RA Family

Examples of Transitioning to Low Power Modes

Introduction

This application note describes the use of the low power modes on the target devices listed below. It presents examples of transitioning to various low power modes. The software described in this application note also provides a simple method of measuring current values after transitioning to one of the low power modes.

Target Devices

- RA Family

Confirmed Devices

- RA4E2 Group

When using this application note with other Renesas MCUs, careful evaluation is recommended after making modifications to comply with the alternate MCU.

Contents

1. Overview

This application note describes the use of functionality for reducing power consumption based on code generated by the Code Generator (CG) function of Smart Configurator (SC) to transition to and to resume from the low power modes.

Table 1.1 shows the supported modes for each device.

Table 1.1 Modes Supported by Devices

Mode	+: Supported : Not supported
Sleep	+
Software standby	+
Deep software standby	+
Snooze	+

The conditions applying to the low power mode sample program described in this application note are listed in Table 1.2

Table 1.2 Sample Program Conditions

Low Power Mode		Module Stop Conditions
Sleep		State in which modules other than DMAC/DTC and RAM0 are stopped
Software standby	RTC not used	State in which modules other than DMAC/DTC and RAM0 are stopped
	RTC used	State in which modules other than DMAC/DTC and RAM0 are stopped
Deep software standby		State in which modules other than DMAC/DTC and RAM0 are stopped*1

Note1 : The DMAC and DTC are put into the module stop state by the functionality for reducing power consumption.

2. Operation Confirmation Conditions

The operation of the sample program described in this application note has been confirmed under the conditions listed in Table 2.1.

Table 2.1 Operation Confirmation Conditions

Item	Contents
MCU used	R7FA4E2B93CFM (RA4E2 Group)
Operating frequency	The operating frequency differs for each project. Refer to 7.1, Operating Frequencies of Each Project, for a listing of the specific operating frequencies.
Operating voltage	3.3 V
Integrated development environment	Renesas Electronics e² studio Version 2025-07
C compiler	Eclipse Embedded CDT project. Embedded C/C++ Arm Cross Compiler Version: 6.4.0.202502060831
	Compiled options –lang = C99
Emulator*1	E2 Lite
Endian order	Both little endian and big endian
Operating mode	Single-chip mode
Processor mode	Supervisor mode
Sample program version	Version 1.00
Board used (EK)	Renesas Evaluation kit for RA4E2 (product No.: RTK7EKA4E2SxxxxxBE)

Note 1. Make sure the emulator is disconnected when measuring the current.

3. Hardware

3.1 Pins Used

Table 3.1 lists the pins used and their functions.

Table 3.1 Pins Used and Their Functions

Pin Name	I/O	Description
P005/IRQ10-DS	Input	SW1 input (transition to and resume from low power mode)
PD104	Output	LED2 output • After initial settings, on when in normal operating state • Off at transition to low power mode • On at after resume from low power mode

3.2 Hardware Configuration

3.2.1 Measurement Points for MCU Current Consumption

For the RA4E2 EK board, test points TP1 and TP3 are connected via a 5 mΩ resistor for measuring MCU current consumption. To measure current, monitor the voltage drop across the resistor and calculate the current using Ohm’s law.

4. Software

The sample program described in this application note is divided into separate projects for each low power mode. For details, refer to 4.3, Project Composition. The sample program described in this application note transitions to low power mode when a low power mode transition trigger occurs, and when another low power mode transition trigger occurs while in low power mode, it resumes from low power mode.

IRQ is used as the low power mode transition trigger. Table 4.1 lists the IRQ settings.

Table 4.1 IRQ Settings

Setting Item	Setting Description
Detection type	Falling edge
Digital filter	Disabled
Priority	Level 15 (highest)

4.1 Operation Overview

4.1.1 Sleep Mode

After initial settings, the program turns on the LED and waits for a low power mode transition trigger to occur. When a low power mode transition trigger occurs, the program turns off the LED and transitions to sleep mode. When another low power mode transition trigger occurs while in sleep mode, the program resumes from sleep mode and turns on the LED.
Figure 4.1 shows an overview of sleep mode operation.

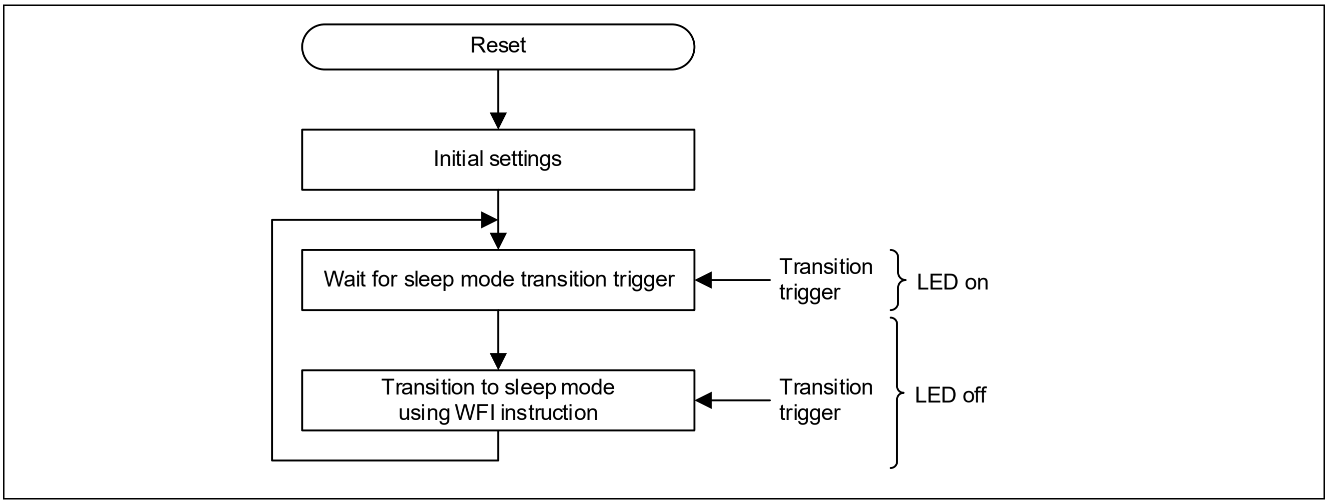

Figure 4.1 Overview of Sleep Mode Operation

4.1.2 Software Standby Mode

After initial settings, the program turns on the LED and waits for a low power mode transition trigger to occur. When a low power mode transition trigger occurs, the program turns off the LED and transitions to software standby mode. When another low power mode transition trigger occurs while in software standby mode, the program resumes from software standby mode and turns on the LED.

