RISC-V

Serial Interface UARTA

Introduction

This application note explains how to use UART communication through the serial interface (UARTA).
ASCII characters transmitted from the device on the opposite side are analyzed to make responses.

Target Device

RISC-V

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

Contents

1. Specifications 3
1.1 Specification Outline 3
1.2 Operation Outline 5

2. Operation Check Conditions 6

3. Hardware 7
3.1 Hardware Configuration Example 7
3.2 List of Pins to be Used 7

4. Software 8
4.1 List of Option Byte Settings 8
4.2 List of Constants 8
4.3 List of Variables 9
4.4 List of Functions 9
4.5 Function Specifications 10
4.6 Flowcharts 12
4.6.1 Main Function 12

5. Sample Code 15

6. Reference Documents 15

Revision History 16

1. Specifications

1.1 Specification Outline

In this application note, UART communication is performed through the serial interface UARTA. ASCII characters transmitted from the device on the opposite side are analyzed to make responses.

Table 1.1 shows the peripheral function to be used and its use. Figure 1.1 and Figure 1.2 illustrate UART communication operation.

Table 1.1 Peripheral Function to be Used and its Use

Peripheral Function	Use
UARTA0	Perform UART communication using the TxDA0 pin (transmission) and the RxDA0 pin (reception).

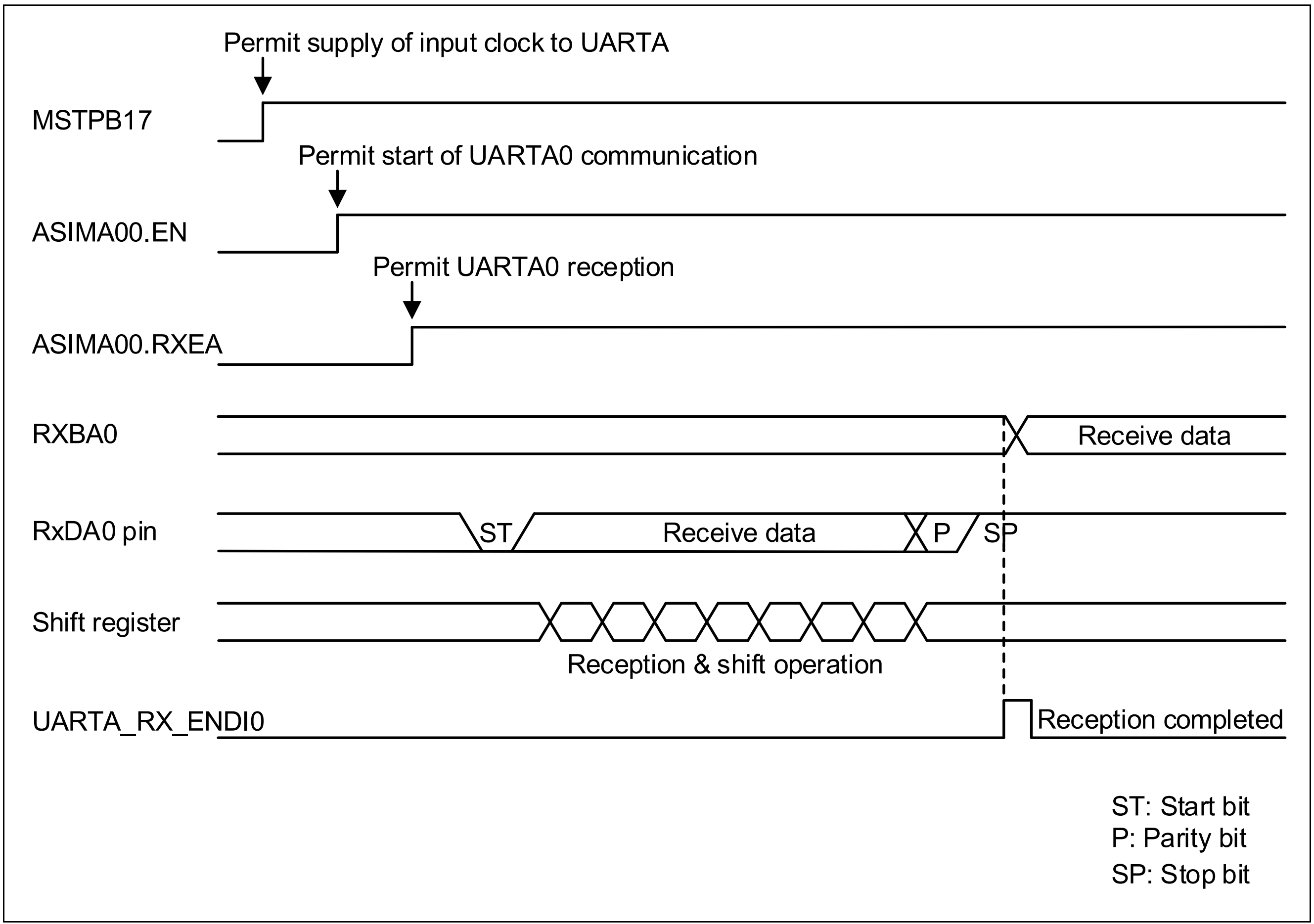

Figure 1.1 UART Reception Timing Chart

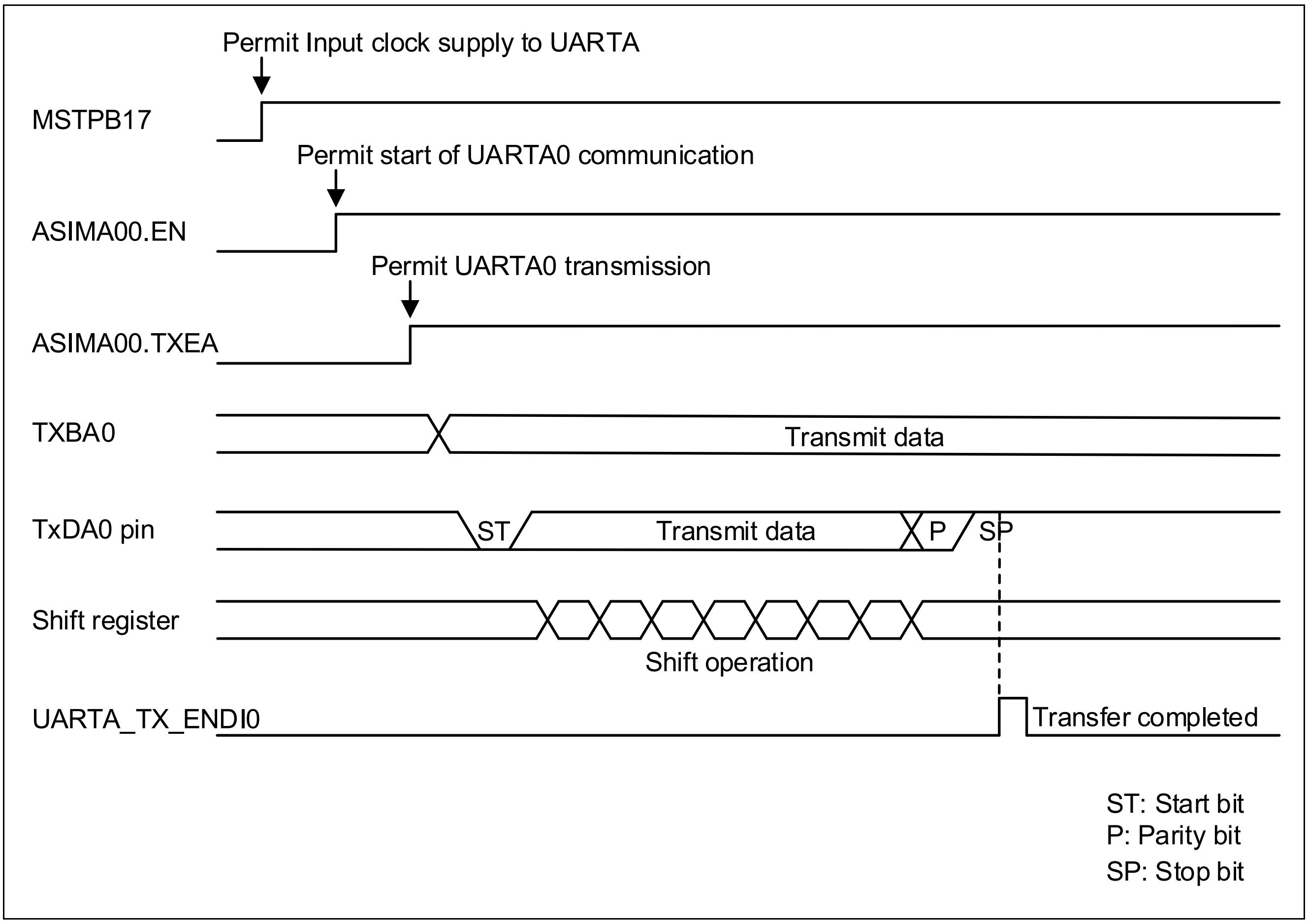

Figure 1.2 UART Transmission Timing Chart

1.2 Operation Outline

This sample code transmits, to the device on the opposite side, the data corresponding to that received from the device. If an error occurs, it transmits to the device the data corresponding to the error. Table 1.2 and Table 1.3 show the correspondence between transmit data and receive data.

