RX23W Group

Bluetooth Mesh Module Using Firmware Integration Technology

Introduction

This application note describes the Bluetooth® Mesh module which uses Firmware Integration Technology (FIT). This module provides the features to perform many-to-many wireless communication in a mesh network which is compliant with Bluetooth Mesh Networking Specifications.

In this document, this module is referred to as the Mesh FIT module.

Target Device

RX23W Group

Related Documents

- Bluetooth Core Specifications (bluetooth.com/specifications/)
- Bluetooth Mesh Networking Specifications (bluetooth.com/specifications/)
- CC-RX Compiler User's Manual (R20UT3248)
- e^2 studio User's Manual: Getting Started Guide (R20UT4374)
- RX Smart Configurator User Guide: e^2 studio (R20AN0451)
- Firmware Integration Technology User's Manual (R01AN1833)
- Adding Firmware Integration Technology Modules to Projects (R01AN1723)
- Adding Firmware Integration Technology Modules to CS+ Projects (R01AN1826)
- RX Family Board Support Package Module Using Firmware Integration Technology (R01AN1685)
- RX Family Flash Module Using Firmware Integration Technology (R01AN2184)
- RX23W Group BLE Module Firmware Integration Technology (R01AN4860)
- RX23W Group Bluetooth Mesh Stack Startup Guide (R01AN4874)
- RX23W Group Bluetooth Mesh Stack Development Guide (R01AN4875)

Contents

1. Overview

1.1 Features

Mesh Fit Module provides many-to-many wireless communication features which are compliant with Bluetooth Mesh Profile 1.0.1 Specification and Bluetooth Mesh Model 1.0.1 Specification. This module supports the following features.

Bluetooth Core Mesh Profile features:

- Provisioning (both Provisioning Server and Provisioning Client)
- Access
- Upper Transport
  - Friendship (both Friend feature and Low Power feature)
- Lower Transport
- Network
  - Relay
  - Proxy (both Proxy Server and Proxy Client)
- Bearer
  - ADV Bearer
  - GATT Bearer
- Foundation Model
  - Configuration Model (both Configuration Server and Configuration Client)
  - Health Model (both Health Server and Health Client)

Bluetooth Mesh Model features:

- Generic Models
  - OnOff, Power OnOff, Power OnOff Setup
  - Level, Power Level, Power Level Setup
  - Default Transition Time
  - Battery
  - Location, Location Setup
  - Manufacturer Property, Admin Property, User Property, Client Property
- Sensor Model
  - Sensor, Sensor Setup
- Time Model
- Scene Model
  - Scene, Scene Setup
- Scheduler Model
  - Scheduler, Scheduler Setup
- Light Models
  - Light Lightness, Light Lightness Setup
  - Light CTL, Light CTL Setup
  - Light HSL, Light HSL Setup
  - Light xyL, Light xyL Setup
  - Light Control

1.2 Software Architecture

Figure 1-1 show the software architecture to use Mesh FIT Module.

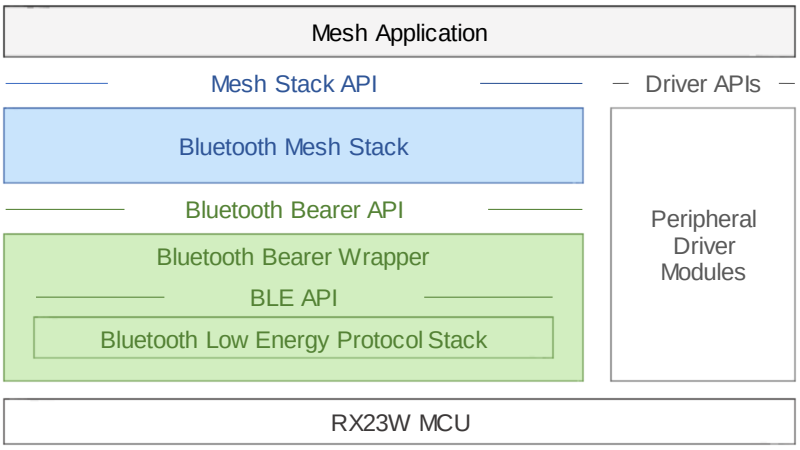

Figure 1-1 Software Architecture

The software architecture to use Mesh FIT Module is composed of the following software:

- **Mesh Application**
  Mesh Application is an application to perform features provided by Bluetooth Mesh Stack.
- **Bluetooth Mesh Stack**
  Bluetooth Mesh Stack is the software that provides applications with many-to-many wireless communication features which are compliant with Bluetooth Mesh Networking Specifications.
- **Bluetooth Bearer**
  Bluetooth Bearer is the abstraction layer that provides wrapper functions of Bluetooth Low Energy Protocol Stack and Bluetooth Mesh Stack.
- **Bluetooth Low Energy Protocol Stack**
  Bluetooth Low Energy Protocol Stack is the software that provides upper layers with wireless communication features which is compliant with the Bluetooth Low Energy specifications.

Sample program of Mesh Application is included in the demo project in the package of Mesh FIT Module (R01AN4930).

Bluetooth Mesh Stack and Bluetooth Bearer are included in Mesh FIT Module (R01AN4930).

Bluetooth Low Energy Protocol Stack is included in BLE FIT Module (R01AN4860).

1.3 File Composition

File composition of Mesh FIT Module (r_mesh_rx23w) is shown as follows:

```
r_mesh_rx23w
   |   r_mesh_rx23w_if.h                          Mesh Stack API Header File
   |   readme.txt                                 Mesh FIT Module Information File
   |
   +---doc\
   |   |   blemesh_api.chm                        Mesh Stack API Specification Manual
   |   +---en\
   |   |       r01an4930ej0130-rx23w-blemesh.pdf  Mesh FIT Module Application Note (en)
   |   +---ja\
   |           r01an4930jj0130-rx23w-blemesh.pdf  Mesh FIT Module Application Note (ja)
   |
   +---json                                       Service Definition Files for the QE for BLE
   |       mesh_provisioning.service.json         Mesh Provisioning Service Definition
   |       mesh_proxy.service.json                Mesh Proxy Service Definition
   |
   +---lib\
   |       lib_ble_ms_ccrx.lib                    Mesh Stack Library
   |
   +---src\
       +---bearer\                                Bluetooth Bearer
       +---drivers\                               Mesh Drivers
       +---include\                               Mesh Stack Header Files
```

To use the features provided by Mesh FIT Module, Mesh FIT Module must be added to a project. Regarding how to add the module to a project, refer to Chapter 4 in this document.

