RL78/F13, F14, F15, F23, F24
Setup Procedures for IICA Multi-Master Communication

R01AN4604EJ0110
Rev.1.10
2022.09.30

Introduction

This application note describes the setup procedures for transmission and reception in a multi-master environment employing the serial interface IICA (IICA) of the RL78/F13, F14, F15 and RL78/F23, F24 products. Under certain use conditions, the operations of the microcontroller might be different from the examples shown in this document. Customers are required to sufficiently evaluate the use of the IICA in their environment. Customers are also required to refer to the user's manual corresponding to their products for detailed functions of the IICA, clock generator, and interrupts.

Target Devices

RL78/F13, F14 and RL78/F15 products that support IICA

RL78/F23, F24 products

Contents

1. Specifications of IICA Communication in Multi-Master Environment

The conditions for use of the IICA communication in a multi-master environment that this application note describes are shown in Table 1-1. This application note adopts a specification that the master generates a STOP condition when a NACK is caused by the slave.

Table 1-1. Conditions for Use of IICA Multi-Master Communication

Items	Conditions for Use
CPU/peripheral hardware clock (f_{CLK})	32 MHz
IICA operation clock (f_{MCK})	f_{CLK}/2 (16 MHz)
IICA operation mode Note 1	Fast mode up to 400 kbps support
Own address	21H in this document
Slave address	Any (7-bit)
Acknowledgment control	Enabled. However, a master receiver switches the setting to "Disabled" after receiving the last byte
Digital filter	ON
Communication reservation function	Disabled
Restart condition generation	Disabled
Generation timing of wait and interrupt request Note 2	Operating as a master transmitter: At the 9th clock's falling edge Operating as a master receiver: At the 8th clock's falling edge Operating as a slave transmitter: At the 9th clock's falling edge Operating as a slave receiver: At the 9th clock's falling edge
Data transmitted by a master	1 byte to 32 bytes
Data received by a master	1 byte to 32 bytes
Data transmitted by a slave	1 byte to 32 bytes
Data received by a slave	1 byte to 32 bytes

Notes: 1. The IICA does not support high-speed mode (up to 3.4 Mbps) or ultra-fast mode (unidirectional bus, up to 5 Mbps).

2. During address transfer, the wait and interrupt request are generated at the falling edge of the ninth clock pulse. When a slave receives an extension code address, however, the wait and interrupt request are generated at the falling edge of the eighth clock pulse. For details of the interrupts, see **Section 2.8**.

The pin connections are shown in Figure 1-1. Pullup resistors need to be connected from the serial clock and data lines to the supplies as the SCLA0 and SDAA0 pins are used for open drain outputs. For each I²C system, the pull-up resistor value should be carefully calculated.

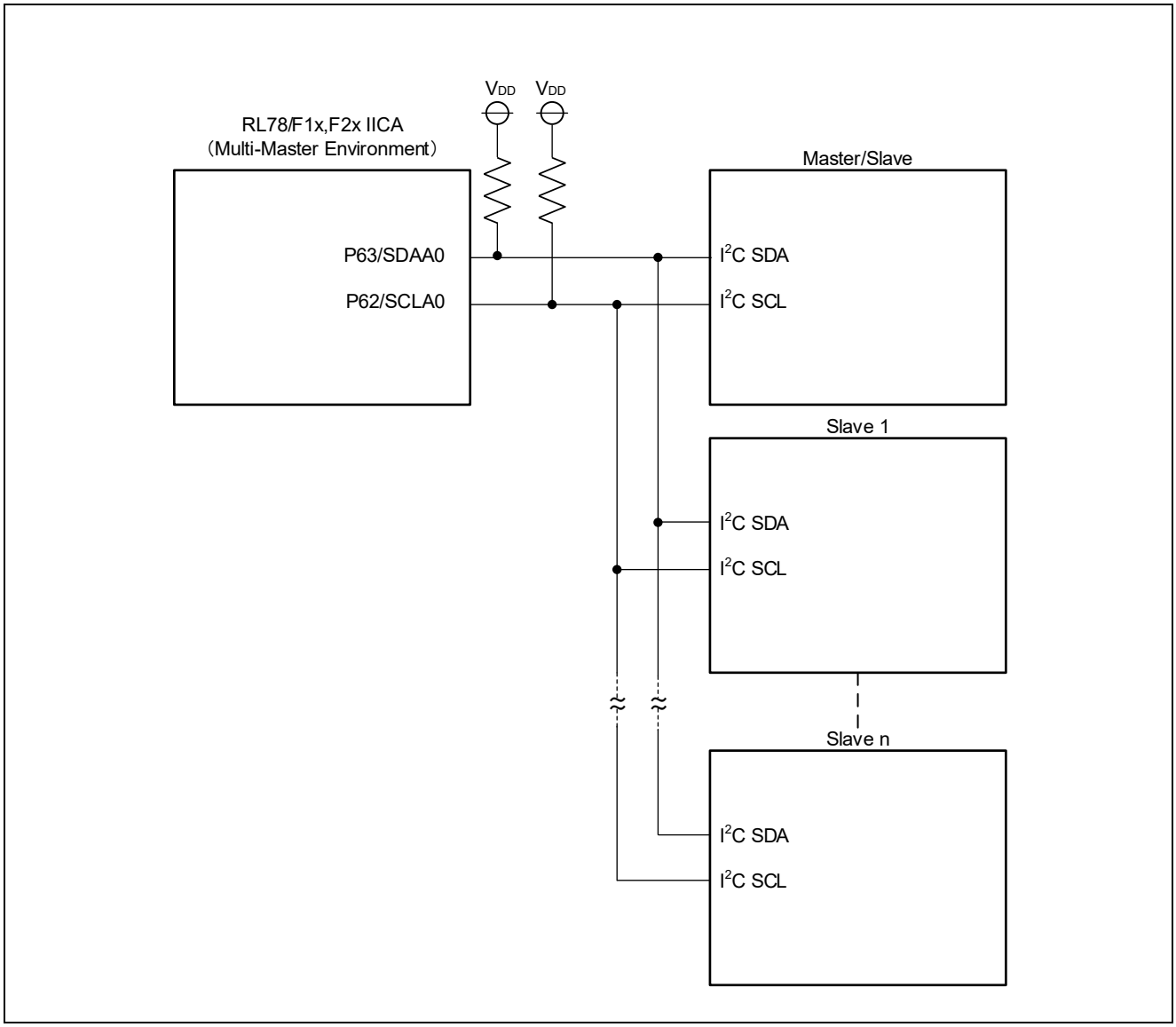


Figure 1-1 Pin Connections

1.1 Description of Memories Used

Memories (SFRs, RAMs) used for the IICA communication in a multi-master environment are listed below.

Table 1-2. SFRs Used for IICA Communication in Multi-Master Environment

Register Name	Read/ Write	Setting Example
Peripheral enable register 0 (PER0)	Write	IICA0EN = 1
IICA shift register 0 (IICA0)	Read/ Write	[During master operation] For address transmission: Address + R/$\overline{W}$ (Write) For data transfer: Any byte data (Read/Write) [During slave operation] For address reception: Address + R/$\overline{W}$ (Read) For data transfer: Any byte data (Read/Write)
Slave address register 0 (SVA0)	Write	42H (Own address "21H" is shifted left by one position)
IICA control register 00 (IICCTL00)	Write	Upon initialization: 9CH Upon IICA Interrupt: See each flow chart in **Section 2.8**
IICA flag register 0 (IICF0)	Read/ Write	03H
IICA status register 0 (IICS0)	Read	-
IICA control register 01 (IICCTL01)	Read/ Write	Upon Initialization: 3DH Upon IICA Interrupt: See each flow chart in **Section 2.8**
IICA low-level width setting register 0 (IICWL0)	Write	12H Note
IICA high-level width setting register 0 (IICWH0)	Write	10H Note
Port mode register 6 (PM6)	Write	PM62 = 0 PM63 = 0
Port register 6 (P6)	Write	P62 = 0 P63 = 0
Port output mode register 6 (POM6)	Write	POM62 = 1 POM63 = 1
Port input threshold control register 6 (PITHL6)	Write	PITHL62 = 0 PITHL63 = 0
Interrupt request flag register 1L (IF1L)	Write	Upon initialization: IICAIF0 = 0
Interrupt mask flag register 1L (MK1L)	Write	Upon initialization: IICAMK0 = 0
Interrupt priority specification flag 1 register 1L (PR11L) Interrupt priority specification flag 0 register 1L (PR01L)	Write	Upon initialization: IICAPR10, IICAPR00 = 00B

Note: Take the value of the pull-up resistors and wiring capacitance into consideration when calculating each value. For details on how to set up each register, refer to the user's manual.

In this document, the following abbreviations are used for each bit of SFRs related to the IICA communication in a multi-master environment.

- IICA0EN: A bit of the PER0 register
- SPT0, STT0, ACKE0, WTIM0, SPIE0, WREL0, LREL0, IICE0: Bits of the IICCTL00 register
- IICRSV0, STCEN0, IICBSY0, STCF0: Bits of the IICF0 register
- SPD0, STD0, ACKD0, TRC0, COI0, EXC0, ALD0, MSTS0: Bits of the IICS0 register
- PRS0, DFC0, SMC0, DAD0, CLD0, WUP0: Bits of the IICCTL01 register
- PM62, PM63: Bits of the PM6 register
- P62, P63: Bits of the P6 register
- POM62, POM63: Bits of the POM6 register
- PITHL62, PITHL63: Bits of the PITHL6 register
- IICAIF0: A bit of the IF1L register
- IICAMK0: A bit of the MK1L register
- IICAPR10, IICAPR00: Bits of the PR11L and PR01L registers

Table 1-3. RAMs/Variables Used for IICA Communication in Multi-Master Environment

Variable Name	Specification
u8_iic_status	IICA multi-master communication status 00H: Not initialized 01H: Initialized 02H: Initialized (Stop) 03H: Requesting master transmitter operation 04H: Requesting master receiver operation 05H: Operating as a master transmitter 06H: Operating as a master receiver 07H: Operating as a slave transmitter 08H: Operating as a slave receiver
u8_iic_txcount	Transmission data byte counter (Counter that holds the number of data transmitted)
u8_iic_rxcount	Reception data byte counter (Counter that holds the number of data received)
u8_iic_comerr	Communication error condition 00H: No communication error 01H: Master transmitter/receiver operation requested in any status except "Initialized" 02H: Command error (Master transmitter/receiver operation requested with an inappropriate parameter) 03H: Bit error (Data on the bus are not identical to the data that the device has transmitted as a master transmitter) 04H: Bit error (Data on the bus are not identical to the data that the device has transmitted as a slave transmitter) 05H: A NACK detection during data transmission (Master transmitter) 06H: Illegal interrupt 07H: Master transmitter operation request discarded due to bus stalls 08H: Master receiver operation request discarded due to bus stalls 09H: Excessive slave transmitter operation request (The device has been requested to transmit data as a slave transmitter beyond the maximum length of slave transmission data Note) 0AH: Excessive slave receiver operation request (The device has been requested to receive data as a slave receiver beyond the maximum length of slave reception data Note) 0BH: Bit error (Slave address on the bus is not identical to the address that the device has transmitted) 0CH: A NACK detection during address transmission (Master transmitter/receiver) 0DH: The device has failed in becoming a master or slave with its master transmitter/receiver operation request discarded (Arbitration loss) 0EH: The device has failed in becoming a master or slave with its master transmitter/receiver operation request discarded (No arbitration loss) 0FH: Master transmitter operation aborted with some data yet to be transmitted 10H: Master receiver operation aborted with some data yet to be received 11H: Communication aborted with master transmitter/receiver operation request discarded due to a STOP condition detection 12H: IICA initialization failure
u8_iic_slaveaddr	Slave address that the device is going to transmit when requesting master transmitter/receiver operation
u8_iic_mtxd_size	Length of data that the device is going to transmit when operating as a master transmitter (in bytes)
u8_iic_mrxd_size	Length of data that the device expects to receive when operating as a master receiver (in bytes)
u8_iic_stxd_rsize	Length of data that the device is going to transmit when operating as a slave transmitter (in bytes)
u8_iic_stxd_size	Length of data that the device has transmitted when operating as a slave transmitter (in bytes) (For user notification)
u8_iic_srxd_size	Length of data that the device has received when operating as a slave receiver (in bytes) (For user notification)
u8_iic_mtxd_buff [32]	Data that the device is going to transmit when operating as a master transmitter
u8_iic_stxd_buff [32]	Data that the device is going to transmit when operating as a slave transmitter
u8_iic_mrxd_buff [32]	Data that the device has received when operating as a master receiver
u8_iic_srxd_buff [32]	Data that the device has received when operating as a slave receiver
u8_iic_rxd_addr	Slave address and R/$\overline{W}$ bit that the device (slave) has received (For user notification. The value of the IICA0 register is written to this variable inside the IICA address receive interrupt routines (slave transmitter/receiver))

Note: This application note provides an example in which the maximum length of slave transmission/reception data is specified as 32 bytes.

2. Setup Procedures for IICA Multi-Master Communication

This chapter describes IICA communication processes (setup procedures) in a multi-master environment. **Figure 2-1** to **Figure 2-5** show IICA communication operations in a multi-master environment.

