R32C/100 Series

I2C-bus Interface Using UARTi Special Mode 1 (Slave Transmit/Receive)

REJ05B1396-0101
Rev.1.01
Mar 10, 2011

1. Abstract

This document describes the slave transmit/receive processes in I2C-bus interface slave communication using the R32C/100 Series serial interface (UART2) special mode 1 (I2C mode).
Seven channels (UART0 to UART6) can be used in special mode 1 in the R32C/118 Group.
If channels other than UART0 to UART6 are used, refer to the hardware user's manual and modify the registers associated with UARTi (i = 0 to 6).

2. Introduction

The application example described in this document applies to the following microcomputer (MCU) and parameter:

- MCU: R32C/118 Group
- XIN Clock: 16 MHz

This application note can be used with other R32C/100 Series MCUs which have the same special function registers (SFRs) as the above group. Check the user's manual for any modifications to functions. Careful evaluation is recommended before using the program described in this application note.

3. Application Example

3.1 Program Outline

I2C-bus interface slave communication (slave transmission/reception) using UART2 special mode 1 is processed in the application example. A maximum of 255 bytes of data can be transmitted/received.

The transmission and reception procedures conform to the I2C-bus communication protocol when used under the following conditions:

- Slave address: 7 bits
- Standard-mode and Fast-mode are supported
- Communication data length: 1 to 255 bytes (not including the slave address)
- Restart condition is not supported

Figure 3.1 shows the Communication Format, Figure 3.2 shows the Block Diagram, Figure 3.3 shows the Outline Flowchart, and Figure 3.4 to Figure 3.6 show Timing Diagrams.

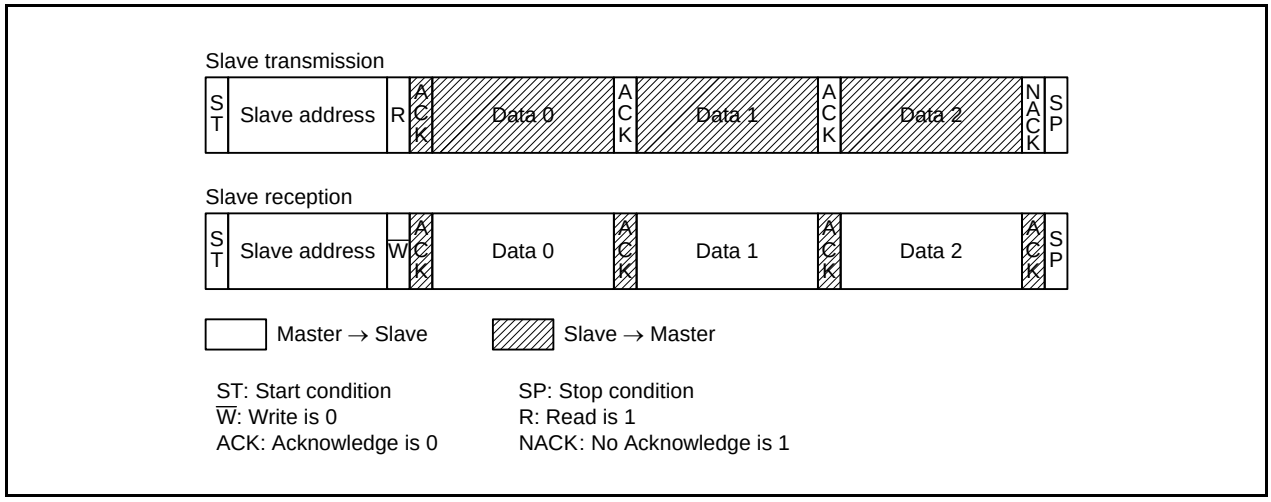


Figure 3.1 Communication Format

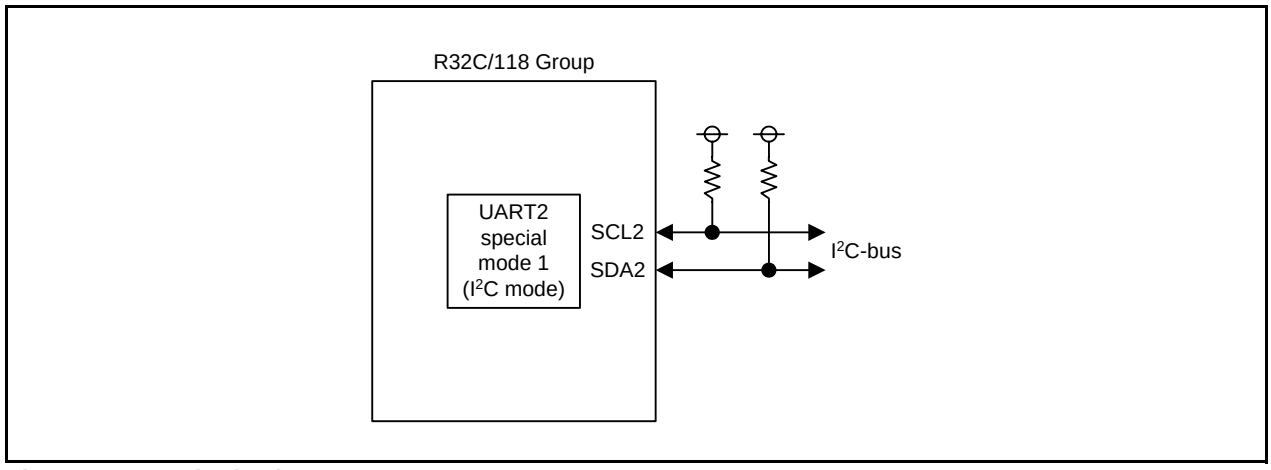


Figure 3.2 Block Diagram

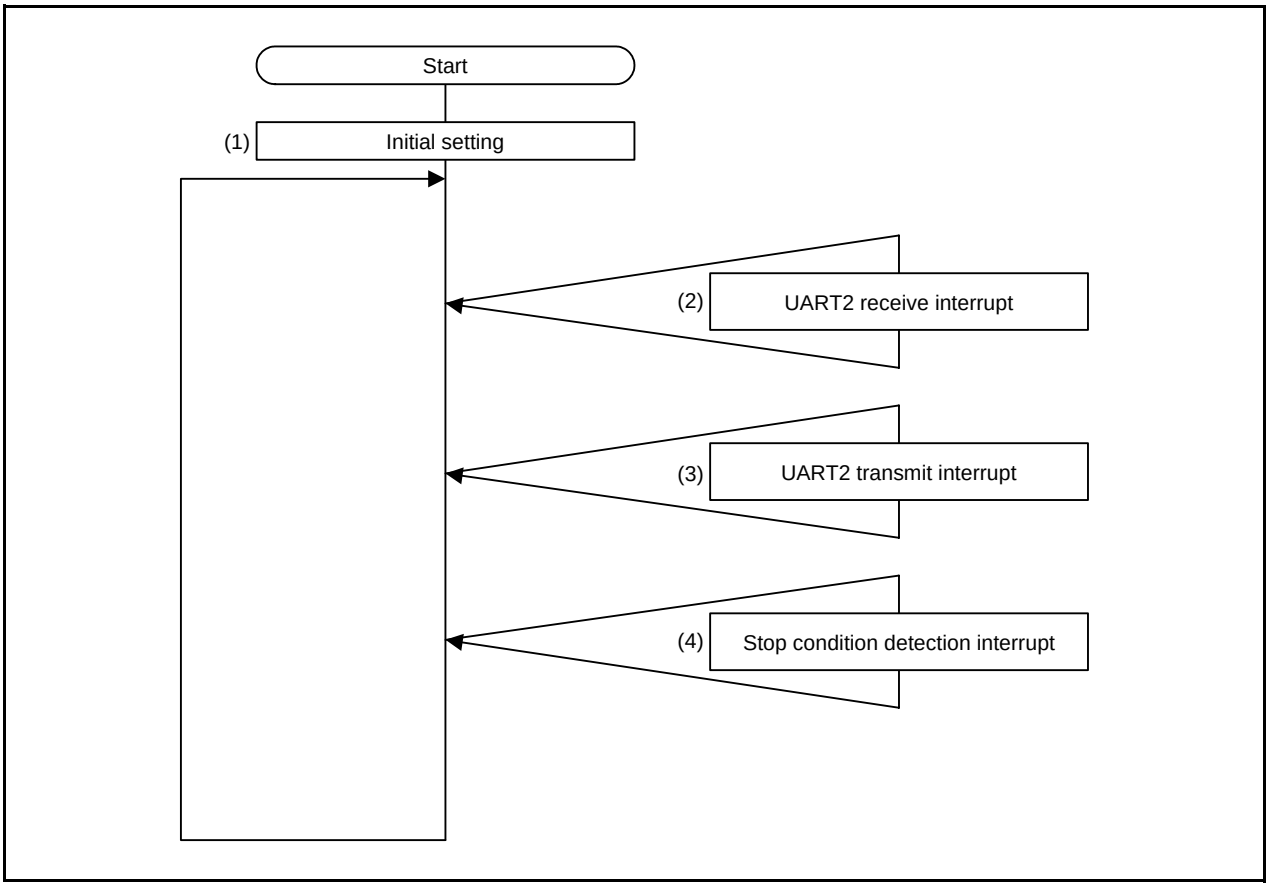


Figure 3.3 Outline Flowchart

The numbers in Figure 3.3 correspond to the numbers indicated in the program processing in the operating timing charts in Figure 3.4 to Figure 3.6.

