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.

H8S Family

Multi-Master Mode Communications Using I^{2}C Bus Interface (IIC)

Introduction

This application note describes the usage of the I^{2}C bus interface (IIC) module in the multi-master mode.

Target Device

H8S/2638

Contents

1. Specifications

- Figure 1 shows the connections for communications using the I²C bus interface 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 100 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 PE2 to PE0 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 P15 and P14 that indicate the state of operation.

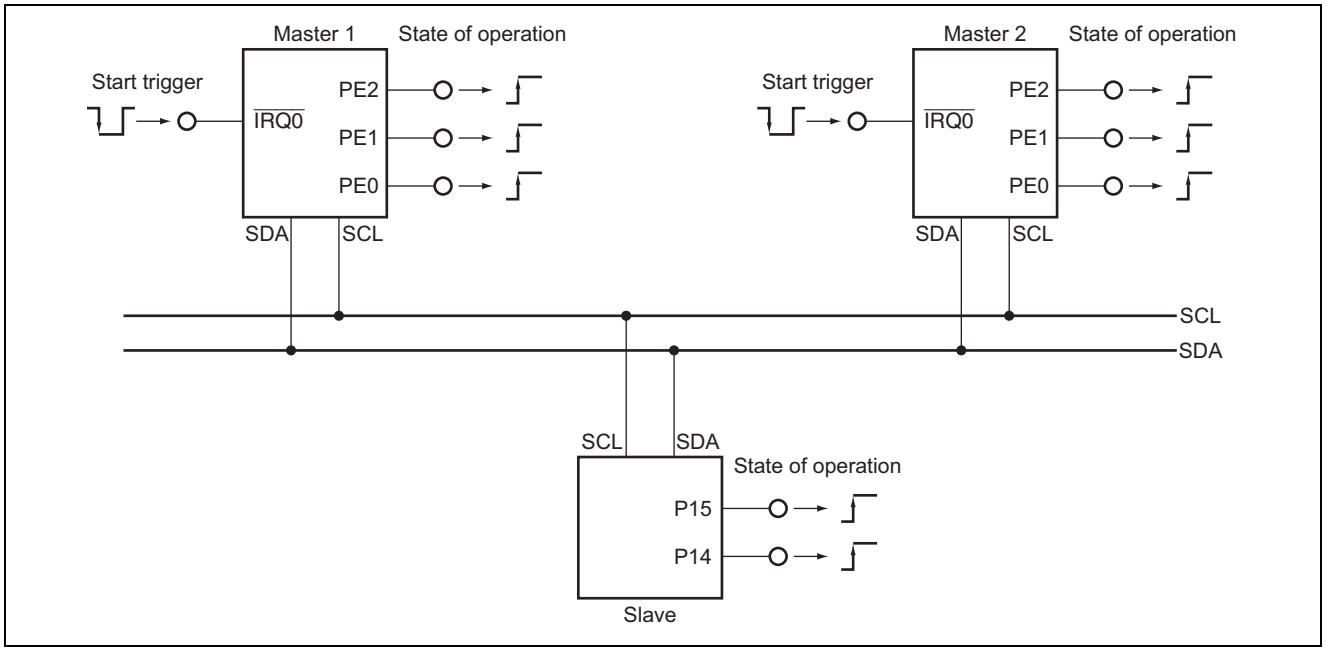


Figure 1 Connections for I²C Bus Interface Multi-Master Mode Communication

Table 1 Slave Addresses

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

2. Applicable Conditions

Table 2 Applicable Conditions

Items	Contents
Operating frequency	Input clock: 20 MHz System clock (ϕ): 20 MHz Peripheral module clock: 20 MHz
Mode of operation	Mode 6 (MD2 = 1, MD1 = 1, MD0 = 0)
Development tools	High-performance Embedded Workshop ver. 4.01.01
C/C++ compiler	Manufactured by Renesas Technology Corp. H8S, H8/300 Series C/C++ Compiler Ver6.01.02
Compiler options	-cpu = 2000a:24, -code = machinecode, -optimize = 1, -regparam = 3, -speed = (register, shift, struct, expression)

Table 3 Section Settings

Address	Section Name	Description
H'001000	P	Program area
	C	Data table
H'FF6000	B	Non-initialized data area (RAM area)

3. Description of Functions

3.1 Description of I²C Bus Interface (IIC)

An I²C bus interface 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. IRQ0 starts the processing of the I²C bus interface communications on the input of a rising 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 PE2 to PE0 Pins

As indicated in table 4, the pins PE2 to PE0 on the master side indicate the state of I²C bus interface communications (reset state or result of operations).

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

PE2	PE1	PE0	State of Operations
0	0	0	Reset
x	0	1	Data mismatch
x	1	0	Data match
1	x	x	Arbitration lost generated.

3.5 Slave Side P15 and P14 Pins

As indicated in table 5, pins P15 and P14 on the slave side indicate the state of I²C bus interface communications (reset state or result of operations).

Table 5 Output Values of Slave Side Pins and State of Operations

P15	P14	State of Operations
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. Description of Operations

4.1 Timing of Operations in Master Transmit Mode

Figure 2 shows the timing of operations of the I²C bus interface in master transmit mode. Table 6 describes processing by hardware and software at the numbered points in figure 2.

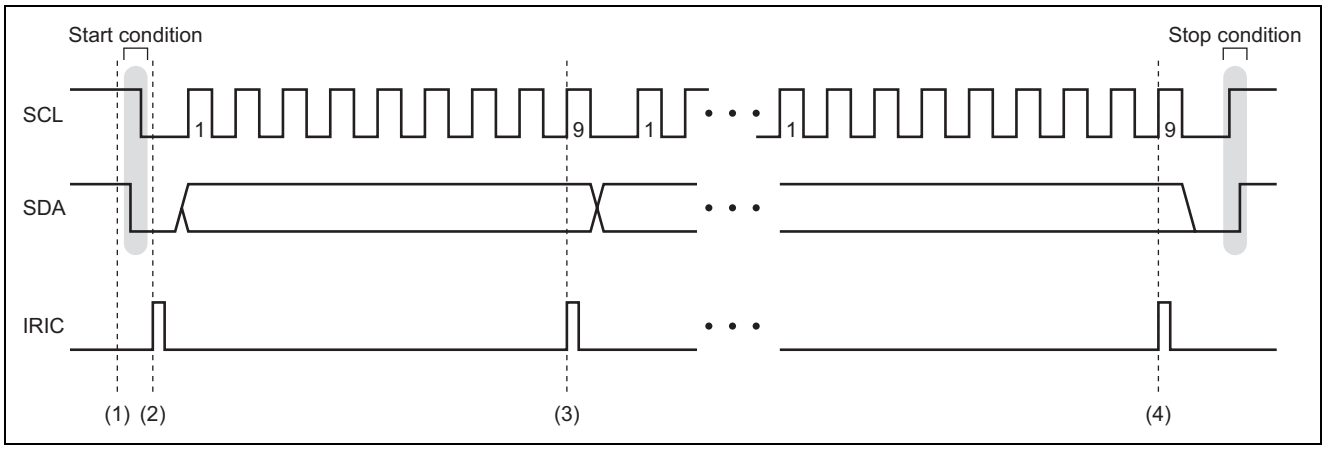


Figure 2 Timing of Operations in Master Transmit Mode

Table 6 Description of Processing

	Hardware Processing	Software Processing
(1)	No processing	a. Set the IEIC bit to 1, enabling I²C bus interface Interrupts. b. Issue the start condition.
(2)	a. I²C bus interface interrupt generation Start condition is detected and the IRIC bit is set to 1	a. Write the slave-side address and data-direction bit (R/$\overline{W}$) to ICDR, then transmit this data. b. Clear the IRIC flag.
(3)	a. Generation of an I²C bus interface interrupt On the rising edge of the ninth cycle of SCL, the IRIC bit is set to 1.	a. Write the data for transmission to ICDR and transmit the data. b. Clear the IRIC flag.
(4)	a. Generation of an I²C bus interface interrupt On the rising edge of the ninth cycle of SCL, the IRIC bit is set to 1.	a. Set the IEIC bit to 0, disabling I²C bus interface interrupts. b. Clear the IRIC flag. c. 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 in master receive mode. Tables 7 and 8 describe processing by hardware and software at the numbered points in figures 3 and 4.

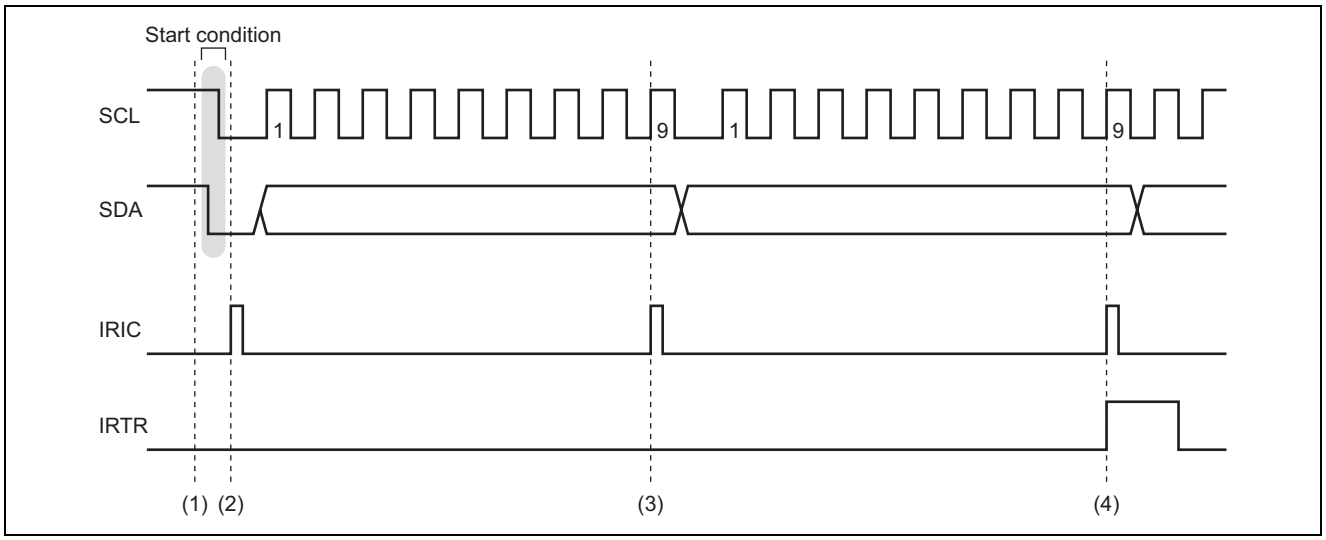


Figure 3 Timing of Operations in Master Receive Mode

Table 7 Description of Processing

	Hardware Processing	Software Processing
(1)	No processing	a. Set the IEIC bit to 1, enabling I²C bus interface interrupts. b. Issue the start condition.
(2)	a. I²C bus interface interrupt generation Start condition is detected and the IRIC bit is set to 1.	a. Write the slave-side address and data-direction bit (R/$\overline{W}$) to ICDR, then transmit this data. b. Clear the IRIC flag.
(3)	a. I²C bus interface interrupt generation On the rising edge of the ninth cycle of SCL, the IRIC bit is set to 1.	a. Set the TRS bit to 0, selecting receive mode. b. Set the ACKB bit to 0 so that 0 is output at the time of acknowledge output. c. Clear the IRIC flag. d. Set the WAIT bit to 1; this inserts a wait between the data bits and the acknowledge bit. e. Execute a dummy read of ICDRR.
(4)	a. I²C bus interface interrupt generation On the rising edge of the ninth cycle of SCL, the IRIC bit is set to 1. b. IRTR is set to 1. When the received data are transferred from ICDRS to ICDRR, both the RDRF and IRTR bits are set to 1.	a. Set the IRTR bit to 1 to read one byte of data from ICDR and store the data in RAM. b. Clear the IRIC flag. (This also clears IRTR.)

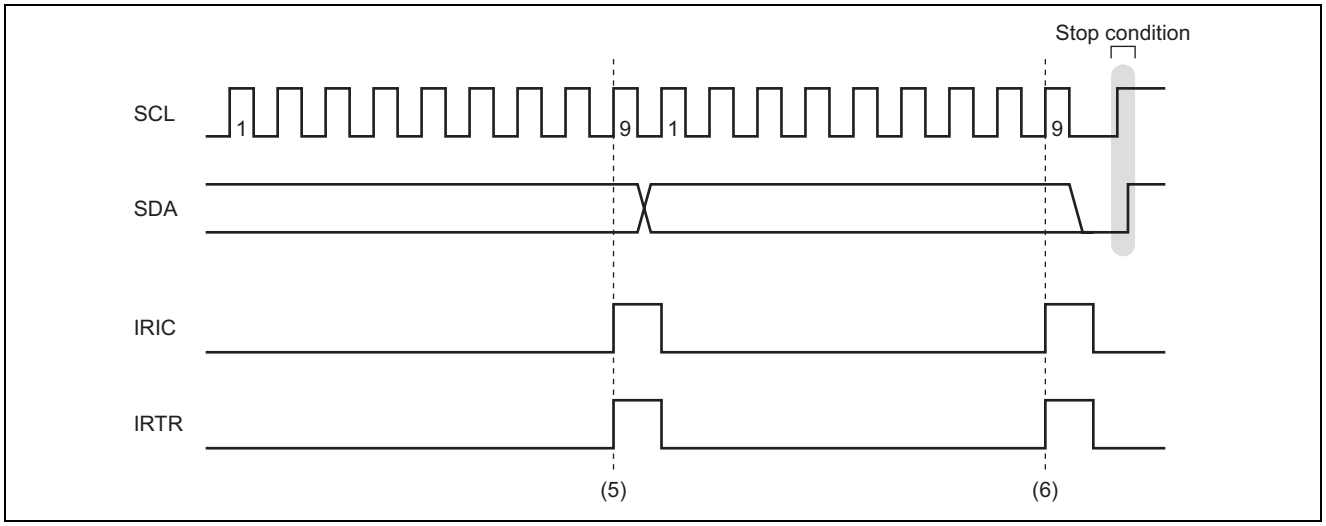


Figure 4 Timing of Operation of Master Receive Mode 2

Table 8 Content of Processing

	Hardware Processing	Software Processing
(5)	a. I²C bus interface interrupt generation On the rising edge of the ninth cycle of SCL, the IRIC bit is set to 1. b. IRTR is set to 1. When the received data are transferred from ICDRS to ICDRR, both the RDRF and IRTR bits are set to 1.	a. Set the ACKB bit to 1, so that 1 is output at the timing of acknowledge output. b. Set the TRS bit to 1, selecting transmit mode so that the stop condition is output. c. Set the IRTR bit to 1 to read one byte of data from ICDR and store the data in RAM. d. Clear the IRIC flag (this also clears IRTR).
(6)	a. I²C bus interface interrupt generation On the rising edge of the ninth cycle of SCL, the IRIC bit is set to 1. b. IRTR is set to 1. When the received data are transferred from ICDRS to ICDRR, both the RDRF and IRTR bits are set to 1.	a. Set the WAIT bit to 0 to transfer the data and acknowledge bits consecutively. b. Clear the IRIC flag (this also clears IRTR). c. Set the IRTR bit to 1 to read the final data from ICDR and store the data in RAM. d. Issue the stop condition.

4.3 Description of Bus Arbitration Operation

The I²C bus interface 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.

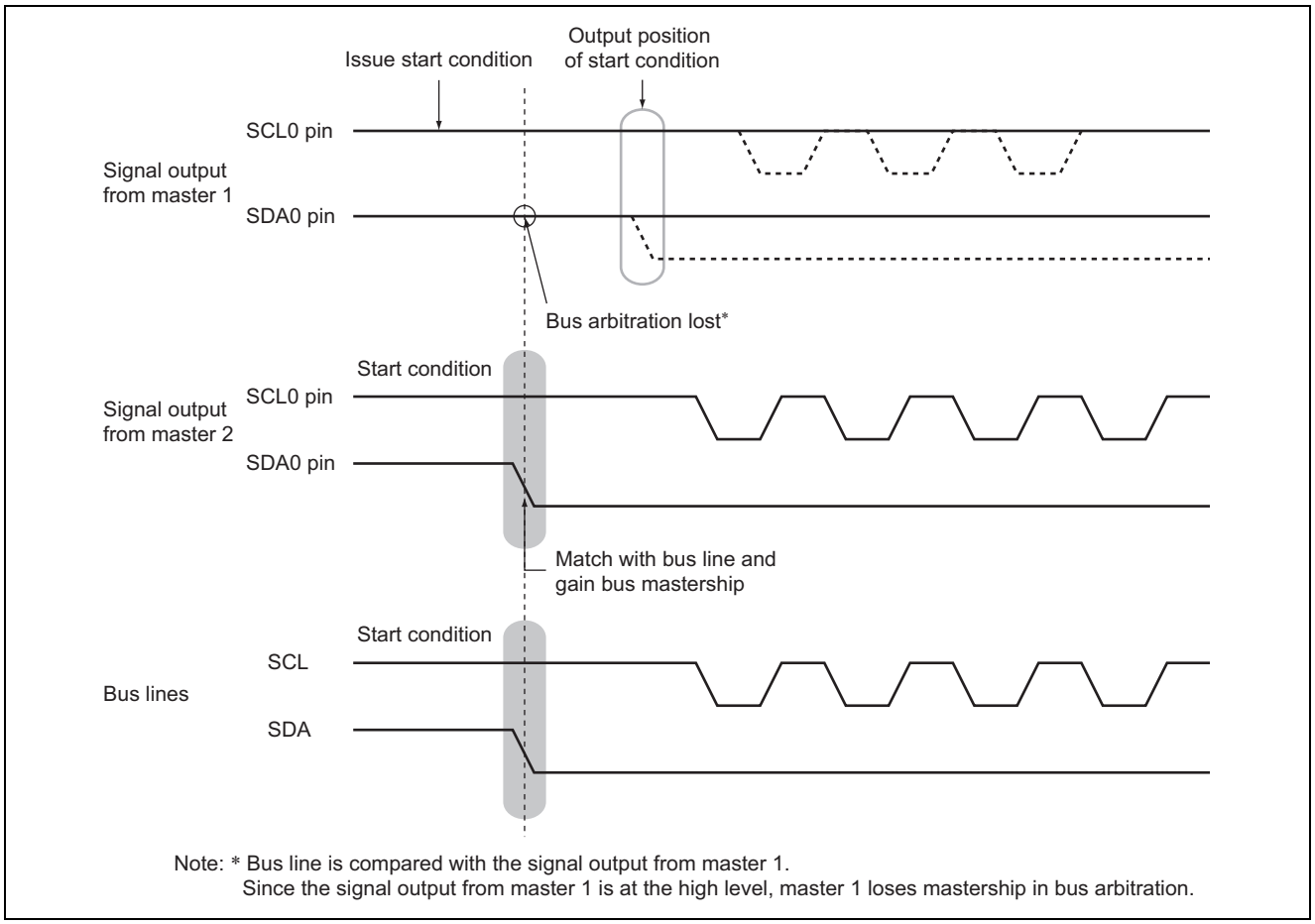


Figure 5 Loss of Bus Arbitration when the Start Condition is Detected

4.3.2 Loss of Bus Arbitration During Data 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 slave receive 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.