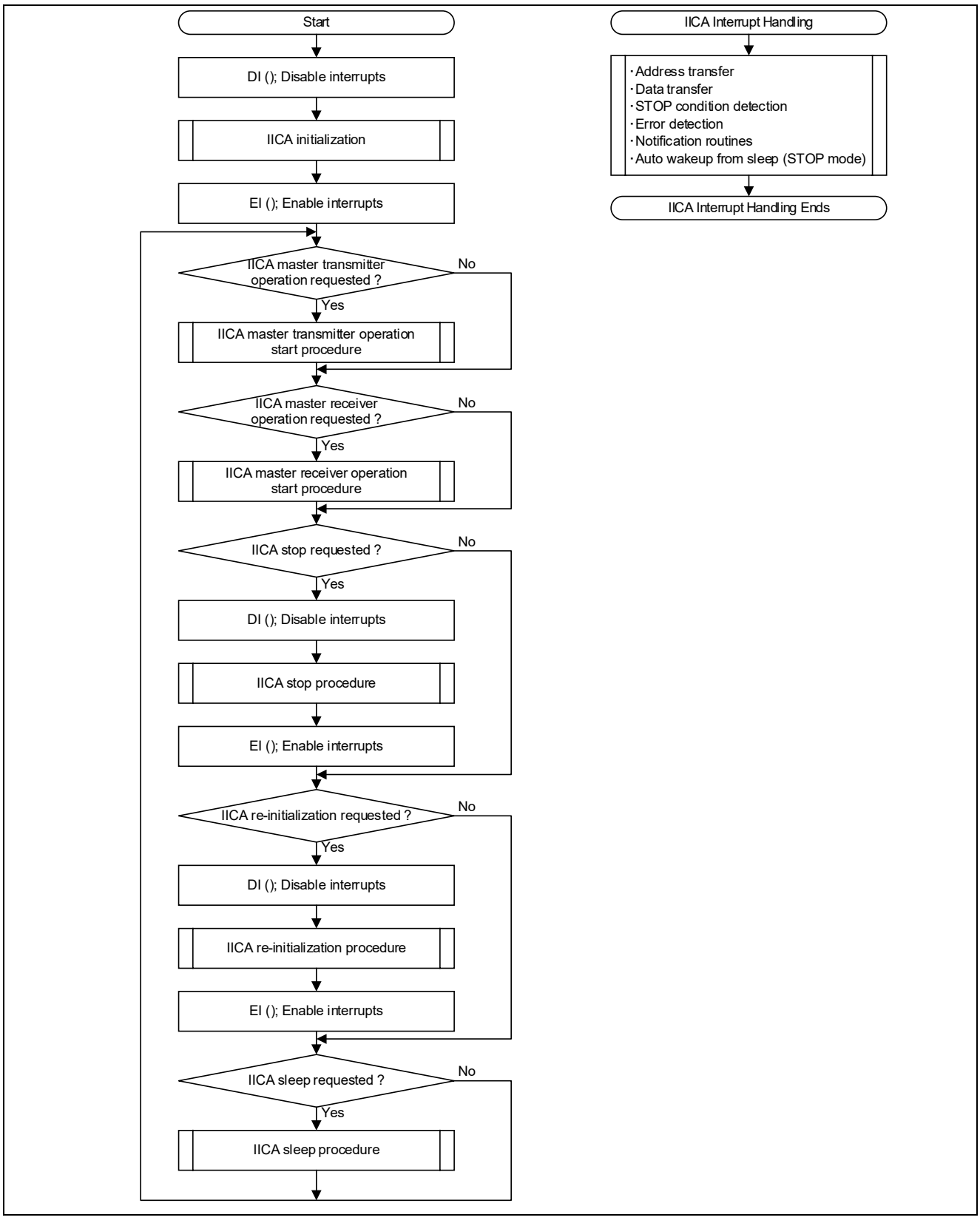
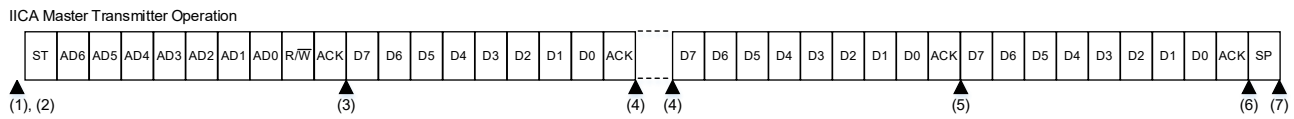


Figure 2-1. IICA Multi-Master Communication Process



Master Transmitter Operation

(1) Initialize the IICA.

(2) Set the START condition trigger (STT0←1) and specify the desired slave address and data direction (transmission) in the IICA0 register to transmit the slave address.

(3) An INTIICA0 interrupt (Address transmit) occurs. Make sure that neither an arbitration loss nor a NACK has occurred. Set the WTIM0 bit to 1 to generate the wait and interrupt request at the falling edge of the ninth clock pulse, and write the first data to be transmitted to the IICA0 register.

(4) An INTIICA0 interrupt (Data transfer) occurs. Make sure that neither an arbitration loss nor a NACK has occurred. And then, write the next data to be transmitted to the IICA0 register.

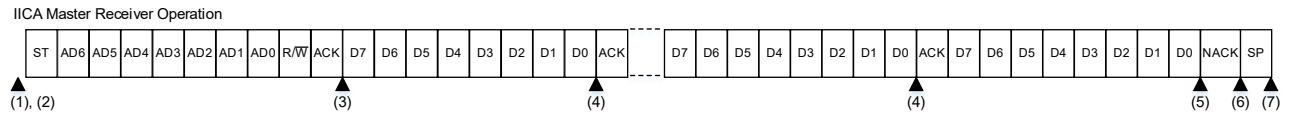
(5) An INTIICA0 interrupt (Data transfer) occurs. Make sure that neither an arbitration loss nor a NACK has occurred. And then, write the last data to be transmitted to the IICA0 register.

(6) An INTIICA0 interrupt (Data transfer) occurs. Set the STOP condition trigger (SPT0←1).

(7) An INTIICA0 interrupt (STOP condition detected) occurs and the master transmitter operation finishes.

Remark ST = START condition, AD6-AD0 = Address, R/$\overline{W}$ = Data direction bit, ACK = Acknowledge, D7-D0 = Data, SP = STOP condition

Figure 2-2. IICA Master Transmitter Operation (Timing Diagram)



Master Receiver Operation

(1) Initialize the IICA.

(2) Set the START condition trigger (STT0←1) and specify the desired slave address and data direction (reception) in the IICA0 register to transmit the slave address.

(3) An INTIICA0 interrupt (Address transmit) occurs. Make sure that neither an arbitration loss nor a NACK has occurred. Clear the WTIM0 bit to 0 to generate the wait and interrupt request at the falling edge of the eighth clock pulse, and cancel the wait state (WREL0←1).

(4) An INTIICA0 interrupt (Data transfer) occurs. Make sure that an arbitration has not been lost. And then, read the received data from the IICA0 register and cancel the wait state (WREL0←1).

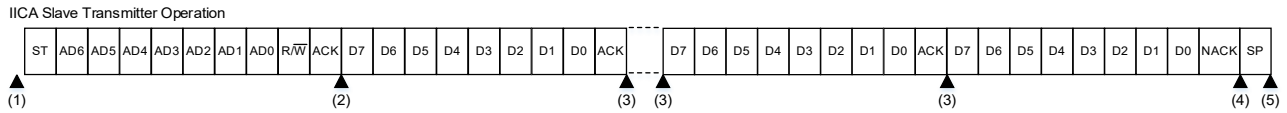
(5) An INTIICA0 interrupt (Data transfer) occurs. Make sure that an arbitration has not been lost. And then, read the last received data from the IICA0 register. Set the WTIM0 bit to 1 to generate the wait and interrupt request at the falling edge of the ninth clock pulse, generate a NACK (ACKE0←0) to signal to the slave that the device has received the last data, and cancel the wait state (WREL0←1).

(6) An INTIICA0 interrupt (Data transfer) occurs. Set the STOP condition trigger (SPT0←1).

(7) An INTIICA0 interrupt (STOP condition detected) occurs and the master receiver operation finishes.

Remark ST = START condition, AD6-AD0 = Address, R/$\overline{W}$ = Data direction bit, ACK = Acknowledge, NACK = Not Acknowledge, D7-D0 = Data, SP = STOP condition

Figure 2-3. IICA Master Receiver Operation (Timing Diagram)



Slave Transmitter Operation

(1) Initialize the IICA.

(2) An INTIICA0 interrupt (Address match) occurs. Make sure that the device has been requested to be a slave through "STD0 = 1 (START condition detected)", "MSTS0 = 0 (slave device status)", "COI0 = 1 (address match), and "TRC0 = 1 (transmission)". Set the WTIM0 bit to 1 to generate the wait and interrupt request at the falling edge of the ninth clock pulse, and write the first data to be transmitted to the IICA0 register.

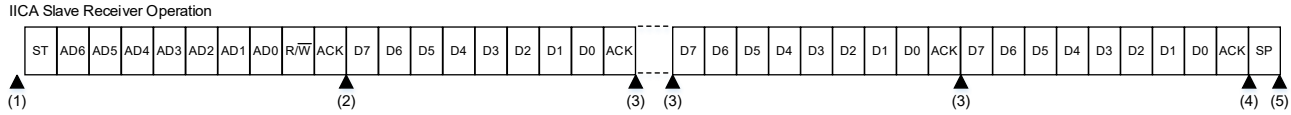
(3) An INTIICA0 interrupt (Data transfer) occurs. Make sure that an ACK has been detected. And then, write the next data to be transmitted to the IICA0 register.

(4) An INTIICA0 interrupt (Data transfer) occurs. Make sure that a NACK has been sent from the master, and then cancel the wait state (WREL0←1).

(5) An INTIICA0 interrupt (STOP condition detected) occurs and the slave transmitter operation finishes.

Remark ST = START condition, AD6-AD0 = Address, R/$\overline{W}$ = Data direction bit, ACK = Acknowledge, NACK = Not Acknowledge, D7-D0 = Data, SP = STOP condition

Figure 2-4. IICA Slave Transmitter Operation (Timing Diagram)



Slave Receiver Operation

(1) Initialize the IICA.

(2) An INTIICA0 interrupt (Address match) occurs. Make sure that the device has been requested to be a slave through "STD0 = 1 (START condition detected)", "MSTS0 = 0 (slave device status)", and "TRC0 = 0 (reception)". Set the WTIM0 bit to 1 to generate the wait and interrupt request at the faling edge of the ninth clock pulse, and cancel the wait state (WREL0←1).

(3) An INTIICA0 interrupt (Data transfer) occurs. Read the received data from the IICA0 register and cancel the wait state (WREL0←1).

(4) An INTIICA0 interrupt (Data transfer) occurs. Read the last received data from the IICA0 register and cancel the wait state (WREL0←1).

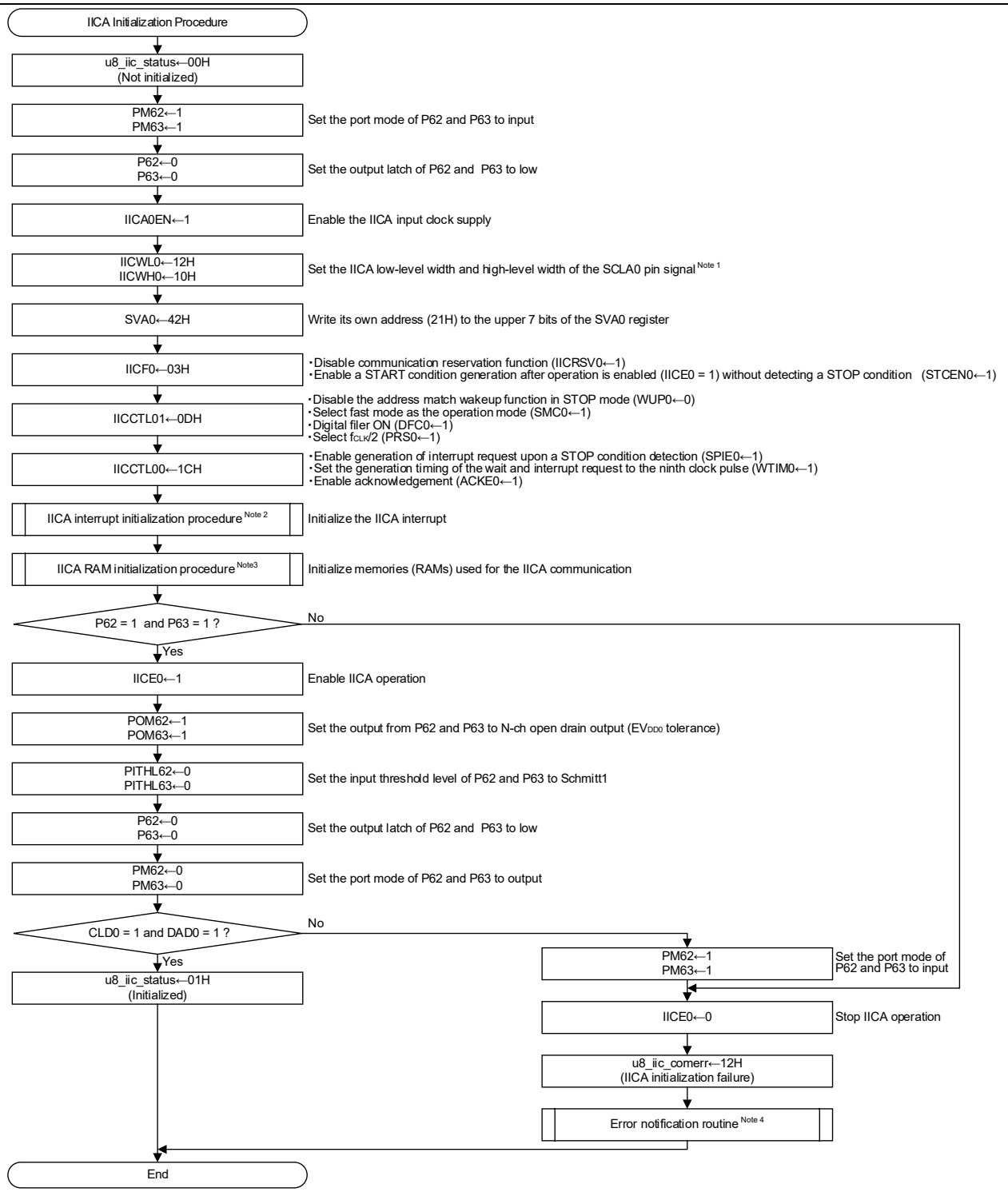
(5) An INTIICA0 interrupt (STOP condition detected) occurs and the slave receiver operation finishes.

Remark ST = START condition, AD6-AD0 = Address, R/$\overline{W}$ = Data direction bit, ACK = Acknowledge, D7-D0 = Data, SP = STOP condition

Figure 2-5. IICA Slave Receiver Operation (Timing Diagram)

2.1 IICA Initialization

A procedure for initializing the IICA is shown in **Figure 2-6**.



Notes: 1. Take the value of the pull-up resistors and wiring capacitance into consideration when calculating each value. For details on how to set up each register, refer to the user's manual.
2. See **Section 2.1.1**.
3. See **Section 2.1.2**.
4. See **Section 2.9.2**.

Figure 2-6. IICA Initialization Procedure

2.1.1 IICA Interrupt Initialization

A procedure for initializing the IICA interrupt is shown in **Figure 2-7**. This document provides an example in which the IICA interrupt priority level is specified as level 0.