(1) Initial setting
Initialize the system clock, UART2 associated SFRs, and variables used.

(2) UART2 receive interrupt
When a slave address is received, a UART2 receive interrupt is generated at the falling edge of the eighth bit of the SCL clock. The slave address is determined after reading the U2RB register.
When the slave address is matched:
- Generate an ACK and set the SCL2 pin to low hold at the ninth bit.
- Enable the stop condition detection interrupt and UART2 transmit interrupt. Disable the UART2 receive interrupt.
- Set transmit/receive data to the U2TB register. (1)

When the slave address is not matched:
- Generate a NACK.

After the above processing, release the SCL2 pin low hold at the eighth bit.

(3) UART2 transmit interrupt
A UART2 transmit interrupt is generated at the falling edge of the ninth bit of the SCL clock. When the first byte (slave address) is received, ACK output set in the UART2 receive interrupt handling is released. When transmitting, determine the ACK/NACK and set the next byte transmit data. When receiving, store the receive data and set ACK for the next byte.

(4) Stop condition detection interrupt
When a stop condition is detected, an interrupt is generated. SFR values which changed in mid-communication are returned to their initial settings. Disable the stop condition detection interrupt and UART2 transmit interrupt. Enable the UART2 receive interrupt.

Note:
1. When the TXEPT bit in the UiC0 register is 0 (data in the transmit register) in slave mode, write data to the UiTB register.
If no data exists in the transmit buffer register, the TXEPT bit becomes 1 at the rising edge of the ninth bit of the SCLi clock.
The following procedure should be met the condition above.
- When receiving the first byte (slave address):
  (1) Set the second byte data to the UiTB register in the receive interrupt.
  (2) Set the third byte data to the UiTB register in the transmit interrupt.
- After the first byte, set fourth or later byte data every transmit interrupt is requested.

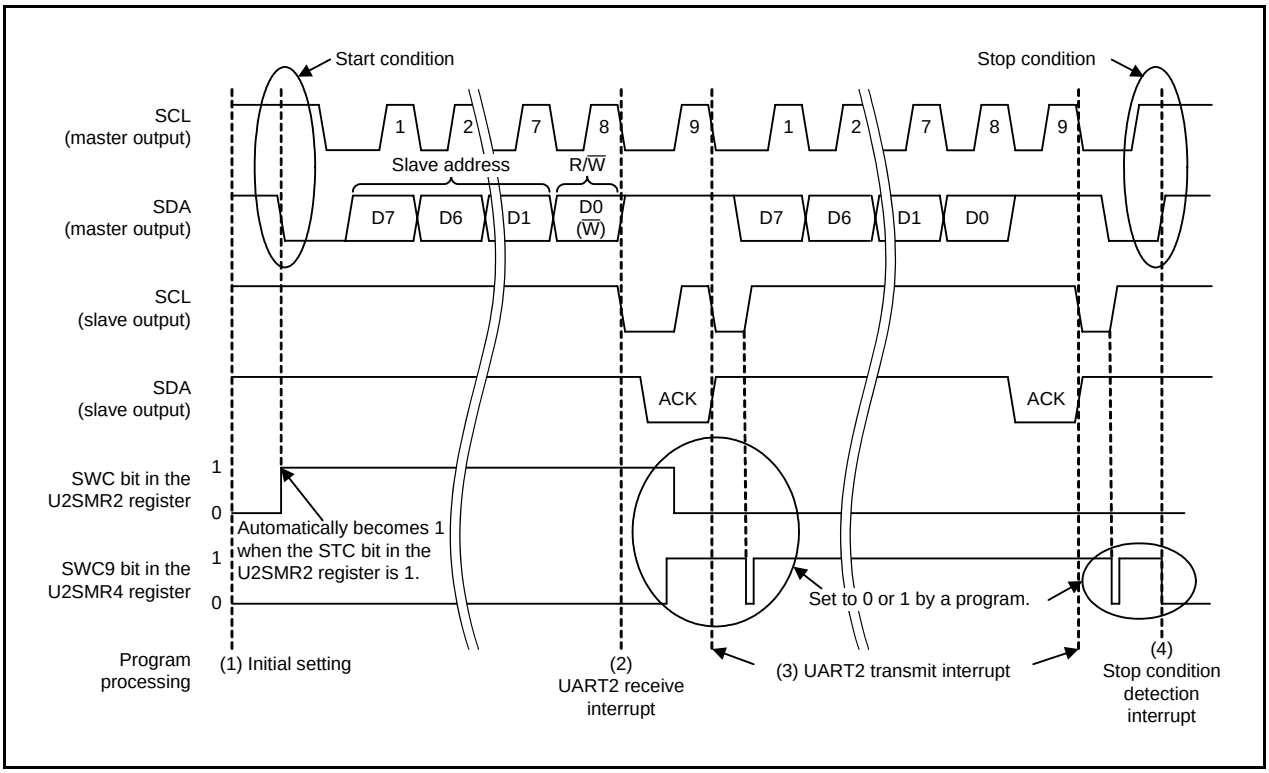


Figure 3.4 Slave Receive Timing

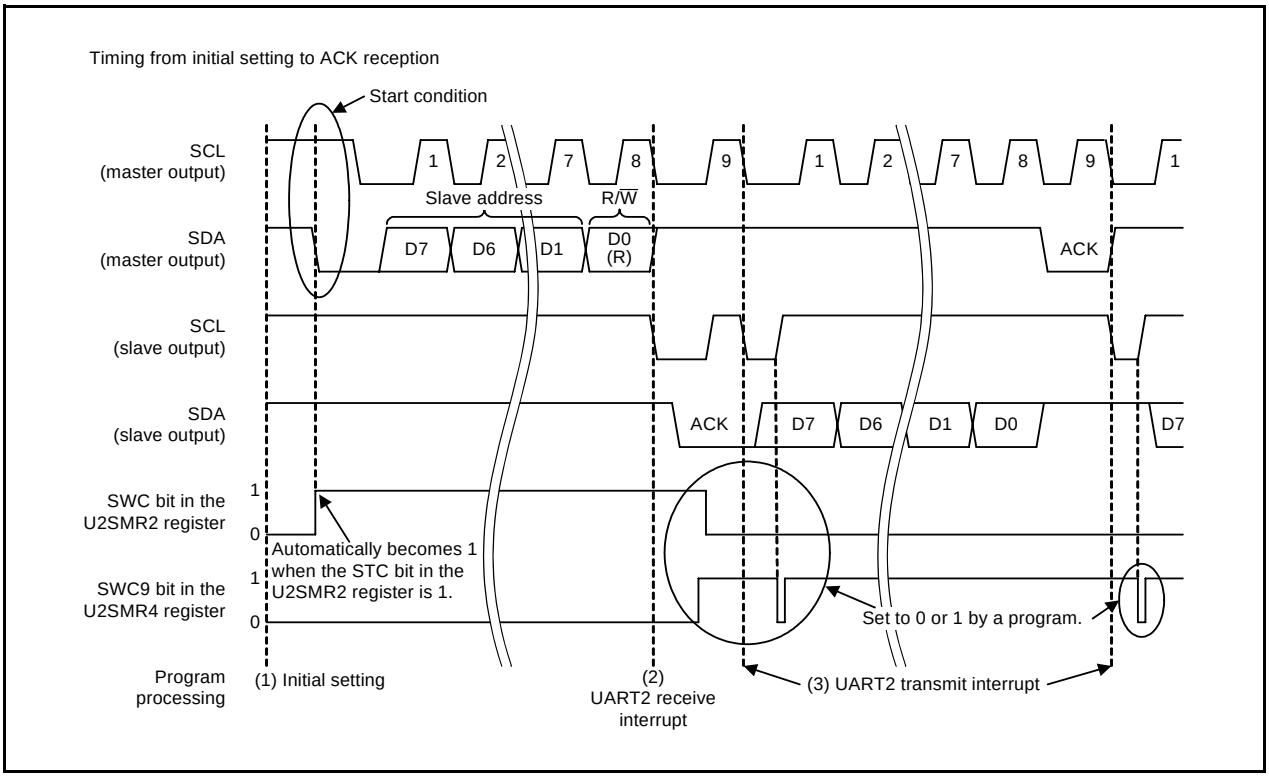


Figure 3.5 Slave Transmit Timing (1)

