RL78/ I1D

R01AN3288EJ0100
Rev. 1.00
2016.11.15

I2C Master Communication Control using Serial Array Unit (Simple I2C) CC-RL

Introduction

This application note describes how to control the slave device functions via the I2C bus, by using the simple I2C function of the serial array unit. Specifically, the functions include LED display, reading operations of the A/D conversion results, and data transfer to/from the RAM function.

Operation Checked Device

RL78/I1D

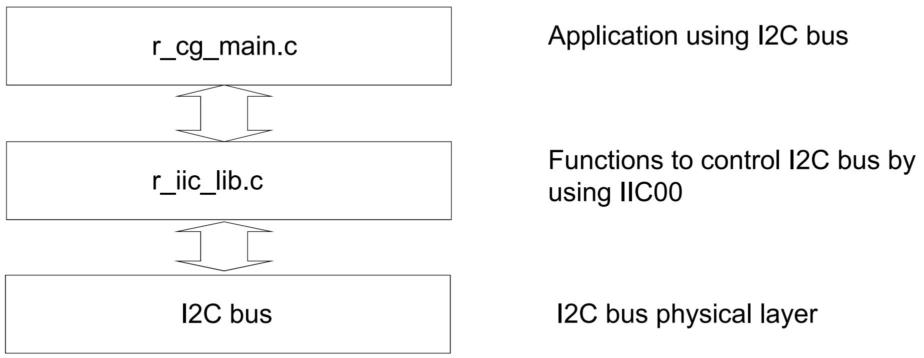
When applied to other microcontrollers, this application note should be modified according to the specifications of the microcontroller used and a thorough evaluation should be made.

Contents

1. Specifications

This application note describes how to control the peripheral functions connected to the I2C bus, using the simple I2C function of the serial array unit incorporated in the RL78 Series.

The processing function part that controls the I2C bus physical layer is independent as "r_iic_lib.c".
The main processing part "r_cg_main.c" controls the I2C bus using the functions provided by "r_iic_lib.c".
By modifying "r_iic_lib.c" and the initial settings of the peripheral functions through code creation, the serial interface IICA, which is the standard I2C bus of the RL78 family, can be easily replaced.


r_cg_main.c
Application using I2C bus
r_iic_lib.c
Functions to control I2C bus by using IIC00
I2C bus
I2C bus physical layer

Since the simple I2C does not have the wait function, the interrupt (INTTM03) of the timer array unit channel 3 (TM03) is combined to implement the wait function. For wait function control, refer to 6.1 Wait Function Control.

The specifications of the target slave here are shown below. For details, refer to the I2C Slave Communication Control using Software (for Multiple Addresses) Application Note (R01AN3289).

- LED display function: Two 8-bit data units are switched by using SW and displayed.
- A/D conversion function: The moving average of the 16 samples of 4-channel analog input is sent.
- RAM function: 128 bytes of data can be read from and written to arbitrary addresses.

The target slave is the board on which the RL78/I1D is mounted, which is described in the I2C Slave Communication Control using Software (for Multiple Addresses) Application Note (R01AN3289).

2. Conditions for Confirming Operations

The sample code operations described in this application note are confirmed under the following conditions.

Table 3.1 Conditions for Confirming Operations

Item	Description
Microcontroller used	RL78/I1D (R5F117GC)
Operating frequency	● High-speed on-chip oscillator (HOCO) clock: 24 MHz ● CPU/peripheral hardware clock: 24 MHz
Operating voltage	3.3 V (operable in a range from 2.7 V to 3.6 V) LVD operating mode: reset mode; voltage: 2.75 V
Integrated development environment	Renesas Electronics CS+ V4.00.00
Assembler	Renesas Electronics CC-RL V1.02.00
Board used	RL78/I1D target board (RTE5117GC0TGB00000R) + I2C slave board (refer to R01AN3289)

3. Related Application Notes

Refer to the following application notes, which are related to this application note.

RL78/G13 Initial Settings Application Note (R01AN2575J)

RL78/I1D I2C Slave Communication Control using Software (for Multiple Addresses) Application Note (R01AN3289)

4. Hardware Descriptions

4.1 Hardware Configuration Example

Figure 4.1 shows an example of the hardware configuration described in this application note.

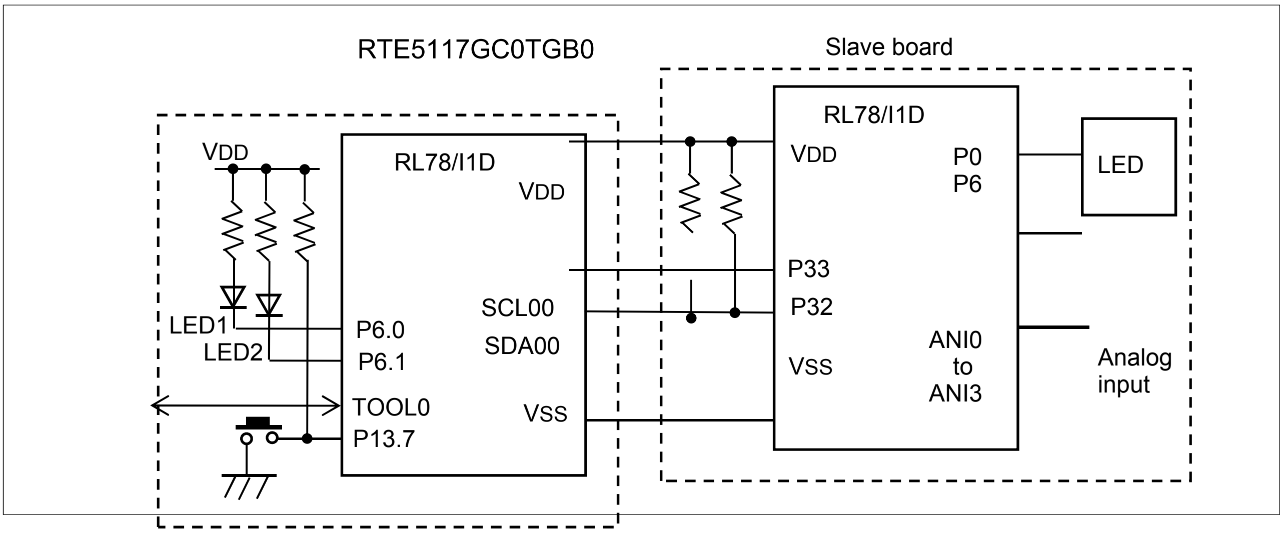


Figure 4.1 Hardware Configuration

Notes: 1. The above figure is a simplified circuit image for showing the outline of the connections. The actual circuit should be designed so that the pins are connected appropriately and that electrical characteristics are satisfied (input-only ports should be each connected to VDD or Vss via a resistor).

2. Set VDD to the reset-release voltage (V_{LVD}) specified by LVD or greater.

4.2 List of Pins Used

Table 4.1 lists the pins used and their functions.

Table 4.1 Pins Used and Their Functions

Pin Name	I/O	Function
SDA00	I/O	I2C communication data signal
SCL00	Output	I2C communication clock signal
P60	Output	Loop-end indication LED1 drive
P61	Output	Error indication LED2 drive
P137	Input	SW1 input

5. Software Descriptions

When the initial settings of the peripheral functions are completed and the I2C bus communication is ready, the following processes (1) to (6) are performed.

(1) Waits for SW1 connected to P137 to be pressed.

(2) When SW1 is pressed, sends 22 sets of data for LED display of the slave, using IIC00 of the serial array unit (SAU) at 50-ms intervals. If detecting the NACK response from the slave, turns on LED2 connected to P61.

(3) Reads 16 bytes of data from the slave RAM function eight times (128 bytes in total). If detecting the NACK response, turns on LED2.

(4) Verifies the data received in process (3). Specifically, compares the received 128 bytes of data to the previously transmitted data among the transmission data stored in the flash memory. If the received data does not agree with the transmitted data, turns on LED2.

(5) Writes data to the slave RAM function. Specifically, transmits two sets (128 bytes) out of eight sets of 64-byte data.

(6) Receives the slave A/D conversion results. Specifically, reads four channels of data (8 bytes), which are divided into the upper and lower data, 16 times. If detecting the NACK response, turns on LED2. Repeats process (6) until SW1 is pressed.

Repeats processes (2) to (5) while SW1 is pressed.

5.1 List of Settings Reflected to Option Bytes

For details on the initial settings, refer to 6.2 Settings through Code Creation. Table 5.1 shows the sample settings reflected to the option bytes.

Table 5.1 Settings Reflected to Option Bytes

Address	Setting	Content
0x000C0	0b11101110	Watchdog timer is stopped. (Counting stopped after a reset release)
0x000C1	0b01111111	LVD reset mode; 2.75 V (2.76 V to 2.87 V)
0x000C2	0b11100000	HS mode; high-speed on-chip oscillator (HOCO): 24 MHz
0x000C3	0b10000100	On-chip debugging is enabled.

5.2 List of Constants

Table 5.2 lists the constants used in the sample codes.

Table 5.2 Constants Used in Sample Codes

Constant	Setting	Content
MAX_DATA	64	Maximum count of data units transferred in a single communication.
SLAVE_ADDR	0x60	A/D and LED slave addresses
SLAVE_ADDR2	0x70	RAM slave address
RAM_TOP	0x80	Slave RAM capacity
LED_ON	0	Value for turning on LED
dispdata[22][2]	-	Data for turning on slave LED
ram_data[8][64]	-	Data to be written to slave RAM
TRUTH	1	Indicates true.
FALSE	0	Indicates false.
INT_MASK	1	Value for masking interrupts
INT_ENABLE	0	Value for enabling interrupts
TM03_trigger	0x08	Value for triggering TM03
IIC_WAIT_TIME	0x04	Wait time (5-us cycle count) upon INTIIC00 generation
IIC_STS_MASK	0x0F	Value for masking I2C unused bus status bits
IIC_USING	0x01	Value indicating that I2C bus communication is in progress
IIC_SUCCESS	0x00	Value indicating that I2C bus communication has been successfully completed.
IIC_RX_SVADDR	0x81	Value indicating that a slave address for reception is currently transmitted.
IIC_RX_MODE	0x41	Value indicating I2C bus reception mode
IIC_ERROR	0xFF	Value indicating a communication error on I2C bus
IIC00_TRG	0x0001	Value for triggering IIC00
IIC00_CK	0x0100	Value for triggering SCL (CK00) of IIC00
DUMMY_DATA	0xFF	Dummy data for starting reception

5.3 List of Variables

Table 5.3 lists the variables used in the sample codes. g_iic00_status and the following variables are for IIC00 hardware control library.

Table 5.3 Variables Used in Sample Codes

Type	Variable Name	Contents	Function Used
uint8_t	g_read_ram[8][16]	Buffer for receiving RAM data	main()
uint8_t	g_read_data[16][8]	Buffer for receiving A/D data	main()

uint8_t	g_write_data [MAX_DATA+1]	Buffer for transmitting data to be written to RAM	g_iic_put_data()
			g_iic_get_data()
			g_iic_put_data2()

uint8_t	g_t50ms_status	Flag for confirming that 50 ms has elapsed.	R_MAIN_UserInit()
			wait_50ms()
			r_it_interrupt()

uint8_t	g_iic00_status	IIC00 operating state
		0x00: Successfully completed
		0x01: Currently transmitting data
		0x41: Currently receiving data
		0x81: Currently transmitting an address for reception
		0xFF: Communication error
			R_IIC00_MasterSend()
			R_IIC00_MasterReceive()
			R_IIC00_CheckComstate()
			R_IIC00_WaitComend()
			r_inttm03_interrupt

uint8_t	gp_iic00_rx_address	Pointer for storing received data	R_IIC00_MasterReceive()
			r_inttm03_interrupt()

uint16_t	g_iic00_rx_len	Count of data to be received	R_IIC00_MasterReceive() r_ inttm03_interrupt()
uint16_t	g_iic00_rx_cnt	Count of data received	R_IIC00_MasterReceive() r_ inttm03_interrupt()
uint8_t	gp_iic00_tx_address	Pointer for reading transmission data	R_IIC00_MasterSend() r_ inttm03_interrupt()
uint16_t	g_iic00_tx_cnt	Count of remaining data to be transmitted	R_IIC00_MasterSend() r_ inttm03_interrupt()

uint8_t	g_TM03flag	Flag indicating that 20 μs is currently measured.	R_IIC00_TM03_Init()
			R_IIC00_MasterSend()
			R_IIC00_MasterReceive()
			R_IIC00_StopCondition()
			r_iic00_interrupt()
			r_inttm03_interrupt()

uint8_t	g_TM03cnt	Counter counting 5 μs to measure 20 μs.	R_IIC00_TM03_Init()
			r_iic00_interrupt()
			r_inttm03_interrupt()

5.4 List of Functions

Table 5.4 lists the functions used for I2C bus control. IIC_TM03_init and the following functions are provided by IIC00 hardware control library.

Table 5.4 List of Functions

Function Name	Summary
R_MAIN_UserInit()	Prepares for IIC00 communication.
g_wait_sw()	Waits for SW1 to be pressed.
g_iic_put_data()	Transmits data to RAM by using the IIC00 library functions.
g_iic_put_data2()	Transmits data to LED by using the IIC00 library functions.
g_iic_get_data()	Receives data from RAM by using the IIC00 library functions.
g_iic_get_data2()	Receives A/D conversion results by using the IIC00 library functions.
wait_50ms()	Waits for 50 ms.
r_it_interrupt	Processes a 50-ms interval timer interrupt.
R_IIC00_TM03_init()	Prepares for starting IIC00 and TM03.
R_IIC00_MasterSend()	Starts transmission of the specified data to slave.
R_IIC00_MasterReceive()	Starts data reception from slave.
R_IIC00_WaitComend()	Waits for completion of I2C bus communication.
R_IIC00_CheckComstate()	Checks the I2C bus communication state.
R_IIC00_StopCondition()	Issues the stop condition to I2C bus.
R_TM03_20us()	Waits for 20 μs.
wait_5us()	Waits for 5 μs.
r_iic00_stop()	Stops IIC00 operation.
r_iic00_startcondition()	Issues the start condition to I2C bus.
r_iic00_interrupt	Starts TM03 upon IIC00 communication completion.
r_inttm03_interrupt	Performs an IIC00 communication completion process.

5.5 Function Specifications

The following gives the specifications of the functions used in the sample code.

[Function name] R_MAIN_UserInit

Summary	Prepares for IIC00 communication.
Header	r_cg_macrodriver.h, r_cg_userdefine.h, r_iic_lib.h
Declaration	void R_MAIN_UserInit(void);
Description	Initializes IIC00 and TM03 to start IT.
Arguments	None
Return values	None
Remarks	None

[Function name] g_wait_sw

Summary	Waits for SW1 to be pressed.
Header	r_cg_userdefine.h
Declaration	void g_wait_sw(void);
Description	Waits for SW1 connected to P137 to be pressed.
Arguments	None
Return values	None
Remarks	Senses the edge.

[Function name] g_iic_put_data

Summary	Transmits data to RAM by using the IIC00 library functions.
Header	r_cg_macrodriver.h, r_iic_lib.h

Declaration MD_STATUS g_iic00_put_data(uint8_t s_addr, uint8_t r_addr, uint8_t __far * const buffer, uint8_t tx_num);

Description Transmits data to the specified RAM address by using the IIC00 functions. Temporarily copies the transmission data to the area for variables and calls the IIC00 transmission function to start transmission. After having successfully started transmission, waits for communication completion while checking the communication status. After communication completion, issues the stop condition to release the I2C bus and stops IIC00.

Arguments

s_addr	Slave address
r_addr	RAM address to be accessed
buffer	Data to be transmitted
tx_num	Number of data to be transmitted

Return values ［IIC_SUCCESS］: Transmission completed
[IIC_ERROR]: Transmission failed (NACK response)

Remarks Address register supported

[Function name] g_iic_put_data2

Summary Transmits data to LED by using the IIC00 library functions.

Header r_cg_macrodriver.h, r_iic_lib.h

Declaration MD_STATUS g_iic_put_data2(uint8_t s_addr, uint8_t __far * const buffer, uint8_t tx_num);

Description Transmits data to LED by using the IIC00 functions. Temporarily copies the transmission data to the area for variables and calls the IIC00 transmission function to start transmission. After having successfully started transmission, waits for communication completion while checking the communication status. After communication completion, issues the stop condition to release the I2C bus and stops IIC00.

Arguments

s_addr	Slave address
buffer	Data to be transmitted
tx_num	Number of data to be transmitted

Return values ［IIC_SUCCESS］: Transmission completed
[IIC_ERROR]: Transmission failed (NACK response)

Remarks No address register

[Function name] g_iic_get_data

Summary Receives data from RAM by using the IIC00 library functions.

Header r_cg_macrodriver.h, r_iic_lib.h

Declaration MD_STATUS g_iic_get_data(uint8_t s_addr, uint8_t r_addr, uint8_t * const buffer, uint8_t rx_num);

Description Receives data from the specified register of the specified slave by using the IIC00 functions. After transmitting the register address to the specified slave, restarts to start data reception. After having successfully started reception, waits for communication completion while checking the communication status. After communication completion, issues the stop condition to release the I2C bus and stops IIC00.

Arguments

s_addr	Slave address
r_addr	RAM address to be accessed as slave
buffer	Buffer for storing received data
rx_num	Number of data to be received

Return values ［IIC_SUCCESS］： Reception completed
［IIC_ERROR］： Transmission failed (NACK response)

Remarks Address register supported

[Function name] g_iic_get_data2

Summary	Receives data from A/D by using the IIC00 library functions.	
Header	r_cg_macrodriver.h, r_iic_lib.h	
Declaration	MD_STATUS g_iic_get_data2(uint8_t s_addr, uint8_t * const buffer, uint8_t rx_num);	
Description	Receives A/D conversion result data by using the IIC00 functions. Starts data reception of the specified slave. After having successfully started reception, waits for communication completion while checking the communication status. After communication completion, issues the stop condition to release the I2C bus and stops IIC00.	
Arguments	s_addr	Slave address
	buffer	Buffer for storing received data
	rx_num	Number of data to be received
Return values	[IIC_SUCCESS]: Reception completed [IIC_ERROR]: Transmission failed (NACK response)	
Remarks	No address register	

[Function name] wait_50ms

Summary	Waits for 50 ms.
Header	r_cg_userdefine.h
Declaration	void wait_50ms(void);
Description	Restarts the 50-ms interval timer and waits for 50 ms.
Arguments	None
Return values	None
Remarks	None

[Function name] r_it_interrupt

Summary	Processes a 50-ms interval timer interrupt.
Header	r_cg_userdefine.h
Declaration	#pragma interrupt r_it_interrupt(vect=INTIT,bank=RB2) static void r_it_interrupt(void)
Description	Started by a 50-ms interrupt (INTIT) of the 12-bit interval timer, sets TRUTH (1) to variable g_50ms_status.
Arguments	None
Return values	None
Remarks	None

The following functions are provided by the IIC00 hardware control library.

[Function name] R_ IIC00_TM03_Init

Summary	Starts IIC00 and TM03.	
Header	r_cg_macrodriver.h, r_cg_userdefine.h, r_iic_lib.h	
Declaration	void R_IIC00_TM03_Init(void);	
Description	Enables IIC00 interrupts and sets TM03 to wait for a trigger.	
Arguments	None	
Return values	None	
Remarks	None	

[Function name] R_IIC00_MasterSend

Summary	Starts transmission of the specified data to slave.	
Header	r_cg_macrodriver.h, r_cg_userdefine.h, r_ iic_lib.h	
Declaration	MD_STATUS R_IIC00_MasterSend(uint8_t adr, uint8_t * const tx_buf, uint16_t tx_num);	
Description	Issues the start condition with arguments set to the control variables and transmits the slave address.	
Arguments	1st argument	Slave address
	2nd argument	Pointer to transmission data
	3rd argument	Number of data to be transmitted
Return values	MD_OK (0x00)	
Remarks	Variable g_iic00_status is IIC_USING (0x01).	

[Function name] R_IIC00_Master Receive

Summary	Starts data reception from slave.	
Header	r_cg_macrodriver.h, r_cg_userdefine.h, r_ iic_lib.h	
Declaration	MD_STATUS R_IIC00_MasterReceive(uint8_t adr, uint8_t * const rx_buf, uint16_t rx_num);	
Description	Issues the start condition with arguments set to the control variables and transmits the slave address.	
Arguments	1st argument	Slave address
	2nd argument	Pointer to reception data
	3rd argument	Number of data to be received
Return values	MD_OK (0x00)	
Remarks	Variable g_iic00_status is IIC_RX_SVADDR (0x81).	

[Function name] R_IIC00_ WaitComend

Summary	Waits for completion of I2C bus communication.
Header	r_cg_macrodriver.h, r_cg_userdefine.h, r_ iic_lib.h
Declaration	MD_STATUS R_IIC00_WaitComend(void);
Description	Waits for completion of I2C bus in HALT mode and returns the communication result. If an error occurs, issues the stop condition to release the I2C bus, stops IIC00, and then returns.
Arguments	None
Return values	[IIC_SUCCESS]: Communication completed [IIC_ERROR]: Communication failed (NACK response)
Remarks	None

[Function name] R_IIC00_CheckComstate

Summary	Checks the I2C bus communication state.
Header	r_cg_macrodriver.h, r_cg_userdefine.h, r_ iic_lib.h
Declaration	MD_STATUS R_IIC00_CheckComstate(void);
Description	Returns the I2C bus communication state.
Arguments	None
Return values	IIC_SUCCESS (0x00): Process successfully completed IIC_USING (0x01): Currently processing master transmission IIC_RX_MODE (0x41): Currently processing master reception IIC_RX_SVADDR (0x81): Currently transmitting a slave address for master reception IIC_ERROR (0xFF): Communication error
Remarks	None

[Function name] R_IIC00_StopCondition

Summary	Issues the stop condition to I2C bus.
Header	r_cg_macrodriver.h, r_cg_userdefine.h, r_ iic_lib.h
Declaration	MD_STATUS R_IIC00_StopCondition(void);
Description	Stops IIC00 and issues the stop condition.
Arguments	None
Return values	MD_OK (0x00)
Remarks	To support standard mode, SCL and SDA are controlled at 5-us intervals.

[Function name] R_TM03_20us

Summary	Waits for 20 us.
Header	r_cg_macrodriver.h, r_cg_userdefine.h
Declaration	void R_TM03_20us(void);
Description	Waits for 20 us by using the TM03 delay counting function.
Arguments	None
Return values	None
Remarks	None

[Function name] r_iic00_ startcondition

Summary	Issues the start condition to I2C bus.
Header	r_cg_userdefine.h
Declaration	void r_iic00_ startcondition (void);
Description	Issues the start condition and starts IIC00 in transmission mode.
Arguments	None
Return values	None
Remarks	To support standard mode, SCL and SDA are controlled at 5-us intervals.

[Function name] r_iic00_stop

Summary	Stops IIC00 operation.
Header	r_cg_macrodriver.h, r_cg_userdefine.h
Declaration	void r_iic00_stop (void);
Description	Stops IIC00 operation to issue the start or stop condition. During communication, waits for communication completion before stopping operation.
Arguments	None
Return values	None
Remarks	None

[Function name] wait_5us

Summary	Waits for 5 us.
Header	r_cg_userdefine.h
Declaration	void wait_5us (void);
Description	Waits for 5 us by using the TM03 delay counting function.
Arguments	None
Return values	None
Remarks	None

[Function name] r_iic00_interrupt

Summary	Performs an IIC00 communication completion process.
Header	r_cg_macrodriver.h, r_cg_userdefine.h
Declaration	#pragma interrupt r_iic00_interrupt(vect=INTIIC00,bank=RB1) static void r_iic00_interrupt(void)
Description	Started by the IIC00 communication end interrupt (INTIIC00); starts a time wait (TM03) to secure the processing time for the slave.
Arguments	None
Return values	None
Remarks	Process is actually performed using INTTM03.

[Function name] r_inttm03_interrupt

Summary	Performs an IIC00 communication completion process.
Header	r_cg_macrodriver.h, r_cg_userdefine.h
Declaration	#pragma interrupt r_inttm03_interrupt(vect=INTTM03,bank=RB2) static void r_inttm03_interrupt (void)
Description	Started by the IIC00 communication end interrupt (INTIIC00); waits for the specified time to secure the processing time for the slave, and when the specified time elapses, performs the IIC00 communication completion process.
Arguments	None
Return values	None
Remarks	The result is stored in the variable g_iic00_status. The I2C bus is not released even when communication is completed.

5.6 Flowcharts

Figure 5.1 shows the overall flow of the process described in this application note.