Figure 4.2 shows an overview of software standby mode operation.

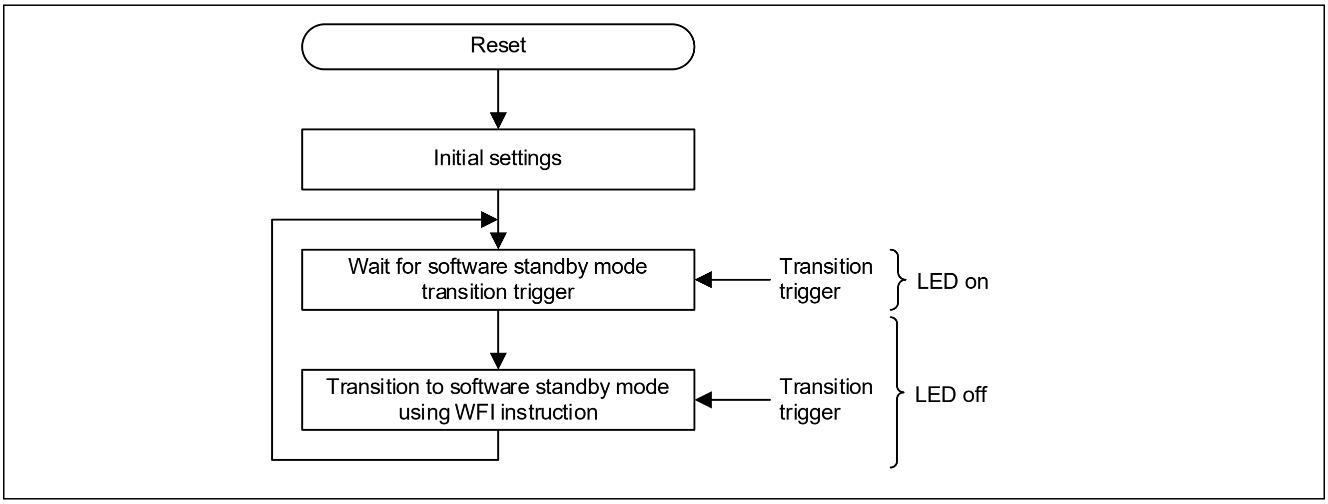

Figure 4.2 Overview of Software Standby Mode Operation

4.1.3 Deep Software Standby Mode

After initial settings, the program turns on the LED and waits for a low power mode transition trigger to occur. When a low power mode transition trigger occurs, the program turns off the LED and transitions to deep software standby mode. When a reset is applied using the RES# pin occurs while in deep software standby mode, operation resumes from deep software standby mode.
Figure 4.3 shows an overview of deep software standby mode operation.

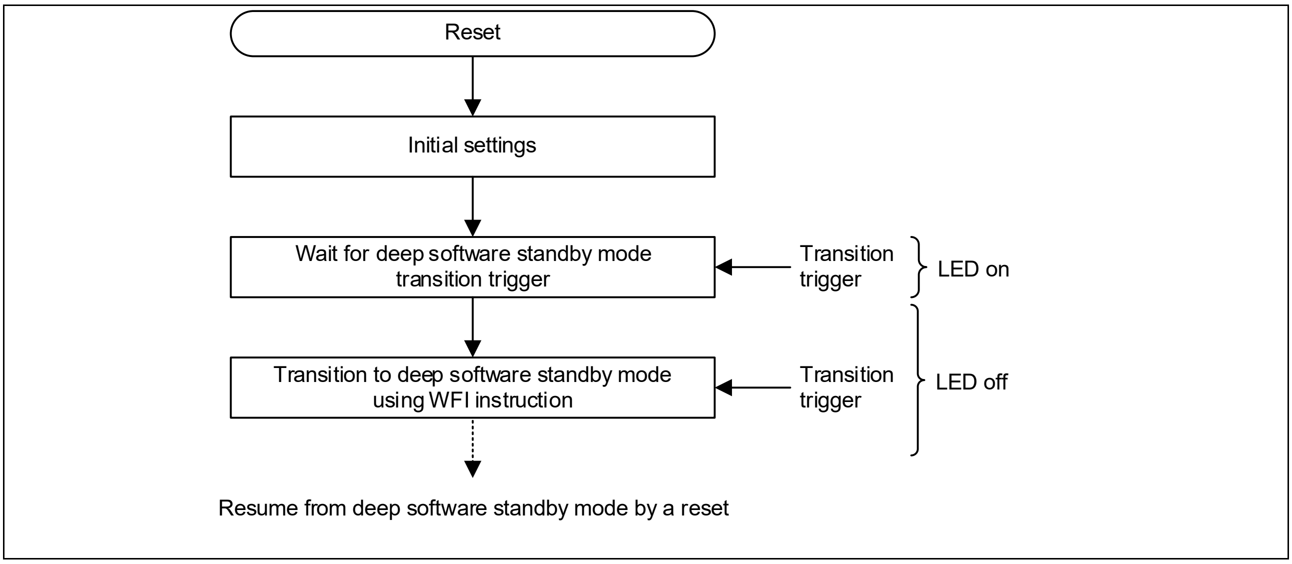

Figure 4.3 Overview of Deep Software Standby Mode Operation (Mode release by reset)

4.2 Module Composition

The sample program described in this application note utilizes source code related to the components listed in Table 4.2, generated by the Code Generator function of Smart Configurator, in combination.

