RX Family

Communication Method in Low Power Mode Utilizing Existing Peripheral Functions

Introduction

This application note explains how to enable communication with external I/O during low power modes by utilizing SCI, SPI, and external pin interrupts (IRQn) available in the RX family.

This application note covers the following target devices. When adapting the contents of this application note to microcontrollers other than the RX66N group, please modify it according to the specifications of the target microcontroller and evaluate it thoroughly.

Target Device

RX Family Devices

Supported Devices

RX66N group

Contents

1. Overview

1.1 Background

It is common to remain in low power mode continuously and to wake up the CPU for processing when receiving data periodically from external I/O or other sources like as for battery-powred applications. After the processing is complete, there is a need to transition back to low power mode to reduce battery consumption and extend battery life. This application note explains how to enable communication with external I/O during low power mode by integrating SCI, SPI, and external pin interrupts (IRQn) available in the RX family.

1.2 Communication method during low power mode

This sample program utilizes the widely integrated SCI (Serial Communication Interface) and IRQn (external interrupt pin) available in the RX family.

1.3 System Configuration

Figure 1-1shows the system block diagram, and Table:1-1 lists the peripheral functions used and their applications.

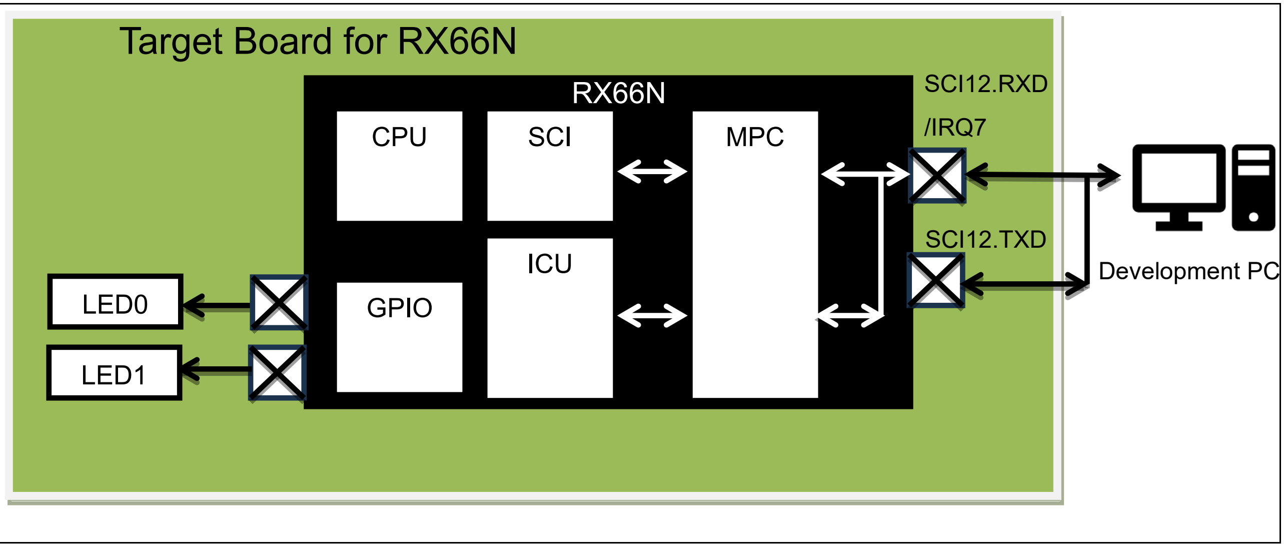

Figure 1-1 System block diagram

Table:1-1 Peripheral functions used and their applications

Peripheral Functions	Applications
ICU	IRQ interrupt reception,Interrupt to CPU
SCI12	UART communication interface
MPC	When transfer to the low power mode: Switch Port E2 to IRQ7 When back to nomal mode : Switch Port E2 to SCI12.RXD
GPIO	Port E2:SCI12.RXD/IRQ7 Port E1:SCI12.TXD Port 32: LED0 turn on/off Port 33: LED1 turn on/off

1.3.1 Status indication by LED

The status of the RX66N group can be confirmed with LED0.

Table 1-2:Status of RX66N group by LED0

LED0	State
OFF	In the low power mode
ON	In the normal operation

LED1 is turned on and off by commands from external I/O.

Table 1-3: Command Execution Results by LED1

LED1	State
OFF	Turn off by the LED off command
ON	Turn on by the LED on command

1.3.2 Modifications of the Target Board for RX66N

To connect the USB-to-serial conversion cable used for the development PC to the Target Board for RX66N, it is necessary to install two male-to-male jumper cables on the Target Board. Please install one jumper cable to pin 76 and the other to pin 77 of CN3.

Figure 1-2 shows the locations for the installation of the jumper cables.

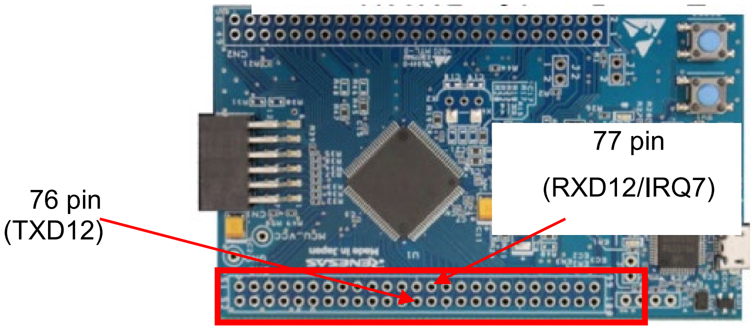

Figure 1-2 Jumper Cable Mounting Position

Please do not use jumper wires with a diameter of less than 0.5 mm, as they may not connect securely to the USB-to-serial conversion cable mentioned later.

1.3.3 Command list

Communication between the RX66N group and external I/O follows a command format to prevent issues such as unintended activation due to noise or erroneous data reception.

Table 1-4 lists the commands used, and Figure 1-3 shows the command format.

Table 1-4: Command List

No.	Command	Operation
1	CMON	Turn on LED1
2	CMOF	Turn on LED1
3	CMRE	Resend request to external IO
4	CMRV	Receive completion notification to external IO

C	M	R	E	NULL*
Command section [0:1]		Data section [2:5]		

Figure 1-3 Command Format

*During command communication, a NULL is always included at the end to indicate completion.

1.3.4 Using Terminal Software (TeraTerm)

This sample program uses TeraTerm for communication between the RX66N group and a PC, which serves as the external I/O.

The following explains how to use TeraTerm . Please note that TeraTerm must be installed on user’s PC in advance.

In addition,TeraTerm is just terminal soft and example.So,you can use other terminal soft when development.

a Start TeraTerm and select the port.
Select '**Serial (E)**, ' choose the appropriate "**Port (R)**" for your environment, and click '**OK**.' Figure 1-4 shows an example using COM8.

Figure 1-4 Starting TeraTerm and Port Configuration

b Configure the terminal settings in TeraTerm .
Select '**Setup**' and then "**Terminal**," and change the settings as shown in the red box in the figure below.