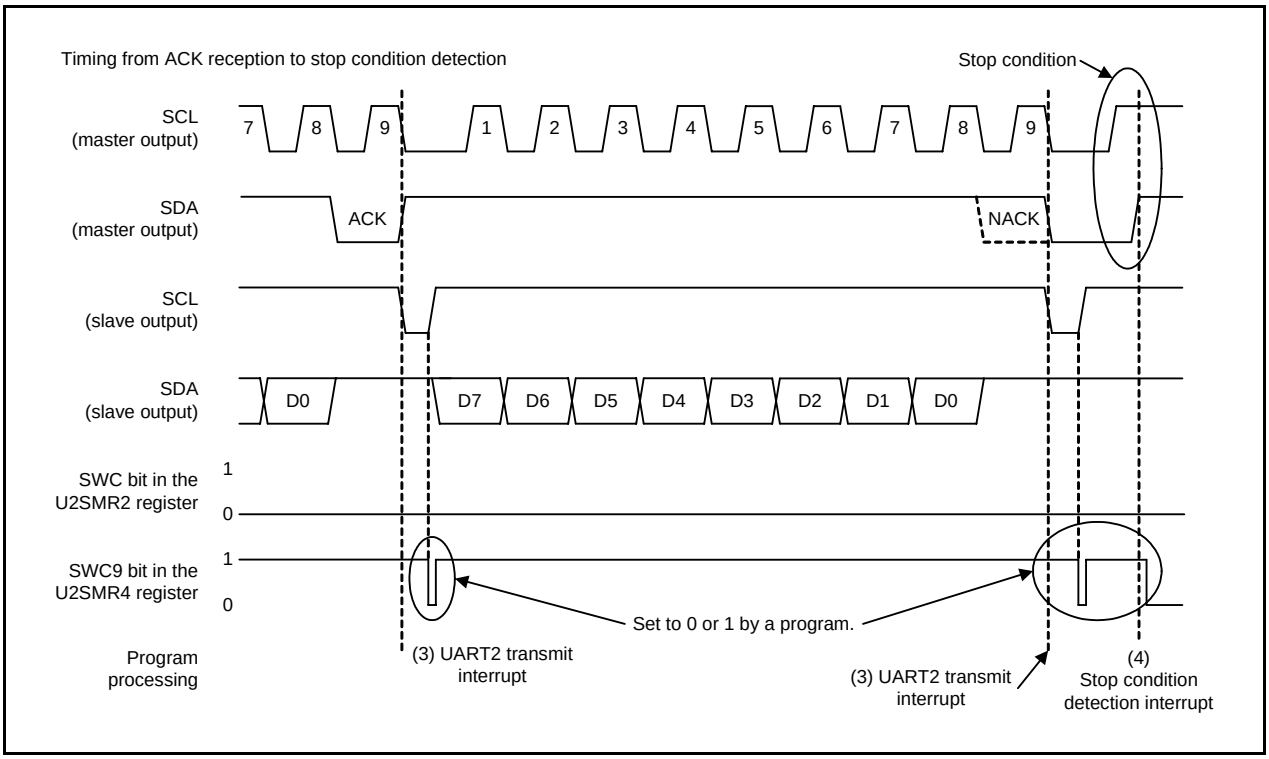


Figure 3.6 Slave Transmit Timing (2)

3.1.1 Peripheral Functions

Serial interface (UART2) special mode 1 (I2C mode) is used under the following setting conditions:

- I2C mode is used.
- Transfer clock is external clock.
- f1 is used as U2BRG count source.
- SDA2 and SCL2 pins are N-channel open drain output.
- Transfer format is MSB first.
- Transmission completed (TXEPT is 1) is selected as the UART2 transmit interrupt source.
- Clock phase setting is clock delay.
- Seven to eight cycles of the U2BRG count source is selected as SDA2 digital delay value.
- UART2 auto initialize function is used.
- SCL2 wait auto insert function is used.
- SCL2 wait output function 2 is not used.
- SCL2 wait auto insert function 3 is used.
- SDA2 output stop function is used.
- Start condition detection interrupt is not used.
- Stop condition detection interrupt is used.
- UART2 transmit interrupt is used.
- UART2 receive interrupt is used.
- PLL clock is 100 MHz.
- Base clock is 50 MHz.
- CPU clock is 50 MHz.
- Peripheral bus clock is 25 MHz.
- Peripheral clock source is 25 MHz.

Table 3.1 Pins Used and Their Function

Pin	I/O	Function
P7_1/SCL2	I/O	I2C mode clock I/O pin
P7_0/SDA2	I/O	I2C mode data I/O pin

3.1.2 Notes on Using the Attached Sample Program

Note the following when using the program included with this application note:

(1) Do not use multiple interrupts.

(2) The size of the receive buffer and the transmit buffer are set to 255 bytes. The buffer size is defined by the BUFSIZE macro (1 to 255 bytes). When the number of transmit/receive bytes exceeds the size of the buffer, the slave disregards the communication. Disable the UART2 transmit interrupt, and release pins SCL2 and SDA2.

(3) After the master generates a stop condition, when the slave processing time (1) has passed, start the next transmit/receive operation (start condition is generated/generate a start condition).

Note:

1. The slave processing time indicates the time between detecting a stop condition and enabling I2C mode in the main processing, and is dependent on the processing of the user program. The maximum processing time for this sample program is approximately 100 µs.

3.2 Memory

Table 3.2 Memory

Memory	Size	Remarks
ROM	657 bytes	In the iic.c module
RAM	6 bytes	In the iic.c module
Maximum user stack	28 bytes	
Maximum interrupt stack	64 bytes	

Usage memory size varies depending on C compiler version and compile options. The above applies under the following conditions:
C compiler: R32C/100 Series C Compiler V.1.02 Release 01
C compile option: -c -finfo -dir “$(CONFIGDIR)”

4. Software

This chapter shows the program example to set the example described in chapter 3. Application Example. Refer to the latest hardware user's manual for details on individual registers.

4.1 Variables

Definition file name: rej05b1396_src.c

Variable Name	Size	Description
unsigned char iic_tx[BUFSIZE]	255 bytes	Transmit buffer
unsigned char iic_rx[BUFSIZE]	255 bytes	Receive buffer
unsigned char rcv_data[BUFSIZE]	255 bytes	Store receive data read from receive buffer

Definition file name: iic.c

Variable Name		Size /Bit Number	Description
static byte_dt iic_str		-	Structure to store statuses
Structure member	iic_status	1 byte	All statuses
	iic_rw	b0	R/$\overline{W}$ flag 0: Write ($\overline{W}$) slave receive 1: Read (R) slave transmit
	iic_buf_full	b1	Buffer full flag 0: Within buffer size (transmit/receive bytes < buffer size) 1: Buffer full (transmit/receive bytes ≥ buffer size)
	iic_end	b2	Communication completed flag 0: Busy (mid-communication) 1: Ready (except for mid-communication)
	-	b7 to b3	Not used (undefined)
static unsigned char far* iic_pointer		4 bytes	Transmit/receive buffer pointer
static unsigned char iic_index		1 byte	Number of transmit/receive bytes

4.2 Function Tables

Declaration	void main(void)		
Outline	Main processing		
Argument	Argument name		Meaning
	None		-
Variable (global)	Variable name		Contents
	unsigned char iic_tx[BUFSIZE]		Transmit buffer
	unsigned char iic_rx[BUFSIZE]		Receive buffer
	unsigned char rcv_data[BUFSIZE]		Store received data
Returned value	Type	Value	Meaning
	None	-	-
Function	After setting the system clock, I2C mode is enabled. Communication status is determined by the returned value of the iic_slave_end function. Each status is processed after communication is completed, and the uart2_init function is called to enable I2C mode.		

