RL78/G15

Serial Interface IICA (for Master Transmission/Reception)

Introduction

This application note describes master transmission and reception implemented via serial interface IICA. Using IICA, the single master system described here performs master operation (address transmission, data transmission and data reception).

Target Device

RL78/G15

When applying the sample program covered in this application note to another microcomputer, modify the program according to the specifications for the target microcomputer and conduct an extensive evaluation of the modified program.

Contents

1. Specifications

This application note describes a method for using a serial interface IICA for master transmission/reception (address transmission data transmission and reception) in a single-master configuration. It is assumed that each slave has a register to specify an address in the slave.

The master specifies a slave address to select the slave. When communication with the slave is established, data transmission and reception become possible.

The following is a summary of the slave specifications assumed in this application note.

- Slave address: 0b1010000[note]
- At the specified address, an arbitrary number of data bytes can be read out or written.
- The slave serial RAM area is register addresses 0x80 to 0xFF (128 bytes). Slave operation is specified using the command register at register address 0x00.
- When 0x01 to 0x7F is specified as the register address, NACK is returned and there is disengagement from communication.
- When the register address exceeds 0xFF, the serial RAM area is selected, and only the lower 7 bits of the specified register address are handled as valid.

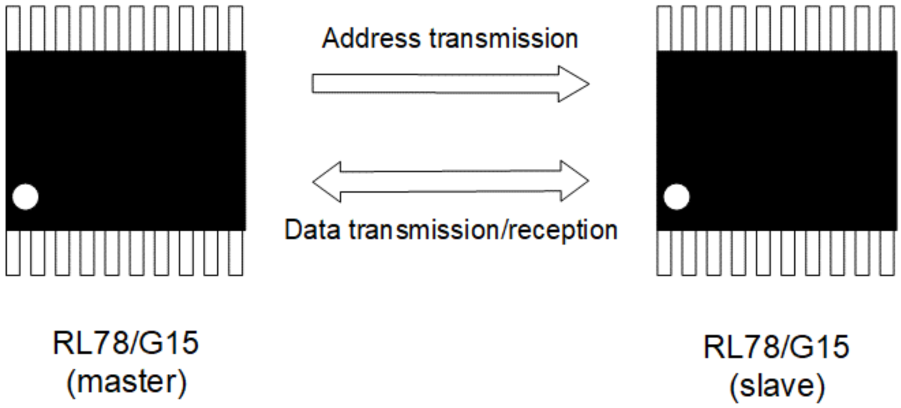
Note: In the RL78 family, the local address (7 bits) is represented by the upper 7 bits of the SVA0 register. The lowermost bit of the SVA0 register is fixed at 0. In address transmission the slave address and the transfer direction (R/$\overline{\text{W}}$) are combined and the 8 bits are written to the IICA shift register 0 (IICA0).

Caution: This sample code corresponds to the Application Note for the RL78/G15 serial interface IICA (slave transmission/reception) (R01ANI7021E).

Peripheral functions used and applications are shown in Table 1.1, and the IIC communication is summarized in Figure 1.1. IIC communication timing charts appear in Figure 1.2 to Figure 1.8.

Table 1.1 Peripheral Function to be Used and Its Use

Peripheral Function	Use
Serial interface IICA	IIC communication in a single master system (using the SCLA0 and SDAA0 pins)

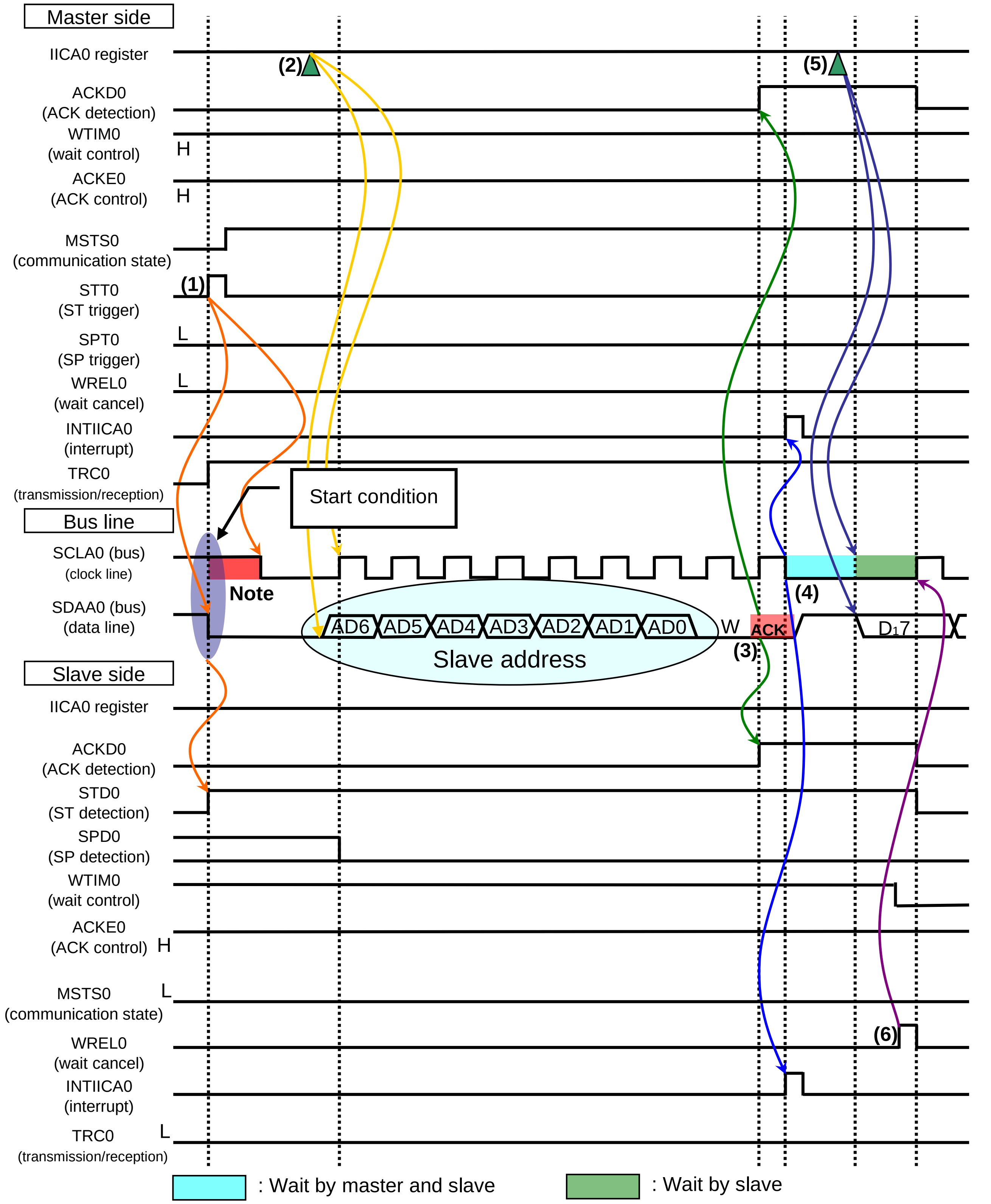


Address transmission: Each device connected to an IIC bus has a unique address. The master transmits a slave address for transmission. The address (7 bits) and the transfer direction (1 bit) are transmitted as one data byte. After receiving the one byte of data, the slave generates an acknowledge signal.
Data transmission/reception: Operation, after address reception, to transmit data to the master or to receive data from the master.

Figure 1.1 Overview of IIC Communication

1.1 IIC Communication Timing Chart

(1) Master-to-slave communication 1 (start condition – address – data)



Note: The time between the falling edges of SDAA0 and SCLA0 is 4.0 μs or more for the standard mode and 0.6 μs or more for the fast mode.

Figure 1.2 IIC Communication Timing Chart (Master-to-Slave Communication Example) (1/4)

(1) The start condition trigger is set (STT0 = 1) on the master side. Then, the SDAA0 line falls, thereby generating a start condition. Later, when the start condition is detected (STD0 = 1), the master enters a master device communication state (MSTS0 = 1). The SCLA0 line falls at the end of the hold period. This completes preparations for communication.
(2) The values of the address and data direction bit W (transmission) are written to the IICA0 register on the master side. Then, the slave address is transmitted.
(3) If the received address and slave address match [Note], the slave hardware sends ACK0 to the master. When the ninth clock signal rises, the master detects ACK (ACKD0 = 1).
(4) When the ninth clock signal falls, an address transmission end interrupt (INTIICA0) occurs on the master side. If the addresses match, an address match interrupt (INTIICA0) occurs on the slave side. Both the master and the slave which has the matching address generate a wait (SCLA0 line: Low)[Note].
(5) The master writes transmit data to the IICA0 register and cancels the wait.
(6) The slave selects an 8-clock wait (WTIM0 = 0) because it receives data. When the slave cancels the wait (WREL0 = 1), the master starts transferring data to the slave.

Note: When there is a mismatch between a received address and the local address, the slave side does not return ACK to the master side (NACK). Moreover, a slave-side INTIICAO interrupt (address match interrupt) does not occur, and a wait state is not entered on the slave side. However, on the master side an INTIICAO interrupt (address transmission complete interrupt) occurs, regardless of whether the result is ACK or NACK.

In the RL78 family, the local address (7 bits) is represented by the upper 7 bits in the SVA0 register. The lowermost bit in the SVA0 register is fixed at 0. In address transmission, the slave address and the transfer direction (R / $\overline{\text{W}}$) are combined and written as 8 bits to the IICA shift register 0 (IICA0).

(2) Master-to-slave communication 2 (address – data – data)

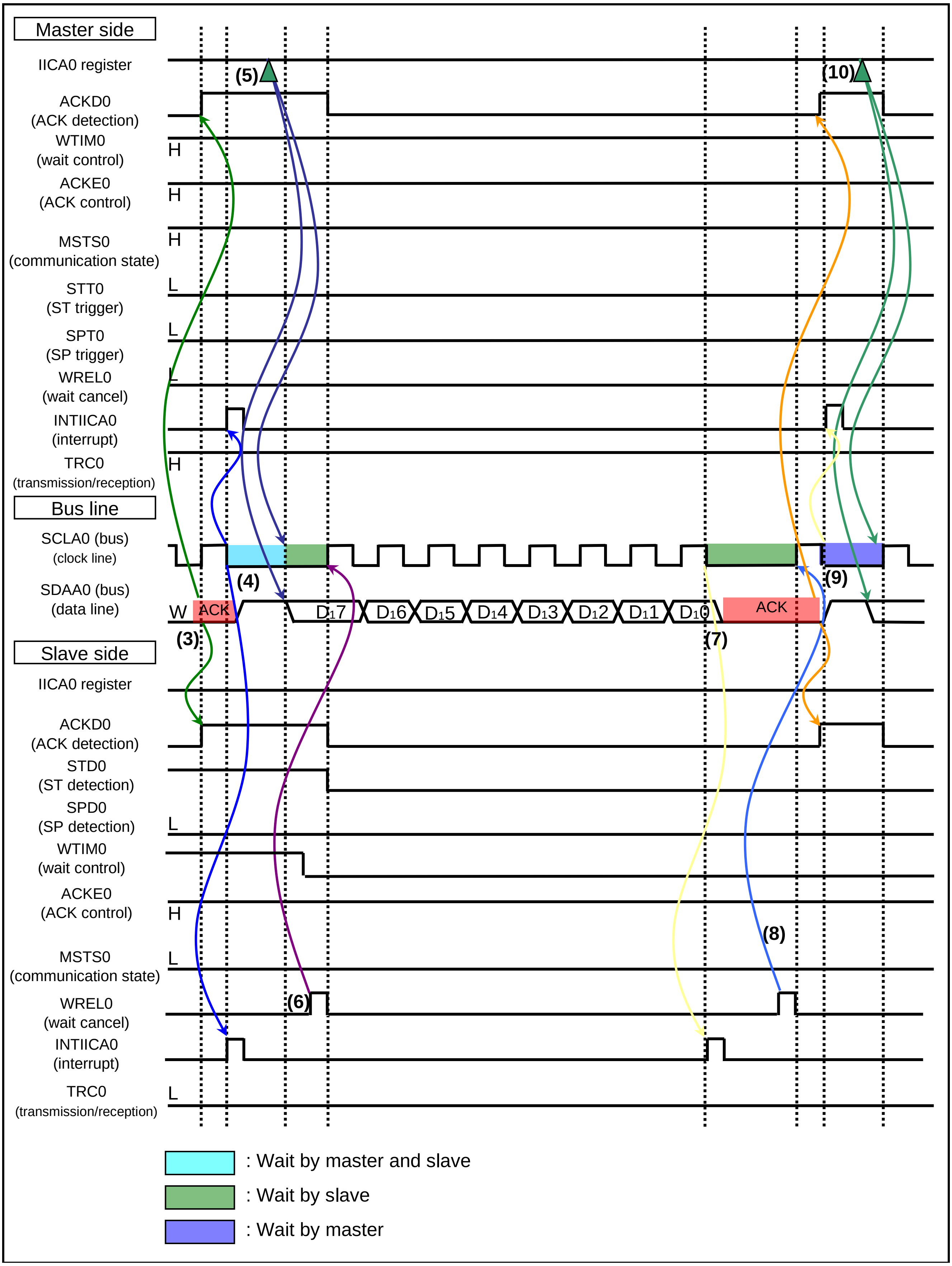
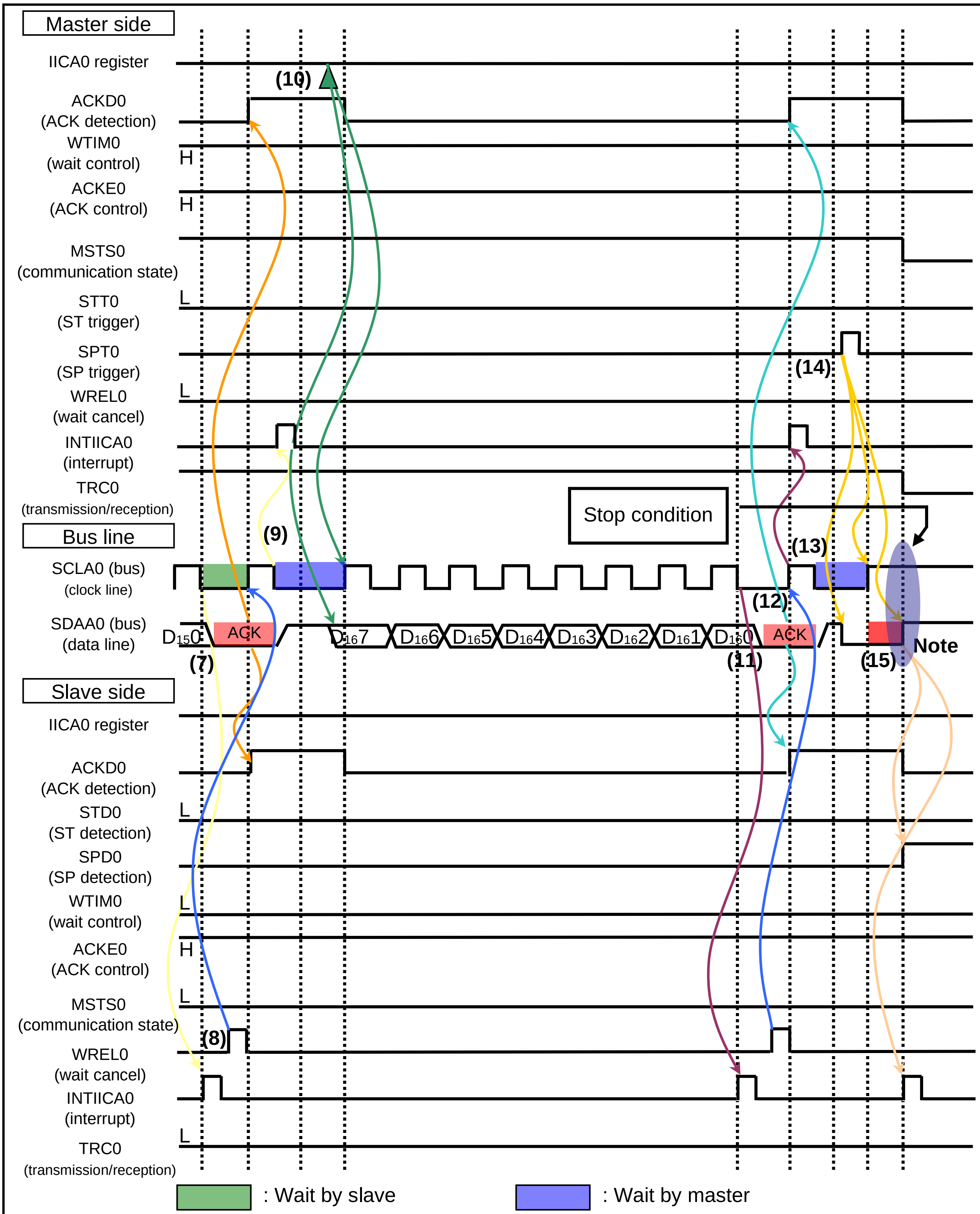


Figure 1.3 IIC Communication Timing Chart (Master-to-Slave Communication Example) (2/4)

(3) If the received address and slave address match, the slave hardware sends ACK to the master. When the ninth clock signal rises, the master detects ACK (ACKD0 = 1).
(4) When the ninth clock signal falls, an address transmission end interrupt (INTIICA0) occurs on the master side. If the addresses match, an address match interrupt (INTIICA0) occurs on the slave side. Both the master and the slave which has the matching address generate a wait (SCLA0 line: Low).
(5) The master writes transmit data to the IICA0 register and cancels the wait.
(6) The slave selects an 8-clock wait (WTIM0 = 0) because it receives data. When the slave cancels the wait (WREL0 = 1), the master starts transferring data to the slave.
(7) When the eighth clock signal falls after the data transfer, the slave hardware generates a wait (SCLA0 line: Low) and a transfer end interrupt (INTIICA0) occurs on the slave side.
(8) When the slave reads the receive data and cancels the wait (WREL0 = 1), the slave sends ACK to the master. When the ninth clock signal rises, the master detects ACK (ACKD0 = 1).
(9) When the ninth clock signal falls, the master generates a wait (SCLA0 line: Low) and a transfer end interrupt (INTIICA0) occurs on the master side.
(10) The master writes transmit data to the IICA0 register and cancels the wait. Then, the master starts transferring data to the slave.

(3) Master-to-slave communication 3 (data – data – stop condition)

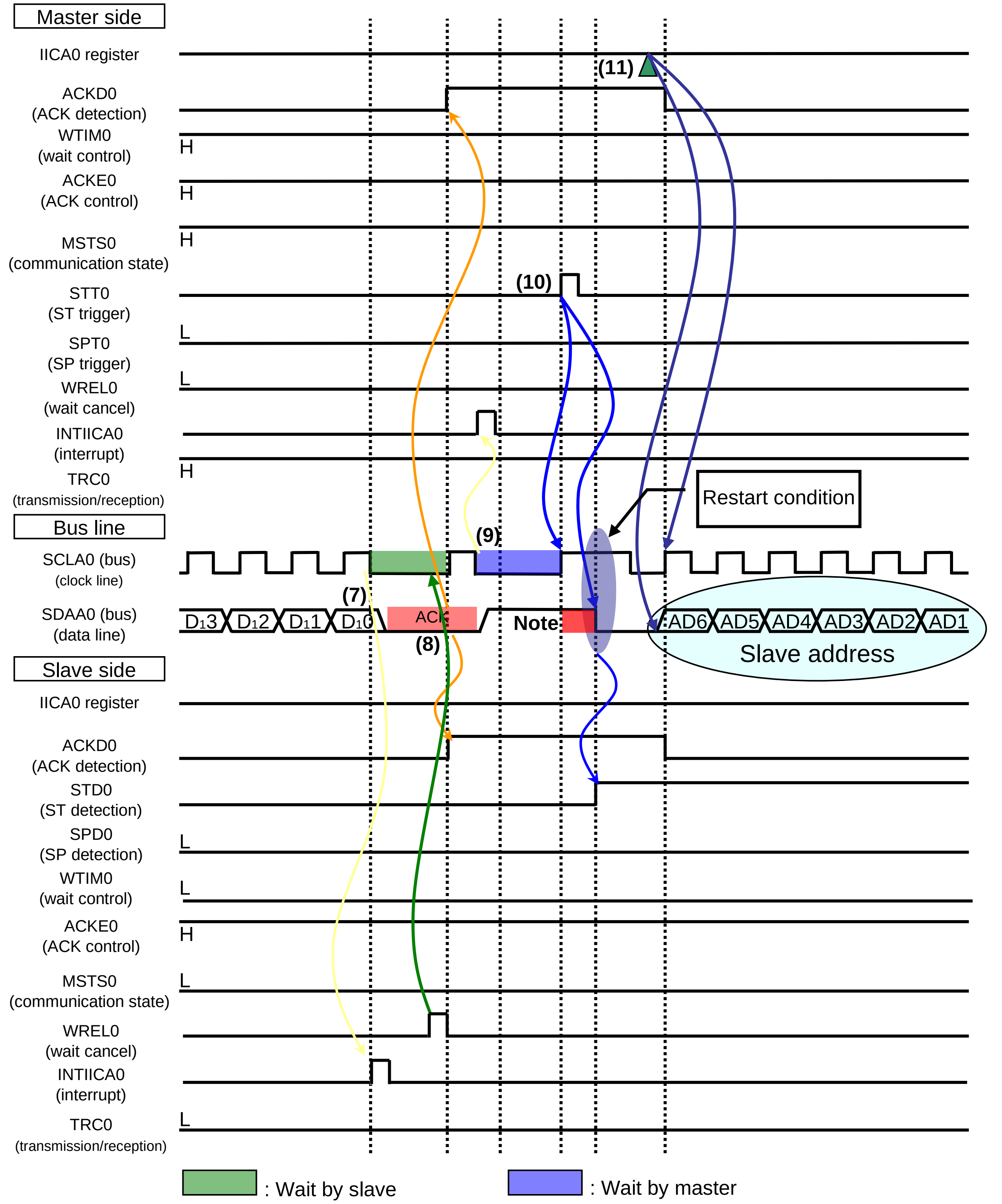


Note: The time from when SCLA0 rises after a stop condition is issued till when a stop condition is generated is 4.0 μs or more for the standard mode and 0.6 μs or more for the fast mode.