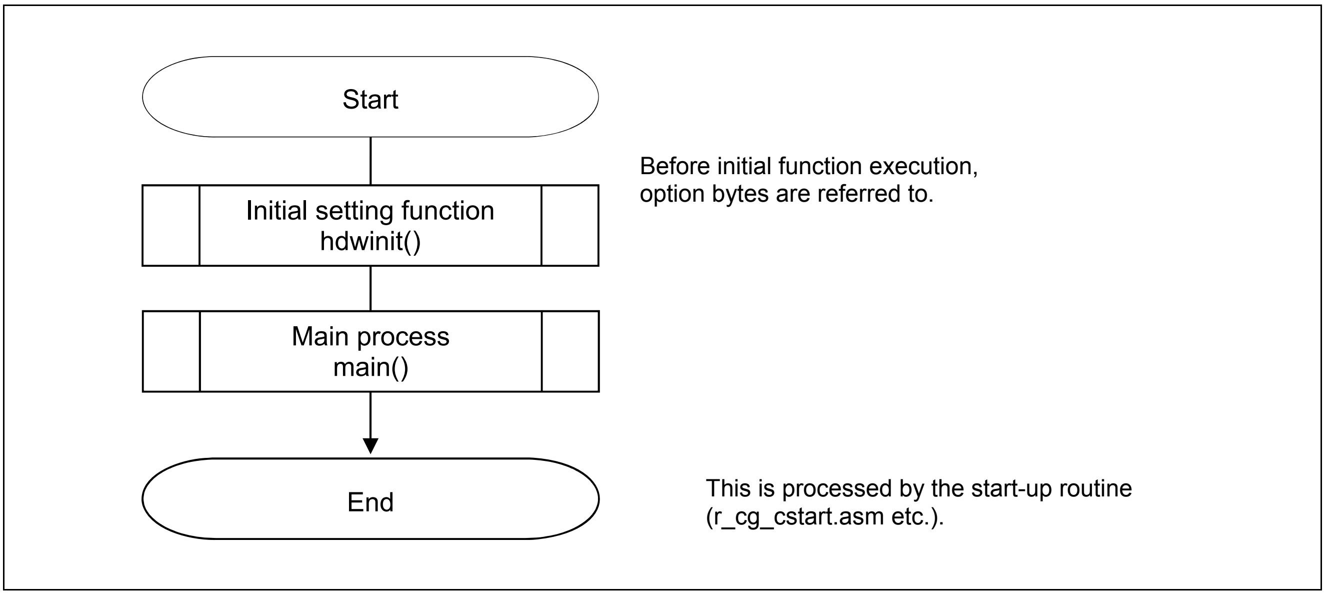

Start
Initial setting function
hdwinit()
Main process
main()
End
Before initial function execution,
option bytes are referred to.
This is processed by the start-up routine
(r_cg_cstart.asm etc.).

Figure 5.1 Overall Flow

Note: This is processed by the start-up routine (r_cg_cstart.asm etc.). The memory-related settings are made between calling the initial setting function and main process function.

5.6.1 Initial Setting Function

Figure 5.2 shows the flowchart of the initial setting function.

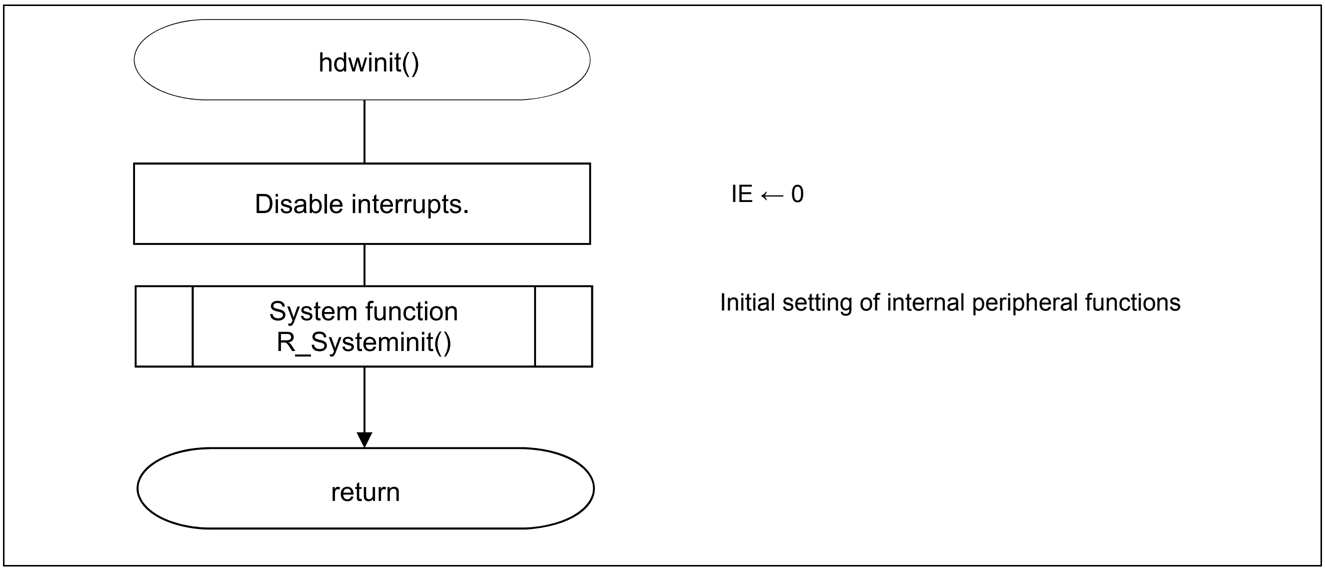


Figure 5.2 Initial Setting Function

5.6.2 System Function

Figure 5.3 shows the flowchart of the system function.

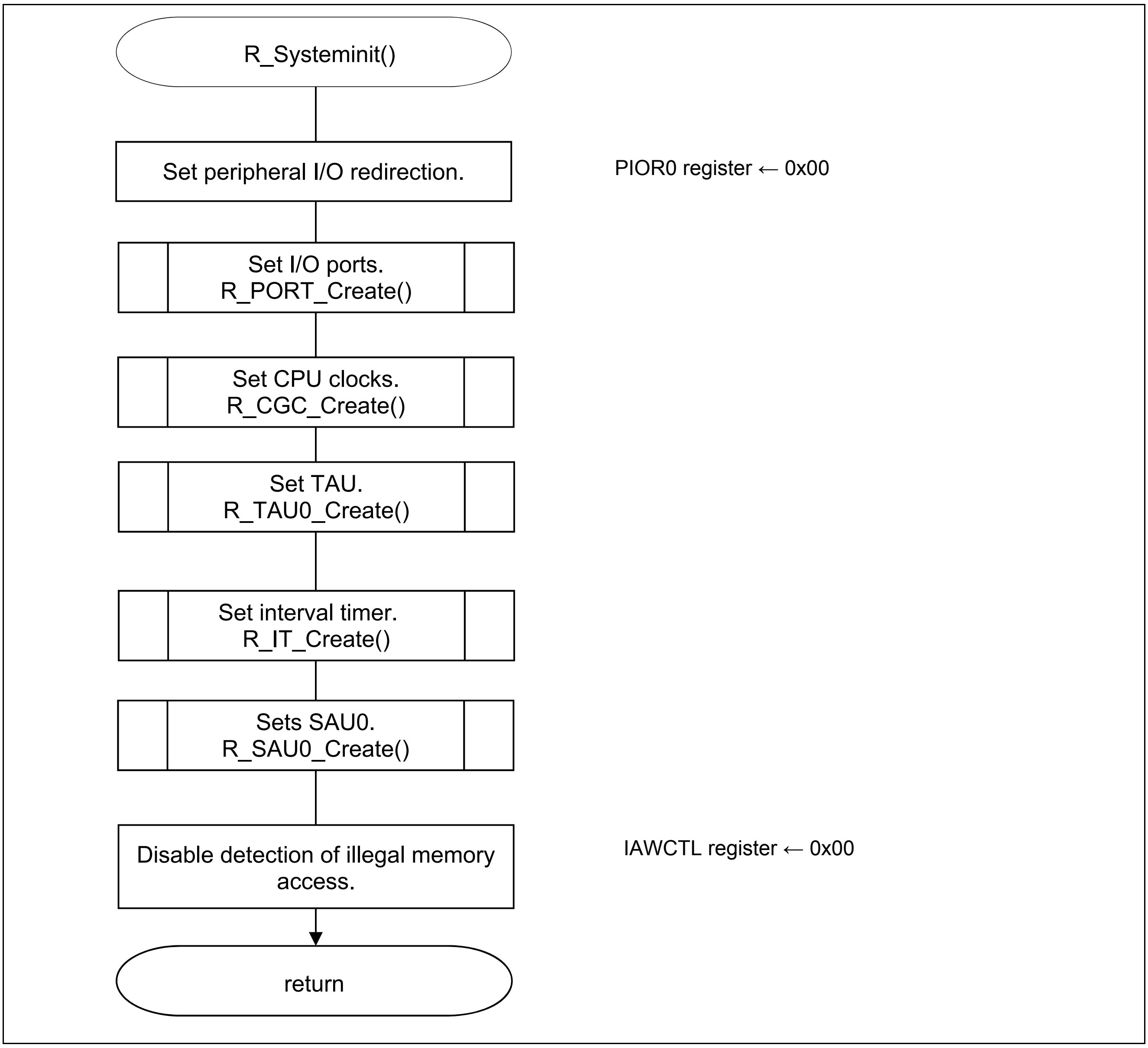


Figure 5.2 System Function

5.6.3 Setting CPU Clocks

Figure 5.4 shows the flowchart for setting the CPU clocks.

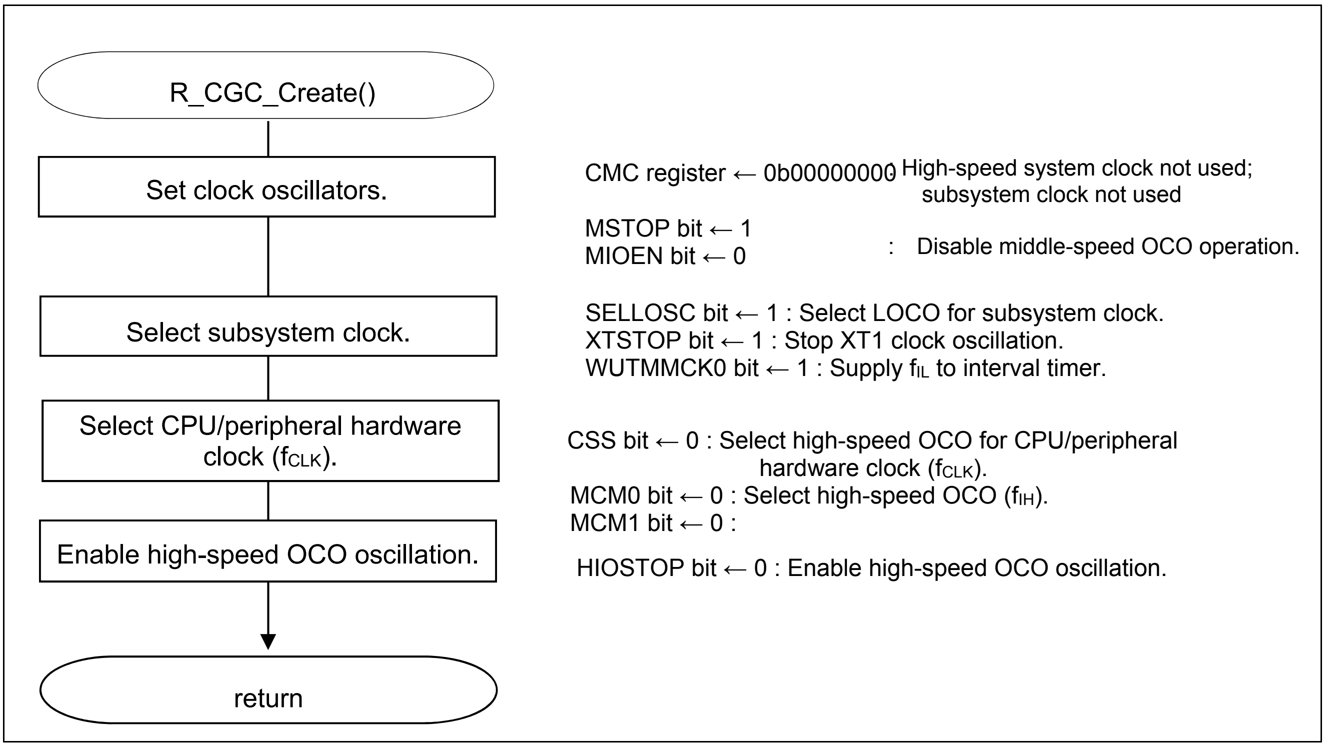

R_CGC_Create()
Set clock oscillators.
Select subsystem clock.
Select CPU/peripheral hardware clock (f_{CLK}).
Enable high-speed OCO oscillation.
return
CMC register ← 0b00000000 : High-speed system clock not used; subsystem clock not used
MSTOP bit ← 1
MIOEN bit ← 0 : Disable middle-speed OCO operation.
SELLOSC bit ← 1 : Select LOCO for subsystem clock.
XTSTOP bit ← 1 : Stop XT1 clock oscillation.
WUTMMCK0 bit ← 1 : Supply f_{IL} to interval timer.
CSS bit ← 0 : Select high-speed OCO for CPU/peripheral hardware clock (f_{CLK}).
MCM0 bit ← 0 : Select high-speed OCO (f_{IH}).
MCM1 bit ← 0 :
HIOSTOP bit ← 0 : Enable high-speed OCO oscillation.

Figure 5.4 Setting CPU Clocks

5.6.4 Setting I/O Ports

Figure 5.5 shows the flowchart for setting the I/O ports.

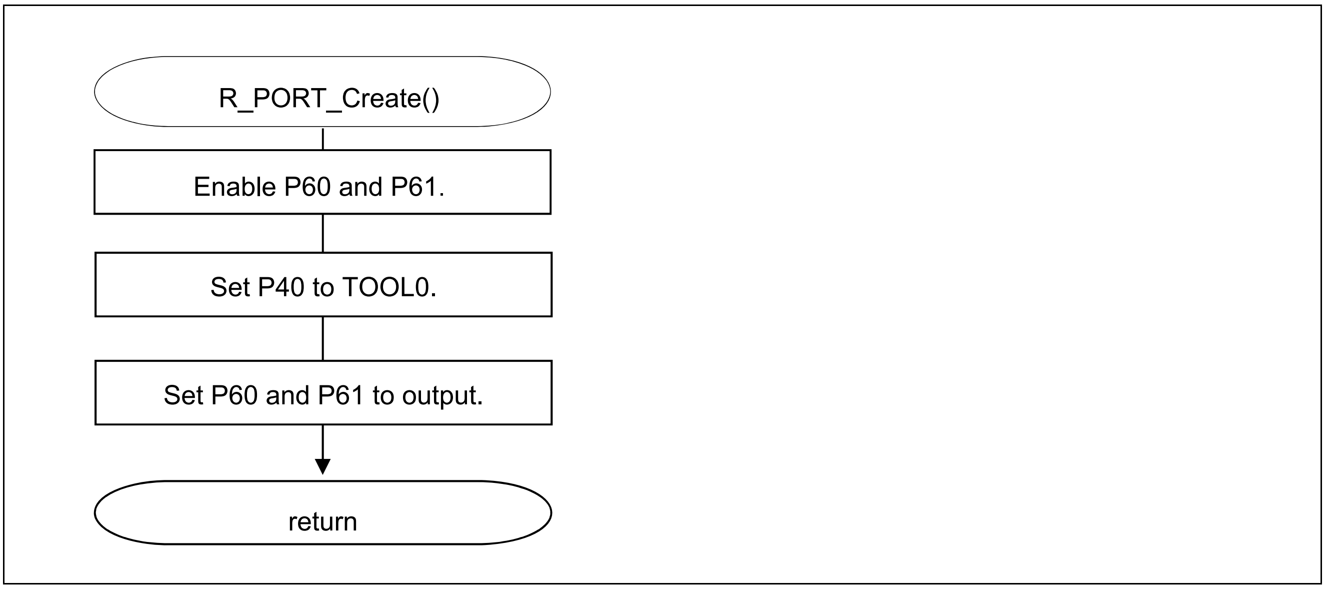


Figure 5.5 Setting I/O Ports

Note: Design unused ports so that the electrical characteristics are satisfied by appropriately treating the pertinent pins. Separately connect unused input-only pins to V_{DD} or V_{SS} via a resistor.

5.6.5 Setting SAU0

Figure 5.6 shows the flowchart for setting the SAU0.

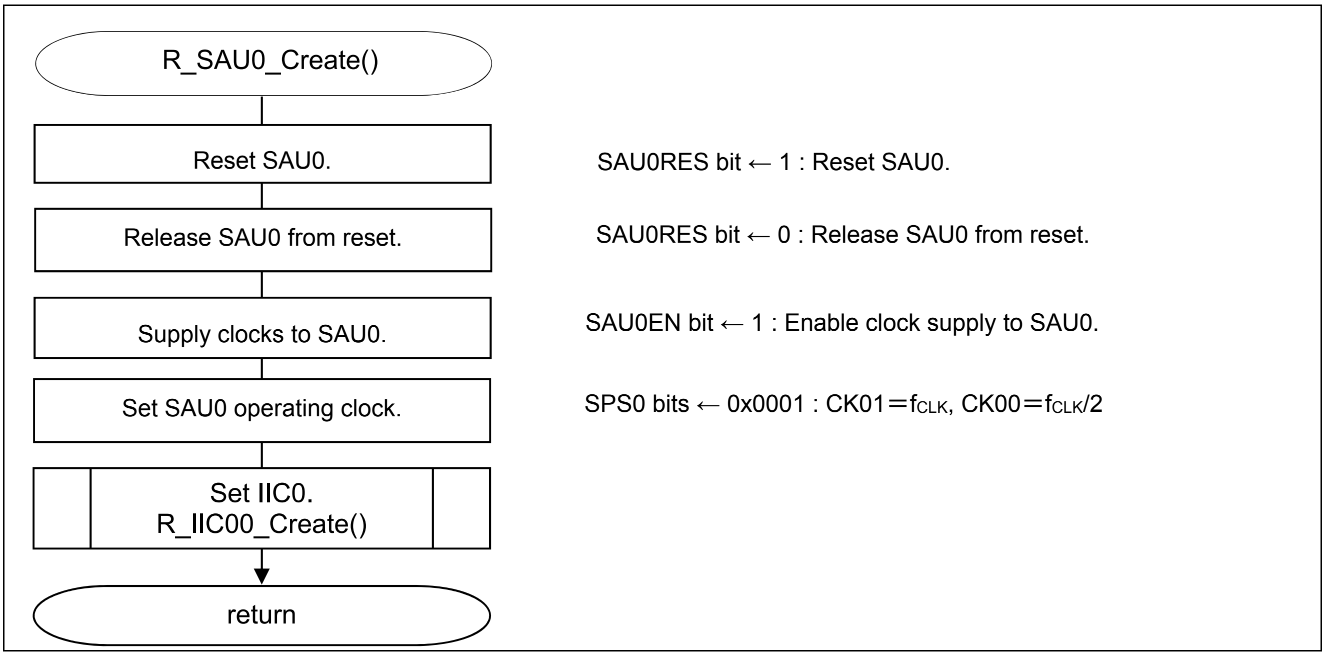

R_SAU0_Create()
Reset SAU0.
Release SAU0 from reset.
Supply clocks to SAU0.
Set SAU0 operating clock.
Set IIC0.
R_IIC00_Create()
return
SAU0RES bit ← 1 : Reset SAU0.
SAU0RES bit ← 0 : Release SAU0 from reset.
SAU0EN bit ← 1 : Enable clock supply to SAU0.
SPS0 bits ← 0x0001 : CK01=fCLK, CK00=fCLK/2

Figure 5.6 Setting SAU0

Resetting SAU0

- Peripheral reset control register 0 (PRR0)
  Reset SAU0.

Symbol: PRR0

7	6	5	4	3	2	1	0
0	0	ADCRES	0	0	SAU0RES	0	TAU0RES
0	0	x	0	0	**1/0**	0	x

Bit 2

SAU0RES	Control of a reset of serial array unit
0	**Releases the serial array unit from the reset state.**
1	**Resets serial array unit.**

Starting clock supply to SAU0

- Peripheral enable register 0 (PER0)
  Start clock supply to SAU0.

Symbol: PER0

7	6	5	4	3	2	1	0
RTCWEN	0	ADCEN	0	0	SAU0EN	0	TAU0EN
x	0	x	x	0	**1**	0	x

Bit 4

SAU0EN	Control of input clock to SAU0
0	Stops input clock supply.
1	**Supplies input clock.**

Setting SAU0 operating clocks

- Serial clock select register 0 (SPS0)

  Set CK00 and CK01.

Symbol: SPS0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	PRS 013	PRS 012	PRS 011	PRS 010	PRS 003	PRS 002	PRS 001	PRS 000
0	0	0	0	0	0	0	0	x	x	x	x	**0**	**0**	**0**	**0**

Bits 3 to 0

PRS 003	PRS 002	PRS 001	PRS 000	Operating clock (CK00) selection					
					f_{CLK}= 1MHz	f_{CLK}= 2MHz	f_{CLK}= 4MHz	f_{CLK}= 16MHz	**f_{CLK}= 24MHz**
0	**0**	**0**	**0**	**f_{CLK}**	1 MHz	2 MHz	4 MHz	16MHz	**24 MHz**
0	0	0	1	$f_{CLK}/2$	500 kHz	1 MHz	2MHz	8 MHz	12 MHz
0	0	1	0	$f_{CLK}/2^{2}$	250 kHz	500 kHz	1 MHz	4MHz	6 MHz
0	0	1	1	$f_{CLK}/2^{3}$	125 kHz	250 kHz	500 kHz	2 MHz	3 MHz
0	1	0	0	$f_{CLK}/2^{4}$	62.5 kHz	125 kHz	250 kHz	1 MHz	1.5 MHz
0	1	0	1	$f_{CLK}/2^{5}$	31.3 kHz	62.5 kHz	125 kHz	500 kHz	750 kHz
0	1	1	0	$f_{CLK}/2^{6}$	15.6 kHz	31.3 kHz	62.5 kHz	250 kHz	375 kHz
0	1	1	1	$f_{CLK}/2^{7}$	7.81 kHz	15.6 kHz	31.3 kHz	125 kHz	187.5 kHz
1	0	0	0	$f_{CLK}/2^{8}$	3.91 kHz	7.81 kHz	15.6 kHz	62.5 kHz	93.8 kHz
1	0	0	1	$f_{CLK}/2^{9}$	1.95 kHz	3.91 kHz	7.81 kHz	31.3 kHz	46.9 kHz
1	0	1	0	$f_{CLK}/2^{10}$	977 Hz	1.95 kHz	3.91 kHz	15.6 kHz	23.4 kHz
1	0	1	1	$f_{CLK}/2^{11}$	488 Hz	977 Hz	1.95 kHz	7.81 kHz	11.7 kHz
1	1	0	0	$f_{CLK}/2^{12}$	244 Hz	488 Hz	977 Hz	3.91 kHz	5.86 kHz
1	1	0	1	$f_{CLK}/2^{13}$	122 Hz	244 Hz	488 Hz	1.95 kHz	2.93 kHz
1	1	1	0	$f_{CLK}/2^{14}$	61 Hz	122 Hz	244 Hz	977 Hz	1.46 kHz
1	1	1	1	$f_{CLK}/2^{15}$	30.5 Hz	61 Hz	122 Hz	488 Hz	732 Hz

Note: For details of register settings, refer to the RL78/I1D User’s Manual: Hardware.

5.6.6 Setting IIC00

Figure 5.7 shows the flowchart for setting the IIC00.