Figure 4.4 shows the module composition of the sample program described in this application note.

Table 4.2 List of Components

Component	Version	Resource
Board support package (BSP)	6.0.0	—
Realtime clock	6.0.0	RTC
Low power consumption	6.0.0	LPM
Interrupt controller	6.0.0	ICU

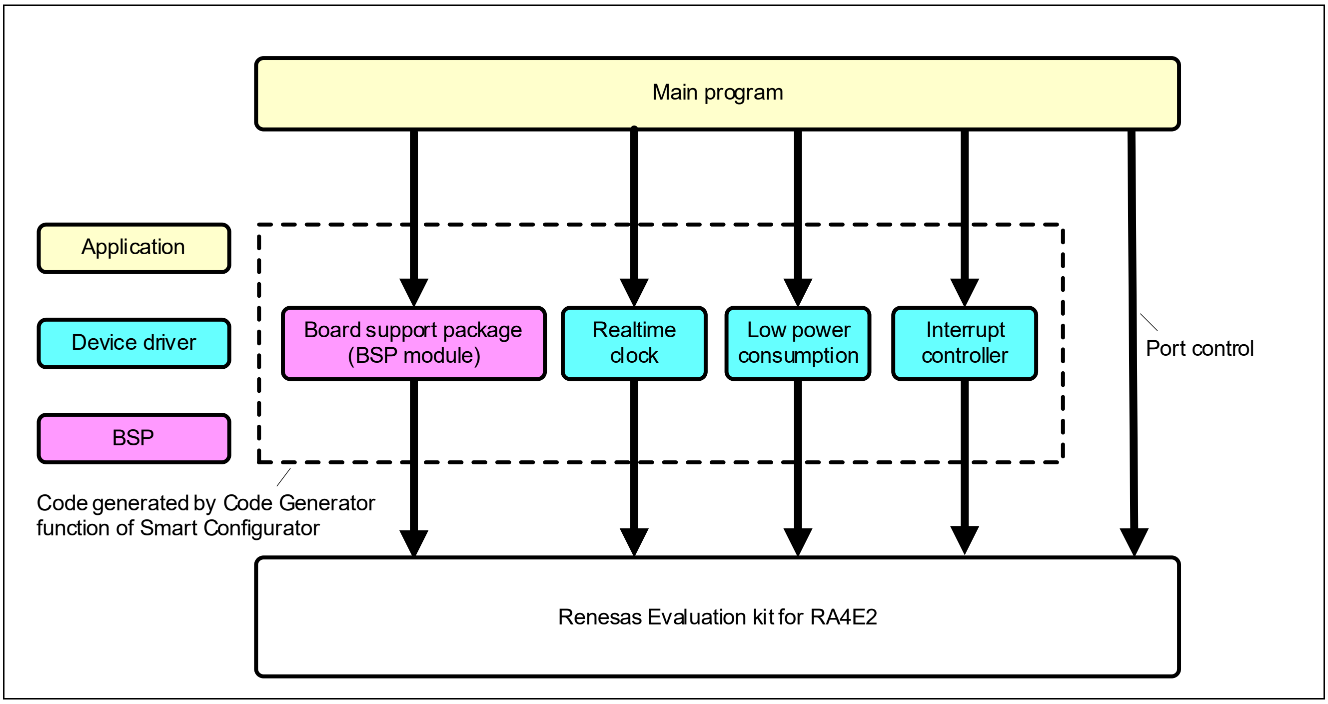

Figure 4.4 Module Composition

4.3 Project Composition

Table 4.3 is list the project composition described in this application note, and Figure 4.5 shows the project composition.

Table 4.3 List of Project Composition

Target Device	Project Name	Contents
RA4E2	power_save_sleep_ra4e2	Project for confirming sleep mode on RA4E2
	power_save_deep_software_standby_ra4e2	Project for confirming deep software standby mode on RA4E2
	power_save_software_standby_ra4e2	Project for confirming software standby mode on RA4E2
	power_save_software_standby_rtc_operation_ ra4e2	Project for confirming software standby mode on RA4E2 with RTC operating