Figure 1.4 IIC Communication Timing Chart (Master-to-Slave Communication Example) (3/4)

(7) When the eighth clock signal falls after the data transfer, the slave hardware generates a wait (SCLA0 line: Low) and a transfer end interrupt (INTIICA0) occurs on the slave side.
(8) When the slave reads the receive data and cancels the wait (WREL0 = 1), the slave sends ACK to the master. When the ninth clock signal rises, the master detects ACK (ACKD0 = 1).
(9) When the ninth clock signal falls, the master generates a wait (SCLA0 line: Low) and an address transmission end interrupt (INTIICA0) occurs on the master side.
(10) The master writes transmit data to the IICA0 register and cancels the wait. Then, the master starts transferring the data to the slave.
(11) When the eighth clock signal falls after the data transfer, the slave hardware generates a wait (SCLA0 line: Low) and a transfer end interrupt (INTIICA0) occurs on the slave side.
(12) When the slave reads the receive data and cancels the wait (WREL0 = 1), the slave sends ACK to the master. When the ninth clock signal rises, the master detects ACK (ACKD0 = 1).
(13) When the ninth clock signal falls, the master generates a wait (SCLA0 line: Low) and a transfer end interrupt (INTIICA0) occurs on the master side.
(14) When the stop condition trigger is set (SPT0 = 1), the SDAA0 line falls and the SCLA0 line rises. Upon the elapse of the stop condition setup time, the SDAA0 line rises, thereby generating a stop condition.
(15) When the stop condition is generated, the slave detects it (SPD0 = 1) and a IICA0 interrupt (stop condition interrupt) occurs on the slave side.

(4) Master-to-slave communication 4 (data – restart condition – address)



Note: The time from when SCLA0 rises after a restart condition is issued till when a start condition is generated is 4.7 μs or more for the standard mode and 0.6 μs or more for the fast mode.

Figure 1.5 IIC Communication Timing Chart (Master-to-Slave Communication Example) (4/4)

(7) When the eighth clock signal falls after the data transfer, the slave hardware generates a wait (SCLA0 line: Low) and a transfer end interrupt (INTIICA0) occurs on the slave side.

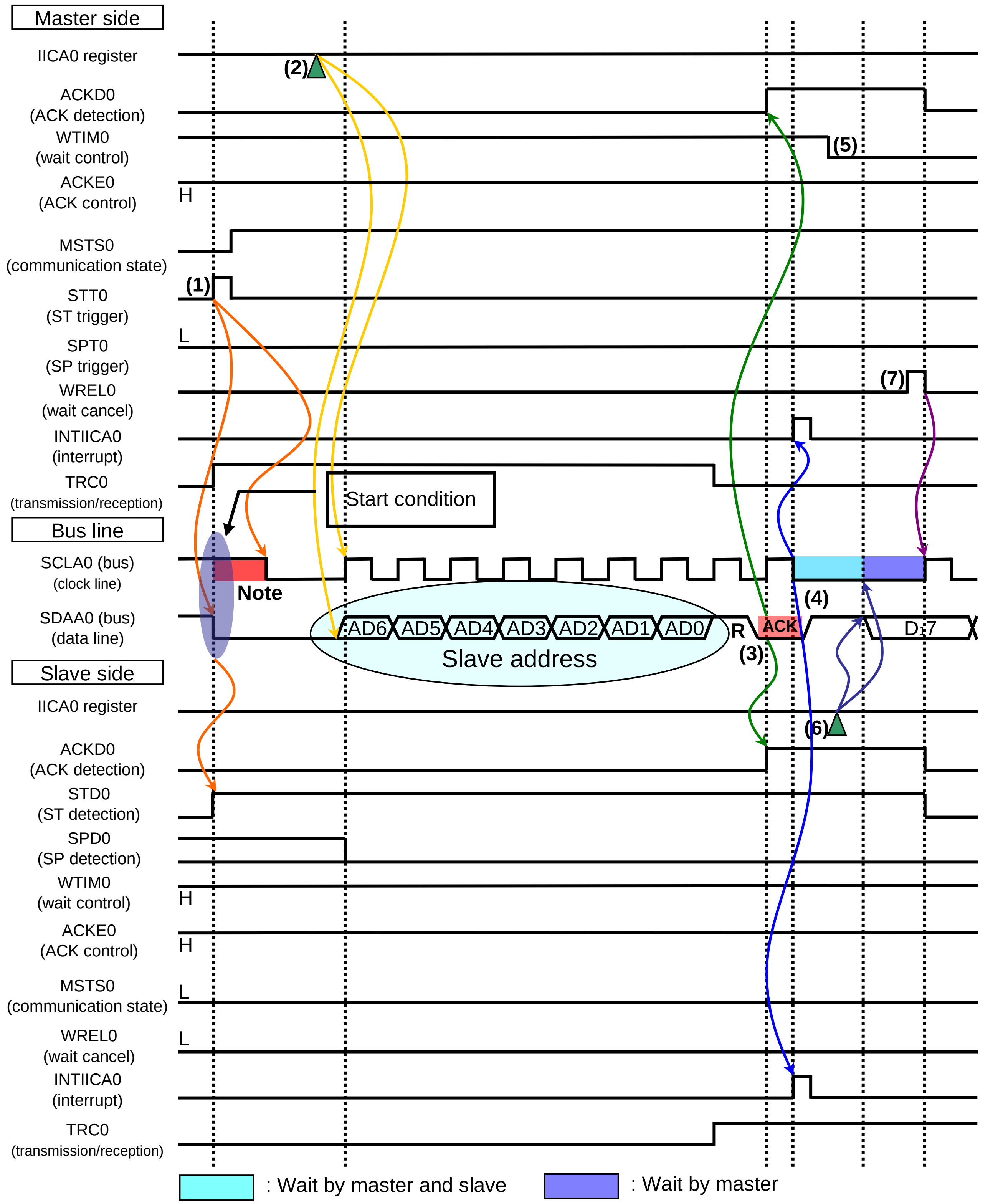
(8) The slave reads the receive data and cancels the wait (WREL0 = 1). Then, the slave sends ACK to the master. When the ninth clock signal rises, the master detects ACK (ACKD0 = 1).

(9) When the ninth clock signal falls, the master generates a wait (SCLA0 line: Low) and a transfer end interrupt (INTIICA0) occurs on the master side.

(10) The start condition trigger is set (STT0 = 1) on the master side again. Then, the SCLA0 line rises. Upon the elapse of the restart condition setup time, the SDAA0 line falls, thereby generating a start condition. Later, at the end of the hold period after the start condition is detected (STD0 = 1), the bus clock line falls, thereby completing preparations for communication.

(11) The master writes the slave address to the IICA0 register and starts transferring the address to the slave.

(5) Slave-to-master communication 1 (start condition – address – data)



Note: The time between the falling edges of SDAA0 and SCLA0 is 4.0 μs or more for the standard mode and 0.6 μs or more for the fast mode.

Figure 1.6 IIC Communication Timing Chart (Slave-to-Master Communication Example) (1/3)

(1) The start condition trigger is set (STT0 = 1) on the master side. Then, the SDAA0 line falls, thereby generating a start condition. Later, when the start condition is detected (STD0 = 1), the master enters a master device communication state (MSTS0 = 1). The SCLA0 line falls at the end of the hold period. This completes preparations for communication.
(2) The values of the address and data direction bit R (reception) are written to the IICA0 register on the master side. Then, the slave address is transmitted.
(3) If the received address and slave address match [Note], the slave hardware sends ACK to the master. When the ninth clock signal rises, the master detects ACK (ACKD0 = 1).
(4) When the ninth clock signal falls, an address transmission end interrupt (INTIICA0) occurs on the master side. If the addresses match, an address match interrupt (INTIICA0) occurs on the slave side. Both the master and the slave which has the matching address generate a wait (SCLA0 line: Low).
(5) The master selects an 8-clock wait (WTIM0 = 0) because it receives data.
(6) The slave writes transmit data to the IICA0 register and cancels the wait.
(7) When the master cancels the wait (WREL0 = 1), the slave starts transferring data to the master.

Note When there is a mismatch between a received address and the local address, the slave side does not return ACK to the master side (NACK). Moreover, a slave-side INTIICAO interrupt (address match interrupt) does not occur, and a wait state is not entered on the slave side. However, on the master side an INTIICAO interrupt (address transmission complete interrupt) occurs, regardless of whether the result is ACK or NACK.

In the RL78 family, the local address (7 bits) is represented by the upper 7 bits in the SVA0 register. The lowermost bit in the SVA0 register is fixed at 0.

(6) Slave-to-master communication 2 (address – data – data)

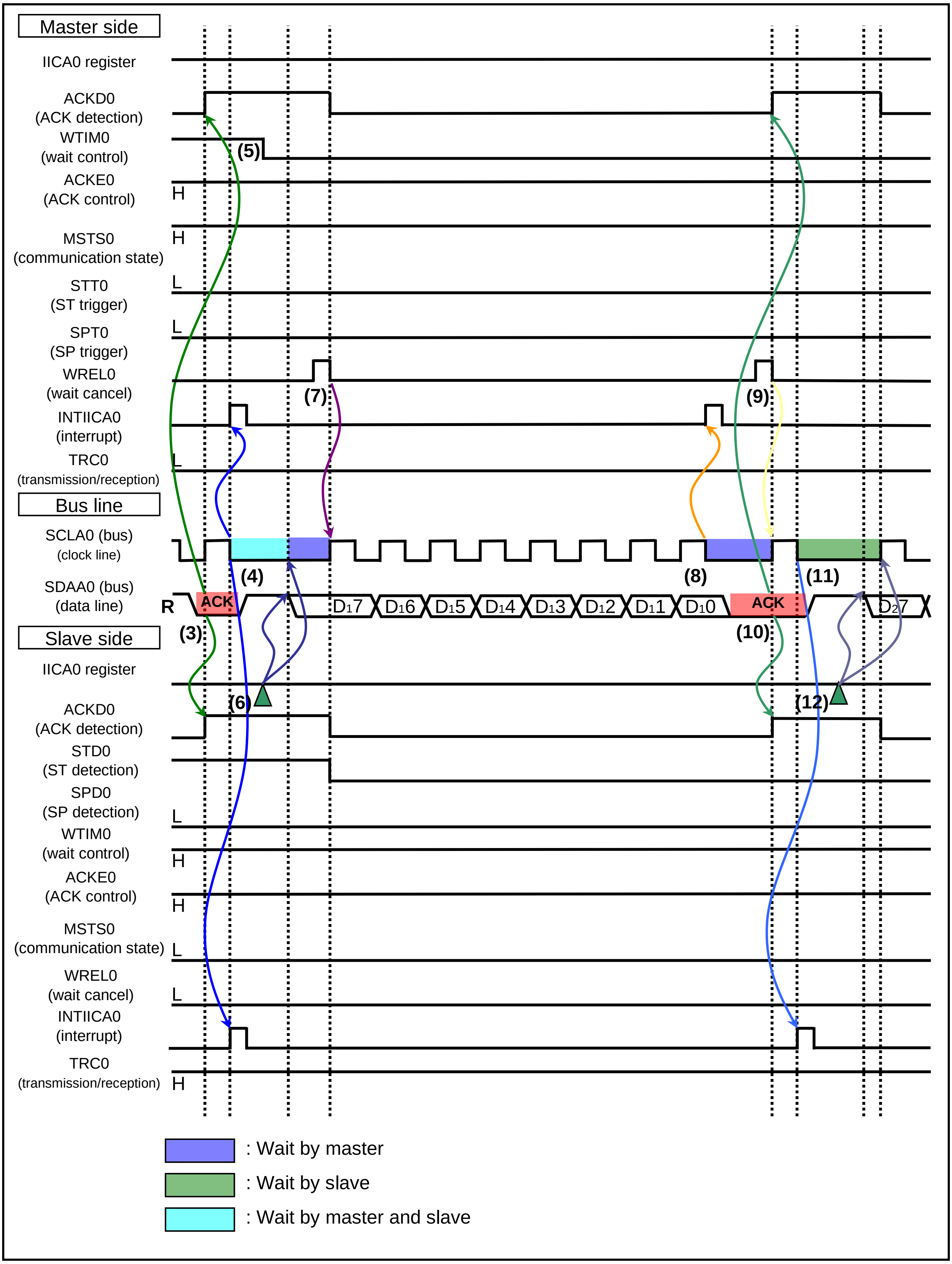
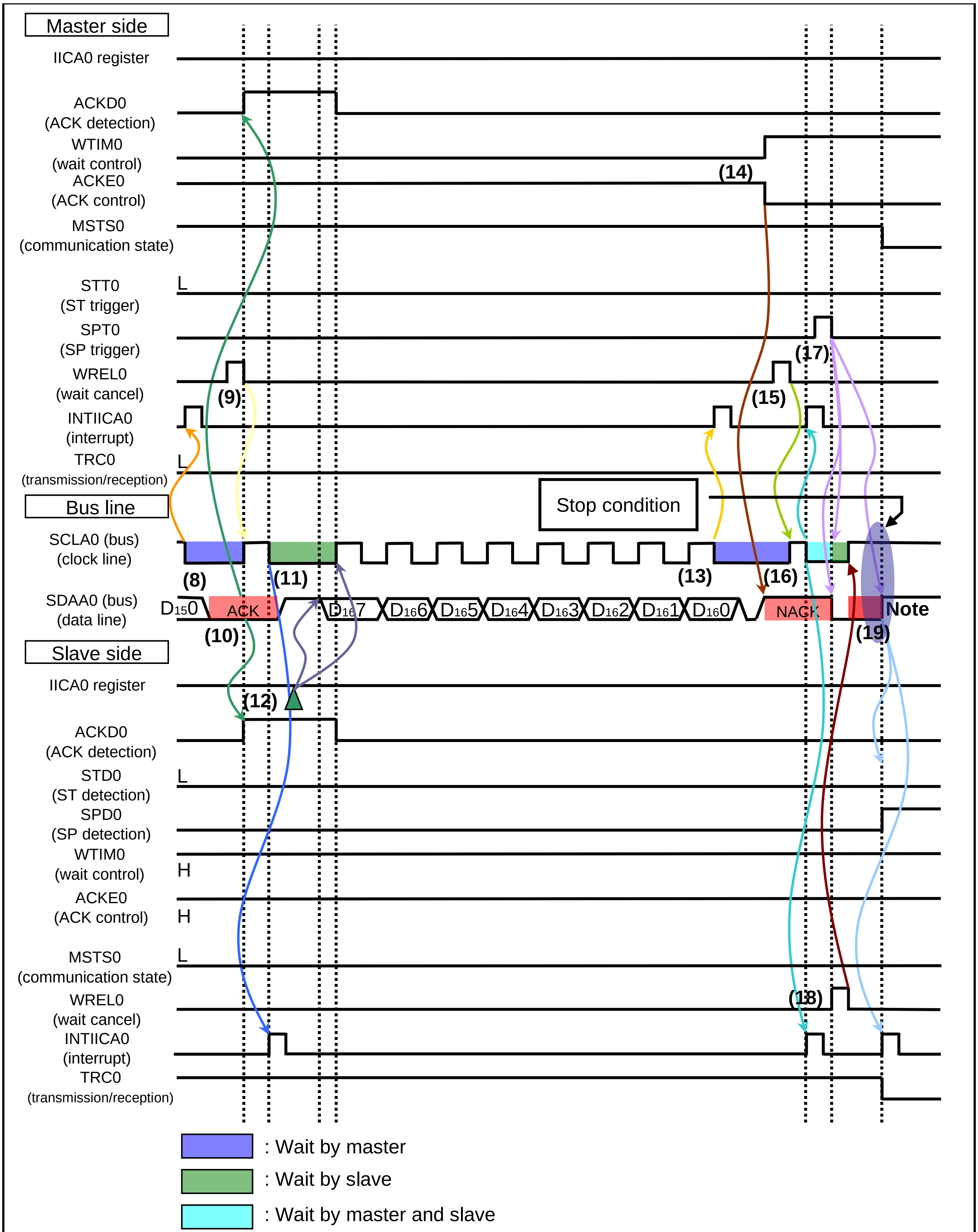


Figure 1.7 IIC Communication Timing Chart (Slave-to-Master Communication Example) (2/3)

(3) If the received address and slave address match, the slave hardware sends ACK to the master. When the ninth clock signal rises, the master detects ACK (ACKD0 = 1).
(4) When the ninth clock signal falls, an address transmission end interrupt (INTIICA0) occurs on the master side. If the addresses match, an address match interrupt (INTIICA0) occurs on the slave side. Both the master and the slave which has the matching address generate a wait (SCLA0 line: Low).
(5) The master selects an 8-clock wait (WTIM0 = 0) because it receives data.
(6) The slave writes transmit data to the IICA0 register and cancels the wait.
(7) When the master cancels the wait (WREL0 = 1), the slave starts transferring data to the master.
(8) When the eighth clock signal falls, the master generates a wait (SCLA0 line: Low) and a transfer end interrupt (INTIICA0) occurs on the master side. The master hardware sends ACK to the slave.
(9) The master reads the receive data and cancels the wait (WREL0 = 1).
(10) When the ninth clock signal rises, the slave detects ACK (ACKD0 = 1).
(11) When the ninth clock signal falls, the slave generates a wait (SCLA0 line: Low) and a transfer end interrupt (INTIICA0) occurs on the slave side.
(12) The slave writes transmit data to the IICA0 register and cancels the wait. Then, the slave starts transferring data to the master.

(7) Slave-to-master communication 3 (data – data – stop condition)



Note: The time from when SCLA0 rises after a stop condition is issued till when a stop condition is generated is 4.0 μs or more for the standard mode and 0.6 μs or more for the fast mode.

Figure 1.8 IIC Communication Timing Chart (Slave-to-Master Communication Example) (3/3)

(8) When the eighth clock signal falls, the master generates a wait (SCLA0 line: Low) and a transfer end interrupt (INTIICA0) occurs on the master side. The master hardware sends ACK to the slave.
(9) The master reads the receive data and cancels the wait (WREL0 = 1).
(10) When the ninth clock signal rises, the slave detects ACK (ACKD0 = 1).
(11) When the ninth clock signal falls, the slave generates a wait (SCLA0 line: Low) and a transfer end interrupt (INTIICA0) occurs on the slave side.
(12) The slave writes transmit data to the IICA0 register and cancels the wait. Then, the slave starts transferring data to the master.
(13) When the eighth clock signal falls, a transfer end interrupt (INTIICA0) occurs on the master side and the master generates a wait (SCLA0 line: Low). The master hardware sends ACK to the slave.
(14) The master sets a NACK response (ACKE0 = 0) to inform the slave that the master has sent the last data (at the end of communication). Then, the master changes the wait time to 9 clock periods (WTIM0 = 1).
(15) After the master cancels the wait (WREL0 = 1), the slave detects NACK (ACKD0 = 0) at the rising edge of the ninth clock signal.
(16) When the ninth clock signal falls, the master and slave generate a wait (SCLA0 line: Low) and a transfer end interrupt (INTIICA0) occurs on the master and slave sides.
(17) When the master issues a stop condition (SPT0 = 1), the SDAA0 line falls, thereby canceling the wait on the master side. Later, the master waits until the SCLA0 line rises.
(18) The slave cancels the wait (WREL0 = 1) to terminate communication. Then, the SCLA0 line rises.
(19) The master confirms that the SCLA0 line has risen. Upon the elapse of the stop condition setup time after this confirmation, the master makes the SDAA0 line rise and issues a stop condition. When the stop condition is generated, the slave detects the stop condition (SPD0 = 1) and a stop condition interrupt (INTIICA0) occurs on the master and slave sides.