Table 1.2 Correspondence between Receive Data and Transmit Data

Receive Data	Response (Transmit) Data
T (54H)	O (4FH), K (4BH)," CR" (0DH)," LF" (0AH)
t (74H)	o (6FH), k (6BH)," CR" (0DH)," LF" (0AH)
Other than above	U (55H), C (43H)," CR" (0DH)," LF" (0AH)

Table 1.3 Correspondence between Error and Transmit Data

Error	Response (Transmit) Data
Parity error	P (50H), E (45H)," CR" (0DH)," LF" (0AH)
Framing error	F (46H), E (45H)," CR" (0DH)," LF" (0AH)
Overrun error	O (4FH), E (45H)," CR" (0DH)," LF" (0AH)

(1) Perform initial setting of UART.
<UART Setting Conditions>
- Use UARTA channels 0.
- Use the P009/TxDA0 pin and the P010/RxDA0 pin for data output and data input, respectively.
- The data length is 8 bits.
- Set the data transfer direction to LSB first.
- Use even parity as the parity setting.
- Set the receive data level to standard.
- Set the transfer rate to 9600 bps.
- Use reception end interrupt (UARTA_RX_ENDI0), transmission end interrupt (UARTA_TX_ENDI0), and error interrupt (UARTA_RX_ERI0).
- Set the interrupt priority orders of UARTA_RX_ENDI0, UARTA_TX_ENDI0 and UARTA_RX_ERI0 to low priority.

(2) After the system is made to enable a UART reception by using Operation Mode Setting Register00, a WFI instruction is executed. Processing is performed in response to reception end interrupt (UARTA_RX_ENDI0) and error interrupt (UARTA_RX_ERI0).
- When an UARTA_RX_ENDI0 occurs, the received data is taken in and the data corresponding to the received data is transmitted. When an UARTA_RX_ERI0 occurs, error handling is performed to transmit the data corresponding to the error.
- After data transmission, a WFI instruction is executed again to wait for reception end interrupt (UARTA_RX_ENDI0) and error interrupt (UARTA_RX_ERI0).

2. Operation Check Conditions

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

Table 2.1 Operation Check Conditions

Item	Description
MCU used	RISC-V (R9A02G021)
Board used	RISC-V-48p Fast Prototyping Board (RTK9FPG021S000W0BJ)
Operating frequency	High-speed on-chip oscillator clock: 48 MHz CPU/peripheral hardware clock: 48 MHz
Operating voltage	3.3 V (can be operated at 1.6 V to 5.5 V)
Integrated development environment (e^2studio)	e^2studio V2024-01.1 (24.1.1) from Renesas Electronics Corp.
C compiler (e^2studio)	LLVM for RISC-V 17.0.2.202401
Smart configurator (SC)	Smart Configurator for RISC-V V24.1.1.v20240125-1623
Board support package (BSP)	V1.00 from Renesas Electronics Corp.

3. Hardware

3.1 Hardware Configuration Example

Figure 3.1 shows an example of hardware configuration that is used for this application note.