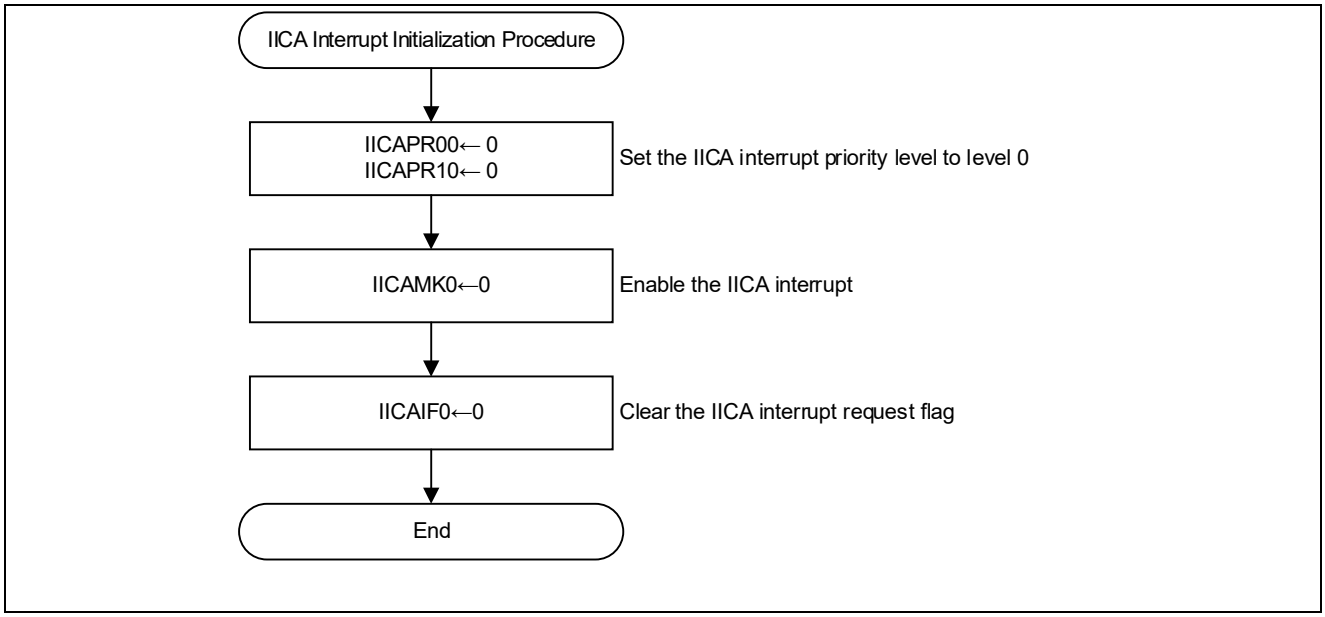


Figure 2-7. IICA Interrupt Initialization Procedure

2.1.2 IICA RAM Initialization

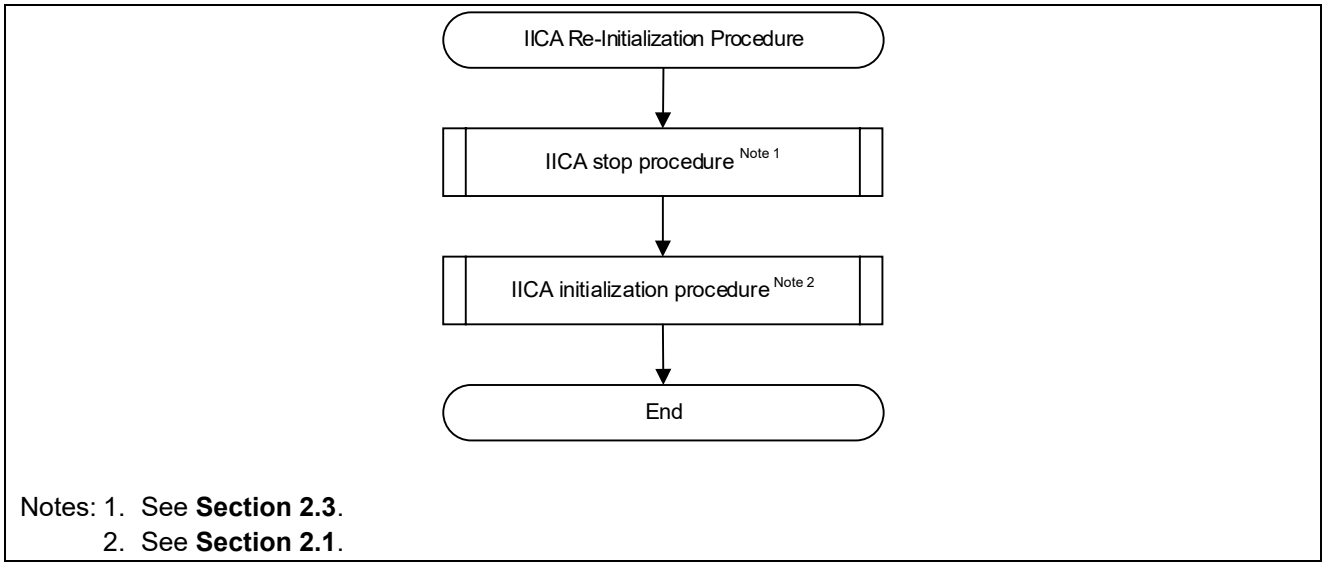
Memories (RAMs) used for the IICA communication in a multi-master environment are initialized in the IICA RAM initialization procedure. The variables initialized in this procedure and their initial values are listed in Table 2-1.

Table 2-1. Variables Initialized in IICA RAM Initialization Procedure

Variable Name	Description	Initial Value
u8_iic_status	IICA multi-master communication status	00H: Not initialized
u8_iic_txcount	Transmission data byte counter (Counter that holds the number of data transmitted)	00H
u8_iic_rxcount	Reception data byte counter (Counter that holds the number of data received)	00H
u8_iic_comerr	Communication error condition	00H: No communication error
u8_iic_slaveaddr	Slave address that the device is going to transmit when requesting master transmitter/receiver operation	00H
u8_iic_mtxd_size	Length of data that the device is going to transmit when operating as a master transmitter (in bytes)	00H
u8_iic_mrxd_size	Length of data that the device expects to receive when operating as a master receiver (in bytes)	00H
u8_iic_stxd_rsize	Length of data that the device is going to transmit when operating as a slave transmitter (in bytes)	00H
u8_iic_stxd_size	Length of data that the device has transmitted when operating as a slave transmitter (in bytes) (For user notification)	00H
u8_iic_srxd_size	Length of data that the device has received when operating as a slave receiver (in bytes) (For user notification)	00H
u8_iic_mtxd_buff [32]	Data that the device is going to transmit when operating as a master transmitter	00H
u8_iic_stxd_buff [32]	Data that the device is going to transmit when operating as a slave transmitter	00H
u8_iic_mrxd_buff [32]	Data that the device has received when operating as a master receiver	00H
u8_iic_srxd_buff [32]	Data that the device has received when operating as a slave receiver	00H
u8_iic_rxd_addr	Slave address and R/$\overline{W}$ bit that the device (slave) has received (For user notification. The value of the IICA0 register is written to this variable inside the IICA address receive interrupt routines (slave transmitter/receiver).)	00H

2.2 IICA Re-Initialization

A procedure for re-initializing the IICA is shown in **Figure 2-8**.



Notes: 1. See **Section 2.3**.
2. See **Section 2.1**.

Figure 2-8. IICA Re-Initialization Procedure

2.3 IICA Stop

A procedure for stopping the IICA is shown in **Figure 2-9**.

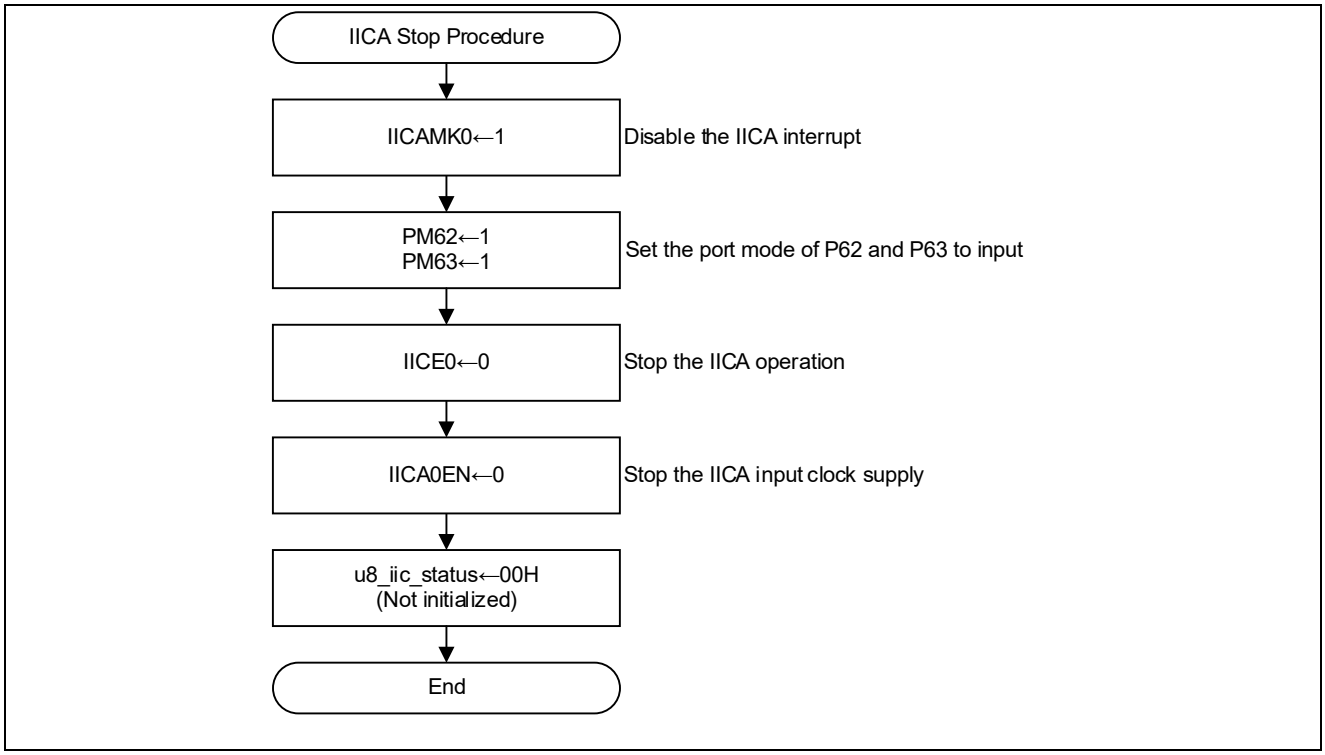


Figure 2-9. IICA Stop Procedure

2.4 IICA Sleep (Procedure for Putting Device into STOP Mode)

After being put into STOP mode with the address match wakeup function (WUP0 = 1) enabled, a device can be wakened up and serve as a slave when receiving an extension code address or its own address. An IICA sleep procedure (procedure for putting the device into STOP mode) is shown in Figure 2-10. Using the IICA status information API (see **Section 2.7**), make sure that the IICA multi-master communication status is "Initialized" before invoking this procedure.