1.4 API Specification

To perform the features provided by Mesh FIT Module, it is necessary to use API of Mesh Stack included in Mesh FIT Module. Regarding the specification of Mesh Stack API, refer to Mesh Stack API Specification Manual "doc\blemesh_api.chm".

1.5 API Header File

To use Mesh FIT Module, include "r_mesh_rx23w_if.h" header file. Mesh Stack API is defined by multiple header files in "src\include\", but you can include all header files by including just "r_mesh_rx23w_if.h".

2. Requirements

Requirements to develop applications using Mesh FIT Module are described in this chapter.

2.1 Hardware Requirements

The following hardware functions must be supported by the MCU you use.

- Bluetooth Low Energy (BLE)
- Compare Match Timer (CMT)
- 8-Bit Timer (TMR)
- E2 Data Flash

2.2 Software Requirements

Mesh FIT Module requires the following FIT modules.

- **r_bsp**: Board Support Package (BSP FIT Module version 5.66 or later)
- **r_ble_rx23w**: Bluetooth Low Energy (BLE FIT Module version 2.11 or later)
- **r_flash_rx**: Data Flash memory (Flash FIT Module version 4.60 or later)

BLE FIT Module needs the following FIT modules.

- **r_lpc_rx**: Low Power Control (LPC FIT Module)
- **r_cmt_rx**: Compare Match Timer NOTE (CMT FIT Module version 4.70 or later)
- **r_sci_rx**: Serial Communication Interface (SCI FIT Module)
- **r_byteq**: Byte Queues/Circular Buffers (BYTEQ FIT Module)
- **r_gpio_rx**: General Purpose I/O (GPIO FIT Module)
- **r_irq_rx**: Interrupt Request (IRQ FIT Module)

NOTE: BLE FIT Module uses CMT2 and CMT3 directly, so CMT FIT Module can only CMT 0 and CMT1.

2.3 Supported Toolchain

It has been confirmed that MESH FIT Module works with the following toolchain.

- **IDE**: Renesas Electronics e^2 studio 2022-10
- **Compiler**: Renesas Electronics C/C++ Compiler for RX Family (CC-RX) V2.08.01
- **Endian**: Little Endian
- **Board**: Target Board for RX23W (RTK5RX23W0C00000BJ)
  Target Board for RX23W module (RTK5RX23W0C01000BJ)
  Renesas Solution Starter Kit (RSSK) for RX23W (RTK5523W8AC00001BJ)

2.4 Sections

Mesh Stack included in Mesh FIT Module will be located by the section names listed in Table 2-1.

Table 2-1 Section Names of Mesh Stack

Program Area Name	Mesh Stack Section		
	Name	Attribute	Alignment
program	MESH_P	code	1byte
constant	MESH_C	romdata	4byte
	MESH_C_2	romdata	2byte
	MESH_C_1	romdata	1byte
initialized data	MESH_D	romdata	4byte
	MESH_D_2	romdata	2byte
	MESH_D_1	romdata	1byte
	MESH_R	data	4byte
	MESH_R_2	data	2byte
	MESH_R_1	data	1byte
uninitialized data	MESH_B	data	4byte
	MESH_B_2	data	2byte
	MESH_B_1	data	1byte
switch statement branch table	MESH_W	romdata	4byte
	MESH_W_2	romdata	2byte
	MESH_W_1	romdata	1byte
literal	MESH_L	romdata	4byte

Attribute: code stores execution instructions
data stores data that can be changed
romdata stores fixed data

Regarding the specification of Sections, refer to Chapter 6 of "CC-RX Compiler User's Manual" (R20UT3248).

Program using Mesh FIT Module is required to transfer initialization data of ROM to initialized data section of RAM. Regarding configuration to transfer initialization data, refer to Subsection 5.6.1.

2.5 Program Size

Table 2-2 shows the program size of Mesh FIT Module. If there are unreferenced variables or functions, actual ROM size used by Mesh FIT Module is reduced by optimization of linkage. Also, RAM size that Mesh FIT Module needs can be changed depends on configuration of the module.

Table 2-2 Total Program Size of Mesh FIT Module

Device	Compiler	Category	Size
RX23W Group	CC-RX V2.08.01	ROM	67,466byte
		RAM	9,974byte
Conditions			
Mesh FIT Module Default Configuration (r_mesh_rx23w\ref\r_mesh_rx23w_config_reference.h) Compile Options Optimization Level: Level 2: Overall Optimization (-optimize=2) Optimization Option: Optimization with emphasis on size (-size) Link Options Optimization Option: No Optimization at Linkage (-nooptimize)			

Table 2-3 shows the program size of the demo project included in the package of Mesh FIT Module. For more information on the demo projects, refer to "RX23W Group Bluetooth Mesh Stack Development Guide" (R01AN4875)

Table 2-3 Program Size of Demo Project included in Mesh FIT Module Package

Device	Compiler	Category	Size
RX23W Group	CC-RX V2.08.01	ROM	306,876byte (Mesh FIT Module 62,622byte)
		RAM	45,795byte (Mesh FIT Module 9,970byte)
Conditions			
Project Server Models Project for Target Board for RX23W (rsskrx23w_mesh_server) Compile Options Optimization Level: Level 2: Overall Optimization (-optimize=2) Optimization Option: Optimization with emphasis on size (-size) Link Options Optimization Option: Deleting variables/functions that are not referenced (-optimize=symbol_delete)			

3. FIT Module Configurations

3.1 Mesh FIT Module

Mesh FIT Module has parameters that can be changed depends on each mesh network scale and each requirement for node. These parameters are defined in "r_ble_rx23w_config.h" as configuration macros listed in Table 3-1.