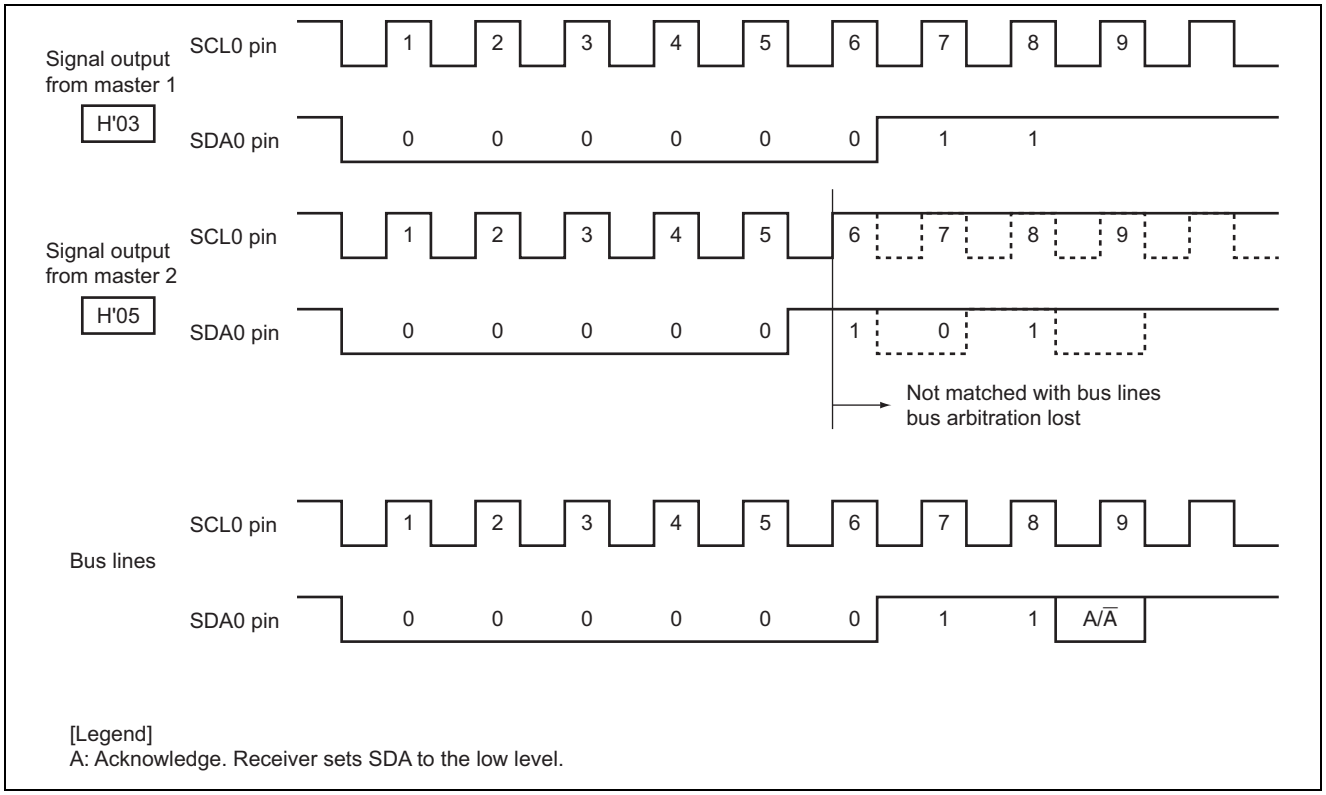
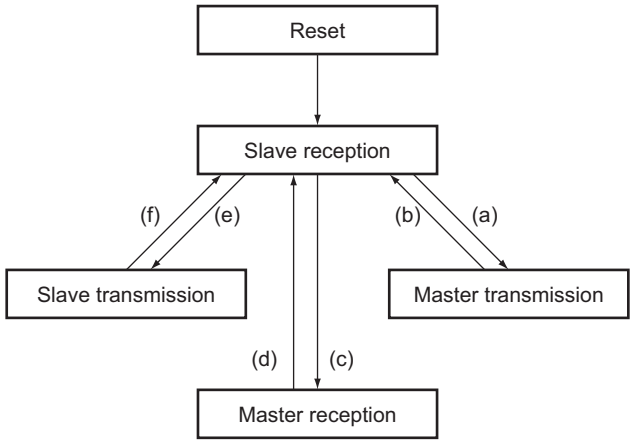


Figure 6 Loss of Bus Arbitration During Data Transmission

4.4 State Transition Diagram

Figure 7 is a state-transition diagram for this sample task. In this sample task, the master transmit mode is selected as the default. After a reset and when the I²C bus interface transfer is in the idle state, the state transition is to the master transmit mode.



(a) Transition initiated by software instructions.
(b) Transition when transmission is complete, arbitration is lost, or an error in communications occurs.
(c) Transition initiated by software instructions.
(d) Transition when reception is complete, arbitration is lost, or an error in communications occurs.
(e) Transition initiated by a slave transmission request from the master side.
(f) Transition when transmission is complete or an error in communications occurs.

Figure 7 State Transition Diagram

5. Description of Software

5.1 List of Functions

Table 9 List of Functions: main.c file

Function	Description
init	Initialization routine Sets the CCR and clock, releases IIC module 0 from module stop mode, and calls function "main".
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 10 List of Functions: iic.c file

Function	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 slave receive 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 slave receive 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.

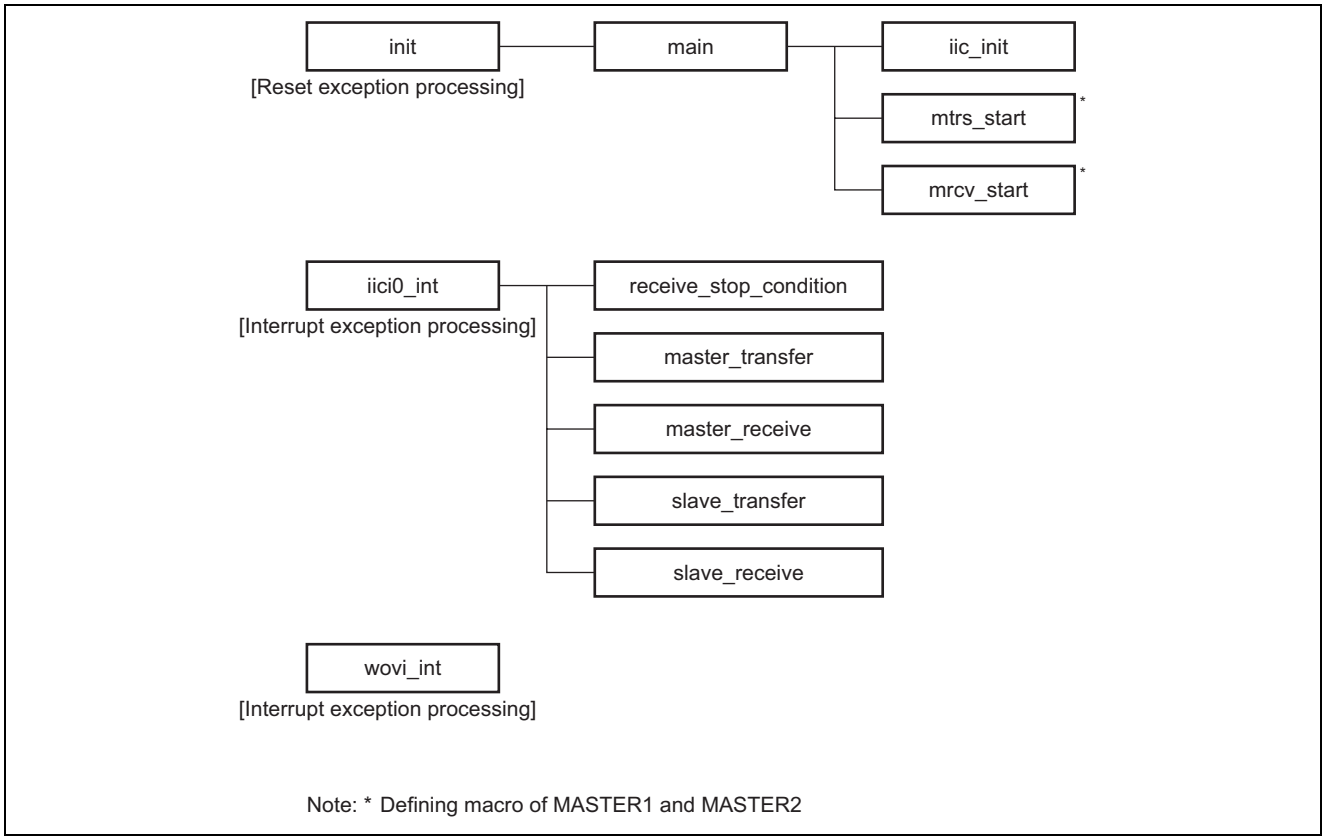


Figure 8 Hierarchy of Calls in the User Program

5.2 Vector Table

Table 11 Exception Handling Vector Table

Origin of Exception	Vector Number	Vector Table Address	Target Function of the Vector
Task "Reset"	0	H'000000	main
WDT interrupt	33	H'000084	wovi_int
IICI0 interrupt	116	H'0001D0	iici0_int

5.3 RAM Usage

Table 12 Description of RAM Usage

Type	Name of Variable	Description	Function Used
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 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 13 Constants

Type	Name of Variable	Setting	Description	Usage in Function
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 14 Macro Definition

Identifier	Description	Usage in Function
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 15 Macro Constants

Name of Variable	Setting	Description	Function Used
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/$\overline{W}$ bit for master transmission Slave-side slave address + 0 (transmission to the slave)	master_transfer
MR_ID	H'07	Slave address + R/$\overline{W}$ bit for master reception Slave-side slave address + 1 (reception from the slave)	master_receive
MODE_MT	3	State of processing of this sample task: Master transmission	mtrs_start iici0_int
MODE_MR	2	State of processing of this sample task: Master reception	mrcv_start iici0_int
MODE_ST	1	State of processing of this sample task: Slave transmission	iici0_int
MODE_SR	0	State of processing of this sample task: Slave reception	iic_init iici0_int receive_stop_condition master_transfer master_receive

5.7 Functions of File main.c

5.7.1 Function init

1. Overview
   This initialization routine releases I²C bus interface 0 from module stop mode, sets the clock, and calls function "main".
2. Arguments
   None
3. Return value
   None
4. Internal register used
   The following describes internal registers used in this sample task. The settings are those used in this sample task rather than the initial settings.

- System Clock Control Register (SCKCR) Address: H'FFFDE6

Bit	Bit Name	Setting	R/W	Function
2	SCK2	0	R/W	System Clock Select 2 to 0
1	SCK1	0	R/W	Selects the bus master clock.
0	SCK0	0	R/W	000: Bus master high-speed mode.

- Low Power Control Register (LPWRCR) Address: H'FFFDEC

Bit	Bit Name	Setting	R/W	Function
1	STC1	0	R/W	System Clock Select 1 to 0
0	STC0	0	R/W	Specify the frequency multiplier for the PLL circuit. 00: x1

- Mode Control Register (MDCR) Address: H'FFFDE7

Bit	Bit Name	Setting	R/W	Function
2	MDS2	—*	R	Mode Select 2 to 0
1	MDS1	—*	R	Indicate the input levels at pins MD2 to MD0 (the current operating mode). Bits MDS2 to MDS0 correspond to pins MD2 to MD0. MDS2 to MDS0 are read-only bits, and cannot be modified. The levels being input on the mode pins (MD2 to MD0) are latched into these bits when MDCR is read. The latching is released by a power-on reset.
0	MDS0	—*	R	

Note: * Determined by the levels on pins MD2 to MD0.

MSTPCRA, MSTPCRB, MSTPCRC, and MSTPCRD control the module stop mode. Setting a bit in these registers places the corresponding module in the module stop mode. Clearing a bit takes the module out of module stop mode.

- Module Stop Control Register A (MSTPCRA) Address: H'FFFDE8

Bit	Bit Name	Setting	R/W	Function
6	MSTPA6	1	R/W	Data transfer controller (DTC)
5	MSTPA5	1	R/W	16-bit timer pulse unit (TPU)
3	MSTPA3	1	R/W	Programmable pulse generator (PPG)
2	MSTPA2	1	R/W	D/A converter (channels 0, 1)
1	MSTPA1	1	R/W	A/D converter

- Module Stop Control Register B (MSTPCRB) Address: H'FFFDE9

Bit	Bit Name	Setting	R/W	Function
7	MSTPB7	1	R/W	Serial communication interface_0 (SCI_0)
6	MSTPB6	1	R/W	Serial communication interface_1 (SCI_1)
5	MSTPB5	1	R/W	Serial communication interface_2 (SCI_2)
4	MSTPB4	0	R/W	I²C bus interface_0 (IIC_0)
3	MSTPB3	1	R/W	I²C bus interface_1 (IIC_1)

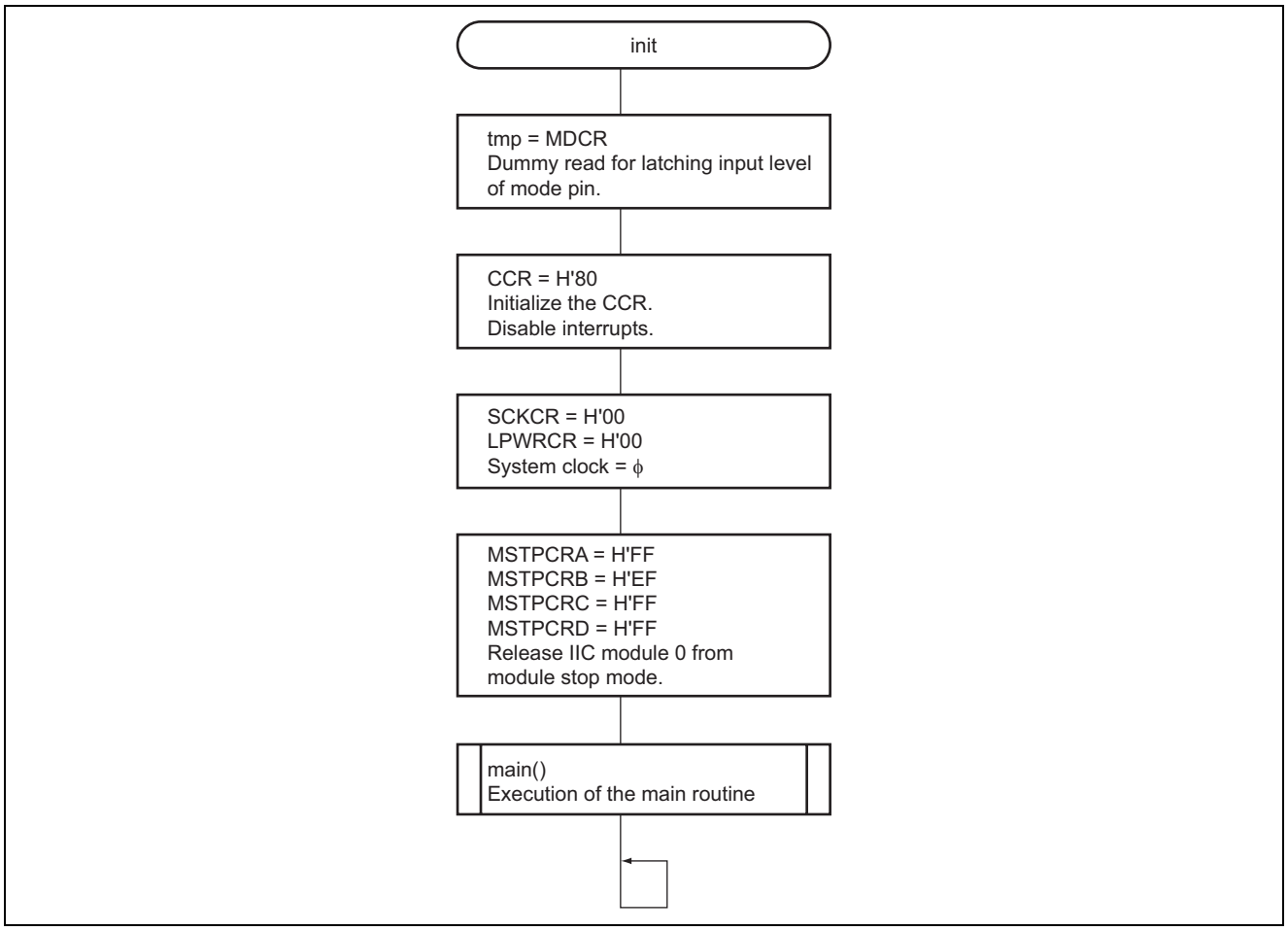
- Module Stop Control Register C (MSTPCRC) Address: H'FFFDEA

Bit	Bit Name	Setting	R/W	Function
4	MSTPC4	1	R/W	PC break controller (PBC)
3	MSTPC3	1	R/W	HCAN0
2	MSTPC2	1	R/W	HCAN1

- Module Stop Control Register D (MSTPCRD) Address: H'FFFC60

Bit	Bit Name	Setting	R/W	Function
7	MSTPD7	1	R/W	Motor control PWM (PWM)

5. Flow Chart


init
tmp = MDCR
Dummy read for latching input level of mode pin.
CCR = H'80
Initialize the CCR.
Disable interrupts.
SCKCR = H'00
LPWRCR = H'00
System clock = ϕ
MSTPCRA = H'FF
MSTPCRB = H'EF
MSTPCRC = H'FF
MSTPCRD = H'FF
Release IIC module 0 from module stop mode.
main()
Execution of the main routine

5.7.2 Function main (Defining Macro of MASTER1 and MASTER2)

1. 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 PE2 to PE0.
2. Arguments
   None
3. Return value
   None
4. Internal registers used
   The following describes internal registers used in this sample task. The settings are those used for this sample task, and are not initial settings.