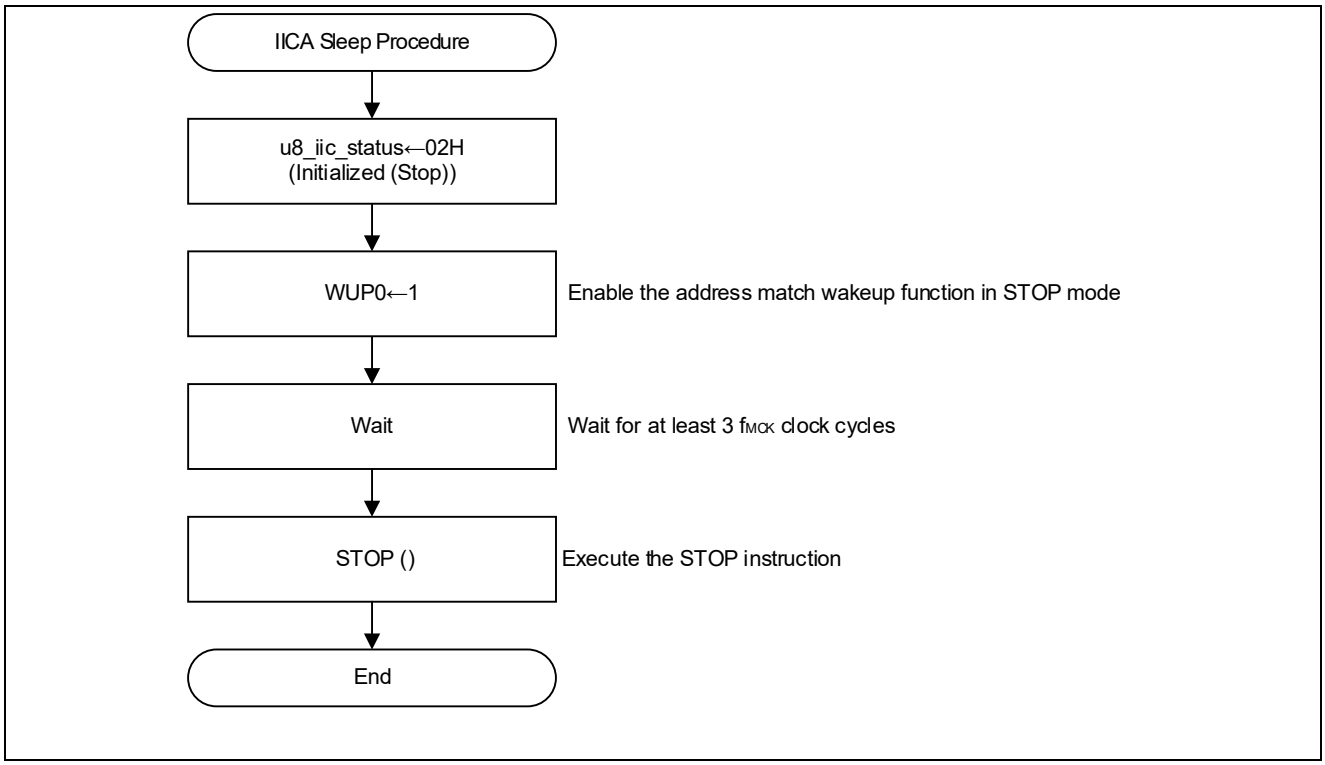
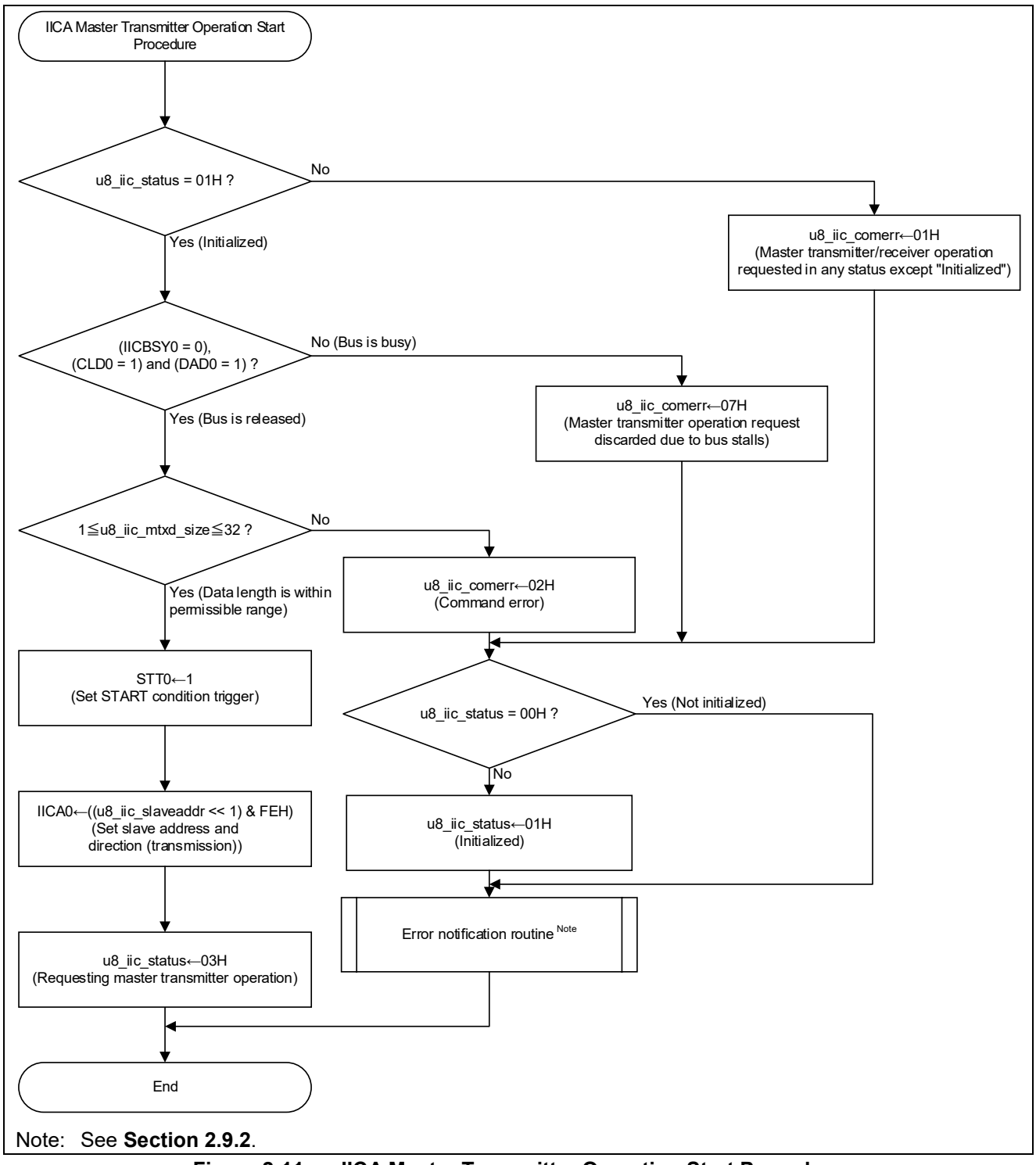


Figure 2-10. IICA Sleep Procedure

2.5 IICA Master Transmitter Operation Start

A procedure for initiating IICA master transmitter operation is shown in **Figure 2-11**. Before invoking this procedure, set up the following variables.

- u8_iic_slaveaddr : Slave address that the device is going to transmit
- u8_iic_mtxd_size : Length of data that the device is going to transmit (in bytes)
- u8_iic_mtxd_buff [32] : Data that the device is going to transmit



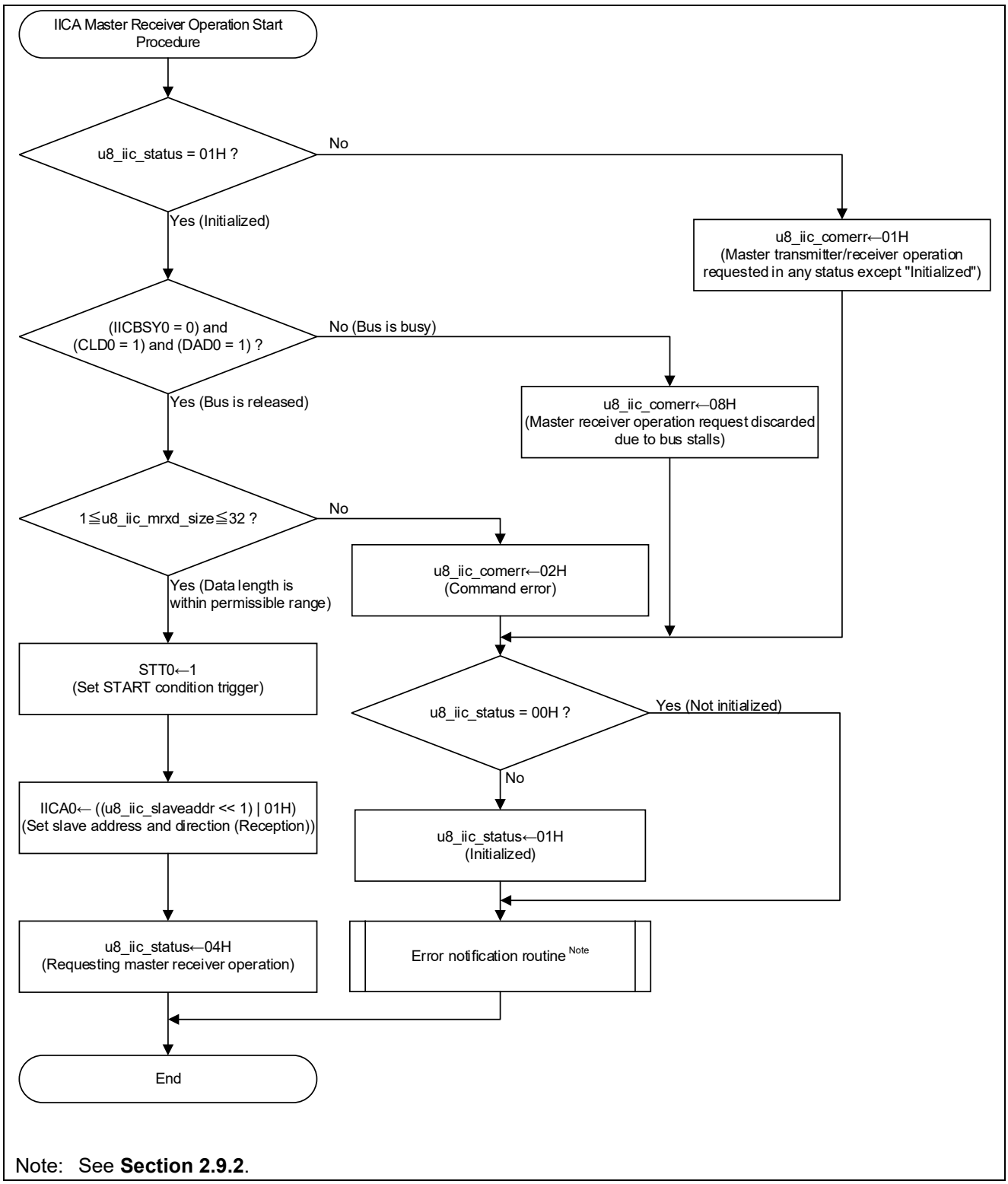
Note: See **Section 2.9.2**.

Figure 2-11. IICA Master Transmitter Operation Start Procedure

2.6 IICA Master Receiver Operation Start

A procedure for initiating IICA master receiver operation is shown in **Figure 2-12**. Before invoking this procedure, set up the following variables.

- u8_iic_slaveaddr : Slave address that the device is going to transmit
- u8_iic_mrxd_size : Length of data that the device expects to receive (in bytes)



Note: See **Section 2.9.2**.

Figure 2-12. IICA Master Receiver Operation Start Procedure

2.7 IICA Status Information API

The details of the IICA status that the IICA status information API provides are described in **Table 2-2.** Examine the I²C bus status and/or IICA multi-master communication status before invoking the IICA master transmitter/receiver operation start procedure or the IICA sleep procedure.

Table 2-2. ICA Status Details Returned from IICA Status Information API

Bit	Description	
b3-b0	IICA internal status (u8_iic_status)	IICA multi-master communication status 0H: Not initialized 1H: Initialized 2H: Initialized (Stop) 3H: Requesting master transmitter operation 4H: Requesting master receiver operation 5H: Operating as a master transmitter 6H: Operating as a master receiver 7H: Operating as a slave transmitter 8H: Operating as a slave receiver
b4	SDAA0 pin level	The value of the DAD0 bit of the IICCTL01 register 0: Low level 1: High level
b5	SCLA0 pin level	The value of the CLD0 bit of the IICCTL01 register 0: Low level 1: High level
b6	I²C bus status	The value of the IICBSY0 bit of the IICF0 register 0: Bus is released 1: Bus is busy
b7	START condition generation status	The value of the STCF0 bit of the IICF0 register 0: The attempt to generate a START condition by STT0 = 1 has been successful 1: The attempt to generate a START condition by STT0 = 1 has been unsuccessful

2.8 IICA Interrupt

The IICA Interrupt (INTIICA0 interrupt) occurs under the following conditions:

- A device transmits a slave address
- A device is requested to be a slave (receives its own address or an extension code address)
- A device transmits data
- A device receives data
- A device detects a STOP condition

This application note provides examples in which this interrupt is broken down by the IICS0 register value when the interrupt has occurred. The interrupts into which the IICA interrupt is broken down are listed in **Table 2-3** and its procedure is shown in **Figure 2-13**.

The IICA Interrupt is not always generated after the master transmitter/receiver operation procedures are invoked.[Note] After invoking these procedures, use the IICA status information API (see **Section 2.7**) to make sure that a START condition has been successfully generated and the IICA internal status has not been fixed. In the case of abnormal situation, invoke the IICA stop procedure (see **Section 2.3**) and then IICA initialization procedure (see **Section 2.1**) to initialize the bus and IICA internal status.

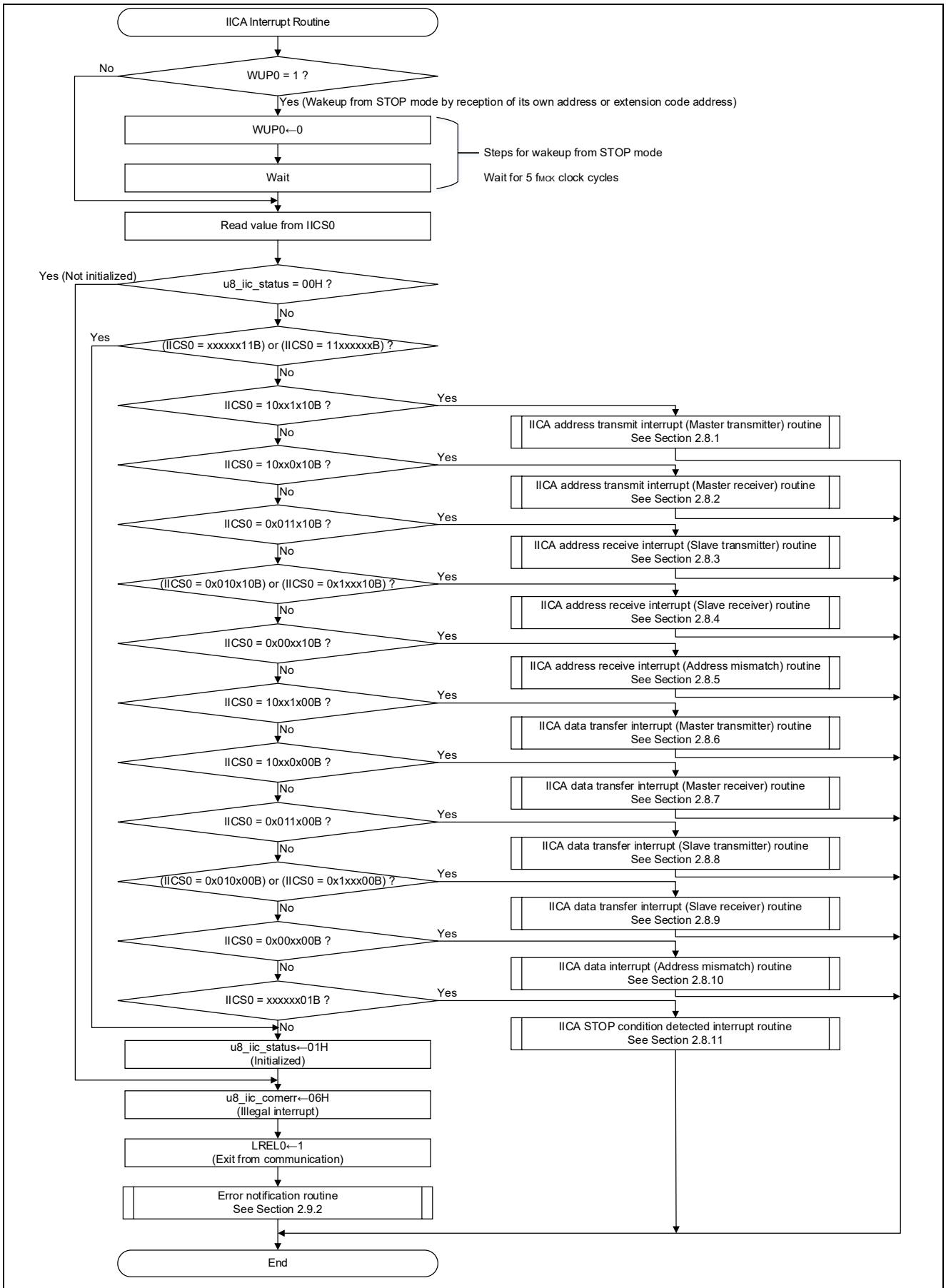
Note: Even though the STT0 bit of the IICCTL00 register is set, an IICA interrupt is not generated under the following conditions:
- SCLA0 line is HIGH and SDAA0 line is LOW
- SCLA0 line is LOW and SDAA0 line is HIGH
- SCLA0 line is LOW and SDAA0 line is LOW

Table 2-3. Interrupts into Which IICA Interrupt Is Broken Down

Bits of IICS0 Register								Interrupt
MSTS0	ALD0	EXC0	COI0	TRC0	ACKD0	STD0	SPD0	
1	0	x	x	1	x	1	0	IICA address transmit interrupt (Master transmitter)
1	0	x	x	0	x	1	0	IICA address transmit interrupt (Master receiver)
0	x	0	1	1	x	1	0	IICA address receive interrupt (Slave transmitter)
0	x	0	1	0	x	1	0	IICA address receive interrupt (Slave receiver)
0	x	1	x	x	x	1	0	
0	x	0	0	x	x	1	0	IICA address receive Interrupt (Address mismatch) [Note]
1	0	x	x	1	x	0	0	IICA data transfer interrupt (Master transmitter)
1	0	x	x	0	x	0	0	IICA data transfer interrupt (Master receiver)
0	x	0	1	1	x	0	0	IICA data transfer interrupt (Slave transmitter)
0	x	0	1	0	x	0	0	IICA data transfer interrupt (Slave receiver)
0	x	1	x	x	x	0	0	
0	x	0	0	x	x	0	0	IICA data transfer interrupt (Address mismatch)
x	x	x	x	x	x	0	1	IICA STOP condition detected interrupt

x: Any bit value

Note: The IICA address receive interrupt (Address mismatch) routine is invoked each time the device loses arbitration while requesting master operation or detects a RESTART condition together with a slave address that is not identical to its own address while operating as a slave.



