RL78/G12

Self-Programming (Received Data via UART) CC-RL

R01AN3022EJ0110
Rev. 1.10
June 01, 2016

Introduction

This application note gives the outline of flash memory reprogramming using a self-programming technique. In this application note, flash memory is reprogrammed using the flash memory self-programming library Type01.

The sample program described in this application note limits the target of reprogramming to a part of the code flash memory (addresses 0x3BFC to 0x3BFF) and uses the other part of the code flash memory as a data area. For details on the procedures for performing self-programming and for reprogramming the entire area of code flash memory, refer to RL78/G13 Microcontroller Flash Memory Self-Programming Execution (R01AN0718E) Application Note.

Target Device

RL78/G12

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

Contents

1. Specifications

This application note explains a sample program that performs flash memory reprogramming using a self-programming library.

The sample program reads values from the code flash memory area ranging from addresses 0x3BFC to 0x3BFF and sets the flashing period of the LEDs. Subsequently, the sample program receives data (4 bytes) from the sending side and carries out self-programming to rewrite the values stored in the code flash memory addresses 0x3BFC to 0x3BFF with the received data. When the rewrite is completed, the sample program reads values from the code flash memory addresses 0x3BFC to 0x3BFF again and resets the flashing period of the LEDs with the read value.

Table 1.1 lists the peripheral functions to be used and their uses.

Table 1.1 Peripheral Functions to be Used and their Uses

Peripheral Function	Use
Channel 0 of serial array unit 0	Receives data via UART.
Channel 1 of serial array unit 0	Sends data via UART.
Port I/O	Turns on and off the LEDs.

1.1 Outline of the Flash Memory Self-Programming Library

The flash memory self-programming library is a software product that is used to reprogram the data in the code flash memory using the firmware installed on the RL78 microcontroller.

The contents of the code flash memory can be reprogrammed by calling the flash memory self-programming library from a user program.

To do flash memory self-programming, it is necessary for the user program to perform initialization for flash memory self -programming and to execute the C or assembler functions that correspond to the library functions to be used.

1.2 Code Flash Memory

The configuration of the RL78/G12 (R5F1026A) code flash memory is shown below.

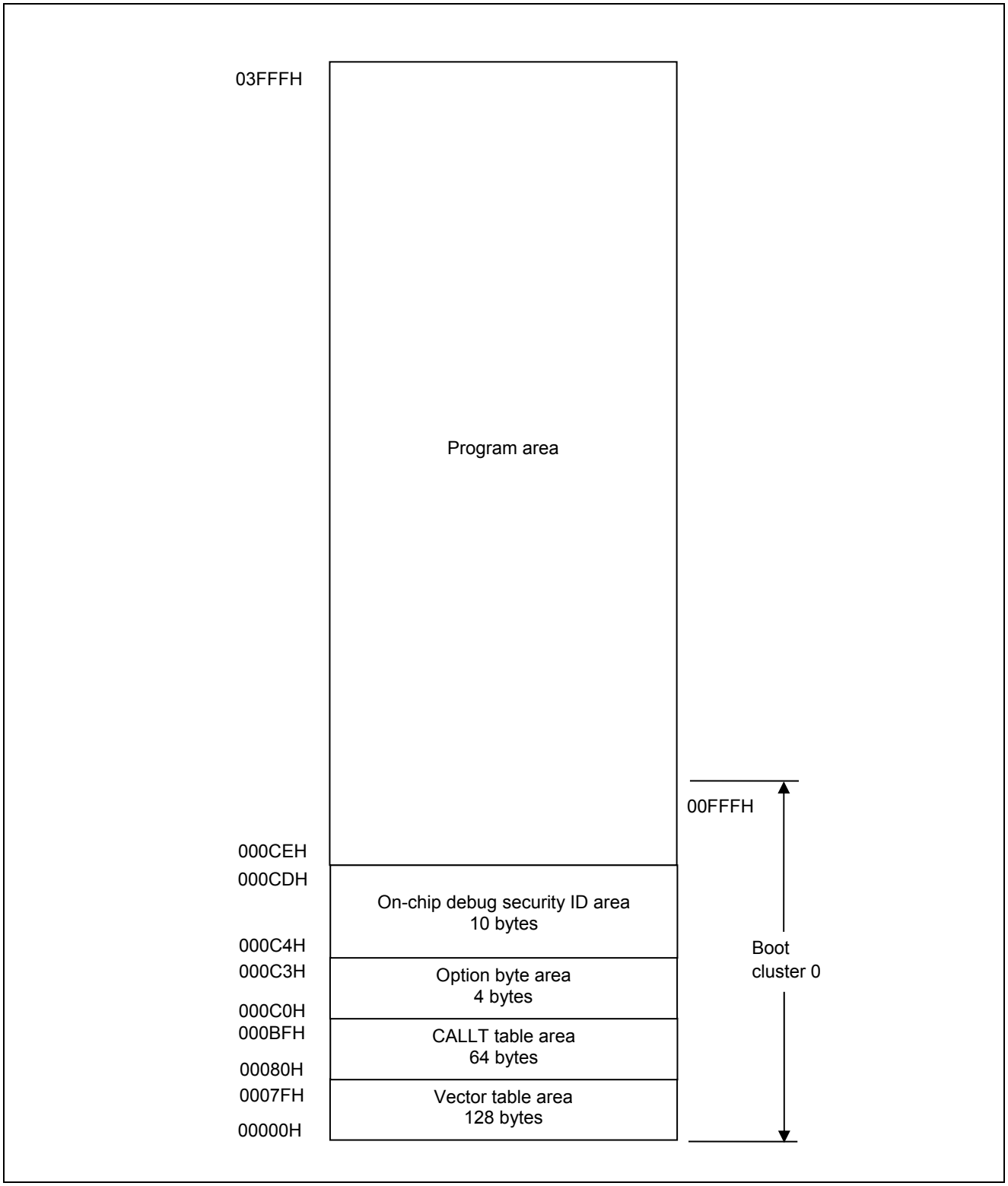

Figure 1.1 Code Flash Memory Configuration

The features of the RL78/G12 code flash memory are summarized below.

Table 1.2 Features of the Code Flash Memory

Item	Description
Minimum unit of erasure and verification	1 block (1024 bytes)
Minimum programming unit	1 word (4 bytes)
Security functions	Block erasure, programming, and boot area reprogramming protection are supported. (They are enabled at shipment)
	Security settings programmable using the flash memory self-programming library

Note: The boot area reprogramming protection setting and the security settings are disabled during flash memory self-programming.

1.3 Flash Memory Self-Programming

The RL78/G12 is provided with a library for flash memory self-programming. Flash memory self-programming is accomplished by calling functions of the flash memory self-programming library from the reprogramming program.

The flash memory self-programming library for the RL78/G12 controls flash memory reprogramming using a sequencer (a dedicated circuit for controlling flash memory). The code flash memory cannot be referenced while control by the sequencer is in progress. When the user program needs to be run while the sequencer control is in progress, therefore, it is necessary to relocate part of the segments for the flash memory self-programming library and the reprogramming program in RAM when erasing or reprogramming the code flash memory or making settings for the security flags. If there is no need to run the user program while the sequencer control is in progress, it is possible to keep the flash memory self-programming library and reprogramming program on ROM (code flash memory) for execution.

1.3.1 Flash Memory Reprogramming

The RL78/G12 does not have the boot swap function. When reprogramming, using the flash memory self-programming function, of the area where vector table data, the basic functions of the program, and flash memory self-programming library are allocated fails due to a temporary power blackout or a reset caused by an external factor, the data that is being reprogrammed will be corrupted, as the result of which the restarting of the user program or reprogramming cannot be accomplished when a reset is subsequently performed.

This subsection describes the outline image of reprogramming using the flash memory self-programming technique. The program that performs flash memory self-programming is placed in boot cluster 0.

The sample program described in this application note limits the target of reprogramming to a part of the code flash memory (addresses 0x3BFC to 0x3BFF) and uses the other part of the code flash memory as a data area. For details on the procedures for perform self-programming and for reprogramming the entire area of code flash memory, refer to RL78/G13 Microcontroller Flash Memory Self-Programming Execution (R01AN0718E) Application Note.

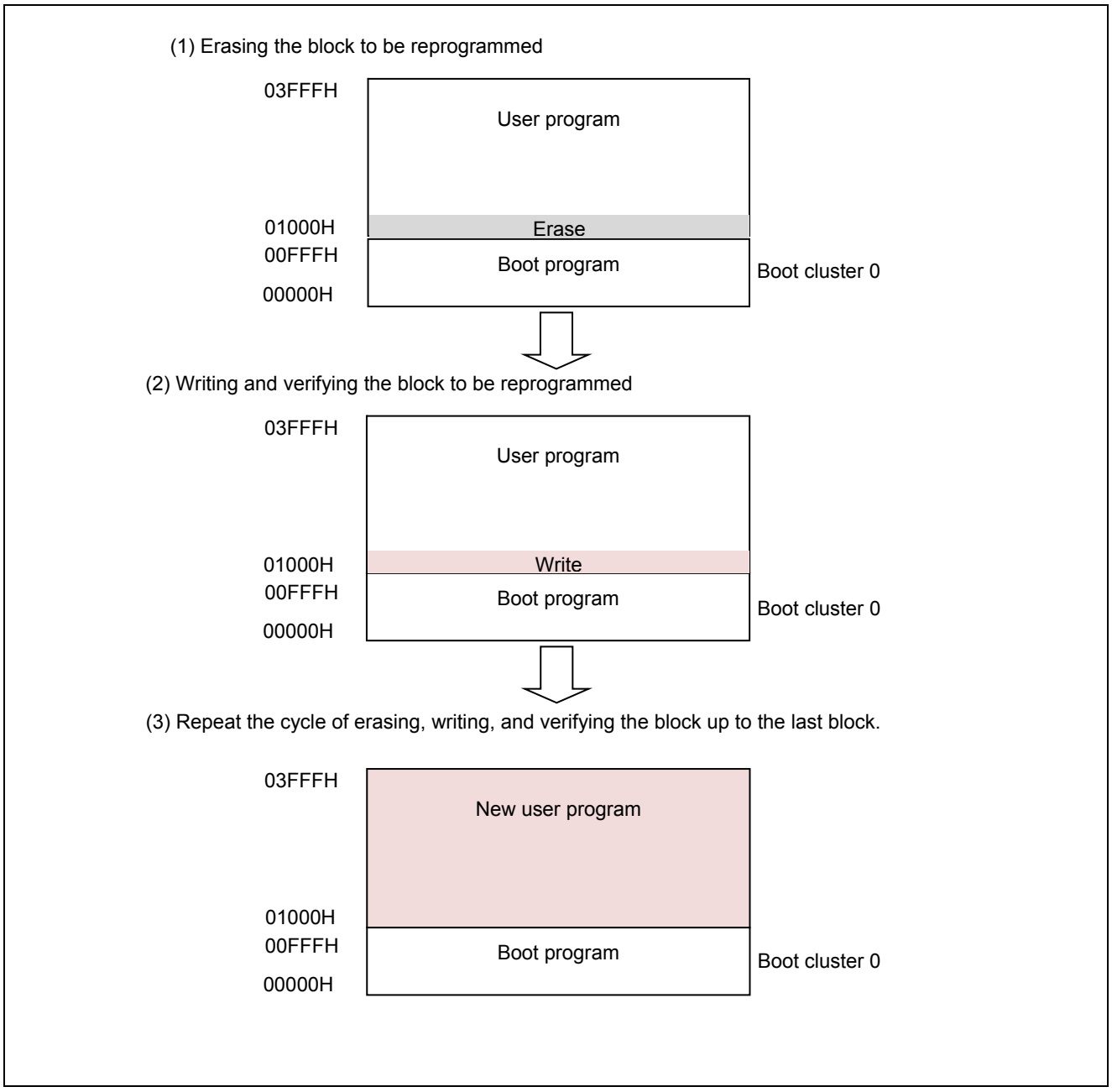

Figure 1.2 Outline of Flash Memory Reprogramming

1.4 How to Get the Flash Memory Self-Programming Library

Before compiling the sample program, please download the latest flash self-programming library and copy the library files to the following folder below “r01an3022_flash”.

incrl78 folder : fsl.h, fsl.inc, fsl_types.h

librl78 folder : fsl.lib

The flash memory self-programming library is available on the Renesas Electronics Website.

Please contact your local Renesas Electronics sales office or distributor for more information.

2. Operation Check Conditions

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

Table 2.1 Operation Check Conditions

Item	Description
Microcontroller used	RL78/G12 (R5F1026A)
Operating frequency	• High-speed on-chip oscillator (HOCO) clock: 24 MHz • CPU/peripheral hardware clock: 24 MHz
Operating voltage	5.0 V (Operation is possible over a voltage range of 2.9 V to 5.5 V.) LVD operation (V_{LVD}): Reset mode which uses 2.81 V (2.76 V to 2.87 V)
Integrated development environment(CS+)	CS+ for CC V3.03.00 from Renesas Electronics Corp.
C compiler(CS+)	CC-RL V1.02.00 from Renesas Electronics Corp.
Integrated development environment(e^{2}studio)	e^{2}studio V4.0.2.008 from Renesas Electronics Corp.
C compiler(e^{2}studio)	CC-RL V1.02.00 from Renesas Electronics Corp.
Board to be used	RL78/G12 target board (QB-R5F1026A-TB)
Flash memory self-programming library for CC-RL compiler(Type, Ver)	FSLRL78 Type01, Ver2.21 Note

Note: Use and evaluate the latest version.

3. Related Application Notes

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

- RL78/G12 Initialization (R01AN2582E) Application Note
-

4. Description of the Hardware

4.1 Hardware Configuration Example

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

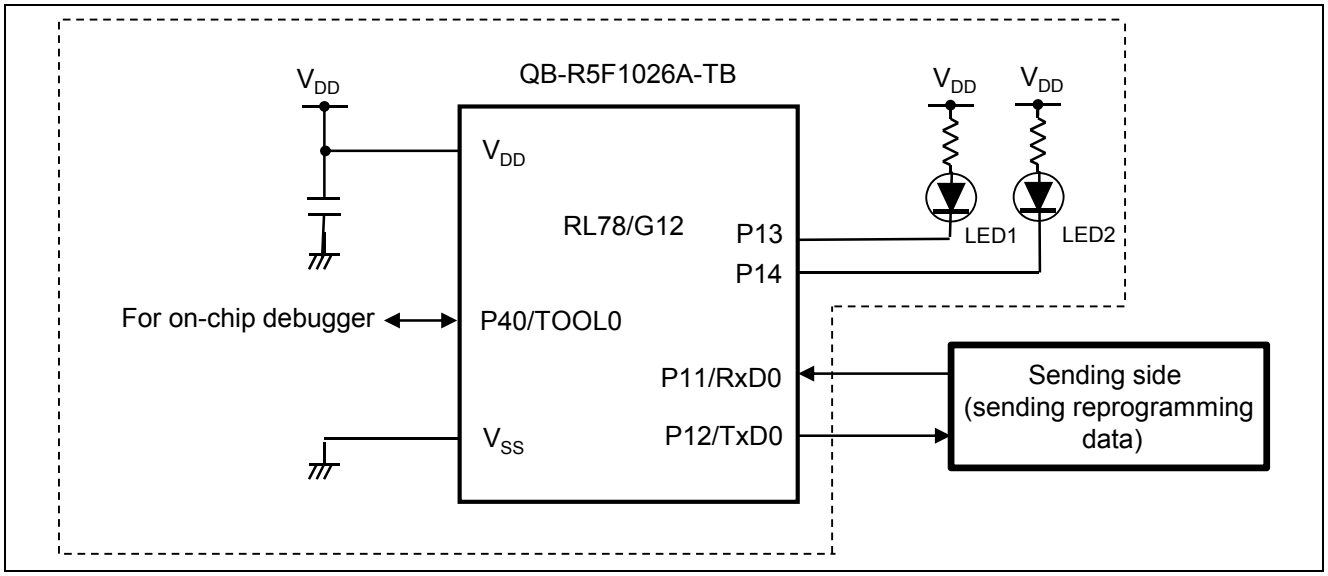

Figure 4.1 Hardware Configuration

Cautions:
1. The purpose of this circuit is only to provide the connection outline and the circuit is simplified accordingly. When designing and implementing an actual circuit, provide proper pin treatment and make sure that the hardware's electrical specifications are met (connect the input-only ports separately to V_{DD} or V_{SS} via a resistor).
2. V_{DD} must be held at not lower than the reset release voltage (V_{LVD}) that is specified as LVD.

4.2 List of Pins to be Used

Table 4.1 lists pins to be used and their functions.

Table 4.1 Pins to be Used and their Functions

Pin Name	I/O	Description
P11/ANI17/SI00/RxD0/SDA00/TOOLRxD	Input	UART serial data receive pin
P12/ANI18/SO00/TxD0/TOOLTxD	Output	UART serial data transmit pin
P13	Output	LED1 on/off control
P14	Output	LED2 on/off control

5. Description of the Software

5.1 Communication Specifications

The sample program covered in this application note receives reprogramming data via the UART bus for flash memory self-programming. The sending side sends three commands, i.e., the START, WRITE, and END commands. The sample program takes actions according to the command it received, and, if the command terminates normally, returns a normal response (0x01). If the command terminates abnormally, the sample program returns no response, turns on LED1 and LED2, and suppresses the execution of the subsequent operations. This section describes the necessary UART communication settings and the specifications for the commands.

Table 5.1 UART Communication Settings

Data length [bits]	8
Data transfer direction	LSB first
Parity setting	No parity
Transfer rate [bps]	115200

5.1.1 START Command

When the sample program receives the START command, it performs initialization for flash memory self-programming. When the command terminates normally, the program returns a normal response (0x01) to the sending side. In the case of an abnormal termination, the sample program returns no response, turns on LED1 and LED2, and suppresses the execution of the subsequent operations.