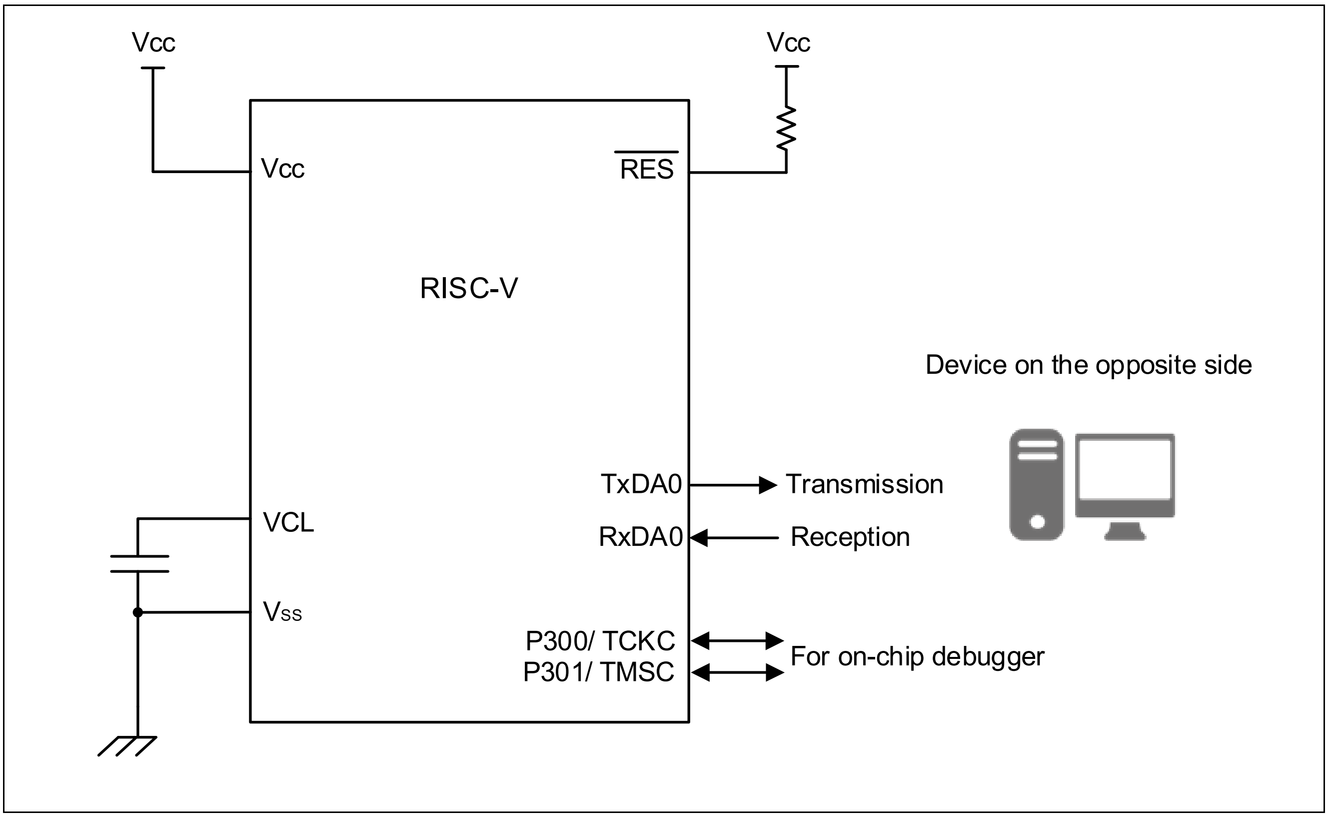

Figure 3.1 Hardware Configuration

Note 1. This simplified circuit diagram was created to show an overview of connections only. When actually designing your circuit, make sure the design includes appropriate pin handling and meets electrical characteristic requirements (connect each input-only port to V_{CC} or V_{SS} through a resistor).

Note 2. V_{CC} must not be lower than the reset release voltage (V_{LVD0}) that is specified for the LVD0.

3.2 List of Pins to be Used

Table 3.1 lists the pins to be used and their function.

Table 3.1 Pins to be Used and their Functions

Pin name	I/O	Description
P009/TxDA0/SCK10/SCL10	Output	Data transmission pin
P010/SDAA1/RxDA0	Input	Data reception pin

Caution: In this application note, only the used pin is properly connected. When designing and implementing an actual circuit, provide proper pin treatment and make sure that the hardware's electrical specifications are met.

4. Software

4.1 List of Option Byte Settings

Table 4.1 summarizes the settings of the option bytes.

Table 4.1 Option Byte Settings

Address	Setting Value	Contents
0000_0400H	FFFF_FFFFH	Disables the watchdog timer. (Counting stopped after reset)
0000_0404H	FFFF_CFFFH	High-speed on-chip oscillator clock : 48 MHz
0101_0008H	FFFF_FFFFH	Enables on-chip debugging

4.2 List of Constants

Table 4.2 lists the constants that are used in this sample program.

Table 4.2 Constants for the Sample Program

Constant	Setting	Description
g_messageOK[4]	"OK\r\n"	Response message to reception of “T”.
g_messageok[4]	"ok\r\n"	Response message to reception of “t”.
g_messageUC[4]	"UC\r\n"	Response message to reception of characters other than “T” or “t”.
g_messageFE[4]	"FE\r\n"	Response message to a framing error.
g_messagePE[4]	"PE\r\n"	Response message to a parity error.
g_messageOE[4]	"OE\r\n"	Response message to an overrun error.

4.3 List of Variables

Table 4.3 lists the global variable that is used by this sample program.