- IRQ0 Sense Control Register (ISCR) Address: H'FFFE12

Bit	Bit Name	Setting	R/W	Function
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

- IRQ0 Status Register (ISR) Address: H'FFFE15

Bit	Bit Name	Setting	R/W	Function
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 E Data Direction Register (PEDDR) Address: H'FFFF3D

Bit	Bit Name	Setting	R/W	Function
2	PE2DDR	1	R/W	0: Sets the PE2 pin as input pin. 1: Sets the PE2 pin as output pin.
1	PE1DDR	1	R/W	0: Sets the PE1 pin as an input pin. 1: Sets the PE1 pin as an output pin.
0	PE0DDR	1	R/W	0: Sets the PE0 pin as an input pin. 1: Sets the PE0 pin as an output pin.

- Port E Data Register (PEDR) Address: H'FFFF0D

Bit	Bit Name	Setting	R/W	Function
2	PE2DR	0/1	R/W	0: PE2 pin is set to the low level. 1: PE2 pin is set to the high level.
1	PE1DR	0/1	R/W	0: PE1 pin is set to the low level. 1: PE1 pin is set to the high level.
0	PE0DR	0/1	R/W	0: PE0 pin is set to the low level. 1: PE0 pin is set to the high level.

- Timer Control/Status Register_0 (TCSR_0) Address: H'FFFF74

Bit	Bit Name	Setting	R/W	Function
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 Select clocks for input to TCNT. 110: Clock Pϕ/32768. When Pϕ is 20 MHz, overflow period is 419.4 ms.
1	CKS1	1	R/W	
0	CKS0	0	R/W	

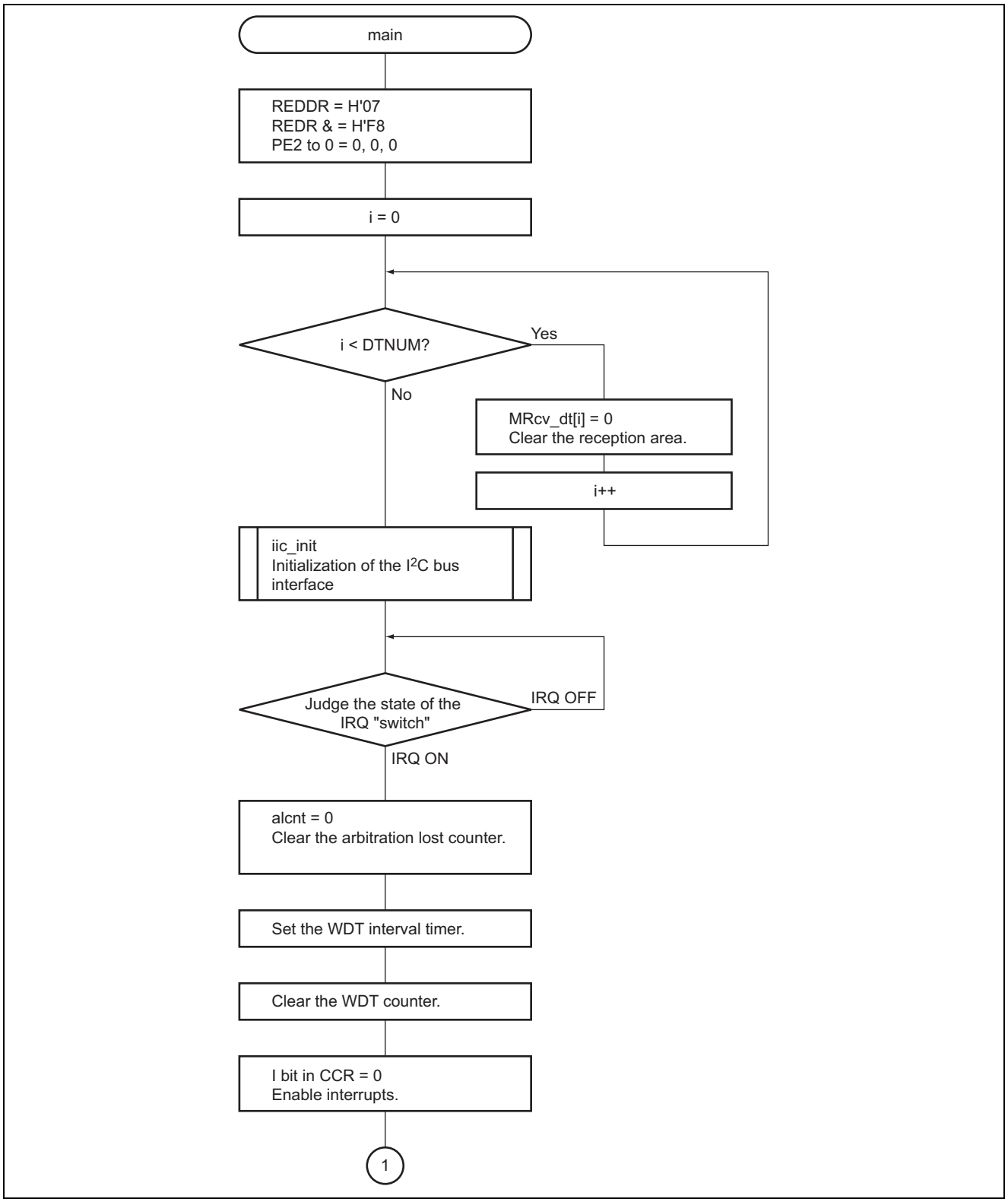
- •Timer Counter_0 (TCNT_0) Address: H'FFFF74 (writing), H'FFFF75 (reading)
  This bit is an 8-bit readable and writable up-counter.
  Setting: H'00

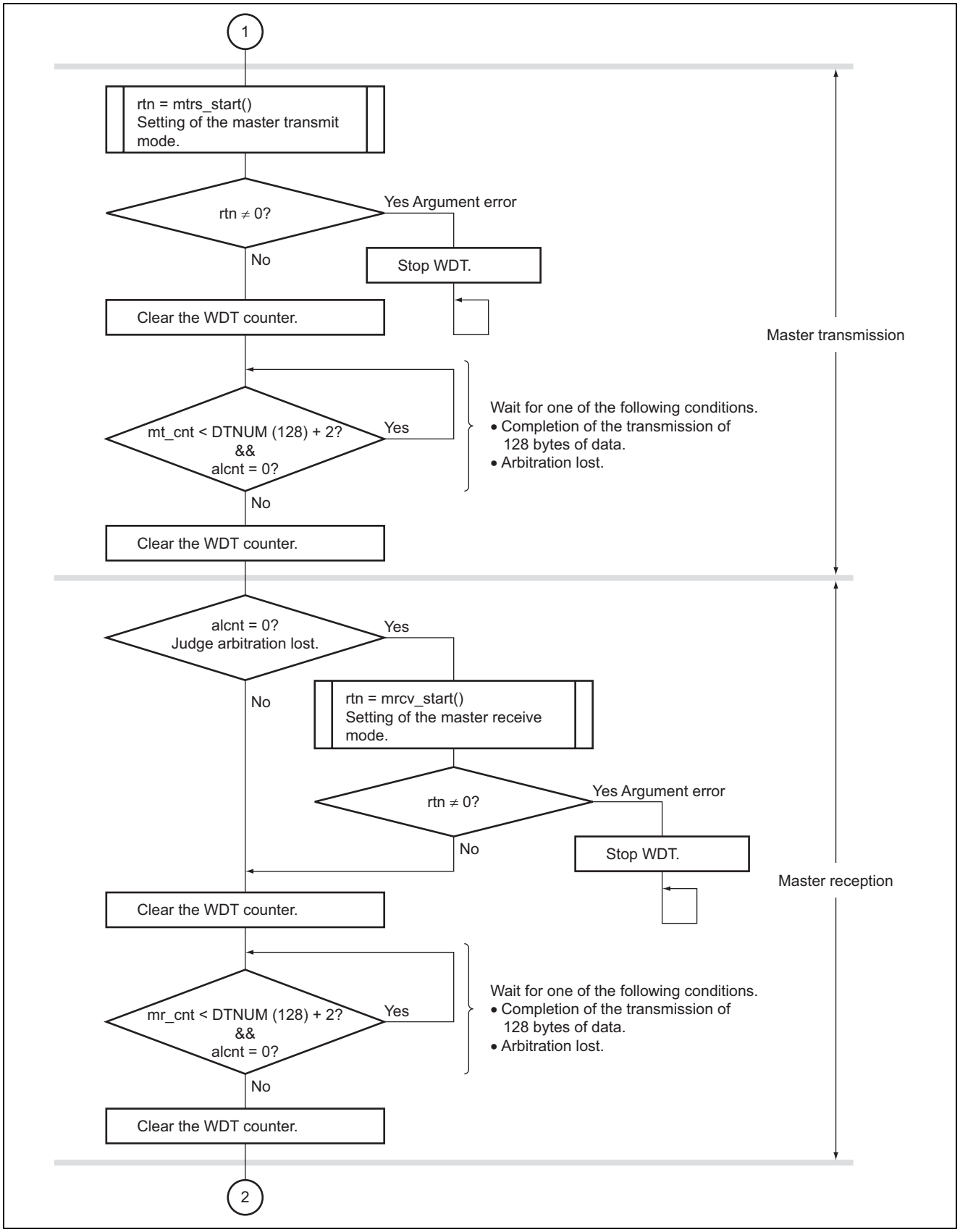
- I²C Bus Control Register_0 (ICCR_0) Address: H'FFFF78*1

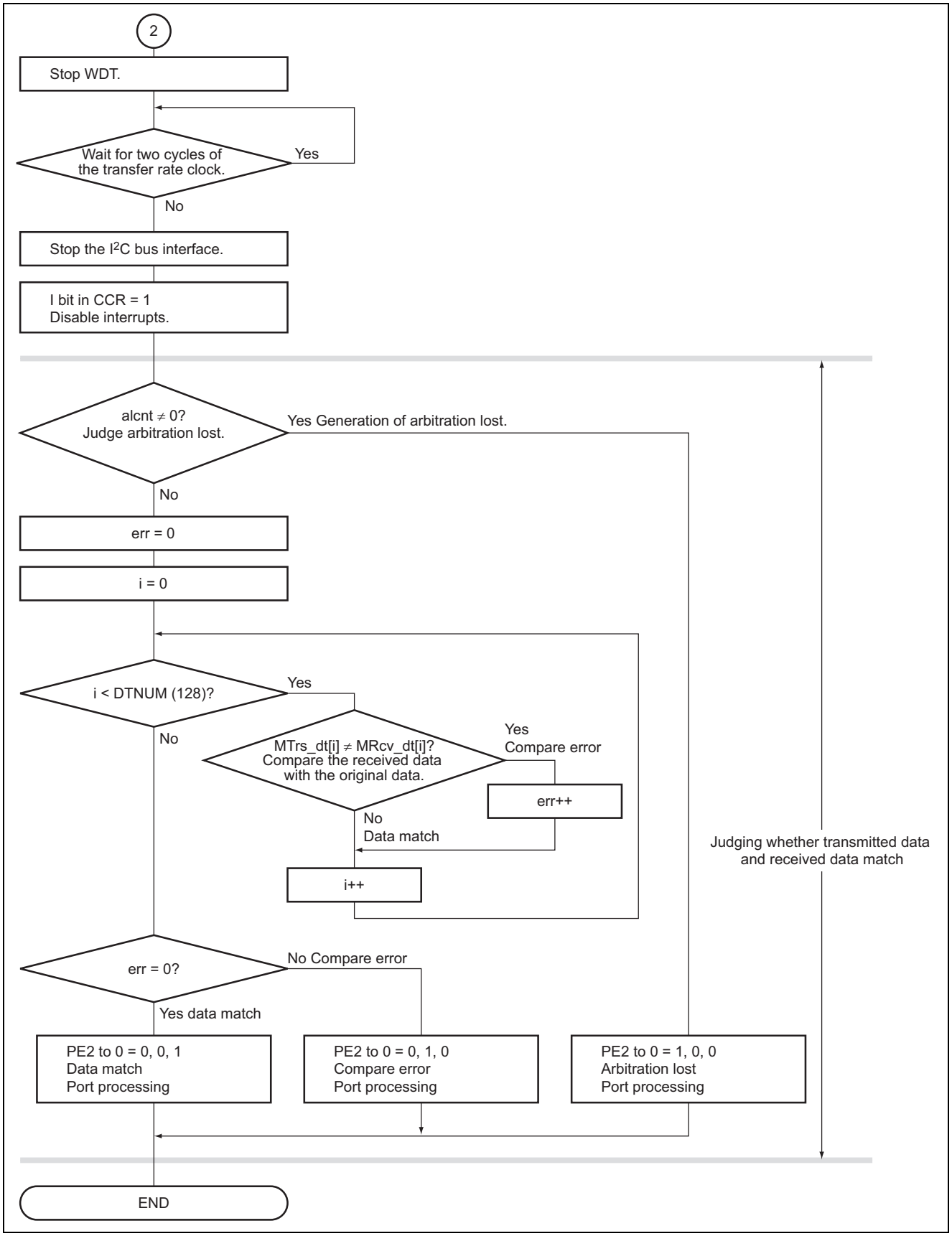
Bit	Bit Name	Setting	R/W	Function
7	ICE	0	R/W	I²C Bus Interface Enable 0: Disables the IIC module and initializes its internal state. SAR and SARX can be accessed. 1: Enables transfer via the IIC module (pins SCL and SDA are driving the bus). ICMR and ICDR can be accessed.

Note: 1. Only accessible when the ICE bit in ICCR_0 is set to 1.

5. Flow Chart


main
REDDR = H'07
REDR & = H'F8
PE2 to 0 = 0, 0, 0
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
No
Stop WDT.
Clear the WDT counter.
Master transmission
mt_cnt < DTNUM (128) + 2?
&&
alcnt = 0?
Yes
Wait for one of the following conditions.
• Completion of the transmission of 128 bytes of data.
• Arbitration lost.
No
Clear the WDT 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.
mr_cnt < DTNUM (128) + 2?
&&
alcnt = 0?
Yes
Wait for one of the following conditions.
• Completion of the transmission of 128 bytes of data.
• Arbitration lost.
No
Clear the WDT counter.
2


2
Stop WDT.
Wait for two cycles of the transfer rate clock.
Yes
No
Stop the I²C bus interface.
I bit in CCR = 1
Disable interrupts.
alcnt ≠ 0?
Judge arbitration lost.
Yes Generation of arbitration lost.
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 0 = 0, 0, 1
Data match
Port processing
PE2 to 0 = 0, 1, 0
Compare error
Port processing
PE2 to 0 = 1, 0, 0
Arbitration lost
Port processing
END

5.7.3 Function main (Defining Macro of SLAVE)

1. 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 P14 = 1. When the address is H'82, outputs P15 = 1.
2. Arguments
   None
3. Return value
   None
4. Internal registers used
   The following describes internal registers used in this sample task. The settings are those used for this sample task, and are not initial settings.

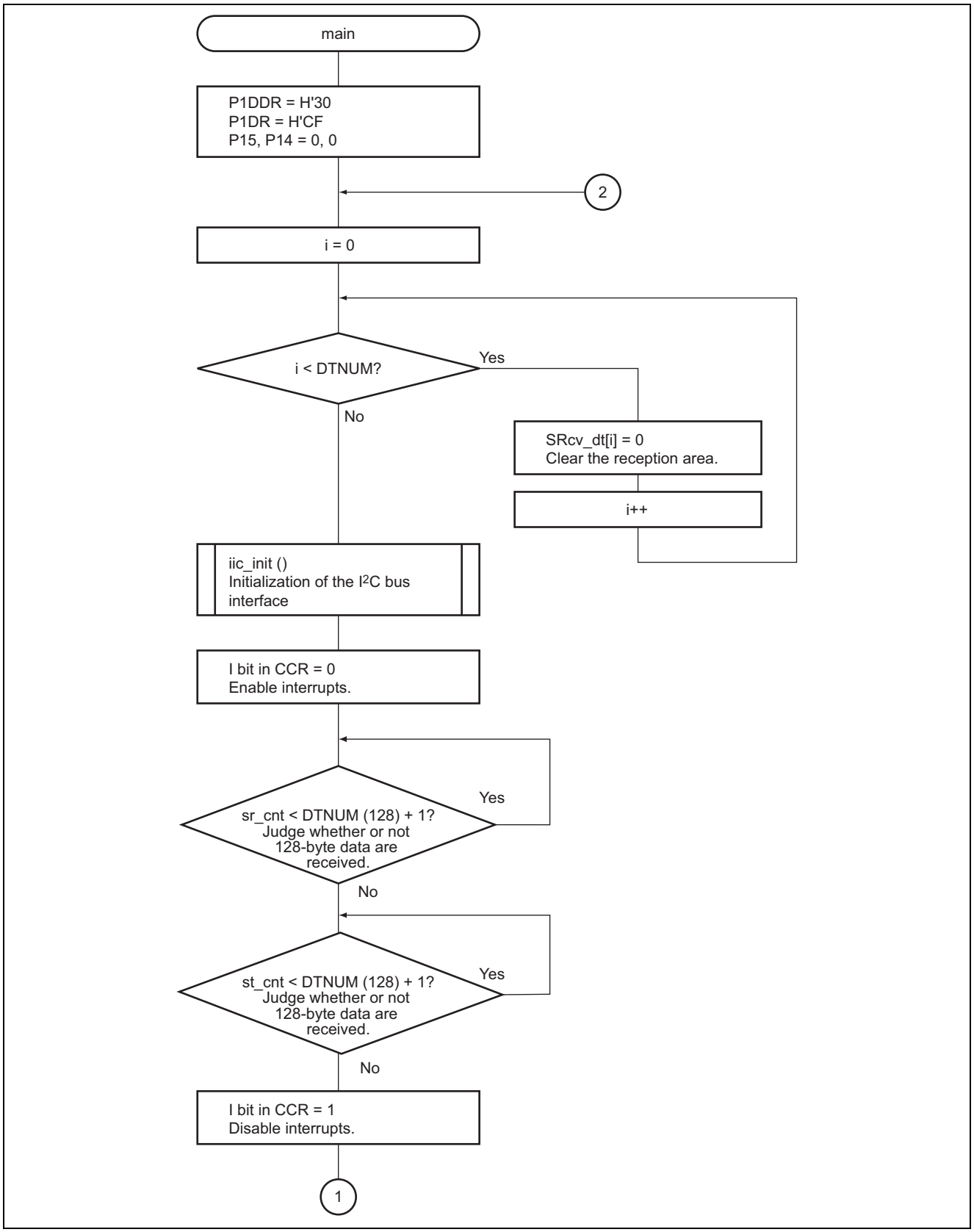
- Port E Data Direction Register (PEDDR) Address: H'FFFE3D