START code (0x01)	Data length (0x0002)	Command (0x02)	Data (None)	Checksum (1 byte)

5.1.2 WRITE Command

When the sample program receives the WRITE command, it writes the data it received into flash memory. The sample program returns a normal response (0x01) on normal termination of the command. In the case of an abnormal termination, the sample program returns no response, turns on LED1 and LED2, and suppresses the execution of the subsequent operations.

START code (0x01)	Data length (0x0006)	Command (0x03)	Data (4 bytes)	Checksum (1 byte)

5.1.3 END Command

When the sample program receives the END command, it performs verify processing on the block that is currently being written. If the verification terminates abnormally, the program turns on LED1 and LED2 and suppresses the execution of the subsequent operations. When the sample program receives the END command, it returns no response to the sending side regardless of whether the command terminates normally or abnormally.

START code (0x01)	Data length (0x0002)	Command (0x04)	Data (None)	Checksum (1 byte)

* The checksum is the sum of the command and data fields in units of bytes.

5.1.4 Communication Sequence

This sample program takes actions according to the sequence described below upon receipt of a command from the sending side.

(1) Sending side:
Sends a START command.

(2) Sample program:
Turns on LED2 to indicate that flash memory is being accessed. The program then performs initialization for flash memory self-programming. The program returns a response (0x01) on normal termination of the initialization processing.

(3) Sending side:
Sends a WRITE command and data (4 bytes).

(4) Sample program:
Writes the received data (4 bytes) into program flash memory addresses 0x3BFC to 0x3BFF and returns a response (0x01) on normal termination of the initialization processing.

(5) Sending side:
Sends an END command.

(6) Sample program:
Performs verify processing on the block that is currently subjected to reprogramming and turns off LED2 to indicate that flash memory is not being accessed.

5.2 Operation Outline

This application note explains a sample program that performs flash memory reprogramming using a self-programming library.

The sample program reads values from the code flash memory addresses 0x3BFC to 0x3BFF and sets the flashing interval of LED1 with the read value. Subsequently, the program receives data (4 bytes) from the sending side and carries out self-programming to rewrite the values that are stored in code flash memory addresses 0x3BFC to 0x3BFF with the received data. When reprogramming is completed, the sample program reads again the values that are stored in code flash memory addresses 0x3BFC to 0x3BFF and sets the flashing interval of LED1 with the read value.

LED1 flashes at the interval that is equal to the average value of the data (4 bytes) received from the sending side (sum of byte values stored in code flash memory addresses 0x3BFC to 0x3BFF divided by 4) × 10 [ms]. For example, if address 0x3BFC contains a value of "15," address 0x3BFD contains "150," address 0x3BFE contains "100," and address 0x3BFF contains "200," according to the calculation (15 + 150 + 100 + 200) / 4 * 10 = 1162.5, LED1 flashes at intervals of 1162.5 [ms].

LED2 indicates that flash memory is being accessed when it is on.

(1) Sets up the I/O port.

<Setting conditions>
- LED on/off control ports (LED1 and LED2): Sets P13 and P14 for output.

(2) Initializes the SAU0 channels 0 and 1.

<Setting conditions>
- Uses the SAU0 channels 0 and 1 as UART.
- Uses the P12/TxD0 pin for data output and the P11/RxD0 pin for data input.
- Sets the data length to 8 bits.
- Sets the order of data transfer mode to LSB first.
- Sets the parity setting to “No parity”.
- Sets the receive data level to standard.
- Sets the transfer rate to 115200 bps.

(3) Initializes the TAU0 channel 0.

<Setting conditions>
- Sets operation clock 0 (CK00) of the TAU0 to 23.44 [KHz], operation clock 1 (CK01) to 24 [MHz], operation clock 2 (CK02) to 12 [MHz], and operation clock 3 (CK03) to 93.75 [KHz].
- Sets the operation clock to operation clock 0 (CK00).
- Enables only software trigger start as the start trigger.
- Sets the operation mode to the interval timer mode in which no timer interrupt occurs at the beginning of counting.

(4) Starts the UART0 and disables transmit end interrupts.

(5) Enables interrupts.

(6) Reads values from code flash memory addresses 0x3BFC to 0x3BFF, calculates an average of the values in addresses 0x3BFC to 0x3BFF, and turns on LED1.

(7) If the read values are greater than 0, sets the interval time of the TAU0 channel 0 to the average value of values in addresses 0x3BFC to 0x3BFF × 10 [ms] and starts the TAU0 channel 0.

(8) Enters the HALT mode and waits for data from the sending side.

- Switches into the normal operation mode from the HALT mode upon a UART receive end interrupt request or a TAU0 channel 0 interrupt request. The program enters the HALT mode again if it returns from the HALT mode upon a TAU0 channel 0 interrupt request.

(9) Disables interrupts.

(10) Stops the TAU0 channel 0 if the read values are greater than 0.

(11) Performs initialization processing for self-programming upon receipt of a START command (0x02) from the sending side.

- Sets P14 to the low level to turn on LED2, indicating that flash memory is being accessed.
- Calls the FSL_Init function to initialize the flash memory self-programming environment and makes the following settings:
  Voltage mode : Full-speed mode
  CPU operating frequency : 24 [MHz]
  Status check mode : Status check internal mode
- Calls the FSL_Open function to start flash memory self-programming (starting the flash memory environment).
- Calls the FSL_PrepareFunctions function to make available the flash memory functions (standard reprogramming functions) that are necessary for the RAM executive.

(12) Sends a normal response (0x01) to the sending side.

(13) Receives the WRITE command (0x03) and reprogramming data (4 bytes).

(14) Computes the reprogramming target block from the write destination address.

(15) Calls the FSL_BlankCheck function to check whether the reprogramming target block has already been reprogrammed.

(16) If the reprogramming target block is reprogrammed, calls the FSL_Erase function to erase the reprogramming target block.

(17) Calls the FSL_Write function to write the received data at the write destination address.

(18) Sends a normal response (0x01) to the sending side.

(19) Receives an END command (0x04).

(20) Calls the FSL_IVerify function to verify the reprogramming target block.

(21) Sets P14 to the high level to turn off LED2, indicating that flash memory is not being accessed.

(22) Returns to step (5).

Caution: When flash memory self-programming could not be terminated normally (error occurring during processing), the sample program turns on LED1 and LED2 and suppresses the execution of the subsequent operations.

5.3 File Configuration

Table 5.2 lists the additional functions for files that are automatically generated in the integrated development environment and other additional files.

Table 5.2 List of Additional Functions and Files

File Name	Outline	Remarks
r_main.c	Main module	Additional functions: r_main_led_blink r_main_clear_uart_flag r_main_packet_analyze r_main_self_execute r_main_self_initialize r_main_write_execute
r_cg_serial_user.c	SAU module	Additional functions: r_uart0_receive_start r_uart0_send_start

5.4 List of Option Byte Settings

Table 5.3 summarizes the settings of the option bytes.

Table 5.3 Option Byte Settings

Address	Setting	Description
000C0H/010C0H	11101111B	Disables the watchdog timer. (Stops counting after the release from the reset status.)
000C1H/010C1H	01111111B	LVD reset mode 2.81 V (2.76 V to 2.87 V)
000C2H/010C2H	11100000B	HS mode, HOCO: 24 MHz
000C3H/010C3H	10000100B	Enables the on-chip debugger Erases the data in the flash memory when on-chip debug security ID authentication fails.

The option bytes of the RL78/G12 comprise the user option bytes (000C0H to 000C2H) and on-chip debug option byte (000C3H).

The option bytes are automatically referenced and the specified settings are configured at power-on time or the reset is released.

5.5 Link Option

The –start option, which is one of the link options, is provided for allocating the Flash Self-Programming Library Type01 to a ROM area.

Use the –start option to specify all sections for which settings are required by the Flash Self-Programming Library Type01.

Caution: For details on the link option procedures, refer to RL78 Compiler CC-RL User's Manual (R20UT3123E).

5.6 List of Constants

Table 5.4 lists constants for the sample program.

Table 5.4 Constants for the Sample Program

Constant	Setting	Description
LED1	P1_bit.no3	LED1 control port
LED2	P1_bit.no4	LED2 control port
LED_ON	0	LED is on.
LED_OFF	1	LED is off.
NORMAL_END	0x00	Normal termination
ERROR	0xFF	Abnormal termination
NO_RECIEVE	0x00	Command reception state: Not received
START_CODE	0x01	Command reception state: START code received
PACKET_SIZE	0x02	Command reception state: Data length received
START	0x02	START command
WRITE	0x03	WRITE command
END	0x04	END command
FULL_SPEED_MODE	0x00	Argument to flash memory self-programming library initialization function: Set operation mode to full-speed mode.
FREQUENCY_24M	0x18	Argument to flash memory self-programming library initialization function: RL78/G12's operating frequency = 24 MHz
INTERNAL_MODE	0x01	Argument to flash memory self-programming library initialization function: Turn on status check internal mode.
BLOCK_SIZE	0x400	One block size of code flash memory (1024 bytes)
TXSIZE	0x01	Size of response data to be sent to sending side
RXSIZE	0x06	Size of receive buffer
WRITESIZE	0x01	Write data size (words)
WRITEADDR	0x3BFC	Write start address
READADDR	0x3BFC	Read start address

5.7 List of Variables

Table 5.5 lists the global variables that are used in this sample program.

Table 5.5 Global Variables for the Sample Program

Type	Variable Name	Contents	Function Used
uint8_t	g_intsr_flag	UART receive end interrupt flag	main r_main_clear_uart_flag r_uart0_interrupt_receive
uint8_t	g_intsre_flag	UART receive error interrupt flag	Main r_main_clear_uart_flag r_uart0_interrupt_error

5.8 List of Functions

Table 5.6 lists the functions that are used in this sample program.

Table 5.6 List of Functions

Function Name	Outline
R_UART0_Start	Starts UART0.
r_uart0_interrupt_receive	UART0 receive end interrupt
r_uart0_interrupt_error	UART0 receive error interrupt
r_main_led_blink	Sets LED1 flashing interval according to read values and starts LED1 flashing.
R_TAU0_Channel0_Start	Starts TAU0 channel 0.
r_tau0_channel0_interrupt	TAU0 channel 0 interrupt
r_uart0_receive_start	Receives data via UART0.
r_main_clear_uart_flag	Clears UART0 receive end interrupt flag and UART0 receive error interrupt flag.
R_TAU0_Channel0_Stop	Stops TAU0 channel 0.
r_main_packet_analyze	Analyzes receive data.
r_main_self_execute	Executes flash memory self-programming.
r_main_self_initialize	Executes initialization for flash memory self-programming.
r_main_write_execute	Executes flash memory reprogramming.
r_uart0_send_start	Sends data via UART0.

5.9 Function Specifications

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

[Function Name] R_ UART0_Start

Synopsis	Start UART0.
Header	r_cg_macrodriver.h r_cg_serial.h r_cg_userdefine.h
Declaration	void R_ UART0_Start(void)
Explanation	This function starts channels 0 of the serial array unit 0 and places them in communication wait state.
Arguments	None
Return value	None
Remarks	None

[Function Name] r_uart_interrupt_receive

Synopsis	UART0 receive end interrupt
Header	r_cg_macrodriver.h r_cg_serial.h r_cg_userdefine.h
Declaration	__interrupt void r_uart_interrupt_receive(void)
Explanation	This function sets the UART0 receive end interrupt flag (flag g_intsr_flag) to 1.
Arguments	None
Return value	None
Remarks	None

[Function Name] r_uart_interrupt_error

Synopsis	UART0 receive error interrupt
Header	r_cg_macrodriver.h r_cg_serial.h r_cg_userdefine.h
Declaration	__interrupt void r_uart_interrupt_error(void)
Explanation	This function sets the UART0 receive error flag (flag g_intsre_flag) to 1.
Arguments	None
Return value	None
Remarks	None

[Function Name] r_main_led_blink

Synopsis	Set LED1 flashing interval according to read values and start LED1 flashing.	
Header	r_cg_macrodriver.h r_cg_cgc.h r_cg_port.h r_cg_serial.h r_cg_timer.h r_cg_userdefine.h	
Declaration	void r_main_led_blink(float average)	
Explanation	This function sets the LED1 flashing interval to the value of argument average × 10 [ms] and starts flashing LED1.	
Arguments	average	Average of values in code flash memory addresses 0x3BFC to 0x3BFF
Return value	None	
Remarks	None	

[Function Name] R_TAU0_Channel0_Start

Synopsis	Start TAU0 channel 0.	
Header	r_cg_macrodriver.h r_cg_timer.h r_cg_userdefine.h	
Declaration	void R_TAU0_Channel0_Start(void)	
Explanation	This function starts the TAU0 channel 0.	
Arguments	None	
Return value	None	
Remarks	None	

[Function Name] r_tau0_channel0_interrupt

Synopsis	TAU0 channel 0 interrupt	
Header	r_cg_macrodriver.h r_cg_timer.h r_cg_userdefine.h	
Declaration	__interrupt void r_tau0_channel0_interrupt(void)	
Explanation	This function inverts the state (ON/OFF) of LED1.	
Arguments	None	
Return value	None	
Remarks	None	

[Function Name] r_uart0_receive_start

Synopsis	Receive data via UART0.	
Header	r_cg_macrodriver.h r_cg_serial.h r_cg_userdefine.h	
Declaration	uint8_t r_uart0_receive_start(uint16_t *rx_length, uint8_t *rx_buf)	
Explanation	This function stores the receive data in the receive buffer (rx_buf) and the receive data length [bytes] in rx_length.	
Arguments	rx_length	Address of area for storing receive data length [bytes]
	rx_buf	Address of receive data buffer
Return value	Normal termination: NORMAL_END Abnormal termination: ERROR	
Remarks	None	

[Function Name] r_main_clear_uart_flag

Synopsis	Clear UART0 receive end interrupt flag and UART0 receive error interrupt flag.
Header	r_cg_macrodriver.h r_cg_cgc.h r_cg_port.h r_cg_serial.h r_cg_timer.h r_cg_userdefine.h
Declaration	void r_main_clear_uart_flag(void)
Explanation	This function clears the UART0 receive end interrupt flag (g_intsr_flag) and the UART0 receive error interrupt flag (g_intsre_flag).
Arguments	None
Return value	None
Remarks	None

[Function Name] R_TAU0_Channel0_Stop

Synopsis	Stop TAU0 channel 0.
Header	r_cg_macrodriver.h r_cg_timer.h r_cg_userdefine.h
Declaration	void R_TAU0_Channel0_Stop(void)
Explanation	This function stops the TAU0 channel 0.
Arguments	None
Return value	None
Remarks	None

[Function Name] r_main_packet_analyze

Synopsis	Analyze receive data.	
Header	r_cg_macrodriver.h r_cg_cgc.h r_cg_port.h r_cg_serial.h r_cg_timer.h r_cg_userdefine.h	
Declaration	uint8_t r_main_packet_analyze(uint16_t rx_length, uint8_t *rx_buf)	
Explanation	This function checks the parameters of the command received, and computes and compares the checksum to check whether the received data is correct.	
Arguments	rx_length	Receive data length [bytes]
	rx_buf	Address of receive data buffer
Return value	START command received: START WRITE command received: WRITE END command received: END Command parameter error or checksum error: ERROR	
Remarks	None	

[Function Name] r_main_self_execute

Synopsis	Execute flash memory self-programming.
Header	r_cg_macrodriver.h r_cg_cgc.h r_cg_port.h r_cg_serial.h r_cg_userdefine.h fsl.h fsl_types.h
Declaration	void r_main_self_execute(void)
Explanation	This function executes flash memory self-programming.
Arguments	None
Return value	None
Remarks	None

[Function Name] r_main_self_initialize

Synopsis	Execute initialization for flash memory self-programming.	
Header	r_cg_macrodriver.h	
	r_cg_cgc.h	
	r_cg_port.h	
	r_cg_serial.h	
	r_cg_userdefine.h	
	fsl.h	
	fsl_types.h	
Declaration	uint8_t r_main_self_initialize(void)	
Explanation	This function executes initialization prior to flash memory self-programming.	
Arguments	None	
Return value	Normal termination: FSL_OK	
	Parameter error: FSL_ERR_PARAMETER	
	Erase error: FSL_ERR_ERASE	
	Internal verify error: FSL_ERR_IVERIFY	
	Write error: FSL_ERR_WRITE	
	Flow error: FSL_ERR_FLOW	
Remarks	None	

[Function Name] r_main_write_execute

Synopsis	Execute flash memory reprogramming.	
Header	r_cg_macrodriver.h	
	r_cg_cgc.h	
	r_cg_port.h	
	r_cg_serial.h	
	r_cg_userdefine.h	
	fsl.h	
	fsl_types.h	
Declaration	uint8_t r_main_write_execute(uint32_t write_addr)	
Explanation	This function executes flash memory reprogramming.	
Arguments	Write_addr	Write start address
Return value	Normal termination: FSL_OK	
	Abnormal termination: ERROR	
Remarks	None	

[Function Name] r_uart0_send_start

Synopsis	Send data via UART0.	
Header	r_cg_macrodriver.h	
	r_cg_serial.h	
	r_cg_userdefine.h	
Declaration	uint8_t r_uart0_send_start(uint16_t tx_length, uint8_t *tx_buf)	
Explanation	This function transmits the number of data bytes specified in tx_length [bytes] from tx_buf.	
Arguments	tx_length	Transmit data length [bytes]
	tx_buf	Address of transmit data buffer
Return value	Normal termination: NORMAL_END	
	Parameter error (txlength smaller than 0): ERROR	
Remarks	None	