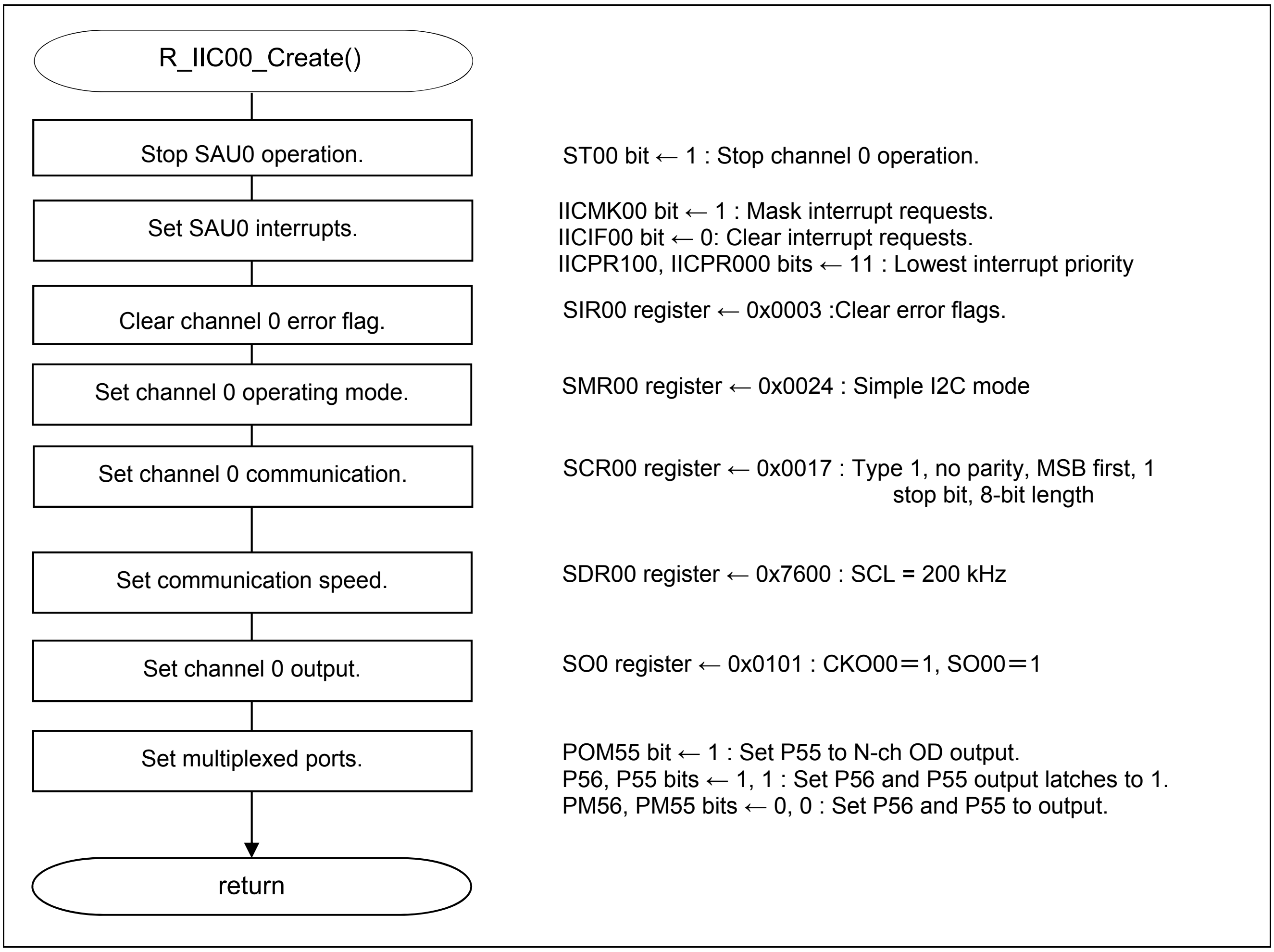

R_IIC00_Create()
Stop SAU0 operation.
Set SAU0 interrupts.
Clear channel 0 error flag.
Set channel 0 operating mode.
Set channel 0 communication.
Set communication speed.
Set channel 0 output.
Set multiplexed ports.
return
ST00 bit ← 1 : Stop channel 0 operation.
IICMK00 bit ← 1 : Mask interrupt requests.
IICIF00 bit ← 0: Clear interrupt requests.
IICPR100, IICPR000 bits ← 11 : Lowest interrupt priority
SIR00 register ← 0x0003 :Clear error flags.
SMR00 register ← 0x0024 : Simple I2C mode
SCR00 register ← 0x0017 : Type 1, no parity, MSB first, 1 stop bit, 8-bit length
SDR00 register ← 0x7600 : SCL = 200 kHz
SO0 register ← 0x0101 : CKO00=1, SO00=1
POM55 bit ← 1 : Set P55 to N-ch OD output.
P56, P55 bits ← 1, 1 : Set P56 and P55 output latches to 1.
PM56, PM55 bits ← 0, 0 : Set P56 and P55 to output.

Figure 5.7 Setting IIC00

Stopping SAU0 operation

- Serial channel stop register 0 (ST0)
  Stop channel 0 operation.

Symbol: ST0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	ST01	ST00
0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	**1**

Bit 0

ST00	Trigger for stopping channel 0 operation
0	**No triggering**
1	**Clears SE00 bit to 0 to stop communication.**

Controlling IIC00 interrupts

- IICMK00 bit in interrupt mask flag register (MK0H)
  Mask IIC00 interrupts.
- IICIF00 bit in interrupt request flag register (IF0H)
  Mask IIC00 interrupt requests.
- IICPR000 and IICPR010 bits in priority setting flag registers (PR00H, PR10H)
  Set IIC00 interrupts to lowest priority.

Symbol: MK0H

Bit 1

IICMK00	Control of INTIIC00 interrupts
0	Enables interrupt processing.
1	**Disables interrupt processing.**

Symbol: IF0H

Bit 1

IICIF00	INTIIC00 interrupt request flag
0	**No interrupt request signals have been generated.**
1	An interrupt request signal has been generated and interrupt is requested.

Symbol: PR00H, PR10H

Bit 1

IICPR010	IICPR000	INTIIC00 priority level selection
0	0	Level 0 (top priority)
0	1	Level 1
1	0	Level 2
1	**1**	**Level 3 (lowest priority)**

Clearing error flags

- Serial flag clear trigger register 00 (SIR00)
  Clear error flags.

Symbol: SIR00 (Bits 15 to 8 fixed to 0)

7	6	5	4	3	2	1	0
0	0	0	0	0	FECT00	PECT00	OVCT00
0	0	0	0	0	0	**1**	**1**

Bits 1 and 0

PECT00, OVCT00	Trigger for clearing error flags
0	Does not clear flags.
1	**Clears flags.**

Setting channel 0 operating mode

- Serial mode register 00 (SMR00)
  Set channel 0 to simple I2C.

Symbol: SMR00

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS 00	CCS 00	0	0	0	0	0	STS 00	0	SIS 00	1	0	0	MD 002	MD 001	MD 000
0	**0**	0	0	0	0	0	**0**	0	**0**	1	0	0	**1**	**0**	**0**

Bit 15

CKS00	Channel 0 operating clock (f_{MCK}) selection
0	**Operating clock CK00 specified with SPS0 register**
1	Operating clock CK01 specified with SPS0 register

Bit 14

CCS00	Channel 0 transfer clock (f_{TCLK}) selection
0	**Division of operating clock f_{MCK} specified with CKS00 bit**
1	Input clock f_{SCK} (slave transfer in CSI mode) from SCK00 pin

Bit 8

STS00	Start trigger source selection
0	**Only software trigger is valid.**
1	Valid edge of RxD0 pin (selected for UART reception)

Bit 6

SIS00	Control of received data level reversal of channel 0 in UART mode
0	**Input communication data is taken in as it is.**
1	Input communication data is taken in after reversed.

Bits 2 and 1

MD002	MD001	Channel 0 operating mode setting
0	0	CSI mode
0	1	UART mode
1	**0**	**Simple I2C mode**
1	1	Setting prohibited

Bit 0

MD000	Channel 0 interrupt source selection
0	**Transfer end interrupt**
1	Buffer empty interrupt

Note: For details of register settings, refer to the RL78/I1D User's Manual: Hardware.

Setting channel 0 communication operation

- Serial communication operation setting register 00(SCR00)
  Set type 1, no parity, MSB first, 1 stop bit, and 8-bit length.

Symbol: SCR00

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXE 00	RXE 00	DAP 00	CKP 00	0	EOC 00	PTC 001	PTC 000	DIR 00	0	SLC 001	SLC 000	0	1	DLS 001	DLS 000
0	**0**	**0**	**0**	0	**0**	**0**	**0**	**0**	0	**0**	**1**	0	1	**1**	**1**

Bits 15 and 14

TXE00	RXE00	Channel 0 operating mode setting
0	**0**	**Communication disabled**
0	1	Reception only
1	0	Transmission only
1	1	Reception and transmission

Bits 13 and 12

DAP00	CKP00	Data and clock phase selection in CSI mode
0	**0**	**Type 1**
0	1	Type 2
1	0	Type 3
1	1	Type 4

Bit 10

EOC00	Control of error interrupt signal (INTSRE0) mask
0	**Disables generation of error interrupt INTSRE0 (INTSR0 is generated).**
1	Enables generation of error interrupt INTSRE0 (INTSR0 is not generated upon error).

Bits 9 and 8

PTC001	PTC000	Parity bit setting in UART mode
0	**0**	**No parity**
0	1	0 parity
1	0	Even parity
1	1	Odd parity

Bit 7

DIR00	Data transfer direction selection in CSI, UART modes
0	**MSB-first input/output**
1	LSB-first input/output

Bits 5 and 4

SLC001	SLC000	Stop bit setting in UART mode
0	0	**No stop bits**
0	**1**	**Stop bit length = 1 bit**
1	0	**Stop bit length = 2 bits**
1	1	Setting prohibited

Symbol: SCR00

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXE00	RXE00	DAP00	CKP00	0	EOC00	PTC001	PTC000	DIR00	0	SLC001	SLC000	0	1	DLS001	DLS000
0	**0**	**0**	**0**	0	**0**	**0**	**0**	**0**	0	**0**	**1**	0	1	**1**	**1**

Bits 1 and 0

DLS001	DLS000	Data length setting In CSI, UART modes
0	0	Communication prohibited
0	1	9-bit data length
1	0	7-bit data length
1	**1**	8-bit data length

Setting communication speed

- Serial data register (SDR00)

  Set SCL to 200 kHz.

Symbol: SDR00

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	**1**	**1**	**1**	**0**	**1**	**1**	**0**	x	x	x	x	x	x	x	x

Setting channel 0 output

- Serial output register 0 (SO0)

  Set the output levels of SCL00 and SDA00 signals.

Symbol: SO0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	CKO01	CKO00	0	0	0	0	0	0	SO01	SO00
0	0	0	0	0	0	x	**1**	0	0	0	0	0	0	x	**1**

Bit 8

CKO00	Channel 0 serial clock output
0	Serial clock output value is 0.
1	**Serial clock output value is 1.**

Bit 0

SO00	Channel 0 serial data output
0	Serial data output value is 0.
1	**Serial data output value is 1.**

Setting multiplexed ports

- Port output mode register 5 (POM5)

  Set ports to N-ch open drain output.
- Port register 5 (P5)

  Set 1 to output latches.
- Port mode register 5 (PM5)

  Set ports to output ports.

Symbol: POM5

7	6	5	4	3	2	1	0
0	POM56	POM55	POM54	POM53	POM52	POM51	0
0	x	**1**	x	x	x	x	0

Bit 5

POM55	P55 pin output mode selection
0	通常出力モード
1	**N-ch open drain output**

Symbol: P5

7	6	5	4	3	2	1	0
P57	P56	P55	P54	P53	P52	P51	P50
x	**1**	**1**	x	x	x	x	x

Bits 6 and 5

P56, P55	P56 and P55 output latch setting
0	0 を設定
1	**Sets 1.**

Symbol: PM5

7	6	5	4	3	2	1	0
PM57	PM56	PM55	PM54	PM53	PM52	PM51	PM50
x	**0**	**0**	x	x	x	x	x

Bits 6 and 5

PM56, PM55	P56 and P55 pin I/O mode selection
0	**Output mode**
1	入力モード

Note: For details of register settings, refer to the RL78/I1D User’s Manual: Hardware.

5.6.7 Setting Timer Array Unit

Figure 5.8 shows the flowchart for setting the timer array unit.

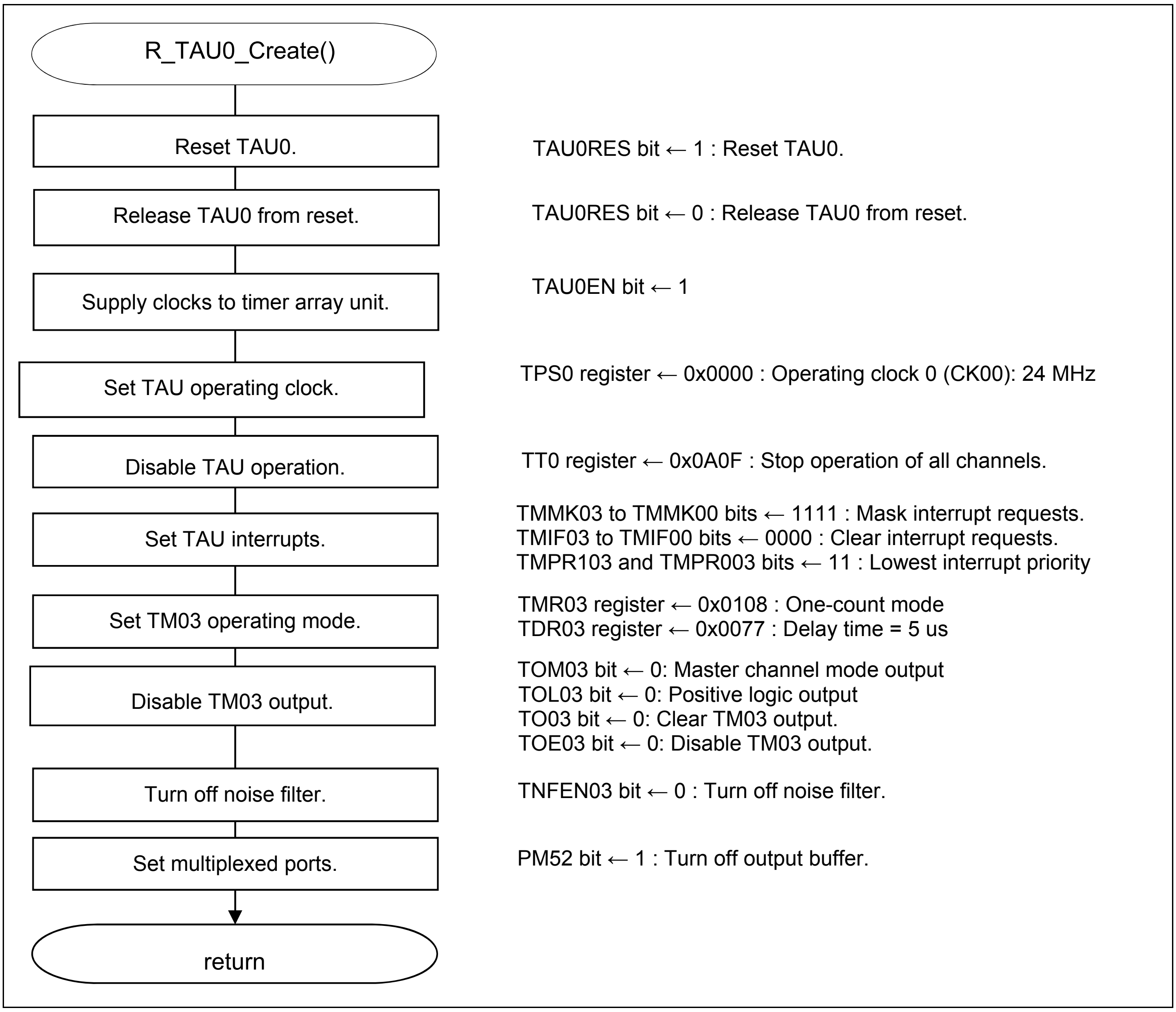

R_TAU0_Create()
Reset TAU0.
TAU0RES bit ← 1 : Reset TAU0.
Release TAU0 from reset.
TAU0RES bit ← 0 : Release TAU0 from reset.
Supply clocks to timer array unit.
TAU0EN bit ← 1
Set TAU operating clock.
TPS0 register ← 0x0000 : Operating clock 0 (CK00): 24 MHz
Disable TAU operation.
TT0 register ← 0x0A0F : Stop operation of all channels.
Set TAU interrupts.
TMMK03 to TMMK00 bits ← 1111 : Mask interrupt requests.
TMIF03 to TMIF00 bits ← 0000 : Clear interrupt requests.
TMPR103 and TMPR003 bits ← 11 : Lowest interrupt priority
Set TM03 operating mode.
TMR03 register ← 0x0108 : One-count mode
TDR03 register ← 0x0077 : Delay time = 5 us
Disable TM03 output.
TOM03 bit ← 0: Master channel mode output
TOL03 bit ← 0: Positive logic output
TO03 bit ← 0: Clear TM03 output.
TOE03 bit ← 0: Disable TM03 output.
Turn off noise filter.
TNFEN03 bit ← 0 : Turn off noise filter.
Set multiplexed ports.
PM52 bit ← 1 : Turn off output buffer.
return

Figure 5.8 Setting Timer Array Unit

Resetting TAU0

- Peripheral reset control register 0 (PRR0)
  Reset TAU0.

Symbol: PRR0

7	6	5	4	3	2	1	0
0	0	ADCRES	0	0	SAU0RES	0	TAU0RES
0	0	x	0	0	x	0	**1/0**

Bit 0

SAU0RES	Reset control of timer array unit
0	**Releases the timer array unit from the reset state.**
1	**Resets timer array unit.**

Starting clock supply to timer array unit 0

- Peripheral enable register 0 (PER0)
  Start clock supply to timer array unit 0.

Symbol: PER0

7	6	5	4	3	2	1	0
RTCWEN	0	ADCEN	0	0	SAU0EN	0	TAU0EN
x	0	x	0	0	x	0	**1**

Bit 0

TAU0EN	Control of input clock to timer array unit 0
0	入力クロック供給停止
1	**Supplies input clock.**

Setting timer clock frequency

- Timer clock select register 0 (TPS0)
  Select the operating clock for timer array unit 0.

Symbol: TPS0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	PRS 031	PRS 030	0	0	PRS 021	PRS 020	PRS 013	PRS 012	PRS 011	PRS 010	PRS 003	PRS 002	PRS 001	PRS 000
0	0	x	x	0	0	x	x	x	x	x	x	**0**	**0**	**0**	**0**

Bits 3 to 0

PRS 003	PRS 002	PRS 001	PRS 000	Operating clock (CK00) selection					
					f_{CLK}= 1MHz	f_{CLK}= 2MHz	f_{CLK}= 4MHz	f_{CLK}= 16MHz	**f_{CLK}= 24MHz**
0	**0**	**0**	**0**	**f_{CLK}**	1 MHz	2 MHz	4 MHz	16MHz	**24 MHz**
0	0	0	1	$f_{CLK}/2$	500 kHz	1 MHz	2 MHz	8 MHz	12 MHz
0	0	1	0	$f_{CLK}/2^2$	250 kHz	500 kHz	1 MHz	4MHz	6 MHz
0	0	1	1	$f_{CLK}/2^3$	125 kHz	250 kHz	500 kHz	2 MHz	3 MHz
0	1	0	0	$f_{CLK}/2^4$	62.5 kHz	125 kHz	250 kHz	1 MHz	1.5 MHz
0	1	0	1	$f_{CLK}/2^5$	31.3 kHz	62.5 kHz	125 kHz	500 kHz	750 kHz
0	1	1	0	$f_{CLK}/2^6$	15.6 kHz	31.3 kHz	62.5 kHz	250 kHz	375 kHz
0	1	1	1	$f_{CLK}/2^7$	7.81 kHz	15.6 kHz	31.3 kHz	125 kHz	187.5 kHz
1	0	0	0	$f_{CLK}/2^8$	3.91 kHz	7.81 kHz	15.6 kHz	62.5 kHz	93.8 kHz
1	0	0	1	$f_{CLK}/2^9$	1.95 kHz	3.91 kHz	7.81 kHz	31.3 kHz	46.9 kHz
1	0	1	0	$f_{CLK}/2^{10}$	977 Hz	1.95 kHz	3.91 kHz	15.6 kHz	23.4 kHz
1	0	1	1	$f_{CLK}/2^{11}$	488 Hz	977 Hz	1.95 kHz	7.81 kHz	11.7 kHz
1	1	0	0	$f_{CLK}/2^{12}$	244 Hz	488 Hz	977 Hz	3.91 kHz	5.86 kHz
1	1	0	1	$f_{CLK}/2^{13}$	122 Hz	244 Hz	488 Hz	1.95 kHz	2.93 kHz
1	1	1	0	$f_{CLK}/2^{14}$	61 Hz	122 Hz	244 Hz	977 Hz	1.46 kHz
1	1	1	1	$f_{CLK}/2^{15}$	30.5 Hz	61 Hz	122 Hz	488 Hz	732 Hz

Note: For details of register settings, refer to the RL78/I1D User's Manual: Hardware.

Stopping timer operation

- Timer channel stop register 0 (TT0)
  Select to stop timer channel operation.

Symbol: TT0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	TT03H	0	TT01H	0	0	0	0	0	TT03	TT02	TT01	TT00
0	0	0	0	**1**	0	**1**	0	0	0	0	0	**1**	**1**	**1**	**1**

Bit n

TT0n	Trigger for stopping channel n operation
0	トリガ動作しない
1	**Clears TE0n bit to 0 to stop counting operation (stop-trigger generated).**

Setting timer count end interrupt

- TMMK03 bit in Interrupt mask flag register (MK1L)
  Set interrupt masks.
- TMIF03 bit in interrupt request flag register (IF1L)
  Clear interrupt request flags.
- TMPR003 and TMPR103 bits in priority setting flag registers (PR01L, PR11L)
  Set TM03 interrupts to lowest priority.

Symbol: MK1L

Bit 5

TMMK03	Control of interrupt processing
0	割り込み処理許可
1	**Disables interrupt processing.**

Symbol: IF1L

Bit 5

TMIF03	Interrupt request flag
0	**No interrupt request signals have been generated.**
1	割り込み要求信号が発生し、割り込み要求状態

Symbol: PR01L、PR11L

Bit 5

TMPR103	TMPR003	INTTM03 priority level selection
0	0	レベル 0 を指定(高優先順位)
0	1	レベル 1 を指定
1	0	レベル 2 を指定
1	**1**	**Level 3 (lowest priority)**

Note: For details of register settings, refer to the RL78/I1D User's Manual: Hardware.

Setting channel 3 operating mode

- Timer mode register 03 (TMR03)

  Select operating clock (f_{MCK}).

  Select the counting clock.

  Set start and capture triggers.

  Select valid edge of timer input.

  Set the operating mode.

Symbol: TMR03

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS031	CKS030	0	CCS03	SPLIT03	STS032	STS031	STS030	CIS031	CIS030	0	0	MD033	MD032	MD031	MD030
0	**0**	0	**0**	**0**	**0**	**0**	**1**	**0**	**0**	0	0	**1**	**0**	**0**	**0**

Bits 15 and 14

CKS031	CKS030	Channel 3 operating clock (f_{MCK}) selection
0	**0**	**Operating clock CK00 specified with timer clock selection register 0 (TPS0)**
0	1	タイマ・クロック選択レジスタ 0（TPS0）で設定した動作クロック CK02
1	0	タイマ・クロック選択レジスタ 0（TPS0）で設定した動作クロック CK01
1	1	タイマ・クロック選択レジスタ 0（TPS0）で設定した動作クロック CK03

Bit 12

CCS03	Channel 3 counting clock (f_{TCL}) selection
0	**Operating clock (f_{MCK}) specified with CKS031 and CKS030 bits**
1	TI03 端子からの入力信号の有効エッジ

Bit 11

SPLIT03	8-bit timer/16-bit timer operation selection of channel 3
0	**16-bit timer operation**
1	8 ビット・タイマとして動作

Bits 10 to 8

STS032	STS031	STS030	Setting for channel 3 start and capture triggers
0	0	0	ソフトウエア・トリガ・スタートのみ有効（他のトリガ要因を非選択にする）
0	**0**	**1**	**Valid edge of TI00 pin input is used for both the start and capture triggers.**
0	1	0	TI00 端子入力の両エッジを、スタート・トリガとキャプチャ・トリガに分けて使用
1	0	0	マスタ・チャネルの割り込み信号を使用（複数チャネル連動動作機能のスレーブ・チャネル時）
上記以外			設定禁止

Note: For details of register settings, refer to the RL78/I1D User’s Manual: Hardware.