If you use Smart Configurator, each value of the configuration macros can be set with GUI, and those value are reflected in "r_ble_rx23w_config.h" when Mesh FIT Module is added to a project.

Table 3-1 Configuration Macros of Mesh FIT Module

Configuration Macro	Description
MESH_CFG_NUM_NETWORK_INTERFACES *Default: 2	The number of bearers used for Mesh Network MIN: 1 MAX: (1 + BLE_CFG_RF_CONN_MAX) First bearer is ADV bearer and subsequent bearers are GATT bearers which can establish connections concurrently. When this configuration is set to 1, only ADV bearer can be used.
MESH_CFG_NUM_PROVISIONING_INTERFACES *Default: 2	The number of bearers used for Provisioning MIN: 1 MAX: 2 When this configuration is set to 1, only PB-ADV bearer can be used. When this configuration is set to 2, PB-ADV bearer and one PB-GATT bearer can be used.
MESH_CFG_UNPROV_DEVICE_BEACON_TIMEOUT *Default: 200	Transmission interval of Unprovisioned Device Beacon [msec] MIN: 20 When only PB-ADV is used, Unprovisioned Device Beacon is transmitted at the intervals of this configuration. When only PB-GATT is used, Connectable Advertising PDU is transmitted at the intervals of this configuration. When both PB-ADV and PB-GATT are used, Unprovisioned Device Beacon and Connectable Advertising PDU are transmitted alternately at the intervals of this configuration.
MESH_CFG_NET_CACHE_SIZE *Default: 10	The maximum number of nodes that Network Message Cache can store MIN: 2 If message from new node is received when Network Message Cache stores cache information for the maximum number of nodes, cache information for the oldest node will be removed.
MESH_CFG_NET_SEQNUM_CACHE_SIZE *Default: 32	The number of SEQ number that Network Message Cache can store for each node MIN: 32
MESH_CFG_MAX_SUBNETS *Default: 4	Maximum number of subnet information such as Network Key and NID MIN: 1

MESH_CFG_MAX_DEV_KEYS *Default: 4	Maximum number of Device Key MIN: 1 When Configuration Client Model is not used, it is enough to set this configuration to 1.
MESH_CFG_PROXY_FILTER_LIST_SIZE *Default: 2	Maximum number of addresses that can be added to each Proxy List MIN: 1
MESH_CFG_NET_SEQ_NUMBER_BLOCK_SIZE *Default: 2048	Distance between SEQ number for writing to Data Flash memory MIN: 1 SEQ number will be saved to Data Flash at the distance of this configuration. When MCU is reset, SEQ number resumes from the next distance. e.g.) When this configuration is 2048, SEQ number is written to Data Flash every time SEQ number reaches a multiple of 2048 such as 2048 and 4096. If MCU is reset when SEQ number is 3000, SEQ number resumes from 4096. The shorter this configuration is, the more frequently SEQ number is written to Data Flash. The longer this configuration is, the bigger SEQ number is skipped after resetting MCU.
MESH_CFG_NET_TX_COUNT *Default: 1	Default value of Network Transmit Count state MIN: 0 MAX: 7
MESH_CFG_NET_TX_INTERVAL_STEPS *Default: 4	Default value of Network Transmit Interval Steps state MIN: 0 MAX: 31
MESH_CFG_NET_RELAY_TX_COUNT *Default: 0	Default value of Relay Retransmit Count state MIN: 0 MAX: 7
MESH_CFG_NET_RELAY_TX_INTERVAL_STEPS *Default: 9	Default value of Relay Retransmit Interval Steps state MIN: 0 MAX: 31
MESH_CFG_PROXY_SUBNET_NETID_ADV_TIMEOUT *Default: 300	Transmission interval of Proxy Advertisement with Network ID [msec] MIN: 20
MESH_CFG_PROXY_SUBNET_NODEID_ADV_TIMEOUT *Default: 300	Transmission interval of Proxy Advertisement with Node Identity [msec] MIN: 20
MESH_CFG_PROXY_NODEID_ADV_TIMEOUT *Default: 60	Transmission period of Proxy Advertisement with Node Identity [sec] MIN: 1
MESH_CFG_NET_TX_QUEUE_SIZE *Default: 64	Size of transmission queue for Network PDUs MIN: 2
MESH_CFG_MAX_LPNS *Default: 1	Maximum number of Low Power Nodes that Friend Node can establish Friendship with MIN: 1
MESH_CFG_REPLAY_CACHE_SIZE *Default: 10	Size of Replay Protection Cache MIN: 2