5.10 Flowcharts

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

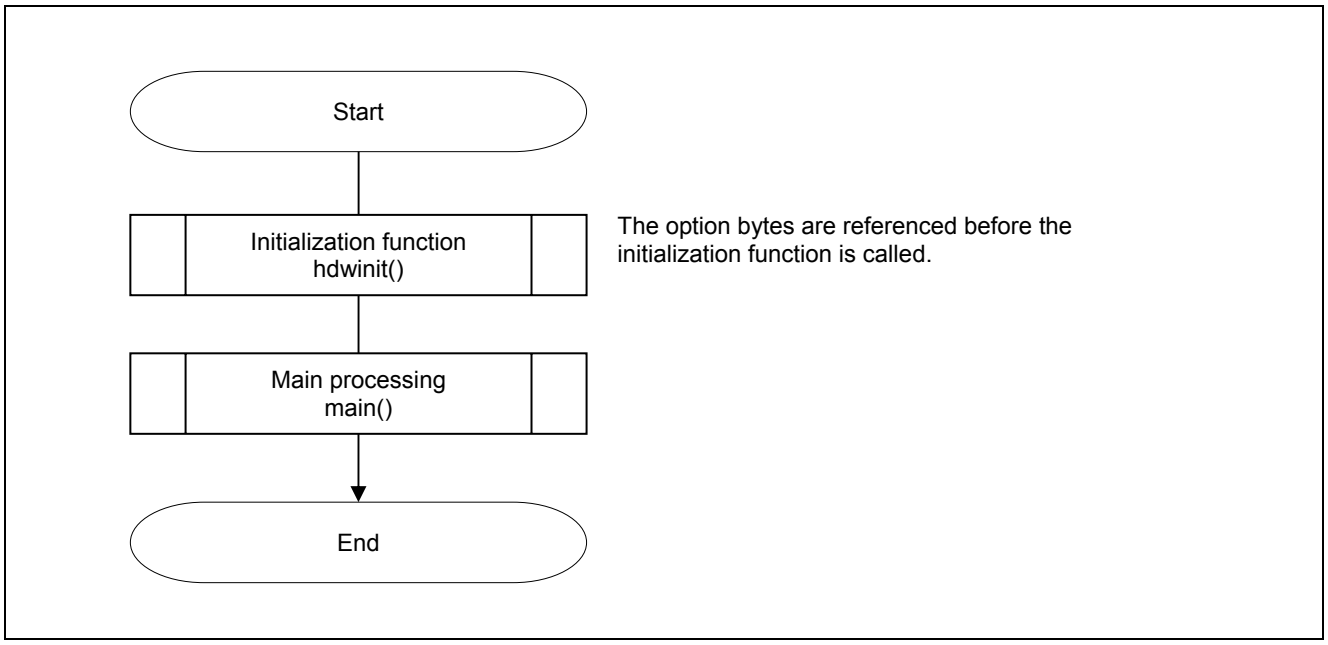

Figure 5.1 Overall Flow

Note: Startup routine is executed before and after the initialization function.

5.10.1 Initialization Function

Figure 5.2 shows the flowchart for the initialization function.

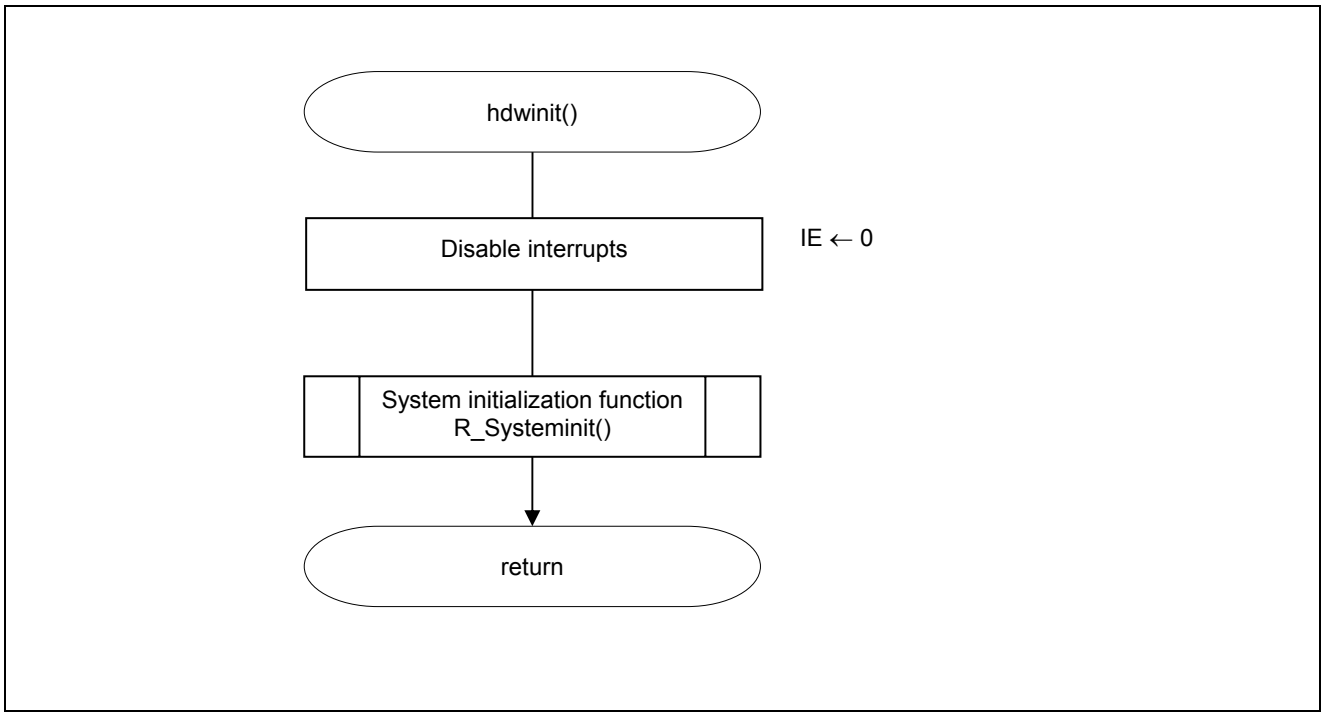

Figure 5.2 Initialization Function

5.10.2 System Initialization Function

Figure 5.3 shows the flowchart for the system initialization function.

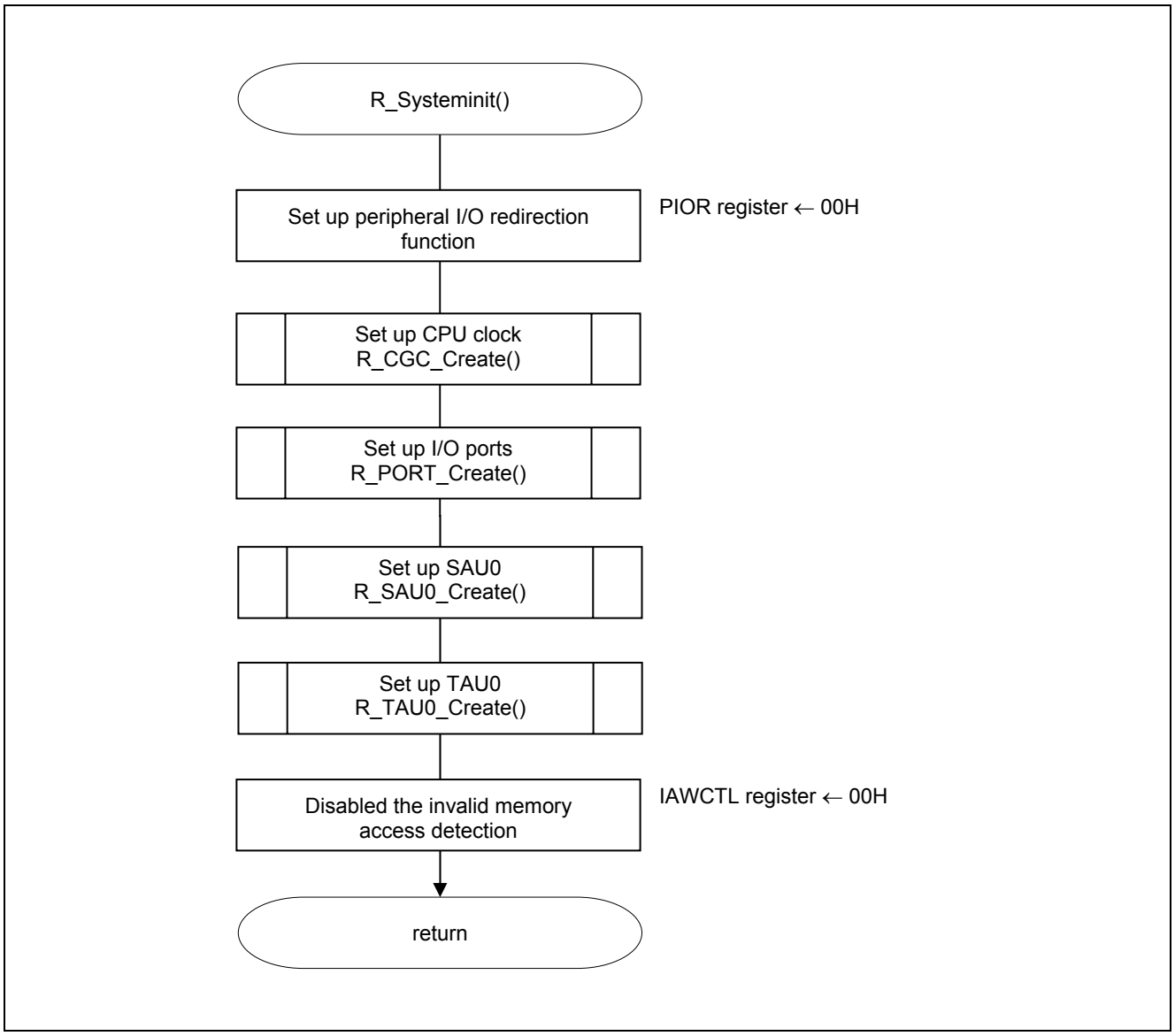

Figure 5.3 System Initialization Function

5.10.3 I/O Port Setup

Figure 5.4 shows the flowchart for I/O port setup.

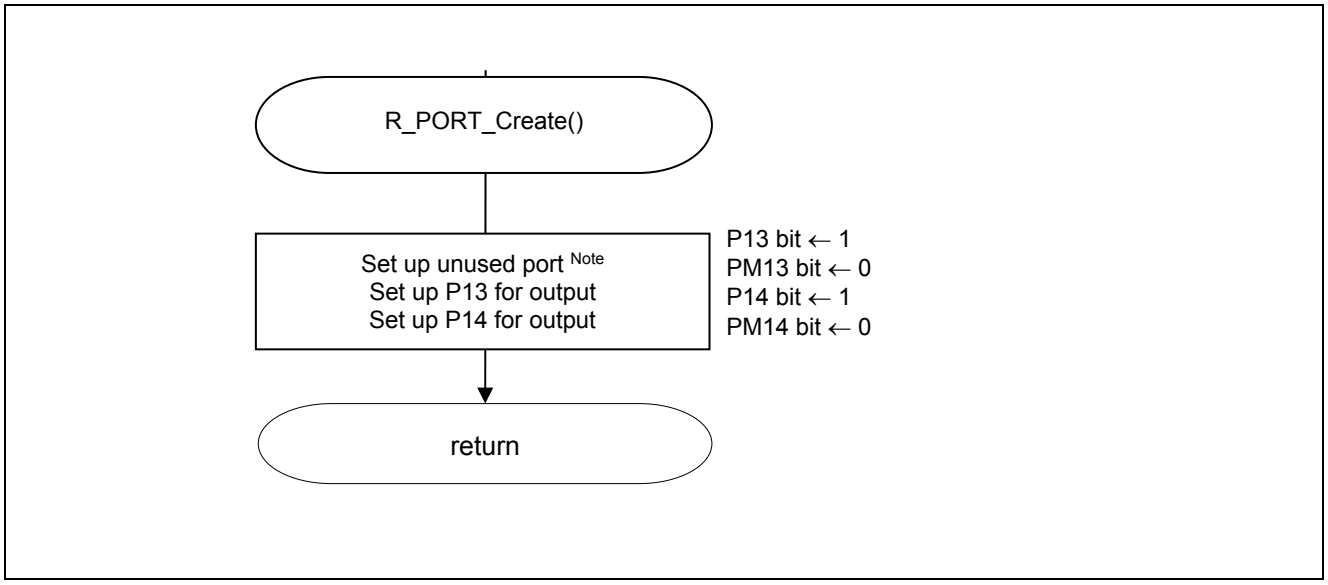

Figure 5.4 I/O Port Setup

Note: Refer to the section entitled "Flowcharts" in RL78/G12 Initialization (R01AN2582E) Application Note for the configuration of the unused ports.

Caution: Provide proper treatment for unused pins so that their electrical specifications are observed. Connect each of any unused input-only ports to V_{DD} or V_{SS} via a separate resistor.

5.10.4 CPU Clock Setup

Figure 5.5 shows the flowchart for CPU clock setup.

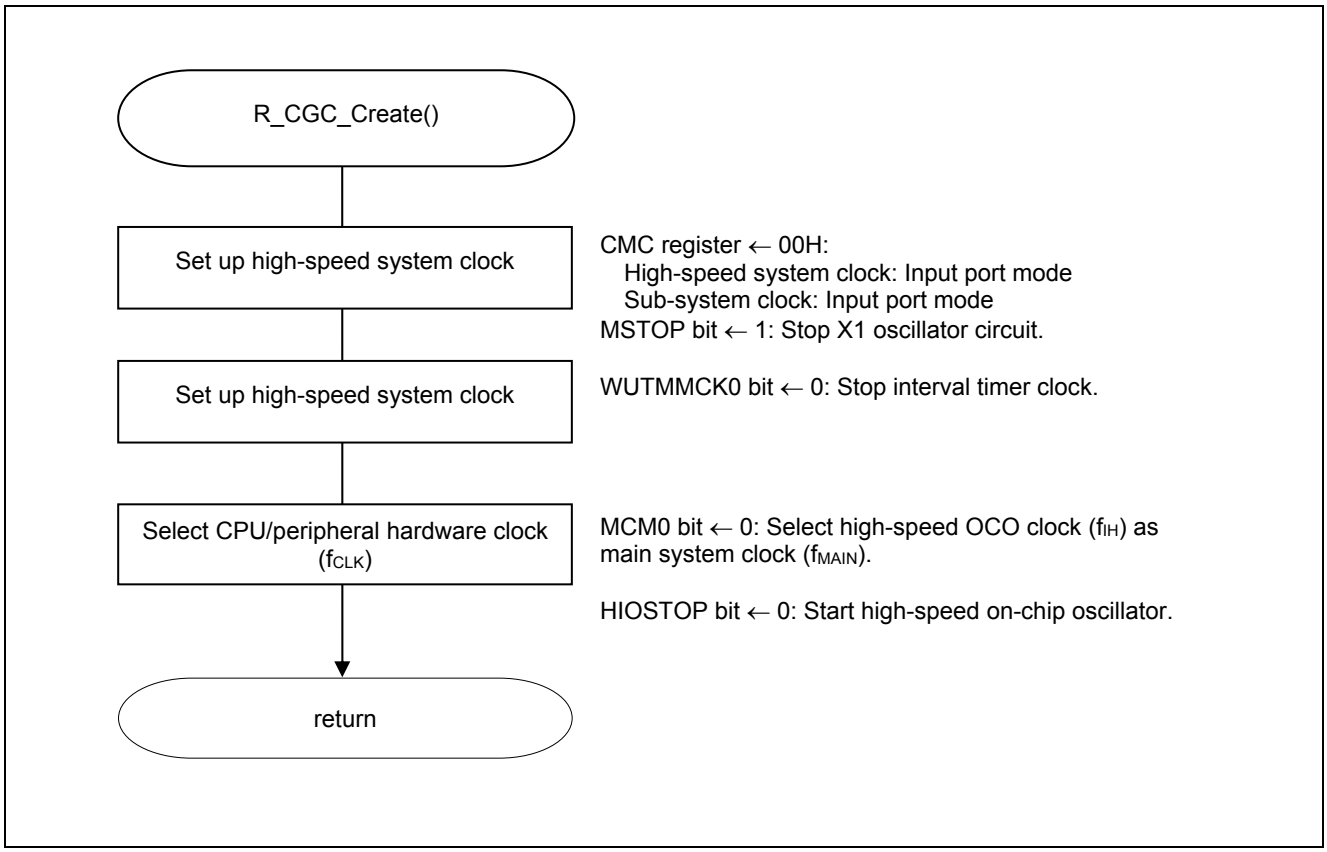

Figure 5.5 CPU Clock Setup

Caution: For details on the procedure for setting up the CPU clock (R_CGC_Create ()), refer to the section entitled "Flowcharts" in RL78/G12 Initialization (R01AN2582E) Application Note.

5.10.5 SAU0 Setup

Figure 5.6 shows the flowchart for SAU0 setup.