Caution: "x" in this figure indicates any bit value.

Figure 2-13. IICA Interrupt Routine

2.8.1 IICA Address Transmit Interrupt (Master Transmitter)

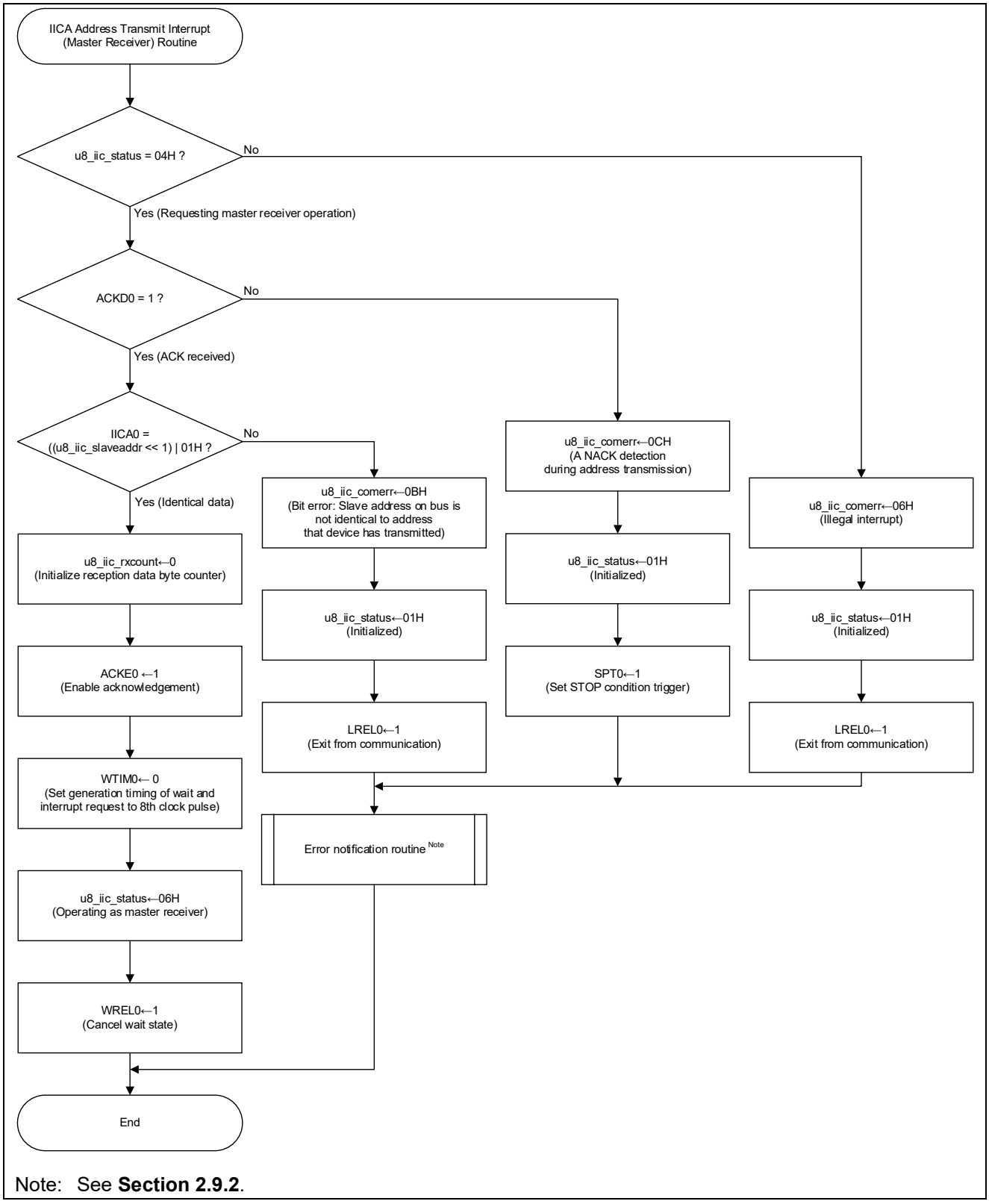
A procedure for handling the IICA address transmit interrupt (Master transmitter) is shown in **Figure 2-14**.

Note: See **Section 2.9.2**

Figure 2-14. IICA Address Transmit Interrupt (Master Transmitter) Routine

2.8.2 IICA Address Transmit Interrupt (Master Receiver)

A procedure for handling the IICA address transmit interrupt (Master receiver) is shown in **Figure 2-15**.

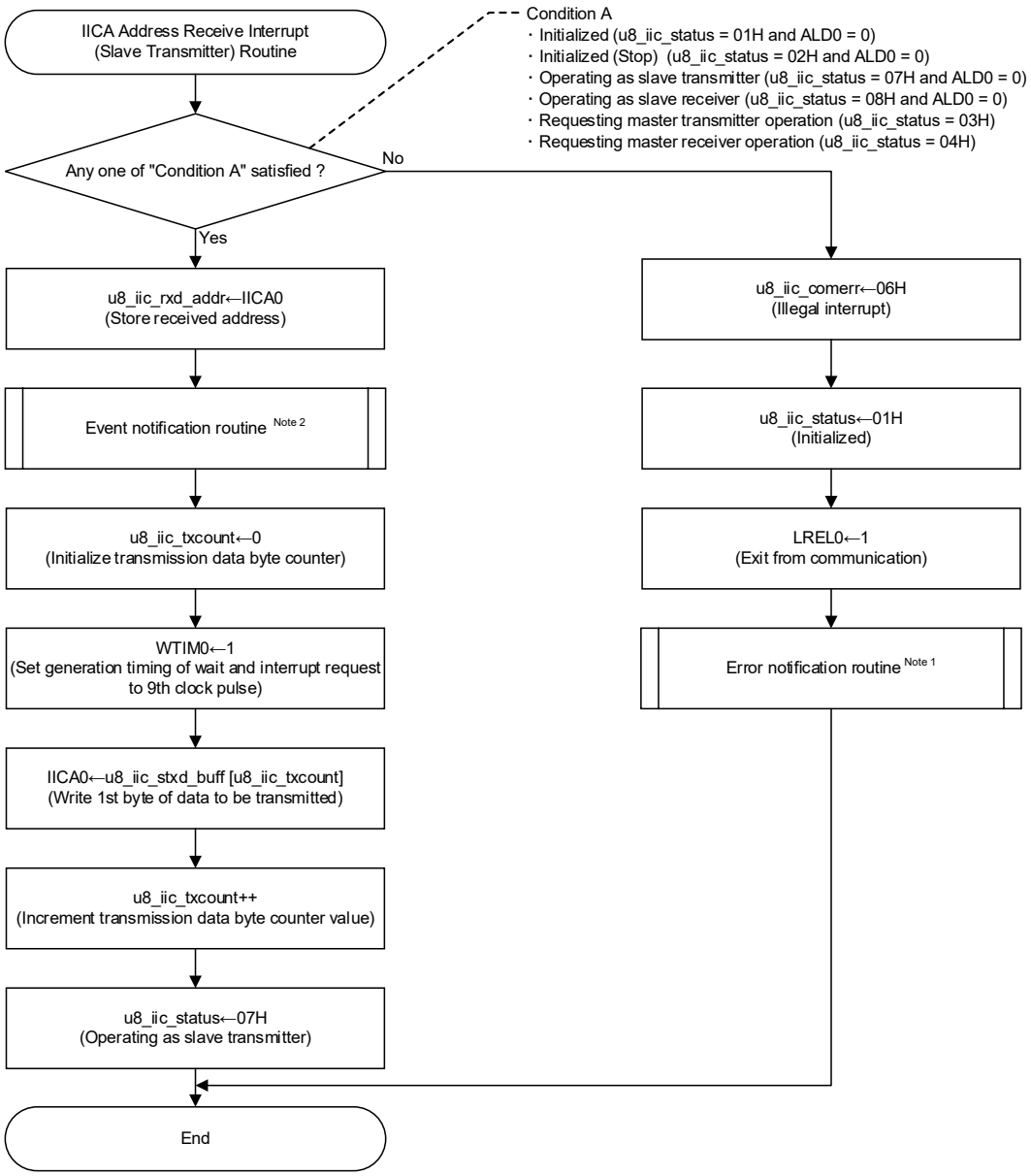


Note: See **Section 2.9.2**.

Figure 2-15. IICA Address Transmit Interrupt (Master Receiver) Routine

2.8.3 IICA Address Receive Interrupt (Slave Transmitter)

A procedure for handling the IICA address receive interrupt (Slave transmitter) is shown in **Figure 2-16**.



Notes: 1. See **Section 2.9.2**.

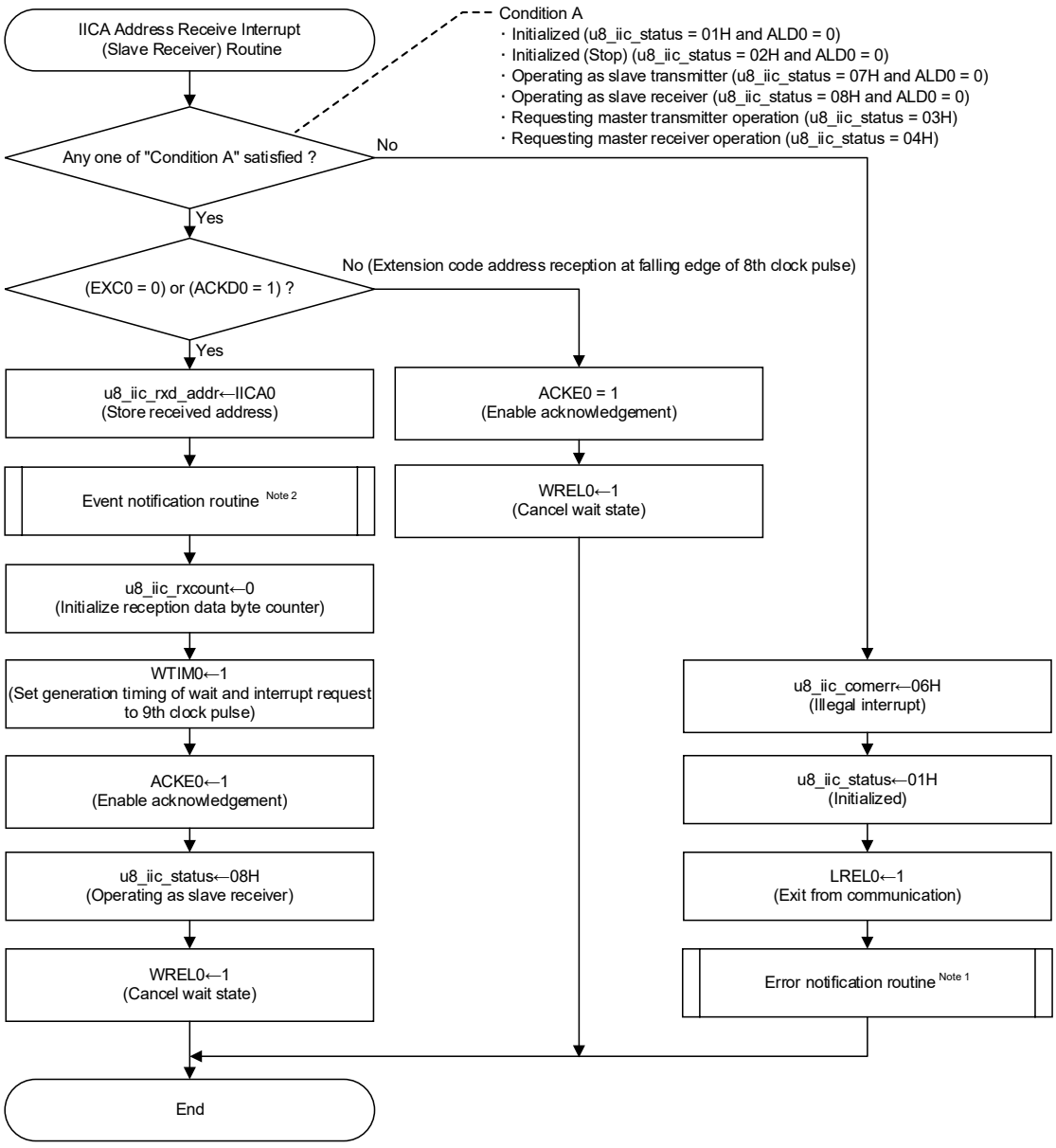
2. The event notification conditions are listed below. For details of the event notification routine, see **Section 2.9.1**.