Declaration	void SetPLLClock(void)		
Outline	PLL mode setting		
Argument	Argument name		Meaning
	None		-
Variable (global)	Variable name		Contents
	None		-
Returned value	Type	Value	Meaning
	None	-	-
Function	Call this function from the main processing. This is the process for PLL mode transmission. Set peripheral clock source to 25 MHz.		

Declaration	void uart2_init(unsigned char ini)		
Outline	UART2 initial setting		
Argument	Argument name		Meaning
	unsigned char ini		0: I2C mode disabled 1: I2C mode enabled
Variable (global)	Variable name		Contents
	(structure member) iic_status		All statuses
Returned value	Type	Value	Meaning
	None	-	-
Function	Call this function from the main processing. Initialize SFRs to use UART2 special mode 1 (I2C mode). When I2C mode is enabled, set iic_status to 0x00 (clear all statuses). When executing this function, interrupts are disabled by the I flag.		

Declaration	void _stop_condition_detection(void)		
Outline	Stop condition detection interrupt handling		
Argument	Argument name		Meaning
	None		-
Variable (global)	Variable name		Contents
	None		-
Returned value	Type	Value	Meaning
	None	-	-
Function	An interrupt is generated when a stop condition is detected, and the stp_int function is called.		

Declaration	static void stp_init(void)		
Outline	Stop condition detection processing		
Argument	Argument name		Meaning
	None		-
Variable (global)	Variable name		Contents
	(structure member) iic_end		Communication completed flag
Returned value	Type	Value	Meaning
	None	-	-
Function	Called from the stop condition detection interrupt handling. UART2 associated SFR values changed mid-communication are returned to their initial values, and the communication completed flag is set to 1.		

Declaration	void_uart2_receive(void)		
Outline	UART2 receive interrupt handling		
Argument	Argument name		Meaning
	None		-
Variable (global)	Variable name		Contents
	unsigned char far* iic_pointer		Transmit/receive buffer pointer
	unsigned char iic_index		Number of transmit/receive bytes
	(structure member) iic_status		All statuses
	(structure member) iic_rw		R/$\overline{W}$ flag
Returned value	Type	Value	Meaning
	None	-	-
Functions	An interrupt is generated at the falling edge of the eighth bit of the SCL clock. This function calls the iic_id_check function after reading the U2RB register in the function header. • When the slave address is matched, generate an ACK, and set the SCL2 pin to low hold at the ninth bit. The receive interrupt is disabled, and the transmit interrupt and stop condition detection interrupt are enabled. The number of transmit/receive bytes and all statuses are cleared. When the slave is receiving, set the ACK for the next byte. When the slave is transmitting, set transmit data for the next byte. • When the slave address is not matched, generate a NACK. After the above processing, release the SCL2 pin low hold.		

Declaration	unsigned char* iic_id_check(unsigned char id, unsigned char rw)		
Outline	Slave address determine processing		
Argument	Argument name		Meaning
	unsigned char id		Received slave address
	unsigned char rw		R/$\overline{W}$ flag
Variable (global)	Variable name		Contents
	None		-
Returned value	Type	Value	Meaning
	unsigned char*	iic_rx	Receive buffer address
		iic_tx	Transmit buffer address
		NULL	Slave address does not match
Function	Called from the UART2 receive interrupt handling. The received slave address is determined. When the slave address is matched, the returned value is the buffer address. When the slave address is not matched, the returned value is NULL.		

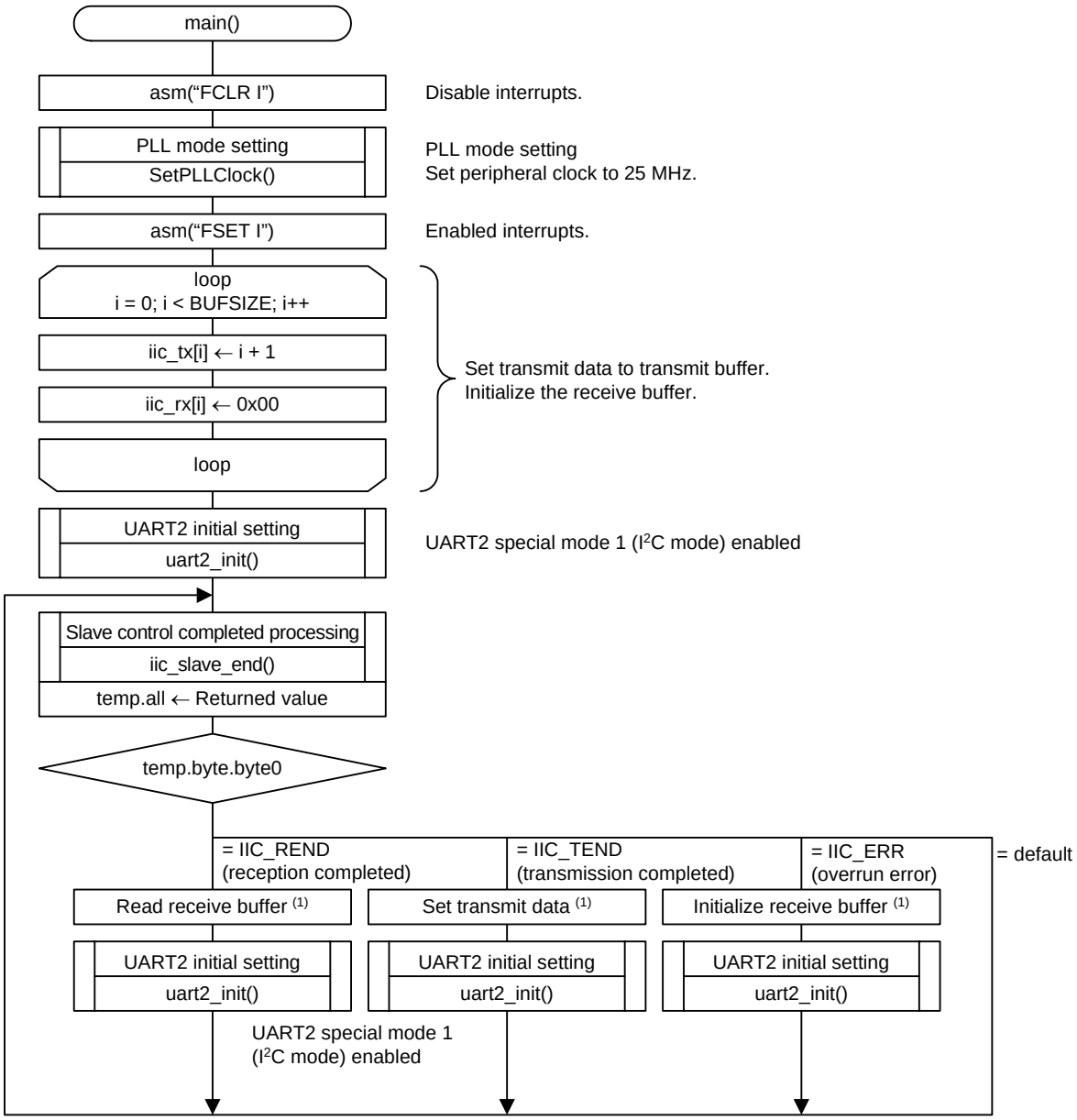
Declaration	void _uart2_trans(void)		
Outline	UART2 transmit interrupt handling		
Argument	Argument name		Meaning
	None		-
Variable (global)	Variable name		Contents
	unsigned char iic_index		Number of transmit/receive bytes
	(structure member) iic_rw		R/$\overline{W}$ flag
Returned value	Type	Value	Meaning
	None	-	-
Function	An interrupt is generated at the falling edge of the ninth bit of the SCL clock. The U2RB register is read in the function header. When the first byte (slave address) is received, disable ACK output set by the receive interrupt handler. After the first byte is received, the slave_rcv_int function is called when the slave is receiving and the slave_trn_int function is called when the slave is transmitting.		

Declaration	static void slave_rcv_int(unsigned char rb_data)		
Outline	Slave receive processing		
Argument	Argument name		Meaning
	unsigned char rb_data		Receive data from the U2RB register
Variable (global)	Variable name		Contents
	unsigned char iic_index		Number of transmit/receive bytes
	unsigned char far* iic_pointer		Transmit/receive buffer pointer
	(structure member) iic_buf_full		Buffer full flag
Returned value	Type	Value	Meaning
	None	-	-
Function	Called from the UART2 transmit interrupt handling. The argument value is stored in the receive buffer (except the slave address). • When the number of received bytes is less than the buffer size, set an ACK for the next byte. Release the SCL2 pin low hold, then enable the SCL2 pin to low hold for the next byte. • When the number of received bytes is the same as or greater than the buffer size, the buffer full flag is set to 1. Release pins SCL2 and SDA2, and disable the UART2 transmit interrupt.		