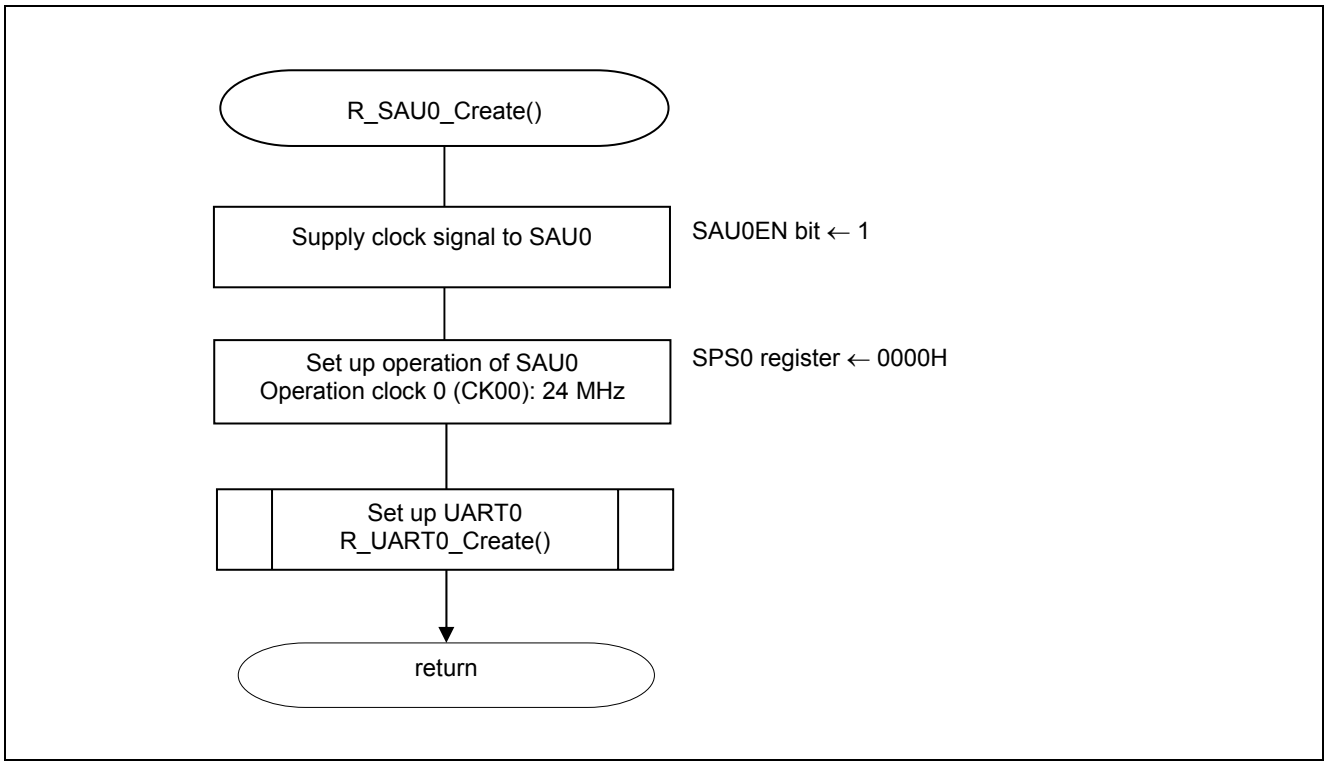

Figure 5.6 SAU0 Setup

5.10.6 UART0 Setup

Figure 5.7 shows the flowchart for UART0 setup (1/3). Figure 5.8 shows the flowchart for UART0 setup (2/3). Figure 5.9 shows the flowchart for UART0 setup (3/3).

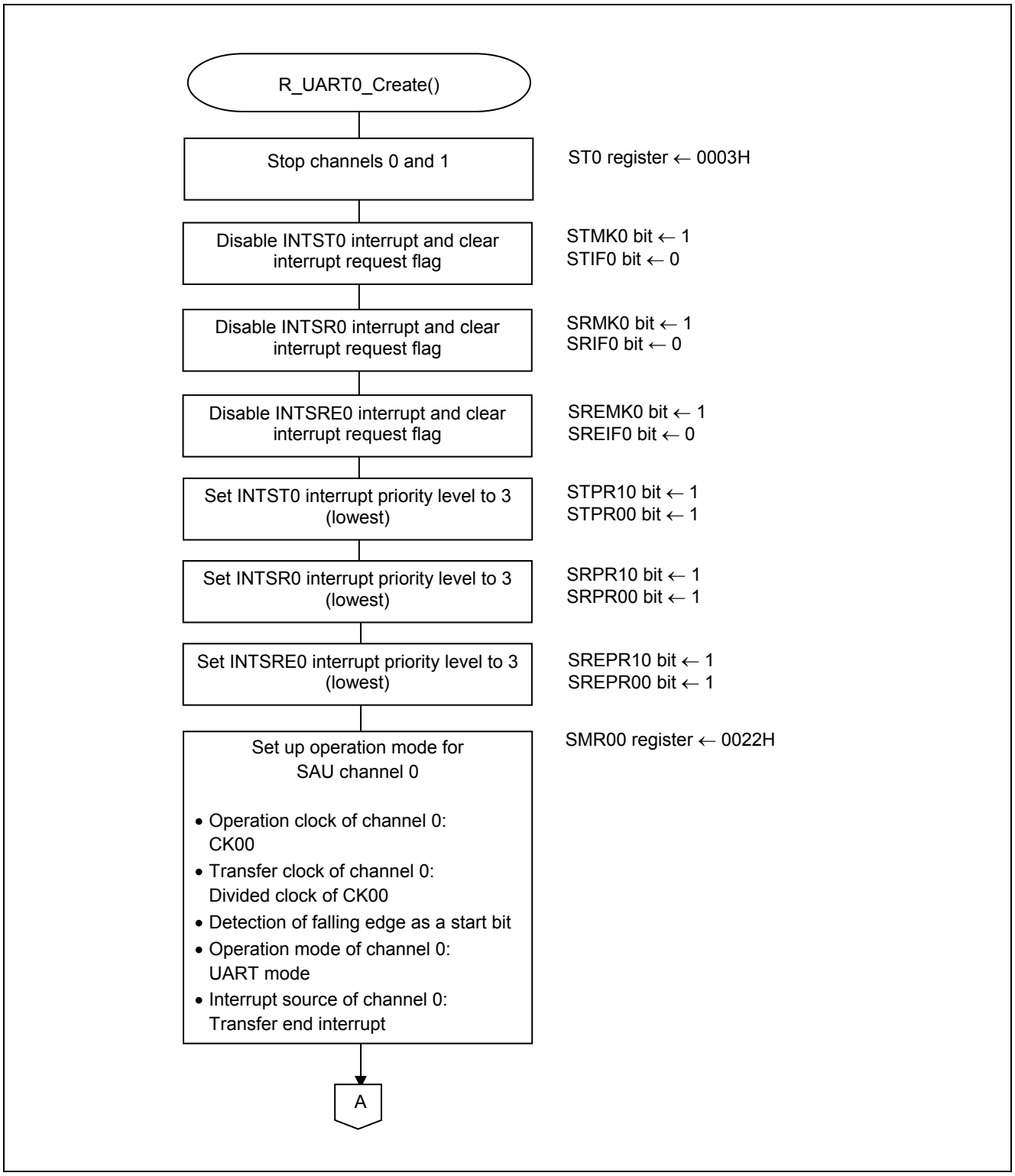

Figure 5.7 UART0 Setup (1/3)

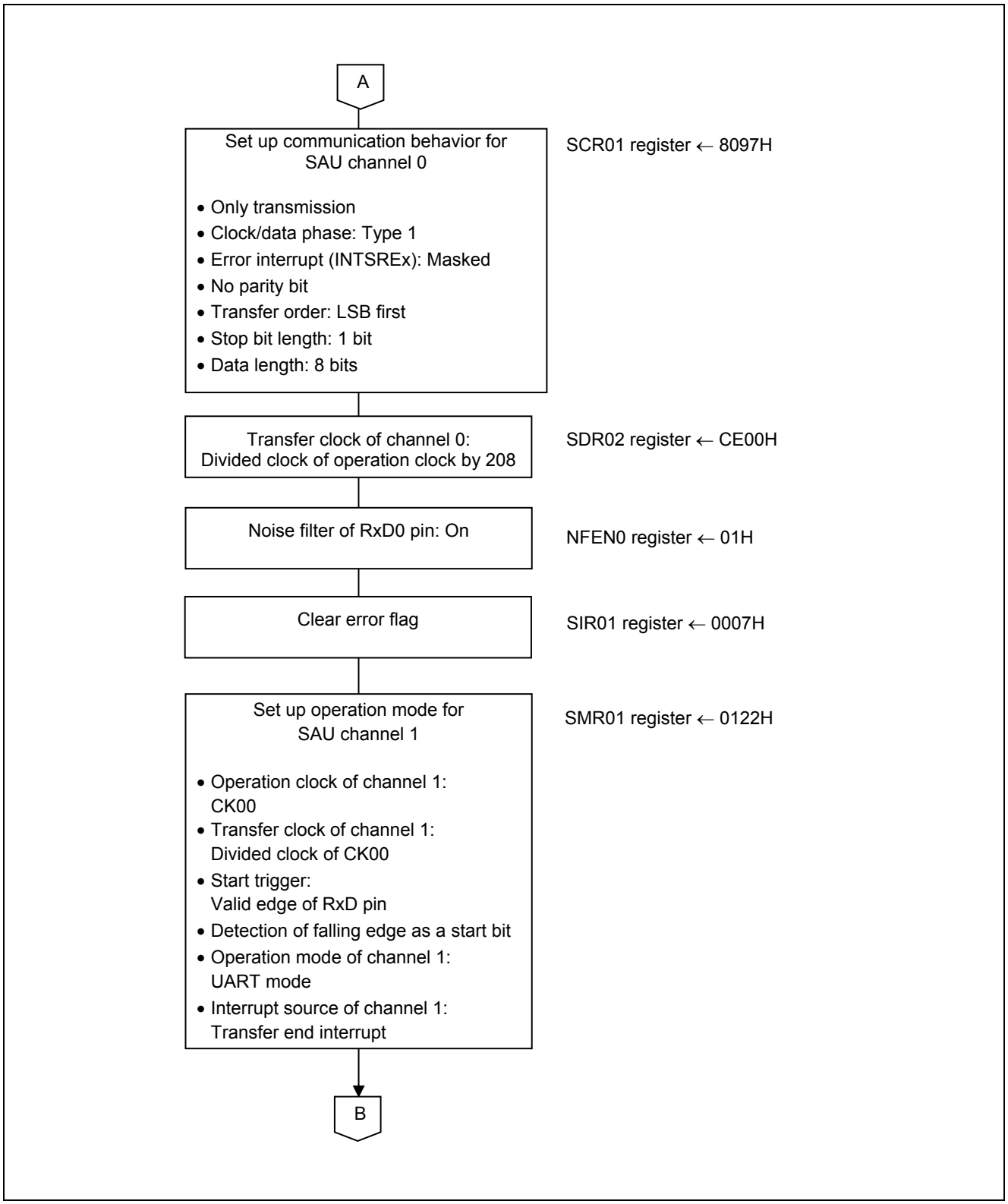

Figure 5.8 Setup UART0 (2/3)

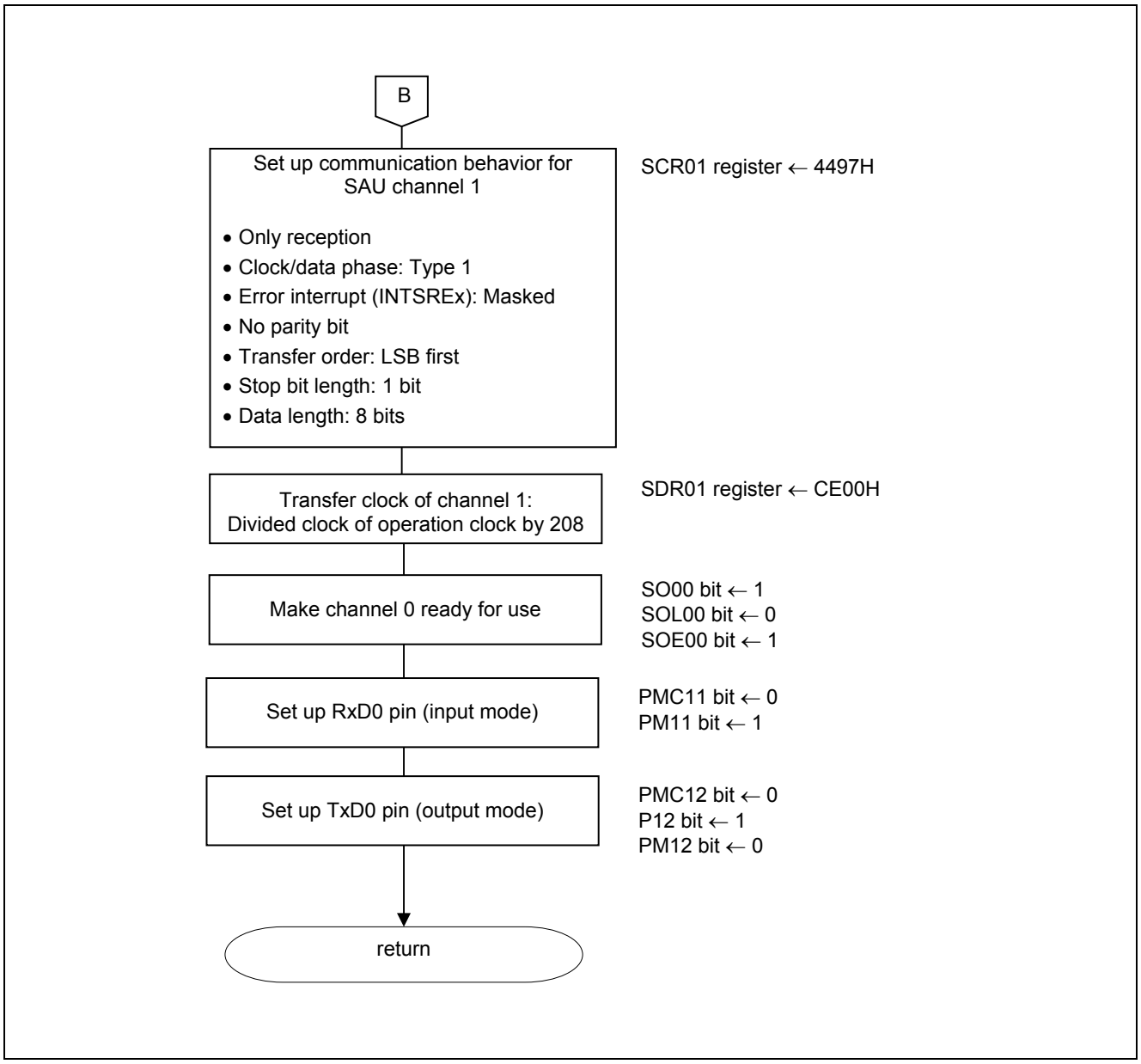

Figure 5.9 Setup UART0 (3/3)

5.10.7 TAU0 Setup

Figure 5.10 shows the flowchart for TAU0 setup.

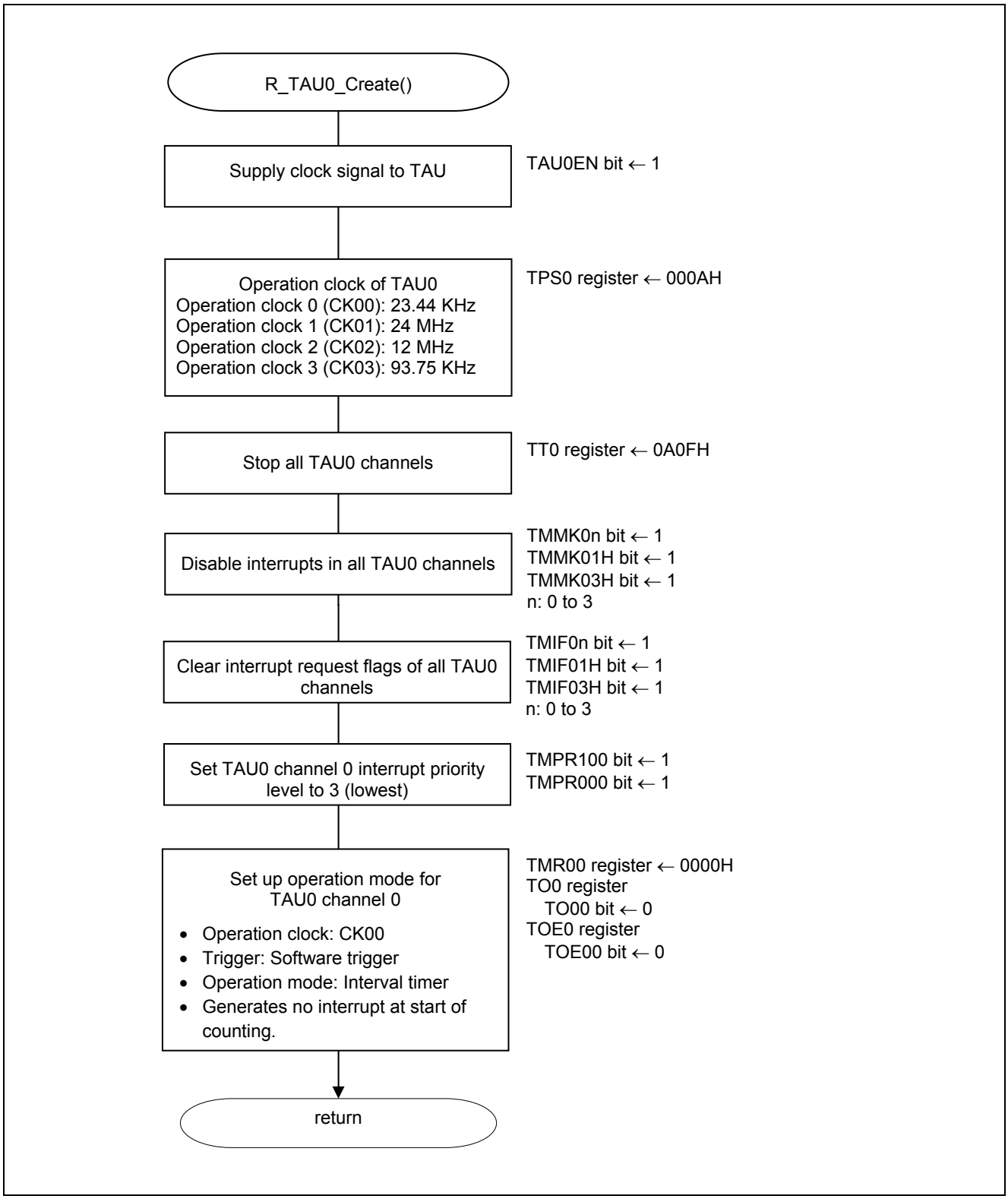

Figure 5.10 TAU0 Setup

5.10.8 Main Processing

Figure 5.11 shows the flowchart for main processing (1/2). Figure 5.12 shows the flowchart for main processing (2/2).