Table 4.3 Global Variable

Type	Variable Name	Contents	Function Used
uint8_t	g_uarta0_rx_buffer	Receive data buffer	main()
uint8_t	gp_uarta0_tx_address	Transmit data pointer	R_Config_UARTA0_Send(), r_Config_UARTA0_interrupt_send()
uint16_t	g_uarta0_tx_count	Transmit data number counter	R_Config_UARTA0_Send(), r_Config_UARTA0_interrupt_send()
uint8_t	gp_uarta0_rx_address	Receive data pointer	R_Config_UARTA0_Receive(), r_Config_UARTA0_interrupt_receive()
uint16_t	g_uarta0_rx_num	Receive data number counter	R_Config_UARTA0_Receive(), r_Config_UARTA0_interrupt_receive()
uint16_t	g_uarta0_rx_total_num	Receive data number	R_Config_UARTA0_Receive(), r_Config_UARTA0_interrupt_receive()
MD_STATUS	g_uarta0_tx_end	Transmit status	main(), r_Config_UARTA0_callback_sendend()
uint8_t	g_uarta0_rx_error	Receive error status	main(), r_Config_UARTA0_callback_receiveend(), r_Config_UARTA0_callback_error()

4.4 List of Functions

Table 4.4 lists the functions that are used in this sample program.

Table 4.4 Functions

Function Name	Outline
R_Config_UARTA0_Start()	UARTA0 operation start
R_Config_UARTA0_Receive()	UARTA0 reception status initialization function
R_Config_UARTA0_Send()	UARTA0 data transmission function
r_Config_UARTA0_interrupt_receive()	UARTA0 reception end interrupt handling
r_Config_UARTA0_callback_receiveend()	UARTA0 receive data classification function
r_Config_UARTA0_interrupt_error()	UARTA0 error interrupt handling
r_Config_UARTA0_callback_error()	UARTA0 reception error classification function
r_Config_UARTA0_interrupt_send()	UARTA0 transmission end interrupt handling
r_Config_UARTA0_callback_sendend()	UARTA0 transmission end processing function

4.5 Function Specifications

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

R_ UARTA0_Start()	
Outline	UARTA0 operation start
Header	r_cg_macrodriver.h 、Config_UARTA0.h、 r_cg_userdefine.h
Declaration	void R_Config_UARTA0_Start(void)
Description	Starts operation of channel 0 of serial array interface UARTA0 to make the system enter a communication wait state.
Argument	None
Return Value	None

R_ Config_UARTA0_Recieve()	
Outline	UARTA0 reception status initialization function
Header	r_cg_macrodriver.h 、Config_UARTA0.h、 r_cg_userdefine.h
Declaration	MD_STATUS R_Config_UARTA0_Receive (uint8_t * const rx_buf, uint16_t rx_num)
Description	Starts operation of channel 0 of serial array interface UARTA0 to make the system enter a communication wait state.
Argument	uint8_t * const rx_buf : [Receive data buffer address] uint16_t rx_num : [Receive data buffer size]
Return Value	[MD_OK]: Reception setting is completed [MD_ARGERROR]: Reception setting failed

R_Config_UARTA0_Send()	
Outline	UARTA0 data transmission function
Header	r_cg_macrodriver.h 、Config_UARTA0.h、 r_cg_userdefine.h
Declaration	MD_STATUS R_Config_UARTA0_Send (uint8_t * const tx_buf, uint16_t tx_num)
Description	Makes initial setting for UARTA0 transmission, and starts data transmission.
Argument	uint8_t * const tx_buf : [Transmit data buffer address] uint16_t tx_num : [Transmit data buffer size]
Return Value	[MD_OK]: Transmission setting is completed [MD_ARGERROR]: Transmission setting failed

r_Config_UARTA0_interrupt_receive()	
Outline	UART0 reception end interrupt function
Header	r_cg_macrodriver.h 、Config_UARTA0.h、 r_cg_userdefine.h
Declaration	static void __near r_Config_UARTA0_interrupt_receive (void)
Description	Makes a response (data transmission) corresponding to received data.
Argument	None
Return Value	None

r_Config_UARTA0_callback_receiveend()

Outline	UARTA0 reception error flag clear function
Header	r_cg_macrodriver.h 、Config_UARTA0.h、 r_cg_userdefine.h
Declaration	static void r_Config_UARTA0_callback_receiveend (void)
Description	Clear UARTA reception error flag.
Argument	None
Return Value	None

r_Config_UARTA0_interrupt_error()