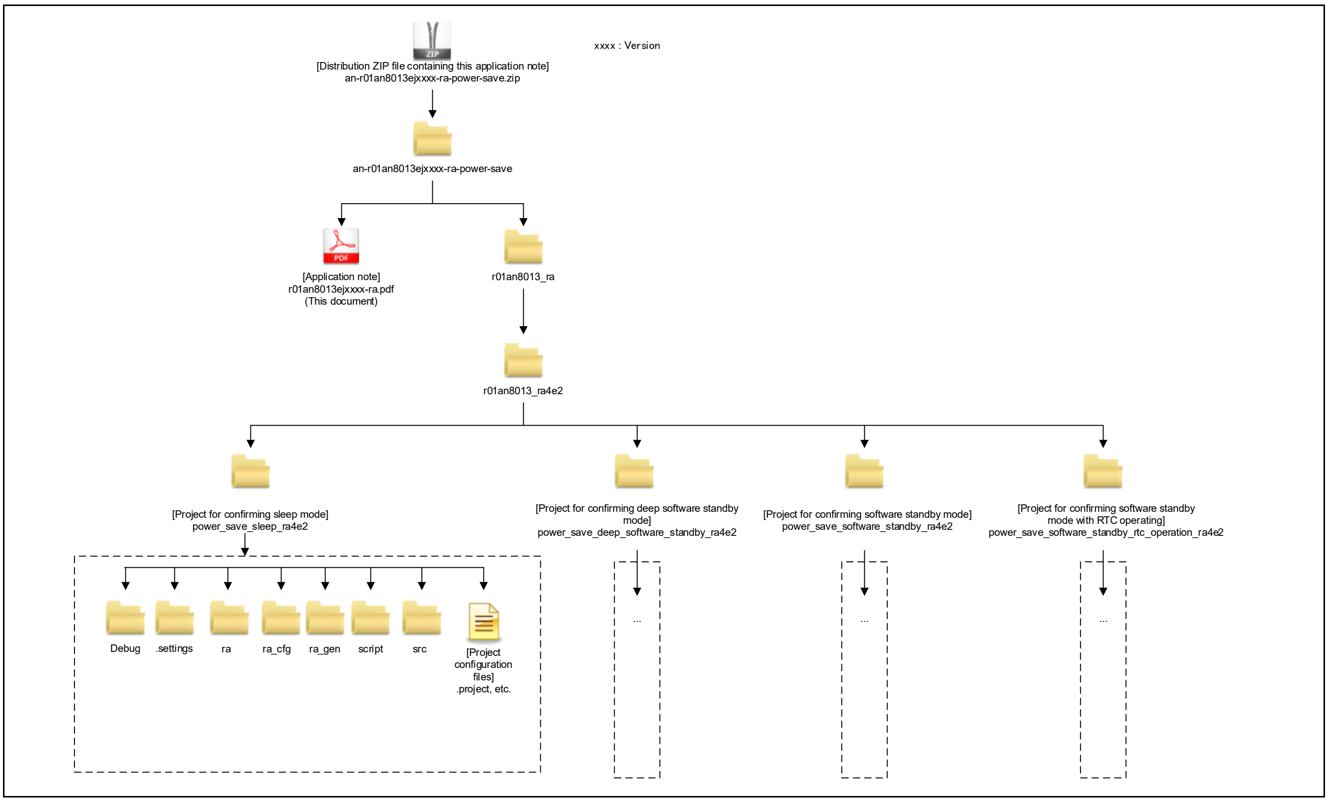

Figure 4.5 Project Composition

Unzipping the ZIP archive in which this application note is distributed creates a folder of the same name that contains the folders and files shown above.

The “r01an8013_ra” folder contains separate folders for each target device, and each of these folders contain projects corresponding to the various low power modes. Each project can be imported into the e^2 studio workspace and then run.

4.4 File Structure

The file structure used by the sample program is shown below. The main program is generated in the folder “project folder\src\”.

“power_save_sleep_ra4e2¥src¥hal_entry.c”

Table 4.4 lists the file related to the main program of the project for confirming the low power mode.

Table 4.5 File Related to Main Program

File Name	Description
hal_entry.c	Main processing routine for confirming low power mode

The source code is generated in the folder “project folder\src\smc_gen” by the Code Generator function of Smart Configurator.

“power_save_sleep_ra4e2¥src¥smc_gen”

Parts of the source code generated by the Code Generator function of Smart Configurator that are used without modification are omitted.

4.5 Option Setting Memory

Table 4.5 is lists the option setting memory configuration used by the sample program. Change the settings to the optimum values for your system as needed.

Table 4.6 Option Setting Memory Configuration of Sample Program

Symbol	Address	Setting Value	Description
OFS0	0x0100_A100～0x0100_A103	FFFF FFFFh	After a reset, IWDT and WDT are stopped.
OSIS	0x0100_A120～0x0100_A12F	FFFF FFFFh	Retains setting value
SAS	0x0100_A134～0x0100_A137	FFFF FFFFh	Retains setting value
OFS1	0x0100_A200～0x0100_A203	FFFF FFFFh	After a reset, voltage monitoring 0 reset and HOCO is disabled.
BPS	0x0100_A240～0x0100_A243	FFFF FFFFh	Retains setting value
PBPS	0x0100_A260～0x0100_A263	FFFF FFFFh	Retains setting value

4.6 Constants

Table 4.6 lists the constants used by the sample program.

Table 4.7 Constants Used by Sample Program

Constant Name	Setting Value	Description
LED_ON	(1)	LED is on
LED_OFF	(0)	LED is off
LED2	(PORT1.PODR.BIT.B4)	LED2 (P104) PODR register bit

4.7 Variables

Table 4.8 lists the global variables.

Table 4.8 Global Variables

Type	Variable Name	Description
extern volatile bool	g_sw_pressed	State indicating whether or not switch is depressed. False: Not depressed True: Depressed

4.8 Functions

Table 4.9 lists the functions used by the sample program.

Table 4.9 Functions Used by Sample Program

Function Name	Description
hal_entry	Main processing routine
save_power_port_init	Initial port settings
save_power_peripheral_init	Initial peripheral function settings
save_power_wait_trigger	Wait for low power mode transition trigger
external_irq10_ds_cd	IRQ10-DS interrupt handler
tc_start	Initial RTC settings and startup

4.9 Function Specifications

The specifications of sample program functions are presented below.

hal_entry	
Outline	Main processing routine
Header	None
Declaration	void main (void)
Description	After initial settings, transitions to and then resumes from low power mode.
Arguments	None
Return Value	None

save_power_port_init	
Outline	Initial port settings
Header	None
Declaration	void save_power_port_init (void)
Description	Makes initial port settings.
Arguments	None
Return Value	None

save_power_peripheral_init	
Outline	Initial peripheral function settings
Header	None
Declaration	void save_power_peripheral_init (void)
Description	Makes initial peripheral function settings.
Arguments	None
Return Value	None

save_power_wait_trigger	
Outline	Wait for low power mode transition trigger
Header	None
Declaration	void save_power_wait_trigger (void)
Description	Waits for the low power mode transition trigger (IRQ) to occur.
Arguments	None
Return Value	None

external_irq10_ds_cd	
Outline	Setting IRQ10-DS interrupt handler
Header	common_data.h
Declaration	void external_irq10_ds_cd(external_irq_callback_args_t *p_args)
Description	Sets the IRQ10-DS interrpt handling. This handler sets the value to true to indicate that the switch has been pressed
Arguments	external_irq_callback_args_t *p_args
Return Value	None
Note	This function is generated by the FSP Configurator's code generation feature; however, its implementation must be defined in hal_entry.c.

rtc_start	
Outline	RTC Initial settings abnd start up
Header	None
Declaration	void rtc_start(void)
Description	Initializes RTC and starts it in binary counter mode.
Arguments	None

4.10 Flowcharts

As an example, a flowchart of the sample program for the RA4E2 is shown below.