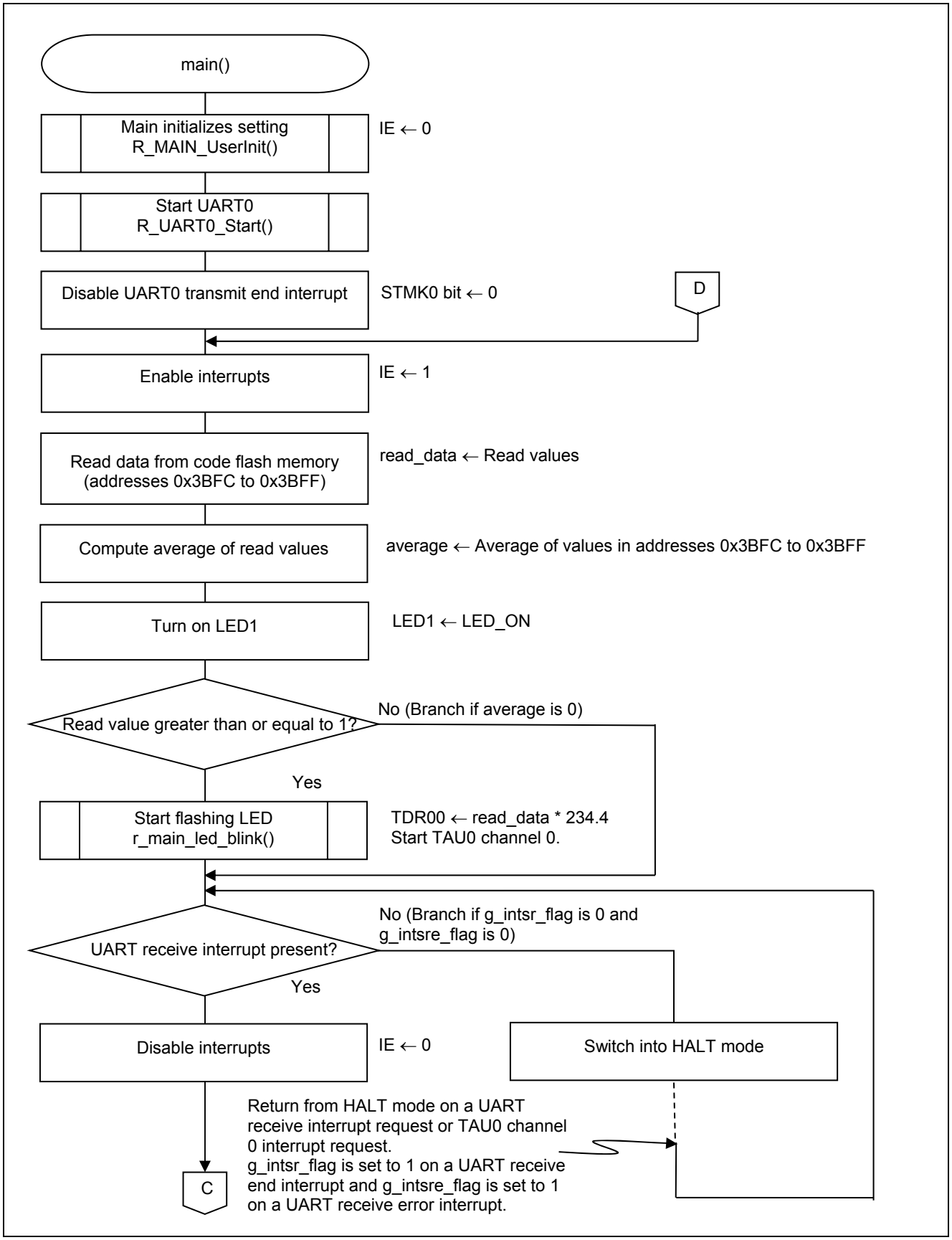

Figure 5.11 Main Processing (1/2)

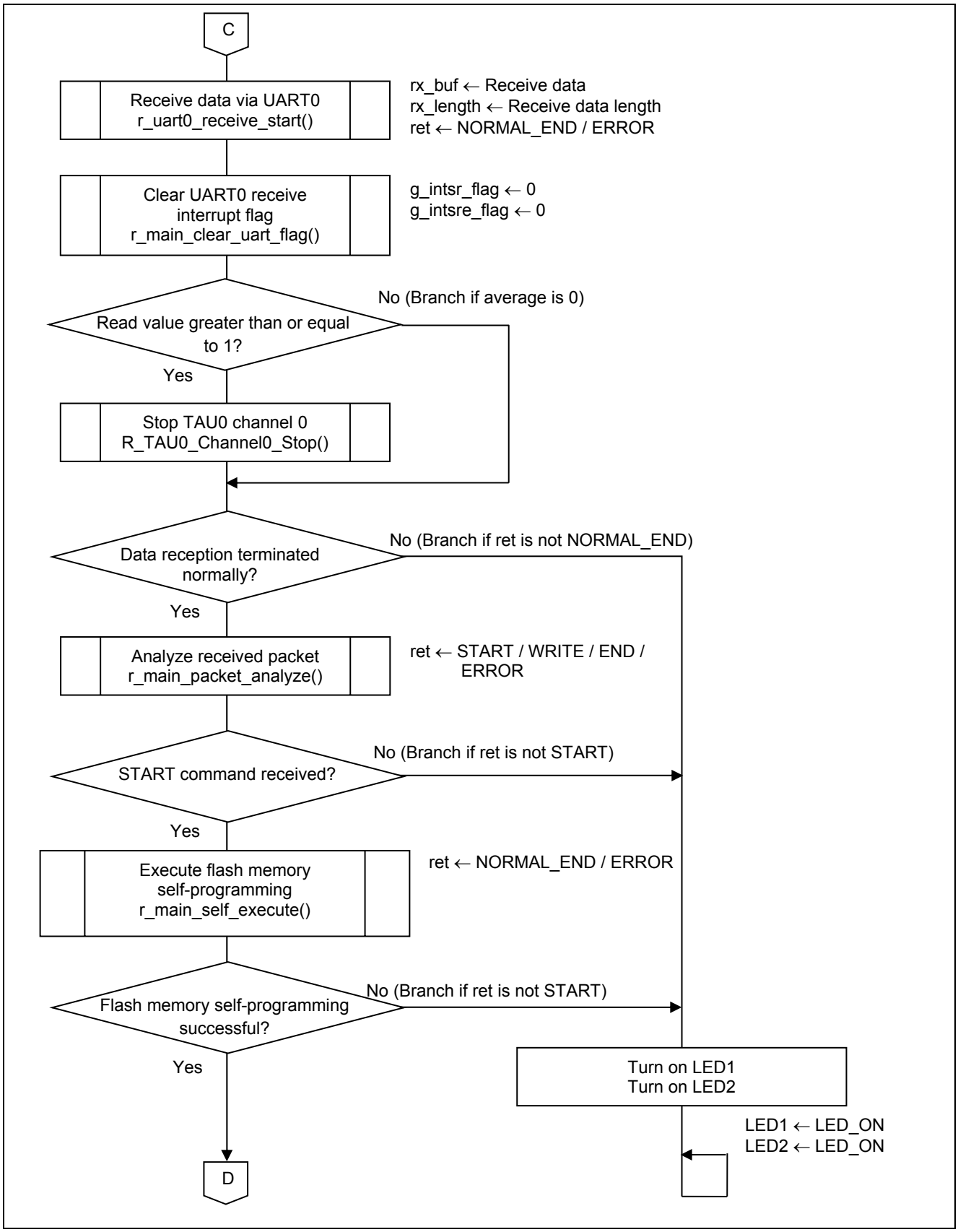

Figure 5.12 Main Processing (2/2)

5.10.9 Main initializes settings

Figure 5.13 shows the flowchart for the main initializes settings.

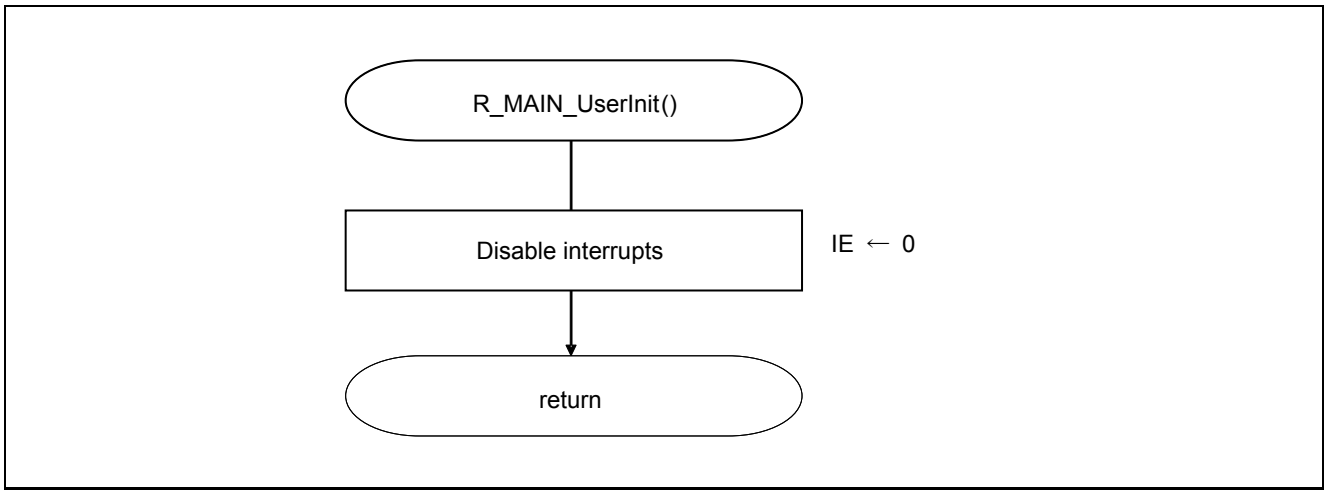

Figure 5.13 Main initializes settings

5.10.10 Starting the UART0

Figure 5.14 shows the flowchart for starting the UART0.

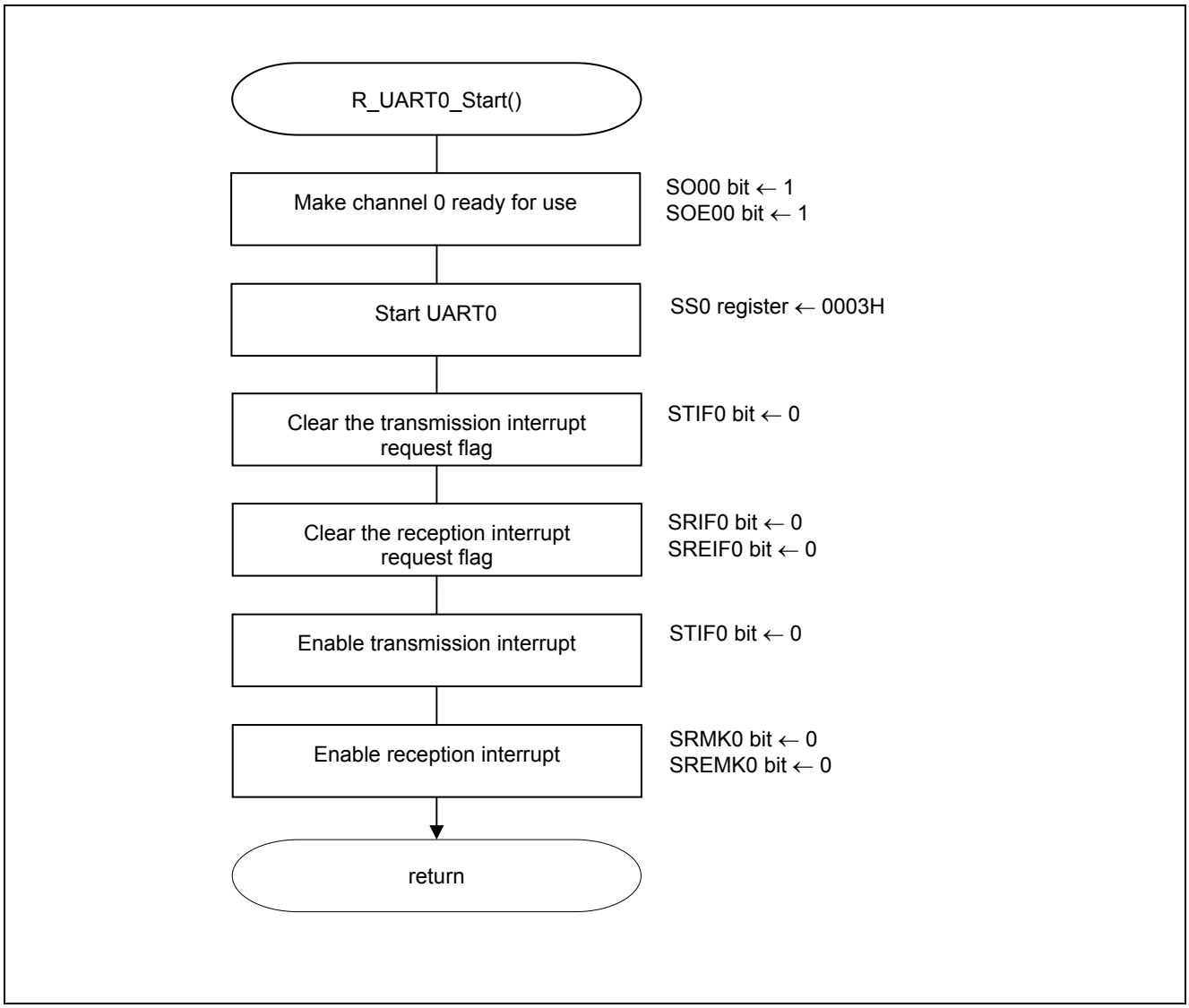

Figure 5.14 Starting the UART0

5.10.11 UART0 Receive End Interrupt

Figure 5.15 shows the flowchart for UART0 receive end interrupt.

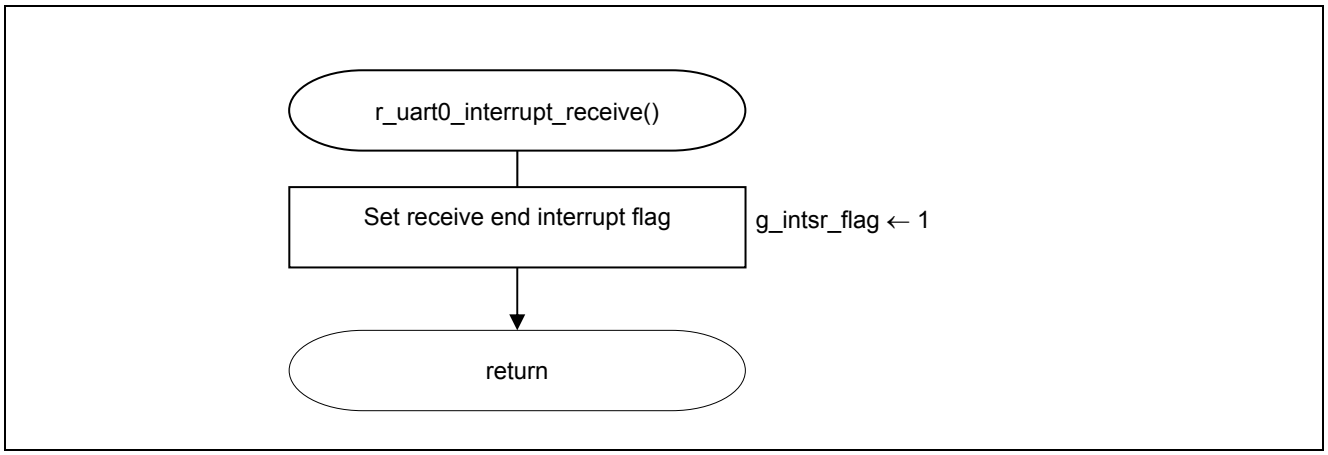

Figure 5.15 UART0 Receive End Interrupt

5.10.12 UART0 Receive Error Interrupt

Figure 5.16 shows the flowchart for UART0 receive error interrupt.

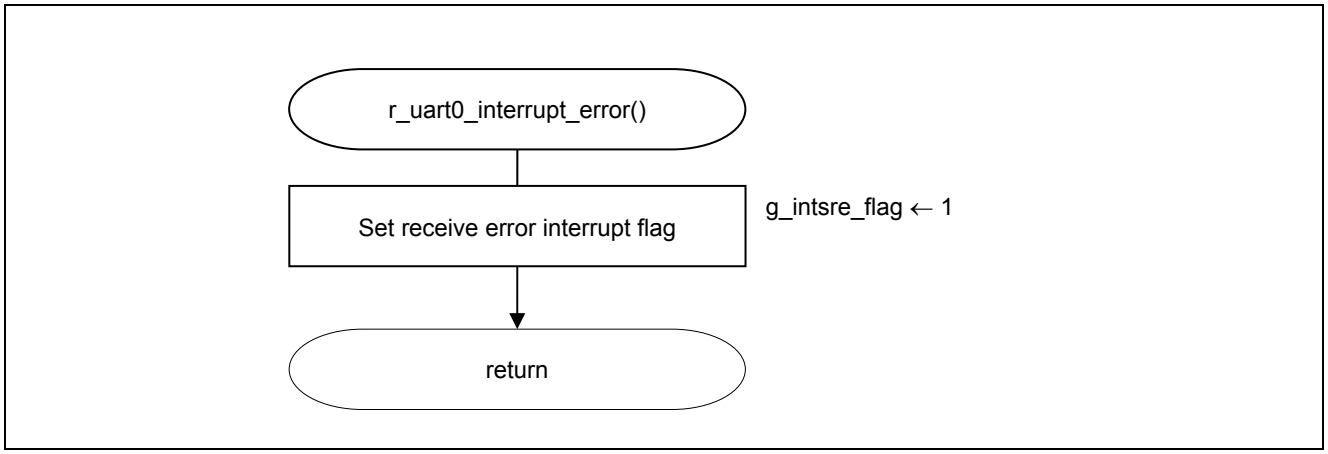

Figure 5.16 UART0 Receive Error Interrupt

5.10.13 Starting to Flash the LED

Figure 5.17 shows the flowchart for starting to flash the LED.