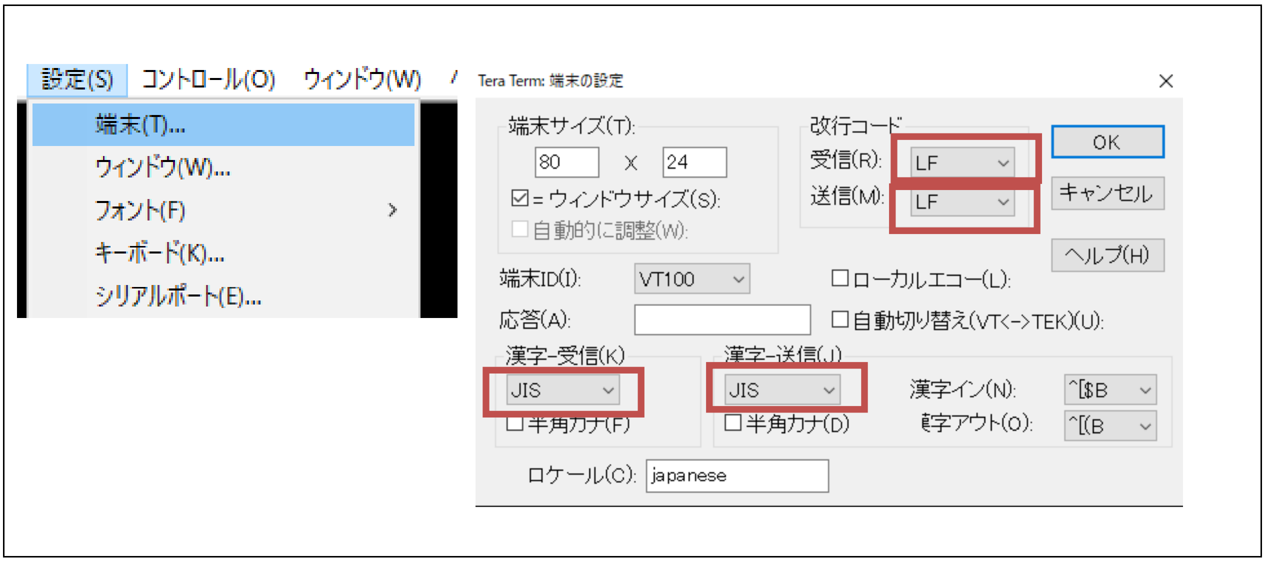

Figure 1-5 TeraTerm Terminal Settings

c Configure the serial settings in TeraTerm .
Select "**Setup**" and then **'Serial Port**, ' and change the settings as shown in the figure below.

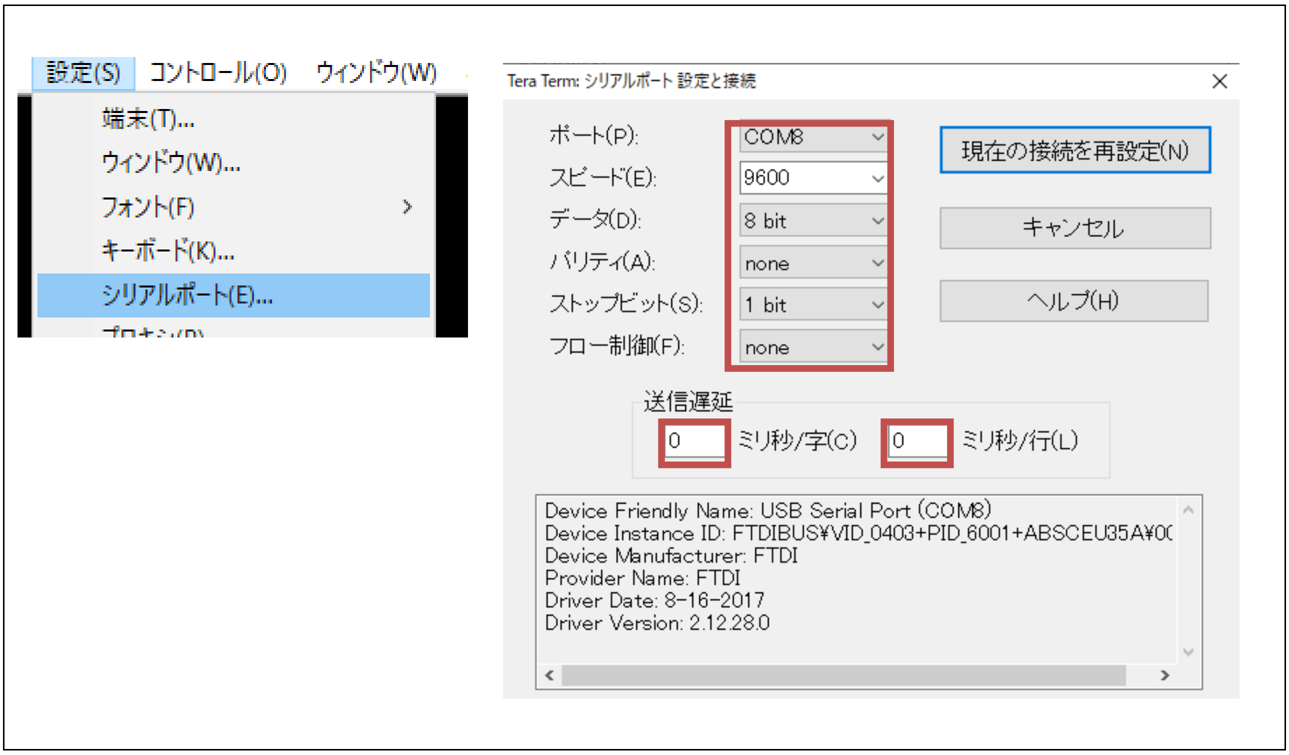

Figure 1-6 Serial Communication Settings

d Select **Control (O)** > **Macro (M)**.

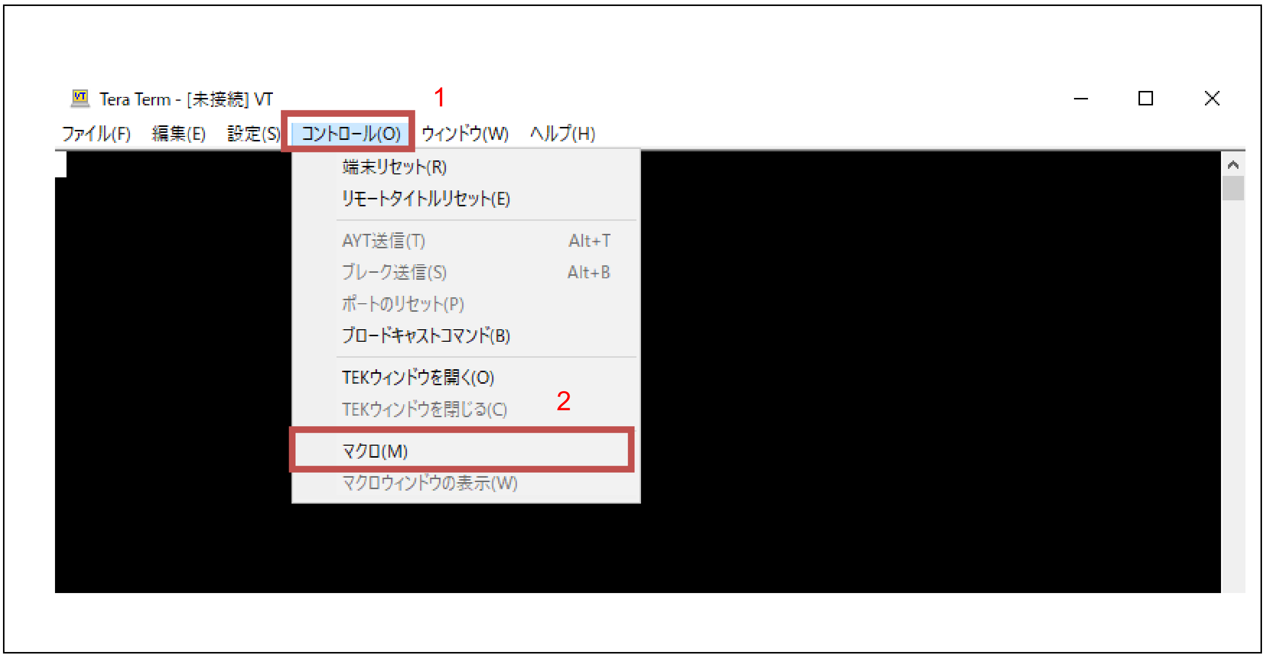

Figure 1-7 Starting Macro

e Open '**send_test.ttl**' from the opened Explorer window.

This '**send_test.ttl**' file is included with this sample code.