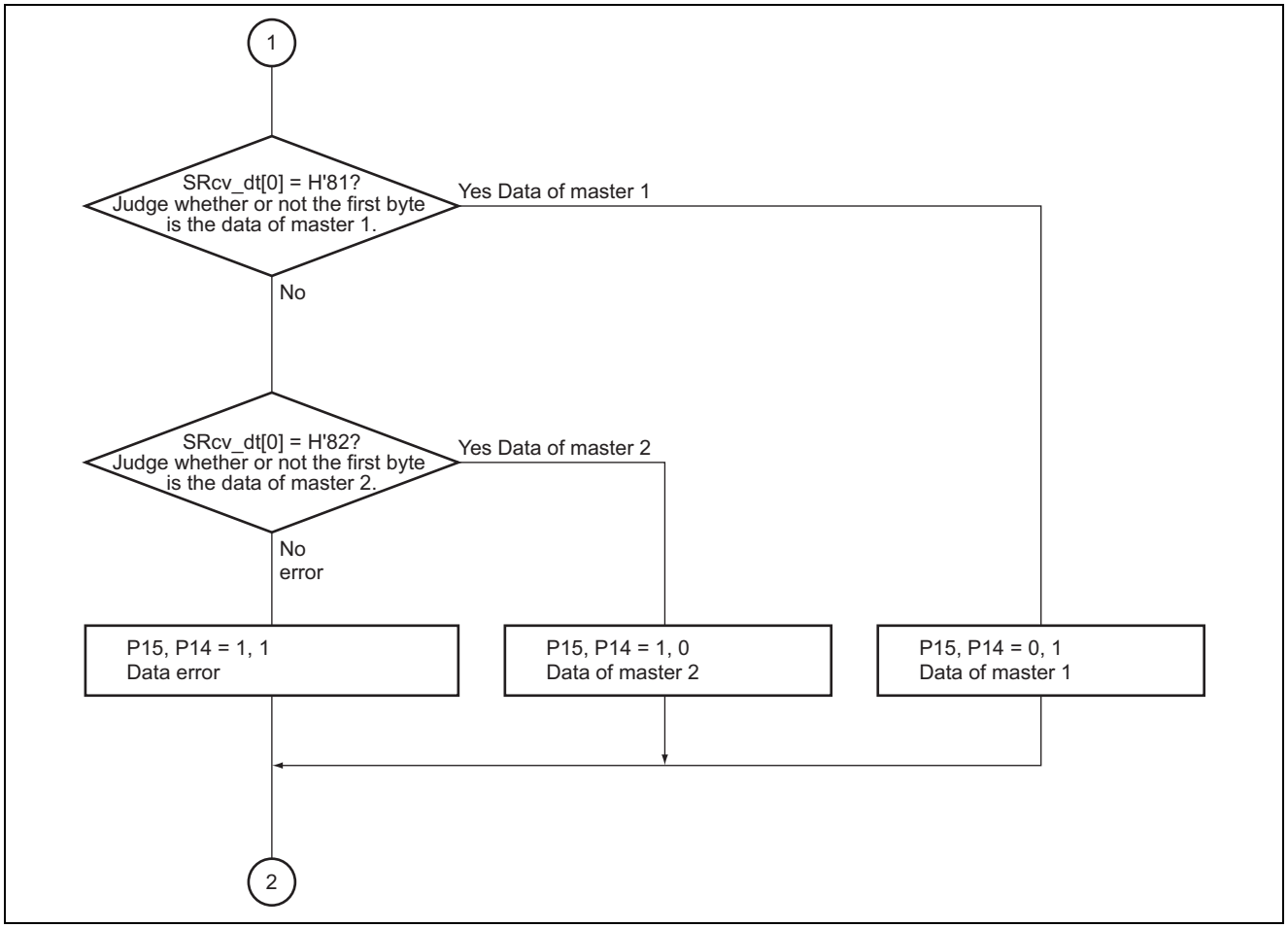
Bit	Bit Name	Setting	R/W	Function
5	PE5DDR	1	R/W	0: Sets the PE5 pin as input pin. 1: Sets the PE5 pin as output pin.
4	PE4DDR	1	R/W	0: Sets the PE4 pin as input pin. 1: Sets the PE4 pin as output pin.

- Port E Data Register (PEDR) Address: H'FFFF0D

Bit	Bit Name	Setting	R/W	Function
5	PE5DR	0/1	R/W	0: PE5 pin is set to the low level. 1: PE5 pin is set to the high level.
4	PE4DR	0/1	R/W	0: PE4 pin is set to the low level. 1: PE4 pin is set to the high level.

5. Flowchart


main
P1DDR = H'30
P1DR = H'CF
P15, P14 = 0, 0
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 received.
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
P15, P14 = 1, 1
Data error
P15, P14 = 1, 0
Data of master 2
P15, P14 = 0, 1
Data of master 1
2

5.7.4 Function wovi_init

1. 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. Arguments
   None
3. Return value
   None
4. Internal registers used
   The following describes internal registers used in this sample task. The settings are those used for this sample task, and are not initial settings.

- I²C Bus Control Register_0 (ICCR_0) Address: H'FFFF78[*1]

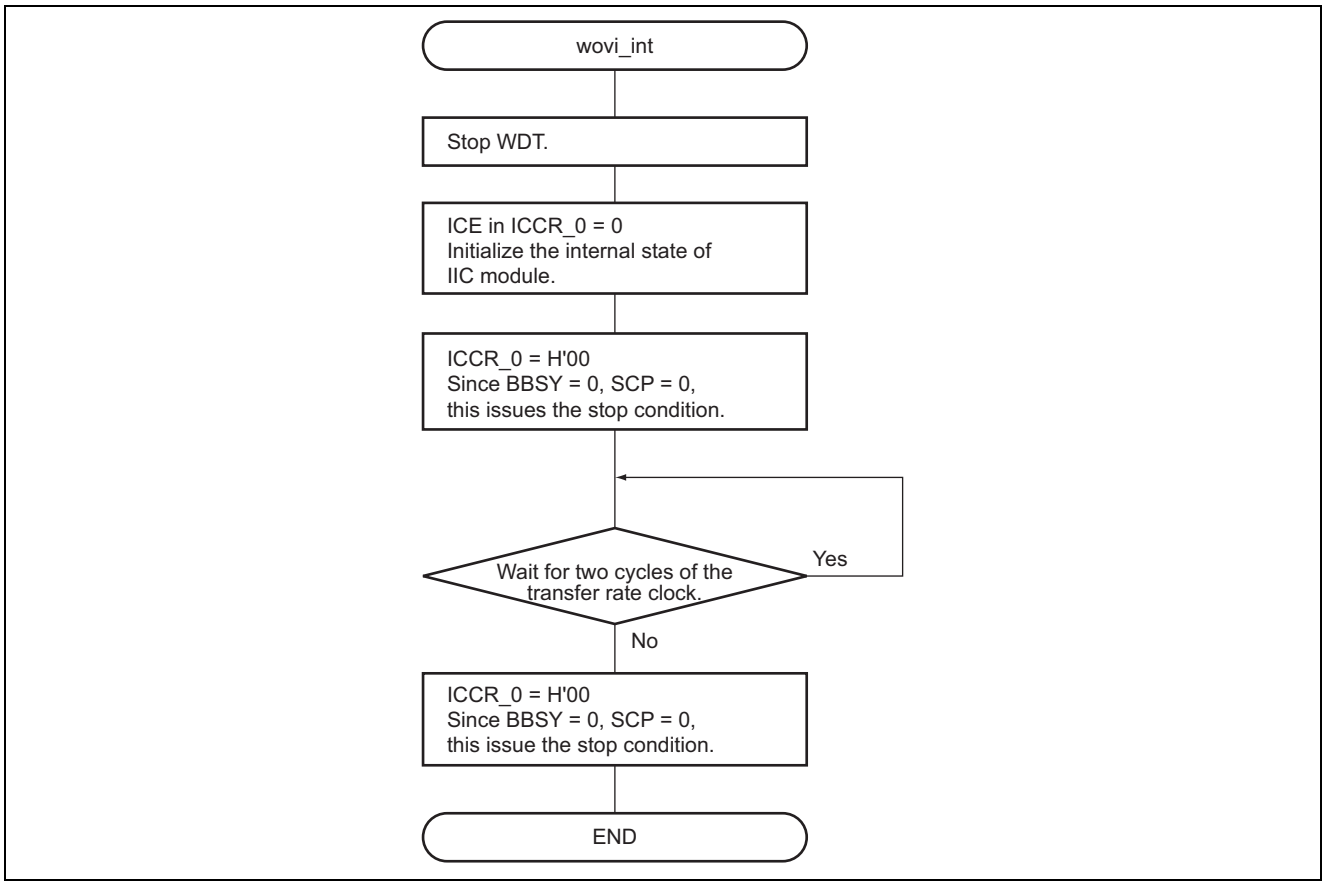
Bit	Bit Name	Setting	R/W	Function
7	ICE	0	R/W	I²C Bus Interface Enable 0: Disables the IIC module and initializes its internal state. SAR and SARX can be accessed. 1: Enables transfer via the IIC module (pins SCL and SDA are driving the bus). ICMR and ICDR can be accessed.
2	BBSY	0	R/W	Bus Busy When BBSY and SCP bits are set to 0, issues stop conditions.
0	SCP	0	W	Start Condition/Stop Condition Prohibit Bit When BBSY and SCP bits are set to 0, issues stop conditions.

Note: 1. Only accessible when the ICE bit in ICCR_0 is set to 1.

- Timer Control/Status Register_0 (TCSR_0) Address: H'FFFF74

Bit	Bit Name	Setting	R/W	Function
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 in ICCR_0 = 0
Initialize the internal state of
IIC module.
ICCR_0 = H'00
Since BBSY = 0, SCP = 0,
this issues the stop condition.
Wait for two cycles of the
transfer rate clock.
Yes
No
ICCR_0 = H'00
Since BBSY = 0, SCP = 0,
this issue the stop condition.
END

5.8 Functions of File iic.c

5.8.1 Function iic_init

1. Overview
   I²C bus interface initialization routine
2. Arguments
   None
3. Return value
   None
4. Internal registers used
   The following describes internal registers used in this sample task. The settings are those used for this sample task, and are not initial settings.

- Serial Control Register X (SCRX) Address: H'FFFDB4

Bit	Bit Name	Setting	R/W	Function
5	IICX0	1	R/W	I²C Bus Transfer Select 1, 0 In combination with bits CKS2 to CKS0 in ICMR, this bit selects the transfer rate in master mode.
4	IICE	1	R/W	I²C Master Enable 0: Disables CPU access to the I²C bus interface data and control registers. 1: Enables CPU access to the I²C bus interface data and control registers.

- I²C Bus Control Register_0 (ICCR_0) Address: H'FFFF78[*1]

Bit	Bit Name	Setting	R/W	Function
7	ICE	0/1	R/W	I²C Bus Interface Enable 0: Disables the IIC module and initializes its internal state. SAR and SARX can be accessed. 1: Enables transfer via the IIC module (pins SCL and SDA are driving the bus). ICMR and ICDR can be accessed.
6	IEIC	1	R/W	I²C Bus Interface Interrupt Enable 0: Disables interrupts. 1: Enables interrupts.
5	MST	0	R/W	Master/Slave Select
4	TRS	0	R/W	Transmit/Receive Select 00: Slave receive mode
3	ACKE	0/1	R/W	Acknowledge Bit Decision Select 0: The value of the acknowledge bit is ignored and data is transferred continuously. 1: Continuous transfer is interrupted if the acknowledge bit is 1.
2	BBSY	0	R/W	Bus Busy [Setting condition] • Detection of the start condition [Clearing condition] • Detection of the stop condition
1	IRIC	0	R/(W)[*2]	I²C Bus Interrupt Request Flag [Setting condition] • Generation of an interrupt [Clearing condition] • Writing of 0 to this bit after reading it as 1
0	SCP	1	W	Start Condition/Stop Condition Prohibit Bit 0: Writing 0 here issues a start or stop condition, according to the value of the BBSY flag. 1: The SCP bit is always read as 1 (the initial value). Writing of 1 has no effect.

Notes: 1. Only accessible when the ICE bit in ICCR_0 is set to 1.
2. Only 0 can be written here, to clear the flag.

- I²C Bus Status Register_0 (ICSR_0) Address: H'FFFF79[*1]

Bit	Bit Name	Setting	R/W	Function
2	AAS	0	R/(W)[*2]	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, or when the general call address (H'00) is detected. [Setting condition] • Detection of the slave address in slave receive mode when FS = 0 • Detection of the general address in slave receive mode when FS = 0 [Clearing condition] • Writing of data to ICDR (transmit mode) or reading of data from ICDR (receive mode) • Writing of 0 to this bit after having read it as 1 • Transition to master mode
0	ACKB	0	R/W	Acknowledge Bit In receive mode, the value of the bit to be returned at acknowledge timing is set here. 0: 0 is returned as acknowledge data. 1: 1 is returned as acknowledge data.

Notes: 1. Only accessible when the ICE bit in ICCR_0 is set to 1.
2. Only 0 can be written here, to clear the flag.

- I²C Bus Mode Register_0 (ICMR_0) Address: H'FFFF7F[*1]

Bit	Bit Name	Setting	R/W	Function
7	MLS	0	R/W	MSB-First/LSB-First Select 0: MSB-first 1: LSB-first
6	WAIT	0	R/W	Wait Insert Bit 0: Data and acknowledge bits are transferred consecutively. 1: A wait is inserted between the data bits and acknowledge bit.
5	CKS2	1	R/W	Serial Clock Select Transfer rate is 100 kbps with ϕ = 20 MHz when IICX0 = 1 in the SCRX register, CKS2 = 1, CKS1 = 0, and CKS0 = 1.
4	CKS1	0	R/W	
3	CKS0	1	R/W	
2	BC2	0	R/W	Bit Counter Specify the number of bits to be transferred next. 000: 9 bits
1	BC1	0	R/W	
0	BC0	0	R/W	

Note: 1. Only accessible when the ICE bit in ICCR_0 is set to 1.

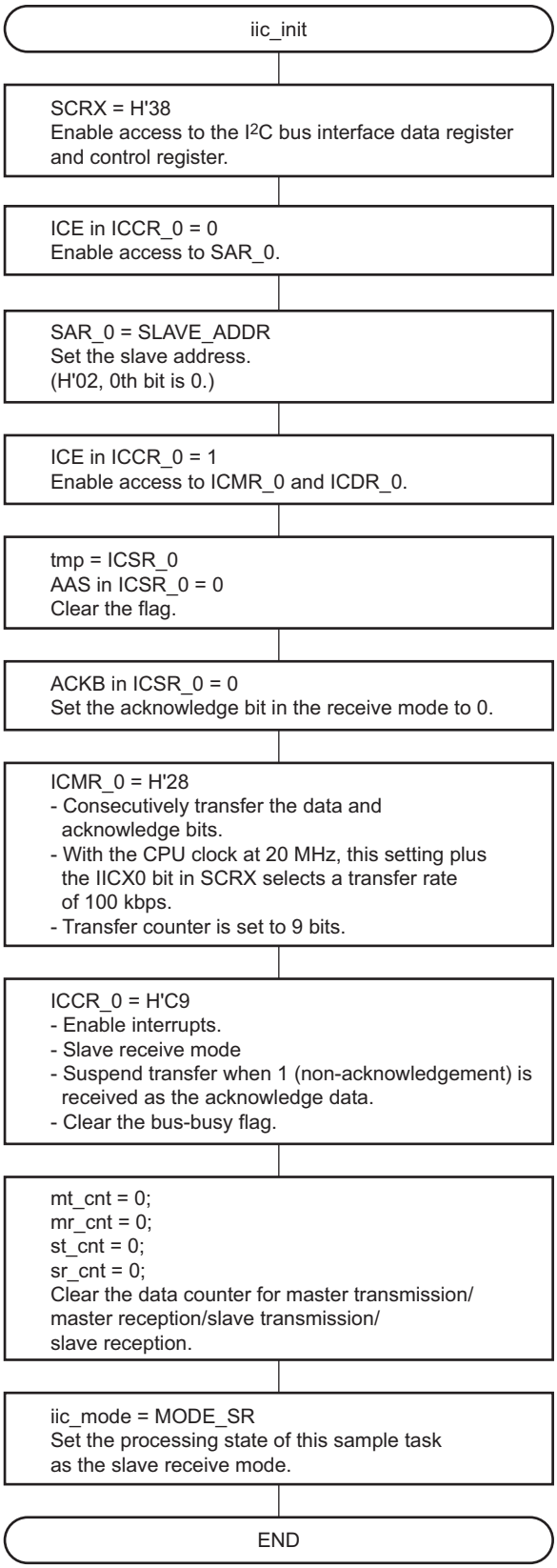
- Slave Address Register_0 (SAR_0) Address: H'FFFF7F[*1]

  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. SAR is assigned to the same address as ICMR, and can be written and read only when the ICE bit is cleared to 0 in ICCR.

Bit	Bit Name	Setting Value	R/W	Function
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	FS		R/W	Format Select Selects recognition of the slave address in the higher-order bits of SAR.

Note: 1. Only accessible when the ICE bit in ICCR_0 is set to 1.