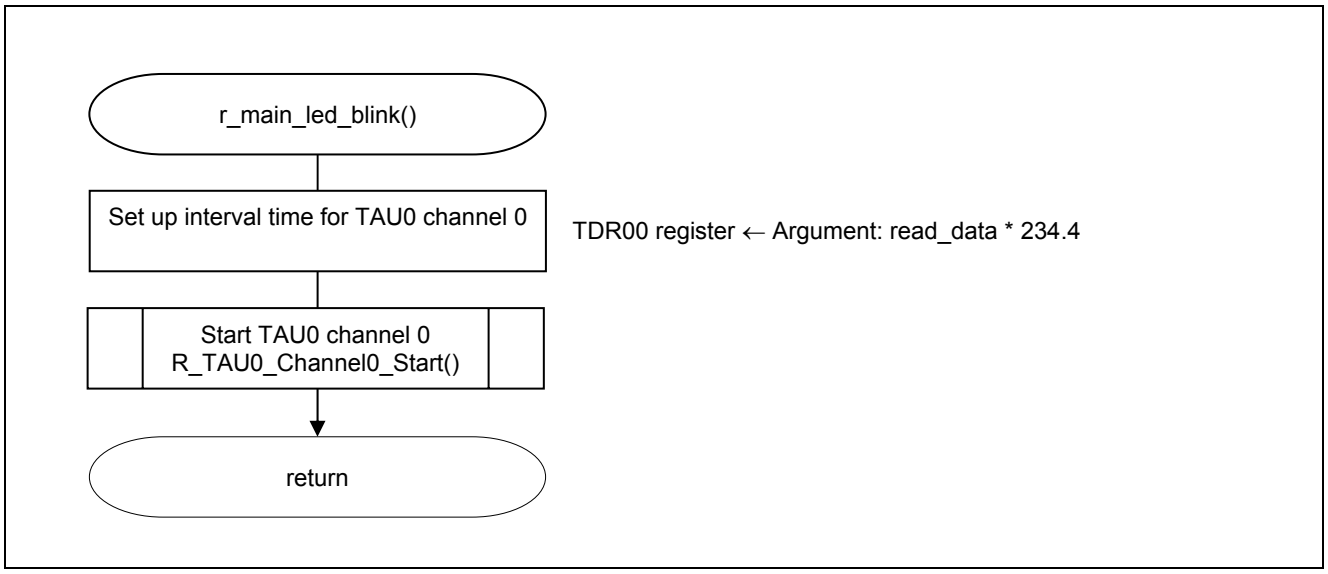

Figure 5.17 Starting to Flash the LED

5.10.14 Starting the TAU0 Channel 0

Figure 5.18 shows the flowchart for starting the TAU0 channel 0.

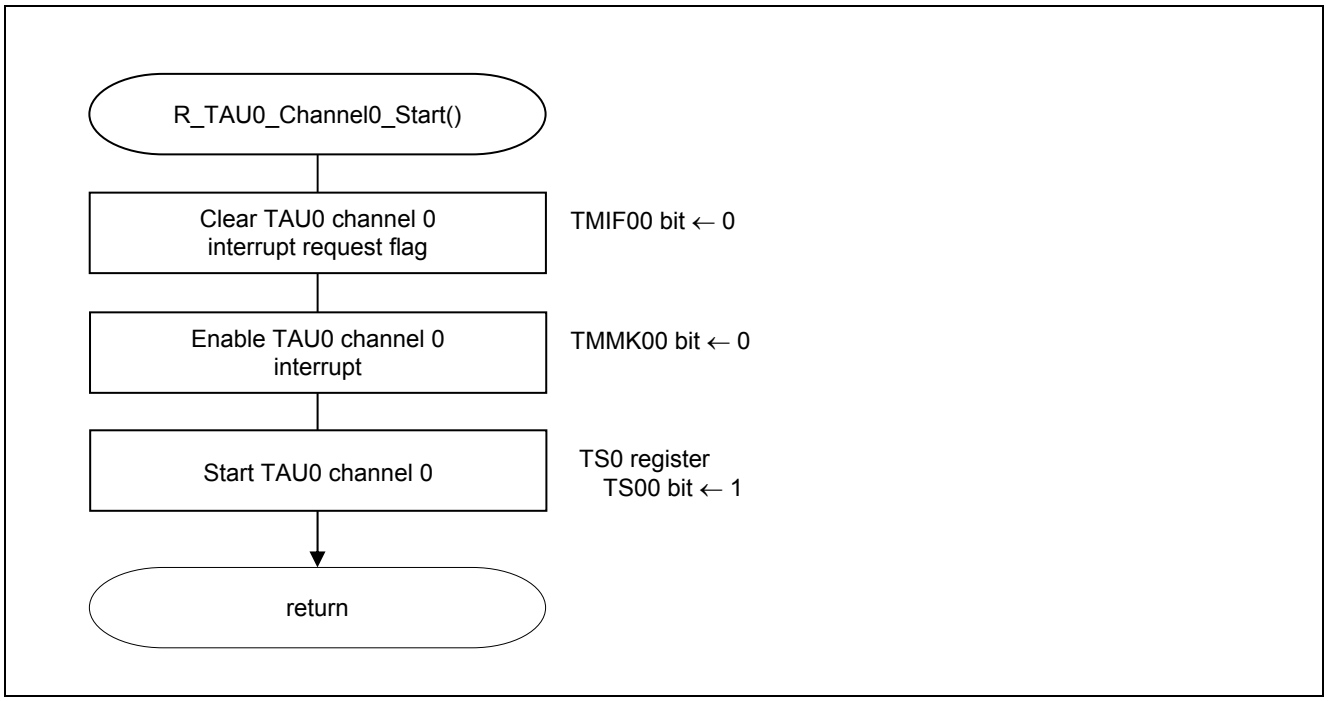

Figure 5.18 Starting the TAU0 Channel 0

5.10.15 TAU0 Channel 0 Interrupt

Figure 5.19 shows the flowchart for TAU0 channel 0 interrupt.

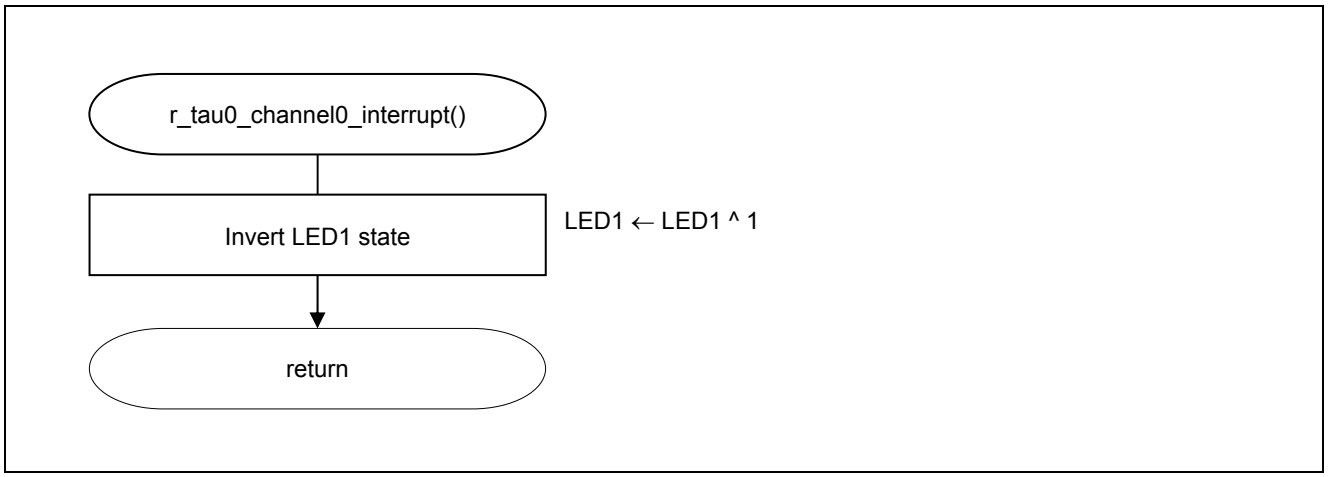

Figure 5.19 TAU0 Channel 0 Interrupt

5.10.16 Data Reception via UART0

Figure 5.20 shows the flowchart for data reception via the UART0 (1/2). Figure 5.21 shows the flowchart for data reception via the UART0 (2/2).

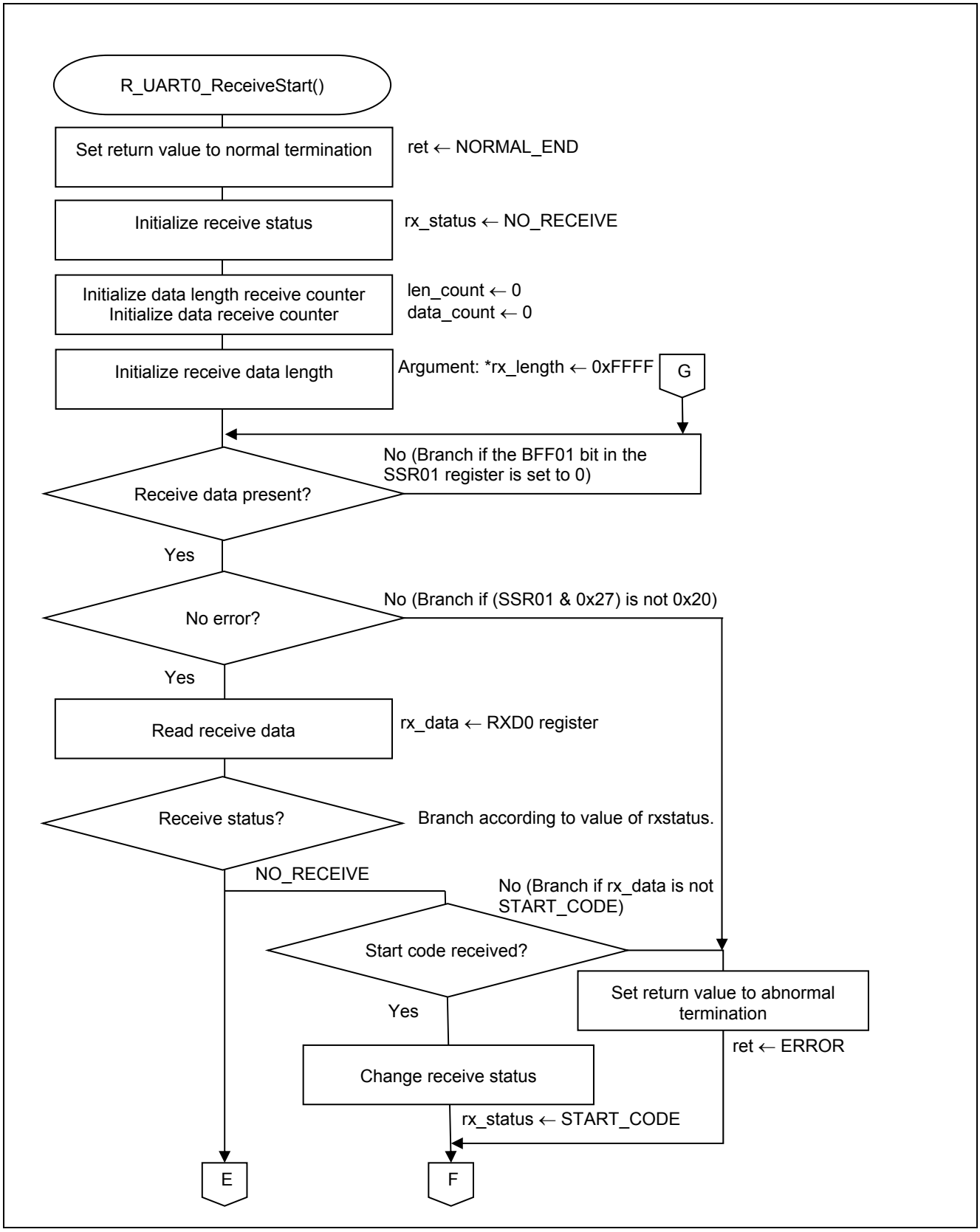

Figure 5.20 Data Reception via UART0 (1/2)

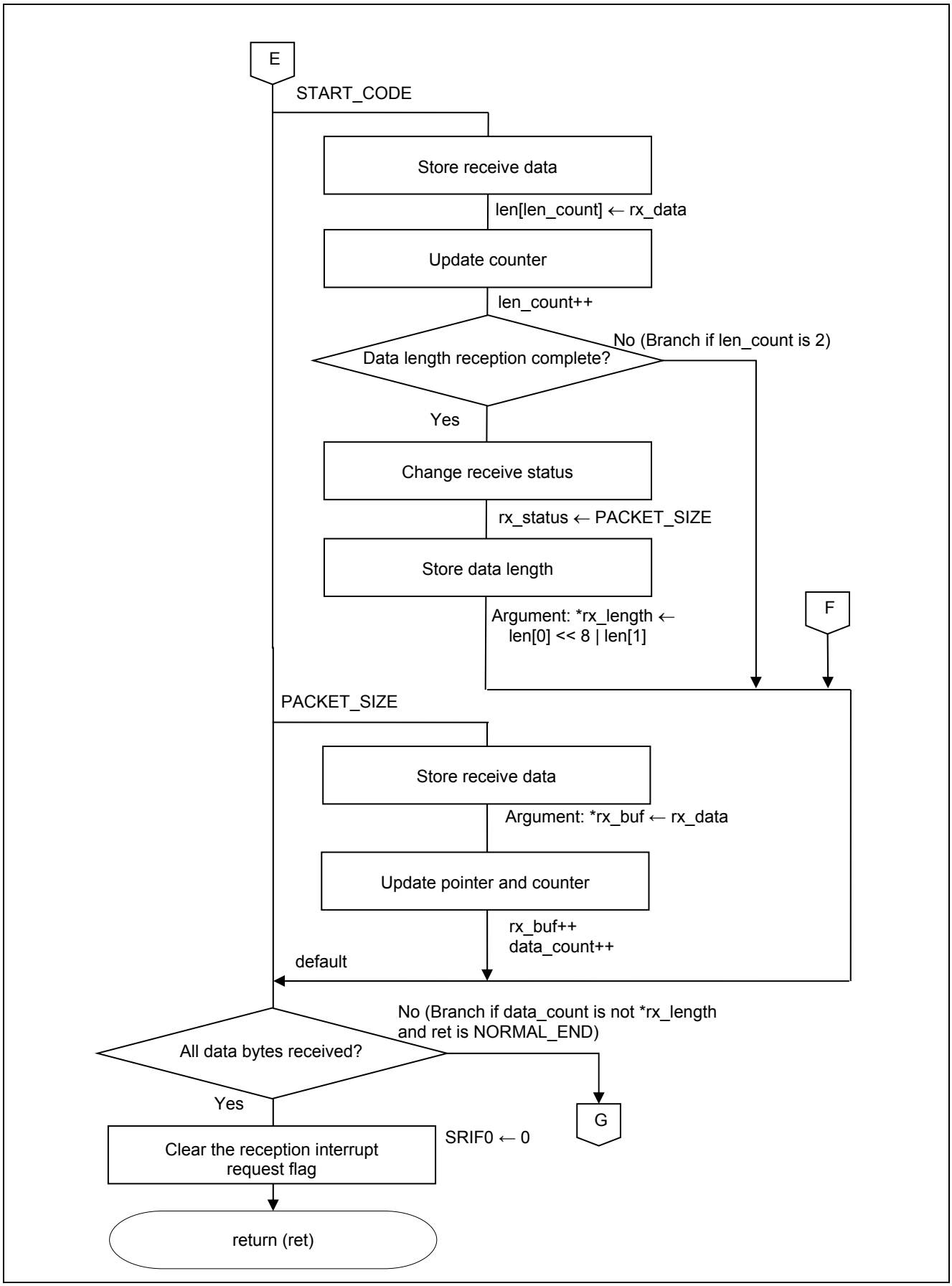

Figure 5.21 Data Reception via UART0 (2/2)

5.10.17 Clearing the UART0 Receive Interrupt Flag

Figure 5.22 shows the flowchart for clearing the UART0 receive interrupt flag.