The sample program described in this application note includes code generated by the Code Generator function of Smart Configurator. Refer to4.10.5, IRQ0 Interrupt Handler, regarding the modified portions of the source code generated by the Code Generator function of Smart Configurator. Flowcharts have been omitted for unmodified source code generated by the Code Generator function of Smart Configurator.

4.10.1 Main Processing Routine

Figure 4.6 is a flowchart of the main processing routine.

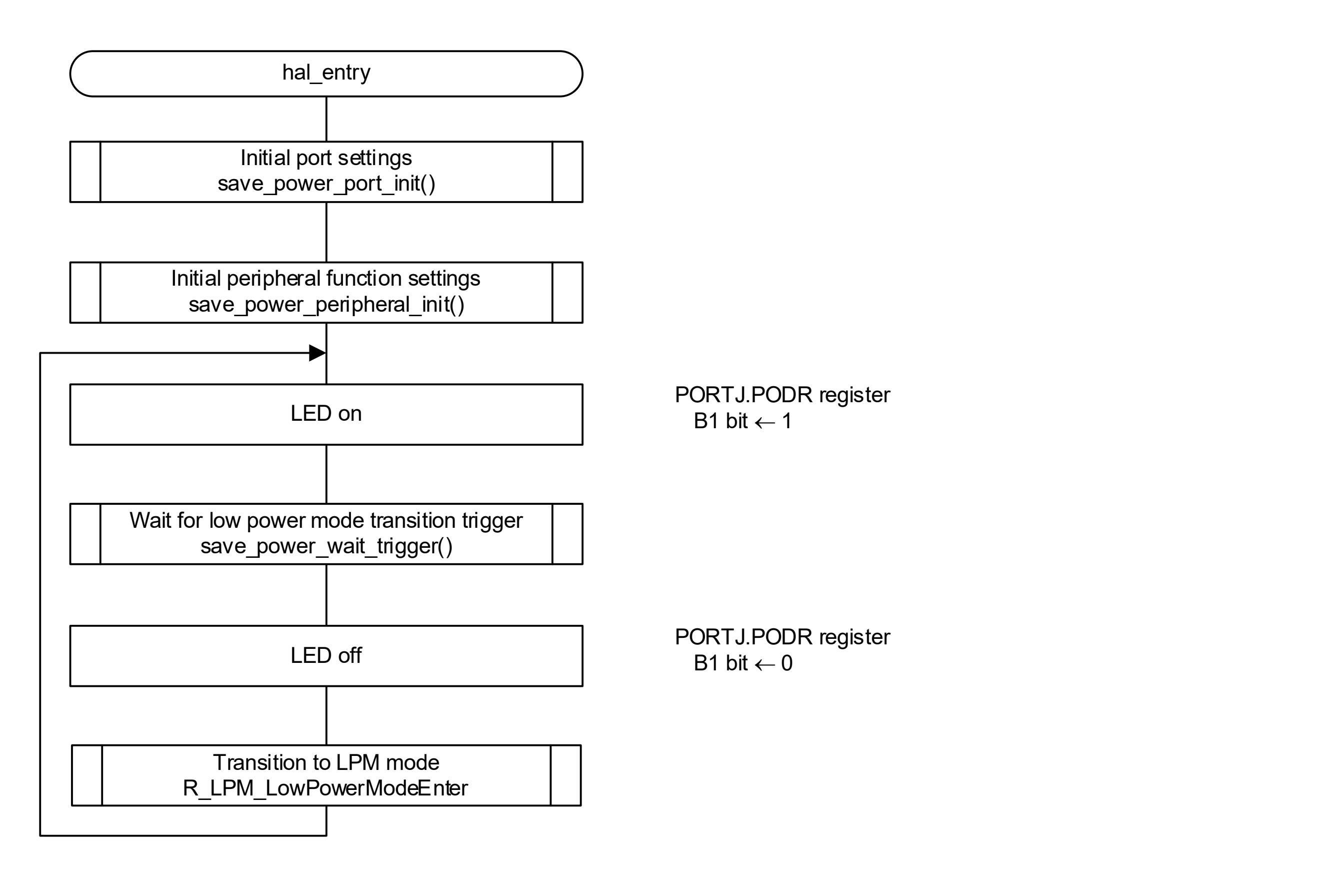

Figure 4.6 Main Processing Routine

Figure 4.7 is a flowchart of the main processing routine of the sample program when transitioning to deep software standby mode.