Declaration	static void slave_trn_int(unsigned char rb_data)		
Outline	Slave transmit processing		
Argument	Argument name		Meaning
	unsigned char rb_data		ACK/NACK read from the U2RB register
Variable (global)	Variable name		Contents
	unsigned char iic_index		Number of transmit/receive bytes
	unsigned char far* iic_pointer		Transmit/receive buffer pointer
	(structure member) iic_buf_full		Buffer full flag
Returned value	Type	Value	Meaning
	None	-	-
Function	Called from the UART2 transmit interrupt handling. • When an ACK is detected and the number of transmit bytes is less than the buffer size, set transmit data for the next byte. Release the SCL2 pin low hold, then enable the SCL2 pin to low hold for the next byte. • When the number of transmit bytes is the same as or greater than the buffer size, set the buffer full flag to 1. Release pins SCL2 and SDA2, and disable the UART2 transmit interrupt. • When a NACK is detected, release pins SCL2 and SDA2, and disable the UART2 transmit interrupt.		

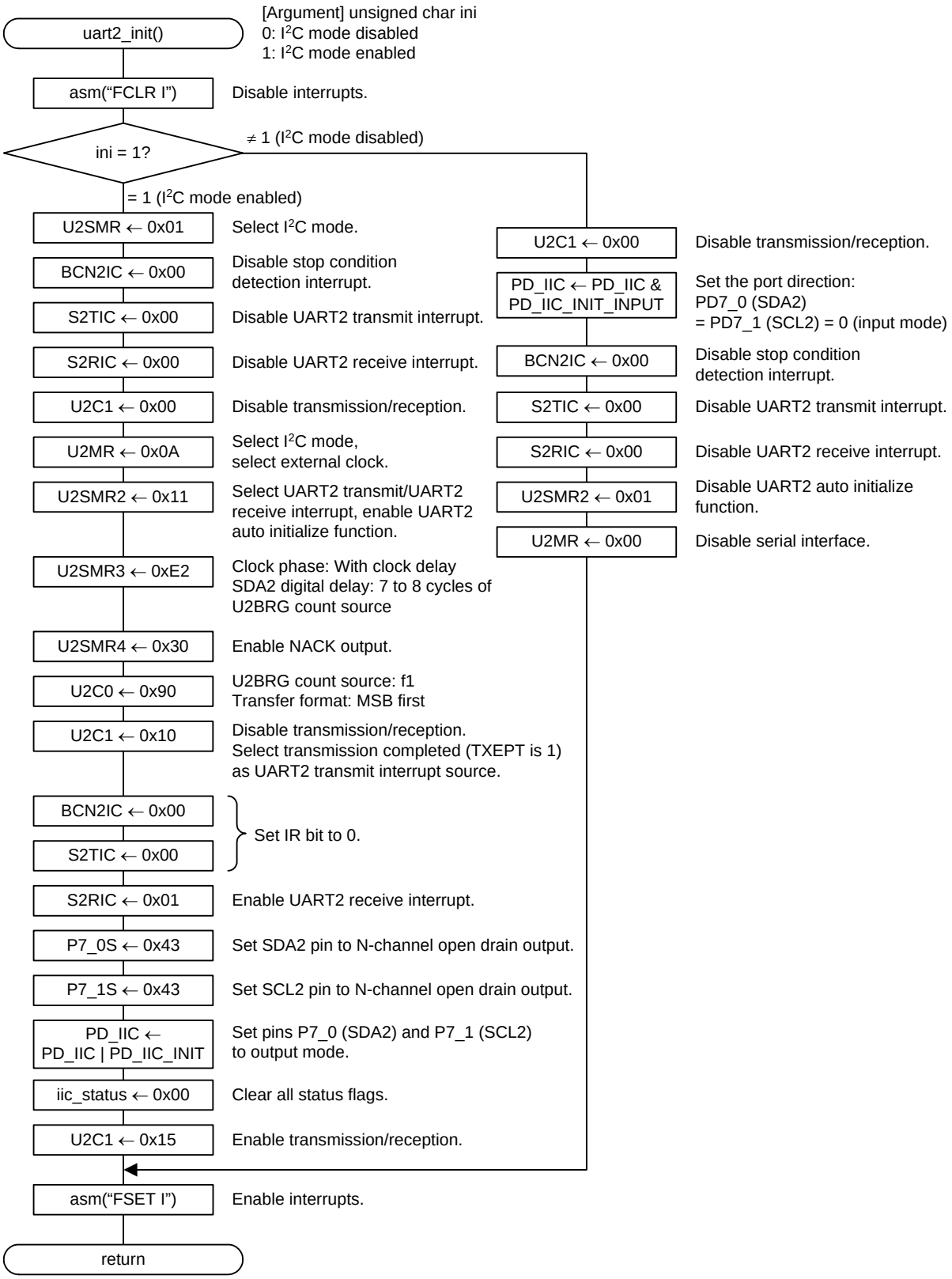
Declaration	unsigned short iic_slave_end(void)			
Outline	Slave control completed processing			
Argument	Argument name			Meaning
	None			-
Variable (global)	Variable name			Contents
	(structure member) iic_end			Communication completed flag
	(structure member) iic_rw			R/$\overline{W}$ flag
	unsigned char iic_index			Number of transmit/receive bytes
Returned value	Type		Value	Meaning
	unsigned short	Lower byte	IIC_BUSY	Mid-communication
			IIC_REND	Reception completed
			IIC_TEND	Transmission completed
			IIC_ERR	Overrun error detected
		Upper byte	1 to 255	Number of transmit/receive bytes
Function	Called from the main processing. It informs the user of the state of slave control completion. When the communication completed flag is 1 and there is transmit/receive data except for the slave address, disable I2C mode. Otherwise, return IIC_BUSY (mid-communication). After disabling I2C mode, when the communication completed flag is 0, the next communication is determined to be started and the IIC_ERR (overrun error detection) function is returned. When the communication completed flag is 1, return IIC_REND (reception completed) or IIC_TEND (transmission completed).			

4.3 Main Processing

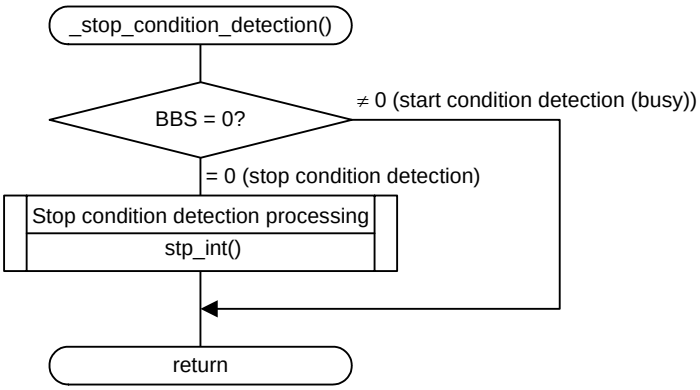


Note:
1. Additional processing can be added as needed.