5. Flow Chart


iic_init
SCRX = H'38
Enable access to the I2C bus interface data register and control register.
ICE in ICCR_0 = 0
Enable access to SAR_0.
SAR_0 = SLAVE_ADDR
Set the slave address.
(H'02, 0th bit is 0.)
ICE in ICCR_0 = 1
Enable access to ICMR_0 and ICDR_0.
tmp = ICSR_0
AAS in ICSR_0 = 0
Clear the flag.
ACKB in ICSR_0 = 0
Set the acknowledge bit in the receive mode to 0.
ICMR_0 = H'28
- Consecutively transfer the data and acknowledge bits.
- With the CPU clock at 20 MHz, this setting plus the IICX0 bit in SCRX selects a transfer rate of 100 kbps.
- Transfer counter is set to 9 bits.
ICCR_0 = H'C9
- Enable interrupts.
- Slave receive mode
- Suspend transfer when 1 (non-acknowledgement) is received as the acknowledge data.
- Clear the bus-busy flag.
mt_cnt = 0;
mr_cnt = 0;
st_cnt = 0;
sr_cnt = 0;
Clear the data counter for master transmission/ master reception/slave transmission/ slave reception.
iic_mode = MODE_SR
Set the processing state of this sample task as the slave receive mode.
END

5.8.2 Function mtrs_start

1. Overview

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

2. Arguments

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: Arguments were normal. 1: Arguments were abnormal.

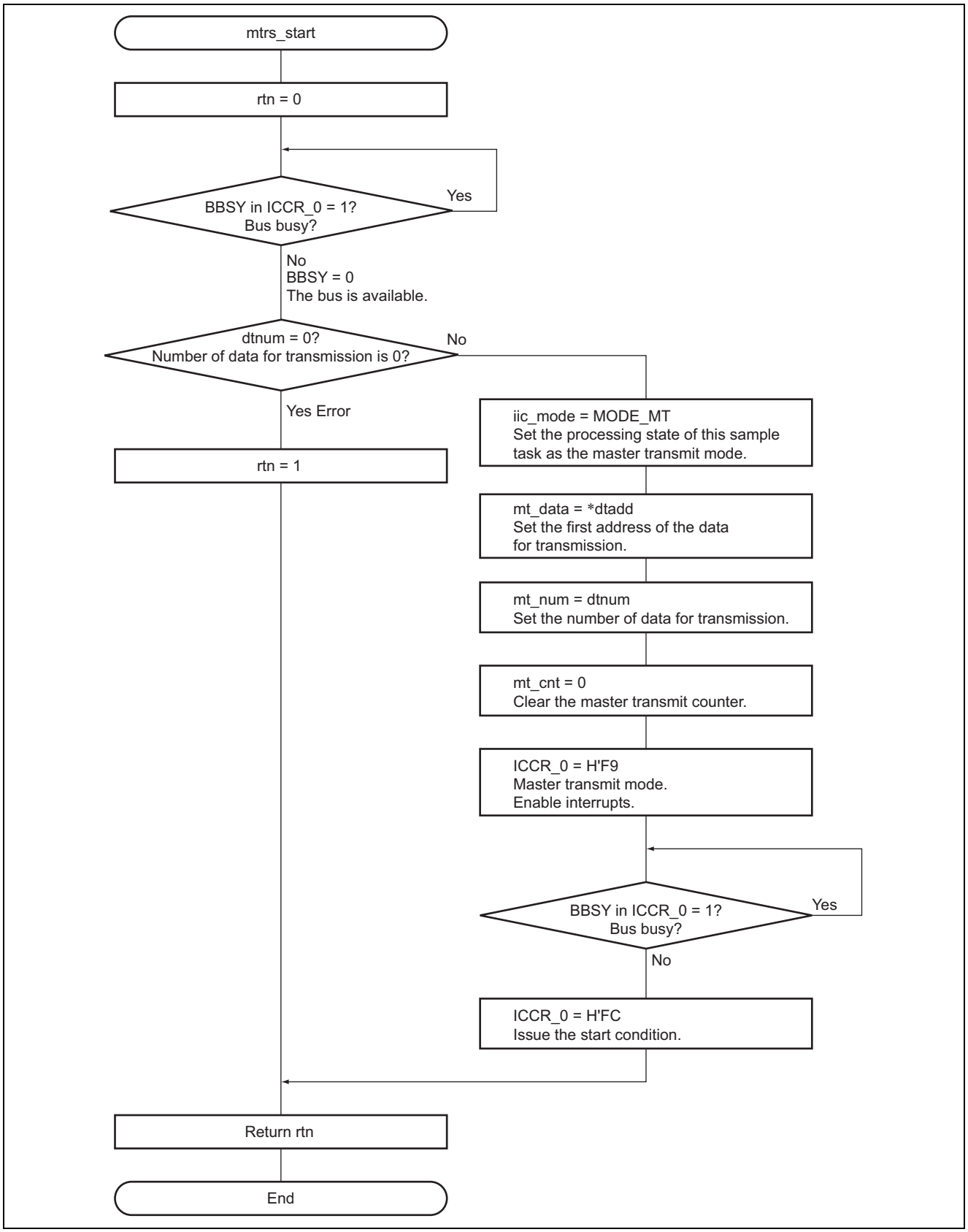
4. Internal registers used
   The following describes internal registers used in this sample task. The settings are those used in this sample task rather than the initial settings.

- I²C Bus Control Register_0 (ICCR_0) Address: H'FFFF78[*1]

Bit	Bit Name	Setting	R/W	Function
6	IEIC	1	R/W	I²C Bus Interface Interrupt Enable 0: Disables interrupts. 1: Enables interrupts.
5	MST	1	R/W	Master/Slave Select
4	TRS	1	R/W	Transmit/Receive Select 11: Master transmit mode
3	ACKE	1	R/W	Acknowledge Bit Judgment Selection 0: The value of the acknowledge bit is ignored and data is transferred continuously. 1: Continuous transfer is interrupted when the acknowledge bit is 1.
2	BBSY	0/1	R/W	Bus Busy [Setting condition] • Detection of the start condition [Clearing condition] • Detection of the stop condition
1	IRIC	0	R/(W)[*2]	I²C Bus Interface Interrupt Request Flag [Setting condition] • Generation of an interrupt [Clearing condition] • Writing of 0 to this bit after reading it as 1
0	SCP	1/0	W	Start Condition/Stop Condition Prohibit Bit 0: Writing 0 here issues a start or stop condition, according to the value of the BBSY flag. The start condition is issued when BBSY = 1 and SCO = 0. 1: This bit is always read as 1 (the initial value). Writing of 1 has no effect.

Notes: 1. Only accessible when the ICE bit in ICCR_0 is set to 1.
2. Only 0 can be written here, to clear the flag.

5. Flow Chart


mtrs_start
rtn = 0
BBSY in ICCR_0 = 1?
Bus busy?
Yes
No
BBSY = 0
The bus is available.
dtnum = 0?
Number of data for transmission is 0?
No
Yes Error
rtn = 1
iic_mode = MODE_MT
Set the processing state 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.
ICCR_0 = H'F9
Master transmit mode.
Enable interrupts.
BBSY in ICCR_0 = 1?
Bus busy?
Yes
No
ICCR_0 = H'FC
Issue the start condition.
Return rtn
End

5.8.3 Function mrcv_start

1. Overview

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

2. Arguments

Type	Name of Variable	Description
const unsigned char	*dtadd	First address of received data
unsigned short	dtnum	Number of data (bytes) received

3. Return value

Type	Description
unsigned char	0: Arguments were normal. 1: Arguments were abnormal.

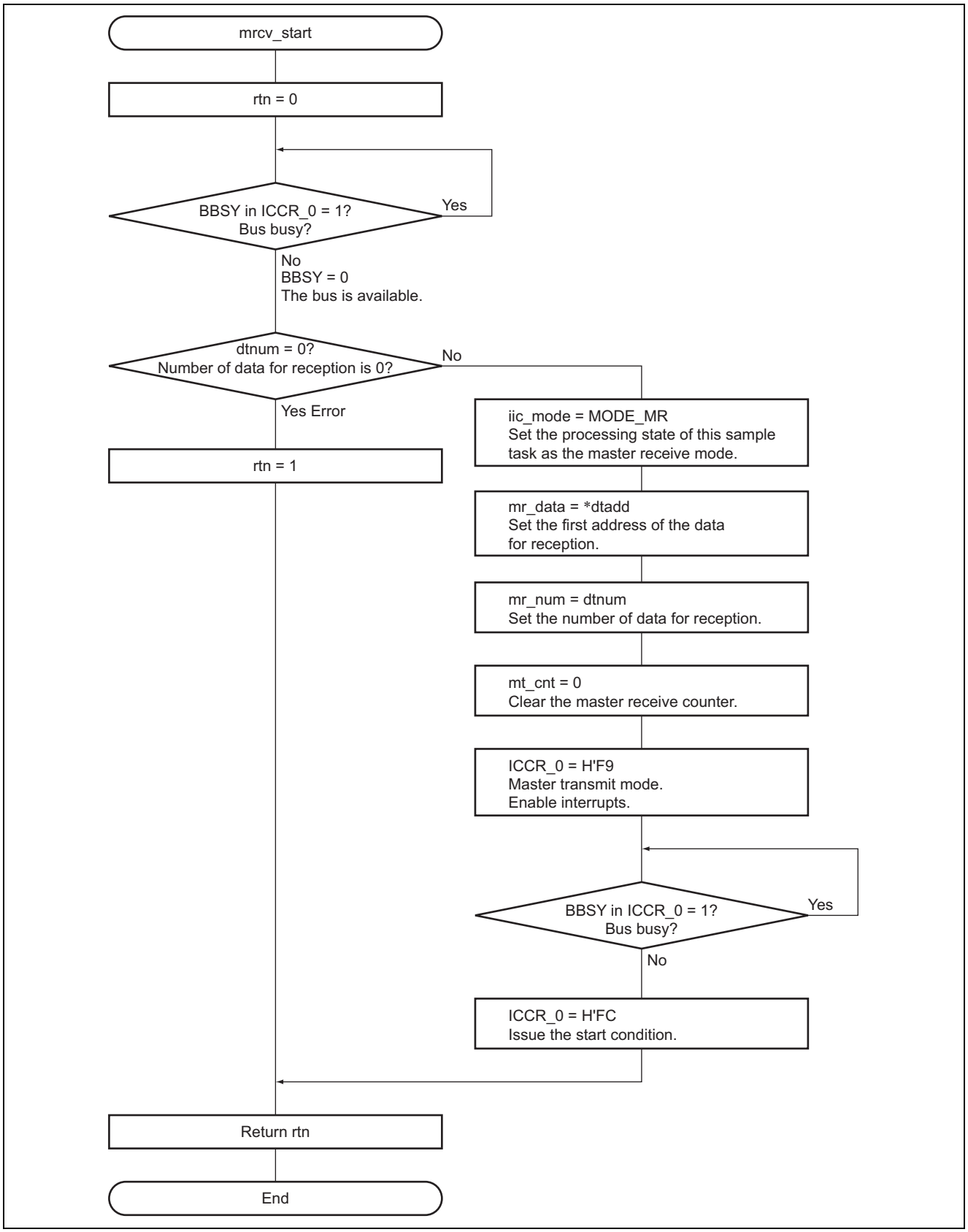
4. Internal registers used
   The following describes internal registers used in this sample task. The settings are those used in this sample task rather than the initial settings.

- I²C Bus Control Register_0 (ICCR_0) Address: H'FFFF78[*1]

Bit	Bit Name	Setting	R/W	Function
6	IEIC	1	R/W	I²C Bus Interface Interrupt Enable 0: Disables interrupts 1: Enables interrupts
5	MST	1	R/W	Master/Slave Select
4	TRS	1	R/W	Transmit/Receive Select 11: Master transmit mode
3	ACKE	1	R/W	Acknowledge Bit Judgment Selection 0: The value of the acknowledge bit is ignored and data is transferred continuously. 1: Continuous transfer is interrupted when the acknowledge bit is 1.
2	BBSY	0/1	R/W	Bus Busy [Setting condition] • Detection of the start condition [Clearing condition] • Detection of the stop condition
1	IRIC	0	R/(W)[*2]	I²C Bus Interface Interrupt Request Flag [Setting condition] • Generation of an interrupt [Clearing condition] • Writing of 0 to this bit after reading it as 1
0	SCP	1/0	W	Start Condition/Stop Condition Prohibit Bit 0: Writing 0 here issues a start or stop condition, according to the value of the BBSY flag. The start condition is issued when BBSY = 1 and SCO = 0. 1: This bit is always read as 1 (the initial value). Writing of 1 has no effect.

Notes: 1. Only accessible when the ICE bit in ICCR_0 is set to 1.
2. Only 0 can be written here, to clear the flag.

5. Flow Chart


mrcv_start
rtn = 0
BBSY in ICCR_0 = 1?
Bus busy?
Yes
No
BBSY = 0
The bus is available.
dtnum = 0?
Number of data for reception is 0?
No
Yes Error
rtn = 1
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.
mt_cnt = 0
Clear the master receive counter.
ICCR_0 = H'F9
Master transmit mode.
Enable interrupts.
BBSY in ICCR_0 = 1?
Bus busy?
Yes
No
ICCR_0 = H'FC
Issue the start condition.
Return rtn
End

5.8.4 Function iici0_int

1. 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.
2. Arguments
   None
3. Return value
   None
4. Internal registers used
   The following describes internal registers used in this sample task. The settings are those used in this sample task rather than the initial settings.

- I²C Bus Control Register_0 (ICCR_0) Address: H'FFFF78*1

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

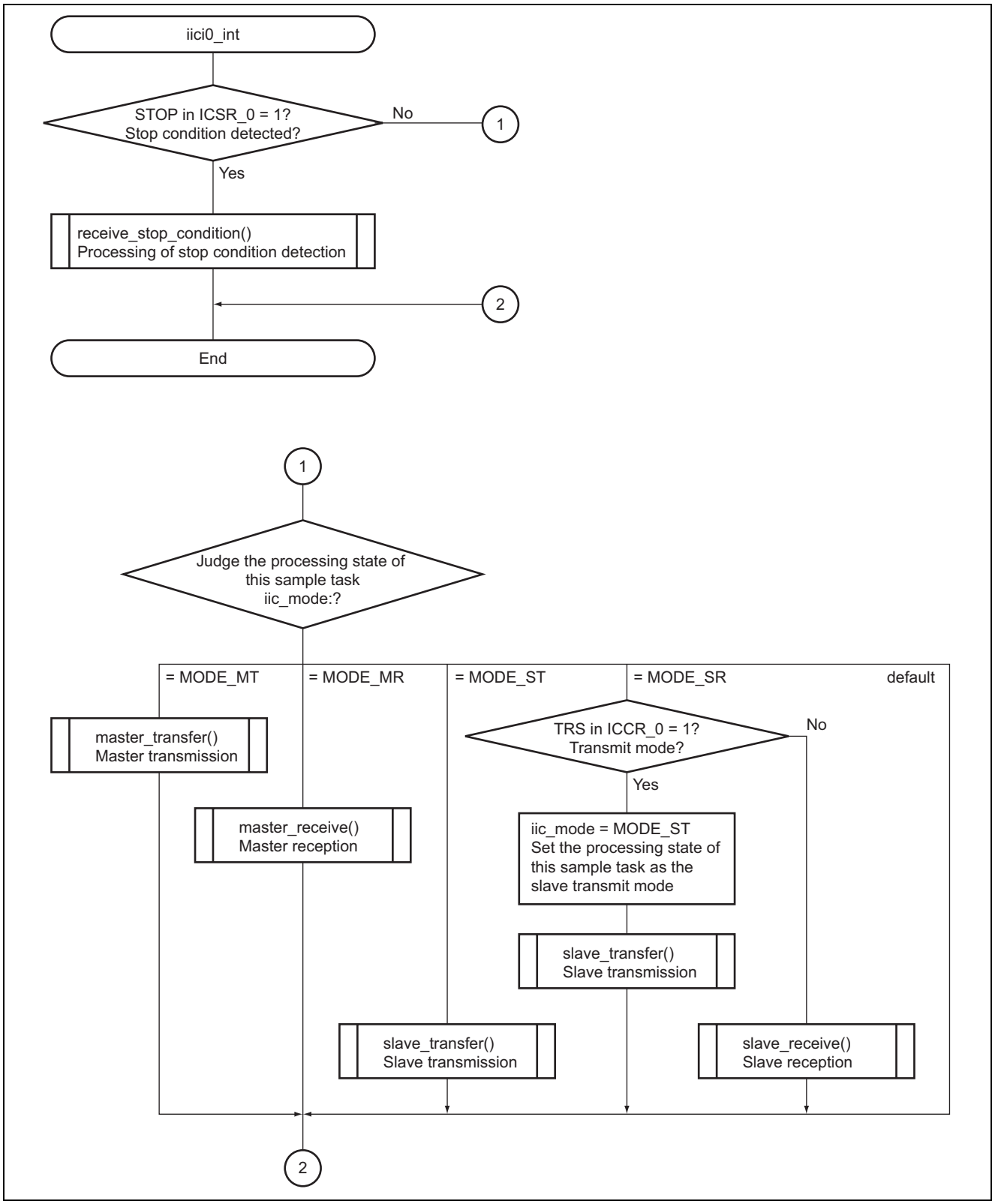
Note: 1. Only accessible when the ICE bit in ICCR_0 is set to 1.

- I²C Bus Status Register_0 (ICSR_0) Address: H'FFFF79*1

Bit	Bit Name	Setting	R/W	Function
6	STOP	Undefined	R/(W)*2	Normal Stop Condition Detection Flag [Setting condition] • Detection of a stop condition after completion of frame transfer in slave mode [Clearing condition] • Writing 0 to this bit after reading it as 1 • Clearing the IRIC flag to 0

Notes: 1. Only accessible when the ICE bit in ICCR_0 is set to 1.
2. Only 0 can be written here, to clear the flag.

5. Flow Chart


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
default
master_transfer()
Master transmission
master_receive()
Master reception
TRS in ICCR_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_transfer()
Slave transmission
slave_receive()
Slave reception
2

5.8.5 Function receive_stop_condition

1. Overview
   This function handles processing on detection of the stop condition.
2. Arguments
   None
3. Return value
   None
4. Internal registers used
   The following describes internal registers used in this sample task. The settings are those used in this sample task rather than the initial settings.