1.2 Control of Serial RAM

1.2.1 Command Settings

In this application note, commands are used to specify slave operations. The command setting sequence is shown in Figure. 1.9, and the command setting timing chart appears in Figure. 1.10.

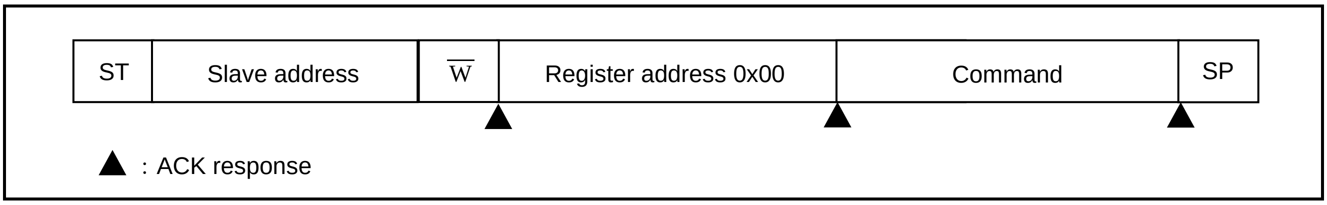


Figure. 1.9 Command Setting Sequence

In succession to a start condition (ST), the slave address 0xA0 (8 bits obtained by combining 0b1010000 and the transfer direction $\overline{W}$) is transmitted. The slave is selected by this slave address. Next, the master transmits the register address (0x00), indicating to the slave that a command will be transmitted next. After transmitting the command, a stop condition (SP) is transmitted, ending communication.

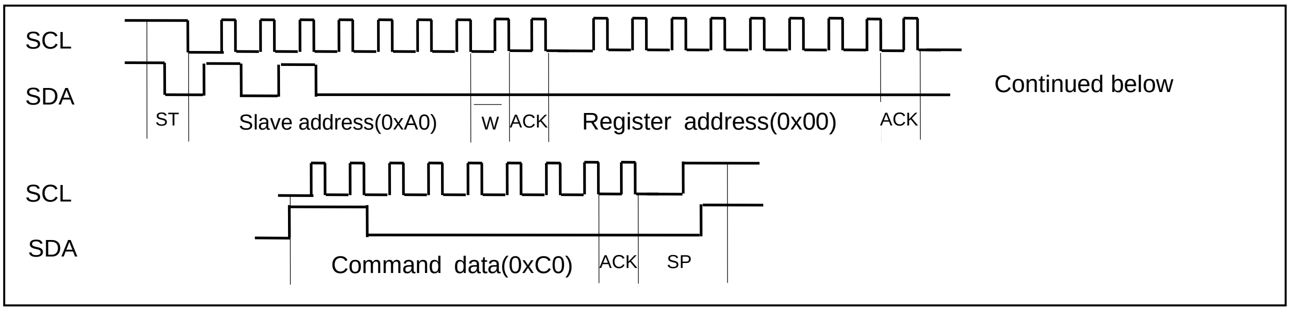


Figure. 1.10 Command Setting Timing Chart

A list of command functions appears in
Table 1.2.

Bit 7 (the MSB) of the data written to register address 0x00 (command register) indicates whether the command is valid or invalid. When bit 7 is 1, the slave judges that the command is valid, and when bit 7 is 0, the slave ignores the command as invalid. Bit 6 indicates whether memory functions are used or not. When memory functions are used, bit 6 is set to 1. Bits 5 to 3 are unused, and so are all set to 0. Bit 2 indicates whether writing is forbidden or not. While bit 2 is set to 1 (from when writing is prohibited until writing is again permitted), upon write data reception the slave sends a NACK response and disengages from communication. Bit 1 indicates whether memory is initialized or not. When bit 1 is set to 1, the slave memory is initialized to initialization data specified by bit 0. After the completion of initialization, the slave sets bit 1 of the command register to 0.

Table 1.2 Command functions

Bit	Meaning	Explanation
7	Command setting	1: command valid, 0: command invalid
6	Memory function selection	1: use memory functions, 0: do not use memory functions
5 to 3	Unused	Fixed at 0
2	Write selection	1: writing forbidden, 0: writing permitted
1	Initialization selection	1: initialize memory (serial RAM area), 0: do nothing
0	Initialization data selection	1: value of lowest 7 bits of register address[note], 0: 0x00

Note: At each address in the serial RAM area, the value of the lowest 7 bits of the address is written. For example, at register addresses 0x80, 0x81, 0x82, the respective values 0x00, 0x01, 0x02 are written.

1.2.2 Continuous Data Writing

Regarding cases in which a register address is specified and data is written continuously to the serial RAM of a slave, the sequence is shown in Figure. 1.11, and the timing chart appears in Figure. 1.12.

The master transmits a start condition (ST), followed by the slave address 0xA0 (the 8 bits that are 0b1010000 and the transfer direction $\overline{W}$). In succession to the slave address, the master transmits a register address specifying an internal address of serial RAM. Thereafter, write data is transmitted in order.

After the last data has been transmitted, a stop condition (SP) is generated, and communication is completed.

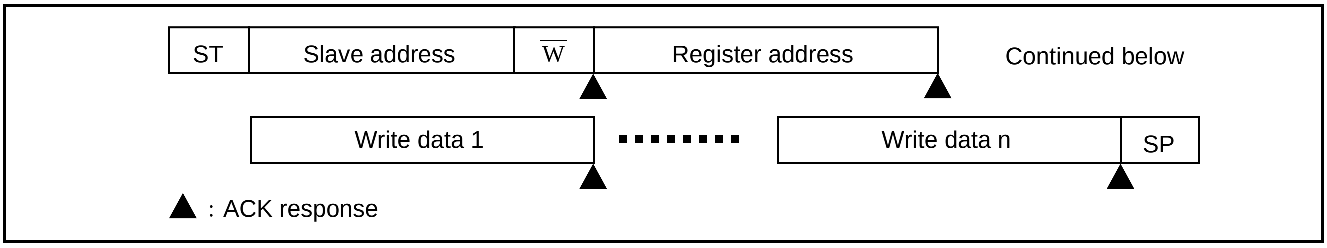


Figure. 1.11 Sequence for Continuous Data Writing by Register Address Specification

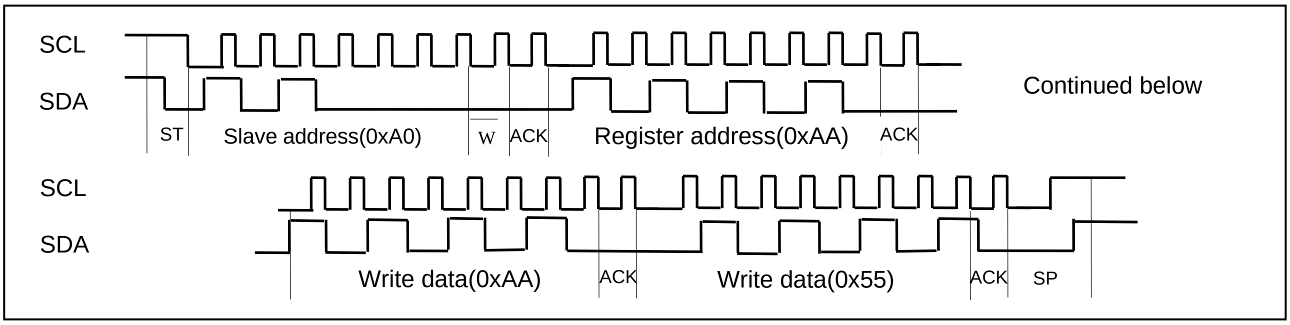


Figure. 1.12 Timing Chart for Continuous Data Writing by Register Address Specification

The timing chart for data writing when writing is forbidden is shown in Figure. 1.13. The slave returns ACK in response to the slave address and a register address, but returns NACK in response to write data, and disengages from communication. The master confirms the NACK response and generates a stop condition, ending communication.

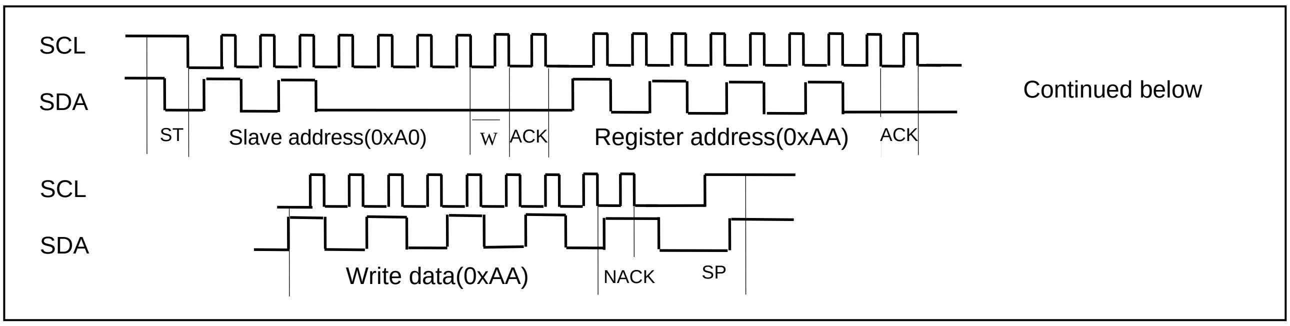


Figure. 1.13 Timing Chart for Data Writing When Writing is Forbidden

1.2.3 Continuous Data Reading

Regarding cases in which a register address is specified and data is read continuously from a slave, the sequence is shown in Figure. 1.14, and the timing chart appears in Figure. 1.15.

The master transmits a start condition (ST), followed by the slave address 0xA0 (the 8 bits that are 0b1010000 and the transfer direction $\overline{\mathrm{W}}$). In succession to the slave address, the master transmits a register address specifying an internal address of serial RAM. Then, after a restart condition (ST), the slave address 0xA1 (the 8 bits that are 0b1010000 and the transfer direction R) is transmitted. Thereafter data is transmitted in order from the specified register address (sequential read). When the master responds to received data with NACK, the slave stops transmission. Finally, a stop condition (SP) is generated to complete communication.

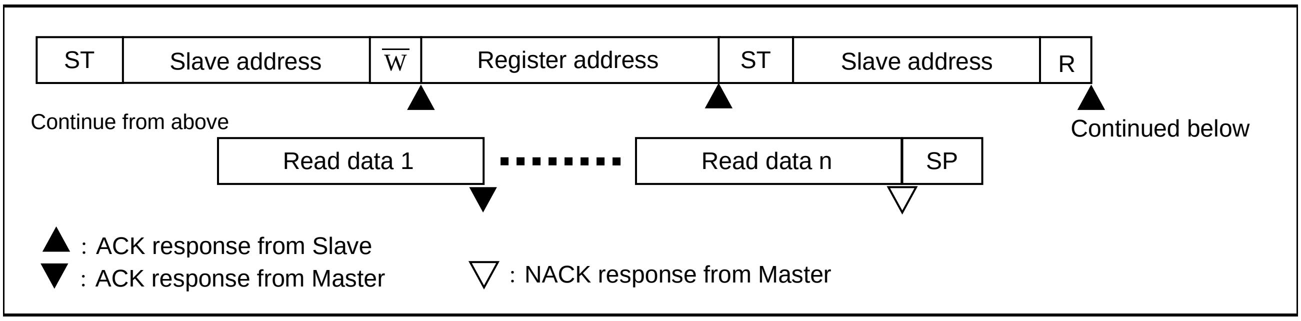


Figure. 1.14 Sequence for Continuous Data Reading by Register Address Specification

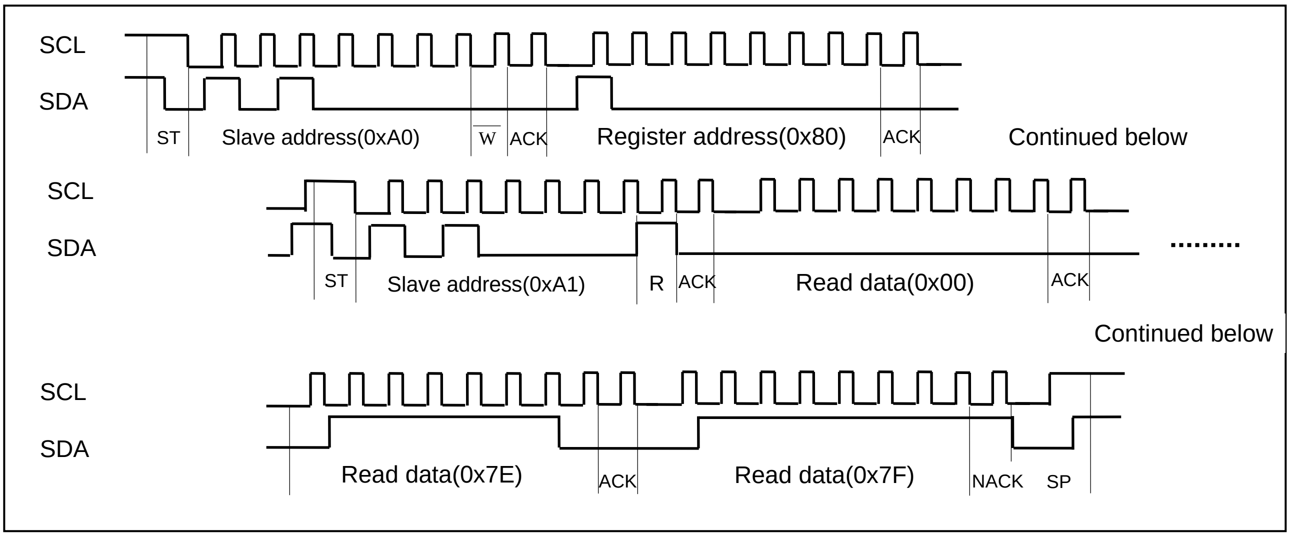


Figure. 1.15 Timing Chart for Continuous Data Reading by Register Address Specification

2. Operation Check Conditions

The sample code contained in this application note has been checked under the conditions listed in the table below.

Table 2.1 Operation Check Conditions

Item	Description
Microcontroller used	RL78/G15 (R5F12608ASP)
Board used	RL78/G15-20p Fast Prototyping Board (RTK5RLG150C00000BJ)
Operating frequency	High-speed on-chip oscillator (HOCO) clock: 16 MHz
Operating voltage	5.0 V (can be operated at 2.4 V to 5.5 V)
Integrated development environment (CS+)	Renesas Electronics CS + for CC V8.10.00
C compiler (CS+)	Renesas Electronics CC-RL V1.12.01
Integrated development environment (e^2 studio)	Renesas Electronics e^2 studio V2023-07 (23.7.0)
C compiler (e^2 studio)	Renesas Electronics CC-RL V1.12.01
Integrated development environment (IAR)	IAR systems IAR Embedded Workbench for Renesas RL78 V5.10.1
C compiler (IAR)	IAR systems IAR Embedded Workbench for Renesas RL78 V5.10.1
Smart configurator (SC)	V1.6.0 from Renesas Electronics Corp.
Board support package (BSP)	V1.6.0 from Renesas Electronics Corp.

3. Related Application Note

The application notes that are related to this application note are listed below for reference.

- RL78/G15 Serial Interface IICA (for Slave Transmission/Reception) (R01AN7021E) Application Note

4. Description of the Hardware

4.1 Hardware Configuration Example

Figure 4.1 shows an example of hardware configuration that is used for this application note.

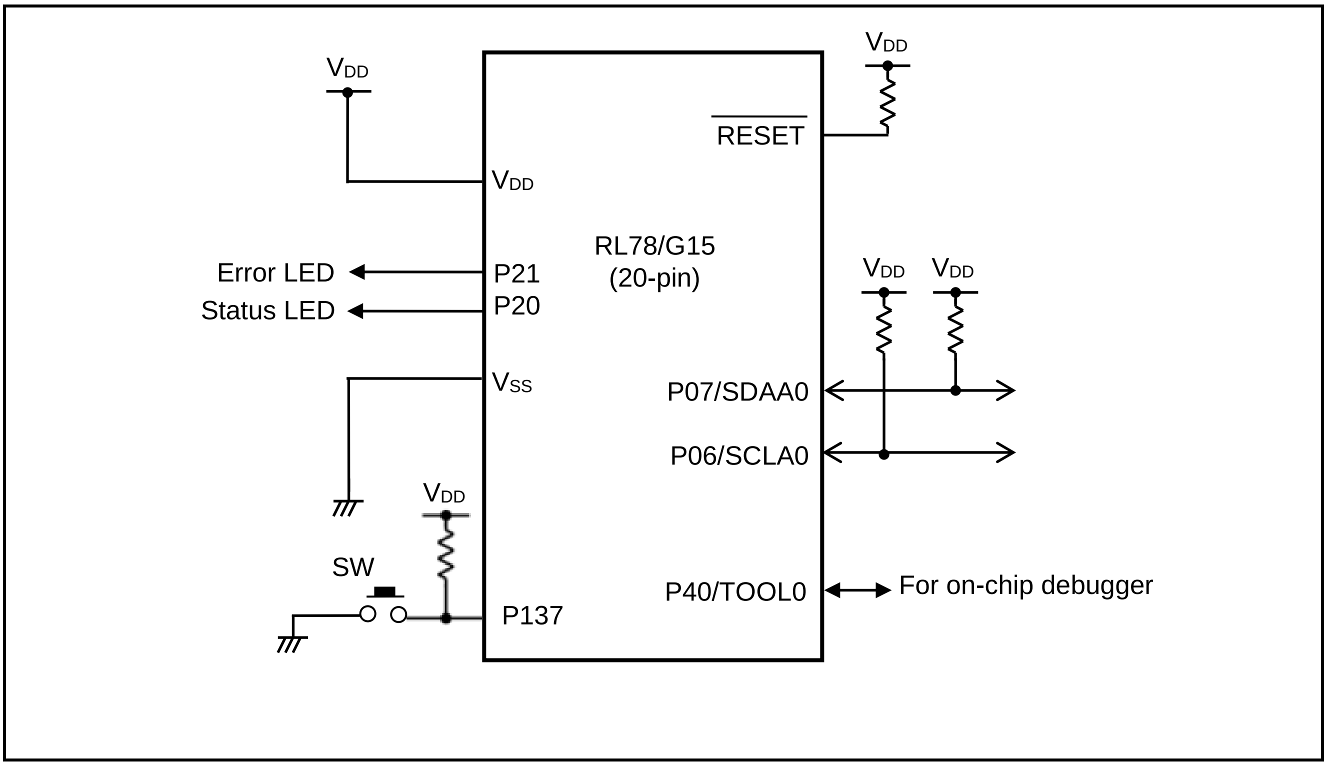


Figure 4.1 Hardware Configuration

Cautions: 1. The purpose of this circuit is only to provide the connection outline and the circuit is simplified accordingly. When designing and implementing an actual circuit, provide proper pin treatment and make sure that the hardware's electrical specifications are met (connect the input-only ports separately to V_{DD} or V_{SS} via a resistor).

2. VDD must not be lower than the reset release voltage (VSPOR) that is specified for the SPOR.

4.2 List of Pins to be Used

Table 4.1 lists the pins to be used and their functions.

Table 4.1 Pins to be Used and their Functions

Pin Name	I/O	Description
P06/SCLA0	Input/Output	Serial clock input/output pin
P07/SDAA0	Input/Output	Serial data transmission/reception pin
P20	Output	Signal to drive Status LED
P21	Output	Signal to drive Error LED
P137	Input	Switch input signal for designating operation start

5. Description of the Software

5.1 Operation Outline

In this application note, a serial interface IICA is used to perform IICA master transmission and reception (address transmission, data transmission and reception) operations.

(1) Initialize serial interface IICA.

<Conditions for setting>

- Select the fast mode as the operation mode.
- Set the transfer clock frequency to 400 kHz.
- Set the local address to 0x10.
- Turn the digital filter on.
- Enable acknowledgements.
- Generate an interrupt in response to the ninth clock signal.
- Disable stop condition interrupts.
- Use the P06/SCLA0 pin for transfer clock output and the P07/SDAA0 pin for data transmission/reception.