Internal Status When Interrupt Occurs		Event Notification Conditions
u8_iic_status	ALD0	
Initialized (01H)	0	Slave transmitter operation start request
Initialized (Stop) (02H)	0	
Operating as a slave transmitter (07H)	0	Slave operation abortion and slave transmitter operation start request
Operating as a slave receiver (08H)	0	
Requesting master transmitter operation (03H)	Any value	Master operation request discard and slave transmitter operation start request
Requesting master receiver operation (04H)	Any value	

Figure 2-16. IICA Address Receive Interrupt (Slave Transmitter) Routine

2.8.4 IICA Address Receive Interrupt (Slave Receiver)

A procedure for handling the IICA address receive interrupt (Slave receiver) is shown in **Figure 2-17**.



Notes: 1. See **Section 2.9.2**.

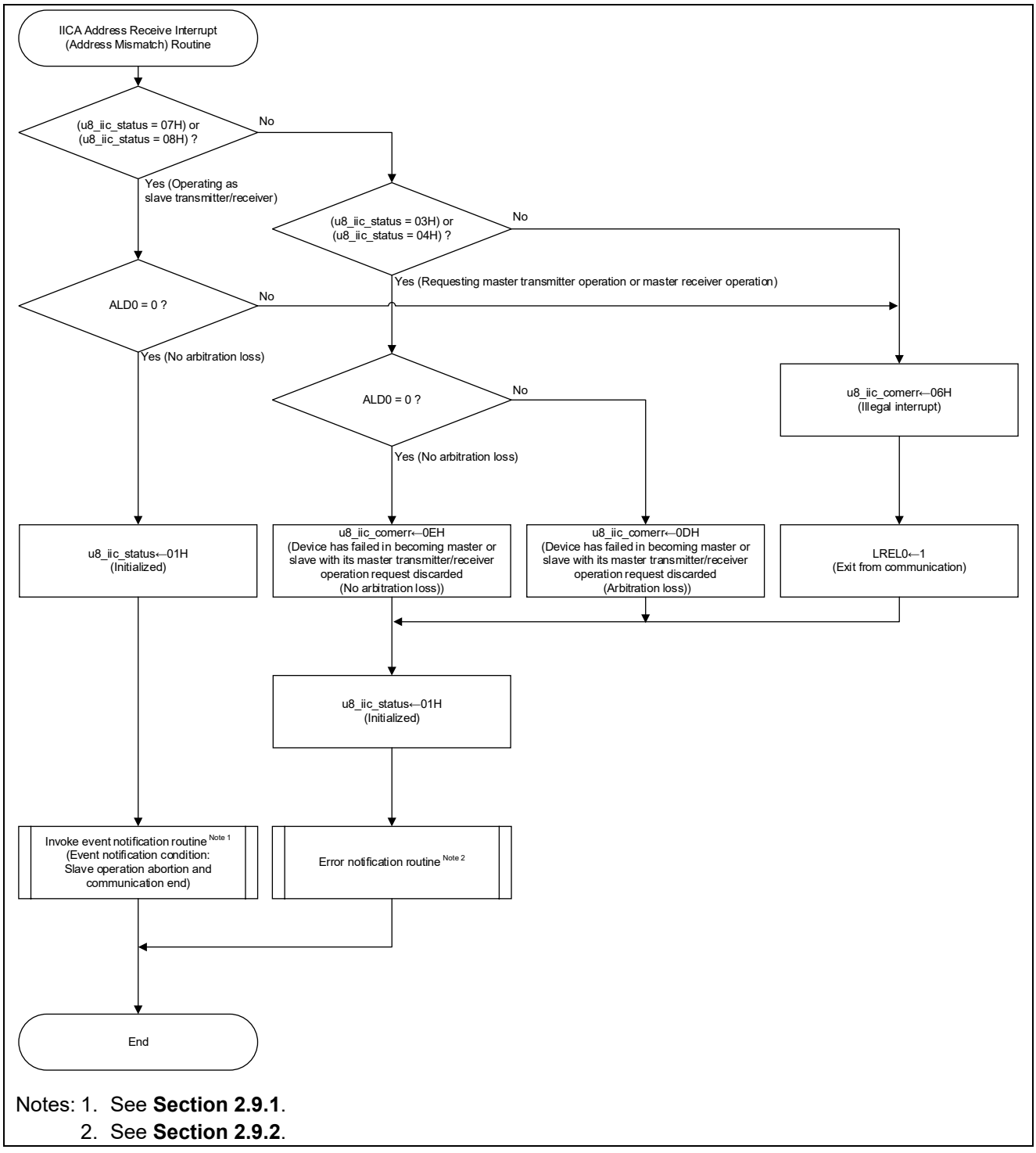
2. The event notification conditions are listed below. For details of the event notification routine, see **Section 2.9.1**.

Internal Status When Interrupt Occurs		Event Notification Conditions
u8_iic_status	ALD0	
Initialized (01H)	0	No event notification (Slave receiver operation is proceeding)
Initialized (Stop) (02H)	0	
Operating as a slave transmitter (07H)	0	Slave operation abortion and slave receiver operation start request
Operating as a slave receiver (08H)	0	
Requesting master transmitter operation (03H)	Any value	Master operation request discard and slave receiver operation start request
Requesting master receiver operation (04H)	Any value	

Figure 2-17. IICA Address Receive Interrupt (Slave Receiver) Routine

2.8.5 IICA Address Receive Interrupt (Address Mismatch)

A procedure for handling the IICA address receive interrupt (Address mismatch) is shown in **Figure 2-18**. This interrupt is generated only when the device is involved in I²C communication. This interrupt is not generated when the device is not involved in I²C communication.



Notes: 1. See **Section 2.9.1**.
2. See **Section 2.9.2**.

Figure 2-18. IICA Address Receive Interrupt (Address Mismatch) Routine

2.8.6 IICA Data Transfer Interrupt (Master Transmitter)

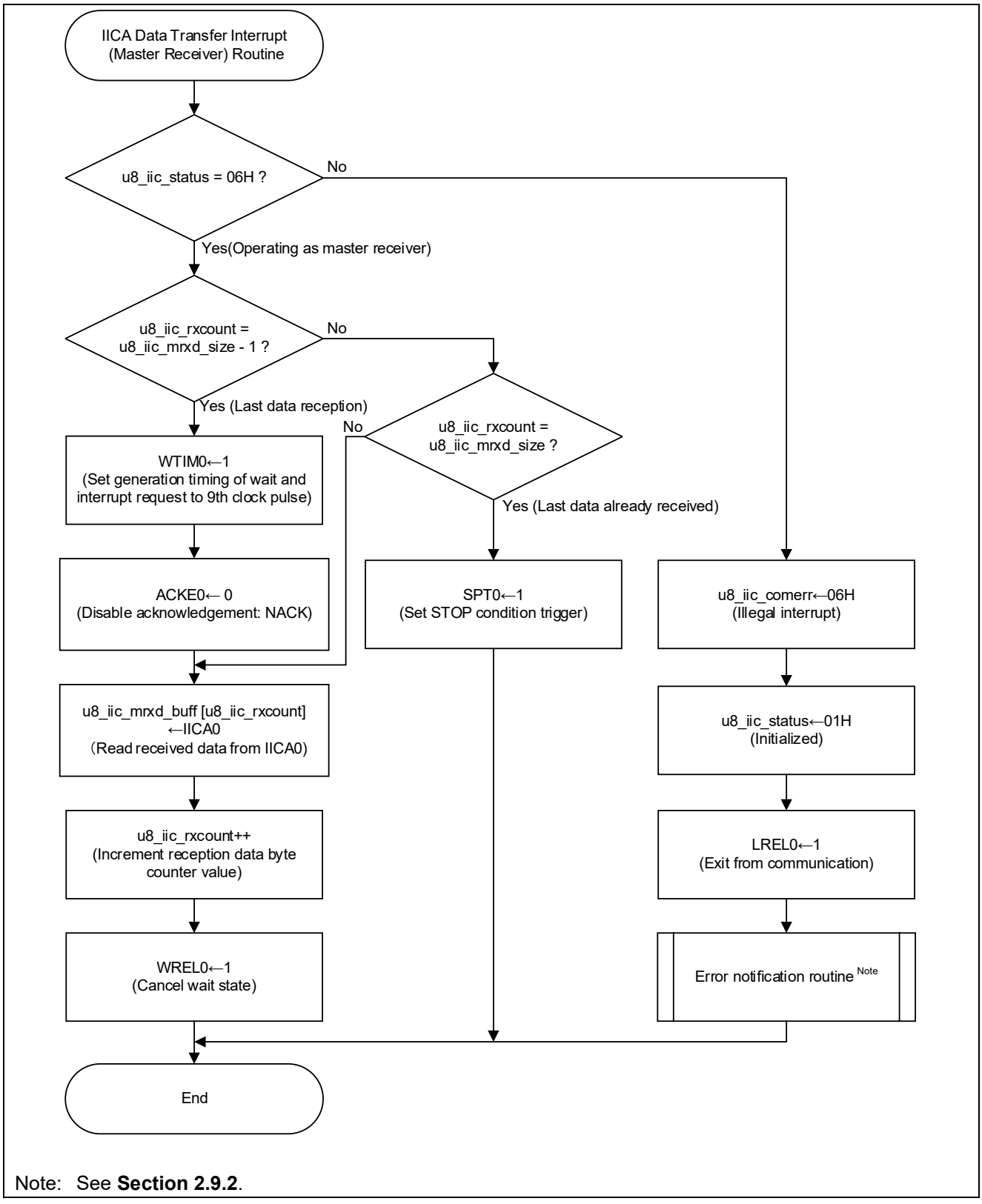
A procedure for handling the IICA data transfer interrupt (Master transmitter) is shown in **Figure 2-19**.

Note: See **Section 2.9.2**.

Figure 2-19. IICA Data Transfer Interrupt (Master Transmitter) Routine

2.8.7 IICA Data Transfer Interrupt (Master Receiver)

A procedure for handling the IICA data transfer interrupt (Master receiver) is shown in **Figure 2-20**. This application note provides an example in which a master receiver sends a NACK to signal to the slave transmitter that it has received the last data.

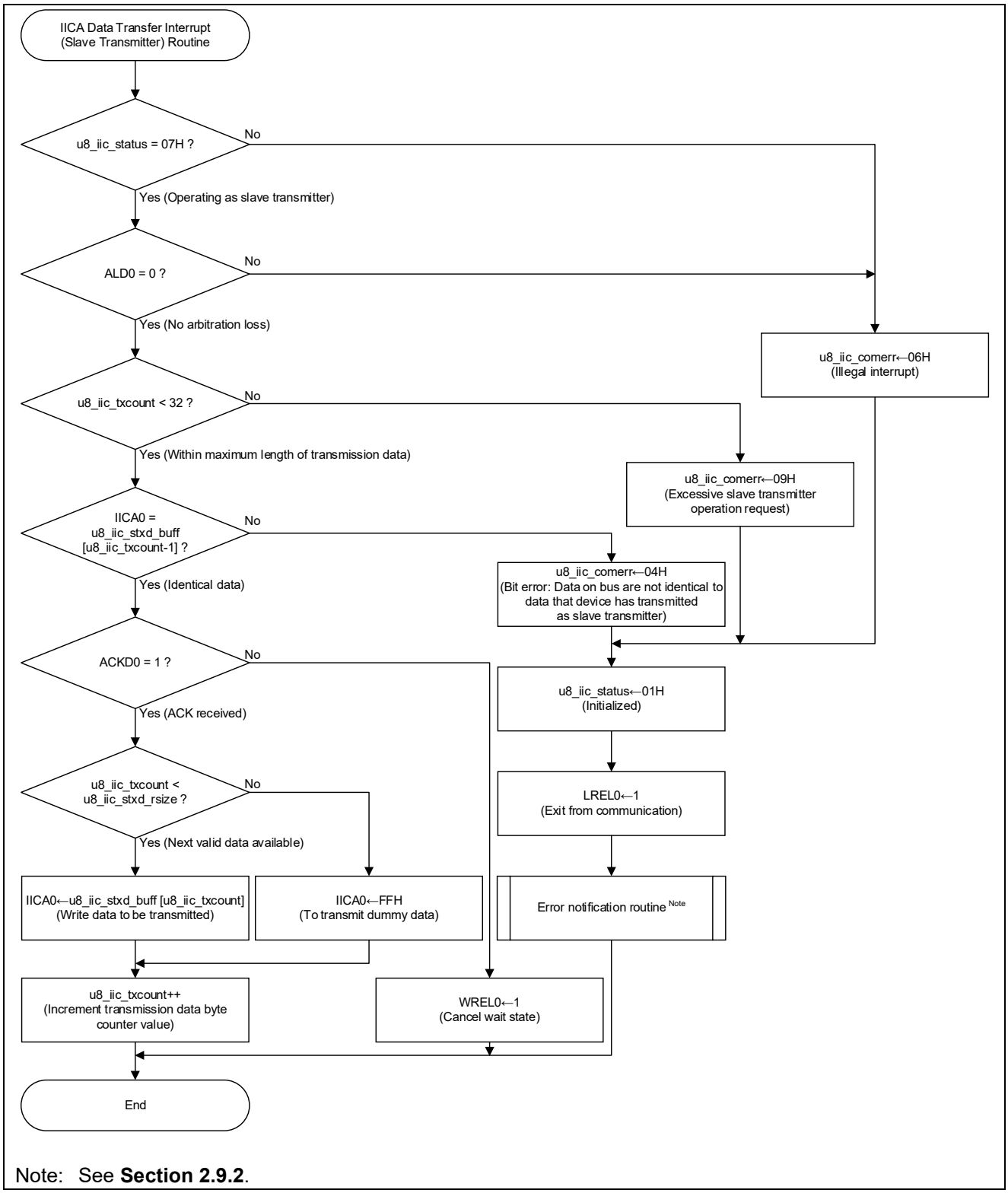


Note: See **Section 2.9.2**.

Figure 2-20. IICA Data Transfer Interrupt (Master Receiver) Routine

2.8.8 IICA Data Transfer Interrupt (Slave Transmitter)

A procedure for handling the IICA data transfer interrupt (Slave transmitter) is shown in **Figure 2-21**. This application note provides an example in which a slave transmitter keeps transmitting data until it receives a NACK from the master receiver.