Bits 7 and 6

CIS031	CIS030	Valid edge selection of TI03 pin
0	**0**	**Falling edge**
0	1	立ち上がりエッジ
1	0	両エッジ（ロウ・レベル幅測定時） スタート・トリガ：立ち下がりエッジ、キャプチャ・トリガ：立ち上がりエッジ
1	1	両エッジ（ハイ・レベル幅測定時） スタート・トリガ：立ち上がりエッジ、キャプチャ・トリガ：立ち下がりエッジ

Bits 3 to 0

MD 033	MD 032	MD 031	MD 030	Channel 3 operating mode	Corresponding functions	TCR counting operation
0	0	0	1/0	インターバル・タイマ・モード	インターバル・タイマ／方形波出力／分周器機能／PWM 出力（マスタ）	ダウン・カウント
0	1	0	1/0	キャプチャ・モード	入力パルス間隔測定	アップ・カウント
0	1	1	0	イベント・カウンタ・モード	外部イベント・カウンタ	ダウン・カウント
1	**0**	**0**	1/**0**	**One-count mode**	Delay counter/one-shot pulse output/PWM output (slave)	Decrementing
1	1	0	0	キャプチャ＆ワンカウント・モード	入力信号のハイ／ロウ・レベル幅測定	アップ・カウント
上記以外				設定禁止		

Setting delay time

- Timer data register 03 (TDR03)
  Set the delay time.

Symbol: TDR03

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**0**	**1**	**1**	**1**

Note: For details of register settings, refer to the RL78/I1D User's Manual: Hardware.

Disabling timer output

- Timer output mode register 0 (TOM0L)
  Set master mode output.
- Timer output level register 0 (TOL0L)
  Set positive logic output.
- Timer output register 0 (TO0L)
  Set output to 0.
- Timer output enable register 0 (TOE0L)
  Enable/disable timer output of each channel.

Symbol: TOM0L

7	6	5	4	3	2	1	0
0	0	0	0	TOM03	TOM02	TOM01	0
0	0	0	0	**0**	x	x	0

Bit 3

TOM03	Control of channel 3 timer output mode
0	**Master channel output mode**
1	スレーブ・チャネル出力モード

Symbol: TOL0L

7	6	5	4	3	2	1	0
0	0	0	0	TOL03	TOL02	TOL01	0
0	0	0	0	**0**	x	x	0

Bit 3

TOL03	Control of channel 3 timer output level TO03
0	**Positive logic output (active high)**
1	反転出力(アクティブ・ロウ)

Symbol: TO0L

7	6	5	4	3	2	1	0
0	0	0	0	TO03	TO02	TO01	TO00
0	0	0	0	**0**	x	x	x

Bit 3

TO03	Control of channel 3 timer output level
0	**Low**
1	ハイ・レベル

Symbol: TOE0L

7	6	5	4	3	2	1	0
0	0	0	0	TOE03	TOE02	TOE01	TOE00
0	0	0	0	**0**	x	x	x

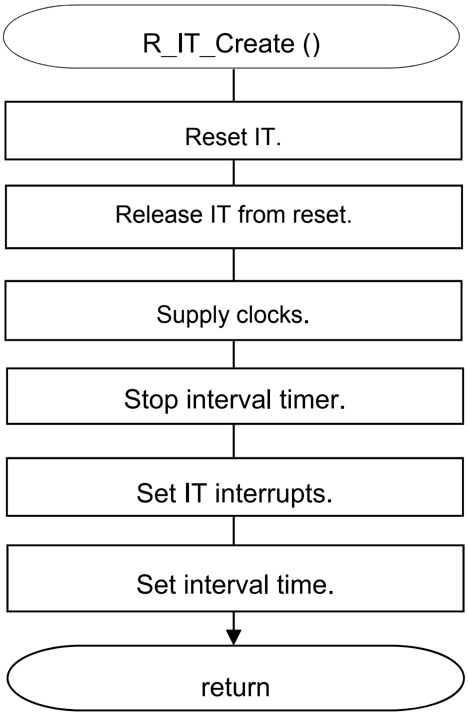
Bit 3

TOE03	Control of channel 3 timer output enable/disable
0	**Disables TO03 (timer channel output bit) operation triggered by counting operation.**
1	カウント動作による TO03（タイマ・チャネル出力ビット）の動作許可。

Note: For details of register settings, refer to the RL78/I1D User’s Manual: Hardware.

5.6.8 Setting Interval Timer

Figure 5.9 shows the flowchart for setting the interval timer.



TMKARES bit ← 1 : Reset interval timer.

TMKARES bit ← 0 : Release reset.

TMKAEN bit ← 1 : Enable clock supply.

RINTE bit ← 0 : Stop counter operation.

TMKAMK bit ← 1 : Mask interrupt requests.
TMKAIF bit ← 0 : Clear interrupt requests.
TMKAPR1 and TMKAPR0 bits ← 11 : Lowest interrupt priority.

ITMC register ← 0x02ED : 50 ms

Figure 5.9 Setting Interval Timer

Resetting interval timer

- Peripheral reset control register 2 (PRR2)
  Reset the interval timer.

Symbol: PRR2

7	6	5	4	3	2	1	0
TMKARES	0	DOCRES	0	0	0	0	0
1/0	0	x	0	0	0	0	0

Bit 7

TMKARES	Reset control of interval timer
0	**Releases the interval timer from the reset state.**
1	**Resets interval time.**

Starting clock supply to interval timer

- Peripheral enable register 2 (PER2)
  Start clock supply to the interval timer.

Symbol: PER2

7	6	5	4	3	2	1	0
TMKAEN	FMCEN	DOEN	0	0	0	0	0
1	x	x	0	0	0	0	0

Bit 7

TMKAEN	Control of input clock to 12-bit interval timer
0	入力クロック供給停止
1	**Supplies input clock.**

Stopping interval timer operation

- 12-bit interval timer control register (ITMC)
  Stop interval timer operation.

Symbol: ITMC

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RINTE	0	0	0	ITCMP11～ITCMP0											
0	0	0	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

Bit 15

RINTE	Control of 12-bit interval timer operation
0	**Stops counter operation (clears count).**
1	カウンタ動作開始

Controlling interval timer interrupts

- TMKAMK bit in interrupt mask flag register (MK1H)
  Mask interval timer interrupts.
- TMKAIF bit in interrupt request flag register (IF1H)
  Clear interval timer interrupt requests.
- TMKAPR0 and TMKAPR1 bits in priority setting flag registers (PR01H and PR11H) bits
  Set interval timer interrupts to lowest priority.

Symbol: MK1H

Bit 2

TMKAMK	Control of interval timer interrupts
0	割り込み処理許可
1	**Disables interrupt processing.**

Symbol: IF1H

Bit 2

TMKAIF	Interval timer interrupt request flag
0	**No interrupt request signals have been generated.**
1	割り込み要求信号が発生し，割り込み要求状態

Symbol: PR01H、PR11H

Bit 3

TMKAPR1	TMKAPR0	Interval timer priority level selection
0	0	レベル 0 を指定(高優先順位)
0	1	レベル 1 を指定
1	0	レベル 2 を指定
1	**1**	**Level 3 (lowest priority)**

Setting interval time

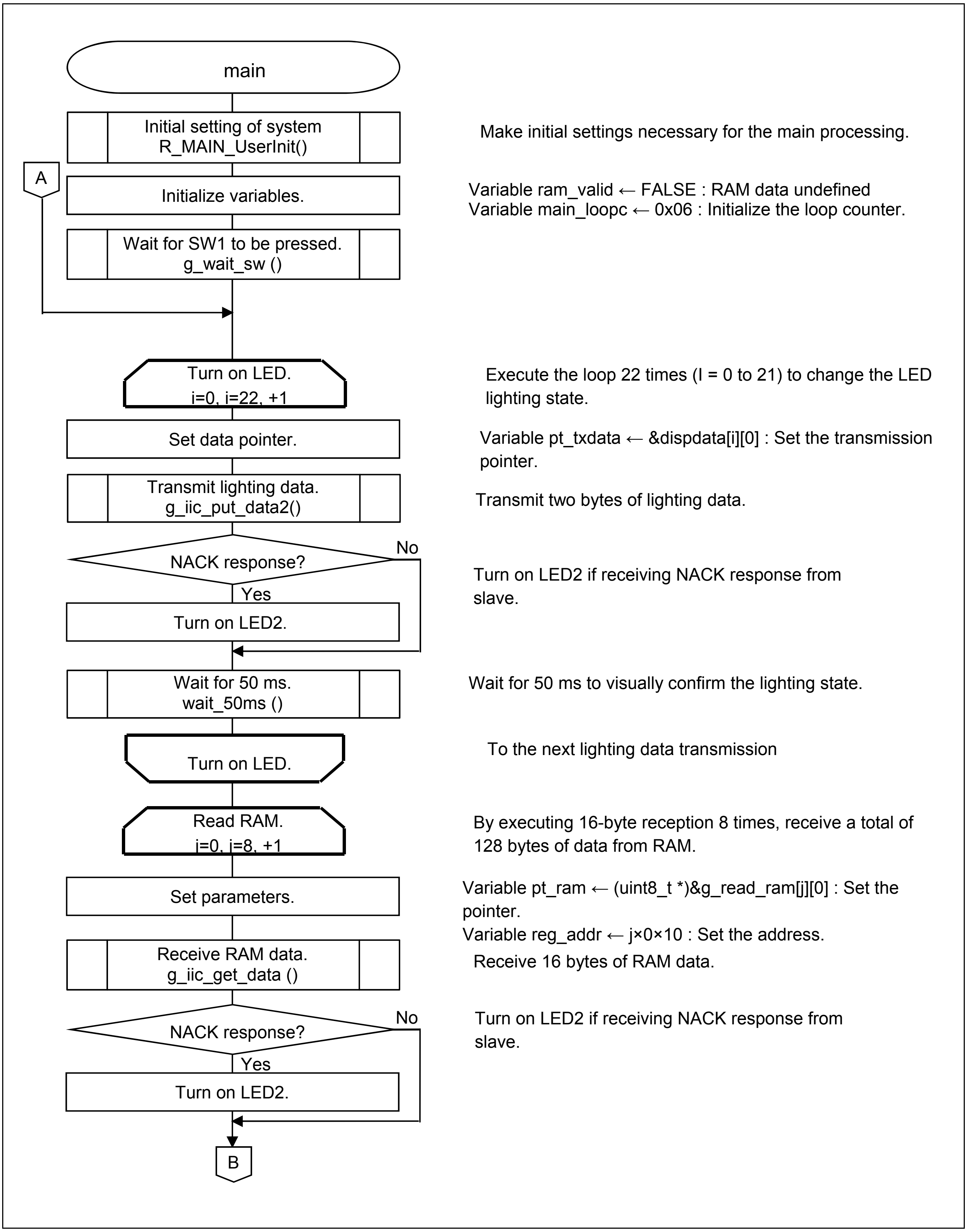
- 12-bit interval timer control register (ITMC)
  Set the interval time.

Symbol: ITMC

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RINTE	0	0	0	ITCMP11 to ITCMP0											
0	0	0	0	**0**	**0**	**1**	**0**	**1**	**1**	**1**	**0**	**1**	**1**	**0**	**1**

5.6.9 Main Process

Figures 5.10 to 5.12 show the flowcharts of the main process.



Make initial settings necessary for the main processing.

Variable ram_valid ← FALSE : RAM data undefined
Variable main_loopc ← 0x06 : Initialize the loop counter.

Execute the loop 22 times (I = 0 to 21) to change the LED lighting state.

Variable pt_txdata ← &dispdata[i][0] : Set the transmission pointer.

Transmit two bytes of lighting data.

Turn on LED2 if receiving NACK response from slave.

Wait for 50 ms to visually confirm the lighting state.

To the next lighting data transmission

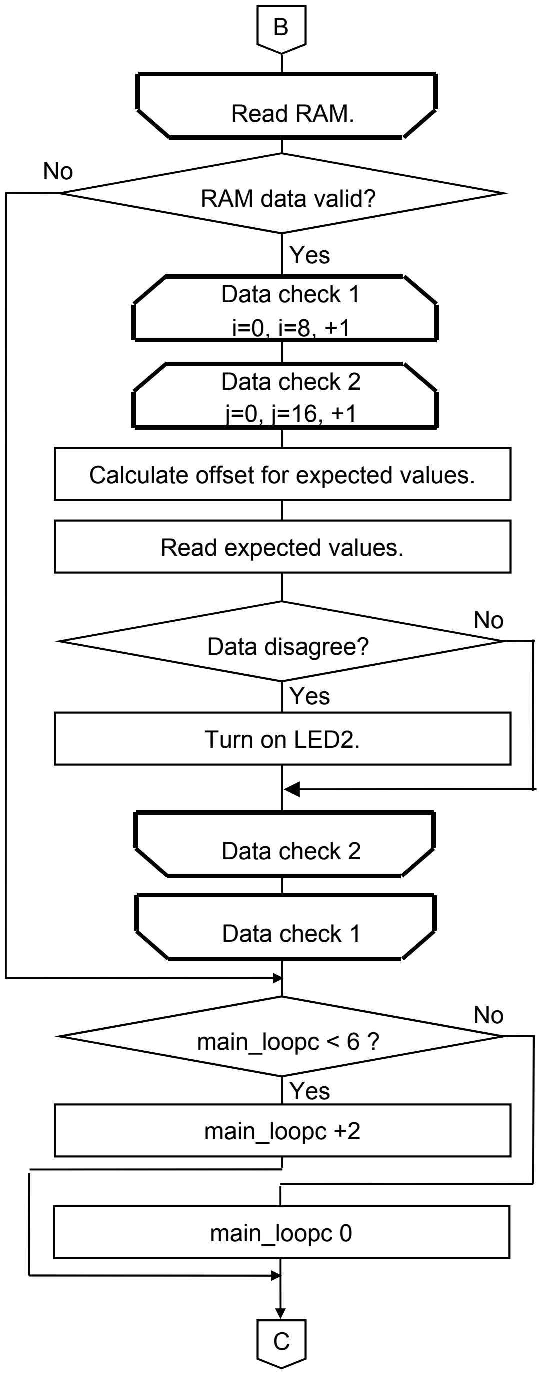
By executing 16-byte reception 8 times, receive a total of 128 bytes of data from RAM.

Variable pt_ram ← (uint8_t *)&g_read_ram[j][0] : Set the pointer.
Variable reg_addr ← j×0×10 : Set the address.

Receive 16 bytes of RAM data.

Turn on LED2 if receiving NACK response from slave.

Figure 5.10 Main Process (1/3)



If variable ram_valid is TRUTH (1), check data.

Compare expected values to received data in 16-character units 8 times to check 128-byte data.

Compare received data for 16 characters (i=0 to 15).

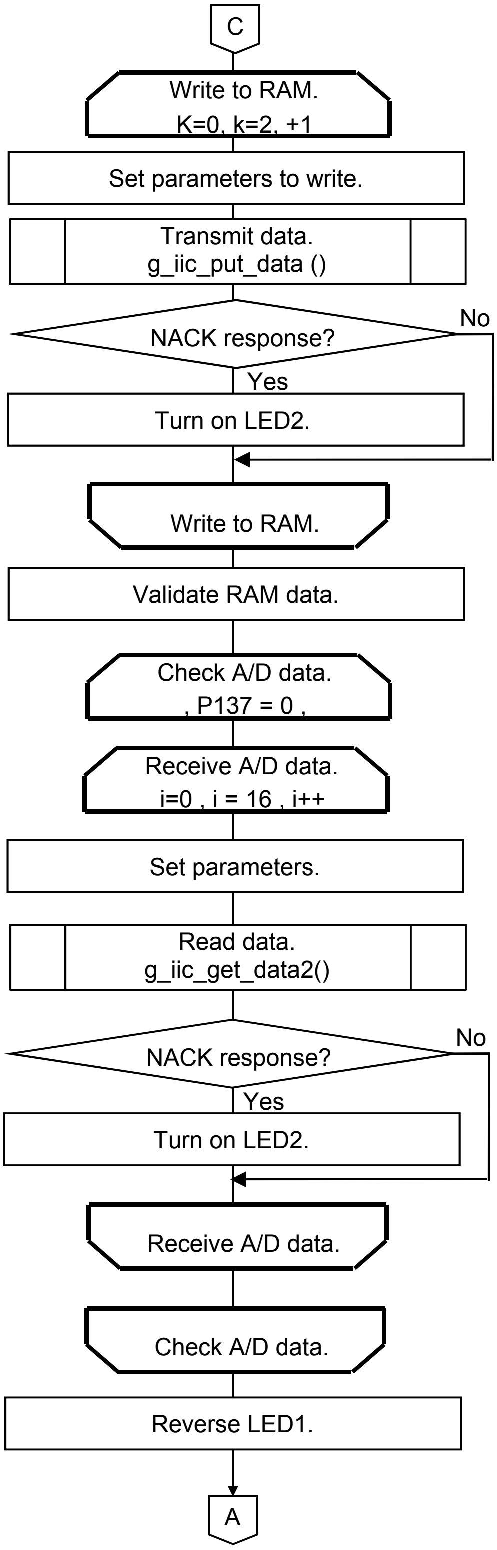
Variable temp_data ← ((i & 0b00000011) X16) + j

Variable temp_data ←
ram_data[(main_loopc+i/4)][temp_data]

Compare read-out data to expected values; if they disagree, turn on LED2.

Update pointer to RAM write data.

Figure 5.11 Main Process (2/3)



Divide 128-byte data into 64-byte data to transmit in 2 times (k=0 to 1).

Variable pt_txdata ← &ram_data[main_loopc+k][0]: pointer
Variable reg_addr ← RAM_TOP + k X 0x40: RAM address

Transmit 64-byte data.

Turn on LED2 if receiving NACK response from slave.

Variable ram_valid ← TRUTH

Read A/D conversion data until SW is pressed.

Read A/D conversion data 16 times (i=0 to 15).

Variable pt_ram ← (uint8_t *)&g_read_data[i][0]: buffer

Variable eg_addr ← 0x00 : Specify A/D conversion results.

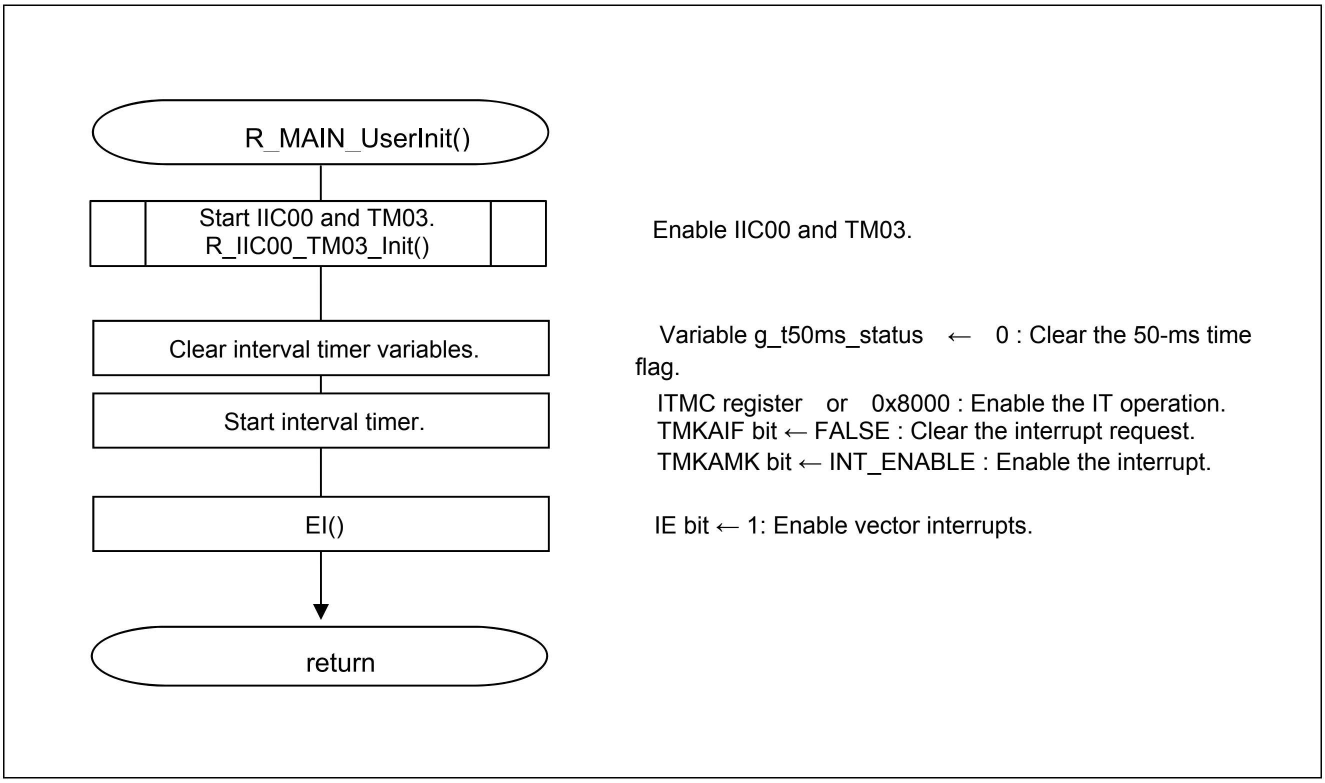
Read 4 channels of data (8 bytes).

Turn on LED2 if receiving NACK response from slave.

Figure 5.12 Main Process (3/3)

5.6.10 R_MAIN_UserInit Process

Figure 5.13 shows the flowchart of the R_MAIN_UserInit process.



Enable IIC00 and TM03.

Variable g_t50ms_status ← 0 : Clear the 50-ms time flag.

ITMC register or 0x8000 : Enable the IT operation.
TMKAIF bit ← FALSE : Clear the interrupt request.
TMKAMK bit ← INT_ENABLE : Enable the interrupt.

IE bit ← 1: Enable vector interrupts.

Figure 5.13 R_MAIN_UserInit Process

Starting interval timer

- 12-bit interval timer control register (ITMC)
  Start the interval timer.
- TMKAIF bit in the interrupt request flag register (IF1H)
  Clear the interval timer interrupt request.
- TMKAMK bit in the interrupt mask flag register (MK1H)
  Cancel mask of the interval timer interrupt.

Symbol: ITMC

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RINTE	0	0	0	ITCMP11 to ITCMP0											
1	0	0	0	**0**	**0**	**1**	**0**	**1**	**1**	**1**	**0**	**1**	**1**	**0**	**1**

Symbol: IF1H

Bit 2

TMKAIF	Interval timer interrupt request flag
0	**Interrupt request signal has not been generated.**
1	割り込み要求信号が発生し，割り込み要求状態

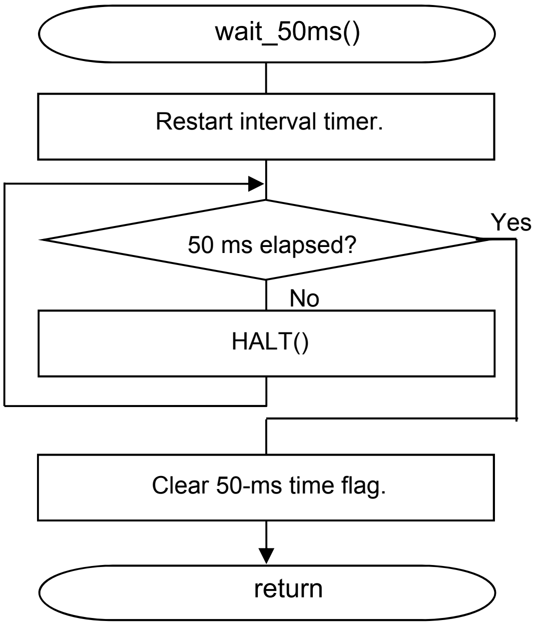
Symbol: MK1H

Bit 2

TMKAMK	Control of interval timer interrupts
0	割り込み処理許可
1	**Disables interrupt processing.**

5.6.11 Waiting for 50 ms

Figure 5.14 shows the flowchart for waiting for 50 ms.



ITMC register and 0x0FFF : Disable the IT operation.
Variable g_t50ms_status ← 0 : Clear the 50-ms time flag.
TMKAIF bit ← 0 : Clear the IT interrupt request flag.
ITMC register or 0x8000 : Enable the IT operation.
Wait for 50 ms in HALT mode.

Wait for 50-ms interval timer interrupt in HALT mode.

Variable g_t50ms_status ← 0 : Clear the 50-ms time flag.

Figure 5.14 Waiting for 50 ms

Restarting interval timer

- 12-bit interval timer control register (ITMC)
  Stop and start the interval timer.
- TMKAIF bit in the interrupt request flag register (IF1H)
  Clear the interval timer interrupt request.