MESH_CFG_REASSEMBLED_CACHE_SIZE *Default: 8	Size of reception message cache of Segmentation and Reassembly (SAR) MIN: 2
MESH_CFG_FRND_POLL_RETRY_COUNT *Default: 10	The number of times to retry Friend Poll message when Low Power Node does not receive Friend Update message MIN: 1
MESH_CFG_LTRN_SAR_CTX_MAX *Default: 8	The number of contexts of Segmentation and Reassembly (SAR) mechanism used for transmitting and receiving Segmented Message MIN: 2
MESH_CFG_LTRN_RTX_TIMEOUT *Default: 300	Retransmission interval of segmented message [msec] MIN: 200
MESH_CFG_LTRN_RTX_COUNT *Default: 2	The number of times to retransmit segmented message MIN: 2 MAX: 255
MESH_CFG_LTRN_ACK_TIMEOUT *Default: 200	Transmission interval of Segmented Acknowledgement message [msec] MIN: 200
MESH_CFG_LTRN_INCOMPLETE_TIMEOUT *Default: 20	Cancel timeout time of receiving segmented message [sec] MIN: 10
MESH_CFG_FRND_RECEIVE_WINDOW *Default: 100	Reception windows size of Low Power Node [msec] MIN: 100 MAX: 255
MESH_CFG_FRIEND_MESSAGEQUEUE_SIZE *Default: 15	Size of Message Queues for each Low Power Node MIN: 2
MESH_CFG_FRIEND_SUBSCRIPTION_LIST_SIZE *Default: 8	Maximum number of Friend Subscription Lists for each Low Power Node MIN: 1
MESH_CFG_LPN_CLEAR_RETRY_TIMEOUT_INITIAL *Default: 1000	Retransmission interval of Friend Clear message [msec] MIN: 1000
MESH_CFG_LPN_CLEAR_RETRY_COUNT *Default: 5	The total number of times an LPN sends Friend clear message if it has not received Friend Clear Confirmation from a Friend Node. MIN: 1
MESH_CFG_TRN_FRNDREQ_RETRY_TIMEOUT *Default: 1200	Transmission period of Friend Request message [msec] MIN: 1100
MESH_CFG_ACCESS_ELEMENT_COUNT *Default: 4	Maximum number of Elements MIN: 1
MESH_CFG_ACCESS_MODEL_COUNT *Default: 20	Maximum number of Models MIN: 1
MESH_CFG_MAX_APPS *Default: 8	Maximum number of Application Keys MIN: 1
MESH_CFG_MAX_VIRTUAL_ADDRS *Default: 8	Maximum number of Virtual Address MIN: 1
MESH_CFG_MAX_NON_VIRTUAL_ADDRS *Default: 8	Maximum number of Non-virtual Address (Unicast Address or Group Address) MIN: 1

MESH_CFG_MAX_NUM_TRANSITION_TIMERS *Default: 5	The number of State Transition Timers for models MIN: 1
MESH_CFG_MAX_NUM_PERIODIC_STEP_TIMERS *Default: 5	The number of Periodic Publication Timers for models MIN: 1
MESH_CFG_CONFIG_SERVER_SNB_TIMEOUT *Default: 10	Transmission Interval of Secure Network Beacon [sec] MIN: 10 MAX: 600
MESH_CFG_HEALTH_SERVER_MAX *Default: 2	Maximum number of Health Server Model MIN: 1
MESH_CFG_LIGHT_LC_SERVER_MAX *Default: 1	Maximum number of Light LC Server Model MIN: 1
MESH_CFG_DEFAULT_COMPANY_ID *Default: 0x0036	Company ID registered with Bluetooth SIG MIN: 0x0000 MAX: 0xFFFF For Company ID, refer to Assigned Numbers
MESH_CFG_DEFAULT_PID *Default: 0x0001	Product ID assigned by vendor MIN: 0x0000 MAX: 0xFFFF
MESH_CFG_DEFAULT_VID *Default: 0x0100	Product Version ID assigned by vendor MIN: 0x0000 MAX: 0xFFFF
MESH_CFG_DATA_FLASH_BLOCK_ID *Default: 1	Data Flash Block ID of the first of Data Flash blocks used for storing mesh information MIN: 0 MAX: 7
MESH_CFG_DATA_FLASH_BLOCK_NUM *Default: 5	The number of Data Flash Blocks used for storing mesh information MIN: 1 MAX: 8

3.2 BSP FIT Module

The configuration macros listed in Table 3-2 of BSP FIT Module must be changed to use Mesh FIT Module. Regarding how to change by using Smart Configurator, refer to Subsection5.4.1.

NOTE: When you use Mesh FIT Module, please be sure to change the following configuration.

Table 3-2 Configuration Macro Settings of BSP FIT Module

Configuration Macro	Default Value	Value for Mesh
BSP_CFG_HEAP_BYTES	0x400	0x1000
BSP_CFG_CLOCK_SOURCE	4	1
BSP_CFG_USB_CLOCK_SOURCE	1	0
BSP_CFG_PCKB_DIV	2	1
BSP_CFG_FCK_DIV	2	1
BSP_CFG_CONFIGURATOR_SELECT	0	1

3.3 BLE FIT Module

The configuration macros listed in Table 3-3 of BLE FIT Module should be changed to reduce resources used. Regarding how to change by using Smart Configurator, refer to Subsection 5.4.2.

Table 3-3 Configuration Macro Settings of BLE FIT Module

Configuration Macro	Default Value	Value for Mesh
BLE_CFG_RF_CONN_MAX	7	1
BLE_CFG_RF_ADV_DATA_MAX	1650	31
BLE_CFG_RF_ADV_SET_MAX	4	1
BLE_CFG_RF_SYNC_SET_MAX	2	1
BLE_CFG_CMD_LINE_CH	1	8
BLE_CFG_BOARD_TYPE	0	1: Target Board for RX23W, Target Board for RX23W module 2: RSSK for RX23W

4. How to add FIT Modules

FIT modules must be added to each project. Recommended ways to add FIT modules are shown in either (1) or (2) below which use **Smart Configurator**.

(1) If you use **Smart Configurator** on e^2 studio to add FIT modules.
You can add FIT modules to your project automatically by using **Smart Configurator** on e^2 studio. For more details, refer to "RX Smart Configurator User Guide: e^2 studio" (R20AN0451) and Chapter 5 in this document.

(2) If you use **Smart Configurator** on CS to add FIT modules.
You can add FIT modules to your project automatically by using **Standalone version of Smart Configurator** on CS+. For more details, refer to "RX Smart Configurator User Guide: e^2 studio" (R20AN0451).

(3) If you use **FIT Configurator** on e^2 studio to add FIT modules.
You can add FIT modules to your project automatically by using **FIT Configurator** on e^2 studio. For more details, refer to "Adding Firmware Integration Technology Modules to Projects" (R01AN1723).

(4) If you use add FIT modules manually on CS+.
You can add FIT modules manually on CS+. For more details, refer to "Adding Firmware Integration Technology Modules to CS+ Projects" (R01AN1826).