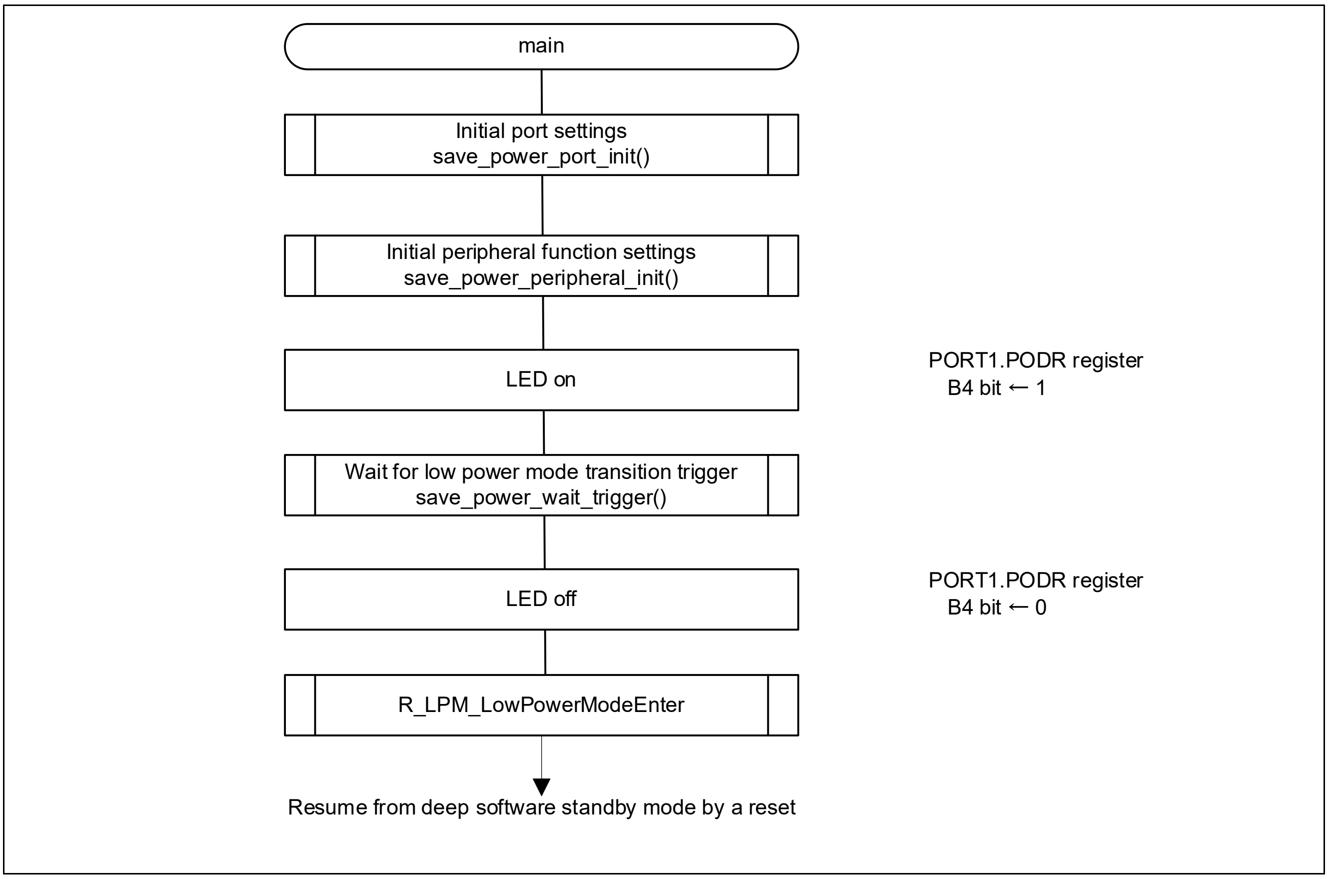

Figure 4.7 Main Processing Routine of Sample Program when Transitioning to Deep Software Standby Mode

4.10.2 Initial Peripheral Function Settings

Figure 4.8 is a flowchart of the initial peripheral function settings without RTC.

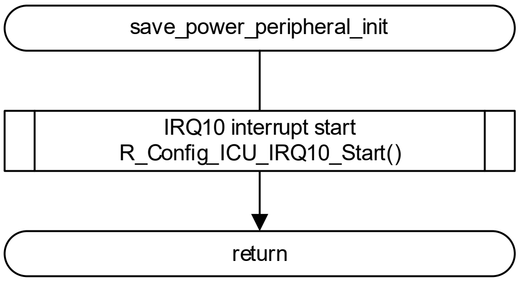

Figure 4.8 Initial Peripheral Function Settings without RTC

4.10.3 RTC Initialization

Figure 4.9 is a flowchart of RTC initialization.

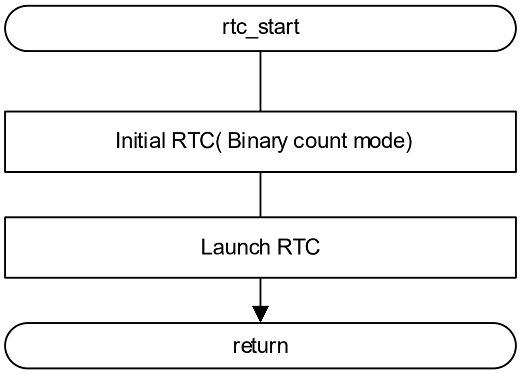

Figure 4.9 RTC Initialization

4.10.4 Wait for Low Power Mode Transition Trigger

Figure 4.10 is a flowchart of the wait for low power mode transition trigger.