Symbol: ITMC

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RINTE	0	0	0	ITCMP11 to ITCMP0											
0/1	0	0	0	**0**	**0**	**1**	**0**	**1**	**1**	**1**	**0**	**1**	**1**	**0**	**1**

Symbol: IF1H

Bit 2

TMKAIF	Interval timer interrupt request flag
0	**Interrupt request signal has not been generated.**
1	割り込み要求信号が発生し，割り込み要求状態

5.6.12 Waiting for SW1 to be Pressed

Figure 5.15 shows the flowchart for waiting for the SW1 to be pressed.

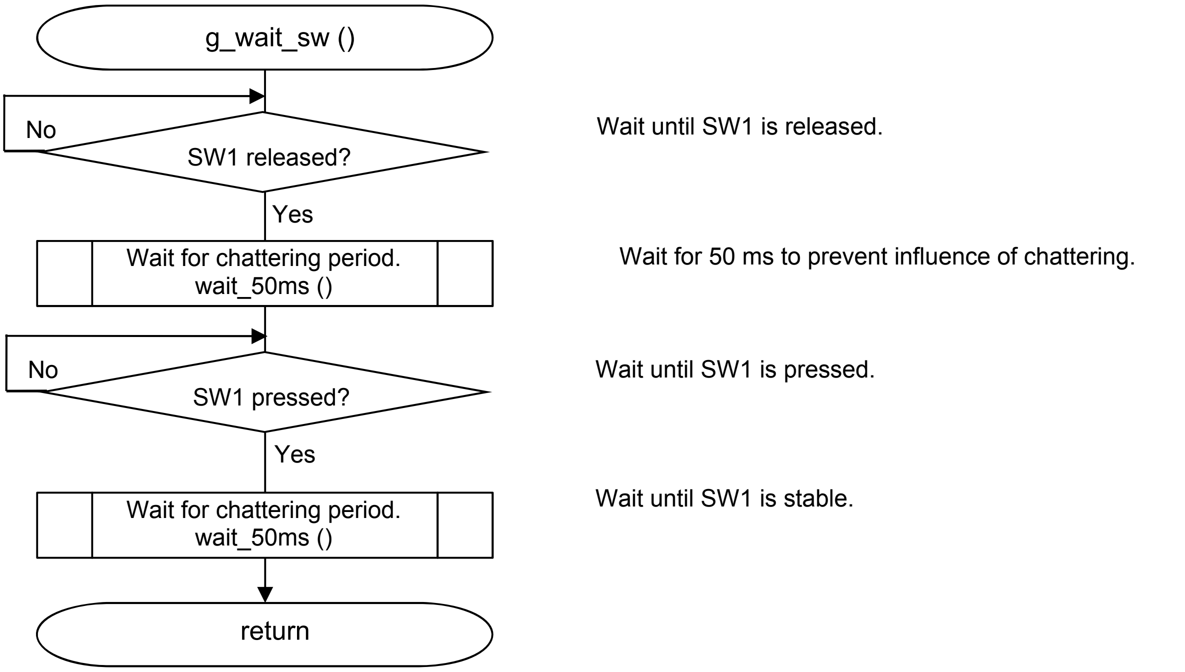

g_wait_sw ()
No
SW1 released?
Yes
Wait until SW1 is released.
Wait for chattering period.
wait_50ms ()
Wait for 50 ms to prevent influence of chattering.
No
SW1 pressed?
Yes
Wait until SW1 is pressed.
Wait for chattering period.
wait_50ms ()
Wait until SW1 is stable.
return

Figure 5.15 Waiting for SW1 to be Pressed

5.6.13 Transmitting Data to RAM

Figure 5.16 shows the flowchart for transmitting data to the RAM.

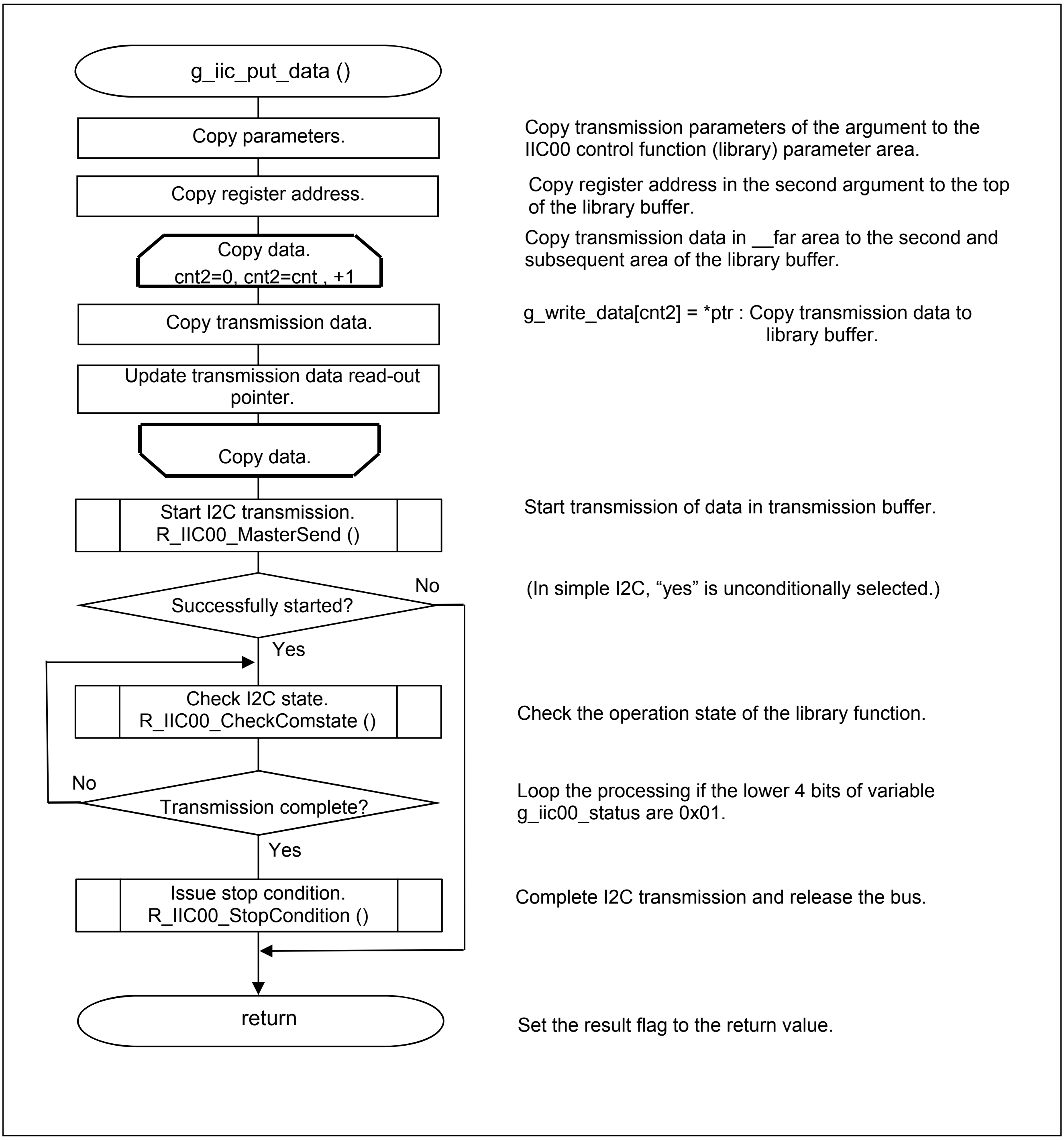

g_iic_put_data ()
Copy parameters.
Copy transmission parameters of the argument to the IIC00 control function (library) parameter area.
Copy register address.
Copy register address in the second argument to the top of the library buffer.
Copy data.
cnt2=0, cnt2=cnt , +1
Copy transmission data in __far area to the second and subsequent area of the library buffer.
Copy transmission data.
g_write_data[cnt2] = *ptr : Copy transmission data to library buffer.
Update transmission data read-out pointer.
Copy data.
Start I2C transmission.
R_IIC00_MasterSend ()
Start transmission of data in transmission buffer.
Successfully started?
No
Yes
(In simple I2C, “yes” is unconditionally selected.)
Check I2C state.
R_IIC00_CheckComstate ()
Check the operation state of the library function.
Transmission complete?
No
Yes
Loop the processing if the lower 4 bits of variable g_iic00_status are 0x01.
Issue stop condition.
R_IIC00_StopCondition ()
Complete I2C transmission and release the bus.
return
Set the result flag to the return value.

Figure 5.16 Transmitting Data to RAM

5.6.14 Transmitting data to LED

Figure 5.17 shows the flowchart for transmitting data to the LED.

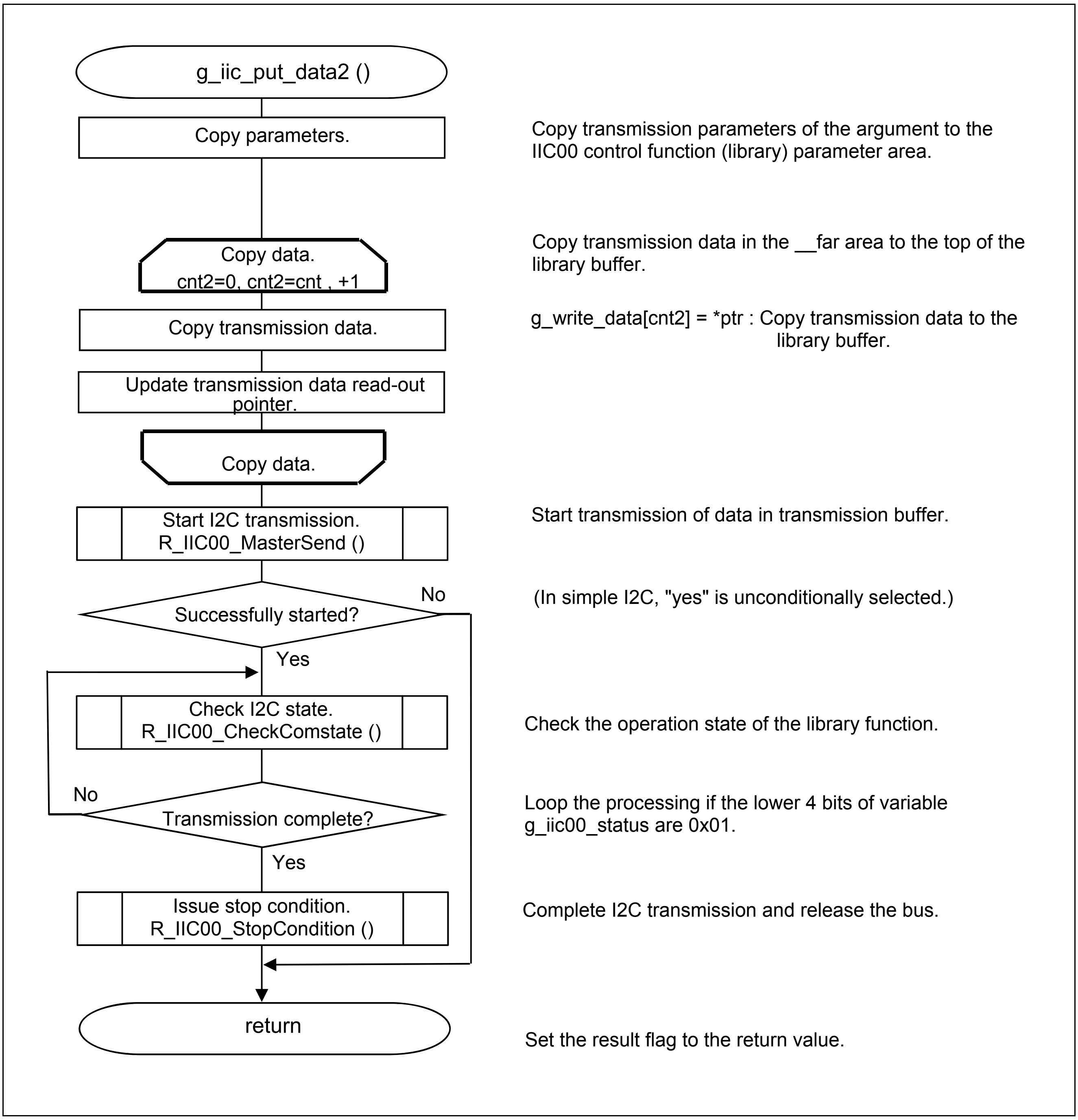

g_iic_put_data2 ()
Copy parameters.
Copy transmission parameters of the argument to the IIC00 control function (library) parameter area.
Copy data.
cnt2=0, cnt2=cnt , +1
Copy transmission data in the __far area to the top of the library buffer.
Copy transmission data.
g_write_data[cnt2] = *ptr : Copy transmission data to the library buffer.
Update transmission data read-out pointer.
Copy data.
Start I2C transmission.
R_IIC00_MasterSend ()
Start transmission of data in transmission buffer.
Successfully started?
No
Yes
(In simple I2C, "yes" is unconditionally selected.)
Check I2C state.
R_IIC00_CheckComstate ()
Check the operation state of the library function.
Transmission complete?
No
Yes
Loop the processing if the lower 4 bits of variable g_iic00_status are 0x01.
Issue stop condition.
R_IIC00_StopCondition ()
Complete I2C transmission and release the bus.
return
Set the result flag to the return value.

Figure 5.17 Transmitting data to LED

5.6.15 Receiving Data from RAM

Figure 5.18 shows the flowchart for receiving data from the RAM.

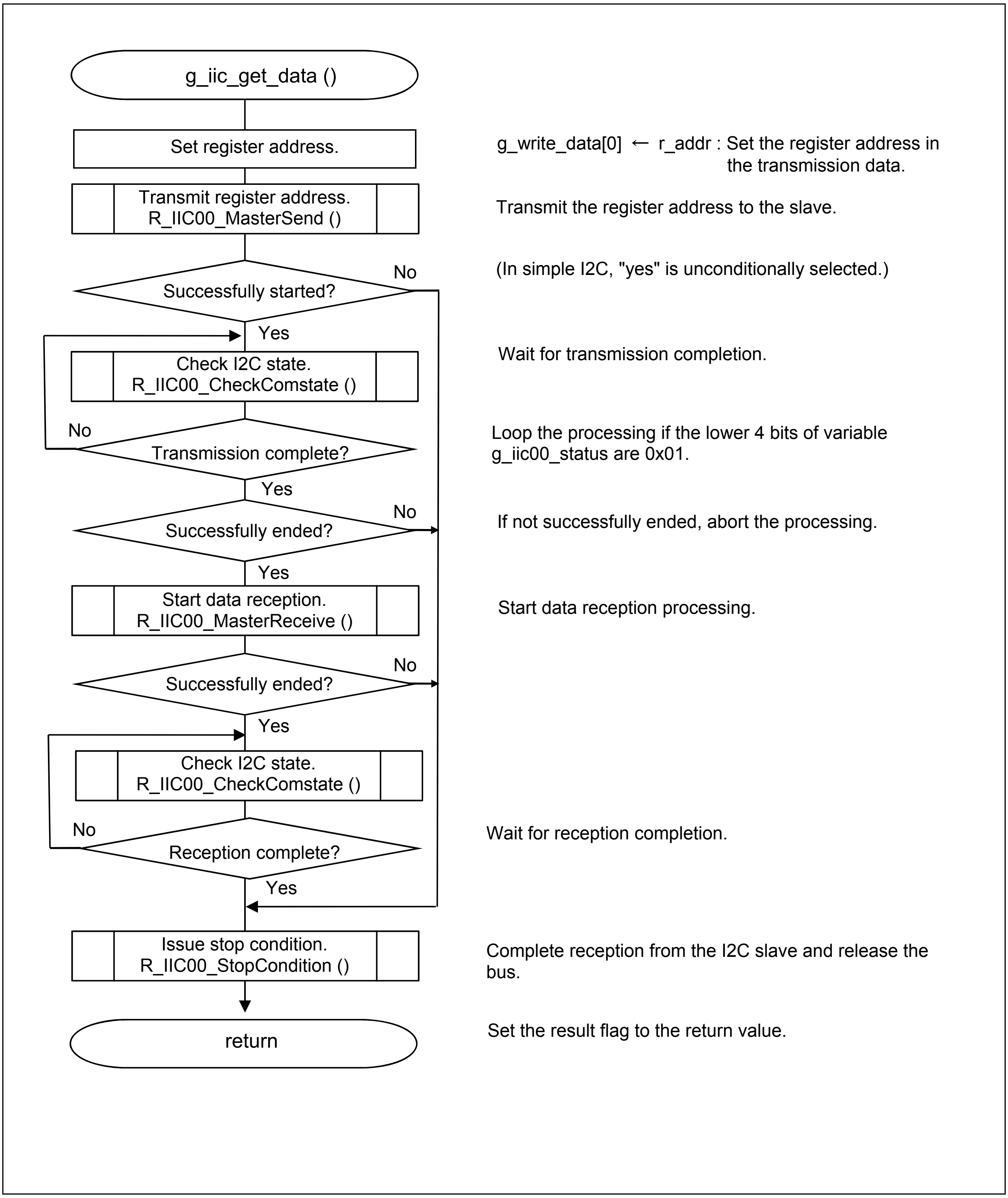

g_iic_get_data ()
Set register address.
Transmit register address.
R_IIC00_MasterSend ()
Successfully started?
No
Yes
Check I2C state.
R_IIC00_CheckComstate ()
No
Transmission complete?
Yes
Successfully ended?
No
Yes
Start data reception.
R_IIC00_MasterReceive ()
Successfully ended?
No
Yes
Check I2C state.
R_IIC00_CheckComstate ()
No
Reception complete?
Yes
Issue stop condition.
R_IIC00_StopCondition ()
return
g_write_data[0] ← r_addr : Set the register address in the transmission data.
Transmit the register address to the slave.
(In simple I2C, "yes" is unconditionally selected.)
Wait for transmission completion.
Loop the processing if the lower 4 bits of variable g_iic00_status are 0x01.
If not successfully ended, abort the processing.
Start data reception processing.
Wait for reception completion.
Complete reception from the I2C slave and release the bus.
Set the result flag to the return value.

Figure 5.18 Receiving Data from RAM

5.6.16 Receiving A/D Conversion Result Data

Figure 5.19 shows the flowchart for receiving A/D conversion result data.

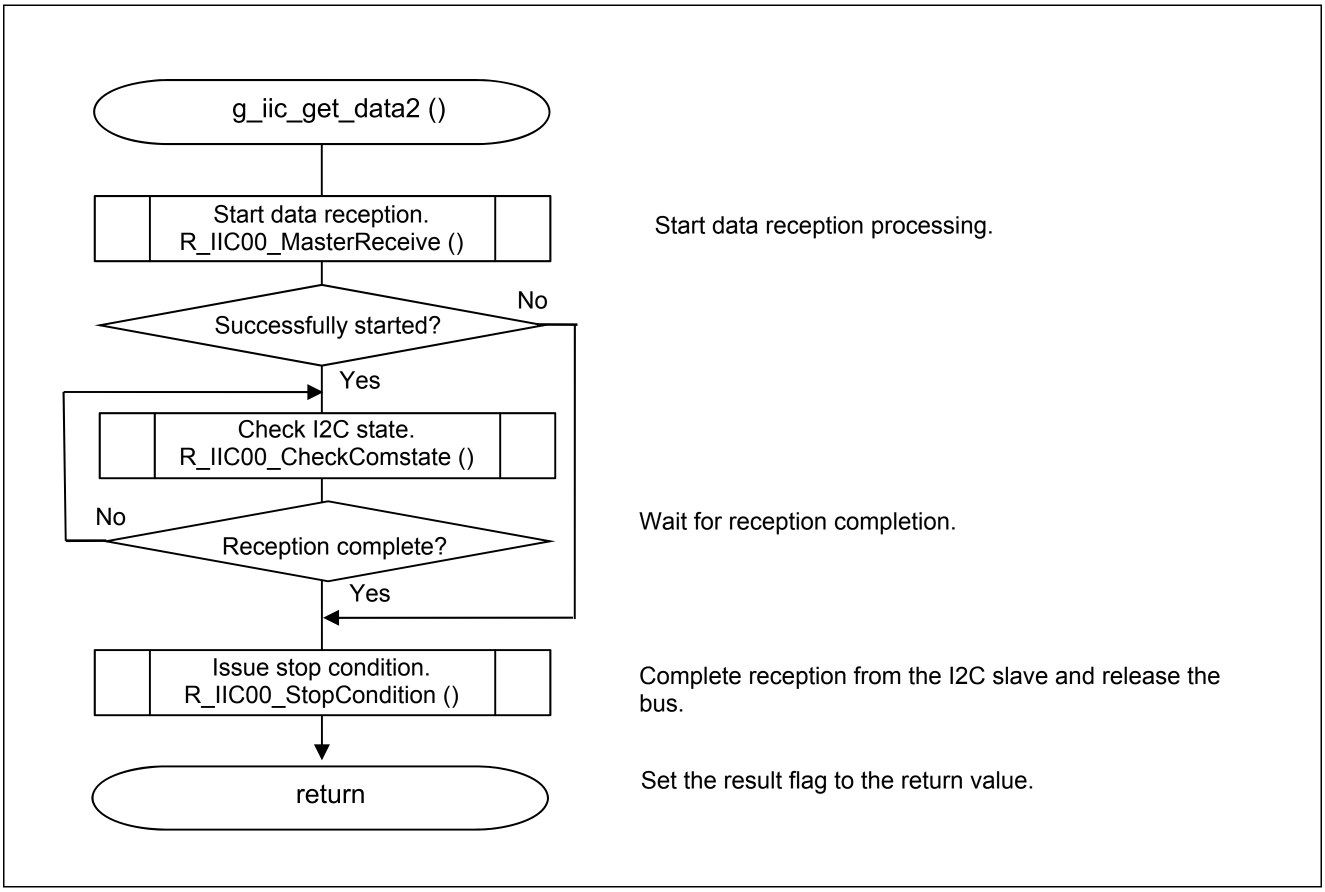


Figure 5.19 Receiving A/D Conversion Result Data

5.6.17 Interval Timer Interrupt Process

Figure 5.20 shows the flowchart of the interval timer interrupt process.

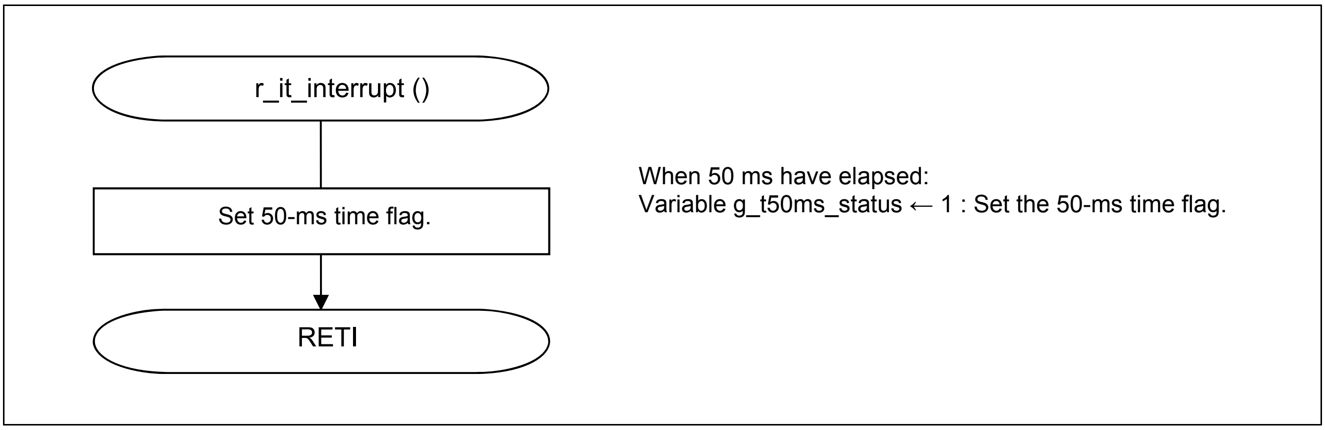


Figure 5.20 Interval Timer Interrupt Process

5.6.18 Preparing for Starting IIC00 and TM03

Figure 5.21 shows the flowchart for preparing for starting the IIC00 and TM03.