5. Usage

This chapter describes how to add Mesh FIT module to a new project by using Smart Configurator on e² studio.

5.1 Create a New Project

Select [New]→[C/C++ Project] in [File] menu. Select [Renesas RX] in the left side of [Templates for New C/C++ Project] dialog and [Renesas CC-RX C/C++ Executable Project] in the right side of the dialog, then click [Next] button.

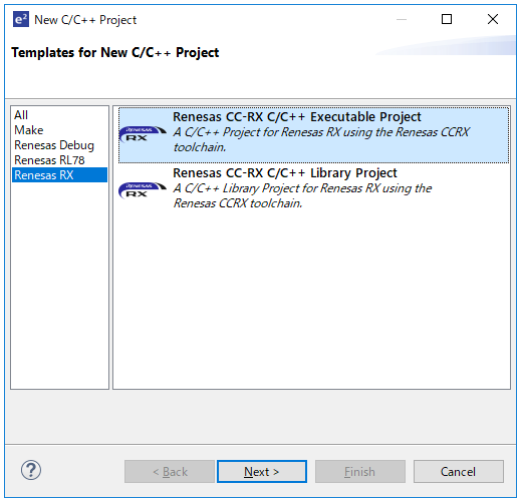

Figure 5-1 Project Template Selection

Key in a project name on [New Renesas CC-RX Executable Project] dialog, then click [Next] button.

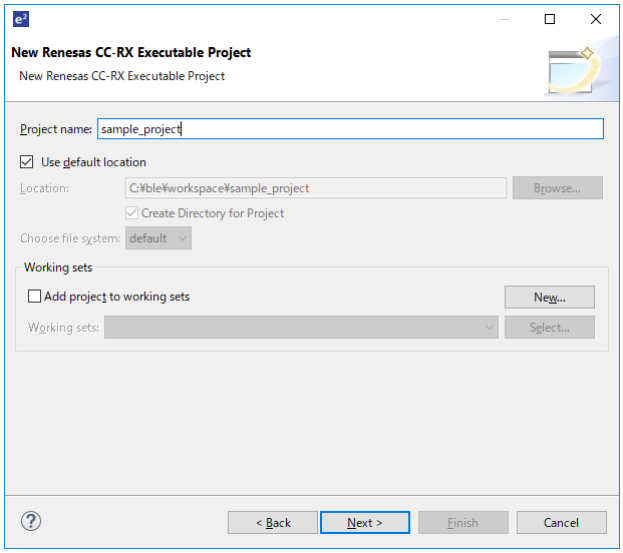

Figure 5-2 New Project Settings

Select [Little] as Endian in Device Settings. Select a device type name of RX23W you use, then click [Next] button.

If you use Target Board for RX23W, select "R5F523W8AxNG". If you use Target Board for RX23W module, select "R5F523W8CxLN". If you use RSSK for RX23W, select "R5F523W8AxBL" when part number of the RSSK is "RTK5523W8AC00001BJ", or select "R5F523W8BxBL" when part number of the RSSK is "RTK5523W8BC00001BJ".

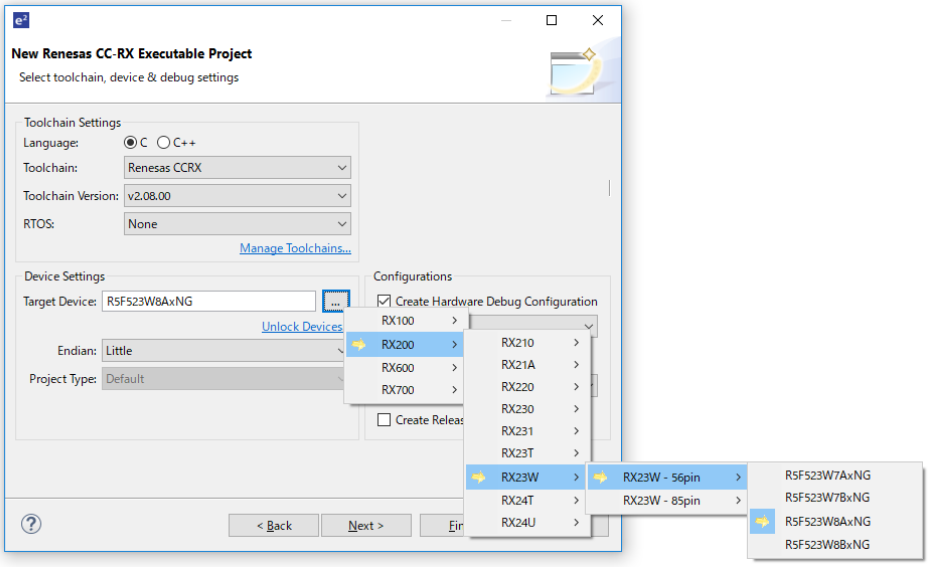

Figure 5-3 Toolchain, Device, and Debug Settings

Put a check in [Smart Configurator] in [Select Coding Assistant settings] dialog, then click [Finish] button.

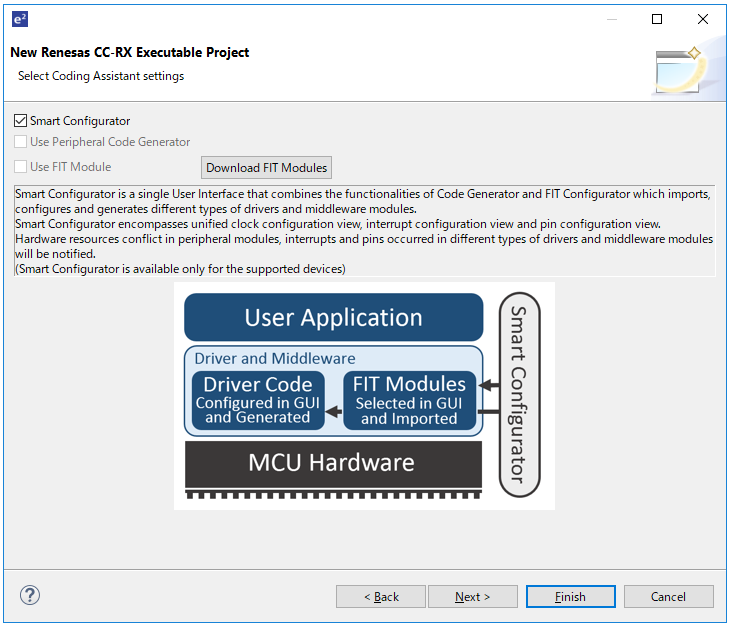

Figure 5-4 Coding Assistant Selection

New project is created in e² studio. Also, Smart Configurator can be shown by clicking "{Project Name}.scfg".