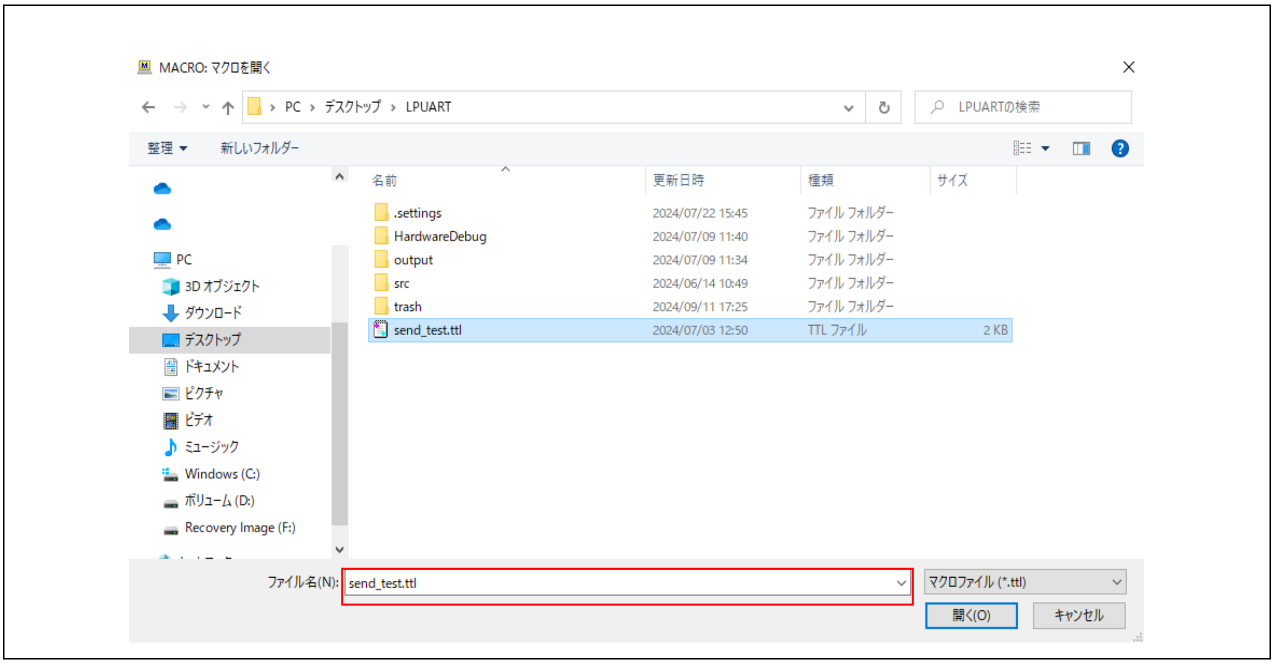

Figure 1-8 Select Macro

f A command input window will open, Then please enter one of the following commands.

・CMON: Send the command to turn on LED1.

・CMOF: Send the command to turn off LED1.

Click '**OK**' or press the ENTER key to send the entered command.

Figure 1-9 Send Command

g When the RX66N group receives the resent command, it will send the received command (CMRV), and the following popup will appear.

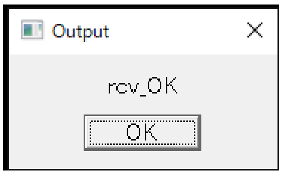

Figure 1-10 Reception Confirmation from the RX66N Group

Note: If a retransmission request (CMRV) is received, TeraTerm remembers the original command, so it will automatically resend the initial command. Therefore, there is no need to re-enter the command.

2. Operational check conditions

The sample programs in this application note have been tested under the conditions shown below.

Table 2-1 Operating environment

Item	Contents
MCU used	R5F566NNHDFP (Target Board for RX66N)
Operating Frequency	Main clock: stopped PLL：240MHz(HOCO x 1/1 x 15) HOCO：16MHz LOCO：Stopped System clock(ICLK) ：120MHz(PLL x 1/2) Peripheral module clock A(PCLKA) : 120MHz(PLL x 1/2) Peripheral module clock B(PCLKB) : 60MHz(PLL x 1/4) Peripheral module clock C(PCLKC) : 60MHz(PLL x 1/4) Peripheral module clock D(PCLKD) : 60MHz(PLL x 1/4) FlashIF clock(FCLK) ：60MHz(PLL x 1/4)
Operating voltage	3.3V
Integrated development environment	Renesas Electronics e² studio Version 2024-07.0
C Compiler Note	Renesas Electronics C/C++ Compiler Package for RX Family V3.06.00
RX Smart configurator	V2.22.0
Board support package(r_bsp)	V7.50
Endian	Little endian
Operation mode	Single chip mode
Processor mode	Supervisor mode
Sample code version	V1.00
Board used	Target Board for RX66N (Product name：RTK5RX66N0C00000BJ)
Emulator	E2-Lite

Note: If the same version of the toolchain (C compiler) specified in this project is not installed, the toolchain will not be selected, and an error will occur.
If an error occurs, please check the selection status of the toolchain in the project settings screen.

Please refer to FAQ 3000404 for instructions on how to configure the settings.

FAQ 3000404: Building an imported project results in the error 'Program 'make' not found in PATH'(e² studio)

3. Operating environment

To write the sample program for this application, user needs to connect the Target Board for the RX66N to the development PC via USB.

In this section, write using the on-chip debugger on the Target Board for RX66N.
Table 3-1 shows the writing environment for the sample program and Figure 3-2 shows the connection method.

Table 3-1 Sample program writing environment

Item	Contents
Development environment	・Windows PC for development ・e^{2}studio Version 2024-07.0
Debugging tools	・E2 Lite(RX)
Boards	・Target Board for RX66N (Product name：RTK5RX66N0C00000BJ)
Cables	・USB serial conversion cable (type-A ⇔ serial connector)* ・Jumper cable (male ⇔ male)00

* This cable will connect to the implemented jumper cable (male-to-male), so please prepare one with a female connector on the serial side.

Figure 3-1 USB Serial Converter Cable (for reference)

3.1 Connection

a. Connect the jumper cable mounted on pins 76 (RXD12/IRQ7) and 77 (TXD12) of CN3 to the serial connector of the USB serial converter cable.

b. Connect the Type-A side of the USB cable to the USB port on the development PC. This will connect to the on-chip debugging emulator.

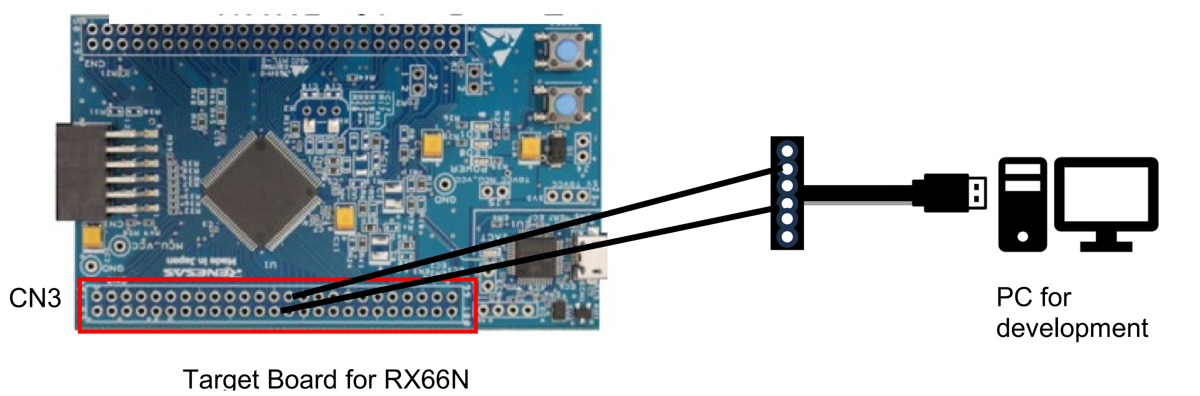

Figure 3-2 Target Board for RX66N Connection diagram

3.2 Writing the Sample Program

The connections are complete, follow the steps below to write the project.

a. Import the project.