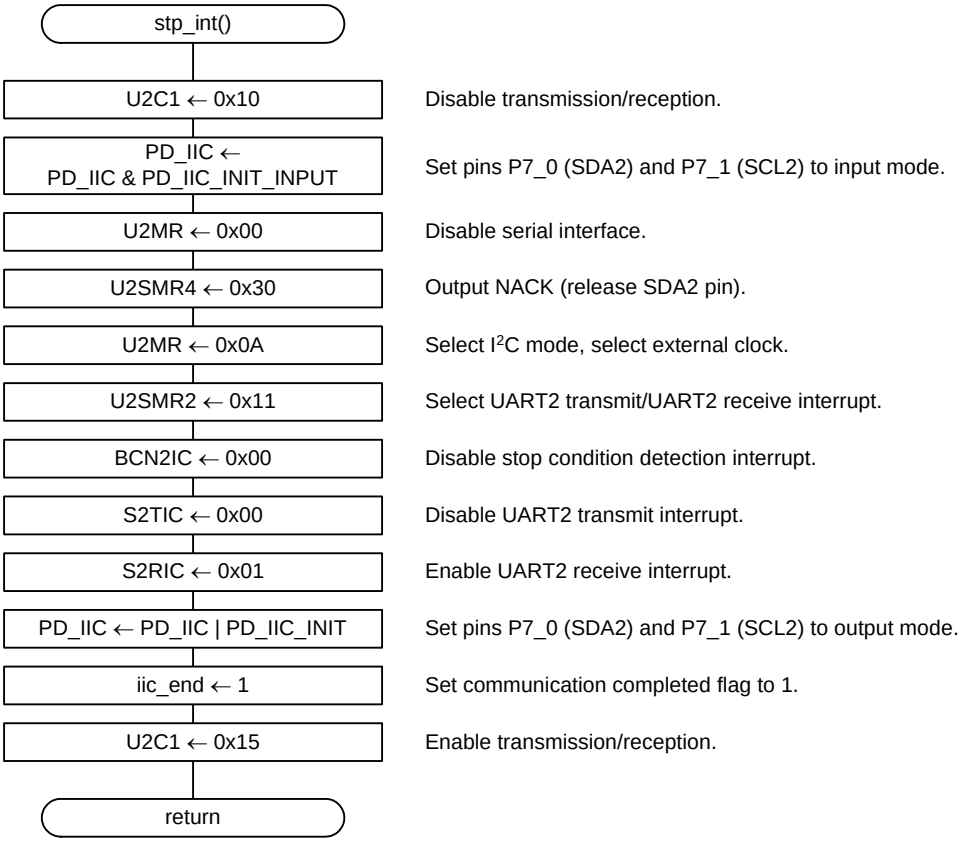
4.4 UART2 Initial Setting



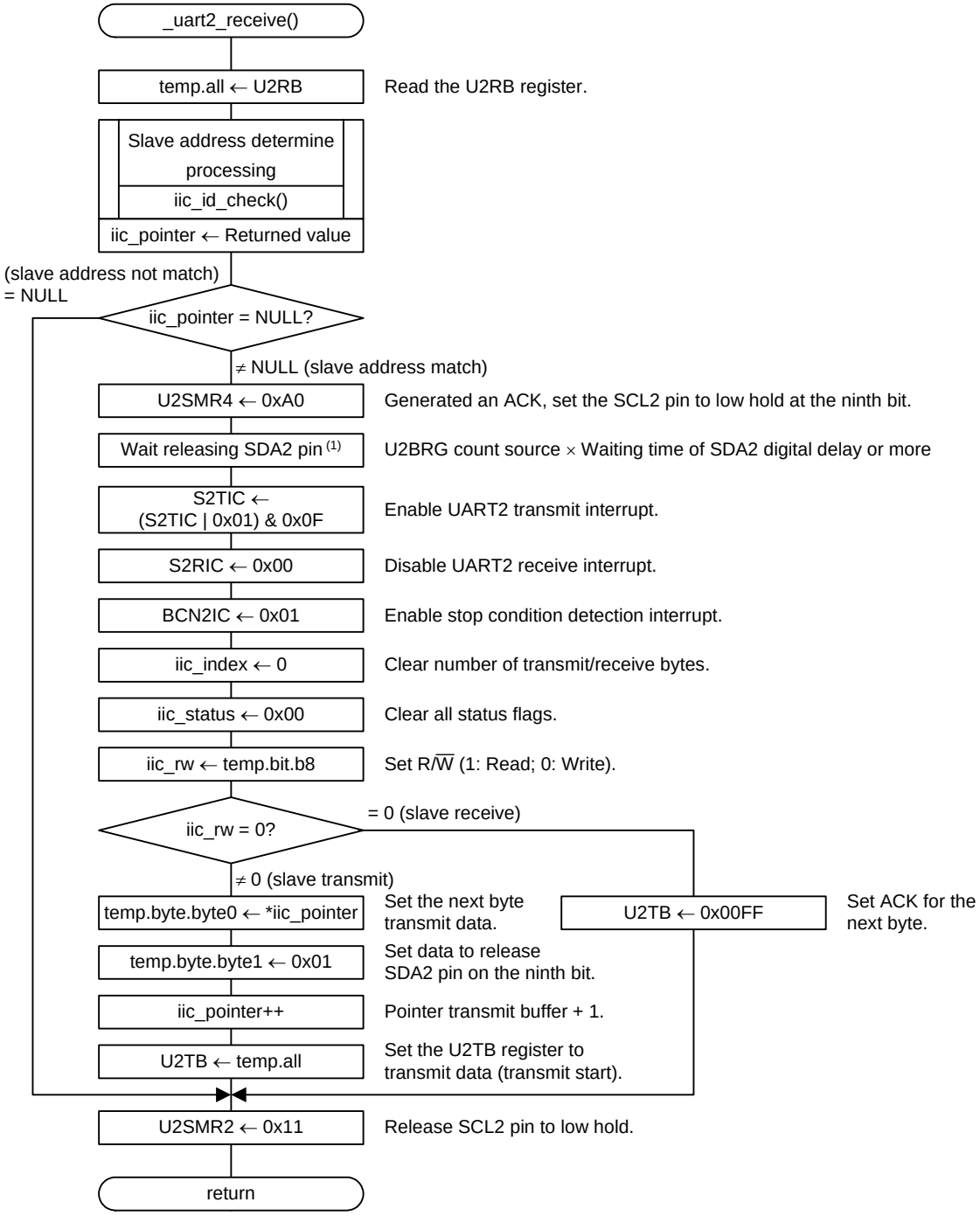
4.5 Stop Condition Detection Interrupt Handling