(2) Communication buffers (reception buffers 8×16 = 128 bytes, transmission buffer 65 bytes) are prepared.

(3) When the switch connected to P137 is turned on, the command 0×C0 is transmitted from master to slave. Thereafter, the command 0×C3 is transmitted, and the slave serial RAM area is initialized to "the value of the lower 7 bits of the register address".

(4) Continuous data reading is performed 8 times in 16-byte units (for a total of 128 bytes) from the slave register address 0×80.
When there is a NACK response from the slave, the LED connected to P21 lights.

(5) A check is performed to determine whether data read from the slave (128 bytes) matches an expected value prepared in advance. If the data differs from the expected value, the LED connected to P21 is made to flash with a period of 500 ms.

(6) Write data is modified and transmitted to the slave. If there is a NACK response from the slave, the LED connected to P21 is lit.

Transmission data:
When the received data in (4) above is 0×00 to 0×7F (monotonically increasing), 0×FF to 0×80 (monotonically decreasing)
When the received data in (4) above is 0×FF to 0×80 (monotonically decreasing), 0×0 to 0×7F (monotonically increasing)

(7) The status display LEDs connected to P20 are reversed, and after waiting 10 ms, the steps from (4) are repeated.

Caution: This sample code corresponds to the R01AN7021E Application Note for the RL78/G15 serial interface IICA (slave transmission/reception).

5.2 List of Option Byte Settings

Table 5.1 summarizes the settings of the option bytes.

Table 5.1 Option Byte Settings

Address	Value	Description
000C0H	11101110B	Disables the watchdog timer. (Stops counting after the release from the reset state.)
000C1H	11111011B	LVD0 detection voltage: reset mode At rising edge TYP. 2.57 V (2.44 V to 2.68 V) At falling edge TYP. 2.52 V (2.40 V to 2.62 V) P125 is used as the reset pin.
000C2H	11111001B	High-speed on-chip oscillator clock (fIH): 16 MHz
000C3H	10000101B	Enables the on-chip debugger.

5.3 List of Constants

Table 5.2 lists the constants that are used in this sample program.

Table 5.2 Constants for the Sample Program

Constant	Setting	Description
MAX_DATA	64	IIC transmission data length
SLAVE_ADDR	0xA0	Slave address
RAM_TOP	0x80	Start address of slave RAM area
LED_ON	0	Data to light the LED
RETRY	10	Upper limit of the number of retries
WAITTIME	1000	Loop waiting for start condition detection
STS_MASK	0x03	Mask data of IICA 0 status
DUMMY_DATA	0xFF	Dummy data for reception activation
BUS_BUSY	0x8C	IIC bus · busy
ON_COMMU	0x01	Data indicating that communication is in ogress
IIC_SUCCESS	0x00	Communication terminated normally
IIC_USING	0x01	connecting
NO_ACK	0x84	NACK response to transmission data
NO_SLAVE	0x88	NACK response to slave address
RAM_COMMAND[4]	-	List of commands to slave
RAM_DATA[4][64]	-	A table of data to be written to the slave

5.4 List of Variables

Table 5.3 lists the global variables that are used in this sample program.

Table 5.3 Global Variables for the Sample Program

Type	Variable Name	Contents	Function Used
uint8_t	g_read_ram[8][16]	IICA 0 receive buffer	main()
uint8_t	g_write_data	IICA0 transmit buffer	r_IICA0_put_data(), r_IICA0_get_data(), main()
uint8_t	g_1ms_status	1ms count flag	wait_ms(), r_Config_IT_interrupt()
uint8_t	g_1ms_count	1ms counter	wait_ms(), r_Config_IT_interrupt()
uint8_t	g_iica0_status	Status of IICA0	R_IICA0_check_comstate() , R_IIC_wait_comend(), R_IICA0_bus_check(), r_Config_IICA0_interrupt()
uint8_t *	gp_iica0_rx_address	Receive-data buffer address	R_IIC_Master_Receive(), r_Config_IICA0_interrupt()
uint16_t	g_iica0_rx_len	Number of bytes to be received	R_IIC_Master_Receive(), r_Config_IICA0_interrupt()
uint16_t	g_iica0_rx_cnt	Number of data bytes already received	R_IIC_Master_Receive(), r_Config_IICA0_interrupt()
uint8_t *	gp_iica0_tx_address	Transmit-data buffer address	R_IIC_Master_Send(), r_Config_IICA0_interrupt()
uint16_t	g_iica0_tx_cnt	Number of data bytes already sent	R_IIC_Master_Send(), r_Config_IICA0_interrupt()
uint16_t	g_us_rest_time	Wait time in µs units	set_delay_us(), r_tau0_channel2_interrupt()
uint16_t	g_ms_timer1S	1ms counter	r_tau0_channel3_interrupt()

5.5 List of Functions

Table 5.4 summarizes the functions that are used in this sample program.

Table 5.4 Functions

Function Name	Outline
wait_ms	Time wait processing in ms units
r_Config_IT_interrupt	12-bit interval timer interrupt processing
r_wait_SW	Waiting for press of SW
r_IICA0_put_data	Processing of sending data to slave
r_IICA0_get_data	Processing to receive data from slave
R_IICA0_Master_Send	Master transmission start request processing
R_IICA0_Master_Receive	Master reception start request processing
R_IICA0_wait_comend	Communication completion wait processing
R_IICA0_check_comstate	IICA 0 communication state confirmation processing
R_IICA0_StopCondition	Stop condition generation
R_IICA0_bus_check	IIC bus status check and start condition issue processing
r_Config_IICA0_interrupt	IICA0 interrupt processing
delay_us	Time wait processing in μs units
set_delay_us	Wait time setting process in μs units
r_Config_TAU0_2_interrupt	Timer Array Unit Channel 2 Interrupt Processing

5.6 Function Specifications

This section describes the specifications for the functions that are used in this sample program

[Function Name] wait_ms		
Synopsis	Time wait processing in ms units	
Header	Config_IT.h	
Declaration	void wait_ms(uint8_t)	
Explanation	Wait for the time specified in the argument (ms units)	
Arguments	wait_time	Latency
Return value	None	
Remarks	None	

[Function Name] r_Config_IT_interrupt		
Synopsis	12-bit interval timer interrupt processing	
Header	-	
Declaration	static void r_Config_IT_interrupt(void)	
Explanation	Perform 1 ms interval interrupt processing using a 12-bit interval timer	
Arguments	None	-
Return value	None	
Remarks	None	

[Function Name] r_wait_SW		
Synopsis	Waiting for press of SW	
Header	-	
Declaration	void r_wait_SW(void)	
Explanation	Wait for the switch connected to P137 to be turned on	
Arguments	None	-
Return value	None	
Remarks	None	

[Function Name] r_IICA0_put_data		
Synopsis	Processing of sending data to slave	
Header	Config_IICA0.h	
Declaration	uint8_t r_IICA0_put_data(uint8_t s_addr, uint8_t r_addr, uint8_t __far * const buffer, uint8_t tx_num)	
Explanation	Transmit a specified number of bytes of data from the transmission buffer to a specified address of a specified slave	
Arguments	s_addr	Slave address
	r_addr	Register address
	buffer	Transmit-data buffer address
	txnum	Transmit-data byte count
Return value	Communication status. If 0x00, normal termination, otherwise error.	
Remarks	None	

[Function Name] r_IICA0_get_data		
Synopsis	Processing to receive data from slave	
Header	Config_IICA0.h	
Declaration	uint8_t r_IICA0_get_data(uint8_t s_addr, uint8_t r_addr, uint8_t * const buffer, uint8_t rx_num);	
Explanation	Receive a specified number of bytes of data to the reception buffer from a specified address of a specified slave	
Arguments	s_addr	Slave address
	r_addr	Register address
	buffer	Receive-data buffer address
	rxnum	Receive-data byte count
Return value	Communication status. If 0x00, normal termination, otherwise error.	
Remarks	None	

[Function Name] R_IICA0_Master_Send		
Synopsis	Master transmission start request processing	
Header	Config_IICA0.h	
Declaration	uint8_t R_IIC_Master_Send(uint8_t adr, uint8_t * const tx_buf, uint16_t tx_num)	
Explanation	Set master transmission.	
Arguments	adr	Slave address
	tx_buf	Transmit-data buffer address
	tx_num	Transmit-data byte count
Return value	Communication status. 0x00 successfully acquires the bus, otherwise it is an error.	
Remarks	None	

[Function Name] R_IICA0_Master_Receive		
Synopsis	Master reception start request processing	
Header	Config_IICA0.h	
Declaration	uint8_t R_IIC_Master_Receive(uint8_t adr, uint8_t * const rx_buf, uint16_t rx_num)	
Explanation	Set master reception.	
Arguments	adr	Slave address
	rx_buf	Receive-data buffer address
	rx_num	Number of bytes to be received
Return value	Communication status. 0x00 successfully acquires the bus, otherwise it is an error.	
Remarks	None	

[Function Name] R_IICA0_wait_comend		
Synopsis	Communication completion wait processing	
Header	Config_IICA0.h	
Declaration	uint8_t R_IICA0_wait_comend (void)	
Explanation	Wait for the end of IIC communication. Upon error detection, issue a stop condition	
Arguments	None	-
Return value	Communication status. If 0x00, normal termination, otherwise error.	
Remarks	None	

[Function Name] R_IICA0_check_comstate	
Synopsis	IICA 0 communication state confirmation processing
Header	Config_IICA0.h
Declaration	uint8_t R_IICA0_check_comstate(void)
Explanation	Return to the status of the IIC communication state (g_iica0_status)
Arguments	None
Return value	Communication status. Value of g_iica0_status
Remarks	None

[Function Name] R_IICA0_StopCondition	
Synopsis	Stop condition generation
Header	Config_IICA0.h
Declaration	uint8_t R_IIC_StopCondition(void)
Explanation	Release IIC bus.
Arguments	None
Return value	Communication status. If 0x00, normal termination, otherwise error.
Remarks	None

[Function Name] R_IICA0_bus_check	
Synopsis	IIC bus status check and start condition issue processing
Header	Config_IICA0.h
Declaration	uint8_t R_IICA0_bus_check(void)
Explanation	Check whether use of the IIC bus is possible; if possible, issue a start condition, and after the start condition has been issued, set g_iica0_status to communication (0×01). If not possible, return an error.
Arguments	None
Return value	Communication status. If 0x00, normal termination, otherwise error.
Remarks	None

[Function Name] r_Config_IICA0_interrupt	
Synopsis	IICA0 interrupt processing
Header	-
Declaration	static void __near r_Config_IICA0_interrupt(void);
Explanation	This function performs IICA0 interrupt processing.
Arguments	None
Return value	None
Remarks	In this sample code, the code generator uses initial settings only, and all interrupt processing is nearly created.

[Function Name] delay_us		
Synopsis	Time wait processing in μs units	
Header	Config_TAU0_2.h	
Declaration	void delay_us(uint16_t us)	
Explanation	Wait for the time specified in the argument (μs units).	
Arguments	us	Waiting time
Return value	None	
Remarks	None	

[Function Name] set_delay_us		
Synopsis	Wait time setting process in µs units	
Header	Config_TAU0_2.h	
Declaration	void set_delay_us(uint16_t us)	
Explanation	Set a wait time in µs units.	
Arguments	us	Waiting time
Return value	None	
Remarks	None	

[Function Name] r_Config_TAU0_2_interrupt	
Synopsis	Timer Array Unit Channel 2 Interrupt Processing
Header	-
Declaration	static void __near r_tau0_channel2_interrupt(void)
Explanation	Perform interrupt processing for timer array unit channel 2 (maximum 100 us interval).
Arguments	None
Return value	None
Remarks	None

5.7 Flowcharts

5.7.1 Main Processing

Figure 5.1 through Figure 5.3 show the flowchart for the main processing.

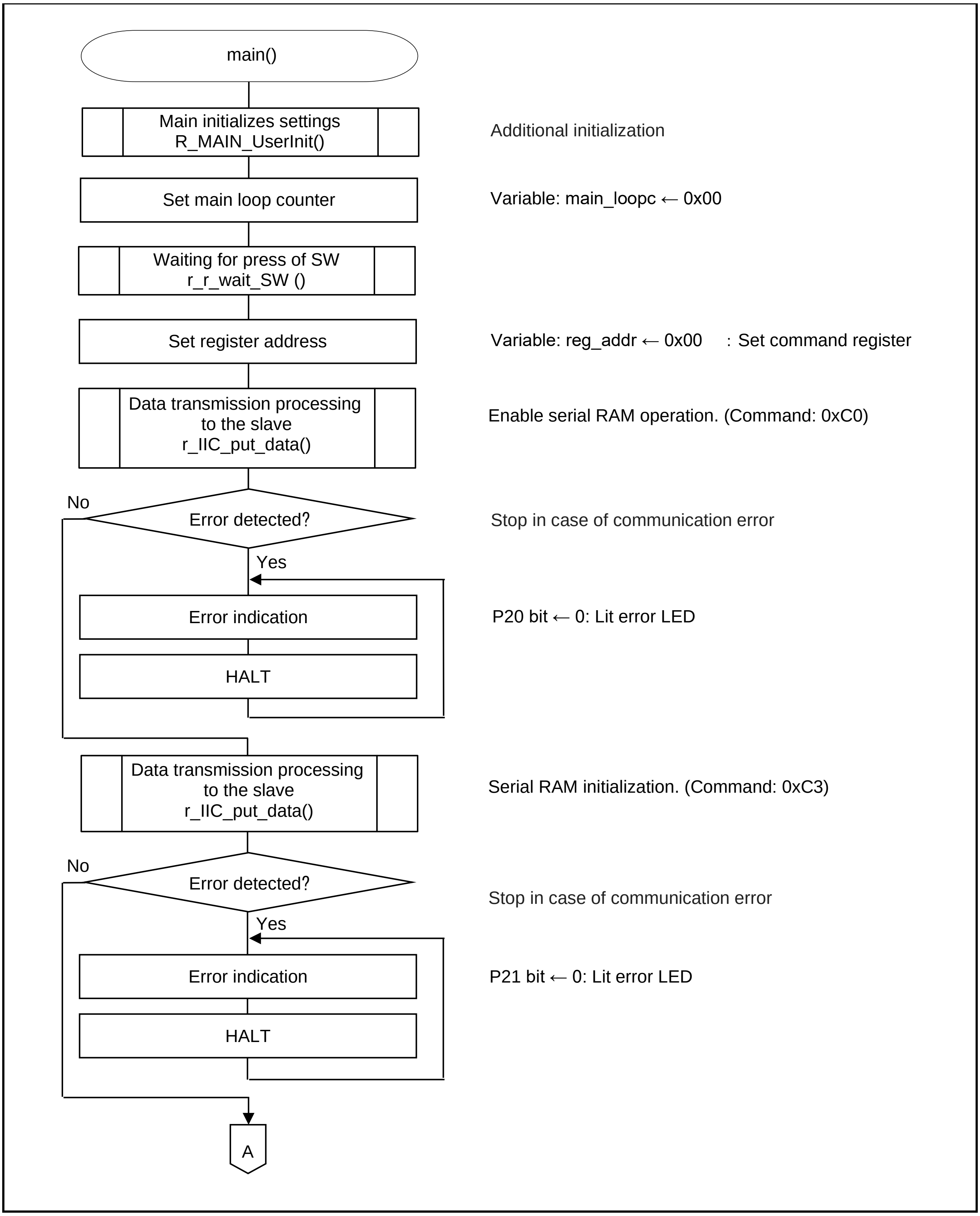


Figure 5.1 Main Processing (1/3)

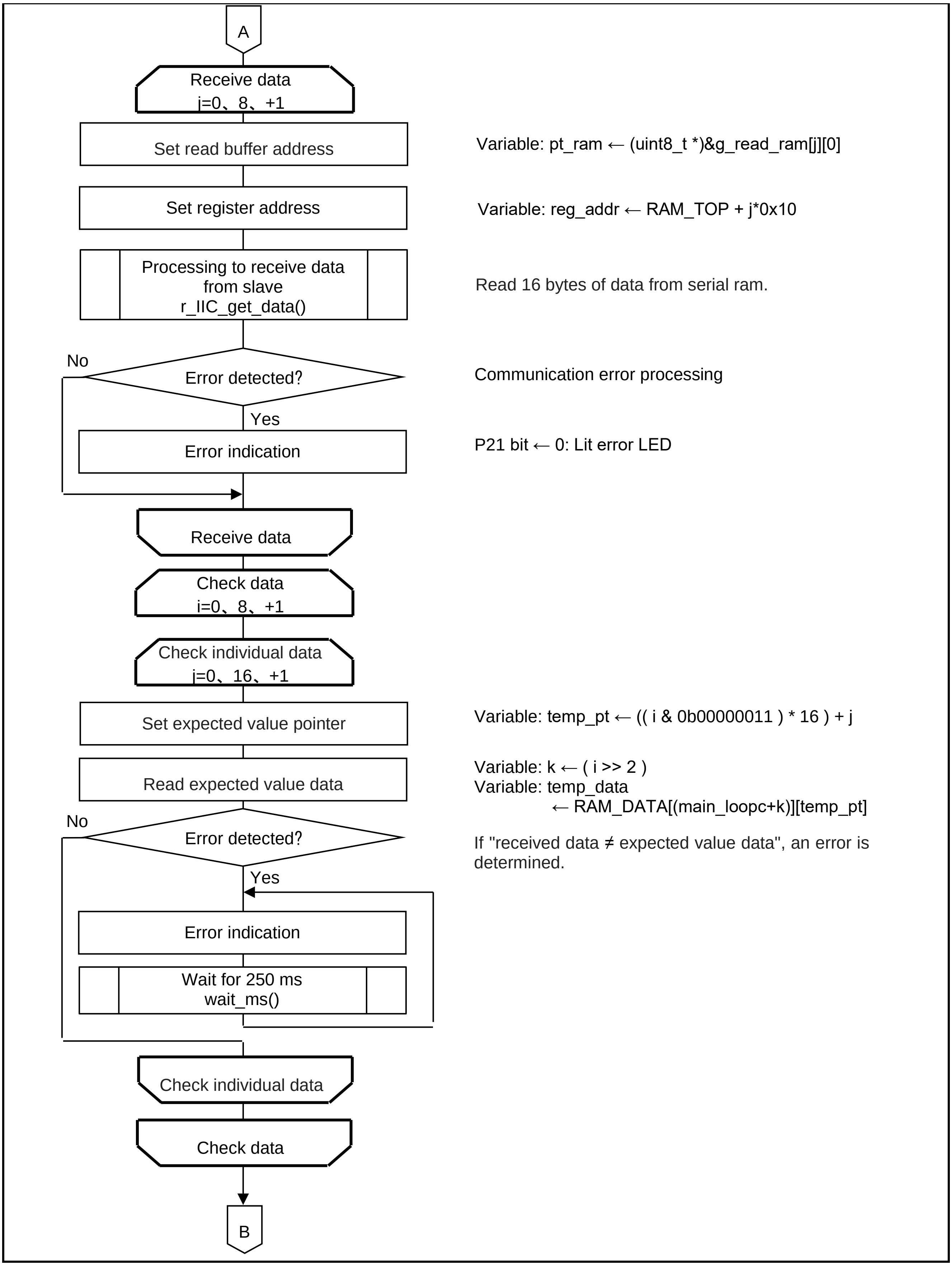


Figure 5.2 Main Processing (2/3)

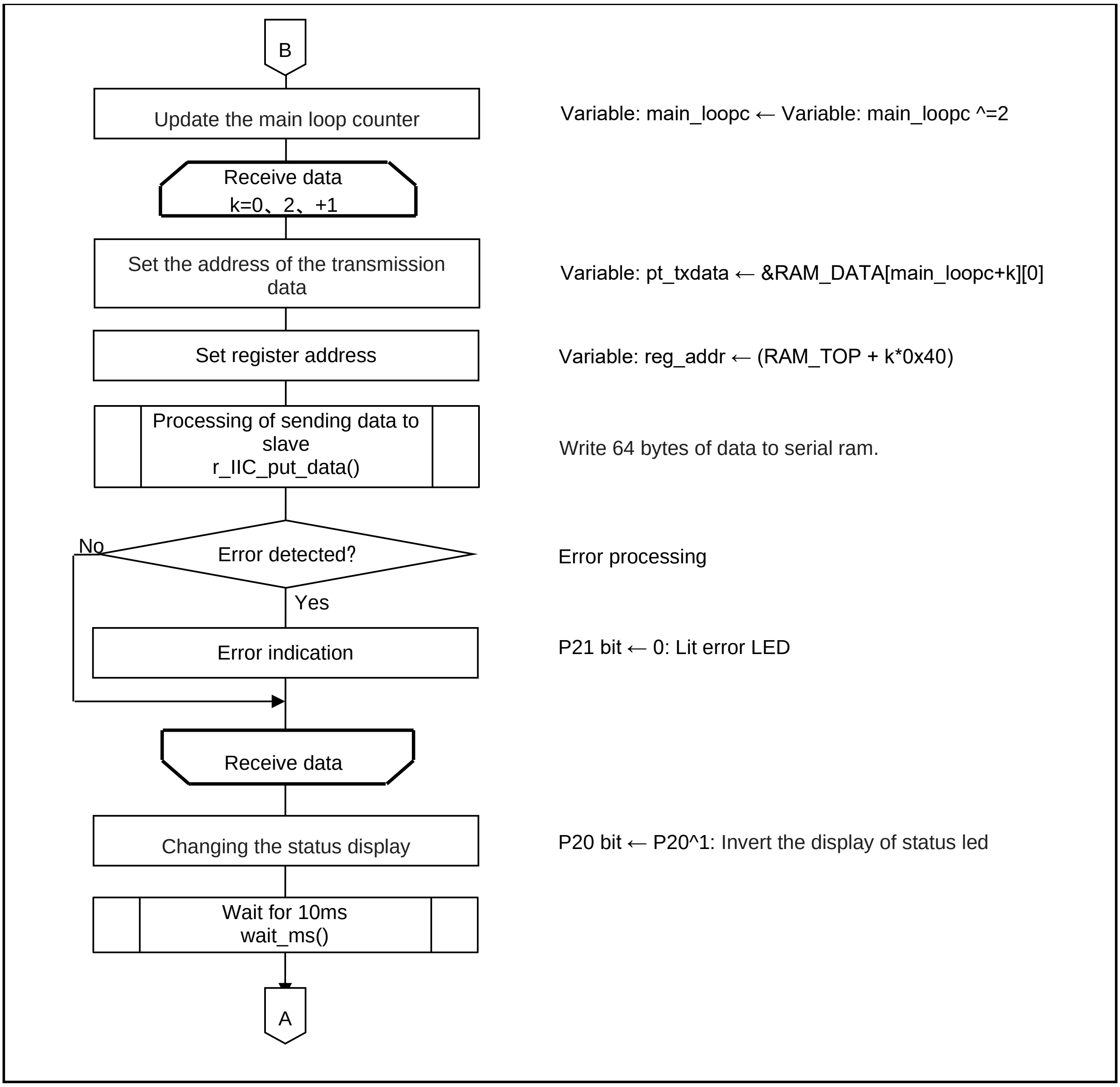


Figure 5.3 Main Processing (3/3)

5.7.2 Main Initial Setting

Figure 5.4 show the flowchart for the main initial setting.