- I²C Bus Control Register_0 (ICCR_0) Address: H'FFFF78[*1]

Bit	Bit Name	Setting	R/W	Function
1	IRIC	0	R/W	I²C Bus Interface Interrupt Request Flag [Setting condition] • Generation of an interrupt [Clearing condition] • Writing of 0 to this bit after reading it as 1

Note: 1. Only accessible when the ICE bit in ICCR_0 is set to 1.

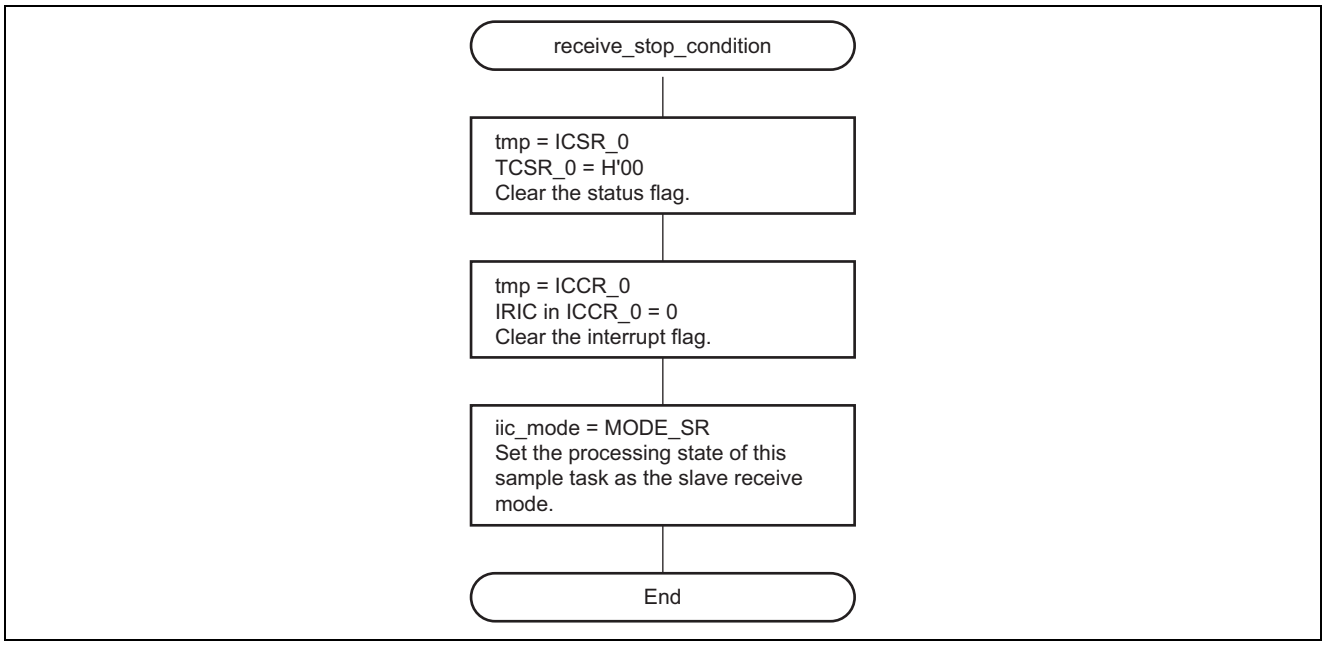
- I²C Bus Status Register_0 (ICSR_0) Address: H'FFFF79[*1]

Bit	Bit Name	Setting	R/W	Function
7	ESTP	0	R/(W)[*2]	Error Stop Condition Detection Flag [Setting condition] Detected erroneous stop condition • Detection of a stop condition during frame transfer in slave mode [Clearing condition] No erroneous stop condition • Writing of 0 to this bit after reading it as 1 • Clearing of the IRIC flag to 0
6	STOP	1	R/(W)[*2]	Normal Stop Condition Detection Flag [Setting condition] • Detection of a stop condition by the I²C bus interface after it has finished transferring a frame in slave mode [Clearing condition] • Writing of 0 to this bit after reading it as 1 • Clearing of the IRIC flag to 0

Bit	Bit Name	Setting	R/W	Function
5	IRTR	0	R/(W)*2	I²C Bus Interface Continuous Transmission/Reception Interrupt Request Flag [Setting condition] In I²C bus interface slave mode • Setting of the TDRE or RDRF flag to 1 when AASX = 1 In other modes • Setting of the TDRE or RDRF flag to 1 [Clearing condition] • Writing of 0 to this bit after reading it as 1 • Clearing of the IRIC flag
3	AL	0	R/(W)*2	Arbitration Lost [Setting condition] Failure in bus contention (loss of arbitration) • Different values for the internal SDA signal and SDA pin on a rising edge of SCL in master transmit mode • The internal SCL line being at the high level on a falling edge of SCL in master transmit mode [Clearing condition] • Writing of data to ICDR (transmit mode) or reading of data from ICDR (receive mode) • Writing of 0 to this bit after reading it as 1
2	AAS	0	R/(W)*2	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, or when the general call address (H'00) is detected. [Setting condition] • In slave receive mode with FS = 0: detection of the slave address • In slave receive mode with FS = 0: detection of the general call address [Clearing condition] • Writing of data to ICDR (transmit mode) or reading of data from ICDR (receive mode) • Writing of 0 to this bit after reading it as 1 • In master mode
0	ACKB	0	R/W	Acknowledge Bit In receive mode, the value of the bit to be sent at acknowledge timing is set here. 0: 0 is output at acknowledge timing. 1: 1 is output at acknowledge timing.

Notes: 1. Only accessible when the ICE bit in ICCR_0 is set to 1.
2. Only 0 can be written here, to clear the flag.

5. Flow Chart


receive_stop_condition
tmp = ICSR_0
TCSR_0 = H'00
Clear the status flag.
tmp = ICCR_0
IRIC in ICCR_0 = 0
Clear the interrupt flag.
iic_mode = MODE_SR
Set the processing state of this sample task as the slave receive mode.
End

5.8.6 Function master_transfer

1. 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. Arguments
   None
3. Return value
   None
4. Internal registers used
   The following describes internal registers used in this sample task. The settings are those used in this sample task rather than the initial settings.

- I²C Bus Control Register_0 (ICCR_0) Address: H'FFFF78[*1]

Bit	Bit Name	Setting	R/W	Function
7	ICE	1	R/W	I²C Bus Interface Enable 0: Disables the IIC module and initializes its internal state. SAR and SARX can be accessed. 1: Enables transfer via the IIC module (pins SCL and SCA are driving the bus). ICMR and ICDR can be accessed.
6	IEIC	0	R/W	I²C Bus Interface Interrupt Enable 0: Disables interrupts. 1: Enables interrupts.
5	MST	1	R/W	Master/Slave Select
4	TRS	1	R/W	Transmit/Receive Select 11: Master transmit mode
3	ACKE	0/1	R/W	Acknowledge Bit Judgment Selection 0: The value of the acknowledge bit is ignored and data is transferred continuously. 1: Continuous transfer is interrupted when the acknowledge bit is 1.
2	BBSY	0	R/W	Bus Busy [Setting condition] • Detection of the start condition [Clearing condition] • Detection of the stop condition
1	IRIC	0	R/(W)[*2]	I²C Bus Interface Interrupt Request Flag [Setting condition] • Generation of an interrupt [Clearing condition] • Writing of 0 to this bit after reading it as 1
0	SCP	0	W	Start Condition/Stop Condition Prohibit 0: Writing 0 here issues a start or stop condition, according to the value of the BBSY flag. The stop condition is issued when BBSY = 0 and SCO = 0. 1: Reading always returns a value of 1 (the initial value). Writing of 1 has no effect.

Notes: 1. Only accessible when the ICE bit in ICCR_0 is set to 1.
2. Only 0 can be written here, to clear the flag.

- I²C Bus Status Register_0 (ICSR_0) Address: H'FFFF79[*1]

Bit	Bit Name	Setting	R/W	Function
3	AL	Undefined	R/(W)[*2]	Arbitration Lost [Setting condition] Arbitration lost • Different values for the internal SDA signal and SDA pin on a rising edge of SCL in master transmit mode. • The internal SCL line being at the high level on a falling edge of SCL in master transmit mode [Clearing condition] • Writing of data to ICDR (transmit mode) or reading of data from ICDR (receive mode) • Writing of 0 to this bit after reading it as 1
0	ACKB	Undefined	R/W	Acknowledge Bit In transmit mode, the value of the acknowledge data returned from the receiving device is loaded into the ACKB bit. 0: Indicates that the receiving device has acknowledged the data (signal is 0). 1: Indicates that the receiving device has not acknowledged the data (signal is 1).

Notes: 1. Only accessible when the ICE bit in ICCR_0 is set to 1.
2. Only 0 can be written here, to clear the flag.

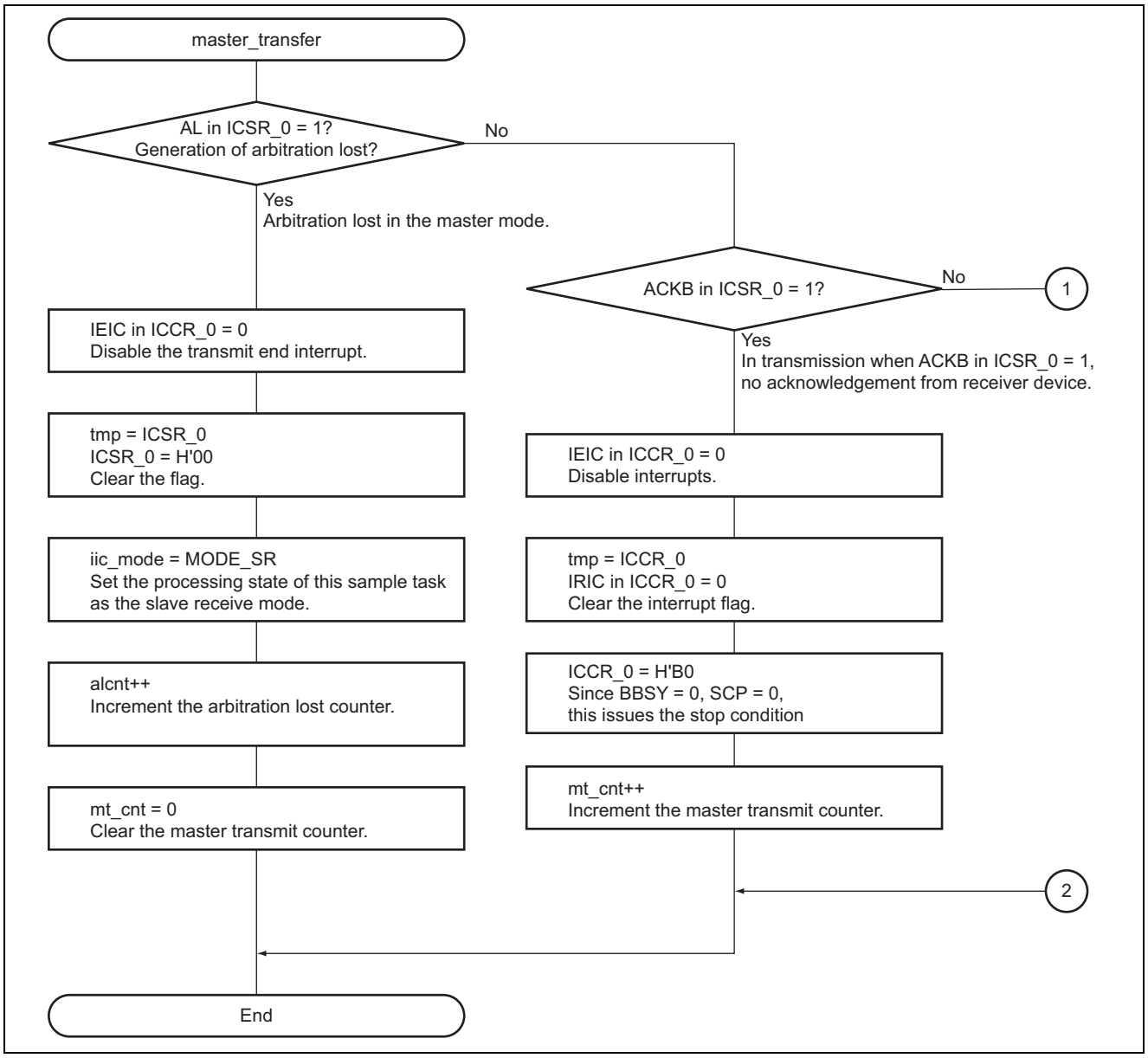
- I²C Bus Data Register_0 (ICDR_0) Address H'FFFF7E[*1]

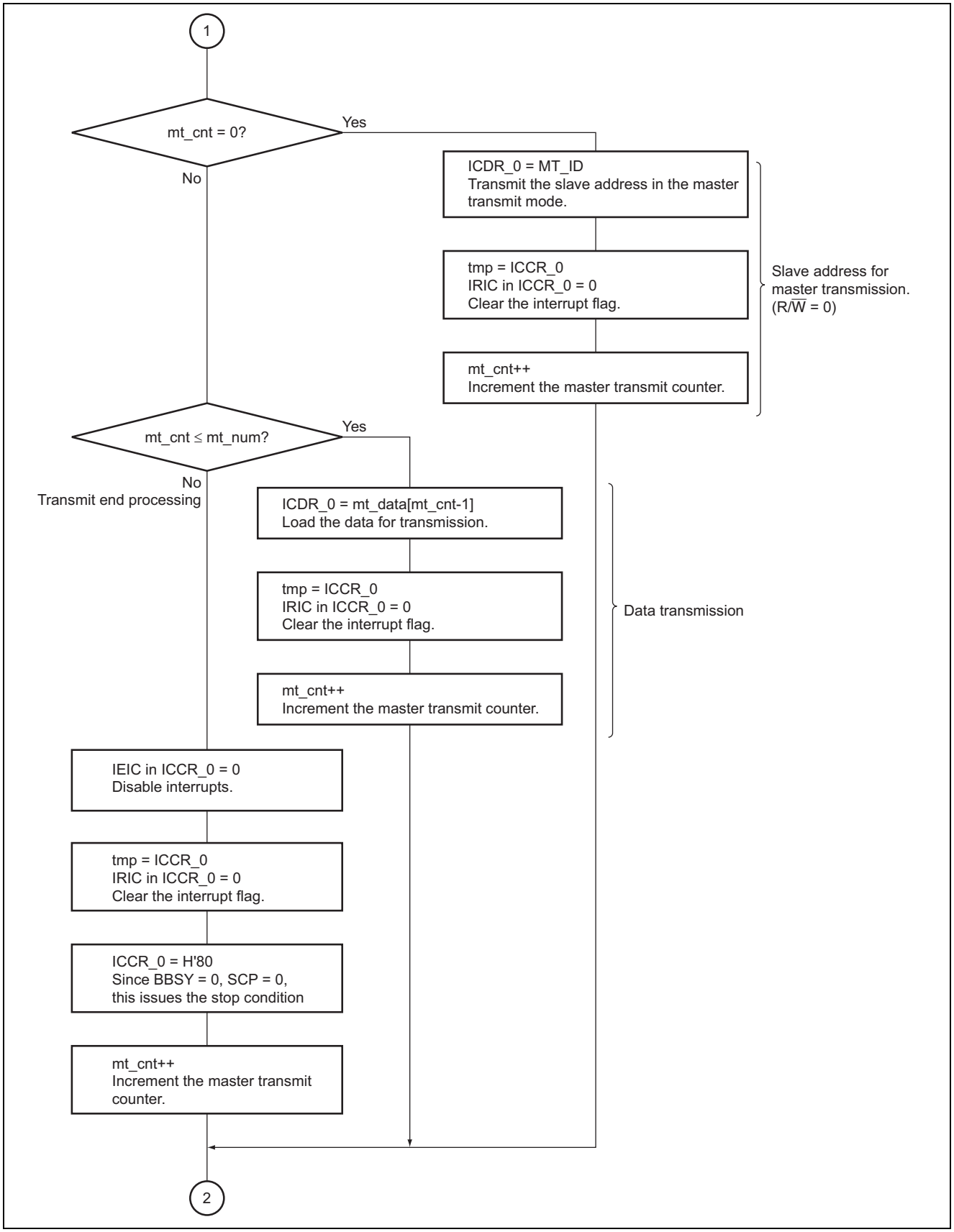
Function: ICDR is an 8-bit readable/writable register that is used as a transmit data register when transmitting and a receive data register when receiving. ICDR is internally divided into a shift register (ICDRS), receive buffer (ICDRR), and transmit buffer (ICDRT). ICDRS cannot be read or written by the CPU; ICDRR is read-only, and ICDRT is write-only. Data transfers among the three registers are performed automatically in coordination with changes in the bus state, and affect the states of internal flags such as TDRE and RDRF.

Setting: MT_ID, mt_data[mt_cnt-1]

Note: 1. Only accessible when the ICE bit in ICCR_0 is set to 1.

5. Flow Chart


master_transfer
AL in ICSR_0 = 1?
Generation of arbitration lost?
No
Yes
Arbitration lost in the master mode.
ACKB in ICSR_0 = 1?
No
1
Yes
In transmission when ACKB in ICSR_0 = 1,
no acknowledgement from receiver device.
IEIC in ICCR_0 = 0
Disable the transmit end interrupt.
tmp = ICSR_0
ICSR_0 = H'00
Clear the flag.
iic_mode = MODE_SR
Set the processing state of this sample task
as the slave receive mode.
alcnt++
Increment the arbitration lost counter.
mt_cnt = 0
Clear the master transmit counter.
IEIC in ICCR_0 = 0
Disable interrupts.
tmp = ICCR_0
IRIC in ICCR_0 = 0
Clear the interrupt flag.
ICCR_0 = H'B0
Since BBSY = 0, SCP = 0,
this issues the stop condition
mt_cnt++
Increment the master transmit counter.
2
End


1
mt_cnt = 0?
Yes
No
ICDR_0 = MT_ID
Transmit the slave address in the master transmit mode.
tmp = ICCR_0
IRIC in ICCR_0 = 0
Clear the interrupt flag.
mt_cnt++
Increment the master transmit counter.
Slave address for master transmission. (R/W = 0)
mt_cnt ≤ mt_num?
Yes
No
Transmit end processing
ICDR_0 = mt_data[mt_cnt-1]
Load the data for transmission.
tmp = ICCR_0
IRIC in ICCR_0 = 0
Clear the interrupt flag.
mt_cnt++
Increment the master transmit counter.
Data transmission
IEIC in ICCR_0 = 0
Disable interrupts.
tmp = ICCR_0
IRIC in ICCR_0 = 0
Clear the interrupt flag.
ICCR_0 = H'80
Since BBSY = 0, SCP = 0,
this issues the stop condition
mt_cnt++
Increment the master transmit counter.
2

5.8.7 Function master_receive

1. 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. Arguments
   None
3. Return value
   None
4. Internal registers used
   The following describes internal registers used in this sample task. The settings are those used in this sample task rather than the initial settings.