4.6 Stop Condition Detection Processing

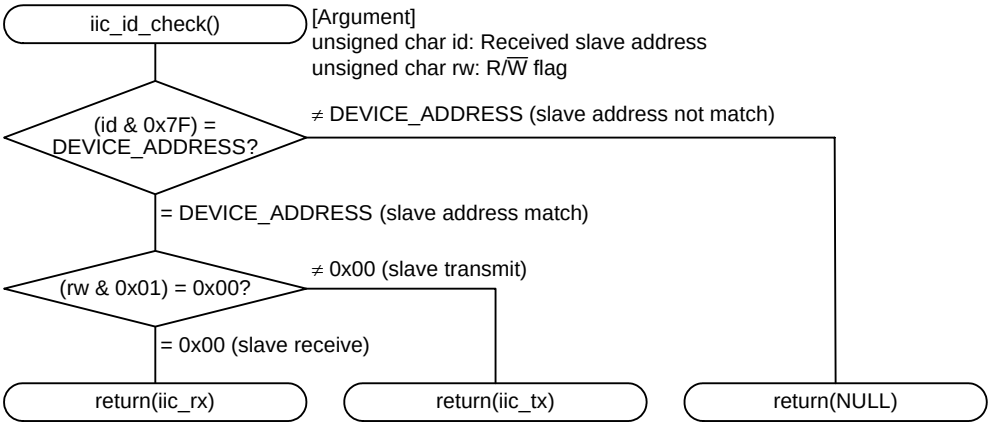


4.7 UART2 Receive Interrupt Handling

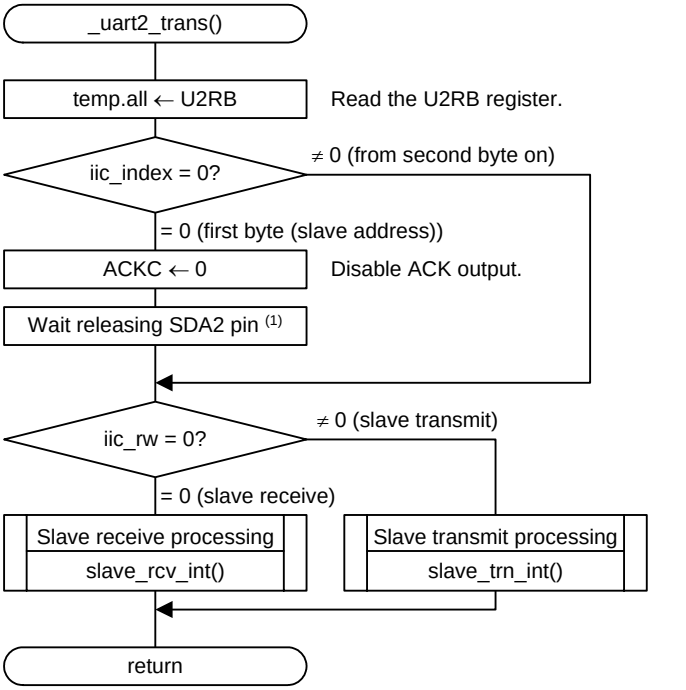


Note:
1. U2BRG count source × Waiting time of SDA2 digital delay or more

4.8 Slave Address Determine Processing

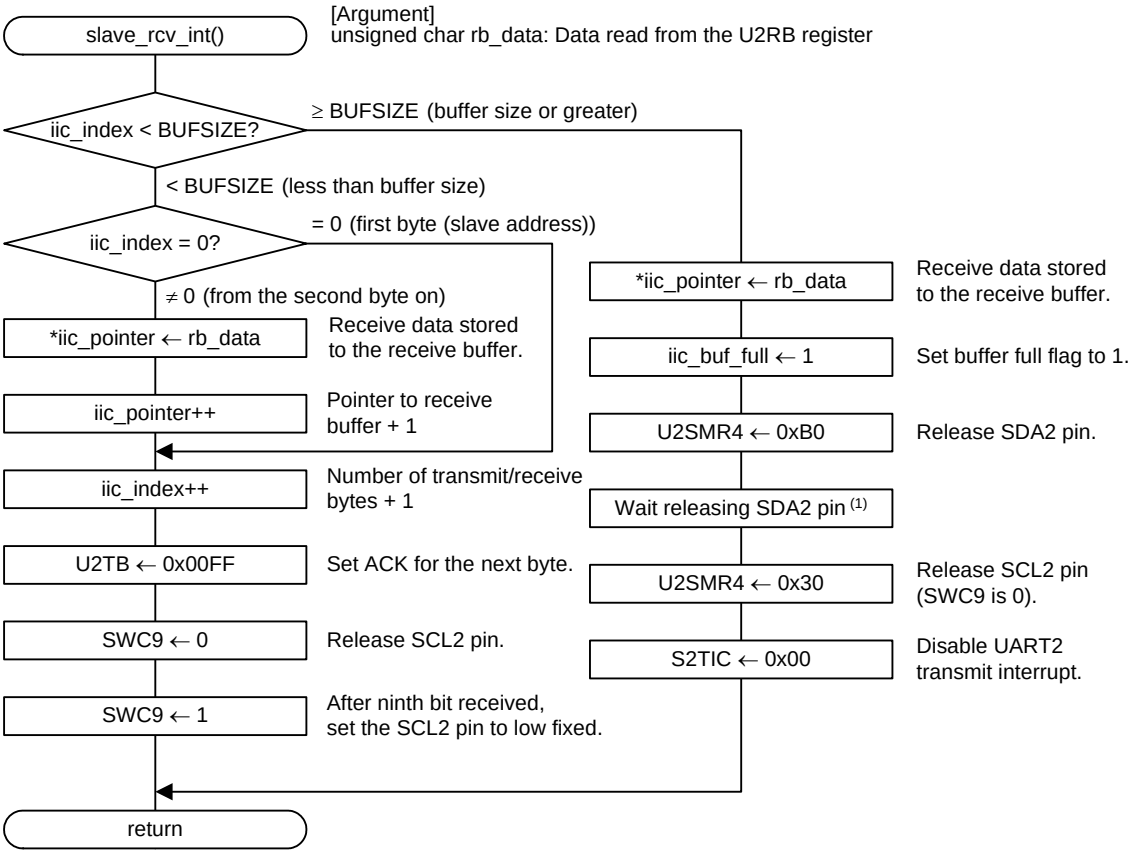


4.9 UART2 Transmit Interrupt Handling



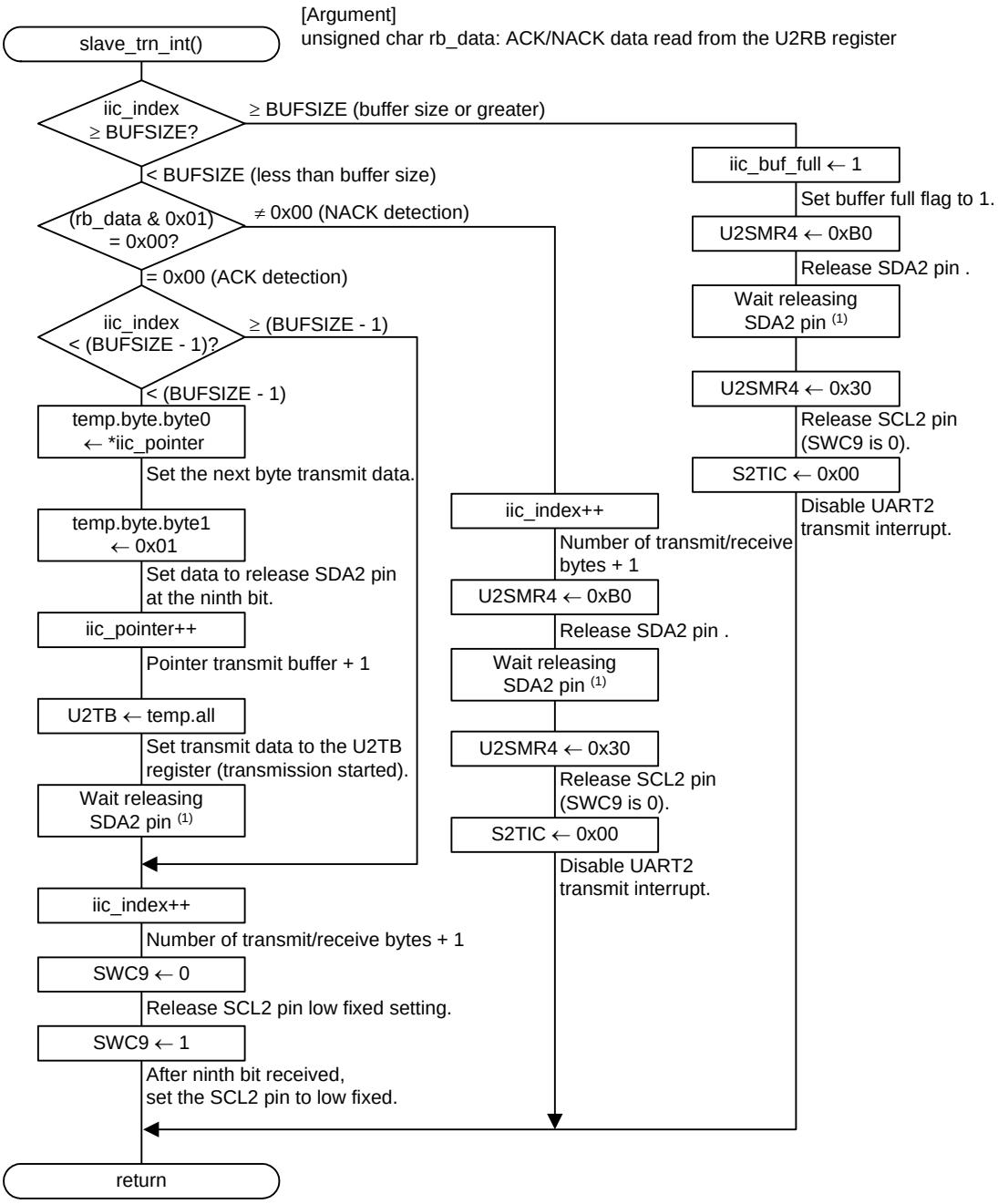
Note:
1. U2BRG count source × Waiting time of SDA2 digital delay or more