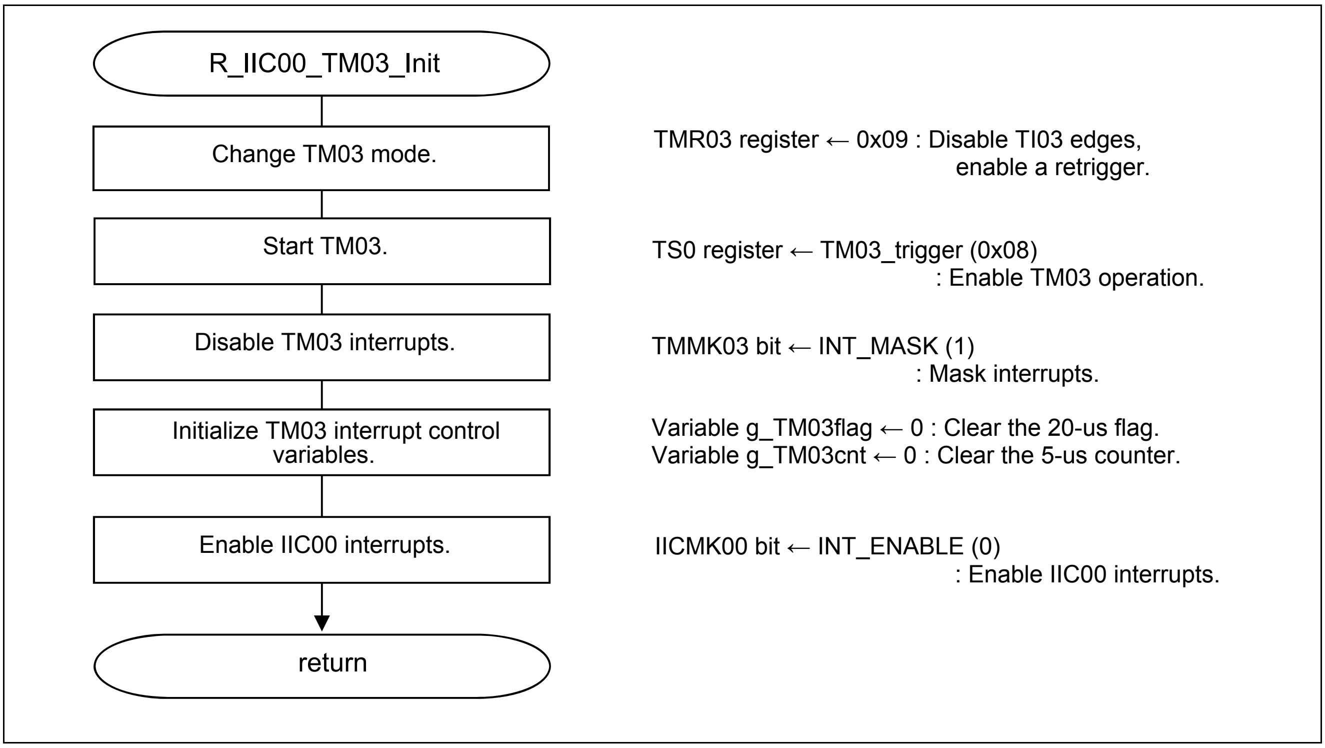

R_IIC00_TM03_Init
Change TM03 mode.
Start TM03.
Disable TM03 interrupts.
Initialize TM03 interrupt control variables.
Enable IIC00 interrupts.
return
TMR03 register ← 0x09 : Disable TI03 edges, enable a retrigger.
TS0 register ← TM03_trigger (0x08) : Enable TM03 operation.
TMMK03 bit ← INT_MASK (1) : Mask interrupts.
Variable g_TM03flag ← 0 : Clear the 20-us flag.
Variable g_TM03cnt ← 0 : Clear the 5-us counter.
IICMK00 bit ← INT_ENABLE (0) : Enable IIC00 interrupts.

Figure 5.21 Preparing for Starting IIC00 and TM03

Starting TM03

- Timer mode register 03 (TMR03)
  Change to disable edge detection in channel 03.
  Change to enable a retrigger function during counting.
- Timer channel start register 0 (TS0)
  Enable channel 03 operation.
- TMMK03 bit in the interrupt mask flag register (MK1L)
  Mask TM03 interrupts.

Symbol: TMR03

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS 031	CKS 030	0	CCS 03	SPLIT 03	STS 032	STS 031	STS 030	CIS 031	CIS 030	0	0	MD 033	MD 032	MD 031	MD 030
0	**0**	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	0	0	**1**	**0**	**0**	**1**

Bits 15 and 14

CKS031	CKS030	Selection of operation clock for channel 3 (f_{MCK})
0	**0**	**Operation clock CK00 set by the timer clock selection register 0 (TPS0)**

Bit 12

CCS03	Selection of count clock for channel 3 (f_{TCLK})
0	**Operation clock specified by CKS031 and CKS030 bits (f_{MCK})**

Bit 11

SPLIT03	Selection between 8-bit and 16-bit timer operation for channel 3
0	**16-bit timer**

Symbol: TMR03

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS 031	CKS 030	0	CCS 03	SPLIT 03	STS 032	STS 031	STS 030	CIS 031	CIS 030	0	0	MD 033	MD 032	MD 031	MD 030
0	**0**	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	0	0	**1**	**0**	**0**	**1**

Bits 10 to 8

STS032	STS031	STS030	Setting of start trigger and capture trigger for channel 3
0	**0**	**0**	**Only enables software trigger start (deselects the other trigger sources).**

Bits 7 and 6

CIS031	CIS030	Selection of valid edge of TI03 pin
0	**0**	**Falling edge**

Bits 3 to 0

MD 033	MD 032	MD 031	MD 030	Operating mode of channel 3	Corresponding functions	TCR counting operation
1	**0**	**0**	**1**	**One-count mode**	Delay counter/one-shot pulse output/PWM output (slave)	Decrementing

Symbol: TS0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	TS03H	0	TS01H	0	0	0	0	0	TS03	TS02	TS01	TS00
0	0	0	0	0	0	0	0	0	0	0	0	**1**	0	0	0

Bit 3

TS03	Channel 03 operation enable trigger
0	トリガ動作しない
1	**Sets TE03 bit to 1 and enables counting.**

Symbol: MK1L

Bit 5

TMMK03	Control of interrupt processing
0	割り込み処理許可
1	**Disables interrupt processing.**

Symbol: MK0H

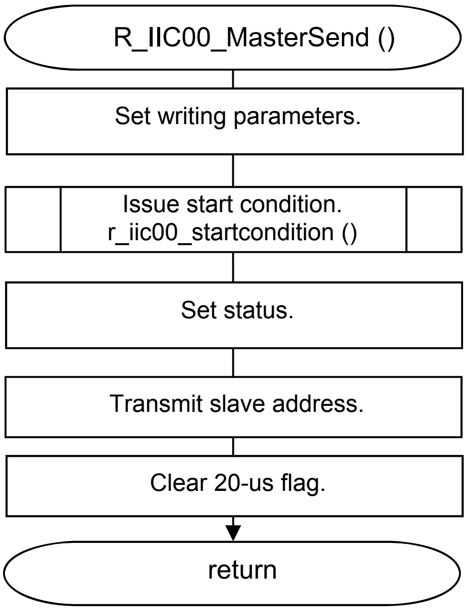
Bit 1

IICMK00	Control of interrupt processing
0	**Enables interrupt processing.**
1	割り込み処理禁止

Note: For details of register settings, refer to the RL78/I1D User’s Manual: Hardware.

5.6.19 Starting Data Transmission

Figure 5.22 shows the flowchart for starting data transmission.



Variable g_iic00_tx_cnt ←Number of transmission data
Variable gp_iic00_tx_address ← Pointer to transmission buffer

Issue the start condition and enable transmission.

Variable g_iic00_status ← IIC_USING (0x01)
: Set IIC00 during transmission.

SIO00 register ← Slave address & 0xFE
: Transmit the slave address.

Variable g_TM03flag ← FALSE(0x00)
: IIC00 is not in the 20-us wait state.

Return value ← MD_OK : No error

Figure 5.22 Starting Data Transmission

Transmitting slave address

- Serial data register 00 (SIO00)
  Set transmission data (slave address).

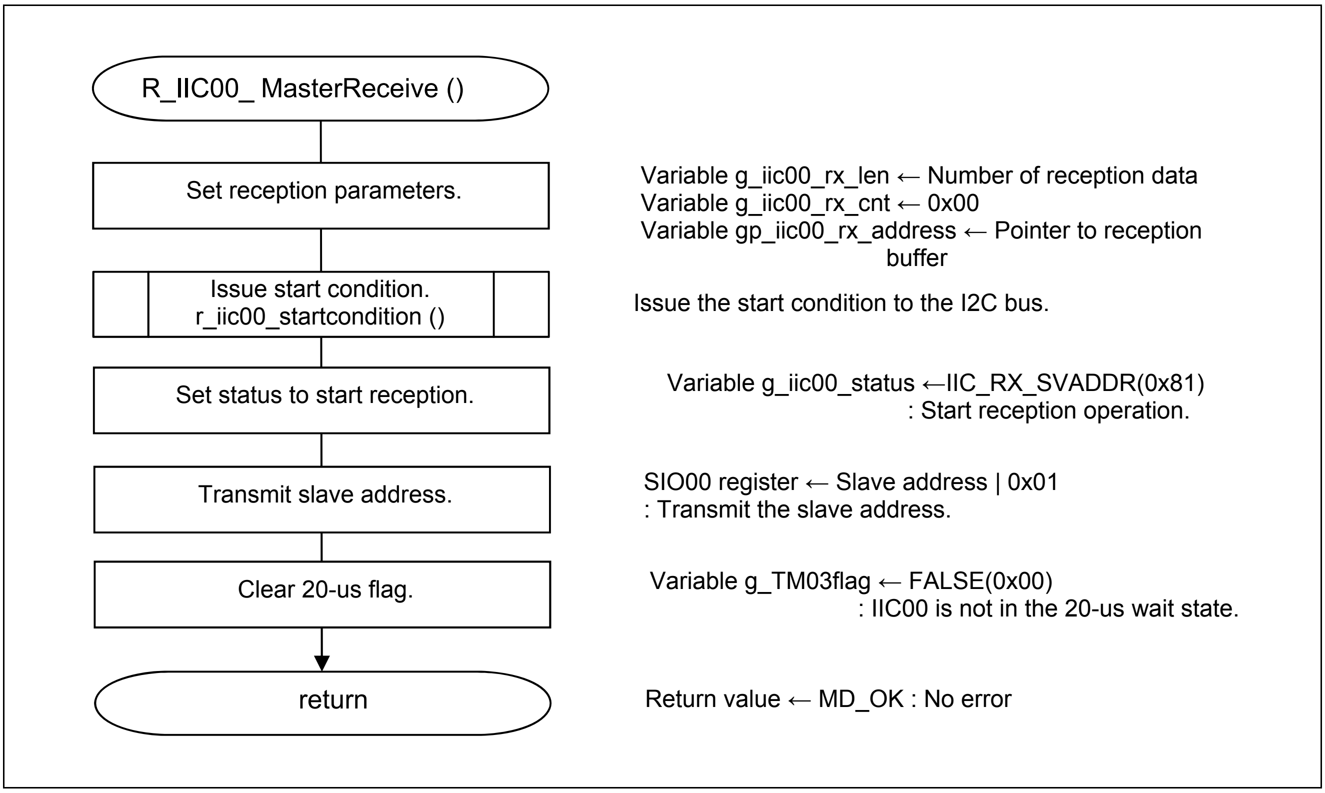
Symbol: SIO00

7	6	5	4	3	2	1	0
0	1	1	0/1	0	0	0	0

Slave address is 0x60 (LED) or 0x70 (RAM).

5.6.20 Starting Data Reception

Figure 5.23 shows the flowchart of data reception start processing.


R_IIC00_ MasterReceive ()
Set reception parameters.
Issue start condition.
r_iic00_startcondition ()
Set status to start reception.
Transmit slave address.
Clear 20-us flag.
return

Variable g_iic00_rx_len ← Number of reception data
Variable g_iic00_rx_cnt ← 0x00
Variable gp_iic00_rx_address ← Pointer to reception buffer

Issue the start condition to the I2C bus.

Variable g_iic00_status ←IIC_RX_SVADDR(0x81)
: Start reception operation.

SIO00 register ← Slave address | 0x01
: Transmit the slave address.

Variable g_TM03flag ← FALSE(0x00)
: IIC00 is not in the 20-us wait state.

Return value ← MD_OK : No error

Figure 5.23 Starting Data Reception

Transmitting addresses

- Lower bits in the serial data register 00 (SDR00) (SIO00 register)
  Set transmission data (slave address).

Symbol: SIO00

7	6	5	4	3	2	1	0
0	1	1	0/1	0	0	0	1

Slave address is 0x61 (A/D) or 0x71 (RAM).

5.6.21 Waiting for IIC00 Transmission Completion

Figure 5.24 shows the flowchart for waiting for IIC00 transmission completion.

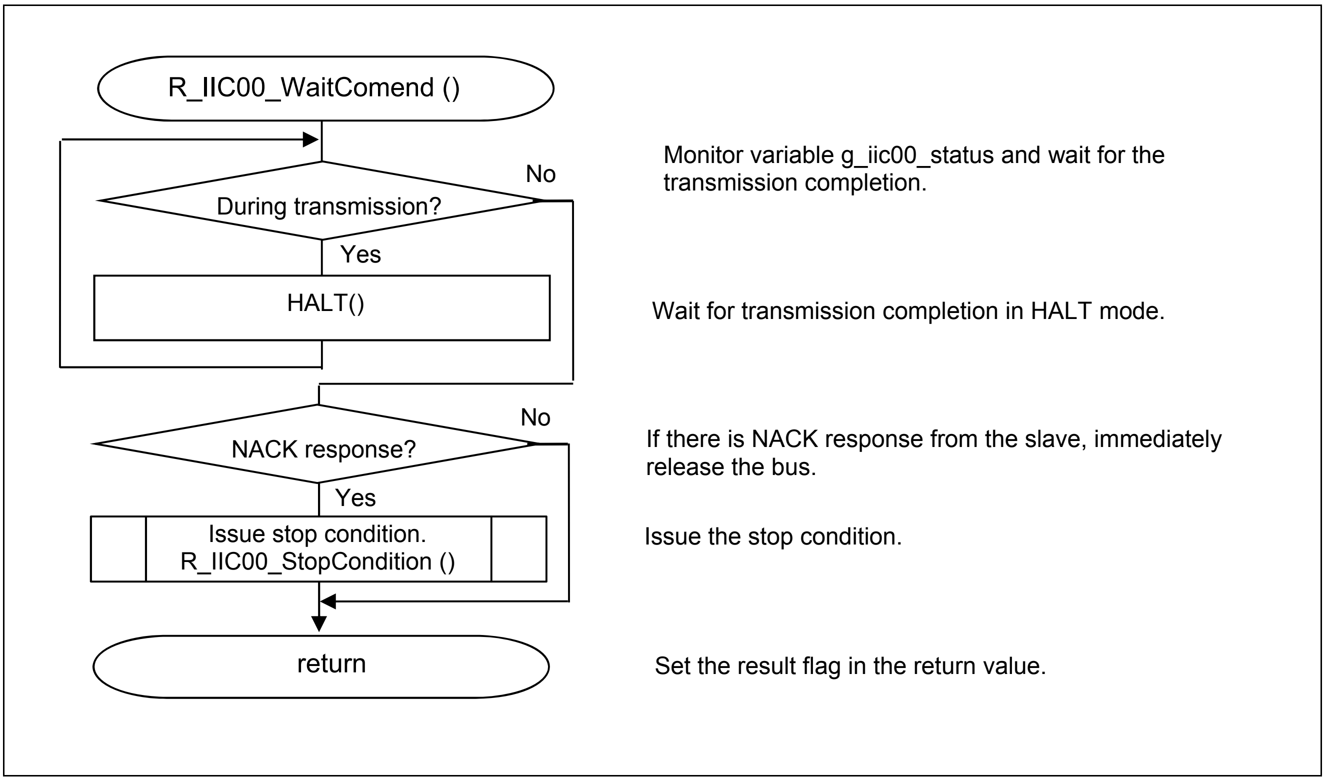

R_IIC00_WaitComend ()
During transmission?
No
Yes
HALT()
NACK response?
No
Yes
Issue stop condition.
R_IIC00_StopCondition ()
return
Monitor variable g_iic00_status and wait for the transmission completion.
Wait for transmission completion in HALT mode.
If there is NACK response from the slave, immediately release the bus.
Issue the stop condition.
Set the result flag in the return value.

Figure 5.24 Waiting for IIC00 Transmission Completion

5.6.22 Checking IIC00 State

Figure 5.25 shows the flowchart for checking the IIC00 state.

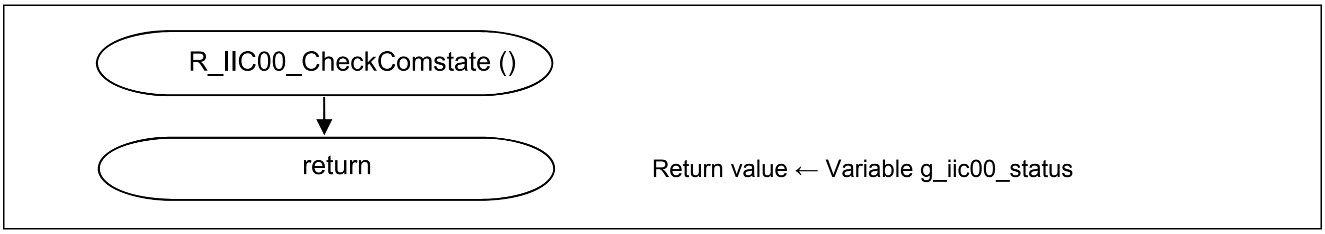


Figure 5.25 Checking IIC00 State

5.6.23 Issuing Stop Condition

Figure 5.26 shows the flowchart for issuing the stop condition.

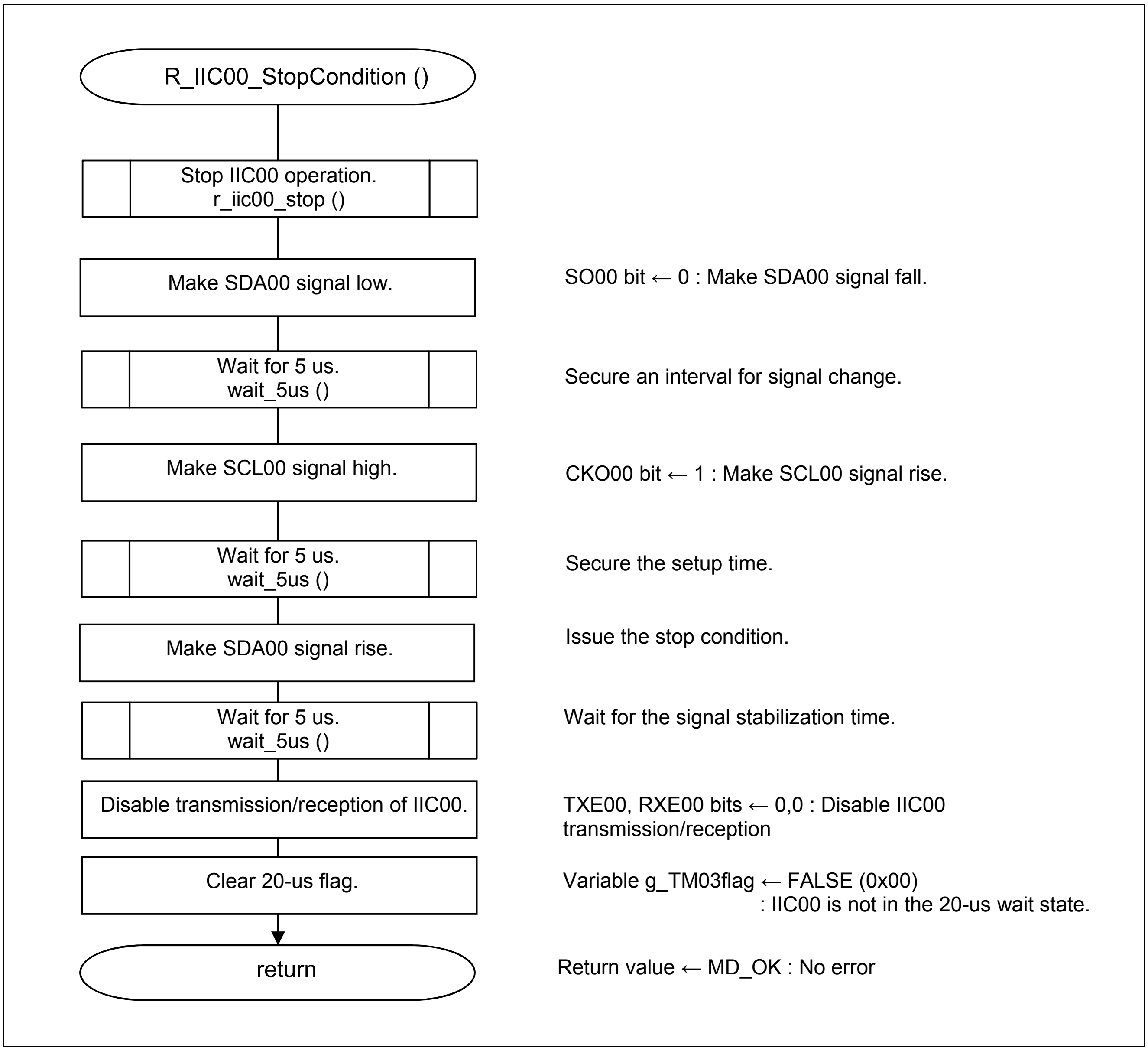

R_IIC00_StopCondition ()
Stop IIC00 operation.
r_iic00_stop ()
Make SDA00 signal low.
SO00 bit ← 0 : Make SDA00 signal fall.
Wait for 5 us.
wait_5us ()
Secure an interval for signal change.
Make SCL00 signal high.
CKO00 bit ← 1 : Make SCL00 signal rise.
Wait for 5 us.
wait_5us ()
Secure the setup time.
Make SDA00 signal rise.
Issue the stop condition.
Wait for 5 us.
wait_5us ()
Wait for the signal stabilization time.
Disable transmission/reception of IIC00.
TXE00, RXE00 bits ← 0,0 : Disable IIC00 transmission/reception
Clear 20-us flag.
Variable g_TM03flag ← FALSE (0x00)
: IIC00 is not in the 20-us wait state.
return
Return value ← MD_OK : No error

Figure 5.26 Issuing Stop Condition

Controlling SDA00 and SCL00 signals

- CKO00 and SO00 bits in the serial output register 0 (SO0)

  Make SDA00 signal fall.

  Make SCL00 signal rise.

  Make SDA00 signal rise.

Symbol: SO0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	CKO01	CKO00	0	0	0	0	0	0	SO01	SO00
0	0	0	0	0	0	x	**0**	0	0	0	0	0	0	0	**0/1**

Bit 0

SO00	Channel 0 serial data output
0	**Serial data output value is 0.**
1	**Serial data output value is 1.**

Bit 8

CKO00	Channel 0 serial clock output
0	シリアル・クロック出力値が “0”
1	**Serial clock output value is 1.**

Disabling IIC00 transmission/reception

- TXE00 and RXE00 bits in the serial communication operation setting register 00 (SCR00)
  Disable transmission.
  Disable reception

Symbol: SCR00

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXE 00	RXE 00	DAP 00	CKP 00	0	EOC 00	PTC 001	PTC 000	DIR 00	0	SLC 001	SLC 000	0	1	DLS 001	DLS 000
0	**0**	0	0	0	0	0	0	0	0	0	1	0	1	1	1

Bits 15 and 14

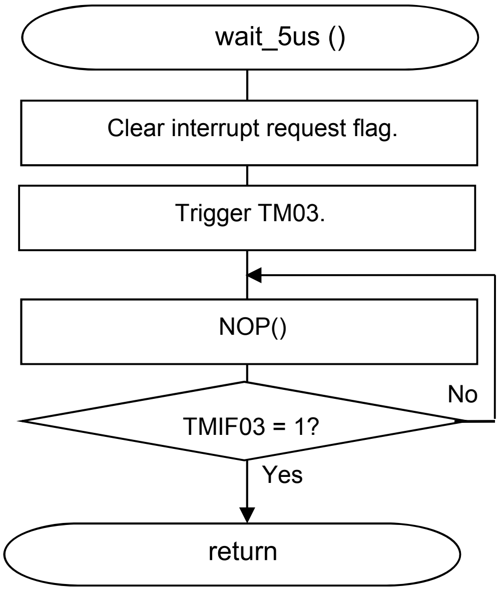
TXE00	RXE00	Channel 0 operating mode setting
0	**0**	**Disables communication.**
0	1	受信のみを行う
1	0	送信のみを行う

11 送受信を行う（簡易 I2C では設定禁止）

Note: For details of register settings, refer to the RL78/I1D User’s Manual: Hardware.

5.6.24 Waiting for 5 us

Figure 5.27 shows the flowchart for waiting for 5 us.