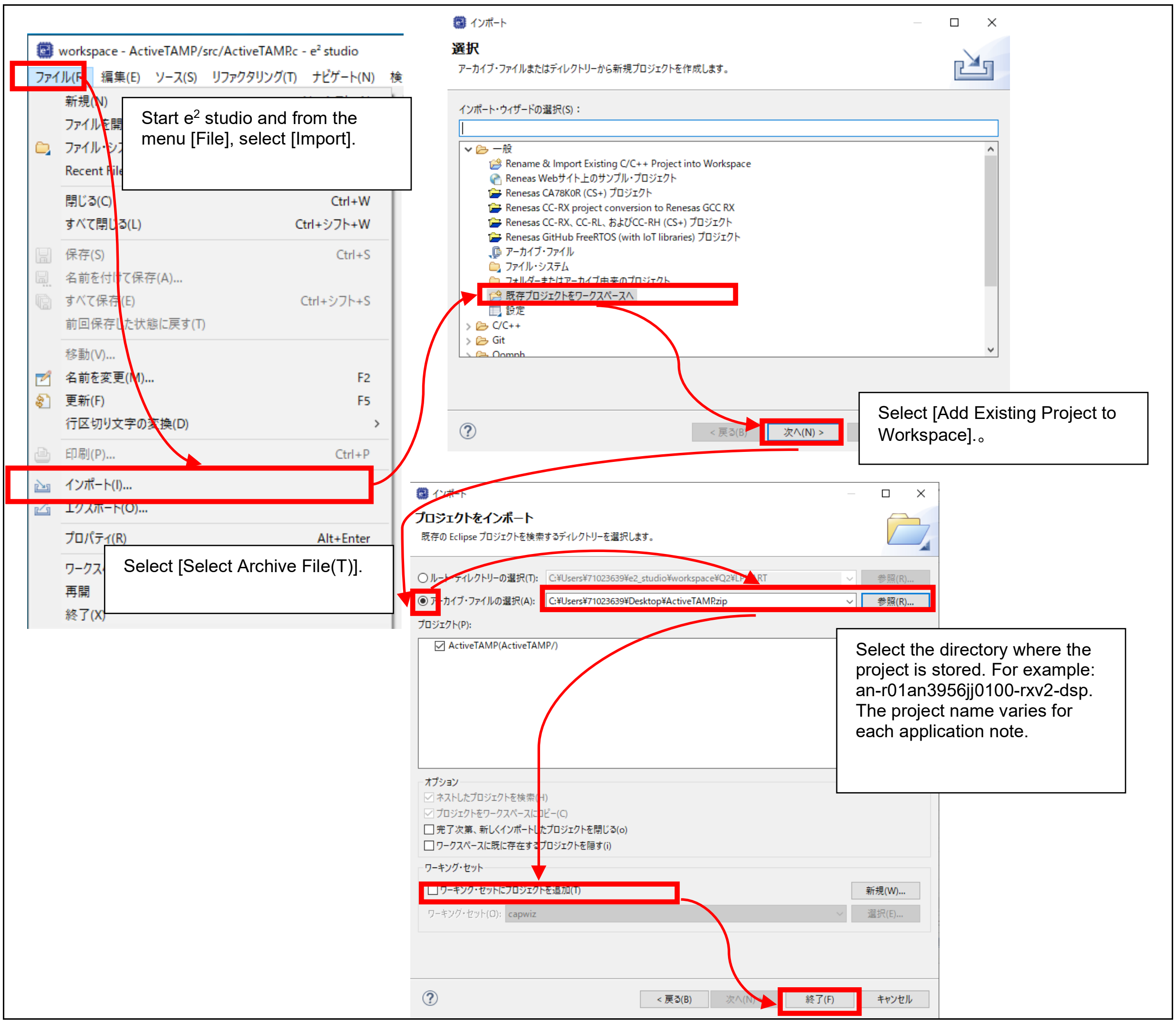

Figure 3-3 Import the Project.

b. After building the project, press the debug button in e^2 studio as per the rules. This will initiate the writing process.

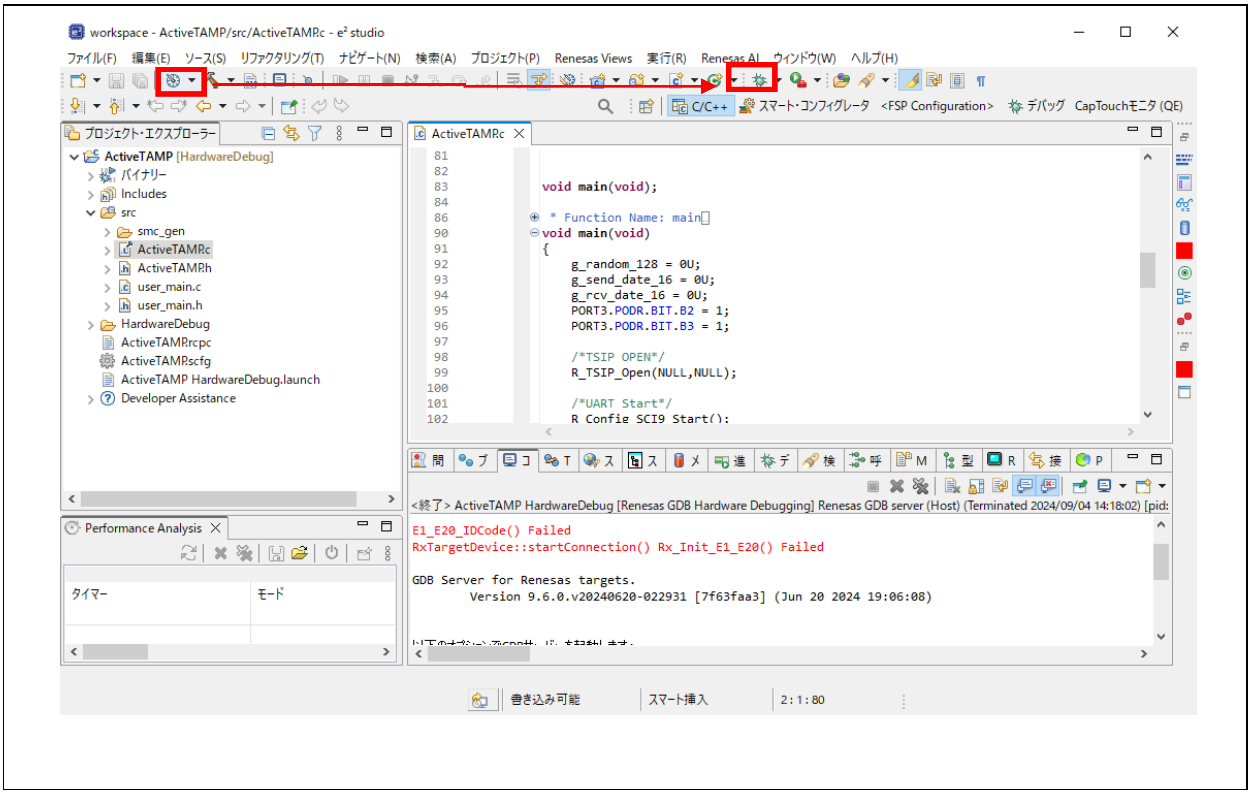

Figure 3-4 Build and Debug

c. After writing, press **'Exit'** to automatically boot into FINE mode, which will disconnect e^2 studio and the on-chip debugging emulator.