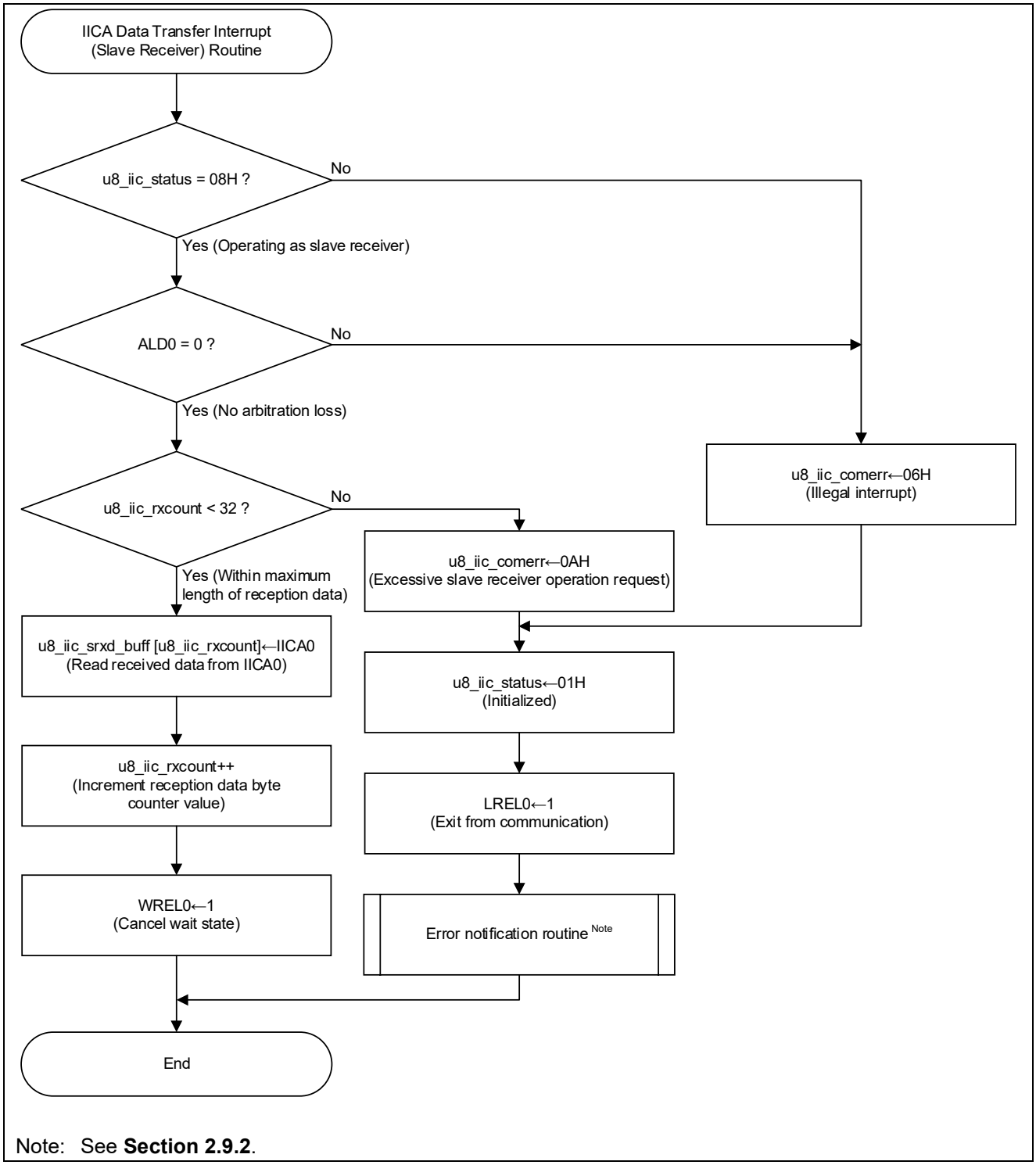


Note: See **Section 2.9.2**.

Figure 2-21. IICA Data Transfer Interrupt (Slave Transmitter) Routine

2.8.9 IICA Data Transfer Interrupt (Slave Receiver)

A procedure for handling the IICA data transfer interrupt (Slave receiver) is shown in **Figure 2-22**.

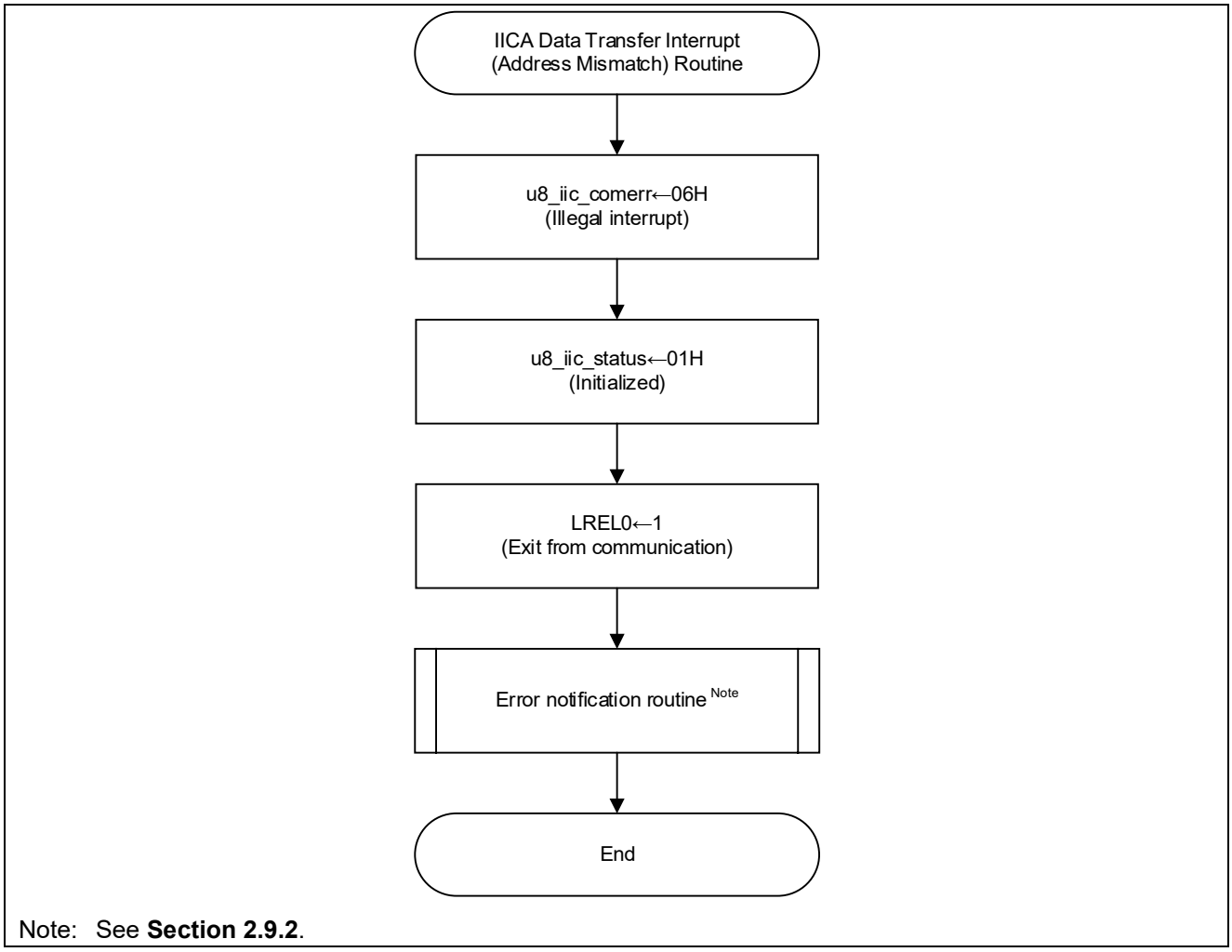


Note: See **Section 2.9.2**.

Figure 2-22. IICA Data Transfer Interrupt (Slave Receiver) Routine

2.8.10 IICA Data Transfer Interrupt (Address Mismatch)

A procedure for handling the IICA data transfer interrupt (Address mismatch) is shown in **Figure 2-23**. This interrupt is treated as an illegal interrupt in this application note since the device exits from communication when receiving an address that is not identical to its own address (inside the IICA address receive interrupt (Address mismatch) routine).

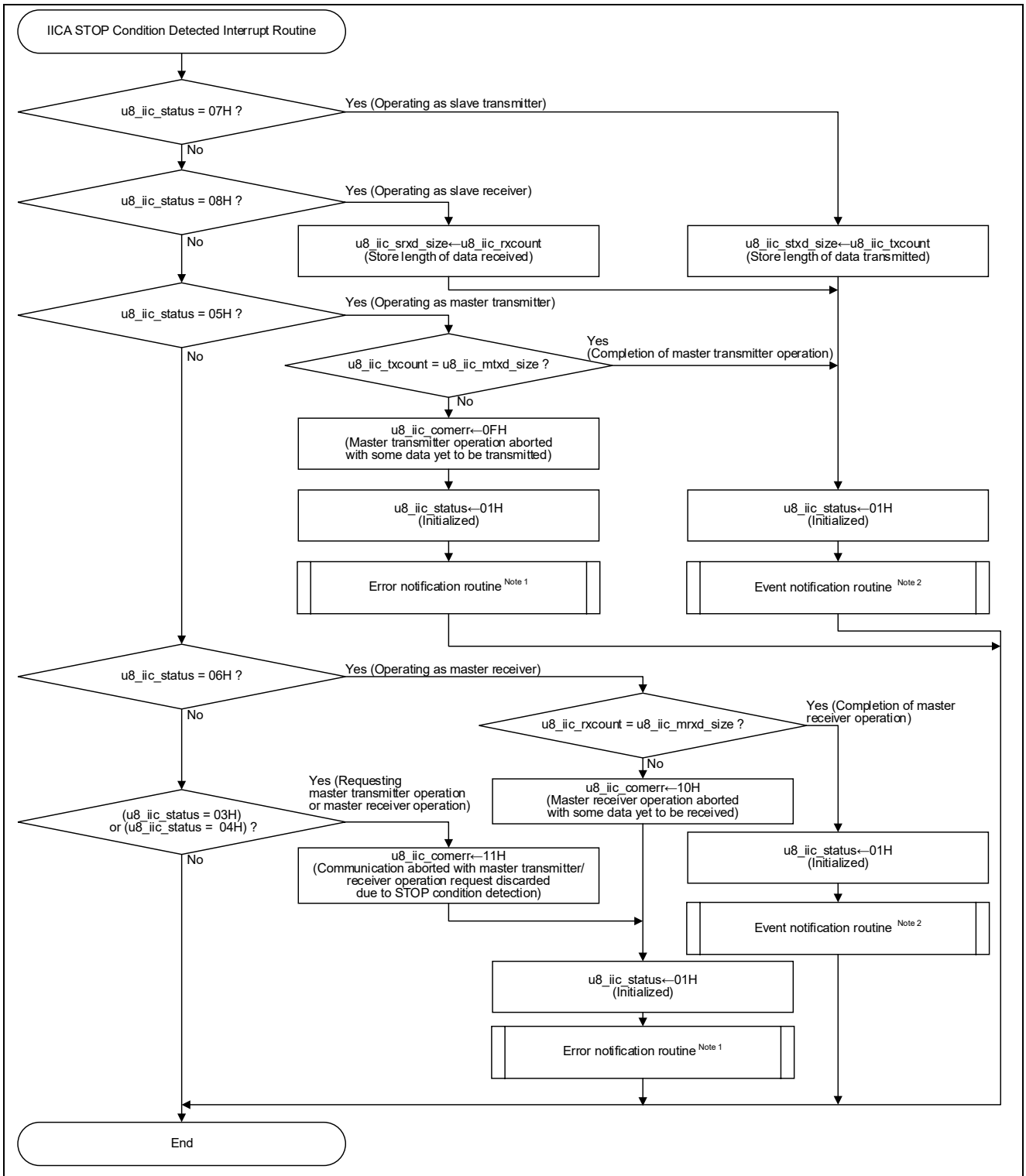


Note: See **Section 2.9.2**.

Figure 2-23. IICA Data Transfer Interrupt (Address Mismatch) Routine

2.8.11 IICA STOP Condition Detected Interrupt

The IICA STOP condition detected interrupt routine invokes either the event notification routine or error notification routine depending on the situation. **Figure 2-24** shows a procedure for handling this interrupt and **Table 2-4** lists the notification conditions notified based on the internal status.



Notes: 1. See **Table 2-4** and **Section 2.9.2**.
2. See **Table 2-4** and **Section 2.9.1**.

Figure 2-24. IICA STOP Condition Detected Interrupt Routine

Table 2-4. Notification Conditions Notified from IICA STOP Condition Detected Interrupt Routine

Internal Status When Interrupt Occurs		Notification Type	Notification Conditions
u8_iic_status	Data Transmitted/Received		
Operating as a master transmitter (05H)	All data have been transmitted (u8_iic_txcount = u8_iic_mtxd_size)	Event notification	Completion of master transmitter operation
	Data yet to be transmitted exist (u8_iic_txcount < u8_iic_mtxd_size)	Error notification	Master transmitter operation aborted with some data yet to be transmitted (u8_iic_comerr←0FH)
Operating as a master receiver (06H)	All data have been received (u8_iic_rxcount = u8_iic_mrxd_size)	Event notification	Completion of master receiver operation
	Data yet to be received exist (u8_iic_rxcount < u8_iic_mrxd_size)	Error notification	Master receiver operation aborted with some data yet to be received (u8_iic_comerr←10H)
Operating as a slave transmitter (07H)	-	Event notification	Completion of slave transmitter operation
Operating as a slave receiver (08H)	-	Event notification	Completion of slave receiver operation
Requesting master transmitter operation (03H)	-	Error notification	Communication aborted with master transmitter/receiver operation request discarded due to a STOP condition detection (u8_iic_comerr←11H)
Requesting master receiver operation (04H)	-		
Other than those above	-	-	No notification

2.9 Notification Routines

The flows of the notification routines are shown in **Figure 2-25** and **Figure 2-26**. However, customers are required to modify these notification routines for their system. In addition, the processing routines should not be invoked inside these notification routines but their main function because these notification routines are basically invoked inside the IICA interrupt routine. Requests can be set inside these notification routines.

2.9.1 Event Notification

The flow of the event notification routine is shown in **Figure 2-25**. The event notification conditions and examples on how to handle them are shown in **Table 2-5**.