4.10 Slave Receive Processing



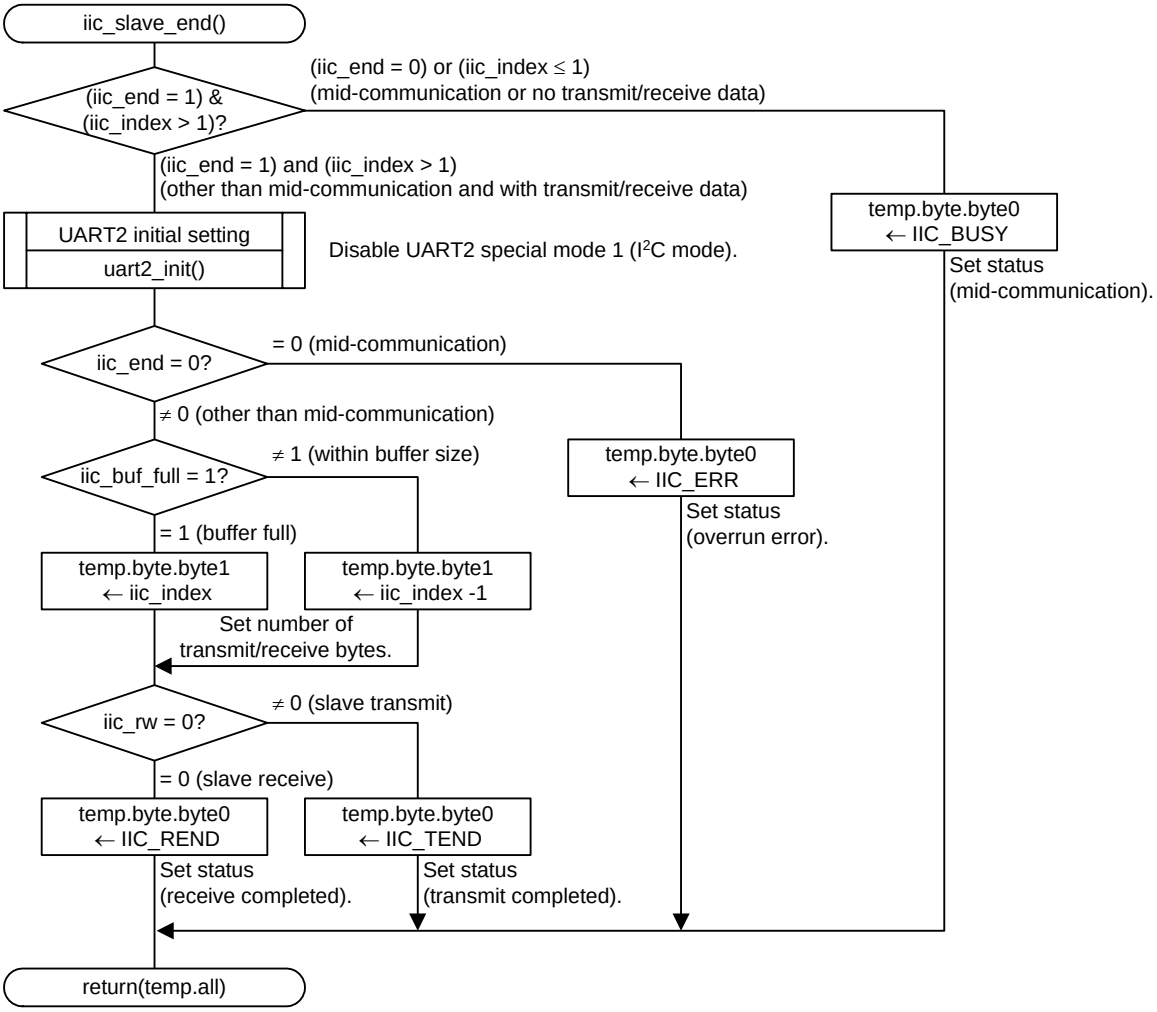
Note:
1. U2BRG count source × Waiting time of SDA2 digital delay or more

4.11 Slave Transmit Processing



Note:
1. U2BRG count source × Waiting time of SDA2 digital delay or more

4.12 Slave Control Completed Processing



5. Sample Program

A sample program can be downloaded from the Renesas Electronics website.

6. Reference Documents

R32C/118 Group User's Manual: Hardware Rev.1.10
The latest version can be downloaded from the Renesas Electronics website.

Technical Update/Technical News
The latest information can be downloaded from the Renesas Electronics website.

C Compiler Manual
R32C/100 Series C Compiler Package V.1.02
C Compiler User's Manual Rev.2.00
The latest version can be downloaded from the Renesas Electronics website.

Website and Support

Renesas Electronics website
http://www.renesas.com/

Inquiries
http://www.renesas.com/inquiry

Revision History	R32C/100 Series I2C-bus Interface Using UARTi Special Mode 1 (Slave Transmit/Receive)

Rev.	Date	Description	
		Page	Summary
1.00	Aug 31, 2010	—	First edition issued
1.01	Mar 10, 2011	8	Modify: ROM size is 657 byte in Table3.2.
		17	Add: “Wait releasing SDA2 pin” in “4.7 UART2 Receive Interrupt Handling”.
		18	Add: “Wait releasing SDA2 pin” in “4.9 UART2 Transmit Interrupt Handling”.
		19	Add: “U2SMR4 ← 0xB0”, “Wait releasing SDA2 pin” in “4.10 Slave Receive Processing”
		20	Add: “U2SMR4 ← 0xB0”, “Wait releasing SDA2 pin” in “4.11 Slave Transmit Processing”

General Precautions in the Handling of MPU/MCU Products

The following usage notes are applicable to all MPU/MCU products from Renesas. For detailed usage notes on the products covered by this manual, refer to the relevant sections of the manual. If the descriptions under General Precautions in the Handling of MPU/MCU Products and in the body of the manual differ from each other, the description in the body of the manual takes precedence.

1. Handling of Unused Pins

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

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

2. Processing at Power-on

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

   — The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the moment when power is supplied.

   In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the moment when power is supplied until the reset process is completed.

   In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the moment when power is supplied until the power reaches the level at which resetting has been specified.

3. Prohibition of Access to Reserved Addresses

   Access to reserved addresses is prohibited.

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

4. Clock Signals

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

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

5. Differences between Products

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

   — The characteristics of MPU/MCU in the same group but having different part numbers may differ because of the differences in internal memory capacity and layout pattern. When changing to products of different part numbers, implement a system-evaluation test for each of the products.

Notice

1. All information included in this document is current as of the date this document is issued. Such information, however, is subject to change without any prior notice. Before purchasing or using any Renesas Electronics products listed herein, please confirm the latest product information with a Renesas Electronics sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas Electronics such as that disclosed through our website.
2. Renesas Electronics does not assume any liability for infringement of patents, copyrights, or other intellectual property rights of third parties by or arising from the use of Renesas Electronics products or technical information described in this document. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
3. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part.
4. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
5. When exporting the products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations. You should not use Renesas Electronics products or the technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. Renesas Electronics products and technology may not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations.
6. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
7. Renesas Electronics products are classified according to the following three quality grades: "Standard", "High Quality", and "Specific". The recommended applications for each Renesas Electronics product depends on the product's quality grade, as indicated below. You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application categorized as "Specific" without the prior written consent of Renesas Electronics. Further, you may not use any Renesas Electronics product for any application for which it is not intended without the prior written consent of Renesas Electronics. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for an application categorized as "Specific" or for which the product is not intended where you have failed to obtain the prior written consent of Renesas Electronics. The quality grade of each Renesas Electronics product is "Standard" unless otherwise expressly specified in a Renesas Electronics data sheets or data books, etc.
   "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; and industrial robots.
   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; safety equipment; and medical equipment not specifically designed for life support.
   "Specific": Aircraft; aerospace equipment; submersible repeaters; nuclear reactor control systems; medical equipment or systems for life support (e.g. artificial life support devices or systems), surgical implantations, or healthcare intervention (e.g. excision, etc.), and any other applications or purposes that pose a direct threat to human life.
8. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or system manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.

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

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

RENESAS

SALES OFFICES Renesas Electronics Corporation http://www.renesas.com

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

Renesas Electronics America Inc.
2880 Scott Boulevard Santa Clara, CA 95050-2554, U.S.A.
Tel: +1-408-588-6000, Fax: +1-408-588-6130

Renesas Electronics Canada Limited
1101 Nicholson Road, Newmarket, Ontario L3Y 9C3, Canada
Tel: +1-905-898-5441, Fax: +1-905-898-3220

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

Renesas Electronics Europe GmbH
Arcadiastrasse 10, 40472 Düsseldorf, Germany
Tel: +49-211-65030, Fax: +49-211-6503-1327

Renesas Electronics (China) Co., Ltd.
7th Floor, Quantum Plaza, No.27 ZhiChunLu Haidian District, Beijing 100083, P.R.China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

Renesas Electronics (Shanghai) Co., Ltd.
Unit 204, 205, AZIA Center, No.1233 Lujiazui Ring Rd., Pudong District, Shanghai 200120, China
Tel: +86-21-5877-1818, Fax: +86-21-6887-7858 / -7898

Renesas Electronics Hong Kong Limited
Unit 1601-1613, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2886-9318, Fax: +852 2886-9022/9044

Renesas Electronics Taiwan Co., Ltd.
7F, No. 363 Fu Shing North Road Taipei, Taiwan
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670

Renesas Electronics Singapore Pte. Ltd.
1 harbourFront Avenue, #06-10, keppel Bay Tower, Singapore 098632
Tel: +65-6213-0200, Fax: +65-6278-8001

Renesas Electronics Malaysia Sdn.Bhd.
Unit 906, Block B, Menara Amcorp, Amcorp Trade Centre, No. 18, Jln Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: +60-3-7955-9390, Fax: +60-3-7955-9510

Renesas Electronics Korea Co., Ltd.
11F., Samik Lavied' or Bldg., 720-2 Yeoksam-Dong, Kangnam-Ku, Seoul 135-080, Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5141

Colophon 1.0