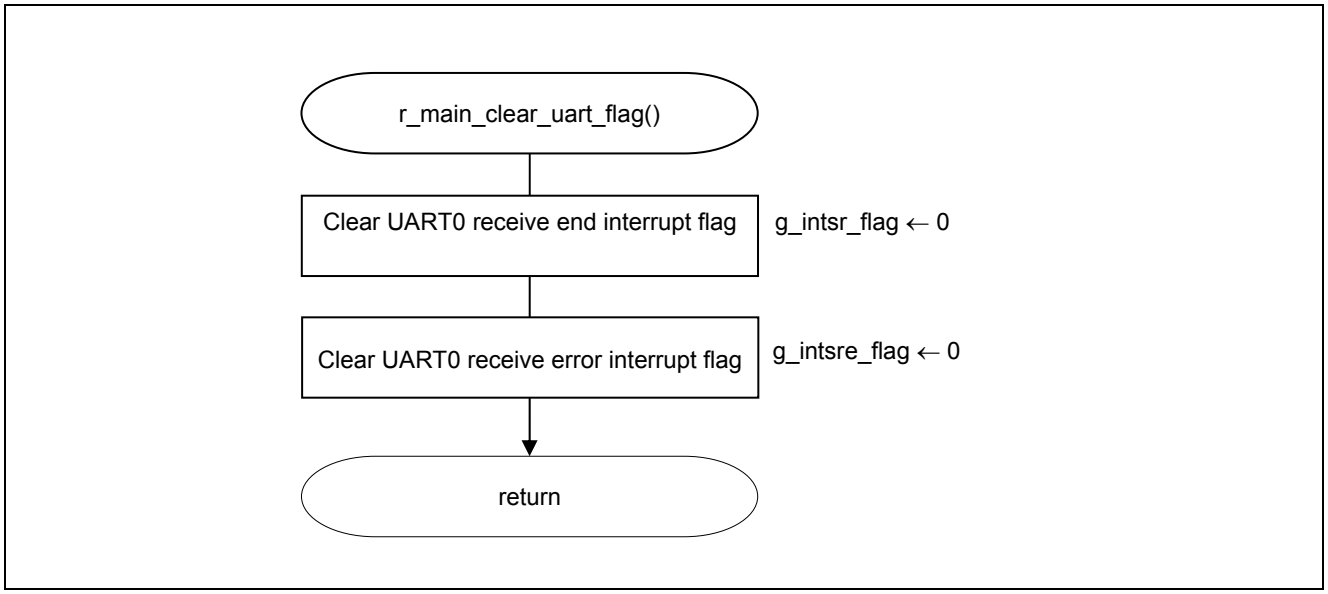

Figure 5.22 Clearing the UART0 Receive Interrupt Flag

5.10.18 Stopping the TAU0 Channel 0

Figure 5.23 shows the flowchart for stopping the TAU0 channel 0.

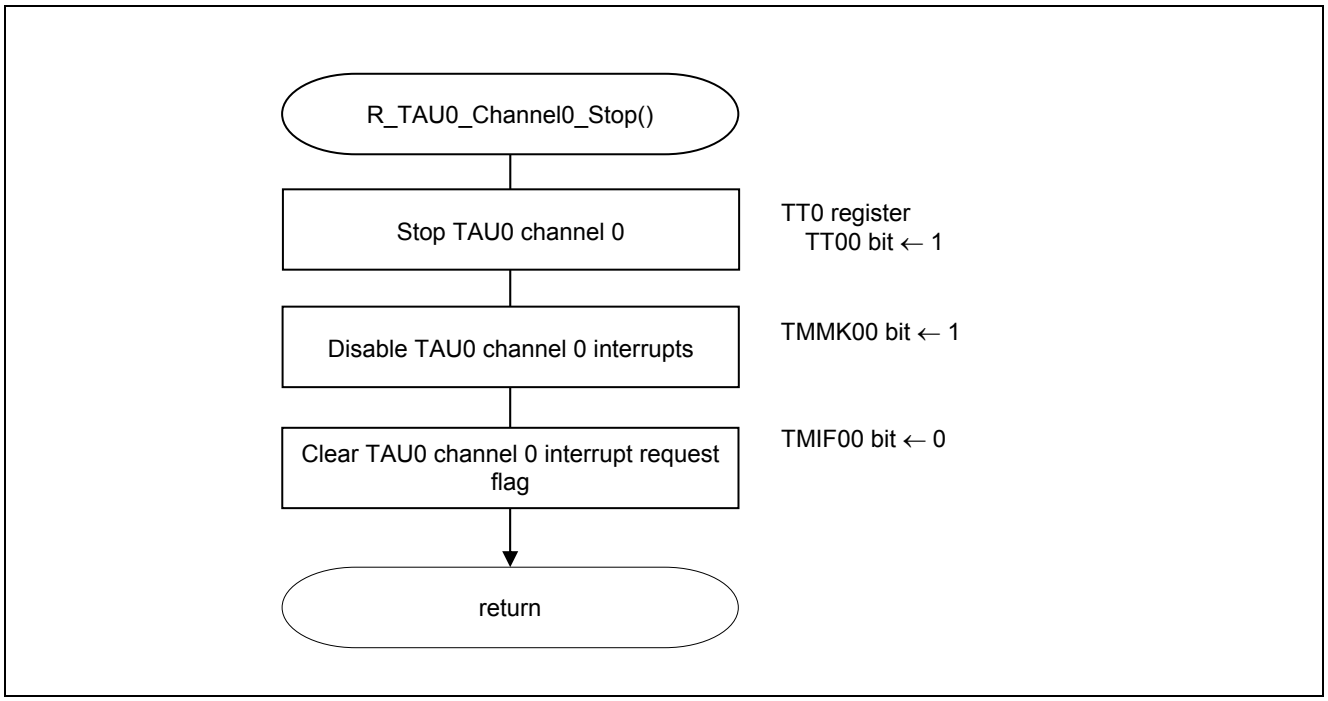

Figure 5.23 Stopping the TAU0 Channel 0

5.10.19 Receive Packet Analysis

Figure 5.24 shows the flowchart for receive packet analysis.

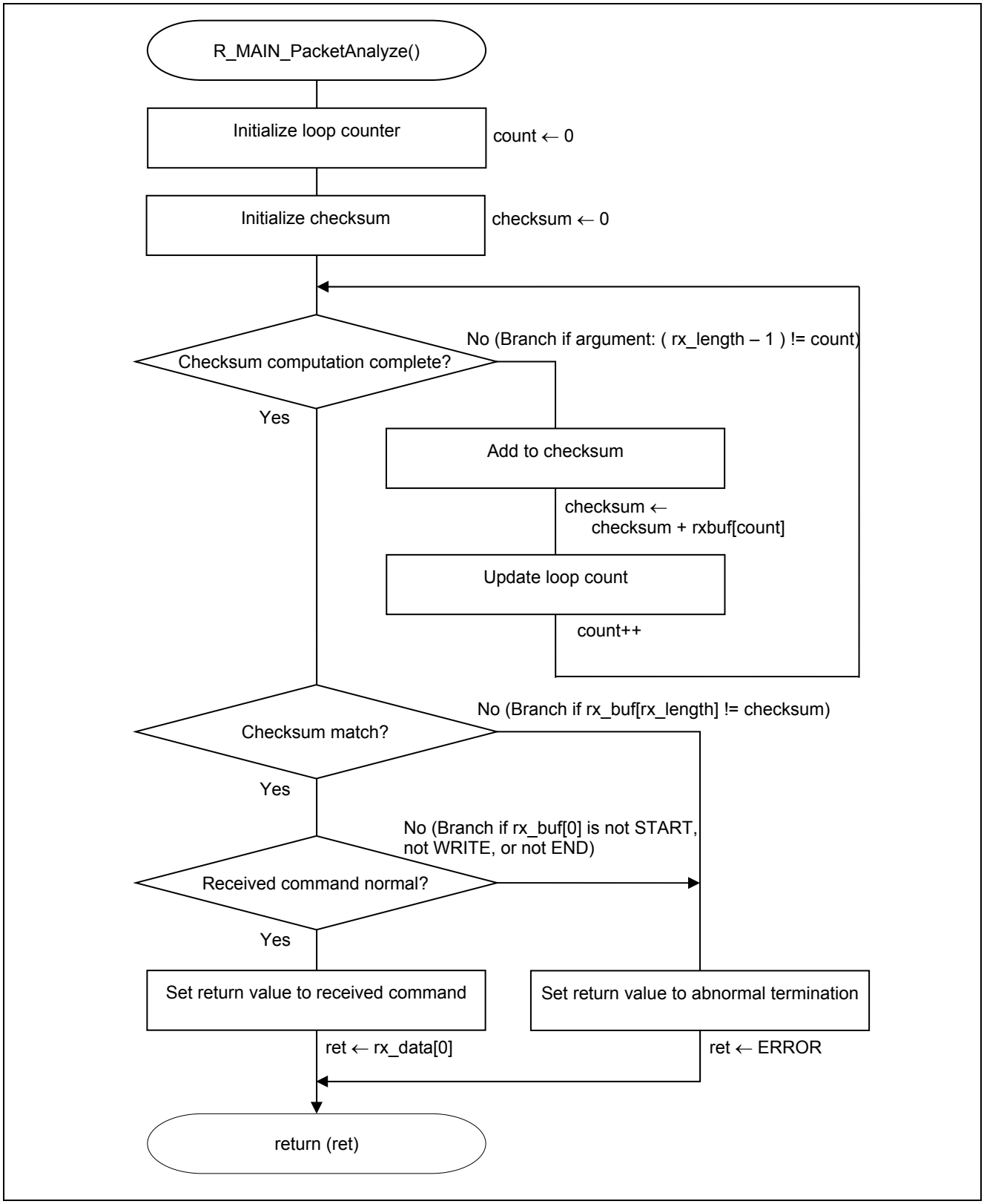

Figure 5.24 Receive Packet Analysis

5.10.20 Flash Memory Self-Programming Execution

Figure 5.25 shows the flowchart for flash memory self-programming execution.

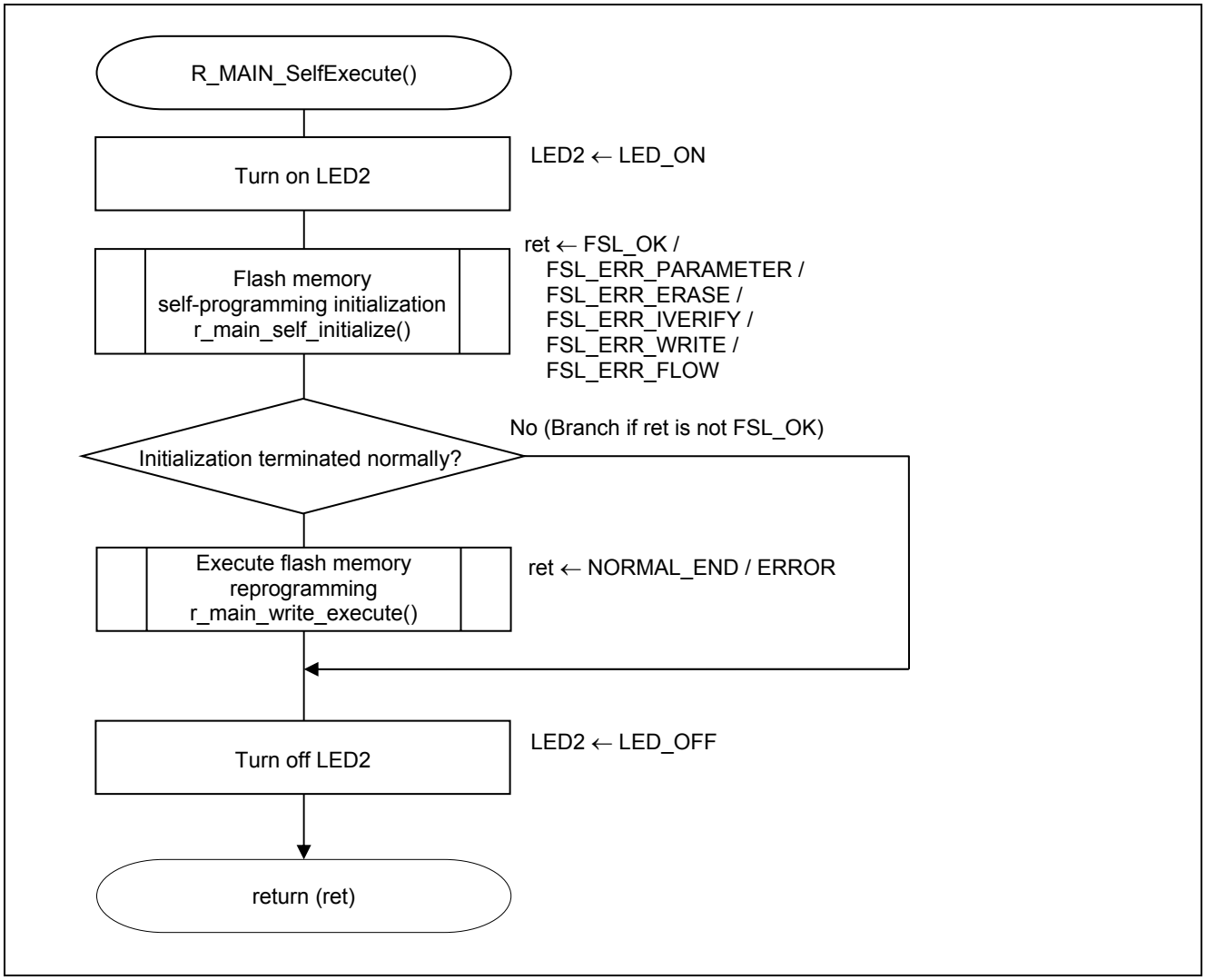

Figure 5.25 Flash Memory Self-Programming Execution

5.10.21 Flash Memory Self-Programming Initialization

Figure 5.26 shows the flowchart for flash memory self-programming initialization.

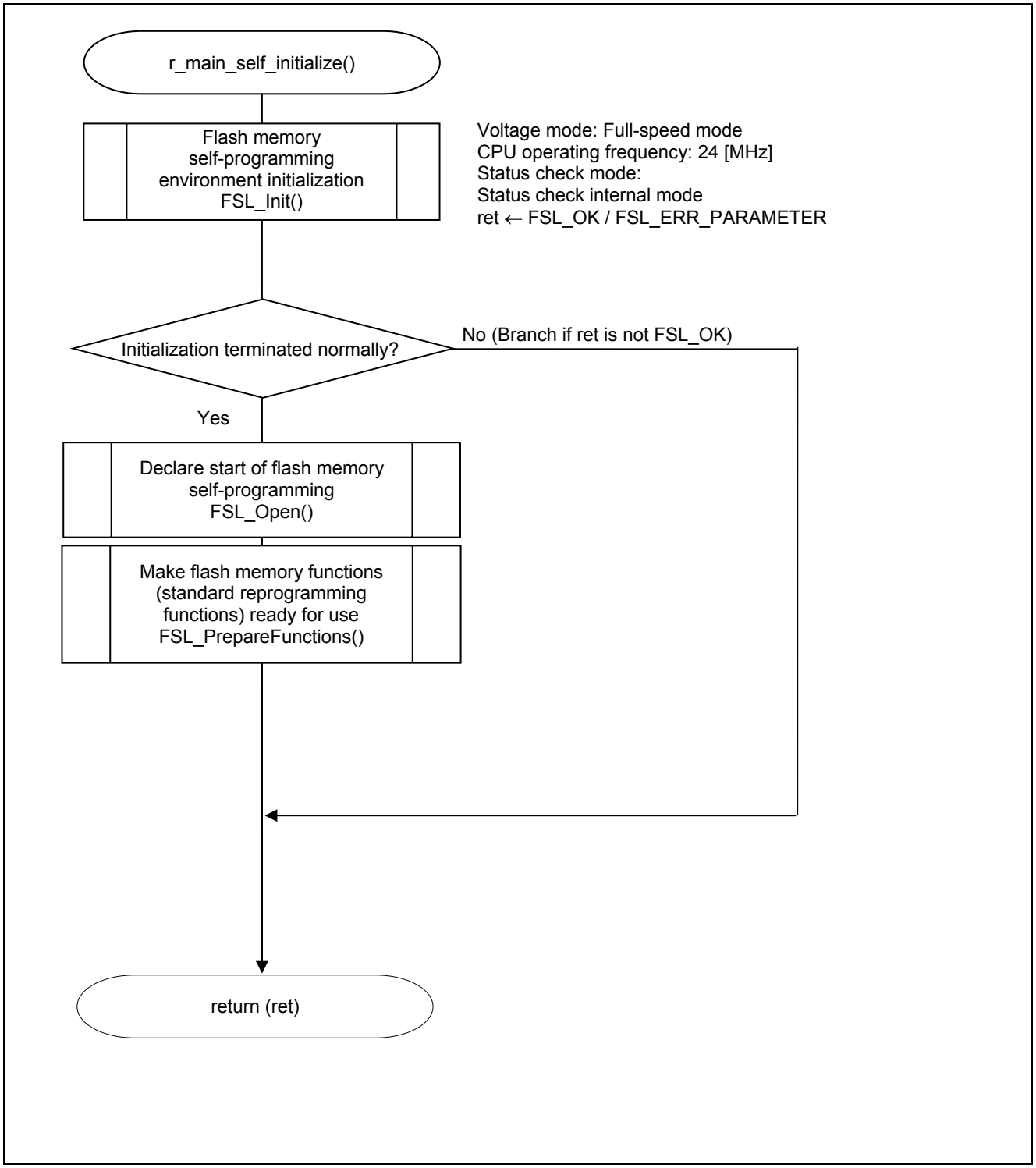

Figure 5.26 Flash Memory Self-Programming Initialization

5.10.22 Flash Memory Reprogramming Execution

Figure 5.27 shows the flowchart for flash memory reprogramming execution (1/3). Figure 5.28 shows the flowchart for flash memory reprogramming execution (2/3). Figure 5.29 shows the flowchart for flash memory reprogramming execution (3/3).