Outline	UART error interrupt function
Header	r_cg_macrodriver.h 、Config_UARTA0.h、 r_cg_userdefine.h
Declaration	static void __near r_Config_UARTA0_interrupt_error (void)
Description	Save the received error content as err_type.
Argument	None
Return Value	None

r_uart0_callback_error()

Outline	UARTA0 reception error classification function
Header	r_cg_macrodriver.h 、Config_UARTA0.h、 r_cg_userdefine.h
Declaration	static void r_Config_UARTA0_callback_error (uint32_t err_type)
Description	Save the determined error in g_uarta0_rx_error.
Argument	uint32_t err_type : Error type
Return Value	None

r_Config_UARTA0_interrupt_send()

Outline	UARTA0 transmission end interrupt function
Header	r_cg_macrodriver.h 、Config_UARTA0.h、 r_cg_userdefine.h
Declaration	static void __near r_Config_UARTA0_interrupt_send (void)
Description	Transmits a specified number of pieces of data.
Argument	None
Return Value	None

r_Config_UARTA0_callback_sendend()

Outline	UARTA0 transmission end processing function
Header	r_cg_macrodriver.h, r_cg_serial.h, r_cg_userdefine.h
Declaration	static void r_uart0_callback_sendend(void)
Description	Set the transmission completion flag.
Argument	None
Return Value	None

4.6 Flowcharts

4.6.1 Main Function

Figure 4.1, Figure 4.2 and Figure 4.3 show the flowchart for the main function.