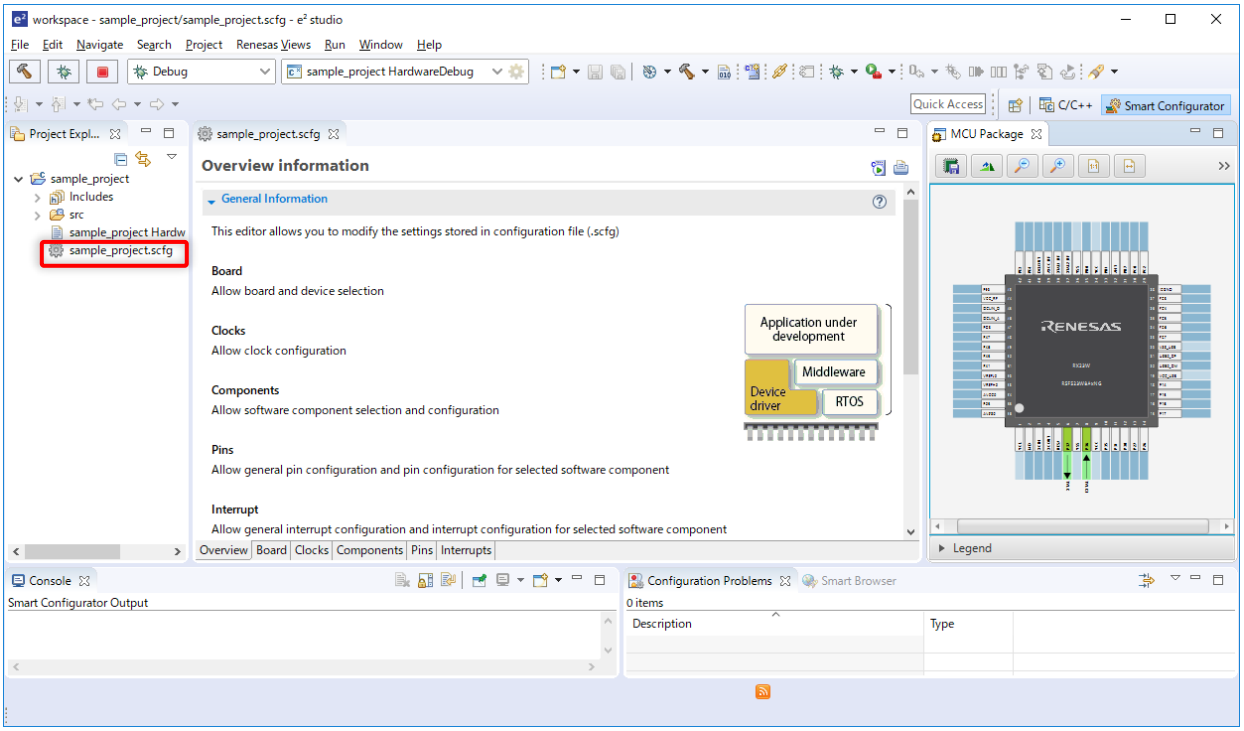

Figure 5-5 Completion of New Project Creation

5.2 Configure Clocks

In [Clocks] tab on Smart Configurator, select clocks and set their clock frequency. To use Mesh FIT Module, following settings are required.

- System Clock (ICLK): 8MHz or over
- Peripheral module Clock B (PCLKB): 8MHz or over

Bluetooth Low Energy Protocol Stack included in BLE FIT Module is optimized for the case that clock frequency of both ICLK and PCLKB is 32MHz. Thus, it is recommended to set clock configuration in which clock frequency of both ICLK and PCLKB become 32MHz.