- I²C Bus Control Register_0 (ICCR_0) Address: H'FFFF78[*1]

Bit	Bit Name	Setting	R/W	Function
6	IEIC	0	R/W	I²C Bus Interface Interrupt Enable 0: Disables interrupts. 1: Enables interrupts.
5	MST	B'10 and B'11	R/W	Master/Slave Select
4	TRS		R/W	Transmit/Receive Select 10: Master receive mode 11: Master transmit mode
2	BBSY	0	R/W	Bus Busy [Setting condition] • Detection of the start condition [Clearing condition] • Detection of the stop condition
1	IRIC	0	R/(W)[*2]	I²C Bus Interface Interrupt Request Flag [Setting condition] • Generation of an interrupt [Clearing condition] • Writing of 0 to this bit after reading it as 1.
0	SCP	1	W	Start Condition/Stop Condition Prohibit Bit 0: Writing 0 here issues a start or stop condition, according to the value of the BBSY flag. The stop condition is issued when BBSY = 0 and SCP = 0. 1: Reading always returns a value of 1 (the initial value). Writing of 1 has no effect.

Notes: 1. Only accessible when the ICE bit in ICCR_0 is set to 1.
2. Only 0 can be written here, to clear the flag.

- I²C Bus Status Register_0 (ICSR_0) Address: H'FFFF79[*1]

Bit	Bit Name	Setting	R/W	Function
5	IRTR	0	R/(W)[*2]	I²C Bus Interface Continuous Transmission/Reception Interrupt Request Flag [Setting condition] In I²C bus interface slave mode • Setting of the TDRE or RDRF flag to 1 when AASX = 1 In other modes • Setting of the TDRE or RDRF flag to 1 [Clearing condition] • Writing of 0 to this bit after reading it as 1 • Clearing of the IRIC flag
3	AL	0	R/(W)[*2]	Arbitration Lost [Setting condition] Arbitration lost • Different values for the internal SDA signal and SDA pin on a rising edge of SCL in master transmit mode. • The internal SCL line being at the high level on a falling edge of SCL in master transmit mode [Clearing condition] • Writing of data to ICDR (transmit mode) or reading of data from ICDR (receive mode) • Writing of 0 to this bit after reading it as 1
0	ACKB	0	R/W	Acknowledge Bit In receive mode, the value of the bit to be sent at acknowledge timing is set here. 0: 0 is output at acknowledge timing. 1: 1 is output at acknowledge timing.

Note: 1. Only accessible when the ICE bit in ICCR_0 is set to 1.
2. Only 0 can be written here, to clear the flag.

- I²C Bus Data Register_0 (ICDR_0) Address H'FFFF7E[*1]

Function: ICDR is an 8-bit readable/writable register that is used as a transmit data register when transmitting and a receive data register when receiving. ICDR is internally divided into a shift register (ICDRS), receive buffer (ICDRR), and transmit buffer (ICDRT). ICDRS cannot be read or written by the CPU; ICDRR is read-only, and ICDRT is write-only. Data transfers among the three registers are performed automatically in coordination with changes in the bus state, and affect the states of internal flags such as TDRE and RDRF.

Setting: MT_ID

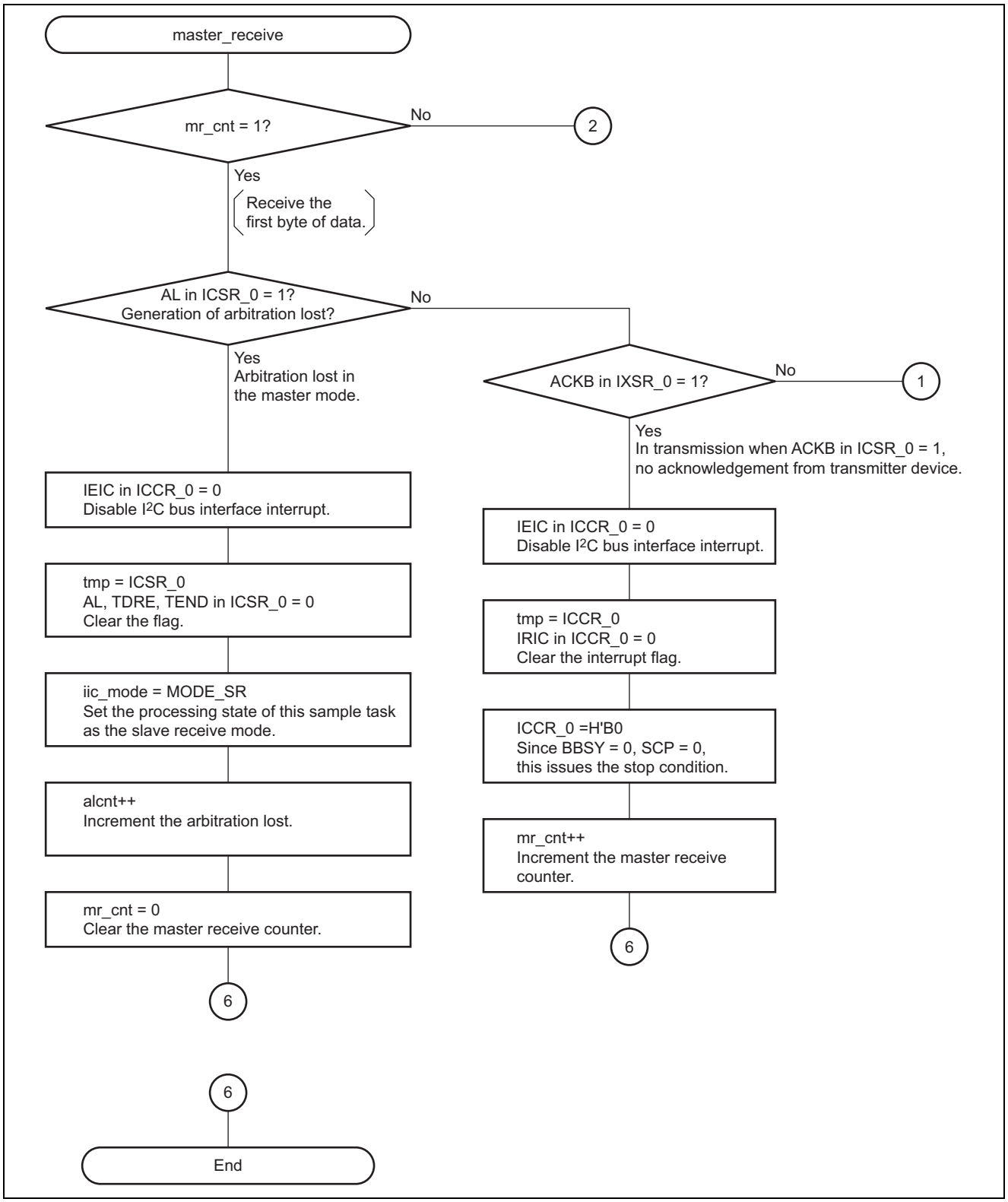
Note: 1. Only accessible when the ICE bit in ICCR_0 is set to 1.

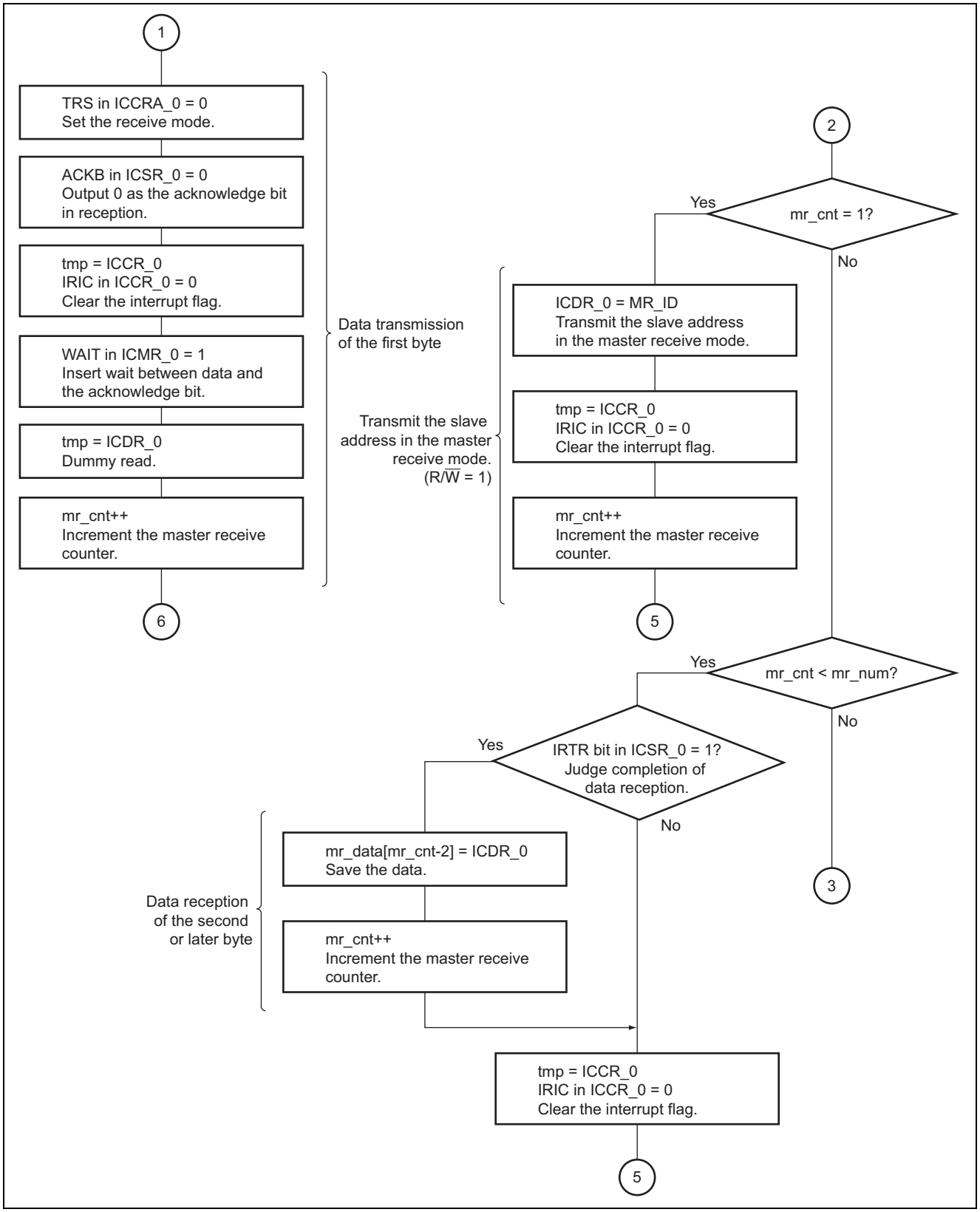
- •I²C Bus Mode Register_0 (ICMR_0) Address: H'FFFF7F[*1]

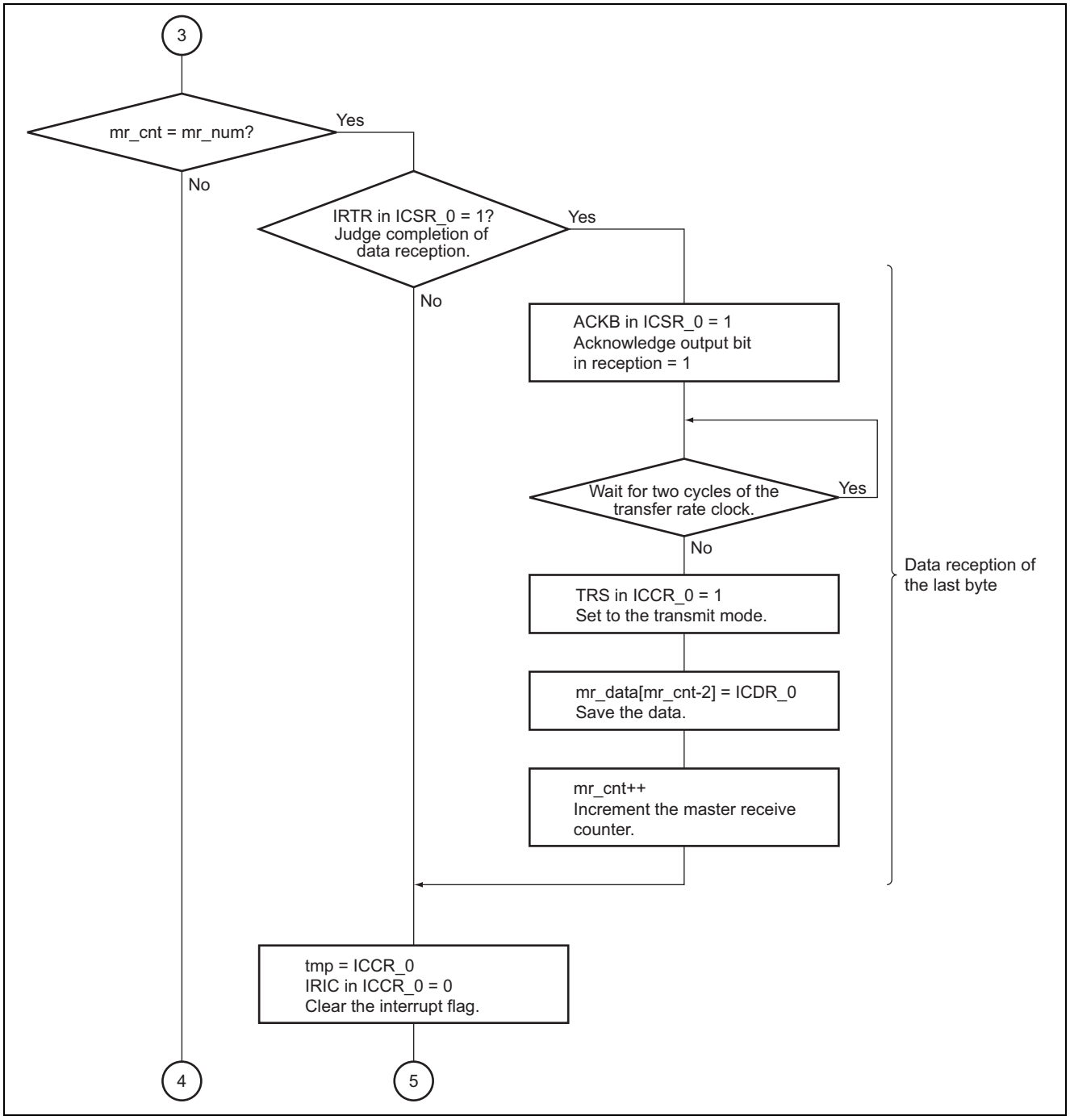
Bit	Bit Name	Setting	R/W	Function
6	WAIT	0/1	R/W	Wait Insertion Bit 0: Data and acknowledge bits are transferred consecutively. 1: A wait is inserted between data and acknowledge bits.

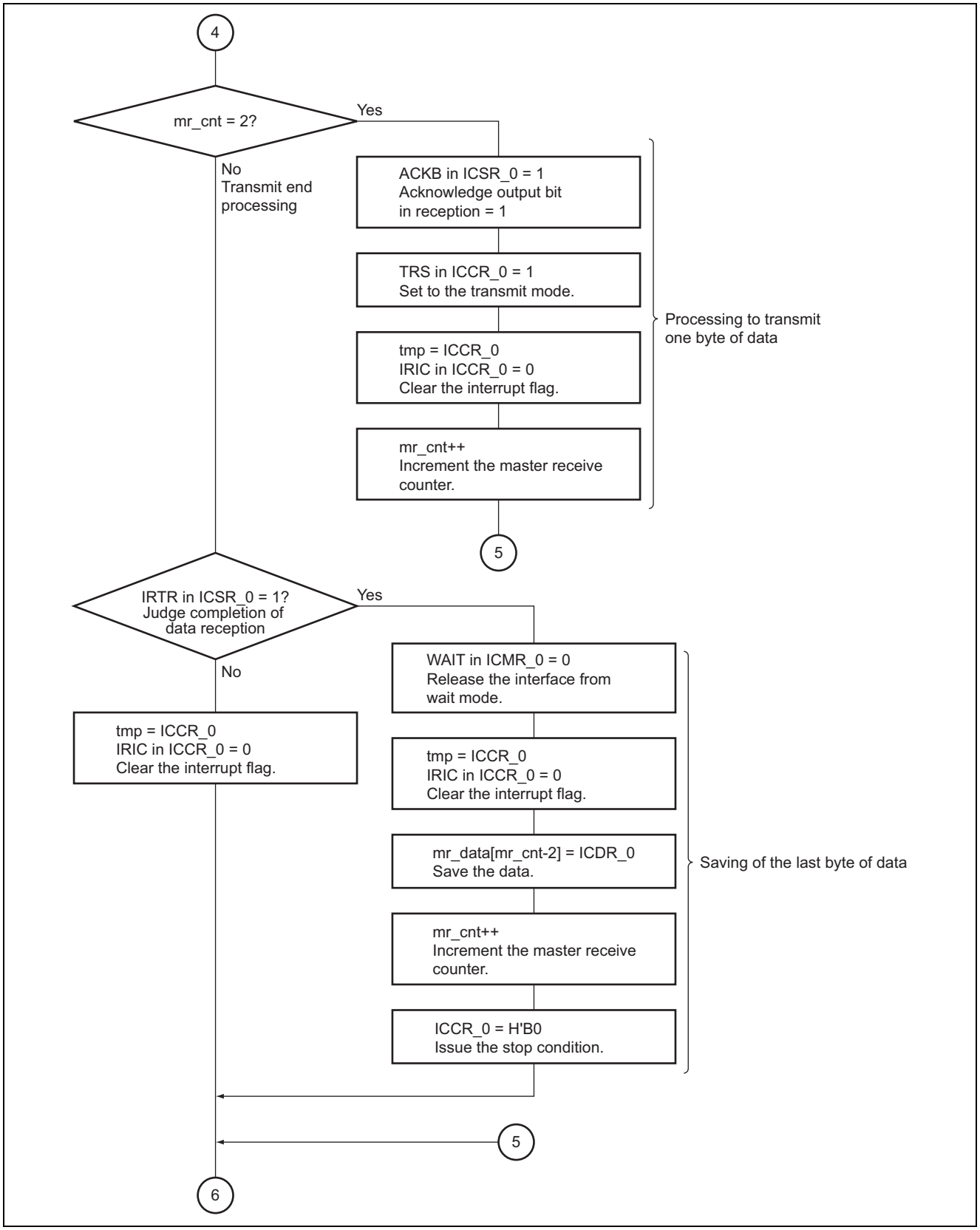
Note: 1. Only accessible when the ICE bit in ICCR_0 is set to 1.