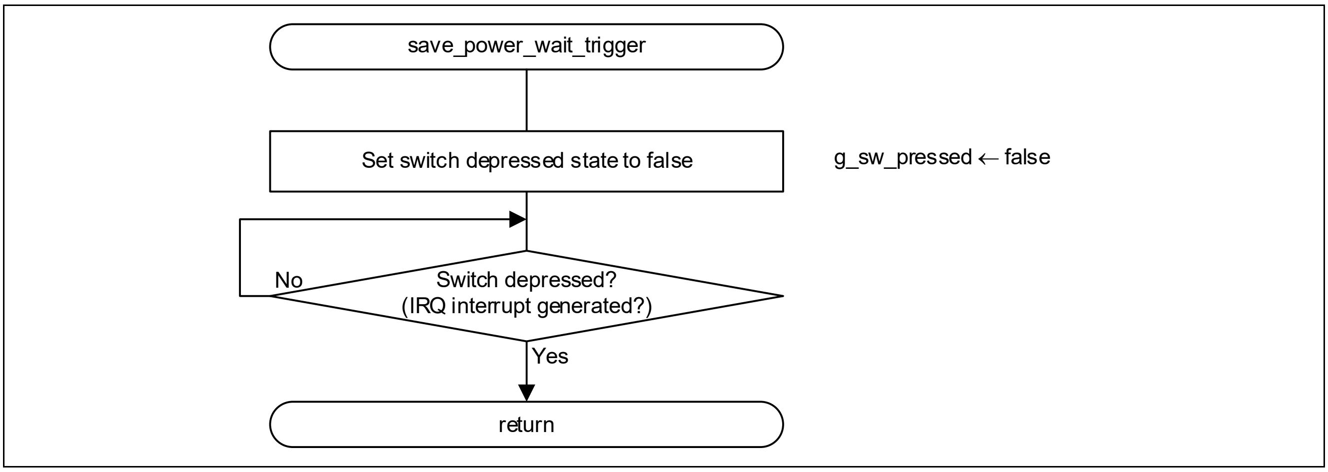

Figure 4.10 Wait for low power mode transition trigger.

4.10.5 IRQ10 Interrupt Handler

Figure 4.11 is a flowchart of the IRQ10 interrupting handler.

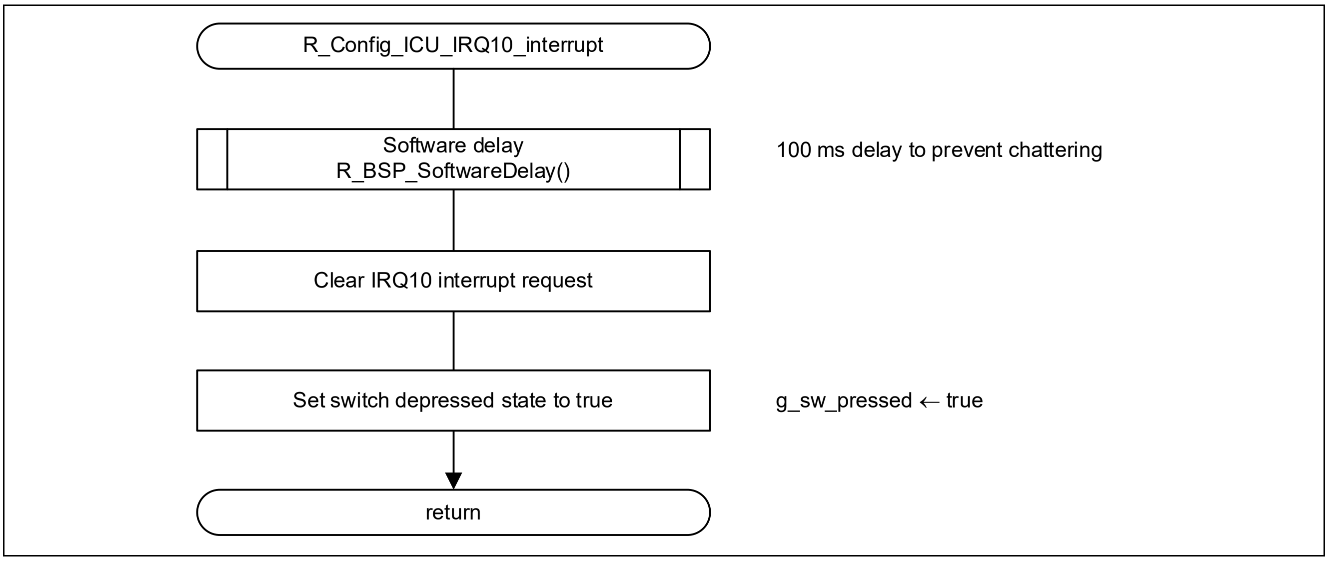

Figure 4.11 IRQ10 Interrupt Handler

5. Overview of Development Environments Including the Smart Configurator

For details on the development environment used in this sample project — including the Smart Configurator — please refer to the instructional video available at the following site.

RA Family Software & Tool Course

6. Notes on Transition to a Low Power Mode

(1) Clocks operate in some low power modes such as sleep mode. To reduce current consumption, stop unnecessary clocks or lower their frequencies by changing the division ratio.

(2) Function-specific notes might apply to some peripheral functions when the operating state changes to the module stop state or when the operating mode changes to a low power mode. For the notes that apply to a specific peripheral function, refer to the documentation for the peripheral function.

(3) In some low power modes such as software standby mode, operation of the I/O ports cannot be changed. Therefore, make sure that the appropriate I/O settings are specified before the operating mode changes to such a low power mode.
Table 6.1 shows Setting example of I/O port.

Table 6.1 Setting example of I/O port

State of the I/O port	Port settings to reduce current consumption
Input setting	Fix the input level (High or Low). There are many pins that become high impedance during standby mode by setting them to input.
Output setting	Output a level that does not create a potential difference in external pull-up/pull-down resistors. If a pin of connected device is in an input state and there is no external/internal pull-up/pull-down resistor, output low.

(4) A board consumes more current if an emulator is connected to the board. Therefore, when you measure the current consumption of a board, make sure that no emulator is connected to the board.

(5) A current consumption measurement program cannot correctly measure the current consumption if the program is written from e^2 studio in debug mode. Use Renesas Flash Programmer or another flash memory programming software tool to write the program.

7. Appendix

7.1 Operating Frequencies of Project

Table 7.1 is list the operating frequencies used when confirming the operation of the program described in this application note.