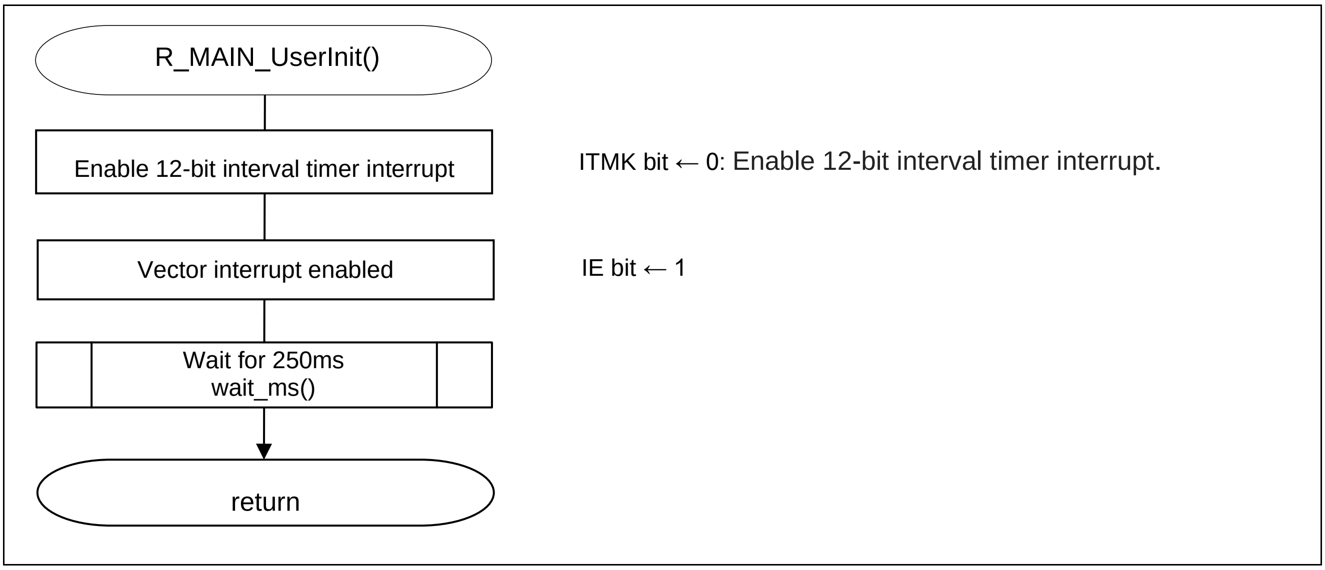


Figure 5.4 Main Initial Setting

5.7.3 Waiting for Press of SW

Figure 5.5 show the flowchart for the waiting for press of SW.

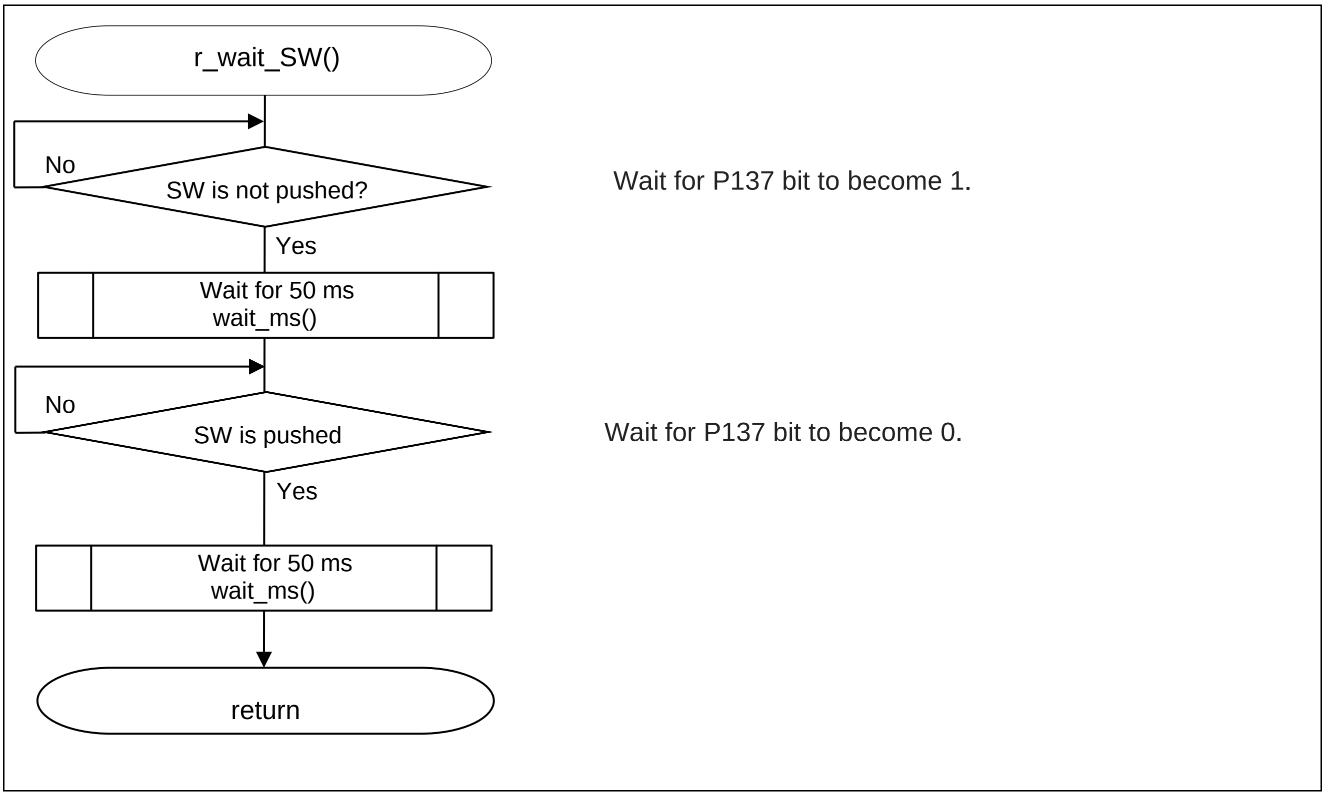


Figure 5.5 Waiting for Press of SW

5.7.4 Time Wait Processing in ms units

Figure 5.6 show the flowchart for the time wait processing in ms units.

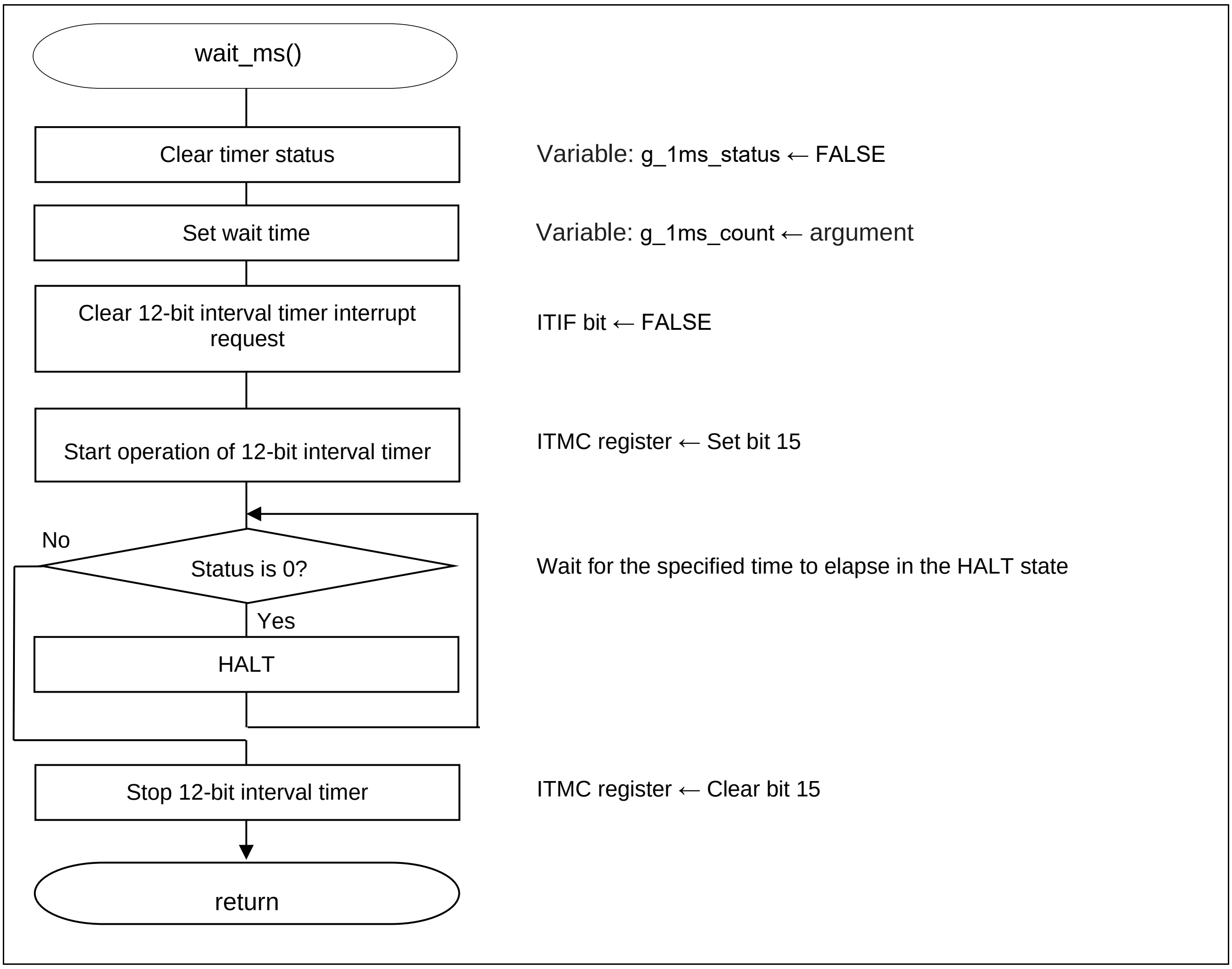


Figure 5.6 Time Wait Processing in ms units

5.7.5 12-bit interval timer interrupt processing

Figure 5.7 show the flowchart for the 12-bit interval timer interrupt processing.

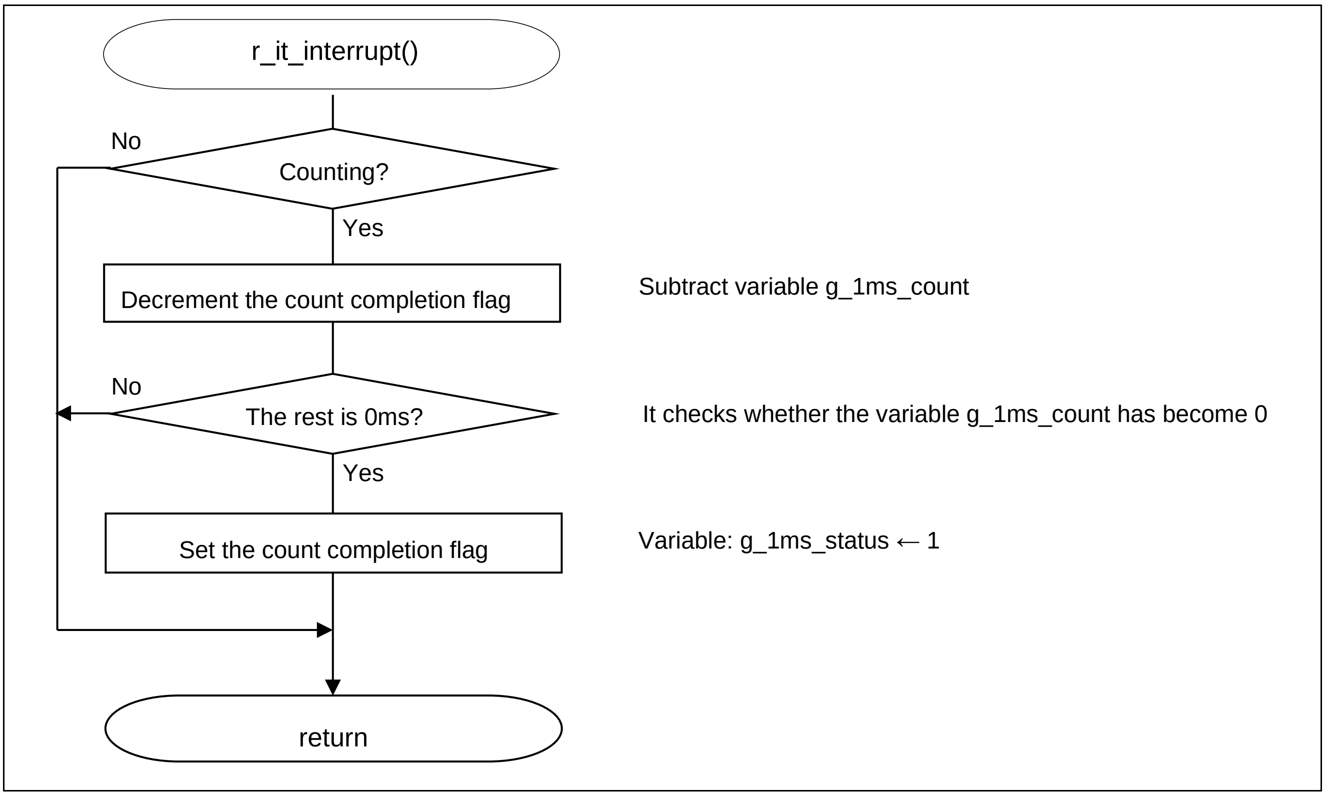


Figure 5.7 12-bit interval timer interrupt processing

5.7.6 Processing of sending data to slave

Figure 5.8 through Figure 5.9 show the flowchart for the processing of sending data to slave.

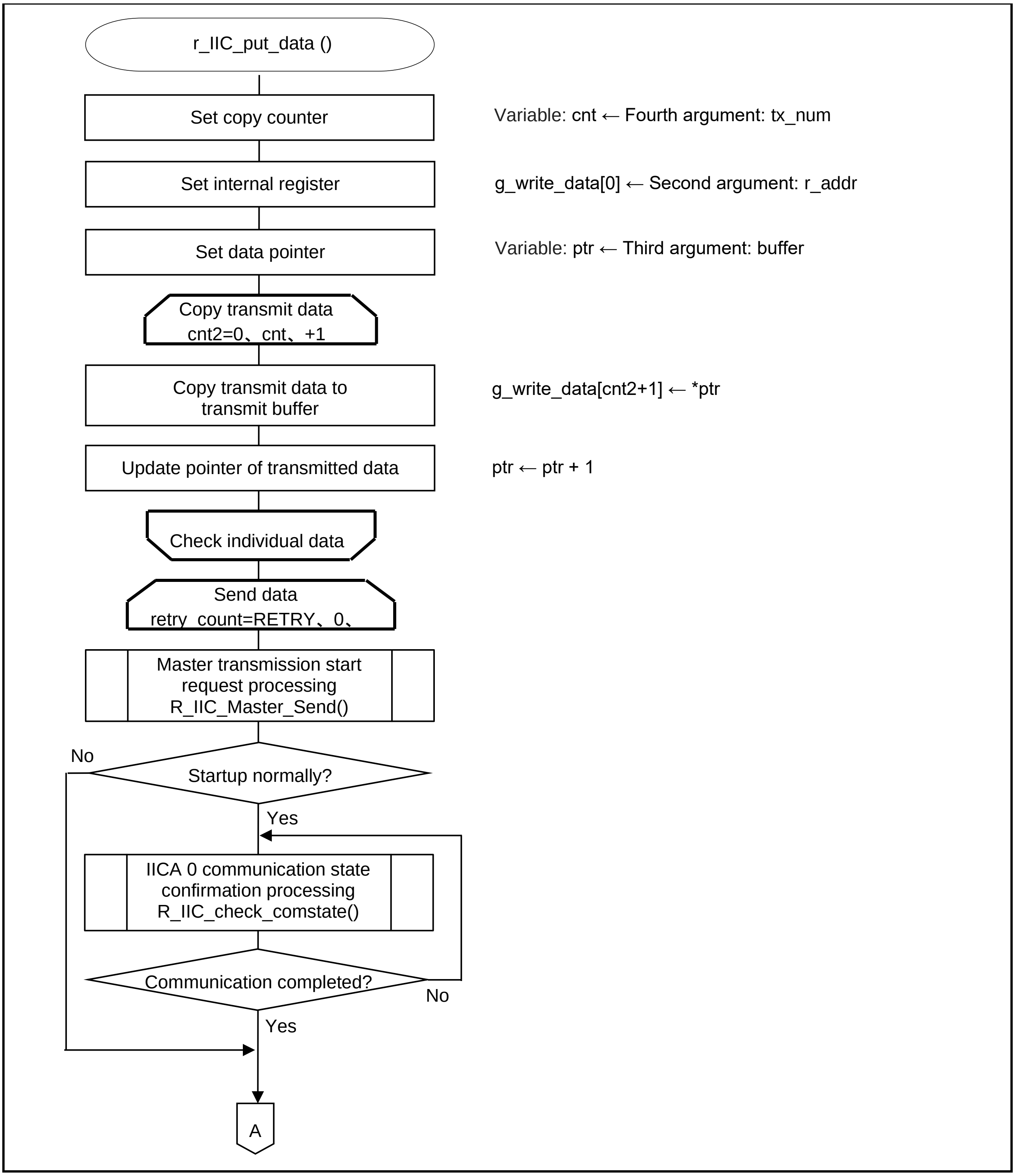


Figure 5.8 Processing of sending data to slave (1/2)