5. Flow Chart


master_receive
mr_cnt = 1?
No
2
Yes
Receive the first byte of data.
AL in ICSR_0 = 1? Generation of arbitration lost?
No
Yes
Arbitration lost in the master mode.
ACKB in IXSR_0 = 1?
No
1
Yes
In transmission when ACKB in ICSR_0 = 1, no acknowledgement from transmitter device.
IEIC in ICCR_0 = 0
Disable I²C bus interface interrupt.
tmp = ICSR_0
AL, TDRE, TEND in ICSR_0 = 0
Clear the flag.
iic_mode = MODE_SR
Set the processing state of this sample task as the slave receive mode.
alcnt++
Increment the arbitration lost.
mr_cnt = 0
Clear the master receive counter.
6
IEIC in ICCR_0 = 0
Disable I²C bus interface interrupt.
tmp = ICCR_0
IRIC in ICCR_0 = 0
Clear the interrupt flag.
ICCR_0 =H'B0
Since BBSY = 0, SCP = 0, this issues the stop condition.
mr_cnt++
Increment the master receive counter.
6
6
End


1
TRS in ICCRA_0 = 0
Set the receive mode.
ACKB in ICSR_0 = 0
Output 0 as the acknowledge bit in reception.
tmp = ICCR_0
IRIC in ICCR_0 = 0
Clear the interrupt flag.
WAIT in ICMR_0 = 1
Insert wait between data and the acknowledge bit.
tmp = ICDR_0
Dummy read.
mr_cnt++
Increment the master receive counter.
6
Data transmission of the first byte
2
Yes
mr_cnt = 1?
No
ICDR_0 = MR_ID
Transmit the slave address in the master receive mode.
tmp = ICCR_0
IRIC in ICCR_0 = 0
Clear the interrupt flag.
mr_cnt++
Increment the master receive counter.
5
Transmit the slave address in the master receive mode. (R/W = 1)
Yes
mr_cnt < mr_num?
No
Yes
IRTR bit in ICSR_0 = 1? Judge completion of data reception.
No
mr_data[mr_cnt-2] = ICDR_0
Save the data.
mr_cnt++
Increment the master receive counter.
Data reception of the second or later byte
3
tmp = ICCR_0
IRIC in ICCR_0 = 0
Clear the interrupt flag.
5


3
mr_cnt = mr_num?
Yes
No
IRTR in ICSR_0 = 1?
Judge completion of
data reception.
Yes
No
ACKB in ICSR_0 = 1
Acknowledge output bit
in reception = 1
Wait for two cycles of the
transfer rate clock.
Yes
No
TRS in ICCR_0 = 1
Set to the transmit mode.
mr_data[mr_cnt-2] = ICDR_0
Save the data.
mr_cnt++
Increment the master receive
counter.
Data reception of
the last byte
tmp = ICCR_0
IRIC in ICCR_0 = 0
Clear the interrupt flag.
4
5


4
mr_cnt = 2?
Yes
No
Transmit end processing
ACKB in ICSR_0 = 1
Acknowledge output bit in reception = 1
TRS in ICCR_0 = 1
Set to the transmit mode.
tmp = ICCR_0
IRIC in ICCR_0 = 0
Clear the interrupt flag.
mr_cnt++
Increment the master receive counter.
Processing to transmit one byte of data
5
IRTR in ICSR_0 = 1?
Judge completion of data reception
Yes
No
WAIT in ICMR_0 = 0
Release the interface from wait mode.
tmp = ICCR_0
IRIC in ICCR_0 = 0
Clear the interrupt flag.
mr_data[mr_cnt-2] = ICDR_0
Save the data.
mr_cnt++
Increment the master receive counter.
ICCR_0 = H'B0
Issue the stop condition.
Saving of the last byte of data
tmp = ICCR_0
IRIC in ICCR_0 = 0
Clear the interrupt flag.
5
6

5.8.8 Function slave_transfer

1. Overview
   Slave-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.
2. Arguments
   None
3. Return value
   None
4. Internal registers used
   The following describes internal registers used in this sample task. The settings are those used in this sample task rather than the initial settings.

- I²C Bus Control Register_0 (ICCR_0) Address: H'FFFF78[*1]

Bit	Bit Name	Setting	R/W	Function
4	TRS	0	R/W	Transmit/Receive Select
		1	R/W	0: Receive mode 1: Transmit mode
3	ACKE	0/1	R/W	Acknowledge Bit Judgment Selection 0: The value of the acknowledge bit is ignored and data is transferred continuously. 1: Continuous transfer is interrupted when the acknowledge bit is 1.
1	IRIC	0	R/(W)[*2]	I²C Bus Interface Interrupt Request Flag [Setting condition] • Generation of an interrupt [Clearing condition] • Writing of 0 to this bit after reading it as 1

Notes: 1. Only accessible when the ICE bit in ICCR_0 is set to 1.
2. Only 0 can be written here, to clear the flag.

- I²C Bus Status Register_0 (ICSR_0) Address: H'FFFF79[*1]

Bit	Bit Name	Setting	R/W	Function
2	AAS	Undefined	R/(W)[*2]	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, or when the general call address (H'00) is detected. [Setting condition] • In slave receive mode with FS = 0: detection of the slave address • In slave receive mode with FS = 0: detection of the general call address [Clearing condition] • Writing of data to ICDR (transmit mode) or reading of data from ICDR (receive mode) • Writing of 0 to this bit after reading it as 1. • In master mode
0	ACKB	0	R/W	Acknowledge Bit In transmit mode, the value of the acknowledge data returned from the receiving device is loaded into the ACKB bit. 0: Indicates that the receiving device has acknowledged the data (signal is 0). 1: Indicates that the receiving device has not acknowledged the data (signal is 1).

Notes: 1. Only accessible when the ICE bit in ICCR_0 is set to 1.
2. Only 0 can be written here, to clear the flag.

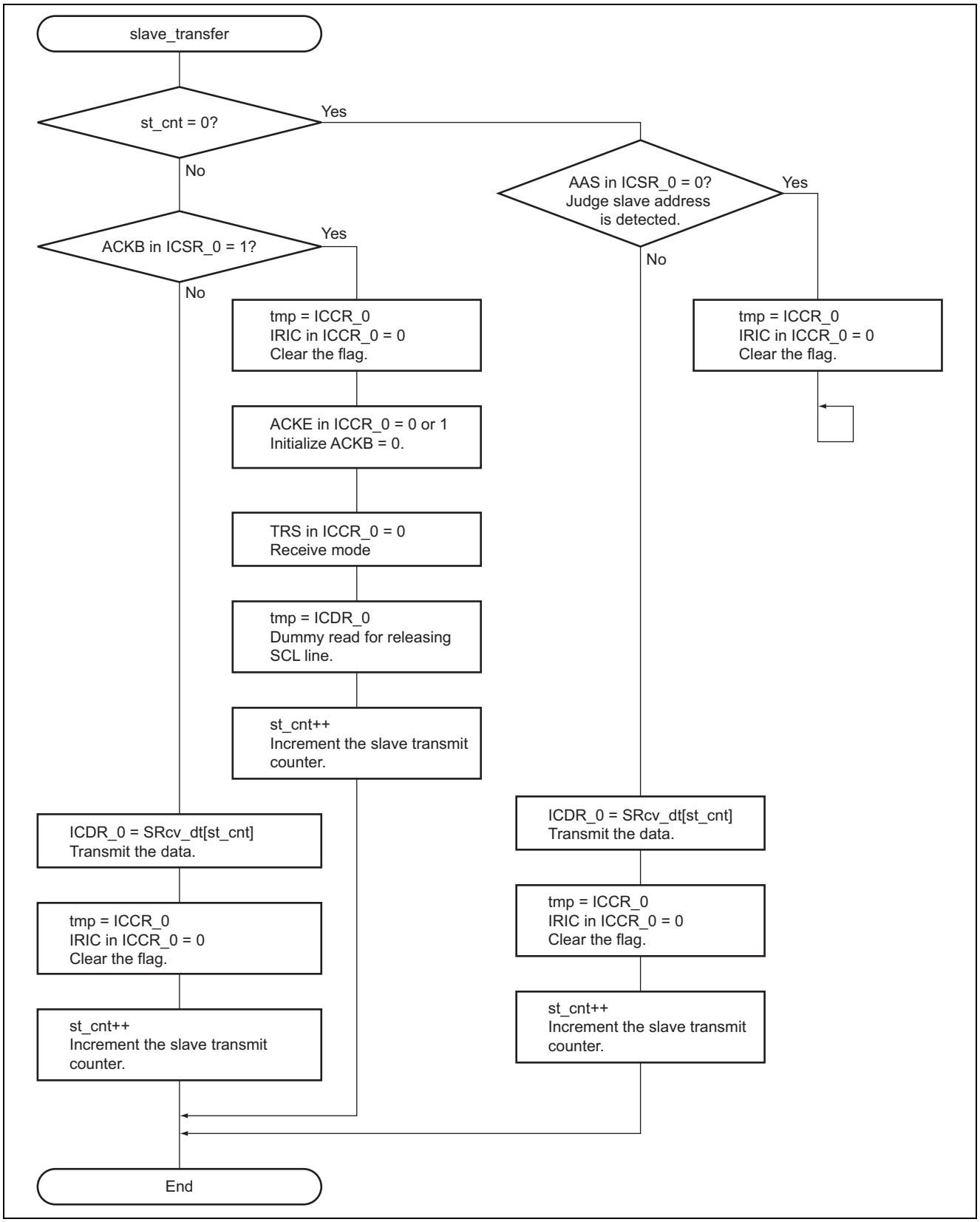
- I²C Bus Data Register_0 (ICDR_0) Address: H'FFFF7E[*1]

Function: ICDR is an 8-bit readable/writable register that is used as a transmit data register when transmitting and a receive data register when receiving. ICDR is internally divided into a shift register (ICDRS), receive buffer (ICDRR), and transmit buffer (ICDRT). ICDRS cannot be read or written by the CPU; ICDRR is read-only, and ICDRT is write-only. Data transfers among the three registers are performed automatically in coordination with changes in the bus state, and affect the states of internal flags such as TDRE and RDRF.

Setting: SRcv_dt[st_cnt]

Note: 1. Only accessible when the ICE bit in ICCR_0 is set to 1.

5. Flow Chart


slave_transfer
st_cnt = 0?
Yes
No
AAS in ICSR_0 = 0?
Judge slave address
is detected.
Yes
No
ACKB in ICSR_0 = 1?
Yes
No
tmp = ICCR_0
IRIC in ICCR_0 = 0
Clear the flag.
tmp = ICCR_0
IRIC in ICCR_0 = 0
Clear the flag.
ACKE in ICCR_0 = 0 or 1
Initialize ACKB = 0.
TRS in ICCR_0 = 0
Receive mode
tmp = ICDR_0
Dummy read for releasing
SCL line.
st_cnt++
Increment the slave transmit
counter.
ICDR_0 = SRcv_dt[st_cnt]
Transmit the data.
ICDR_0 = SRcv_dt[st_cnt]
Transmit the data.
tmp = ICCR_0
IRIC in ICCR_0 = 0
Clear the flag.
tmp = ICCR_0
IRIC in ICCR_0 = 0
Clear the flag.
st_cnt++
Increment the slave transmit
counter.
st_cnt++
Increment the slave transmit
counter.
End

5.8.9 Function slave_receive

1. 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. Arguments
   None
3. Return value
   None
4. Internal registers used
   The following describes internal registers used in this sample task. The settings are those used in this sample task rather than the initial settings.

- I²C Bus Control Register_0 (ICCR_0) Address: H'FFFF78*1

Bit	Bit Name	Setting	R/W	Function
1	IRIC	0	R/(W)*2	I²C Bus Interface Interrupt Request Flag [Setting condition] • Generation of an interrupt [Clearing condition] • Writing of 0 to this bit after reading it as 1

Notes: 1. Only accessible when the ICE bit in ICCR_0 is set to 1.
2. Only 0 can be written here, to clear the flag.

- I²C Bus Status Register_0 (ICSR_0) Address: H'FFFF79*1

Bit	Bit Name	Setting	R/W	Function
2	AAS	Undefined	R/(W)*2	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, or when the general call address (H'00) is detected. [Setting condition] • In slave receive mode with FS = 0: detection of the slave address • In slave receive mode with FS = 0: detection of the general call address [Clearing condition] • Writing of data to ICDR (transmit mode) or reading of data from ICDR (receive mode) • Writing of 0 to this bit after reading it as 1. • In master mode
0	ACKB	0	R/W	Acknowledge Bit In receive mode, the value of the bit to be sent at acknowledge timing is set here. 0: 0 is output at acknowledge timing. 1: 1 is output at acknowledge timing.

Notes: 1. Only accessible when the ICE bit in ICCR_0 is set to 1.
2. Only 0 can be written here, to clear the flag.

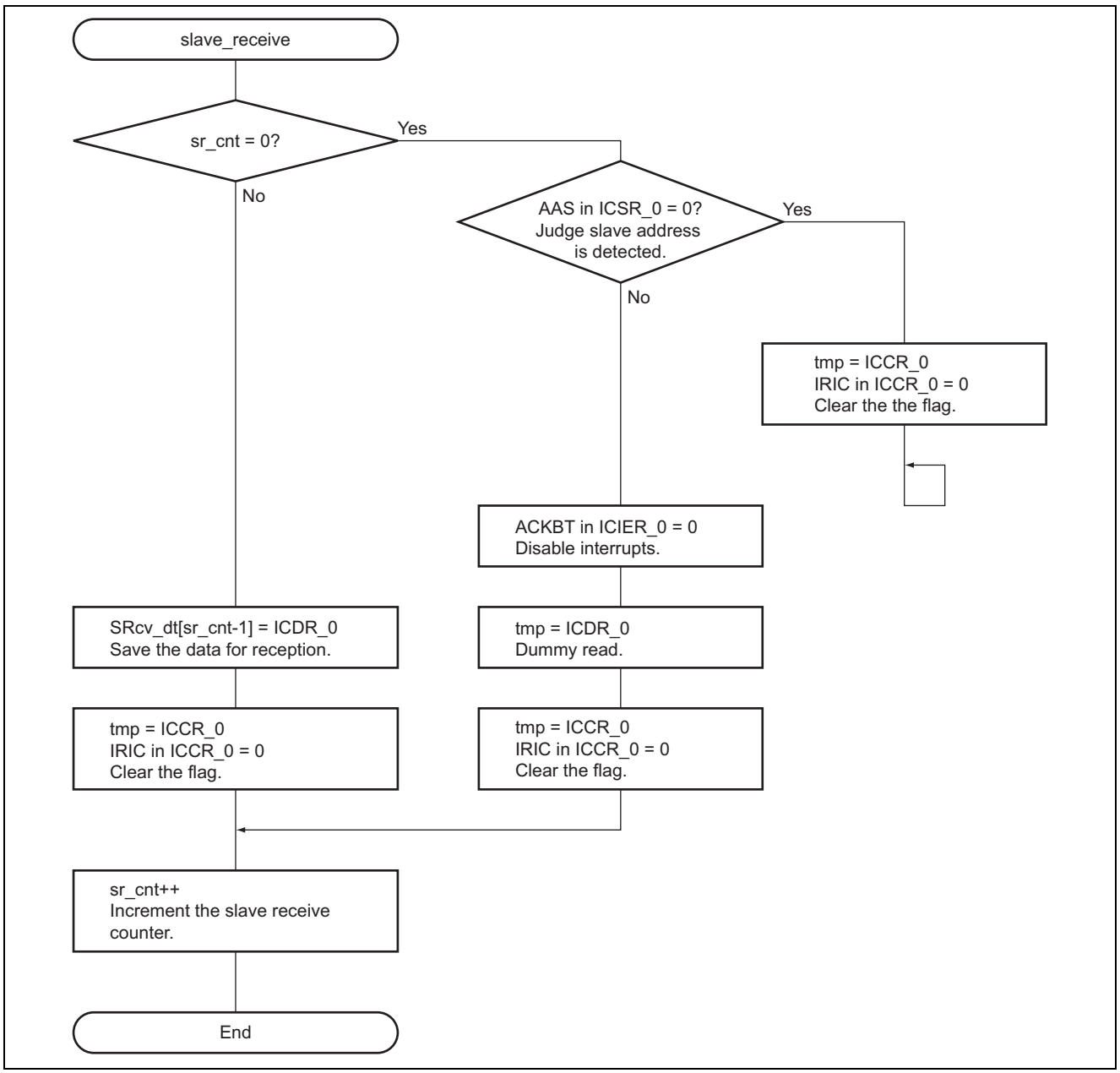
- I²C Bus Data Register_0 (ICDR_0) Address: H'FFFF7E*1

  Function: ICDR is an 8-bit readable/writable register that is used as a transmit data register when transmitting and a receive data register when receiving. ICDR is internally divided into a shift register (ICDRS), receive buffer (ICDRR), and transmit buffer (ICDRT). ICDRS cannot be read or written by the CPU; ICDRR is read-only, and ICDRT is write-only. Data transfers among the three registers are performed automatically in coordination with changes in the bus state, and affect the states of internal flags such as TDRE and RDRF.

  Setting: Undefined

Note: 1. Only accessible when the ICE bit in ICCR_0 is set to 1.

5. Flow Chart



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	Jan.31.07	—	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.