Table 7.1 Operating Frequencies of Project for Confirmation of Sleep Mode and Deep Sleep Mode

Name	Operating Frequency	Clock Source	Division Ratio	Multiplication Ratio
Main clock	20 MHz	—	—	—
Sub-clock	Stopped	—	—	—
High-speed on-chip oscillator (HOCO)	Stopped	—	—	—
Low-speed on-chip oscillator (LOCO)	Stopped	—	—	—
IWDT dedicated on-chip oscillator	Stopped	—	—	—
PLL	200 MHz	Main clock	2	2
System clock (ICLK)	100 MHz	PLL	1	—
Peripheral module clock A (PCLKA)	1.563 MHz	PLL	64	—
Peripheral module clock B (PCLKB)	1.563 MHz	PLL	64	—
Peripheral module clock C (PCLKC)	1.563 MHz	PLL	64	—
Peripheral module clock D (PCLKD)	1.563 MHz	PLL	64	—
FlashIF clock (FCLK)	6.250 MHz	PLL	16	—
USB clock (USBCLK)	Stopped	—	—	—

8. Importing a Project

The sample code is provided as the e² studio project. This section describes importing a project into the e² studio. After importing a project, confirm that the build settings and the debug settings are correct.

8.1 Importing a Project into e² studio

Follow the steps below to import your project into the e² studio.
(Windows/dialogs may differ depending on the e² studio version used.)

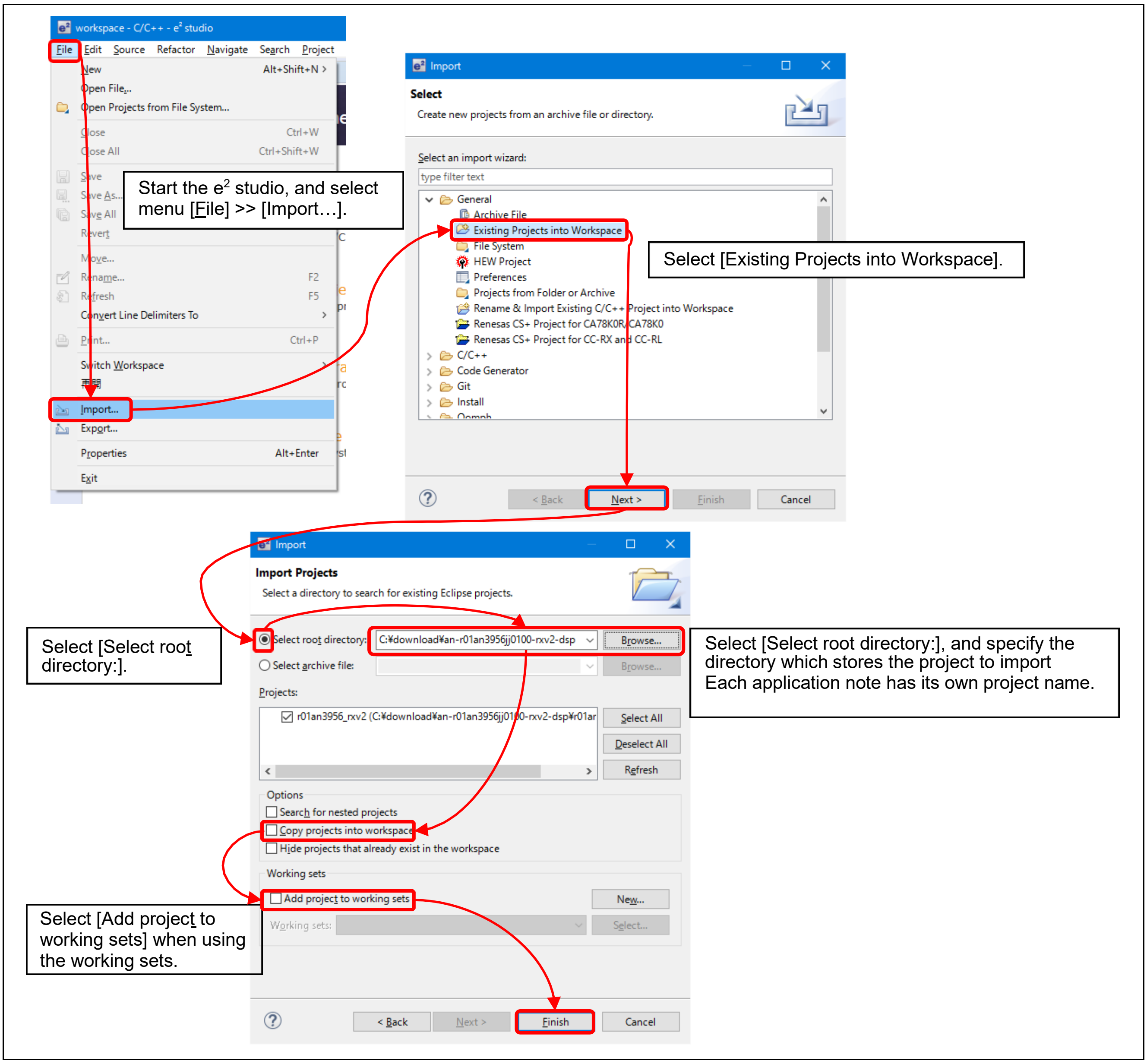

Figure 8.1 Importing a Project into e² studio.

9. Sample Code

Sample code can be downloaded from the Renesas Electronics website.

10. Reference Documents

User's Manual: Hardware
RA4E2 Group User's Manual: Hardware (R01UH0996)
(The latest version can be downloaded from the Renesas Electronics website.)

User's Manual: Flexible Software Package
Flexible Software Package User's manual(R11UM0155)
(The latest version can be downloaded from the Renesas Electronics website.

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

User's Manual: Development Tools
Refer below Site,
e² studio

User's Manual: Renesas Evaluation Kit
EK-RA4E2 v1 User's manual(R20UT5175)
(The latest information can be downloaded from the Renesas Electronics website.)

Revision History

Rev.	Date	Description	
		Page	Summary
1.00	Sep. 30, 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/.