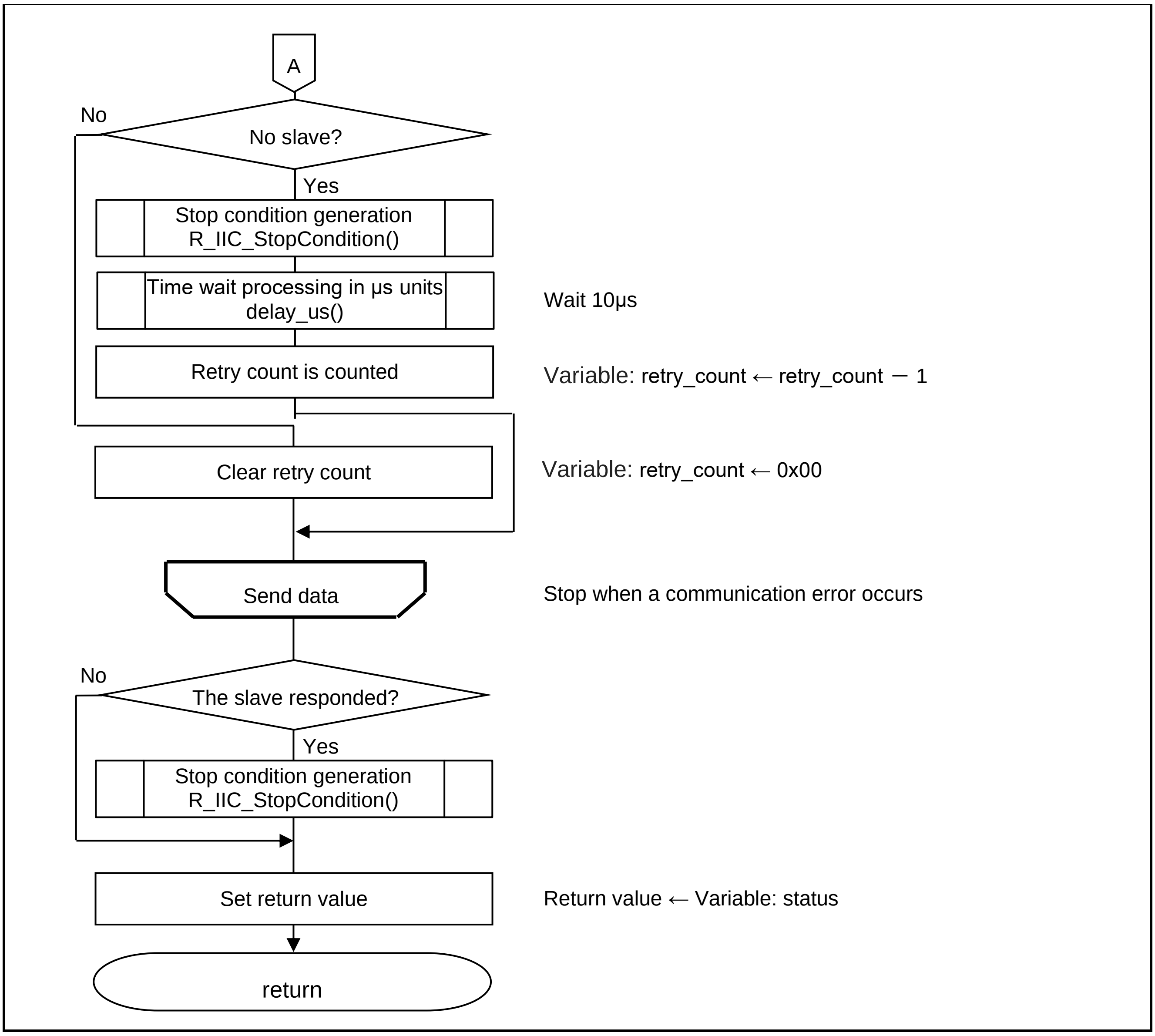


Figure 5.9 Processing of sending data to slave (2/2)

5.7.7 Processing to receive data from slave

Figure 5.10 through Figure 5.11 show the flowchart for the processing to receive data from slave.

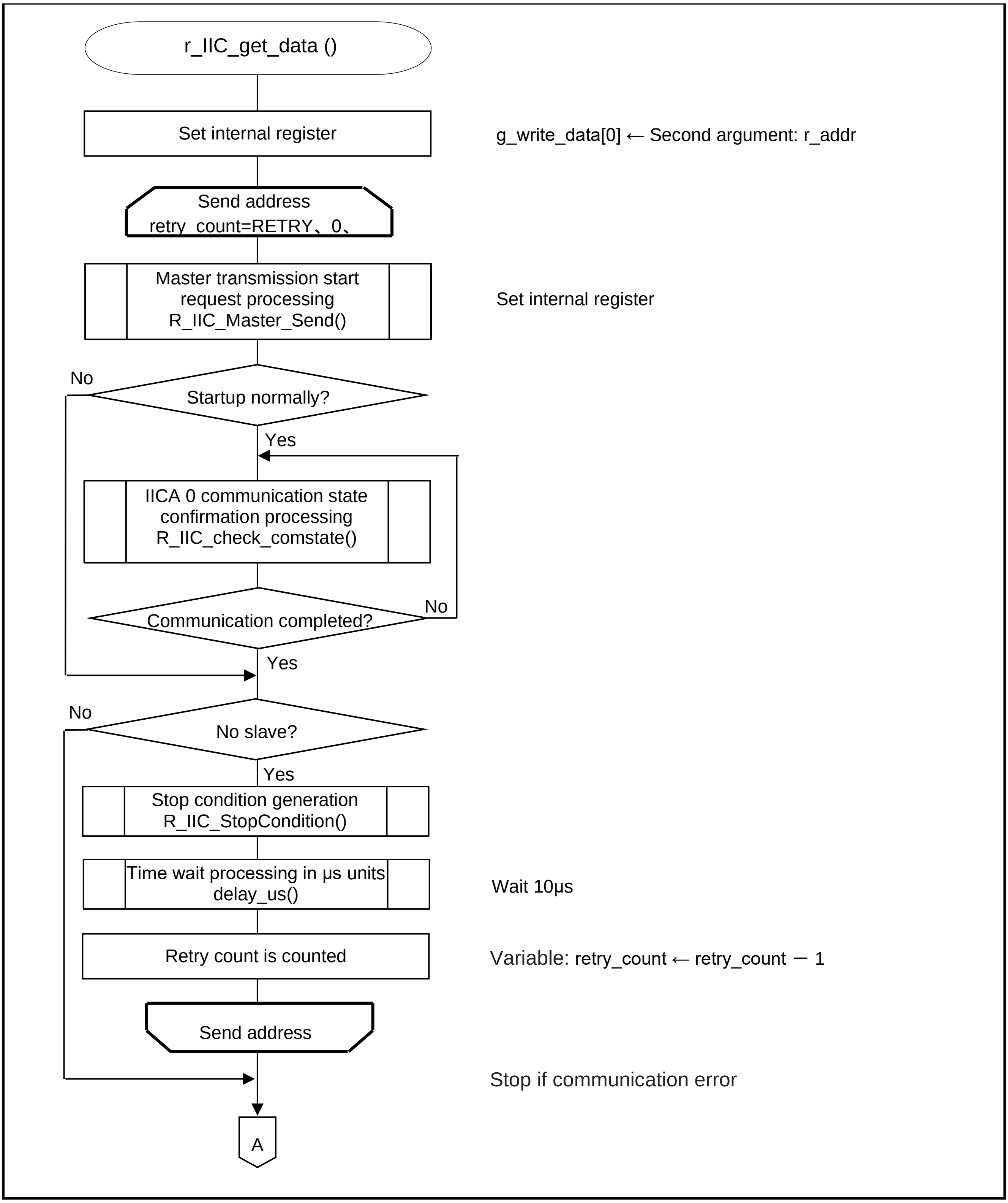


Figure 5.10 processing to receive data from slave (1/2)

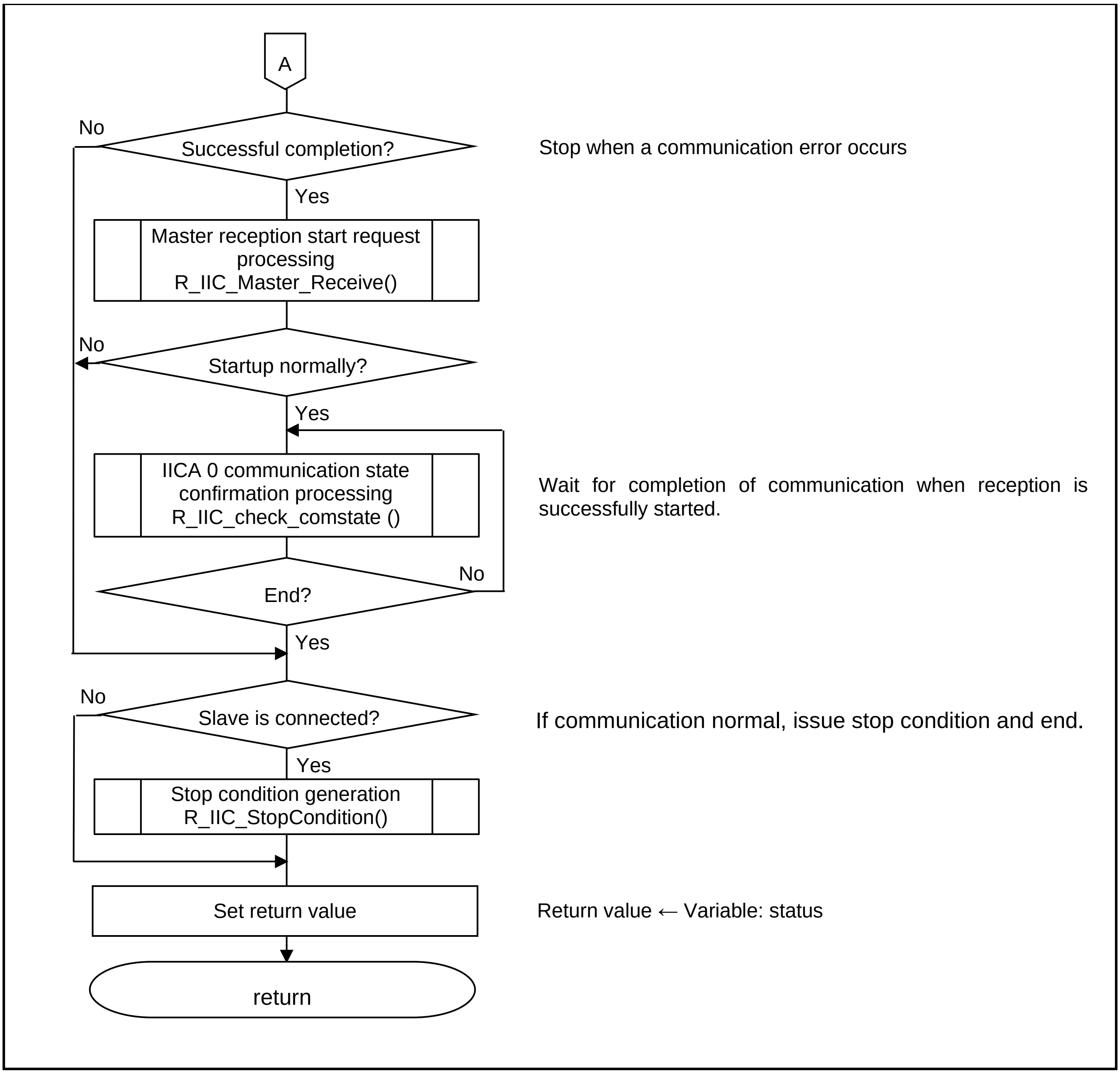


Figure 5.11 processing to receive data from slave (2/2)

5.7.8 Master Transmission Start Request Processing

Figure 5.12 shows the flowchart for starting master transmission.

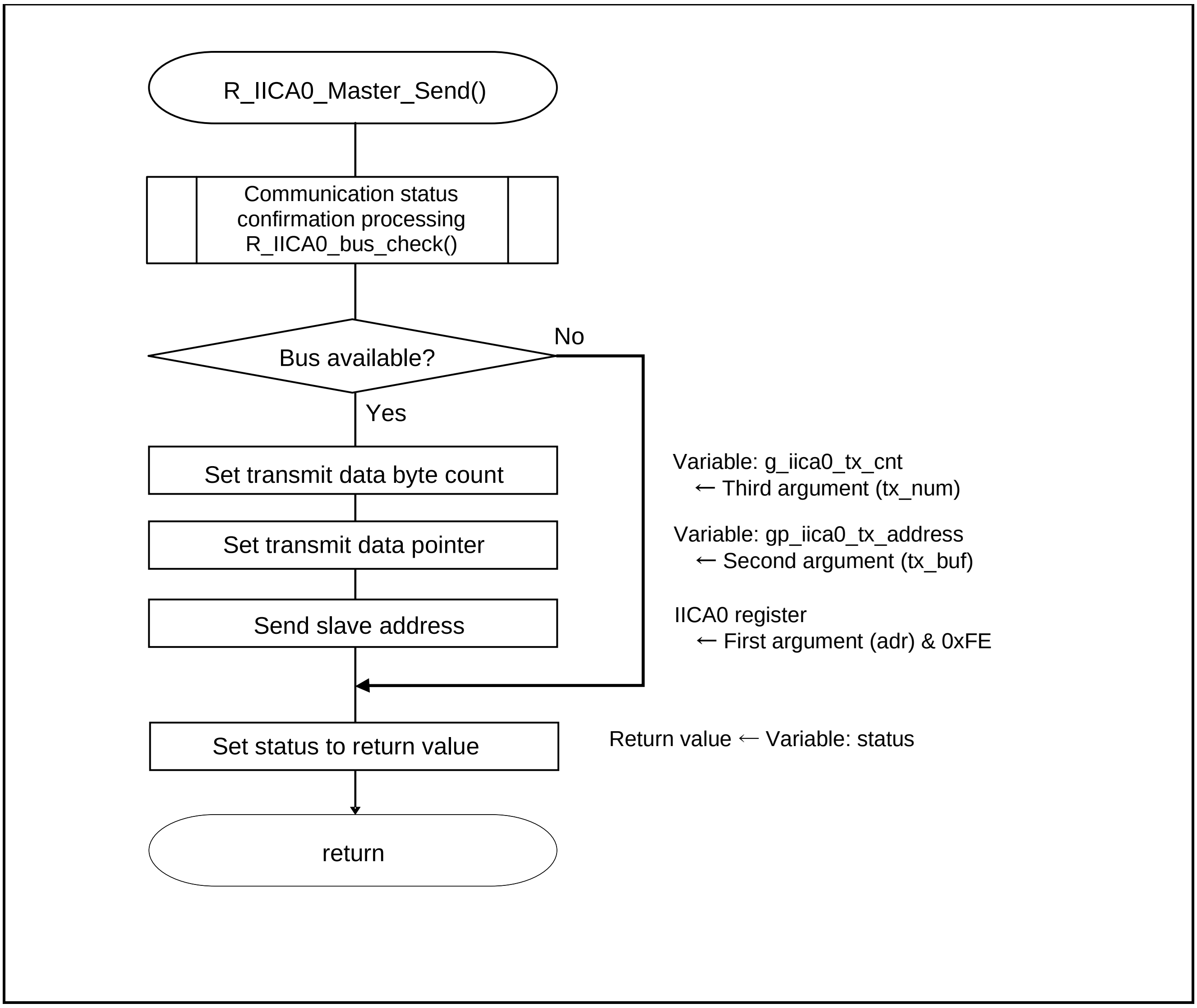


Figure 5.12 Master Transmission Start Request Processing

5.7.9 Master Reception Start Request Processing

Figure 5.13 shows the flowchart for the master reception start request processing.

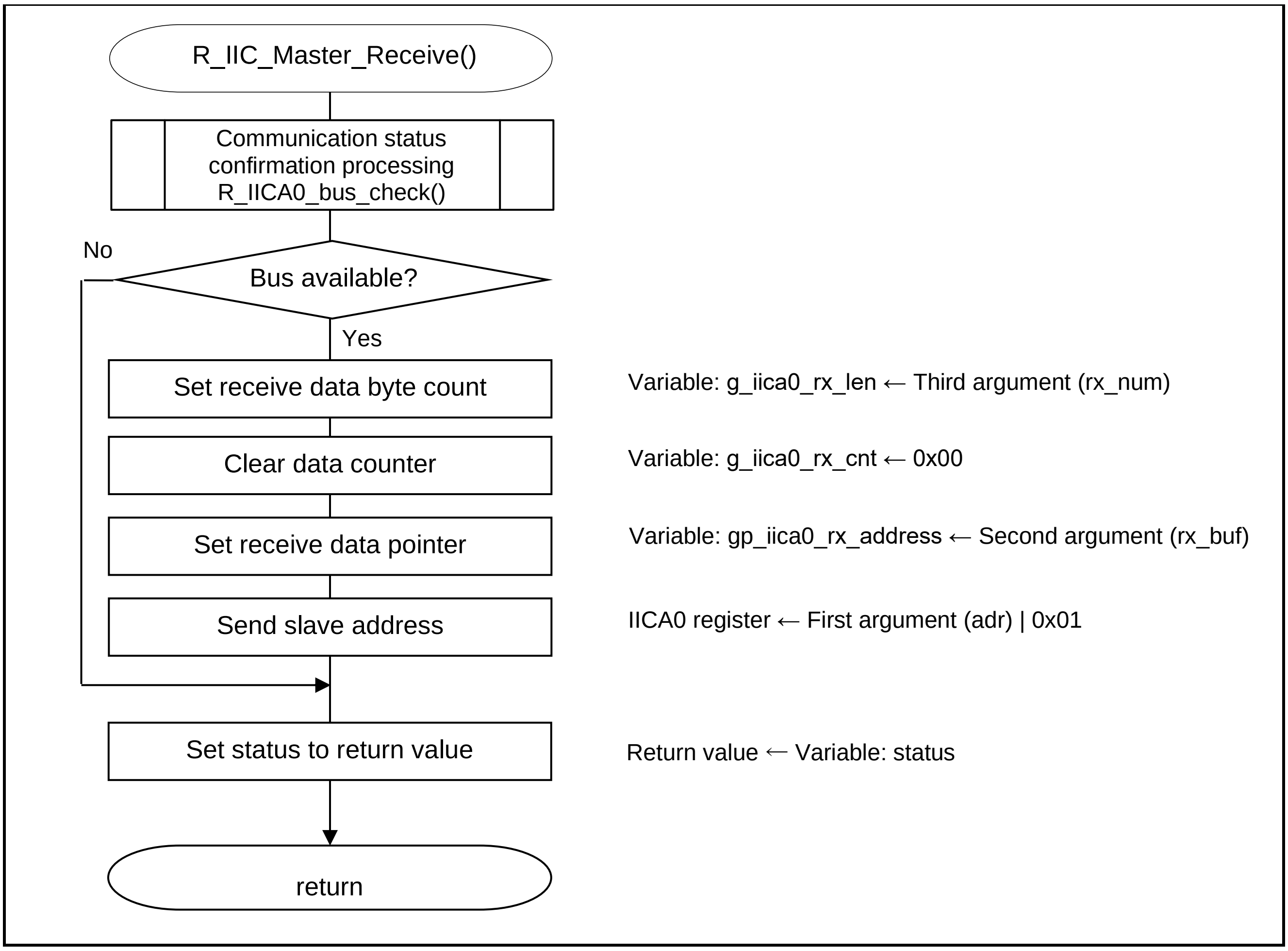


Figure 5.13 Master Reception Start Request Processing

5.7.10 IICA0 communication state confirmation processing

Figure 5.14 shows the flowchart for the IICA0 communication state confirmation processing.

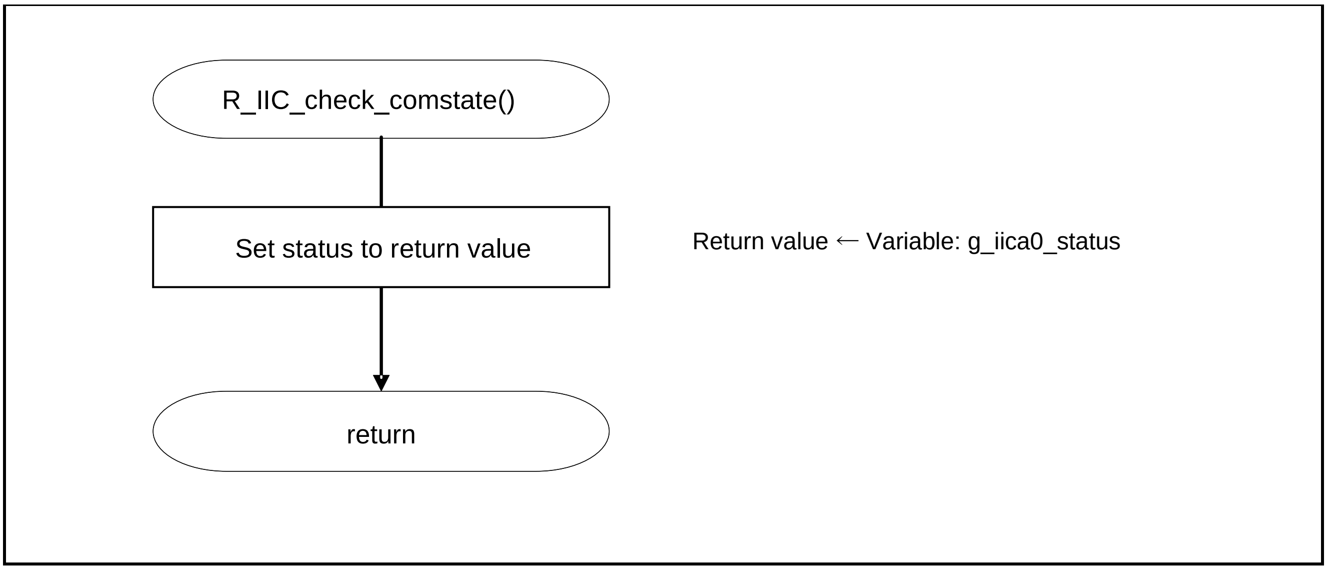


Figure 5.14 IICA0 communication state confirmation processing

5.7.11 Communication completion wait processing

Figure 5.15 shows the flowchart for the communication completion wait processing.