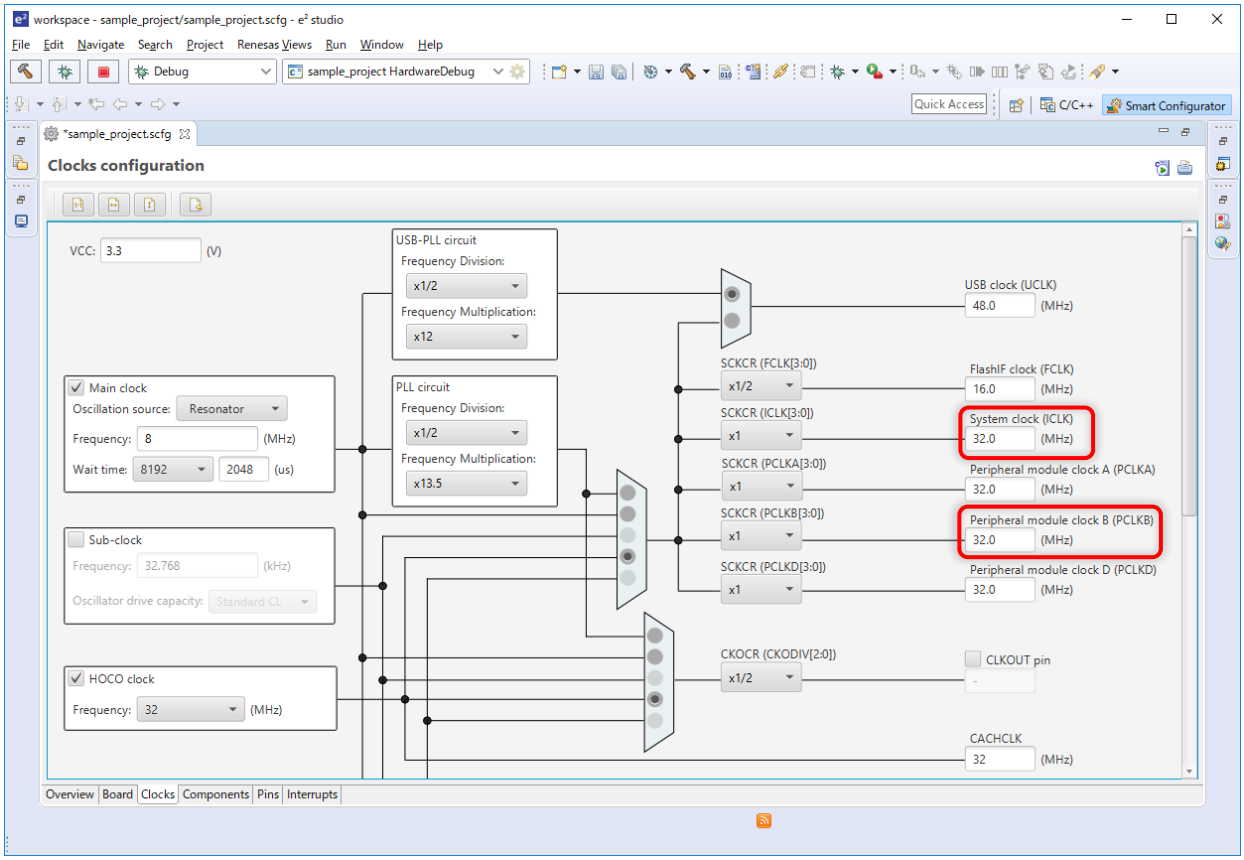

Figure 5-6 Clock Configuration

5.3 Add Components

In [Components] tab on Smart Configurator, add Mesh FIT Module and other necessary FIT modules. Regarding necessary FIT modules, refer to Section 2.2.

Click [Add component] button . In [Software Component Selection] dialog, select necessary FIT modules: r_mesh_rx23w, r_bsp, r_ble_rx23w, r_byteq, r_cmt_rx, r_flash_rx, r_gpio_rx, r_irq_rx, r_lpc_rx, and r_sci_rx, then clock [Finish] button.

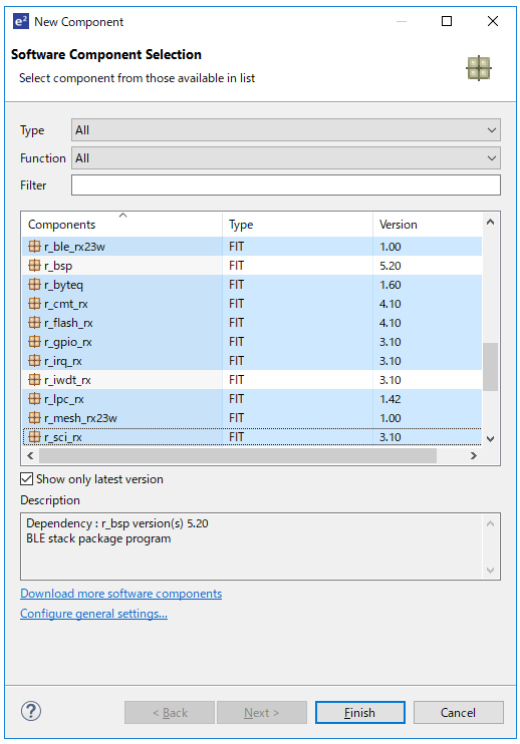

Figure 5-7 Software Components Selection

NOTE: When the necessary FIT modules are not found, click [Download more software components] and download them in accordance with the procedure in [FIT Module Download] dialog.

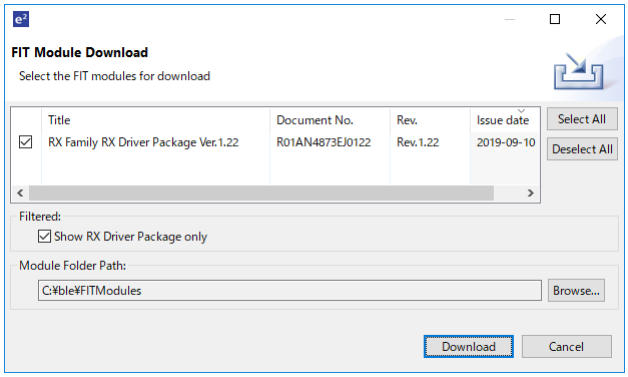

Figure 5-8 FIT Module Download

NOTE: When downloaded FIT modules are not displayed, click [Configure general settings…] and put a check in [Allow blocked FIT modules to be displayed] in [Preferences] dialog.

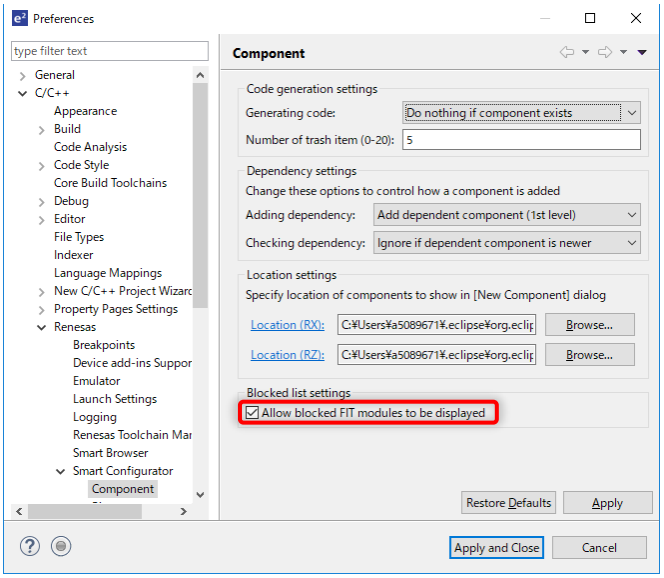

Figure 5-9 Display All FIT Modules

Selected FIT Modules are added on [Components] tab.

