICDRR is an 8-bit register that stores the receive data. When data of one byte is received, ICDRR transfers the received data from ICDRS to ICDRR and the next data can be received. ICDRR is a receive-only register, therefore the CPU cannot be written to this register. The initial value of ICDRR is H'FF.

Setting: Undefined

5. Flowchart


slave_receive
sr_cnt = 0?
Yes
No
ACKBT in ICIER_0 = 0
Disable interrupts.
r_data[sr_cnt-1] = ICDRR_0
Save the data for reception.
tmp = ICDRR_0
Dummy read.
sr_cnt++
Increment the slave receive
counter.
End

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	Sep.25.07	—	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.