TMIF03 bit ← FALSE (0)
: Clear the TM03 interrupt request.

TS0 register ← TM03_trigger (0x08)
: Start TM03 counting operation.

Wait for generation of INTTM03 (5 us) with the NOP instruction.

Figure 5.27 Waiting for 5 us

Triggering TM03 start

- TMIF03 bit in the interrupt request flag register (IF1L)
  Clear the TM03 interrupt request.
- Timer channel start register 0 (TS0)
  Start channel 03 counting operation.

Symbol: IF1L

Bit 5

TMIF03	Interrupt request flag
0	**Interrupt request signal has not been generated.**
1	割り込み要求信号が発生し、割り込み要求状態

Symbol: TS0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	TS03H	0	TS01H	0	0	0	0	0	TS03	TS02	TS01	TS00
0	0	0	0	0	0	0	0	0	0	0	0	**1**	0	0	0

Bit 3

TS03	Channel 03 operation enable trigger
0	トリガ動作しない
1	**Triggers start of counting operation.**

5.6.25 Issuing Start Condition

Figure 5.28 shows the flowchart for issuing the start condition.

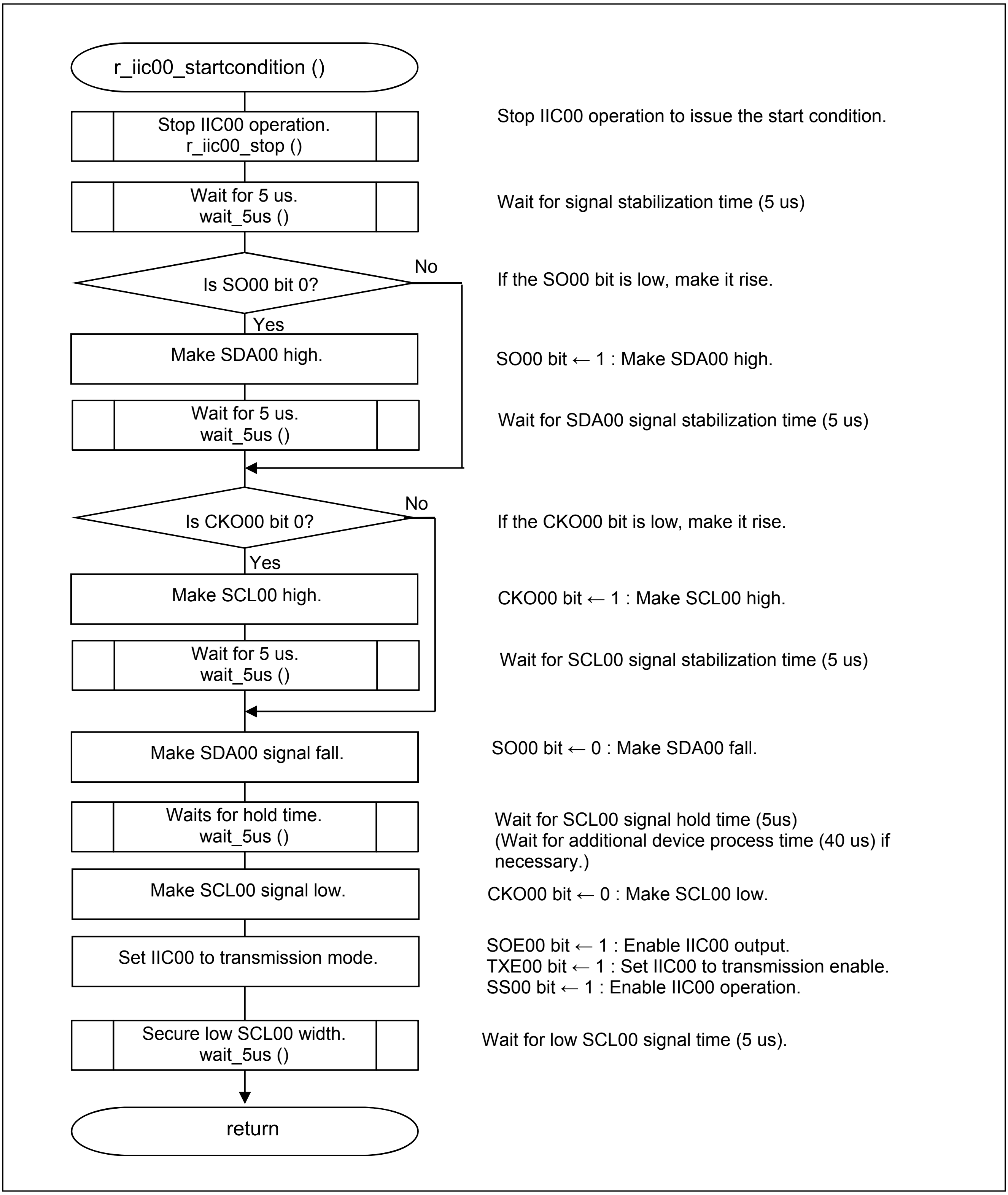

r_iic00_startcondition ()
Stop IIC00 operation.
r_iic00_stop ()
Stop IIC00 operation to issue the start condition.
Wait for 5 us.
wait_5us ()
Wait for signal stabilization time (5 us)
Is SO00 bit 0?
No
Yes
If the SO00 bit is low, make it rise.
Make SDA00 high.
SO00 bit ← 1 : Make SDA00 high.
Wait for 5 us.
wait_5us ()
Wait for SDA00 signal stabilization time (5 us)
Is CKO00 bit 0?
No
Yes
If the CKO00 bit is low, make it rise.
Make SCL00 high.
CKO00 bit ← 1 : Make SCL00 high.
Wait for 5 us.
wait_5us ()
Wait for SCL00 signal stabilization time (5 us)
Make SDA00 signal fall.
SO00 bit ← 0 : Make SDA00 fall.
Waits for hold time.
wait_5us ()
Wait for SCL00 signal hold time (5us)
(Wait for additional device process time (40 us) if necessary.)
Make SCL00 signal low.
CKO00 bit ← 0 : Make SCL00 low.
Set IIC00 to transmission mode.
SOE00 bit ← 1 : Enable IIC00 output.
TXE00 bit ← 1 : Set IIC00 to transmission enable.
SS00 bit ← 1 : Enable IIC00 operation.
Secure low SCL00 width.
wait_5us ()
Wait for low SCL00 signal time (5 us).
return

Figure 5.28 Issuing Start Condition

Checking SO00 state and making it rise

- SO00 bit in the serial output register (SO0)
  Check SDA00 and make it rise.

Symbol: SO0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	CKO01	CKO00	0	0	0	0	0	0	SO01	SO00
0	0	0	0	0	0	x	x	0	0	0	0	0	0	0	**1**

Bit 0

SO00	Channel 0 serial data output
0	シリアル・データ出力値が“0”
1	**Serial data output value is 1.**

Checking CKO00 state and making it rise

- CKO00 bit in the serial output register (SO0)
  Check SCL00 and makes it rise.

Symbol: SO0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	CKO01	CKO00	0	0	0	0	0	0	SO01	SO00
0	0	0	0	0	0	x	**1**	0	0	0	0	0	0	0	1

Bit 8

CKO00	Channel 0 serial clock output
0	シリアル・クロック出力値が“0”
1	**Serial clock output value is 1.**

Issuing start condition

- SO00 bit in the serial output register (SO0)
  Make SDA00 fall.

Symbol: SO0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	CKO01	CKO00	0	0	0	0	0	0	SO01	SO00
0	0	0	0	0	0	x	1	0	0	0	0	0	0	0	**0**

Bit 0

SO00	Channel 0 serial data output
0	**Serial data output value is 0.**
1	シリアル・データ出力値が“1”

Preparing for transmission

- CKO00 bit in the serial output register (SO0)
  Make SCL00 fall.

Symbol: SO0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	CKO01	CKO00	0	0	0	0	0	0	SO01	SO00
0	0	0	0	0	0	x	**0**	0	0	0	0	0	0	0	0

Bit 8

CKO00	Channel 0 serial clock output
0	**Serial clock output value is 0.**
1	シリアル・クロック出力値が“1”

Setting IIC00 transmission mode

- SOE00 bit in the serial output enable register 0L (SOE0L)
  Enable IIC00 output.
- TXE00 bit in the serial communication operation setting register 00 (SCR00)
  Enable IIC00 transmission.
- SS00 bit in the serial channel start register 0 (SS0)
  Enable IIC00 operation.

Symbol: SOE0L

7	6	5	4	3	2	1	0
0	0	0	0	0	0	SOE01	SOE00
0	0	0	0	0	0	x	1

Bit 0

SOE00	Enable/disable of channel 0 serial output
0	シリアル通信動作による出力停止
1	**Enables serial communication operation output.**

Symbol: SCR00

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXE 00	RXE 00	DAP 00	CKP 00	0	EOC 00	PTC 001	PTC 000	DIR 00	0	SLC 001	SLC 000	0	1	DLS 001	DLS 000
1	**0**	0	0	0	0	0	0	0	0	0	1	0	1	1	1

Bits 15 and 14

TXE00	RXE00	Setting of channel 0 operating mode
0	0	通信禁止
0	1	受信のみを行う
1	**0**	**Transmission only**
1	1	送受信を行う（簡易 I2C では設定禁止）

Symbol: SS0L

7	6	5	4	3	2	1	0
0	0	0	0	0	0	SS01	SS00
0	0	0	0	0	0	x	1

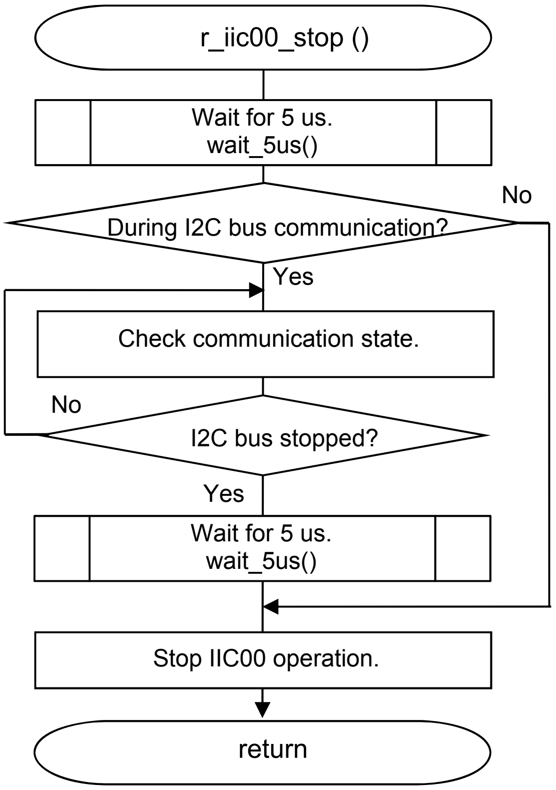
Bit 0

SS00	Channel 0 operation start trigger
0	トリガ動作せず
1	**Sets the SE00 bit to 1 and makes transition to the communication wait state.**

Note: For details of register settings, refer to the RL78/I1D User’s Manual: Hardware.

5.6.26 Stopping IIC00 Operation

Figure 5.29 shows the flowchart for stopping the IIC00 operation.



If I2C bus communication is in progress, wait for the communication completion.

Variable “work” ← SSR00L register : Read the status.
Variable work & 0x40 : Extract the TSF00 bit.

Wait for communication completion (TSF00＝0).

Wait for 5 us, allowing a margin.

SOE00 bit ← 0 : Disable IIC00 output.
ST00 bit ← 0 : Stop IIC00 operation.

Figure 5.29 Stopping IIC00 Operation

Checking IIC00 Communication State

- TSF00 bit in the serial status register 00L (SSR00L)
  Wait for the communication completion.

Symbol: SSR00L

7	6	5	4	3	2	1	0
0	TSF00	BFF00	0	0	FEF00	PEF00	OVF00
0	**0**	0	0	0	x	x	x

Bit 5

TSF00	Channel 0 communication state display flag
0	**Communication stopped state or communication operation wait state**
1	通信動作状態

Stopping IIC00 Operation

- SOE00 bit in the serial output enable register 0 (SOE0)
  Disable IIC00 output.
- ST00 bit in the serial channel stop register 0L (ST0L)
  Stop channel 0.

Symbol: SOE0L

7	6	5	4	3	2	1	0
0	0	0	0	0	0	SOE01	SOE00
0	0	0	0	0	0	x	0

Bit 0

SOE00	Enable/disable of channel 0 serial output
0	**Stops serial communication operation output.**
1	シリアル通信動作による出力許可

Symbol: ST0L

7	6	5	4	3	2	1	0
0	0	0	0	0	0	ST01	ST00
0	0	0	0	0	0	x	**1**

Bit 0

ST00	Channel 0 operation stop trigger
0	トリガ動作せず
1	**Clears the SE00 bit to 0 and stops communication operation.**

5.6.27 Waiting for 5 us

Figure 5.30 shows the flowchart for waiting for 5 us.

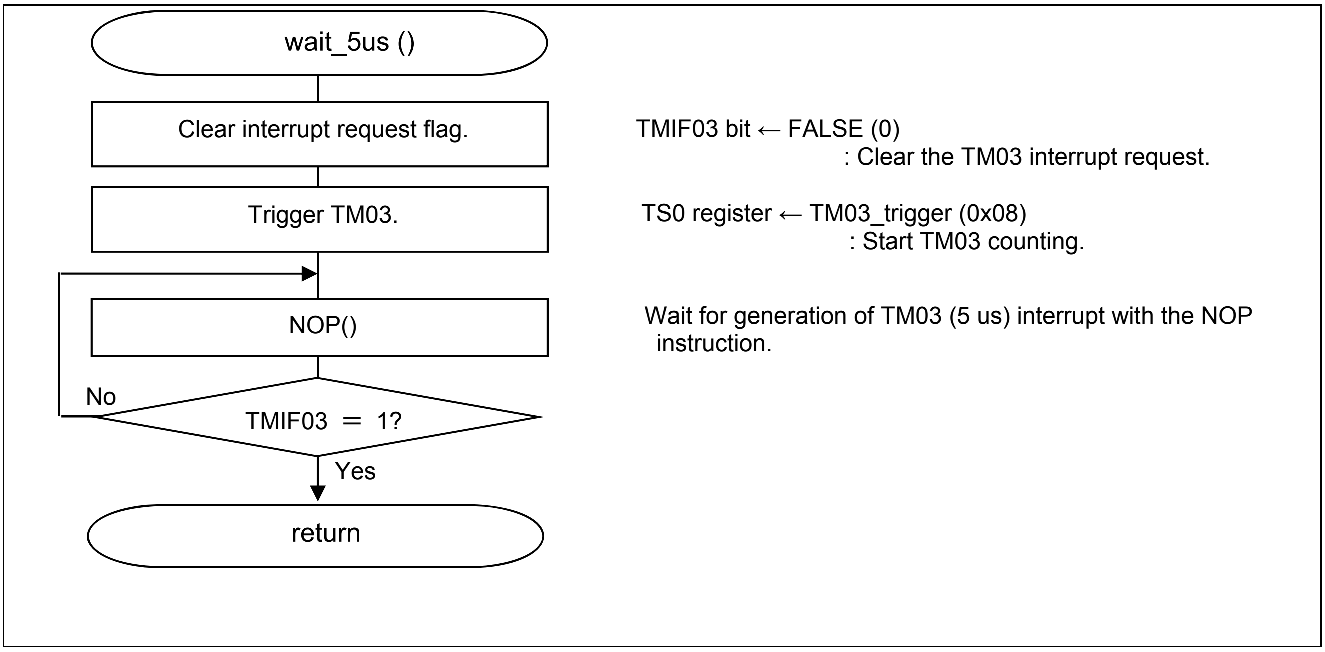

wait_5us ()
Clear interrupt request flag.
Trigger TM03.
NOP()
No
TMIF03 ＝ 1?
Yes
return
TMIF03 bit ← FALSE (0)
: Clear the TM03 interrupt request.
TS0 register ← TM03_trigger (0x08)
: Start TM03 counting.
Wait for generation of TM03 (5 us) interrupt with the NOP instruction.

Figure 5.30 Waiting for 5 us

Clearing timer count end interrupt

- TMIF03 bit in the interrupt request flag register (IF1L)
  Clear the interrupt request flag.

Symbol: IF1L

Bit 5

TMIF03	Interrupt request flag
0	**Interrupt request signal has not been generated.**
1	割り込み要求信号が発生し、割り込み要求状態

Starting TM03 counting operation

- Timer channel start register 0 (TS0)
  Start channel 03 counting operation.

Symbol: TS0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	TS03H	0	TS01H	0	0	0	0	0	TS03	TS02	TS01	TS00
0	0	0	0	0	0	0	0	0	0	0	0	**1**	0	0	0

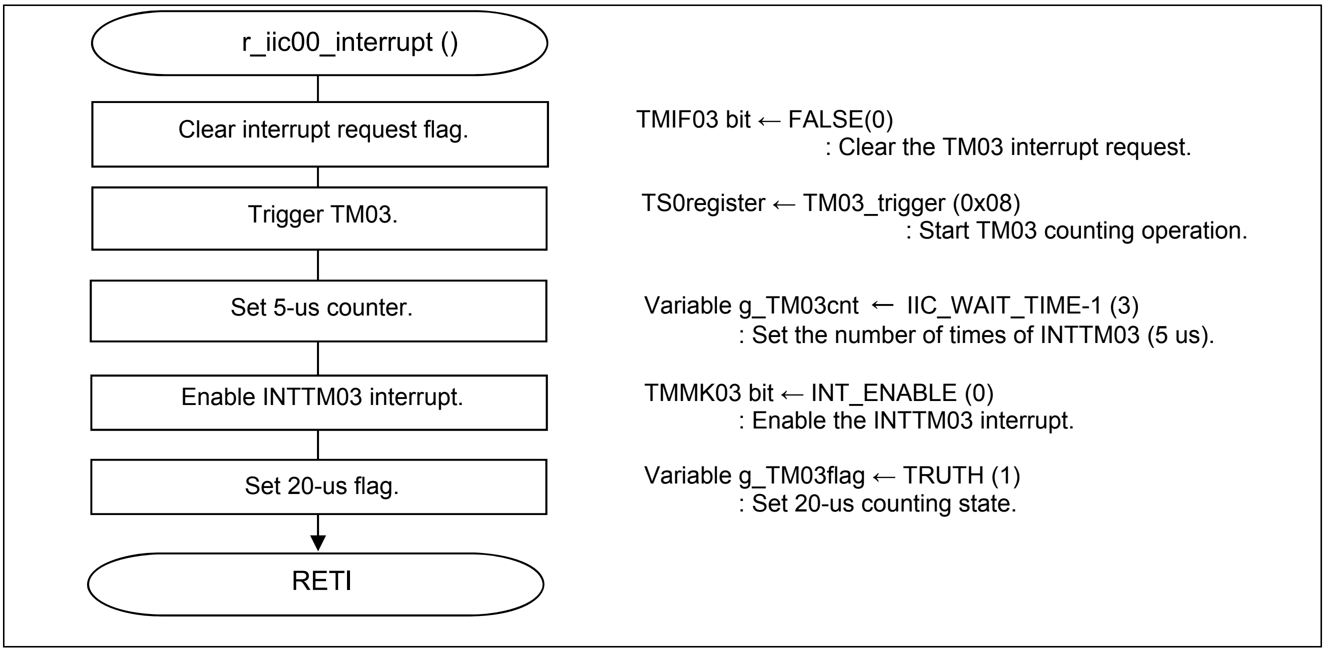
Bit 3

TS03	Channel 03 operation enable trigger
0	トリガ動作しない
1	**Triggers start of counting operation.**

Note: For details of register settings, refer to the RL78/I1D User’s Manual: Hardware.

5.6.28 IIC00 Interrupt Process

Figure 5.31 shows the flowchart of the IIC00 interrupt process.



TMIF03 bit ← FALSE(0)
: Clear the TM03 interrupt request.

TS0register ← TM03_trigger (0x08)
: Start TM03 counting operation.

Variable g_TM03cnt ← IIC_WAIT_TIME-1 (3)
: Set the number of times of INTTM03 (5 us).

TMMK03 bit ← INT_ENABLE (0)
: Enable the INTTM03 interrupt.

Variable g_TM03flag ← TRUTH (1)
: Set 20-us counting state.

Figure 5.31 IIC00 Interrupt Process

Clearing timer count end interrupt

- TMIF03 bit in the interrupt request flag register (IF1L)
  Clear the interrupt request flag.

Symbol: IF1L

Bit 5

TMIF03	Interrupt request flag
0	**Interrupt request signal has not been generated.**
1	割り込み要求信号が発生し、割り込み要求状態

Starting TM03

- Timer channel start register 0 (TS0)
  Start channel 03 counting operation.

Symbol: TS0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	TS03H	0	TS01H	0	0	0	0	0	TS03	TS02	TS01	TS00
0	0	0	0	0	0	0	0	0	0	0	0	**1**	0	0	0

Bit 3

TS03	Channel 03 operation enable trigger
0	トリガ動作しない
1	**Triggers start of counting operation.**

Enabling timer interrupts

- TMMK03 bit in the interrupt mask flag register (MK1L)
  Clear the interrupt request flag.

Symbol: MK1L

Bit 5

TMMK03	Control of interrupt processing
0	**Enables interrupt processing.**
1	割り込み処理禁止

Note: For details of register settings, refer to the RL78/I1D User’s Manual: Hardware.

5.6.29 TM03 Interrupt Process

Figures 5.32 to 5.34 show the flowcharts of the TM03 interrupt process.