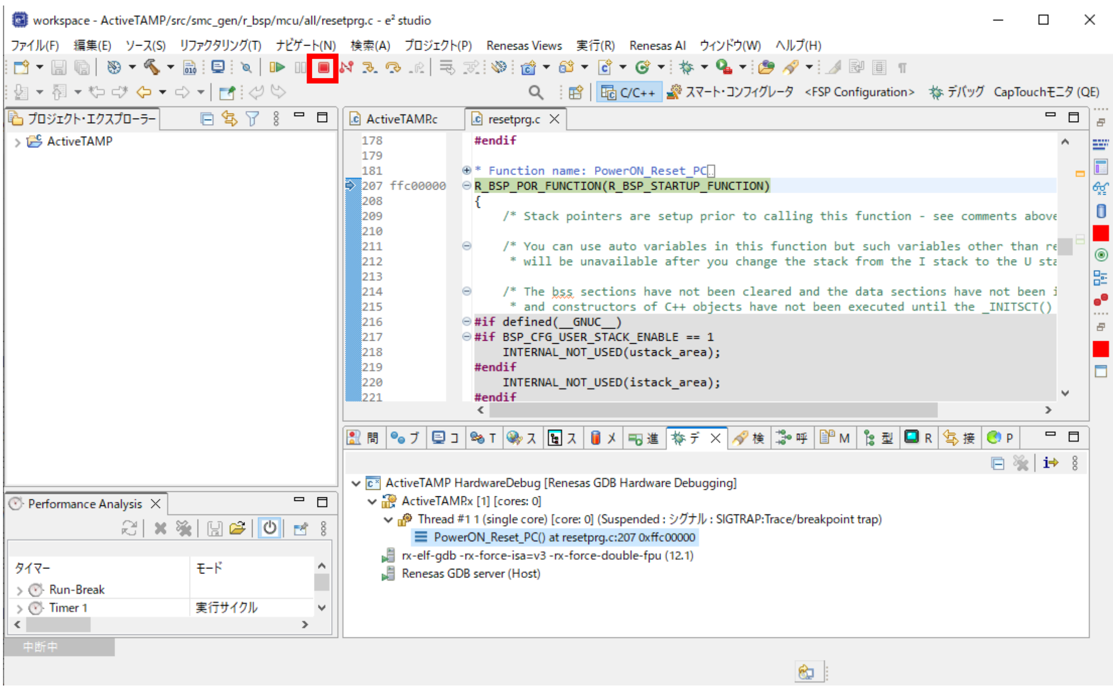

Figure 3-5 Emulator Disconnection

d. By connecting the emulator reset header (J6), the system will start in single-chip mode, which does not require a debugger, and execute the written project.

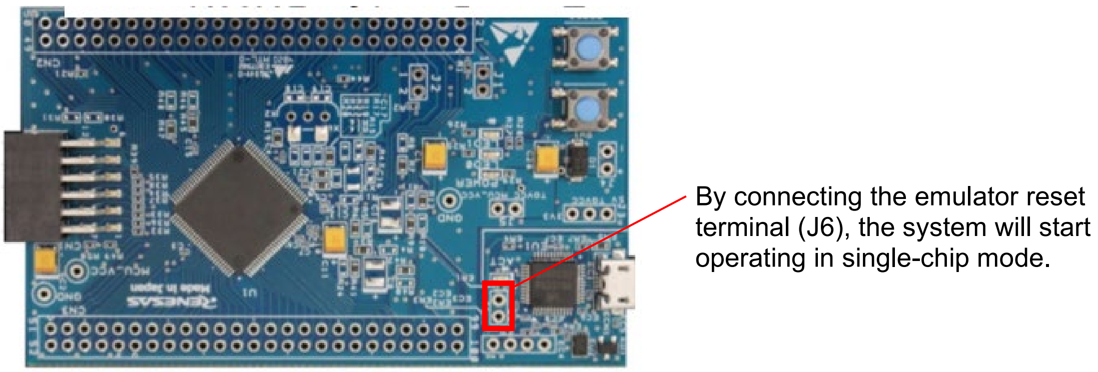

Figure 3-6 Switching Operation Modes

4. System Description

4.1 Explanation of Communication Methods in Low Power Mode

In this sample program, UART communication will be performed with the RX66N group in low power mode (software standby) using a PC with terminal software installed on the external I/O. In this case, the CMON command will be issued.

Upon power-up, the RX66N group sets PE2 to IRQ7 to transition into software standby mode.

The RX66N group resumes from software standby mode via UART communication from the external IO (IRQ7 interrupt for the RX66N). After resuming, the RX66N group allows SCI transmission and reception, changes PE2 from IRQ7 to SCI12.RXD, and issues a retransmission request command. After receiving the retransmission request command, the external IO reissues the command to turn on LED1 on the RX66N Target Board, lighting up LED1 and returning to software standby mode.

After that, the RX66N group turns off LED1 and transitions back to software standby mode.

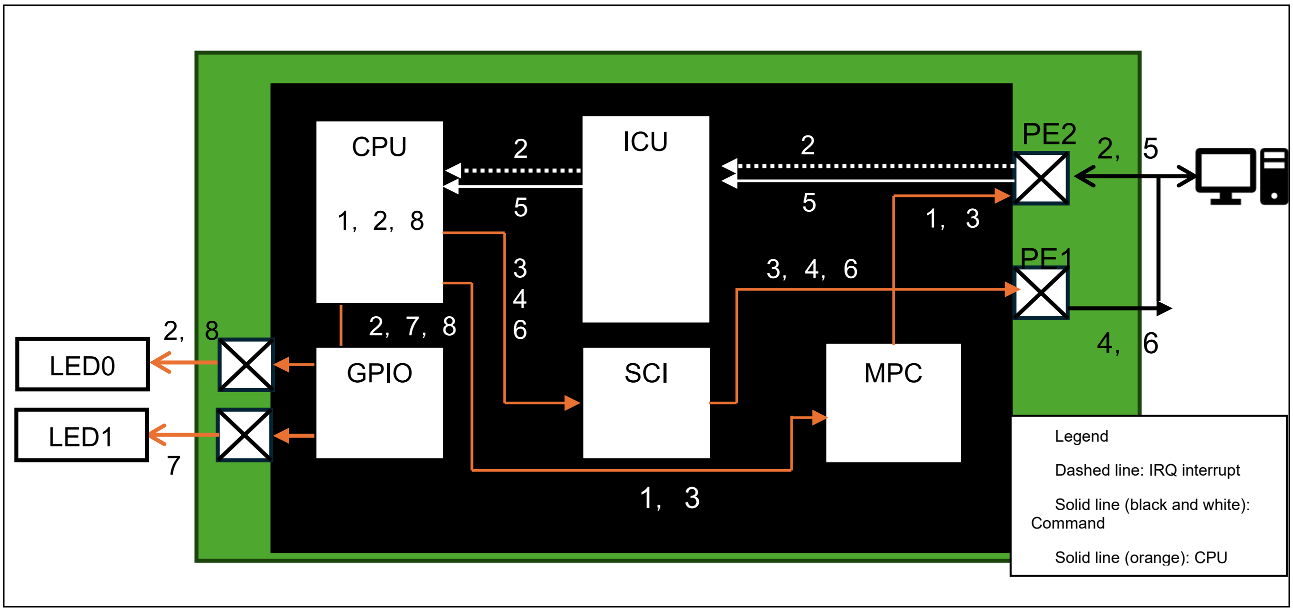

Figure 4-1 Operation of Communication Method in Low Power Mode

1	Power on	The RX66N group sets PE2 to IRQ7 and transitions to software standby mode.
2	Turn on LED Command	Receive the LED turn-on command (CMON) from the PC. As this is an IRQ7 interrupt for the RX66N group, the CPU will return to normal operation. Upon returning, LED1, which indicates the normal state, will be turned on.
3	Enable SCI Transmission and Reception and Change Function of PE2	To prepare for a retransmission from the PC, the CPU will set the SCI send/receive permission. The configuration of PE2 will be changed from IRQ7 to SCI12.RXD.
4	Retransmission Request	The RX66N group sends a retransmission request command (CMRE) to the PC.
5*	Resend command to turn on LED	The TeraTerm on the PC automatically retransmits the command to turn on the LED (CMON).
6	Reception Complete	The CPU sends the receive complete command (CMRV) to the external IO.
7	Turn on LED	The CPU turn on LED1.
8	Transition to Software Standby Mode	Turn off LED1. Change PE2 from SCI12.RXD to IRQ7. Transition back to Software Standby Mode.

*: If no retransmission occurs after the resend command in step 3 (referred to as CMON in this case), the RX66N group will continue to wait for a retransmission from the external I/O. This sample program autonomously handles retransmission; however, when applying this in a system, it is important to ensure that retransmission is performed.

5. Software Description

5.1 Configuration

5.1.1 File Composition

Table 5-1 lists the files for this sample program. Note that files that use the source code generated directly by the Smart Configurator's code generation function are excluded.

Table 5-1 Files used in the sample code

File name	Overview	Remarks
main.c	Main processing	-
main.h	Header file of main.c	-
Config_SCI12_user.c	Pin assignment process for SCI12	Add pin assignment processing to the files generated by the Smart Configurator.
Config_ICU_user.c	Pin assignment process for IRQ7	Add pin assignment processing to the files generated by the Smart Configurator.