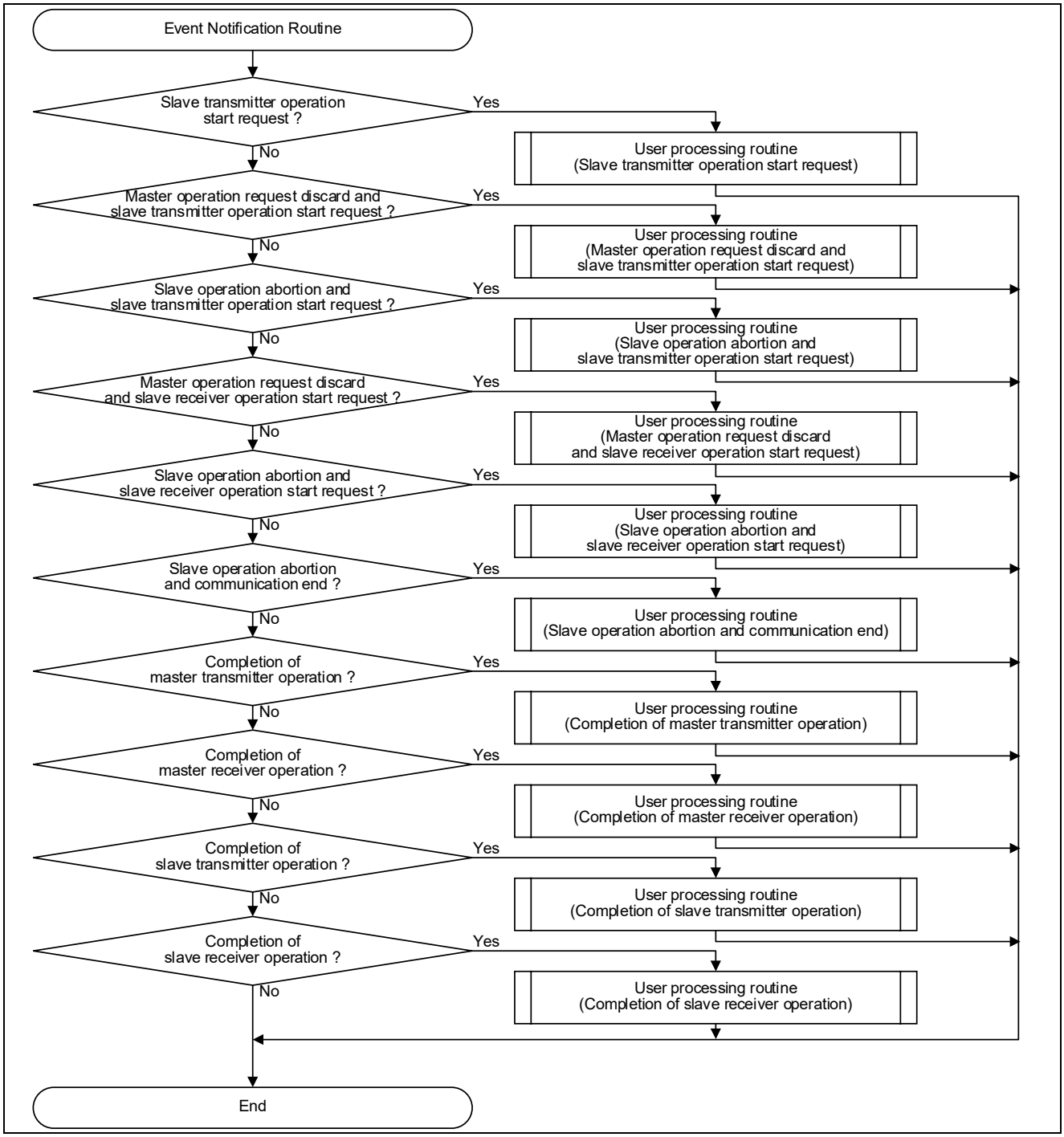


Figure 2-25. Event Notification Routine

Table 2-5. Event Notification Conditions and Examples on How to Handle Them

Event Notification Conditions	Situation	Examples on How to Handle Event Notification Conditions
Slave transmitter operation start request	The device has been addressed as a slave transmitter	Write data to be transmitted and its length into u8_iic_stxd_buff [32] and u8_iic_stxd_rsize, respectively. The received address (including R/$\overline{W}$) can be read from u8_iic_rxd_addr.
Master operation request discard and slave transmitter operation start request	The device has been addressed as a slave transmitter, so that its master operation request has been discarded	Write data to be transmitted and its length into u8_iic_stxd_buff [32] and u8_iic_stxd_rsize, respectively. The received address (including R/$\overline{W}$) can be read from u8_iic_rxd_addr. If the discarded master operation request needs to be sent again, wait until the reception finishes, examine the I^2C bus status and IICA multi-master communication status,[Note] and then invoke the IICA master transmitter/receiver operation start procedure.
Slave operation abortion and slave transmitter operation start request	The device has been addressed as a slave transmitter while operating as a slave	Write data to be transmitted and its length into u8_iic_stxd_buff [32] and u8_iic_stxd_rsize, respectively. The received address (including R/$\overline{W}$) can be read from u8_iic_rxd_addr.
Master operation request discard and slave receiver operation start request	The device has been addressed as a slave receiver with its own address or an extension code address, so that its master operation request has been discarded	The slave receiver operation is proceeding. If the discarded master operation request needs to be sent again, wait until the reception finishes, examine the I^2C bus status and IICA multi-master communication status,[Note] and then invoke the IICA master transmitter/receiver operation start procedure.
Slave operation abortion and slave receiver operation start request	The device has been addressed as a slave receiver while operating as a slave	The device can restart the slave operation if it receives the aborted slave request again from the master after the reception finishes.
Slave operation abortion and communication end	The device has received an address that is not identical to its own address or not an extension code address while operating as a slave, so that the device has exited from communication	The device can restart the slave operation if it receives the aborted slave request again from the master after the communication in which the device is not involved finishes.
Completion of master transmitter operation	The device has succeeded in transmitting all data as a master transmitter	Invoke the necessary procedure.
Completion of master receiver operation	The device has succeeded in receiving all data as a master receiver	Invoke the necessary procedure.
Completion of slave transmitter operation	A STOP condition has been detected while operating as a slave transmitter	The length of data that the device has transmitted can be read from u8_iic_stxd_size.
Completion of slave receiver operation	A STOP condition has been detected while operating as a slave receiver	The data that the device has received and its length can be read from u8_iic_srxd_buff[32] and u8_iic_srxd_size, respectively. Read the received data and invoke the necessary procedure.

Note: Using the IICA status information API (see **Section 2.7**), make sure that the I^2C bus is released, the SCLA0 and SDAA0 are high, and the IICA multi-master communication status is "Initialized".

2.9.2 Error Notification

The flow of the error notification routine is shown in **Figure 2-26**. Customers are required to handle each error condition for their products. Examples on how to handle error conditions can be found in **Section 2.10**.

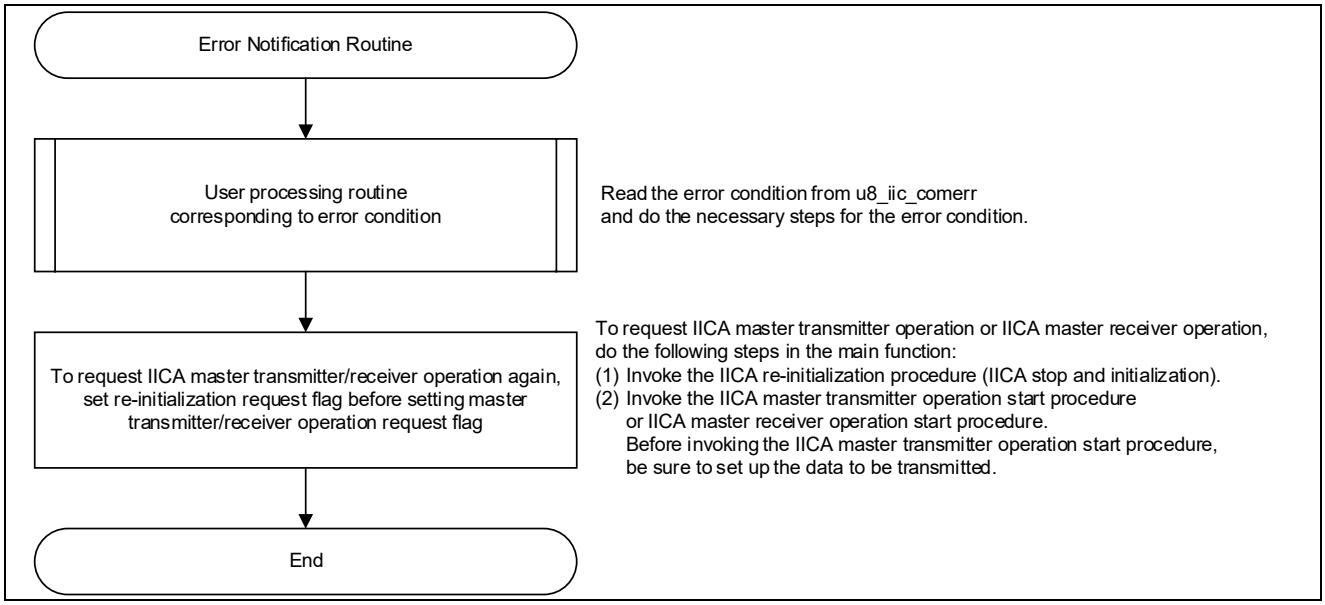


Figure 2-26. Error Notification Routine

2.10 Error Handling Examples

If an error condition such as a communication error is detected, customers should read the variable that holds the error condition (u8_iic_comerr) inside the error notification routine, and then invoke their error handling routine corresponding to the error condition inside their main function.

Table 2-6. Error Conditions and Examples on How to Handle Them

u8_iic_comerr	Error Conditions	Examples on How to Handle Error Conditions
01H	Master transmitter/receiver operation requested in any status except "Initialized"	Firstly, invoke the IICA initialization procedure after the IICA stop procedure. Secondly, examine the I²C bus status and IICA multi-master communication status.Note 1 Lastly, invoke the IICA master transmitter/receiver operation start procedure again.
02H	Command error (Master transmitter/receiver operation requested with an inappropriate parameter)	Setting the parameters to the appropriate values, invoke the IICA master transmitter/receiver operation start procedure again.
03H	Bit error (Data on the bus are not identical to the data that the device has transmitted as a master transmitter)	Examining the I²C bus status and IICA multi-master communication status,Note 1 invoke the IICA master transmitter operation start procedure again.
04H	Bit error (Data on the bus are not identical to the data that the device has transmitted as a slave transmitter)	Invoke the IICA initialization procedure after the IICA stop procedure.
05H	A NACK detection during data transmission (Master transmitter)	Examining the I²C bus status and IICA multi-master communication status,Note 1 invoke the IICA master transmitter operation start procedure again.
06H	Illegal interrupt	Invoke the IICA initialization procedure after the IICA stop procedure.
07H	Master transmitter operation request discarded due to bus stalls	Examining the I²C bus status and IICA multi-master communication status,Note 1 invoke the IICA master transmitter operation start procedure again.
08H	Master receiver operation request discarded due to bus stalls	Examining the I²C bus status and IICA multi-master communication status,Note 1 invoke the IICA master receiver operation start procedure again.
09H	Excessive slave transmitter operation request (The device has been requested to transmit data as a slave transmitter beyond the maximum length of slave transmission data Note 2)	Invoke the IICA initialization procedure after the IICA stop procedure.
0AH	Excessive slave receiver operation request (The device has been requested to receive data as a slave receiver beyond the maximum length of slave reception data Note 2)	Invoke the IICA initialization procedure after the IICA stop procedure.
0BH	Bit error (Slave address on the bus is not identical to the address that the device has transmitted)	Examining the I²C bus status and IICA multi-master communication status,Note 1 invoke the IICA master transmitter operation start procedure again.
0CH	A NACK detection during address transmission (Master transmitter/receiver)	Firstly, make sure that the address the device has sent is correct. Secondly, examine the I²C bus status and IICA multi-master communication status.Note 1 Lastly, invoke the IICA master transmitter/receiver operation start procedure again.
0DH	The device has failed in becoming a master or slave with its master transmitter/receiver operation request discarded (Arbitration loss)	Examining the I²C bus status and IICA multi-master communication status,Note 1 invoke the IICA master transmitter/receiver operation start procedure again.
0EH	The device has failed in becoming a master or slave with its master transmitter/receiver operation request discarded (No arbitration loss)	Examining the I²C bus status and IICA multi-master communication status,Note 1 invoke the IICA master transmitter/receiver operation start procedure again.
0FH	Master transmitter operation aborted with some data yet to be transmitted	Examining the I²C bus status and IICA multi-master communication status,Note 1 invoke the IICA master transmitter operation start procedure again.
10H	Master receiver operation aborted with some data yet to be received	Examining the I²C bus status and IICA multi-master communication status,Note 1 invoke the IICA master receiver operation start procedure again.
11H	Communication aborted with master transmitter/receiver operation request discarded due to a STOP condition detection	Examining the I²C bus status and IICA multi-master communication status,Note 1 invoke the IICA master transmitter/receiver operation start procedure again.
12H	IICA initialization failure	Invoke the IICA initialization procedure again.

Notes: 1. Using the IICA status information API (see **Section 2.7**), make sure that the I²C bus is released, the SCLA0 and SDAA0 are high, and the IICA multi-master communication status is "Initialized".
2. This application note provides an example in which the maximum length of slave transmission/reception data is specified as 32 bytes.

3. References

Documents referenced in this application note are shown below. When referring to these documents, make sure to obtain the latest version of each document from Renesas Electronics website.

- RL78/ F13, F14 User's Manual: Hardware Rev. 2.10
- RL78/ F15 User's Manual: Hardware Rev. 1.00
- RL78/ F23, F24 User's Manual: Hardware Rev. 1.00
- RL78 Family User's Manual: Software Rev. 2.30

Revision History

Rev.	Date	Description	
		Page	Summary
1.00	2019.02.28	-	First edition issued.
1.10	2022.09.30	-	RL78/F23, F24 are added for the target devices.

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 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.
5. 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.
6. 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.
7. 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.
8. 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.
9. 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.
10. 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.
11. This document shall not be reprinted, 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.

(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 November 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/.