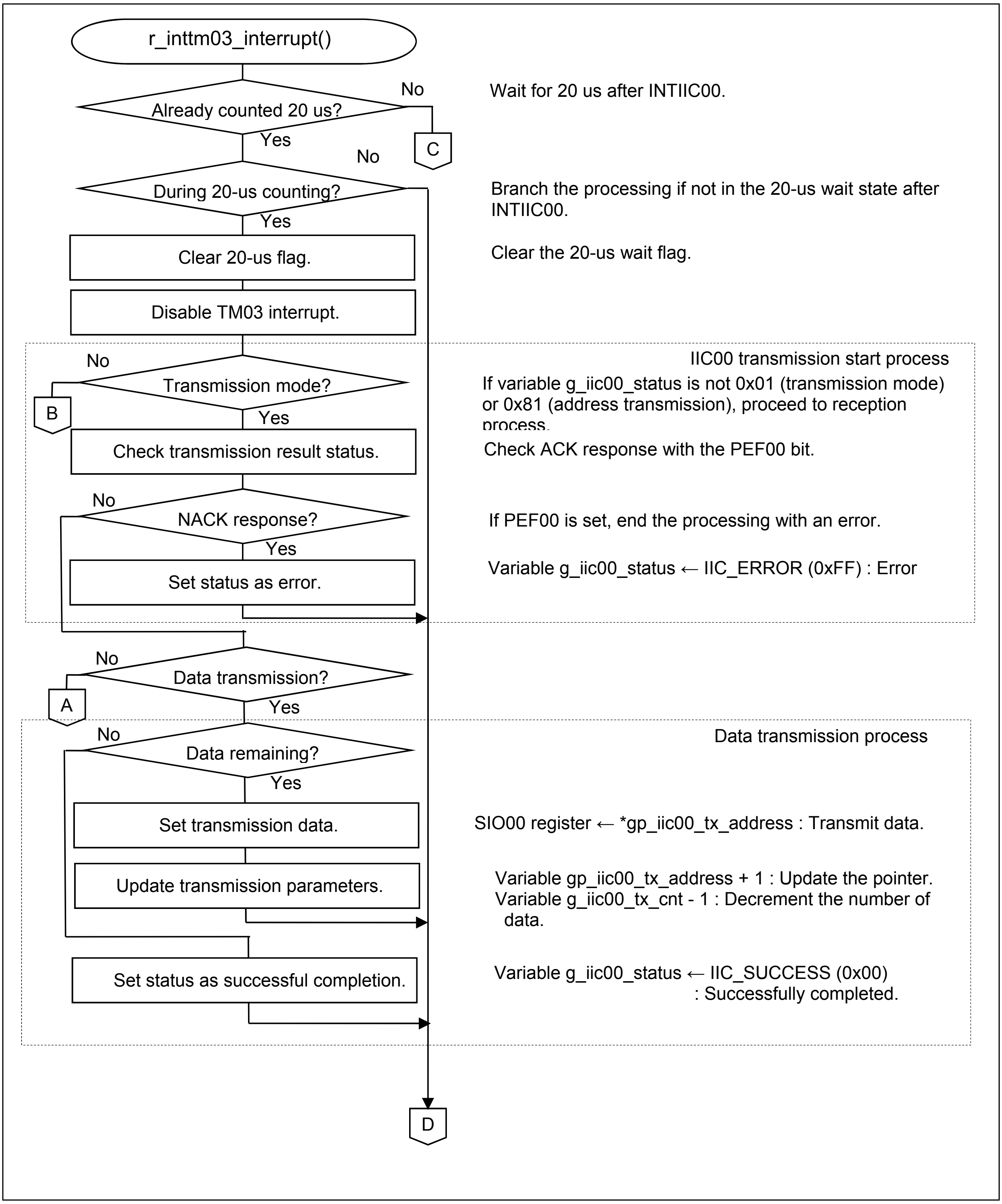

r_inttm03_interrupt()
Already counted 20 us?
No
Yes
C
Wait for 20 us after INTIIC00.
During 20-us counting?
No
Yes
Branch the processing if not in the 20-us wait state after INTIIC00.
Clear 20-us flag.
Clear the 20-us wait flag.
Disable TM03 interrupt.
IIC00 transmission start process
Transmission mode?
No
B
Yes
If variable g_iic00_status is not 0x01 (transmission mode) or 0x81 (address transmission), proceed to reception process.
Check transmission result status.
Check ACK response with the PEF00 bit.
NACK response?
No
Yes
If PEF00 is set, end the processing with an error.
Set status as error.
Variable g_iic00_status ← IIC_ERROR (0xFF) : Error
Data transmission?
No
A
Yes
Data transmission process
Data remaining?
No
Yes
Set transmission data.
SIO00 register ← *gp_iic00_tx_address : Transmit data.
Update transmission parameters.
Variable gp_iic00_tx_address + 1 : Update the pointer.
Variable g_iic00_tx_cnt - 1 : Decrement the number of data.
Set status as successful completion.
Variable g_iic00_status ← IIC_SUCCESS (0x00) : Successfully completed.
D

Figure 5.32 TM03 Interrupt Process (1/3)

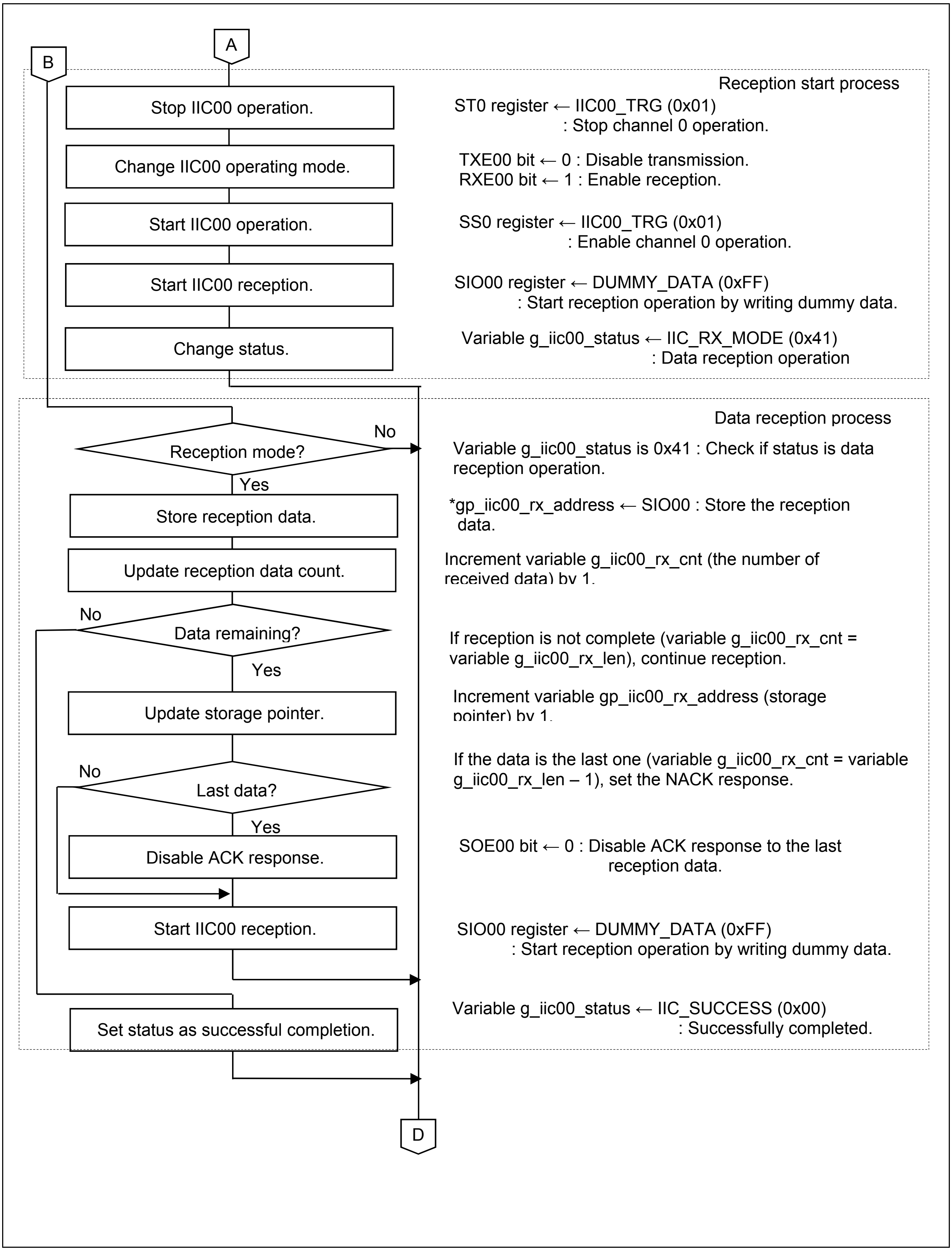

A
B
Reception start process
Stop IIC00 operation.
ST0 register ← IIC00_TRG (0x01)
: Stop channel 0 operation.
Change IIC00 operating mode.
TXE00 bit ← 0 : Disable transmission.
RXE00 bit ← 1 : Enable reception.
Start IIC00 operation.
SS0 register ← IIC00_TRG (0x01)
: Enable channel 0 operation.
Start IIC00 reception.
SIO00 register ← DUMMY_DATA (0xFF)
: Start reception operation by writing dummy data.
Change status.
Variable g_iic00_status ← IIC_RX_MODE (0x41)
: Data reception operation
Data reception process
Reception mode?
No
Yes
Variable g_iic00_status is 0x41 : Check if status is data reception operation.
Store reception data.
*gp_iic00_rx_address ← SIO00 : Store the reception data.
Update reception data count.
Increment variable g_iic00_rx_cnt (the number of received data) by 1.
Data remaining?
No
Yes
If reception is not complete (variable g_iic00_rx_cnt = variable g_iic00_rx_len), continue reception.
Update storage pointer.
Increment variable gp_iic00_rx_address (storage pointer) by 1.
Last data?
No
Yes
If the data is the last one (variable g_iic00_rx_cnt = variable g_iic00_rx_len – 1), set the NACK response.
Disable ACK response.
SOE00 bit ← 0 : Disable ACK response to the last reception data.
Start IIC00 reception.
SIO00 register ← DUMMY_DATA (0xFF)
: Start reception operation by writing dummy data.
Set status as successful completion.
Variable g_iic00_status ← IIC_SUCCESS (0x00)
: Successfully completed.
D

Figure 5.33 TM03 Interrupt Process (2/3)

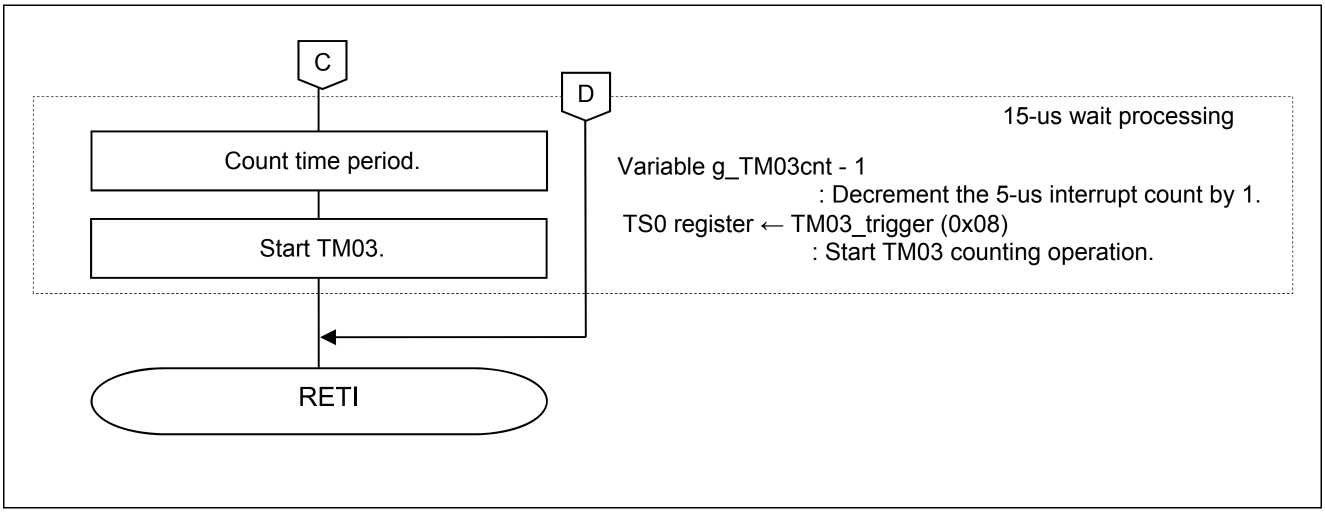

C
D
15-us wait processing
Count time period.
Start TM03.
Variable g_TM03cnt - 1
: Decrement the 5-us interrupt count by 1.
TS0 register ← TM03_trigger (0x08)
: Start TM03 counting operation.
RETI

Figure 5.34 TM03 Interrupt Process (3/3)

Checking IIC00 status.

- PEF00 bit in the serial status register 00L (SSR00L)
  Check if the response is an ACK response.

Symbol: SSR00L

7	6	5	4	3	2	1	0
0	TSF00	BFF00	0	0	FEF00	PEF00	OVF00
0	x	x	0	0	x	**0/1**	x

Bit 1

PEF00	Channel 0 parity error detection flag
0	**ACK response from the slave**
1	**NACK response from the slave**

Clearing IIC00 error flags

- Serial flag clear trigger register 00L (SIR00L)
  Clear the error flag.

Symbol: SIR00L

7	6	5	4	3	2	1	0
0	0	0	0	0	FECT00	PECT00	OVCT00
0	0	0	0	0	**0/1**	**0/1**	**0/1**

Bit 1

PECT00	Channel 0 parity error flag clearance trigger
0	**Does not perform clearance.**
1	**Clears PEF00 bit in SSR00 register to 0.**

Transmitting data

- Lower bits in the serial data register 00 (SDR00) (SIO00 register)
  Set transmission data.

Symbol: SIO00

7	6	5	4	3	2	1	0
x	x	x	x	x	x	x	x

Stopping IIC00 operation

- ST00 bit in the serial channel stop register 0L (ST0L)
  Stop channel-0 operation.

Symbol: ST0L

7	6	5	4	3	2	1	0
0	0	0	0	0	0	ST01	ST00
0	0	0	0	0	0	0	**1**

Bit 0

ST00	Channel 0 operation stop trigger
0	トリガ動作せず
1	**Clears SE00 bit to 0 and stops communication operation.**

Changing IIC00 operating modes

- TXE00 and RXE00 bits in the serial communication operation setting register (SCR00)
  Disable transmission and enable reception.

Symbol: SCR00

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXE00	RXE00	DAP00	CKP00	0	EOC00	PTC001	PTC000	DIR00	0	SLC001	SLC000	0	1	DLS001	DLS000
0	**1**	0	0	0	0	0	0	0	0	0	1	0	1	1	1

Bits 15 and 14

TXE00	RXE00	Channel 0 operating mode setting
0	0	通信禁止
0	**1**	**Enables reception only.**
1	0	送信のみを行う
1	1	送受信を行う（簡易 I2C では設定禁止）

Starting IIC00 operation

- SS00 bit in the serial channel start register 0L (SS0L)
  Start channel 0 operation.

Symbol: SS0L

7	6	5	4	3	2	1	0
0	0	0	0	0	0	SS01	SS00
0	0	0	0	0	0	0	**1**

Bit 0

SS00	Channel 0 operation start trigger
0	トリガ動作せず
1	**Sets SE00 bit to 1 and enters the communication wait state.**

Starting data reception

- Lower bits in the serial data register 00 (SDR00) (SIO00 register)
  Set dummy data.

Symbol: SIO00

7	6	5	4	3	2	1	0
1	1	1	1	1	1	1	1

Reading reception data

- Lower bits in the serial data register 00 (SDR00) (SIO00 register)
  Read out the received data.

Symbol: SIO00

7	6	5	4	3	2	1	0
x	x	x	x	x	x	x	1

Disabling ACK response

- SOE00 bit in the serial output enable register 0L (SOE0L)
  Disable serial output.

Symbol: SOE0L

7	6	5	4	3	2	1	0
0	0	0	0	0	0	SOE01	SOE00
0	0	0	0	0	0	x	**0**

Bit 0

SOE00	Enable/disable of serial output in channel 0
0	**Stops output by serial communication operation.**
1	シリアル通信動作による出力許可

Disabling ACK response

- SOE00 bit in the serial output enable register 0L (SOE0L)
  Disable serial output.

Starting TM03

- Timer channel start register 0 (TS0)
  Start channel 03 counting operation.

Symbol: TS0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	TS03H	0	TS01H	0	0	0	0	0	TS03	TS02	TS01	TS00
0	0	0	0	0	0	0	0	0	0	0	0	**1**	0	0	0

Bit 3

TS03	Channel 03 operation enable trigger
0	トリガ動作しない
1	**Triggers start of counting operation.**

Note: For details of register settings, refer to the RL78/I1D User's Manual: Hardware.

6. Notes

6.1 Wait Function Control

The simple I2C does not have the wait function for synchronization between the master and slave. This is not a problem with the device such as EEPROM, which controls communication by using hardware; however, if the microcontroller is used as the slave, timing gap may be generated thus disabling successful communication, if no measures are taken.

To prevent this problem, in the sample program here, the master side (simple I2C) secures the time (default value is 20 µs) after each data transfer (after every nine SCL clock cycles) before the start of the next communication.

Therefore, in the INTIIC00 process, by triggering TM03 (delay counting function used), a transition is made to the INTTM03 wait state. After INTTM03 is generated for the previously specified number of times (default value is 4 = 20 µs), the communication completion process (starting the next communication, etc.) is performed. Figure 6-1 shows an example of timing from completion of one data communication to start of the next communication.

The number of TM03 interrupt times for securing the wait time is specified with the constant (IIC_WAIT_TIME). If necessary, modify the IIC_WAIT_TIME definition according to the slave processing time.

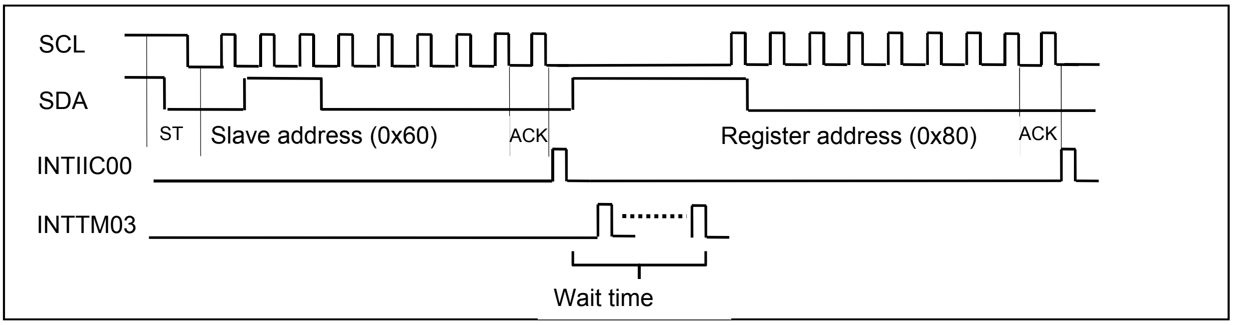


Figure 6.1 Timing between Communications

With the above sequence, if the slave side can complete necessary processing within 20 µs, communication can be successfully performed. Note that communication cannot be stopped at the eighth clock cycle; be sure to set for the slave side so that the wait/interrupt is generated at the ninth clock cycle.

In addition, TM03 is also used for the start and stop conditions to secure 5 µs and the slave processing time.

6.2 Settings through Code Creation

Set the following items with "API function output control" of "File creation mode" of Property set to "Output only initialization functions".

(1) Clock generation circuit settings

(a) Pin assignment settings: Fix as they are.

(b) Clock settings

- Operating mode settings: High-speed main mode 2.7 (V) ≤ VDD ≤ 3.6 (V)
- Main system clock (fMAIN) settings: High-speed on-chip oscillator clock (fIH)
- High-speed on-chip oscillator clock settings: 24 (MHz)
- Middle-speed on-chip oscillator clock settings: Do not check "operate".
- High-speed system clock settings: Do not check "operate".
- Subsystem clock (fSUB) settings: Low-speed system clock: Low-speed on-chip oscillator clock (fIL)
- Low-speed on-chip oscillator clock (fIL) settings: Frequency 15 (kHz)
- RTC, FMC, interval timer, PCLBUZ operating clock settings: 15 (fIL) (kHz)
- CPU and peripheral clock settings: 24000 (fIH) (kHz)

(c) On-chip debugging settings

- On-chip debugging operation settings: Use.
- RRM function settings: Do not use.
- Security ID settings: Set security ID.
- Settings upon security ID authentication failure: Delete flash memory data.

(d) Reset source checking

- Reset source checking function output: Remove the check mark.

(e) Safety functions: Select "Do not use" for all.

(f) Data flash: Prohibit access to data flash.

(2) Port settings

Set P60 and P61 to output 1.

Leave the other ports (Do not use).

(3) Timer settings

(a) General settings Channel 3: Delay counting function

(b) Channel 3

- Operating mode settings: 16 bits
- Delay counting time settings: 5 µs

Leave the other settings as default.

(4) Realtime clock settings

Leave as default (Do not use).

(5) Frequency measurement circuit

Leave as default (Do not use).

(6) 12-bit interval timer settings

- Interval timer operation settings: Use.
- Interval time settings: 50 ms

Leave the other settings as default.

(7) 8-bit interval timer

Leave as default (Do not use).

(8) Clock output/buzzer output settings

Leave as default (Do not use).

(9) Watchdog timer settings

- Operation settings in HALT/STOP/SNOOZE mode: Stop.
- Watchdog timer operation settings: Do not use.

(10) A/D converter settings

Leave as default (Do not use).

(11) Comparator

Leave as default (Do not use).

(12) Op amp.

Leave as default (Do not use).

(13) Serial array unit settings

(a) Channel Channel 0: Set to IIC00.

(b) IIC00 settings

- Transfer rate: Set 200000 (bps).

Leave the other settings as default.

(14) Data operation circuit

Leave as default (Do not use)

(15) Data transfer controller

Leave as default (all check marks removed: Not used)

(16) Event link controller

Leave as default (all check marks removed: Do not use).

(17) Interrupt settings

Leave all interrupts as default (no check marks).

(18) Key interrupt function

Leave all interrupts as default (no check marks)

(19) Voltage detection circuit settings

- Voltage detection operation settings: Use.
- Operating mode settings: Reset mode
- Detected voltage settings: 2.75 (V)

7. Sample Code

The user can get the sample code from the Renesas Electronics website.

8. Reference Documents

RL78/I1D User's Manual: Hardware Rev.2.10 (R01UH0474E)

RL78 Family User's Manual: Software Rev.2.00 (R01US0015E)

(Get the latest version from the Renesas Electronics website.)

Technical Updates/Technical News

(Get the latest information from the Renesas Electronics website.)

Website and Support

Renesas Electronics Website

http://japan.renesas.com/

Inquiries

http://japan.renesas.com/inquiry

Revision History

Rev.	Date	Revision Contents	
		Page	Point
1.00	2016.11.15		Newly created.

General Precautions in the Handling of MPU/MCU Products

The following usage notes are applicable to all MPU/MCU products from Renesas. For detailed usage notes on the products covered by this document, refer to the relevant sections of the document as well as any technical updates that have been issued for the products.

1. Handling of Unused Pins

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

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

2. Processing at Power-on

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

   — The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the moment when power is supplied.
   In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the moment when power is supplied until the reset process is completed.
   In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the moment when power is supplied until the power reaches the level at which resetting has been specified.

3. Prohibition of Access to Reserved Addresses

   Access to reserved addresses is prohibited.

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

4. Clock Signals

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

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

5. Differences between Products

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

   — The characteristics of an MPU or MCU in the same group but having a different part number may differ in terms of the internal memory capacity, layout pattern, and other factors, which can affect the ranges of electrical characteristics, such as characteristic values, operating margins, immunity to noise, and amount of radiated noise. When changing to a product with a different part number, implement a system-evaluation test for the given product.

Notice

1. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
3. Renesas Electronics does not assume any liability for infringement of patents, copyrights, or other intellectual property rights of third parties by or arising from the use of Renesas Electronics products or technical information described in this document. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
4. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from such alteration, modification, copy or otherwise misappropriation of Renesas Electronics product.
5. Renesas Electronics products are classified according to the following two quality grades: "Standard" and "High Quality". The recommended applications for each Renesas Electronics product depends on the product's quality grade, as indicated below.

   "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; and industrial robots etc.

   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; and safety equipment etc.

   Renesas Electronics products are neither intended nor authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems, surgical implantations etc.), or may cause serious property damages (nuclear reactor control systems, military equipment etc.). You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application for which it is not intended. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for which the product is not intended by Renesas Electronics.
6. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
7. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or systems manufactured by you.
8. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
9. Renesas Electronics products and technology may not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations. You should not use Renesas Electronics products or technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. When exporting the Renesas Electronics products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations.
10. It is the responsibility of the buyer or distributor of Renesas Electronics products, who distributes, disposes of, or otherwise places the product with a third party, to notify such third party in advance of the contents and conditions set forth in this document, Renesas Electronics assumes no responsibility for any losses incurred by you or third parties as a result of unauthorized use of Renesas Electronics products.
11. This document may not be reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.

(Note 1) "Renesas Electronics" as used in this document means Renesas Electronics Corporation and also includes its majority-owned subsidiaries.

(Note 2) "Renesas Electronics product(s)" means any product developed or manufactured by or for Renesas Electronics.

RENESAS

Renesas Electronics Corporation

http://www.renesas.com

SALES OFFICES

Refer to "http://www.renesas.com/" for the latest and detailed information.

Renesas Electronics America Inc.
2801 Scott Boulevard Santa Clara, CA 95050-2549, U.S.A.
Tel: +1-408-588-6000, Fax: +1-408-588-6130

Renesas Electronics Canada Limited
9251 Yonge Street, Suite 8309 Richmond Hill, Ontario Canada L4C 9T3
Tel: +1-905-237-2004

Renesas Electronics Europe Limited
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K
Tel: +44-1628-585-100, Fax: +44-1628-585-900

Renesas Electronics Europe GmbH
Arcadiastrasse 10, 40472 Düsseldorf, Germany
Tel: +49-211-6503-0, Fax: +49-211-6503-1327

Renesas Electronics (China) Co., Ltd.
Room 1709, Quantum Plaza, No.27 ZhiChunLu Haidian District, Beijing 100191, P.R.China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

Renesas Electronics (Shanghai) Co., Ltd.
Unit 301, Tower A, Central Towers, 555 Langao Road, Putuo District, Shanghai, P. R. China 200333
Tel: +86-21-2226-0888, Fax: +86-21-2226-0999

Renesas Electronics Hong Kong Limited
Unit 1601-1611, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2265-6688, Fax: +852 2886-9022

Renesas Electronics Taiwan Co., Ltd.
13F, No. 363, Fu Shing North Road, Taipei 10543, Taiwan
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670

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

Renesas Electronics Malaysia Sdn.Bhd.
Unit 1207, Block B, Menara Amcorp, Amcorp Trade Centre, No. 18, Jln Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: +60-3-7955-9390, Fax: +60-3-7955-9510

Renesas Electronics India Pvt. Ltd.
No.777C, 100 Feet Road, HAL II Stage, Indiranagar, Bangalore, India
Tel: +91-80-67208700, Fax: +91-80-67208777

Renesas Electronics Korea Co., Ltd.
12F., 234 Teheran-ro, Gangnam-Gu, Seoul, 135-080, Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5141

Colophon 5.0