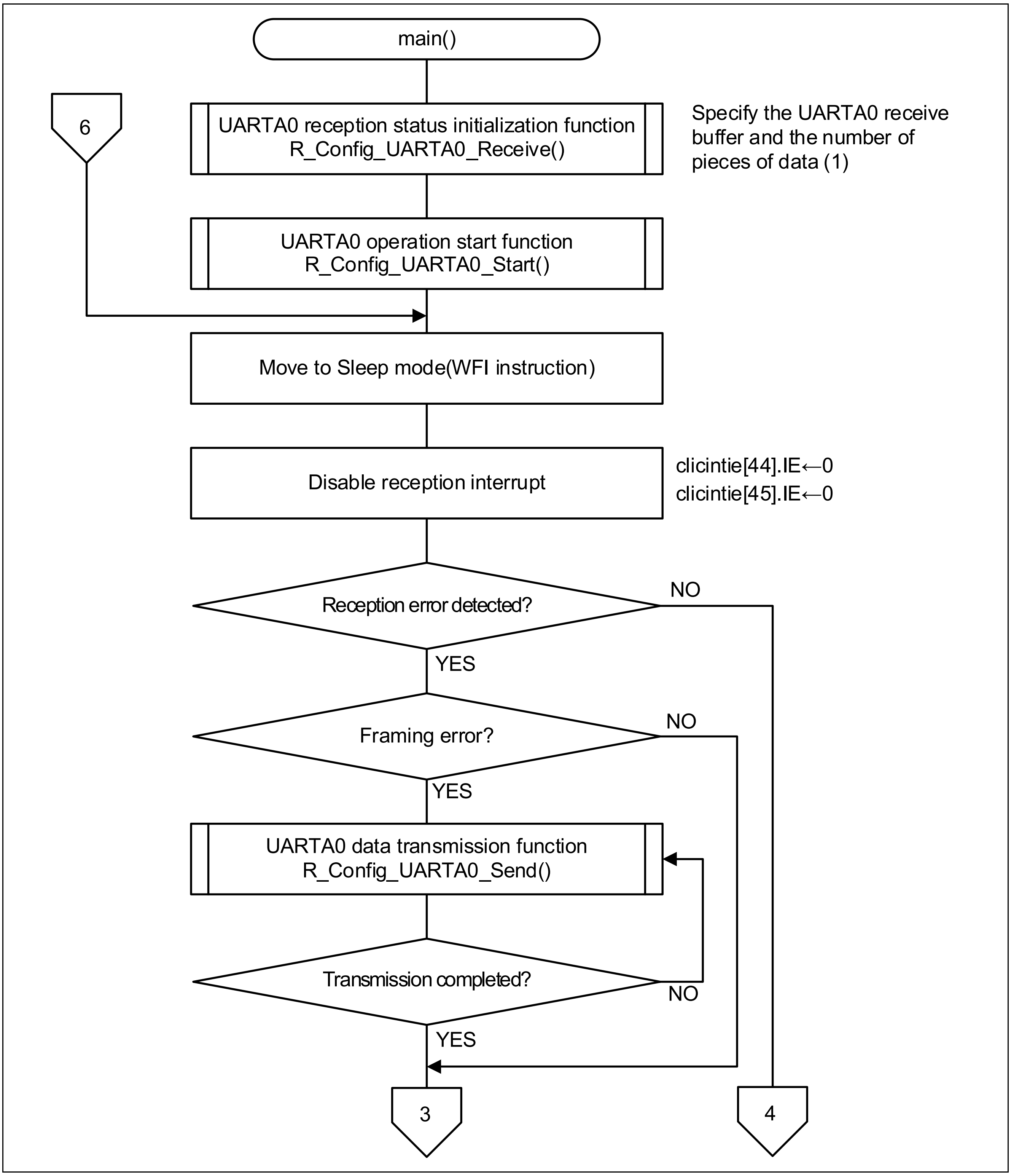

Figure 4.1 Main Function (1/3)

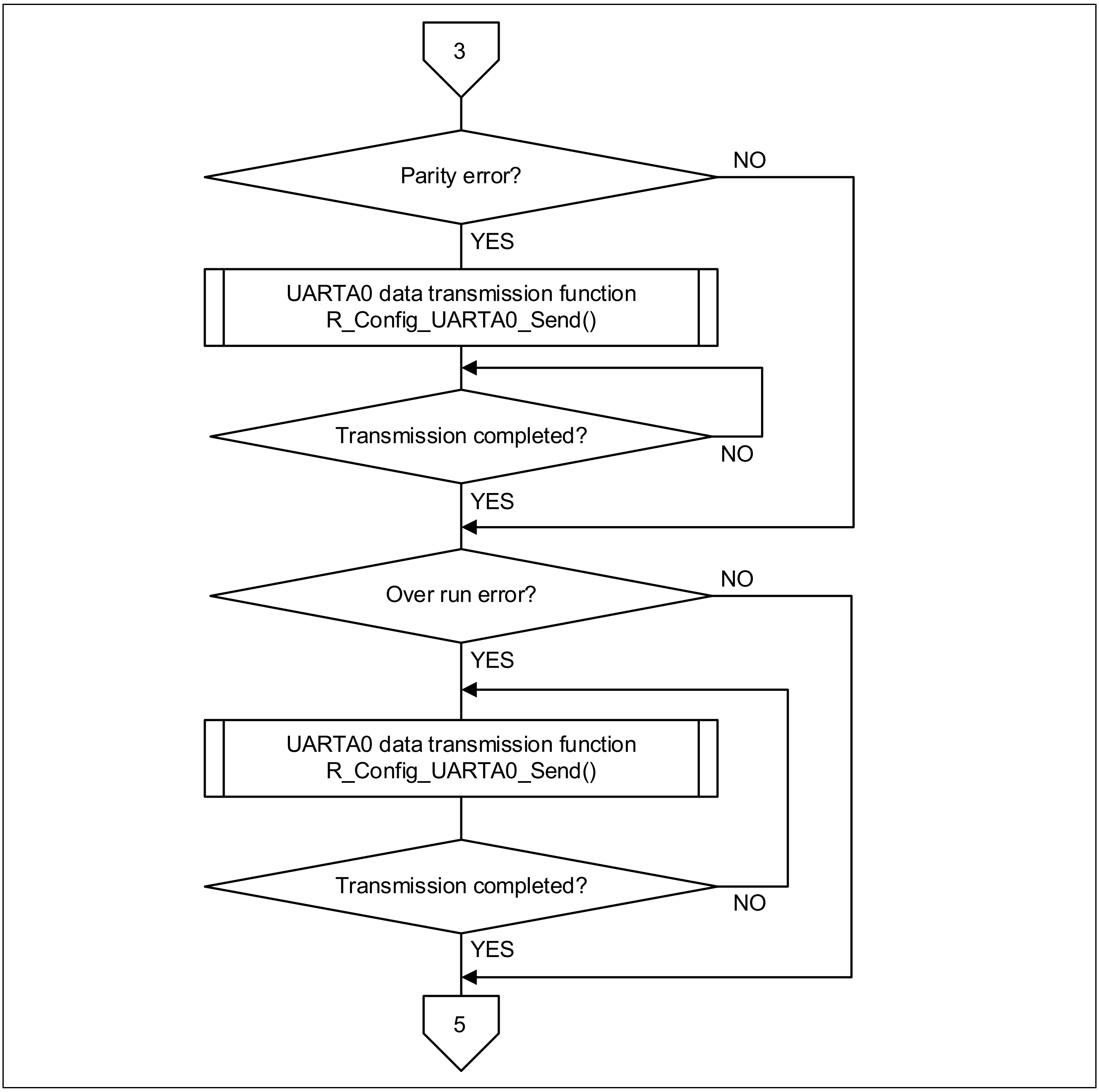

Figure 4.2 Main Function (2/3)