5.1.2 List of constants

Table 5-2 shows the constants used in the sample program.

Table 5-2 List of constants

Constant name	Setting value	Contents
WAKEUP_DELAY	15000L	Oscillation stabilization wait time after transitioning to normal operating mode
WAKEUP_DELAY_UNITS	1000000L	Wait Time Calculation Values in Units of 1 μs
LED_ON	0	LED output data: Turn on
LED_OF	1	LED output data: Turn off
LED0	PORT3.PODR.BIT.B2	LED0 output data storage bit
LED1	PORT3.PODR.BIT.B3	LED1 output data storage bit

5.1.3 List of Variables

Table 5-3 shows the global variables used in the sample program.

Table 5-3 List of Variables

Type	Variable name	Contents	Functions Used
uint8_t	rcv_ok[5U]	Receive complete command	main
uint8_t	resend[5U	Resend request command	main
const char	ledcmd1[5U]	Data for identifying received command (LED turn on command)	main
const char	ledcmd2[5U]	Data for identifying received command (LED turn off command)	main
char	cmd1[2]	Command Identification Reference	main
char	cast_rcv[5U]	Used for type conversion from uint8_t type to char type	main
uint8_t	rcvend	Receive completion event occurrence flag	main

5.1.4 List of Functions

Table 5-4 lists the functions. Note that functions that use the source code generated directly by the Smart Configurator's code generation feature are omitted.

Table 5-4 List of Functions

Function name	Overview
main	Main processing
lpmode	Communication standby processing in software standby mode
r_Config_SCI12_Create_UserInit	Pin setting process for SCI12
r_Config_SCI12_callback_receiveerror	Callback processing for communication errors
r_Config_SCI12_callback_receiveend	Callback processing for reception completion
r_Config_ICU_irq7_interrupt	Interrupt disable processing for IRQ7

5.1.5 Function Specifications

This section presents the function specifications of the sample code.

main

Overview	Main processing
Header	None
Declaration	Void main(void)
Description	After initialization, the system transitions to software standby mode. LED1 can also be turned on and off based on commands from the host.
Arguments	None
Return Value	None
Remarks	-

lpmode

Overview	Communication waiting process in software standby mode
Header	main.h
Declaration	void lpmode(void)
Description	Assign PE2 to IRQ7 and transition to software standby mode. At this point, the CPU will be activated by communication from the PC. After the CPU resumes, PE2 will be changed to RXD12 for SCI12, and the system will handle the sending of retransmission request commands as well as waiting for the incoming retransmission commands.
Arguments	None
Return Value	None
Remarks	-

r_Config_SCI12_Create_UserInit

Overview	Pin setting process for SCI12
Header	Config_SCI12.h
Declaration	void R_Config_SCI12_Create_UserInit(void)
Description	Change the pin assignment of the PE2 pin for SCI12 and make it ready to start SCI12 communication.
Arguments	None
Return Value	None
Remarks	-

r_Config_SCI12_callback_receiveerror	
Overview	Callback processing for communication errors
Header	Config_SCI12.h
Declaration	static void r_Config_SCI12_callback_receiveerror(void);
Description	In the communication error, the system will handle the sending of retransmission request commands.
Arguments	None
Return Value	None
Remarks	-

r_Config_SCI12_callback_receiveend	
Overview	Callback processing upon completion of reception
Header	Config_SCI12.h
Declaration	static void r_Config_SCI12_callback_receiveend(void);
Description	Upon reception completion, the system will set the reception complete flag and handle the sending of the reception complete command.
Arguments	None
Return Value	None
Remarks	-

r_Config_ICU_irq7_interrupt	
Overview	Interrupt disabling process for IRQ7
Header	None
Declaration	static void r_Config_ICU_irq7_interrupt(void)
Description	This is the interrupt handling process for activating the CPU from software standby mode. Within this process, the handling to disable this interrupt (IRQ7) will be performed.
Arguments	None
Return Value	None
Remarks	-

5.2 Flowchart

In this section, the operation of each function in this sample program will be explained using flowcharts. Note that the "initialization of each module" shown in Figure 5-1 is generated by the Smart Configurator in the R_Config_<ModuleName>_Create function (with the exception of TSIP, which is generated in the R_TSIP_Open function). The R_Config_<module name>Create_UserInit function is called by the R_Config<module name>_Create function.

5.2.1 Main Function

The flow of the main function is shown in Figure 5-1.

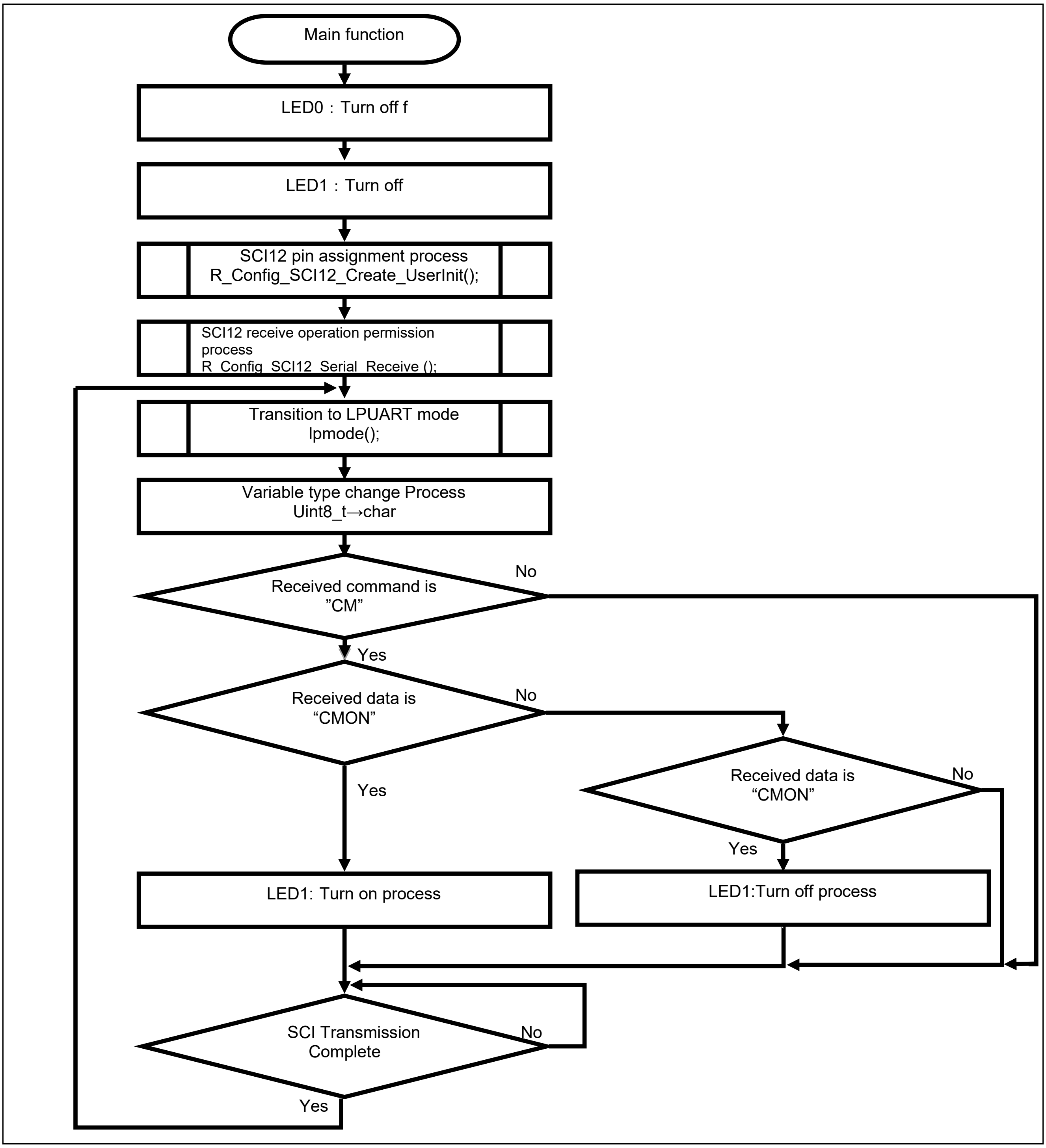

Figure 5-1 Main Function Flow

5.2.2 Lpmode function

The flow of the Lpmode function is shown in Figure 5-2.