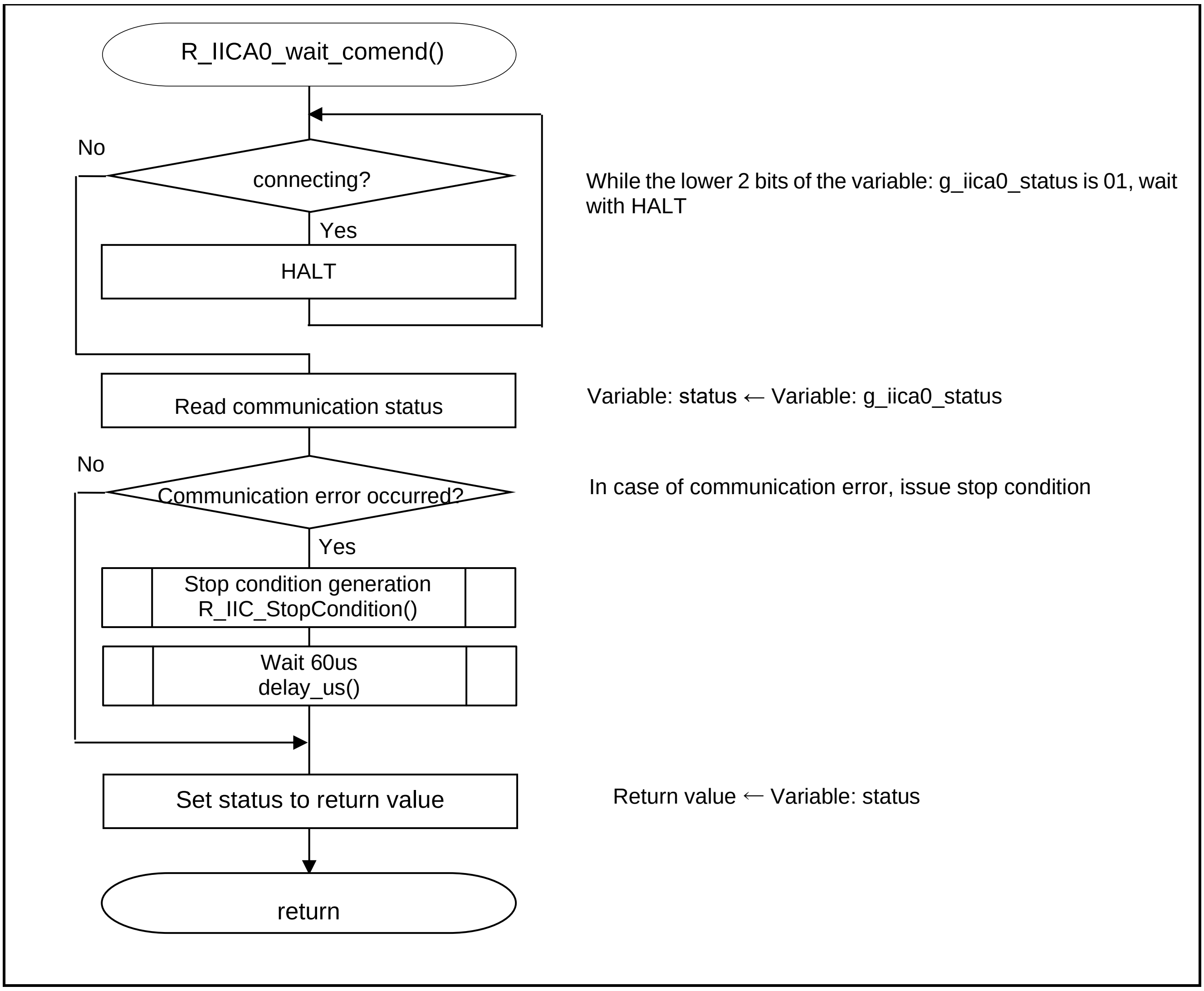


Figure 5.15 Communication completion wait processing

5.7.12 Stop Condition Issuance

Figure 5.16 shows the flowchart for issuing a stop condition.

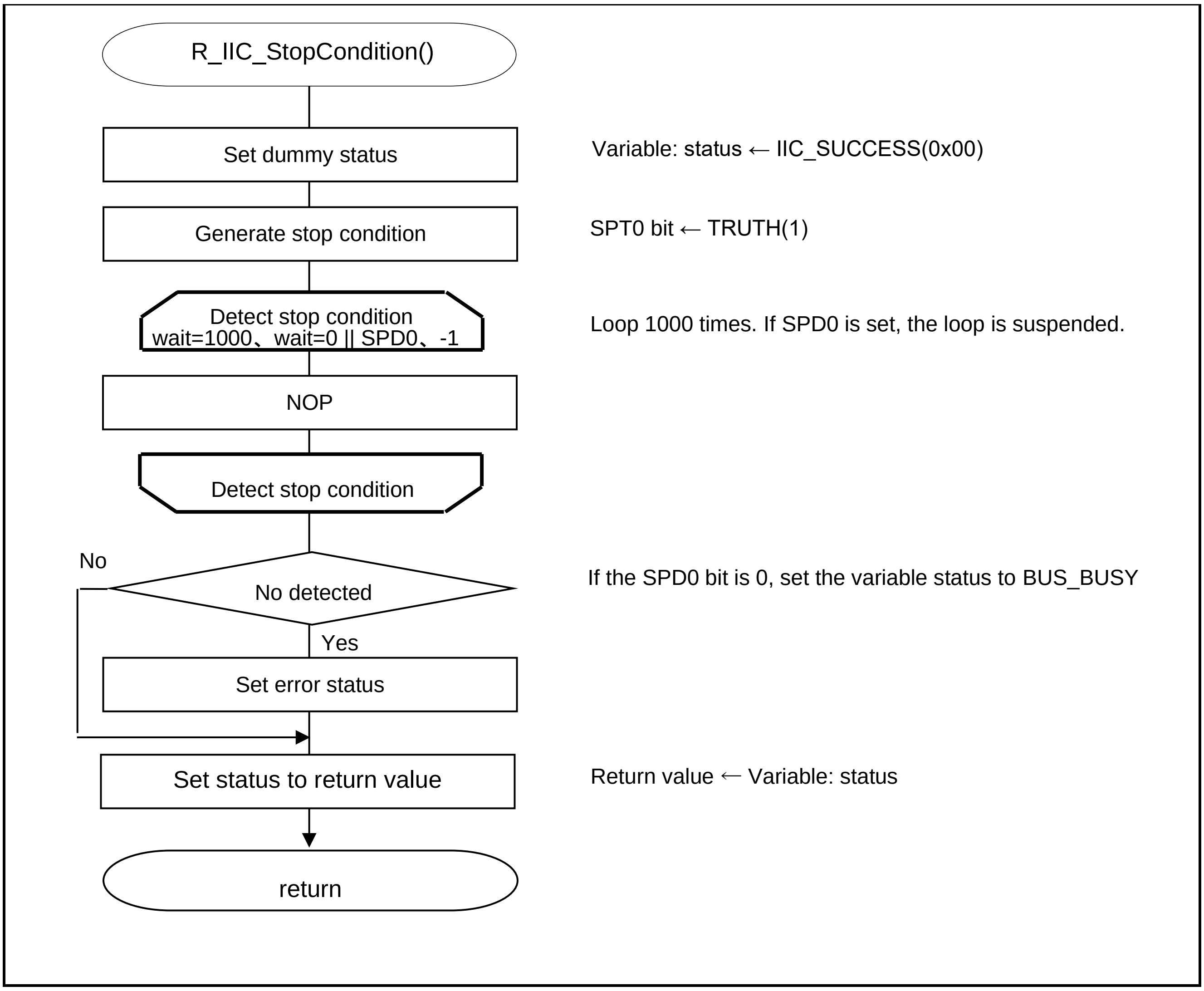


Figure 5.16 Stop Condition Issuance

5.7.13 Bus Status Confirmation Function

Figure 5.17 shows the flowchart for the bus status confirmation function.

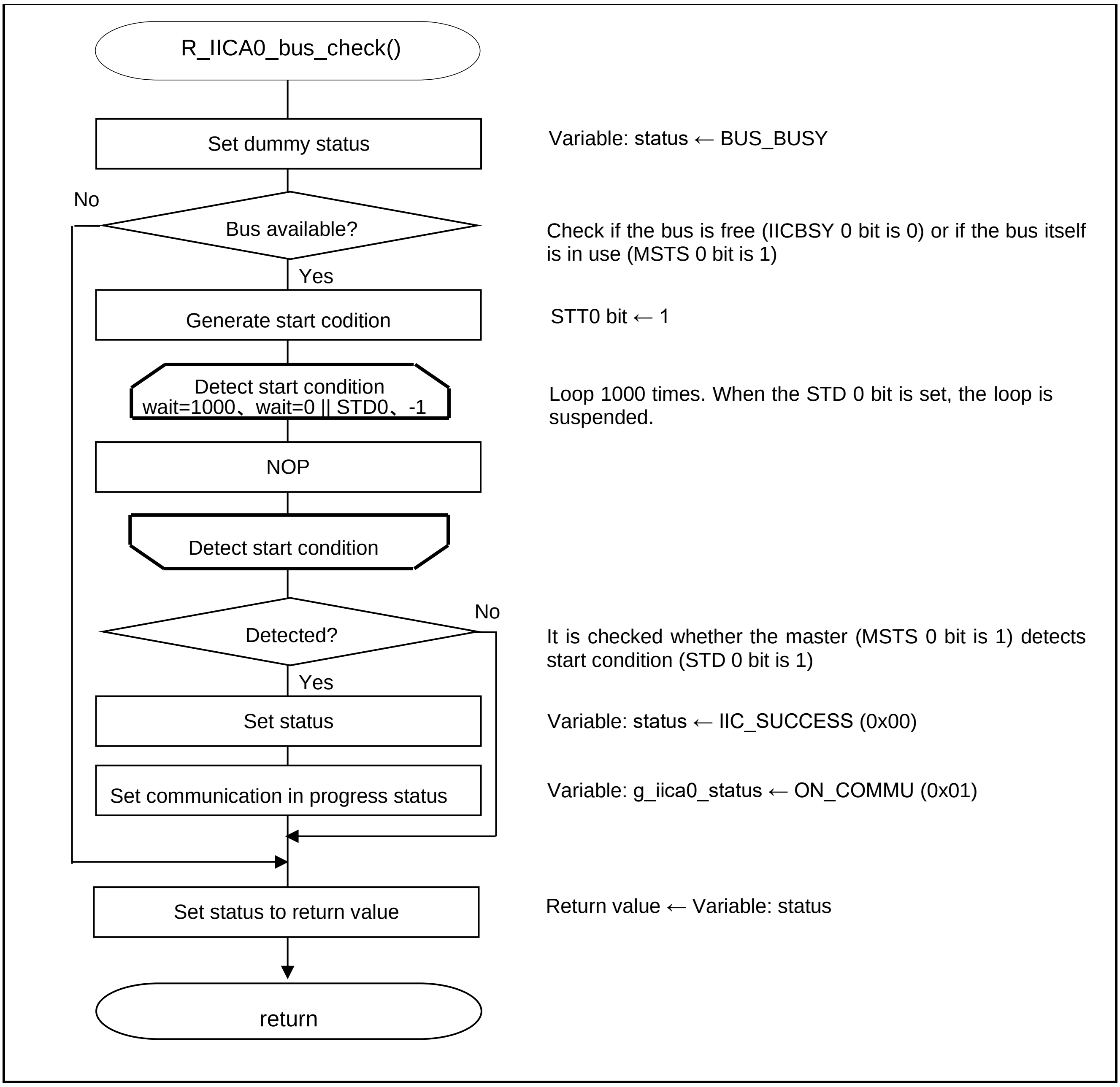


Figure 5.17 Bus Status Confirmation Function

5.7.14 IICA0 Interrupt Processing

Figure 5.18 through Figure 5.20 shows the flowchart for IICA0 interrupt processing.

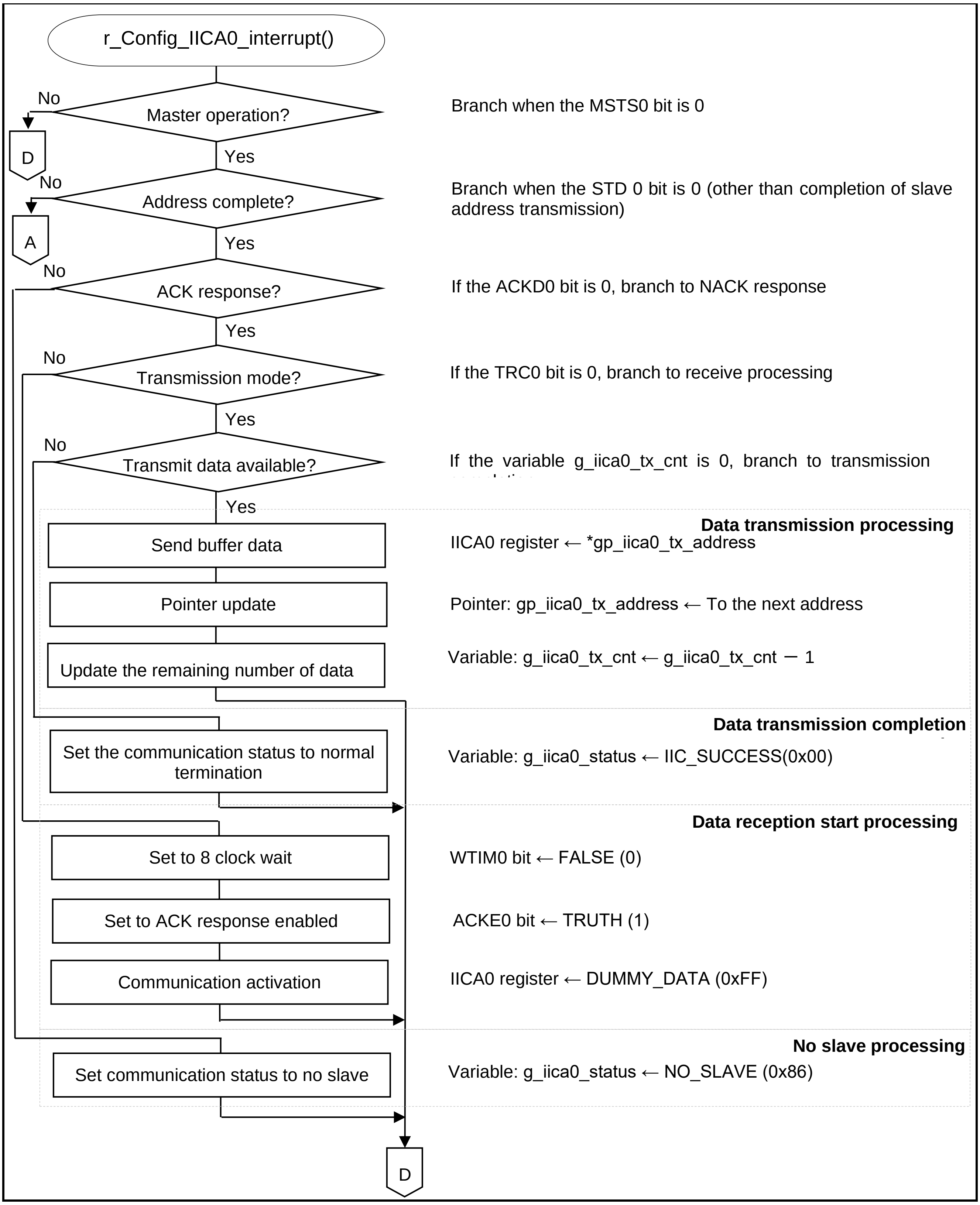


Figure 5.18 IICA0 Interrupt Processing (1/3)

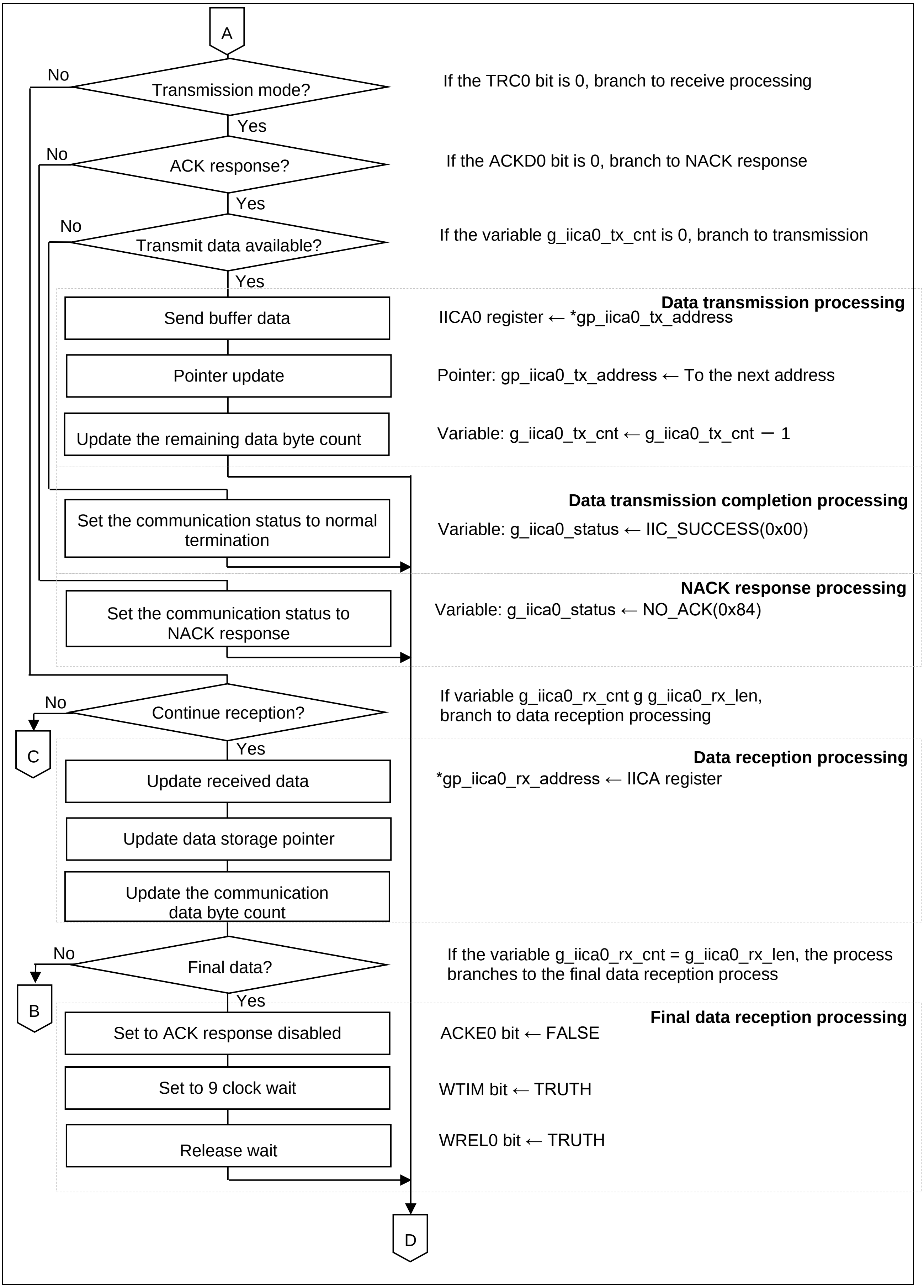


Figure 5.19 IICA0 Interrupt Processing (2/3)

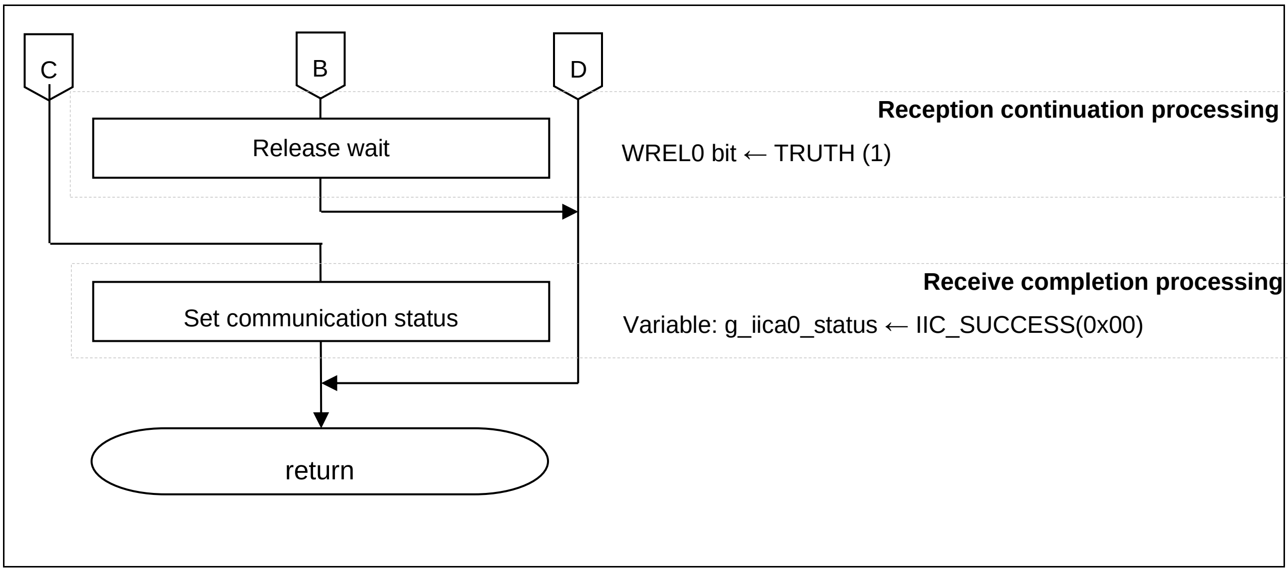


Figure 5.20 IICA0 Interrupt Processing (3/3)

5.7.15 Time wait processing in µs units

Figure 5.21 shows the flowchart for the time wait processing in µs.