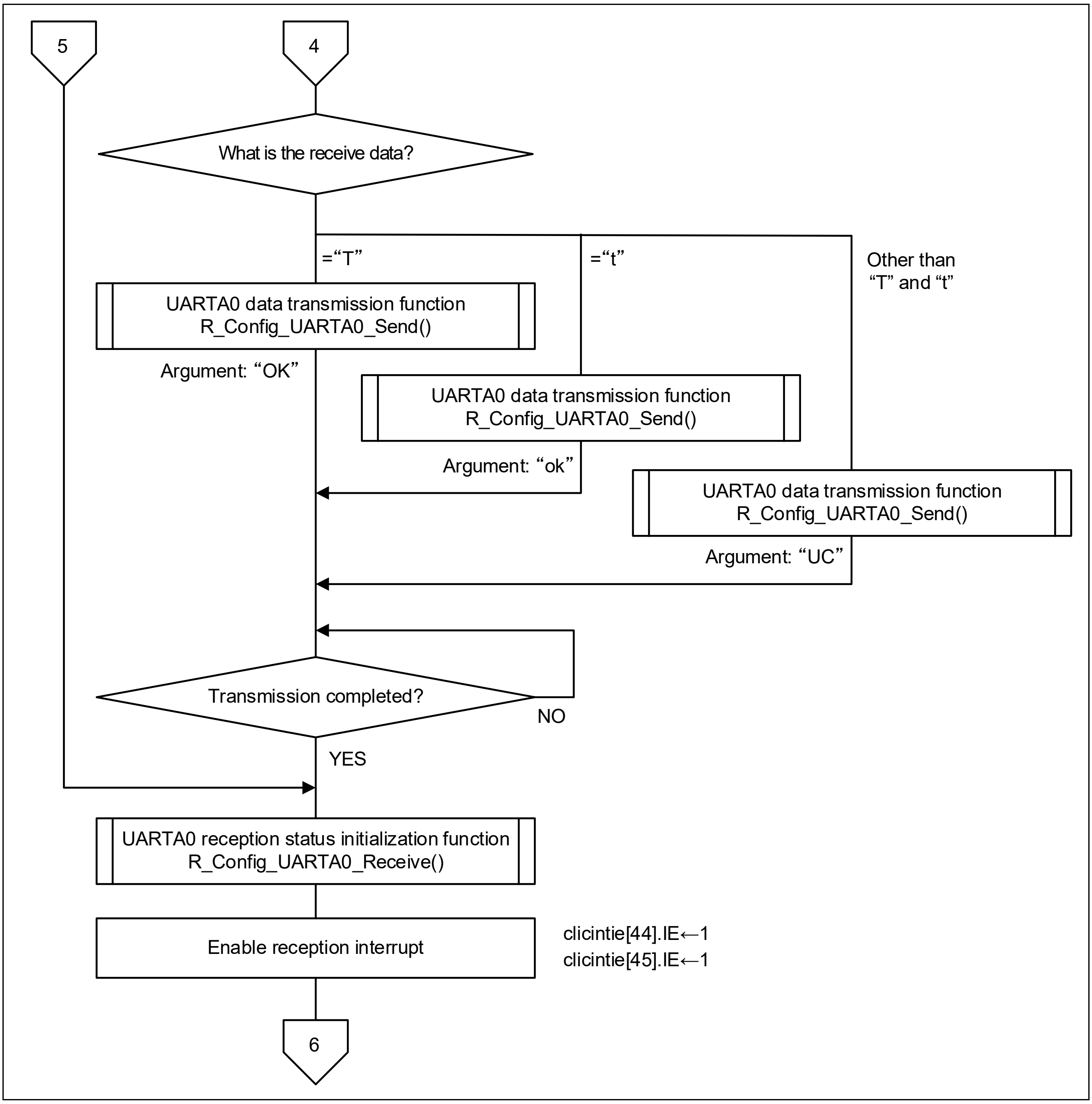

Figure 4.3 Main Function (3/3)

5. Sample Code

Sample code can be downloaded from the Renesas Electronics website.

6. Reference Documents

RISC-V User's Manual: Hardware (R01UH1036EJ)

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

Technical update

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

Revision History

Rev.	Date	Description	
		Page	Summary
1.00	Mar.18.24	—	Initial release

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

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan
www.renesas.com

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/.

Trademarks