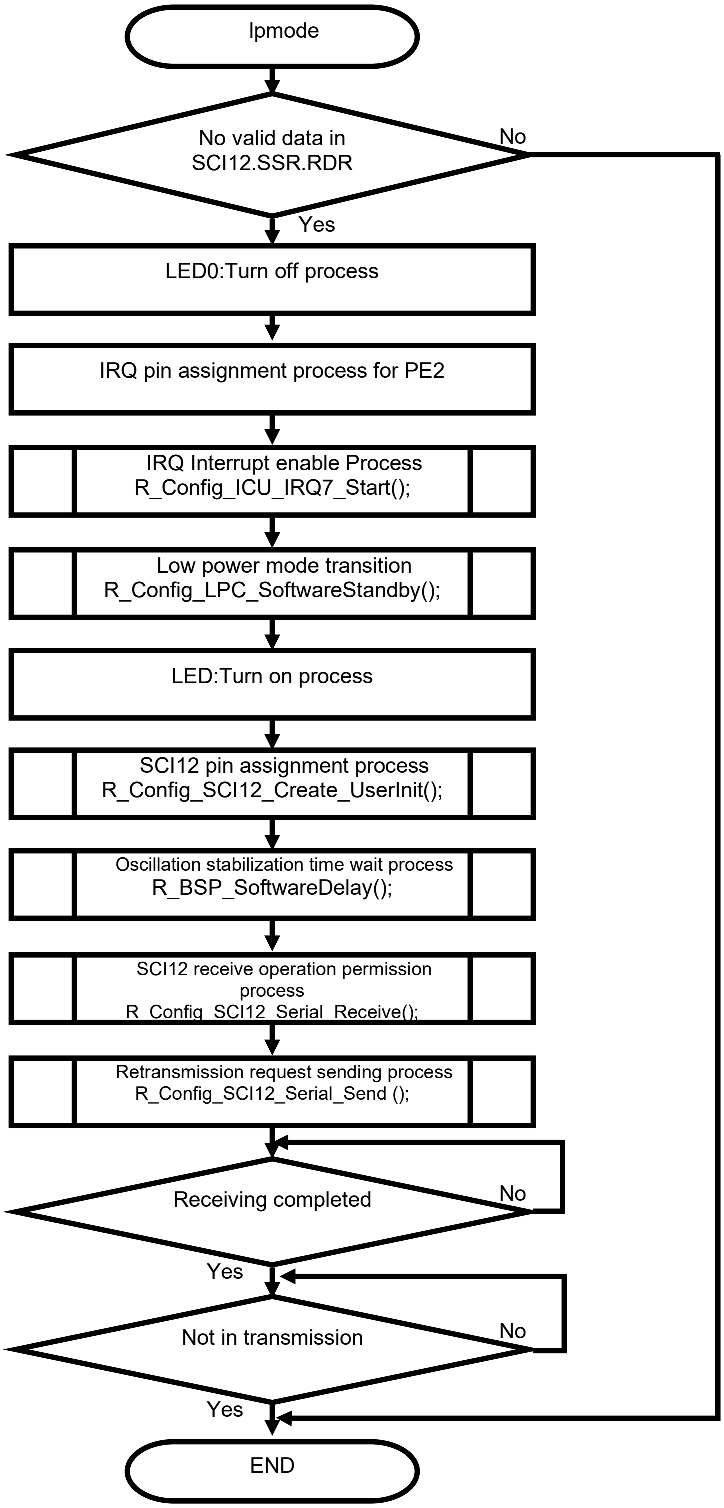

Figure 5-2 lpmode Function Flow

5.2.3 R_Config_SCI12_Create_UserInit function

The flow of R_Config_SCI12_Create_UserInit is shown in Figure 5-3.

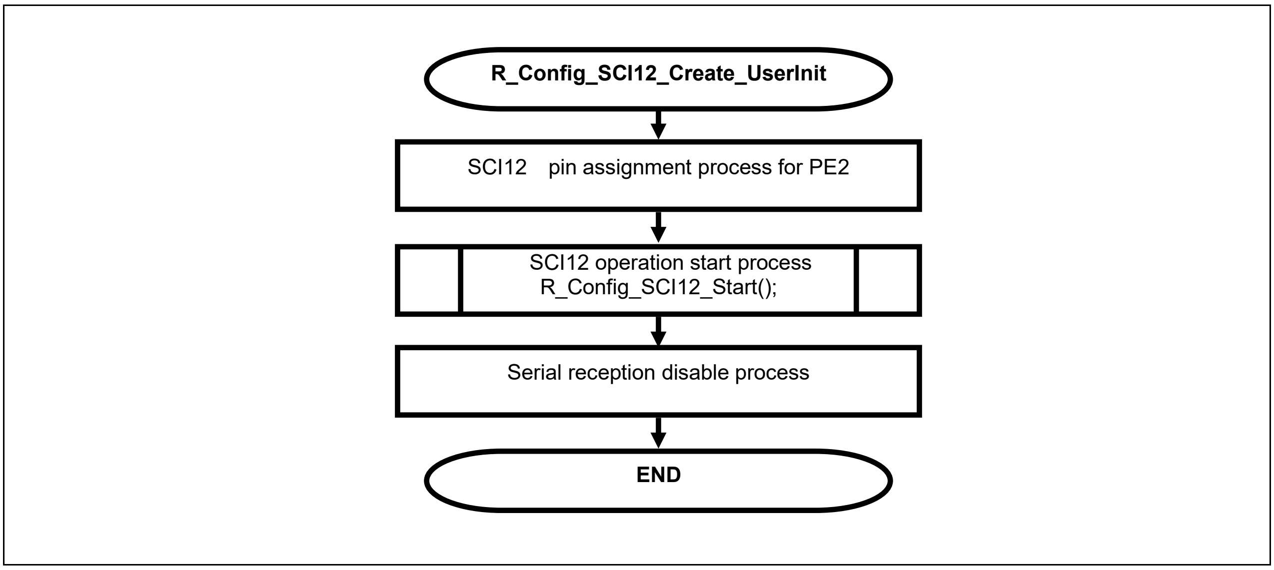

Figure 5-3 Flow of R_Config_SCI12_Create_UserInit

5.2.4 r_Config_ICU_irq7_interrupt function

The flow of r_Config_ICU_irq7_interrupt is shown in Figure 5-4.

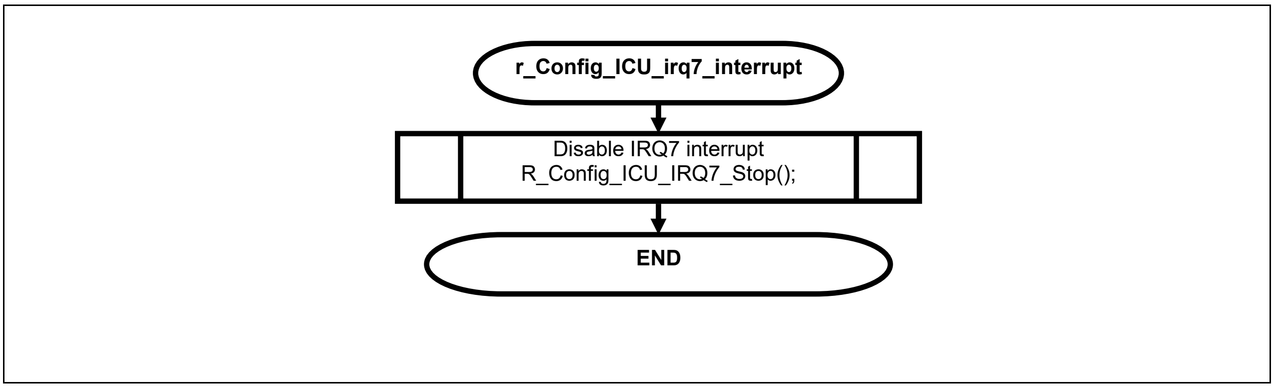

Figure 5-4 Flow of r_Config_ICU_irq7_interrupt

5.2.5 r_Config_SCI12_callback_receiveerror function

The flow of r_Config_SCI12_callback_receiveerro is shown in Figure 5-5.