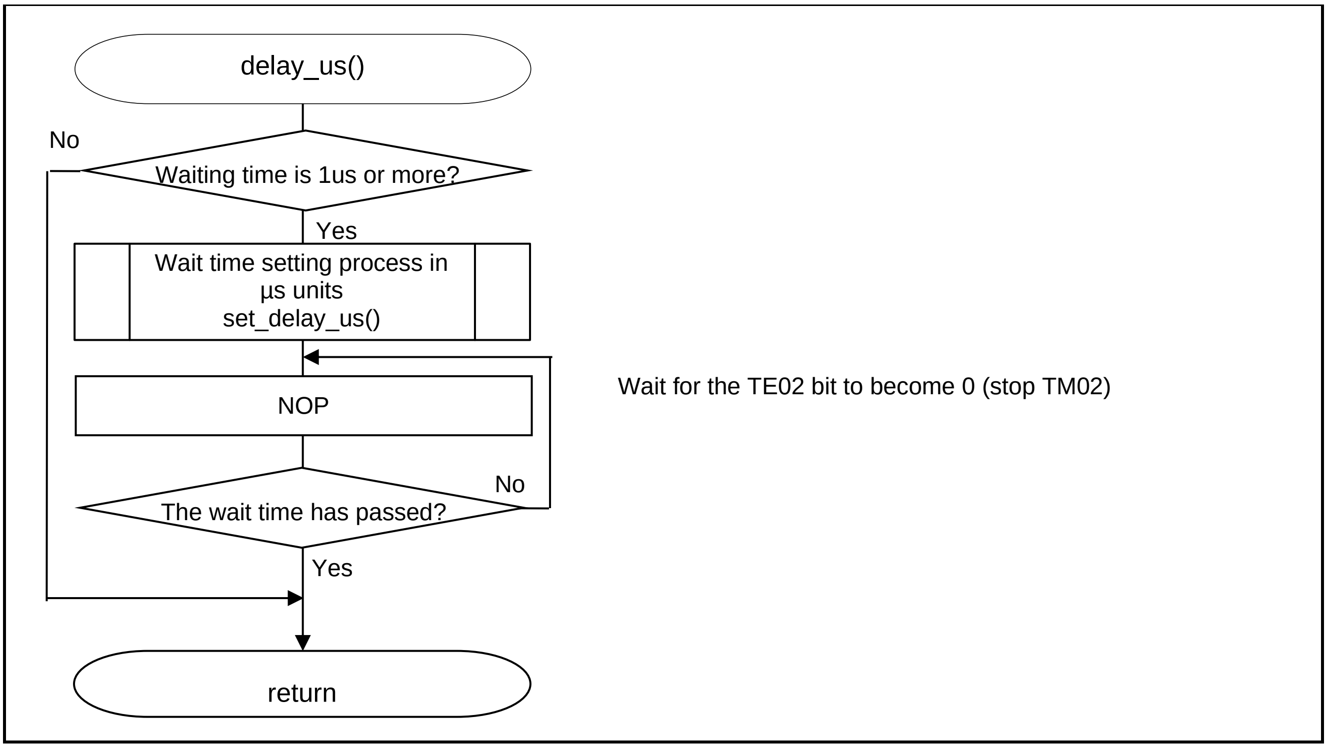


Figure 5.21 Time Wait Processing In µs units

5.7.16 Wait Time Setting Process In μs units

Figure 5.22 shows the flowchart for the wait time setting process in μs units.

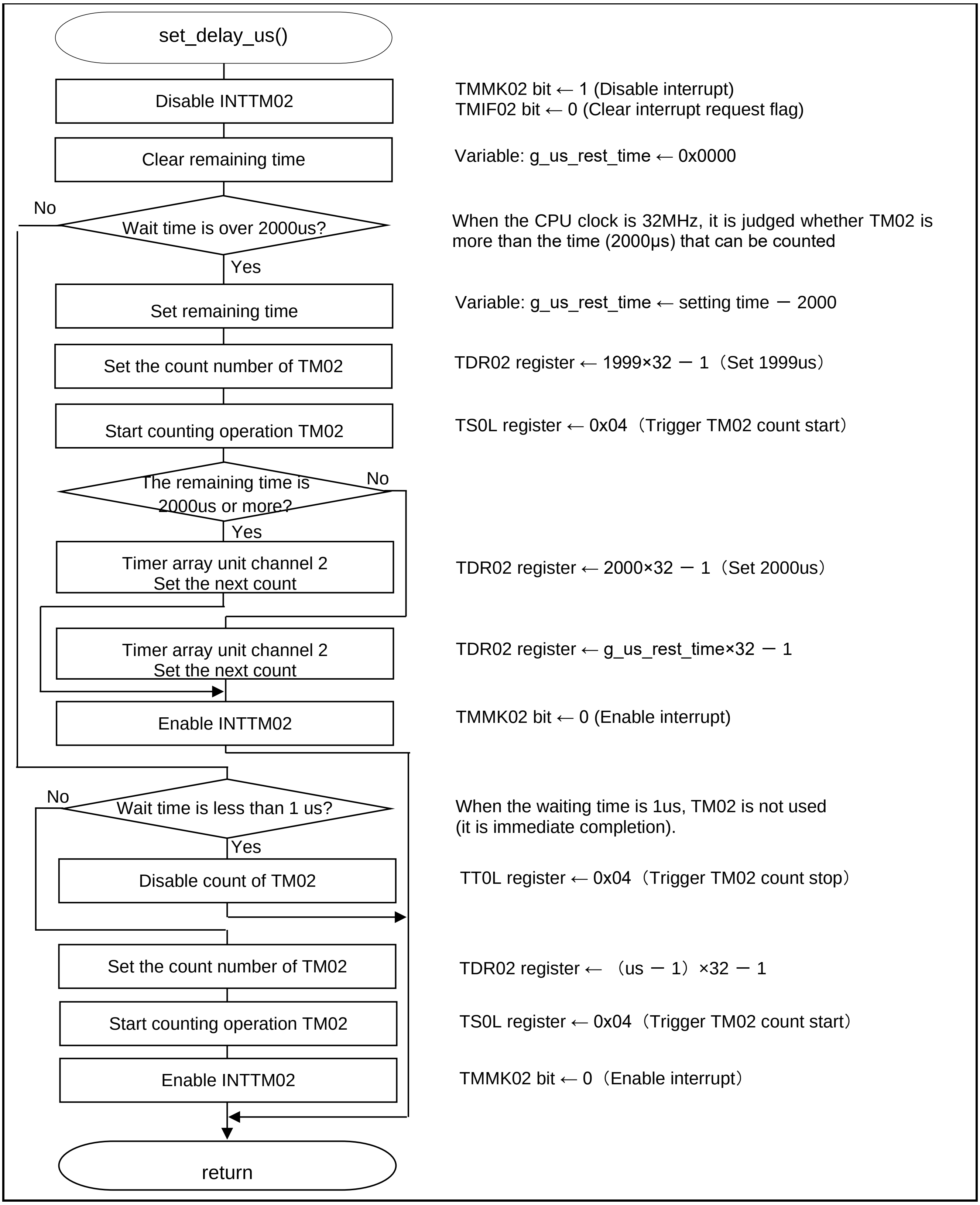


Figure 5.22 Wait Time Setting Process In μs units

5.7.17 Timer Array Unit Channel2 Interrupt Processing

Figure 5.23 shows the flowchart for the timer array unit channel2 interrupt processing.

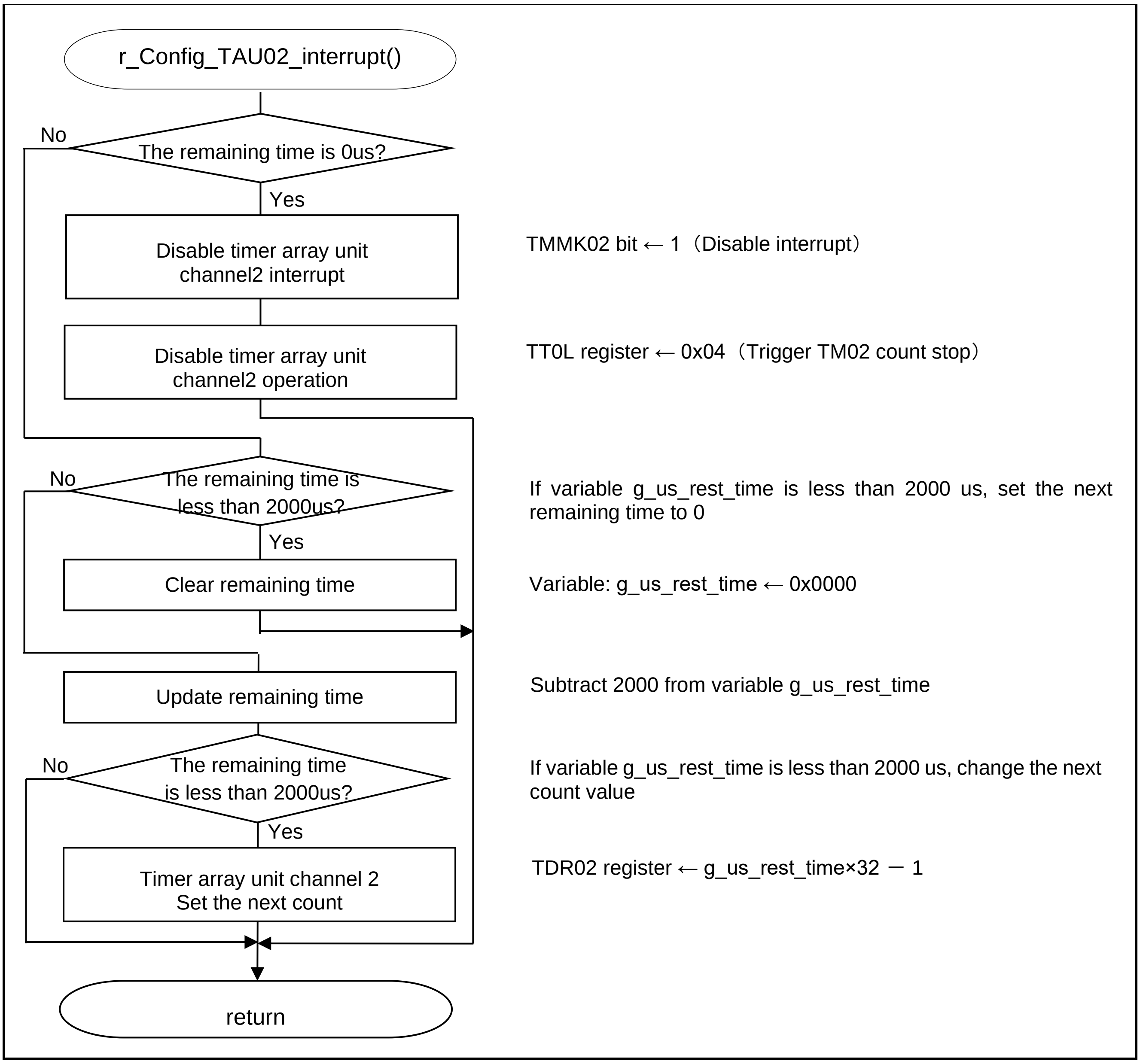


Figure 5.23 Timer Array Unit Channel2 Interrupt Processing

6. Sample Code

The sample code is available on the Renesas Electronics Website.

7. Documents for Reference

User’s Manual:

RL78/G15 User's Manual: Hardware (R01UH0959J)
RL78 Family User's Manual: Software (R01US0015EJ)
The latest version can be downloaded from the Renesas Electronics website.

Technical Updates/Technical News

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

Revision History

Rev.	Date	Description	
		Page	Summary
1.00	Oct. 5, 2023	—	First edition issued

General Precautions in the Handling of Microprocessing Unit and Microcontroller Unit Products

The following usage notes are applicable to all Microprocessing unit and Microcontroller unit 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. Precaution against Electrostatic Discharge (ESD)
   A strong electrical field, when exposed to a CMOS device, can cause destruction of the gate oxide and ultimately degrade the device operation. Steps must be taken to stop the generation of static electricity as much as possible, and quickly dissipate it when it occurs. Environmental control must be adequate. When it is dry, a humidifier should be used. This is recommended to avoid using insulators that can easily build up static electricity. Semiconductor devices must be stored and transported in an anti-static container, static shielding bag or conductive material. All test and measurement tools including work benches and floors must be grounded. The operator must also be grounded using a wrist strap. Semiconductor devices must not be touched with bare hands. Similar precautions must be taken for printed circuit boards with mounted semiconductor devices.
2. Processing at power-on
   The state of the product is undefined at the time 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 time 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 time 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 time when power is supplied until the power reaches the level at which resetting is specified.
3. Input of signal during power-off state
   Do not input signals or an I/O pull-up power supply while the device is powered off. The current injection that results from input of such a signal or I/O pull-up power supply may cause malfunction and the abnormal current that passes in the device at this time may cause degradation of internal elements. Follow the guideline for input signal during power-off state as described in your product documentation.
4. 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 the 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.
5. Clock signals
   After applying a reset, only release the reset line after the operating clock signal becomes stable. When switching the clock signal during program execution, wait until the target clock signal is 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. Additionally, 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.
6. Voltage application waveform at input pin
   Waveform distortion due to input noise or a reflected wave may cause malfunction. If the input of the CMOS device stays in the area between V_{IL} (Max.) and V_{IH} (Min.) due to noise, for example, the device may malfunction. Take care to prevent chattering noise from entering the device when the input level is fixed, and also in the transition period when the input level passes through the area between V_{IL} (Max.) and V_{IH} (Min.).
7. Prohibition of access to reserved addresses
   Access to reserved addresses is prohibited. The reserved addresses are provided for possible future expansion of functions. Do not access these addresses as the correct operation of the LSI is not guaranteed.
8. Differences between products
   Before changing from one product to another, for example to a product with a different part number, confirm that the change will not lead to problems. The characteristics of a microprocessing unit or microcontroller unit products in the same group but having a different part number might differ in terms of 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 or any other use of the circuits, software, and information in the design of your product or system. Renesas Electronics disclaims any and all liability for any losses and damages incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics hereby expressly disclaims any warranties against and liability for infringement or any other claims involving 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, including but not limited to, the product data, drawings, charts, programs, algorithms, and application examples.
3. 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 shall be responsible for determining what licenses are required from any third parties, and obtaining such licenses for the lawful import, export, manufacture, sales, utilization, distribution or other disposal of any products incorporating Renesas Electronics products, if required.
5. You shall not alter, modify, copy, or reverse engineer any Renesas Electronics product, whether in whole or in part. Renesas Electronics disclaims any and all liability for any losses or damages incurred by you or third parties arising from such alteration, modification, copying or reverse engineering.
6. Renesas Electronics products are classified according to the following two quality grades: "Standard" and "High Quality". The intended 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; industrial robots; etc.
   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control (traffic lights); large-scale communication equipment; key financial terminal systems; safety control equipment; etc.
   Unless expressly designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not intended or 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 damage (space system; undersea repeaters; nuclear power control systems; aircraft control systems; key plant systems; military equipment; etc.). Renesas Electronics disclaims any and all liability for any damages or losses incurred by you or any third parties arising from the use of any Renesas Electronics product that is inconsistent with any Renesas Electronics data sheet, user's manual or other Renesas Electronics document.
7. No semiconductor product is absolutely secure. Notwithstanding any security measures or features that may be implemented in Renesas Electronics hardware or software products, Renesas Electronics shall have absolutely no liability arising out of any vulnerability or security breach, including but not limited to any unauthorized access to or use of a Renesas Electronics product or a system that uses a Renesas Electronics product. RENESAS ELECTRONICS DOES NOT WARRANT OR GUARANTEE THAT RENESAS ELECTRONICS PRODUCTS, OR ANY SYSTEMS CREATED USING RENESAS ELECTRONICS PRODUCTS WILL BE INVULNERABLE OR FREE FROM CORRUPTION, ATTACK, VIRUSES, INTERFERENCE, HACKING, DATA LOSS OR THEFT, OR OTHER SECURITY INTRUSION ("Vulnerability Issues"). RENESAS ELECTRONICS DISCLAIMS ANY AND ALL RESPONSIBILITY OR LIABILITY ARISING FROM OR RELATED TO ANY VULNERABILITY ISSUES. FURTHERMORE, TO THE EXTENT PERMITTED BY APPLICABLE LAW, RENESAS ELECTRONICS DISCLAIMS ANY AND ALL WARRANTIES, EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT AND ANY RELATED OR ACCOMPANYING SOFTWARE OR HARDWARE, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE.
8. When using Renesas Electronics products, refer to the latest product information (data sheets, user's manuals, application notes, "General Notes for Handling and Using Semiconductor Devices" in the reliability handbook, etc.), and ensure that usage conditions are within the ranges specified by Renesas Electronics with respect to maximum ratings, operating power supply voltage range, heat dissipation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions, failure or accident arising out of the use of Renesas Electronics products outside of such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of Renesas Electronics products, semiconductor products have specific characteristics, such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Unless designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not subject to radiation resistance design. You are responsible for implementing safety measures to guard against the possibility of bodily injury, injury or damage caused by fire, and/or danger to the public in the event of a failure or malfunction of Renesas Electronics products, 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 and impractical, you are responsible for evaluating the safety of the final products or systems manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. You are responsible for carefully and sufficiently investigating applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive, and using Renesas Electronics products in compliance with all these applicable laws and regulations. Renesas Electronics disclaims any and all liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. Renesas Electronics products and technologies shall 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 shall comply with any applicable export control laws and regulations promulgated and administered by the governments of any countries asserting jurisdiction over the parties or transactions.
12. It is the responsibility of the buyer or distributor of Renesas Electronics products, or any other party who distributes, disposes of, or otherwise sells or transfers the product to a third party, to notify such third party in advance of the contents and conditions set forth in this document.
13. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
14. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products.

(Note1) "Renesas Electronics" as used in this document means Renesas Electronics Corporation and also includes its directly or indirectly controlled subsidiaries.

(Note2) "Renesas Electronics product(s)" means any product developed or manufactured by or for Renesas Electronics.

(Rev.5.0-1 October 2020)

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan

www.renesas.com

Trademarks

Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners.

Contact information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit: www.renesas.com/contact/.