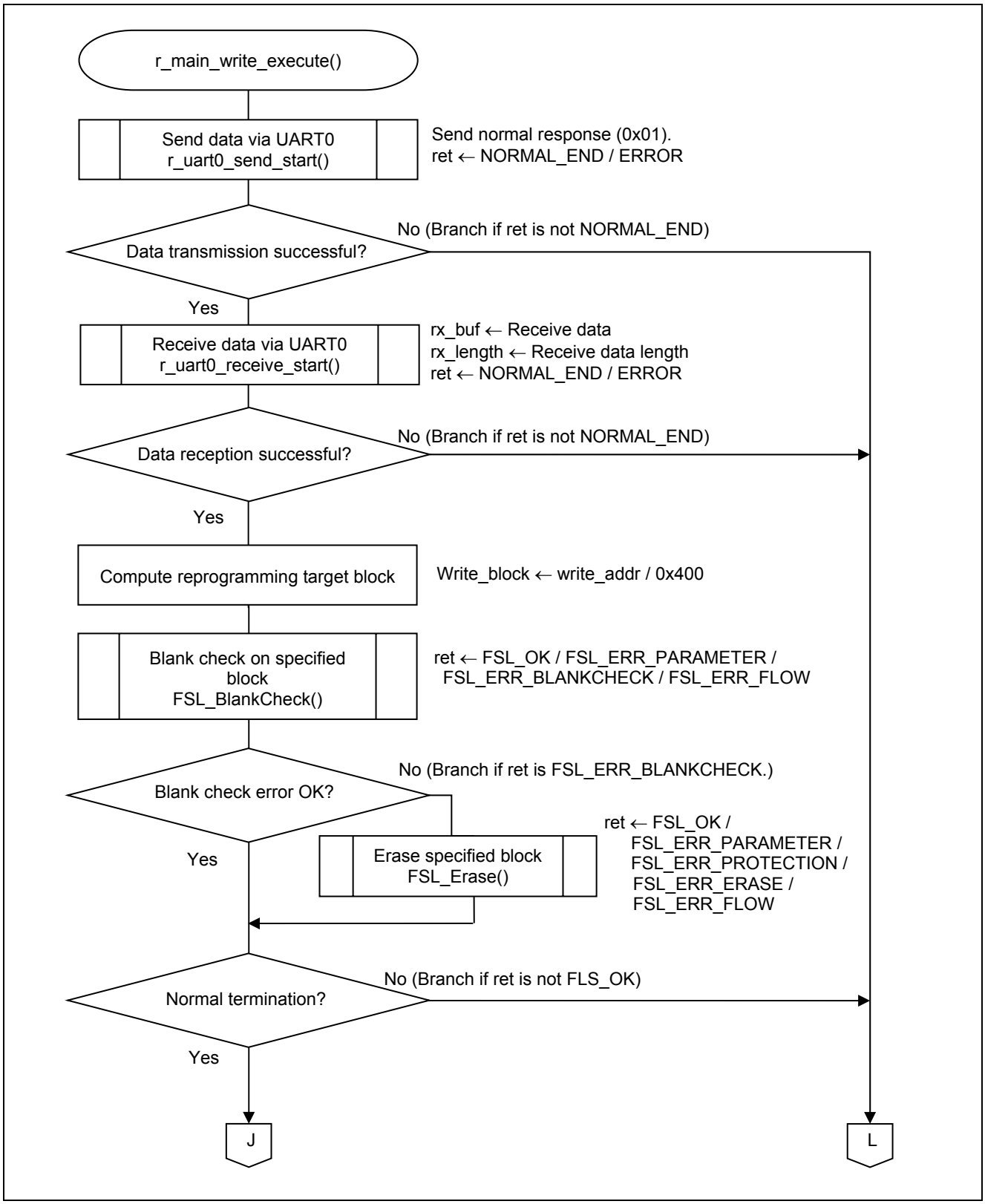

Figure 5.27 Flash Memory Reprogramming Execution (1/3)

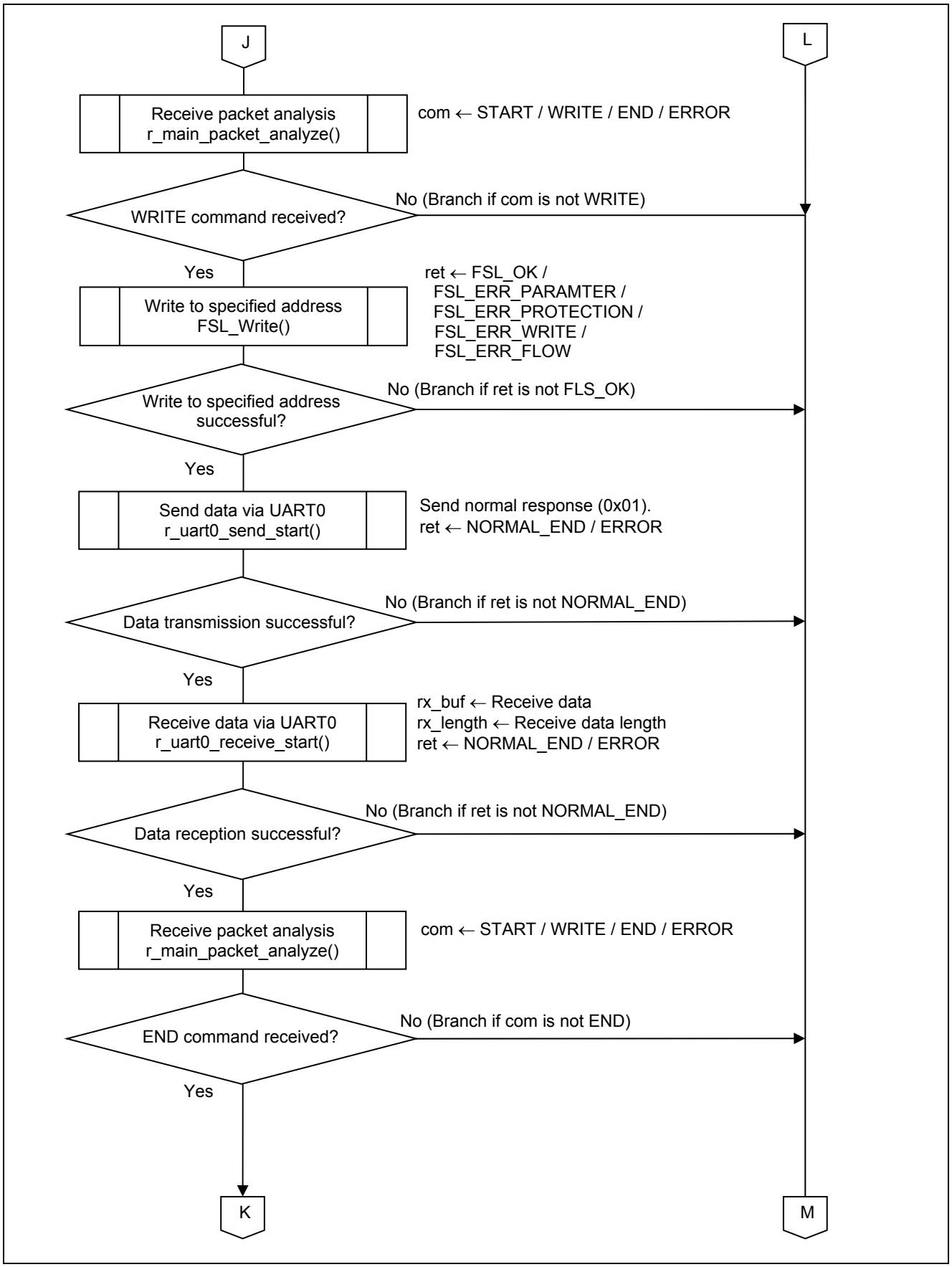

Figure 5.28 Flash Memory Reprogramming Execution (2/3)

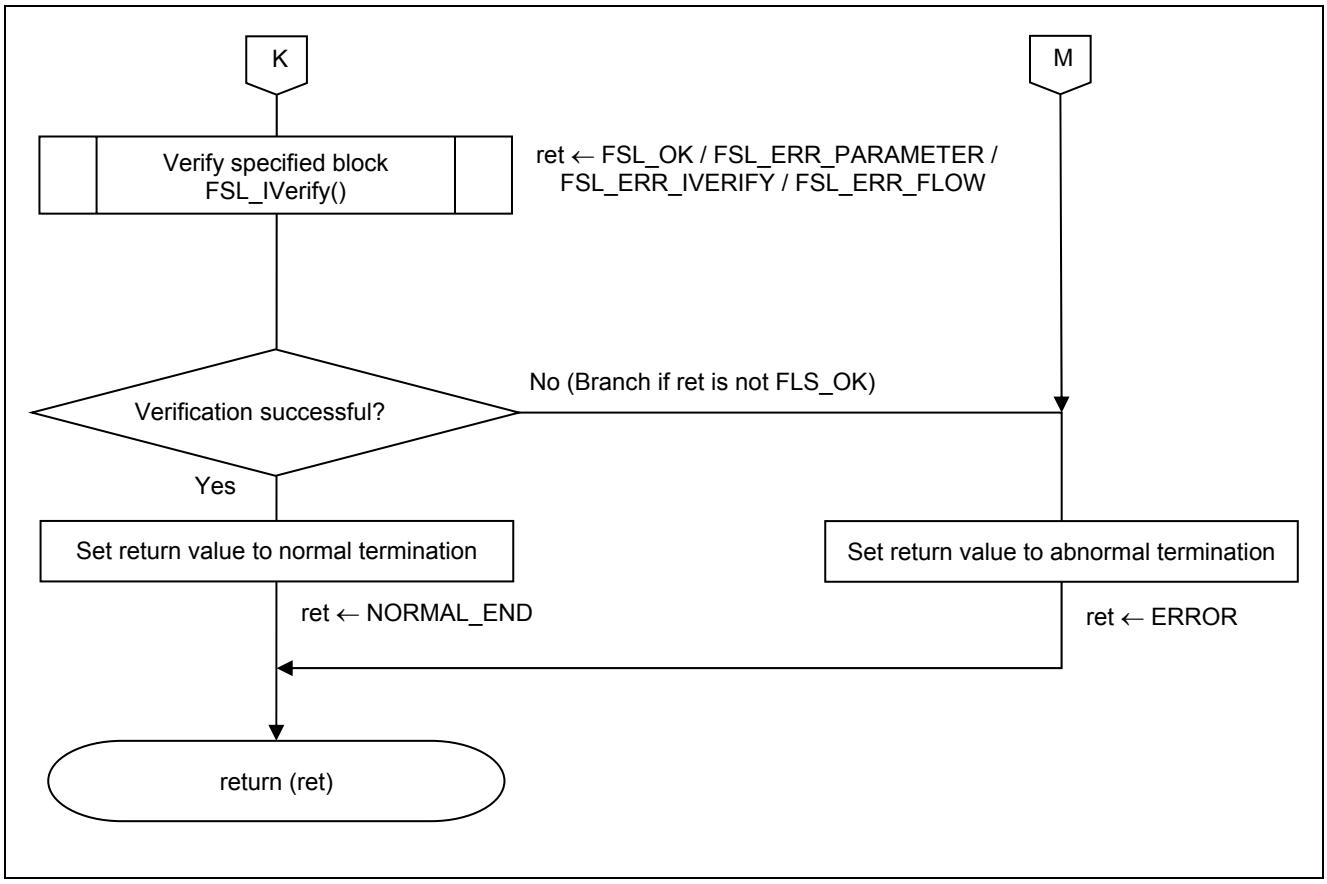

Figure 5.29 Flash Memory Reprogramming Execution (3/3)

5.10.23 Data Transmission via UART0

Figure 5.30 shows the flowchart for data transmission via the UART0.

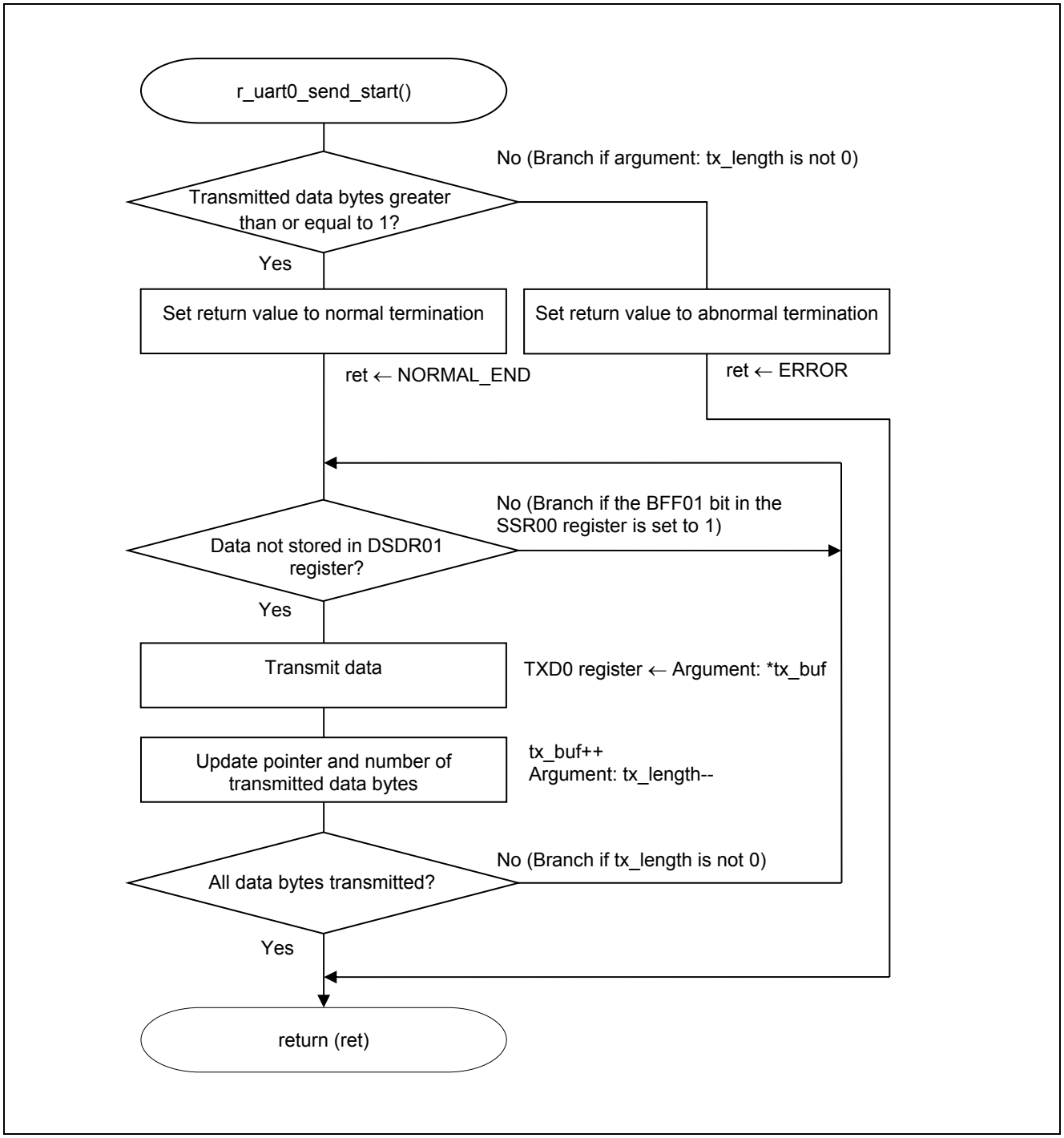

Figure 5.30 Data Transmission via UART0

6. Sample Code

The sample code is available on the Renesas Electronics Website.

7. Documents for Reference

RL78/G12 User's Manual: Hardware (R01UH0200E)

RL78 Family User's Manual: Software (R01US0015E)

RL78 family’s Flash Self Programming Library Type01 User’s Manual (R01US0050E)

(The latest versions of the documents are available on the Renesas Electronics Website.)

Technical Updates/Technical Brochures

(The latest versions of the documents are available on the Renesas Electronics Website.)

Website and Support

Renesas Electronics Website

- http://www.renesas.com/index.jsp

Inquiries

- http://www.renesas.com/contact/

Revision Record	RL78/G12 Self-Programming (Received Data via UART) CC-RL

Rev.	Date	Description	
		Page	Summary
1.00	Oct. 20, 2015	—	First edition issued
1.10	June 01, 2016	7	Modification of 1.4 How to Get the Flash Memory Self-Programming Library.
		53	Addition of Documents for Reference.

General Precautions in the Handling of MPU/MCU Products

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

1. Handling of Unused Pins

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

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

2. Processing at Power-on

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

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

3. Prohibition of Access to Reserved Addresses

   Access to reserved addresses is prohibited.

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

4. Clock Signals

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

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

5. Differences between Products

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

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

Notice

1. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
3. Renesas Electronics does not assume any liability for infringement of patents, copyrights, or other intellectual property rights of third parties by or arising from the use of Renesas Electronics products or technical information described in this document. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
4. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from such alteration, modification, copy or otherwise misappropriation of Renesas Electronics product.
5. Renesas Electronics products are classified according to the following two quality grades: "Standard" and "High Quality". The recommended applications for each Renesas Electronics product depends on the product's quality grade, as indicated below.
   "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; and industrial robots etc.
   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; and safety equipment etc.
   Renesas Electronics products are neither intended nor authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems, surgical implantations etc.), or may cause serious property damages (nuclear reactor control systems, military equipment etc.). You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application for which it is not intended. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for which the product is not intended by Renesas Electronics.
6. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
7. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or systems manufactured by you.
8. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
9. Renesas Electronics products and technology may not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations. You should not use Renesas Electronics products or technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. When exporting the Renesas Electronics products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations.
10. It is the responsibility of the buyer or distributor of Renesas Electronics products, who distributes, disposes of, or otherwise places the product with a third party, to notify such third party in advance of the contents and conditions set forth in this document, Renesas Electronics assumes no responsibility for any losses incurred by you or third parties as a result of unauthorized use of Renesas Electronics products.
11. This document may not be reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.

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

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

RENESAS

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

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

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

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

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

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

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

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

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

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

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

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

Colophon 5.0