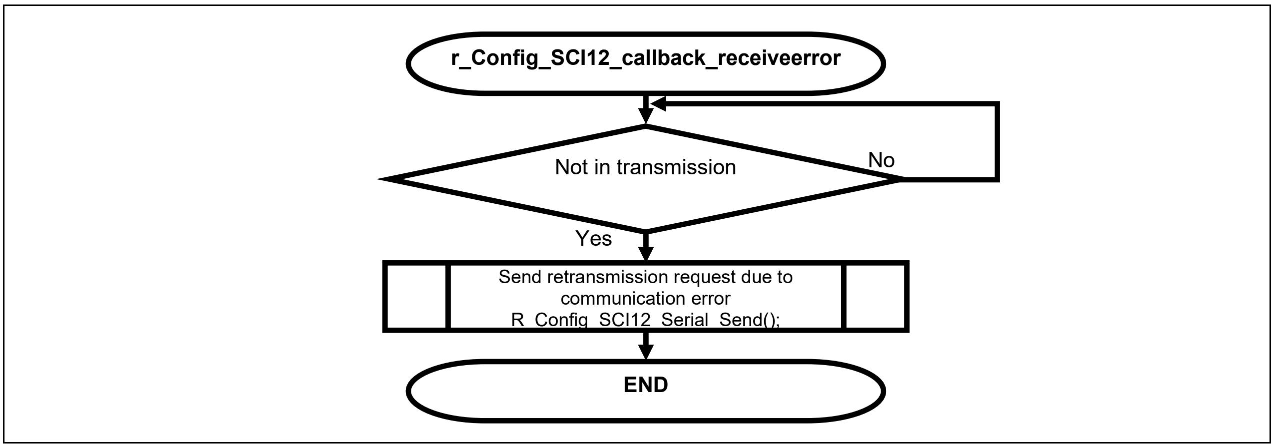

Figure 5-5 Flow of r_Config_SCI12_callback_receiveerror

5.2.6 r_Config_SCI12_callback_receiveend function

The flow of r_Config_SCI12_callback_receiveend is shown in Figure 5-6.

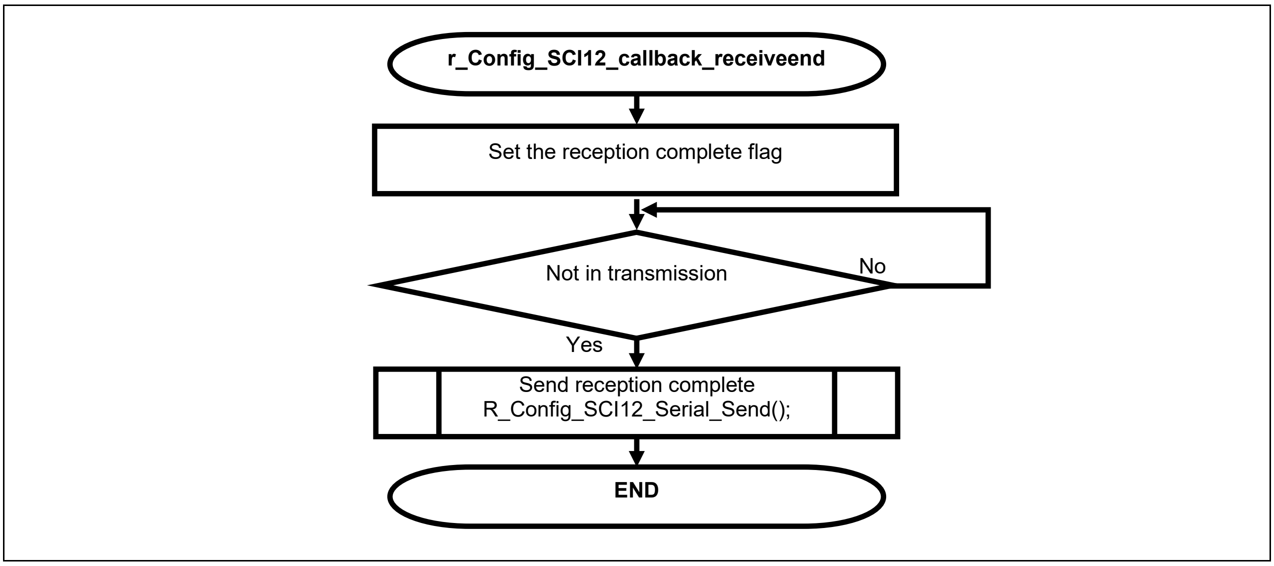

Figure 5-6 Flow of r_Config_SCI12_callback_receiveend

6. Reference documents

RX66NGroup User's Manual Hardware (renesas.com)

Target Board for RX66N User's Manual Rev.1.00 (renesas.com)

Revision History

Rev.	Date	Description	
		Page	Summary
1.00	January 10,2025	-	First Edition issued

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

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

1. Precaution against Electrostatic Discharge (ESD)
   A strong electrical field, when exposed to a CMOS device, can cause destruction of the gate oxide and ultimately degrade the device operation. Steps must be taken to stop the generation of static electricity as much as possible, and quickly dissipate it when it occurs. Environmental control must be adequate. When it is dry, a humidifier should be used. This is recommended to avoid using insulators that can easily build up static electricity. Semiconductor devices must be stored and transported in an anti-static container, static shielding bag or conductive material. All test and measurement tools including work benches and floors must be grounded. The operator must also be grounded using a wrist strap. Semiconductor devices must not be touched with bare hands. Similar precautions must be taken for printed circuit boards with mounted semiconductor devices.
2. Processing at power-on
   The state of the product is undefined at the time when power is supplied. The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the time when power is supplied. In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the time when power is supplied until the reset process is completed. In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the time when power is supplied until the power reaches the level at which resetting is specified.
3. Input of signal during power-off state
   Do not input signals or an I/O pull-up power supply while the device is powered off. The current injection that results from input of such a signal or I/O pull-up power supply may cause malfunction and the abnormal current that passes in the device at this time may cause degradation of internal elements. Follow the guideline for input signal during power-off state as described in your product documentation.
4. Handling of unused pins
   Handle unused pins in accordance with the directions given under handling of unused pins in the manual. The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of the LSI, an associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible.
5. Clock signals
   After applying a reset, only release the reset line after the operating clock signal becomes stable. When switching the clock signal during program execution, wait until the target clock signal is stabilized. When the clock signal is generated with an external resonator or from an external oscillator during a reset, ensure that the reset line is only released after full stabilization of the clock signal. Additionally, when switching to a clock signal produced with an external resonator or by an external oscillator while program execution is in progress, wait until the target clock signal is stable.
6. Voltage application waveform at input pin
   Waveform distortion due to input noise or a reflected wave may cause malfunction. If the input of the CMOS device stays in the area between V_{IL} (Max.) and V_{IH} (Min.) due to noise, for example, the device may malfunction. Take care to prevent chattering noise from entering the device when the input level is fixed, and also in the transition period when the input level passes through the area between V_{IL} (Max.) and V_{IH} (Min.).
7. Prohibition of access to reserved addresses
   Access to reserved addresses is prohibited. The reserved addresses are provided for possible future expansion of functions. Do not access these addresses as the correct operation of the LSI is not guaranteed.
8. Differences between products

Before changing from one product to another, for example to a product with a different part number, confirm that the change will not lead to problems. The characteristics of a microprocessing unit or microcontroller unit products in the same group but having a different part number might differ in terms of internal memory capacity, layout pattern, and other factors, which can affect the ranges of electrical characteristics, such as characteristic values, operating margins, immunity to noise, and amount of radiated noise. When changing to a product with a different part number, implement a system-evaluation test for the given product。

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/.