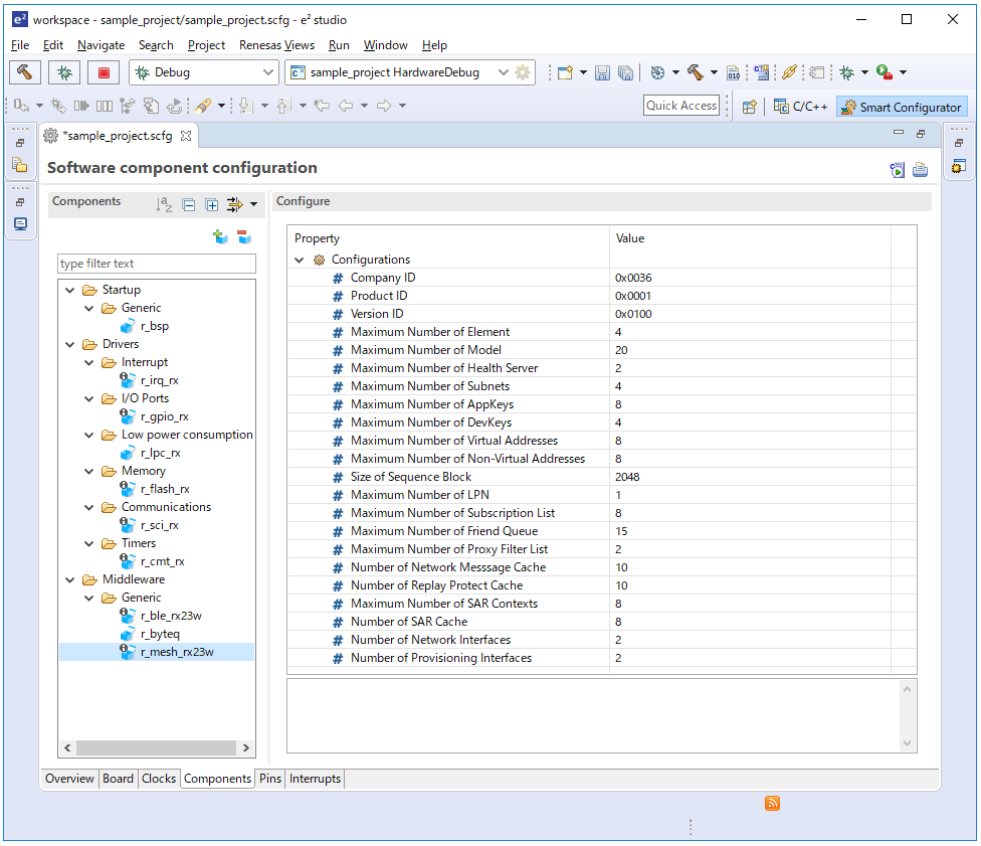

Figure 5-10 Added Software Components

5.4 Component Configurations

In [Components] tab on Smart Configurator, configure FIT modules to use Mesh FIT Module.

5.4.1 r_bsp

To allocate enough heap area size to use Mesh FIT Module, select [r_bsp] and set [Heap size] to "0x1000" in [Components] tab.

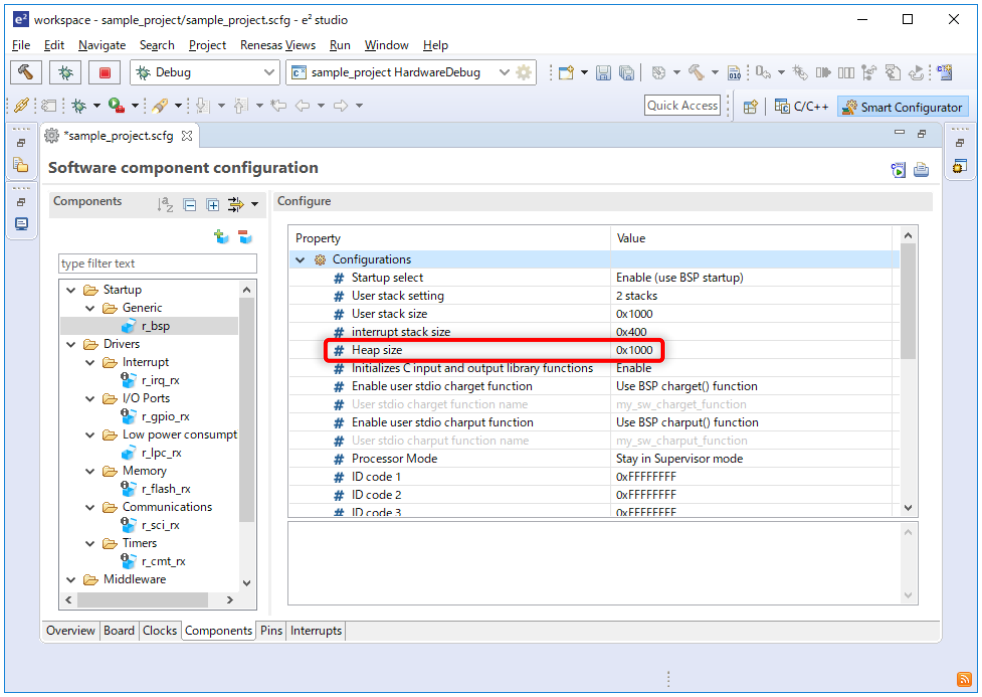

Figure 5-11 r_bsp Configuration

NOTE: If Dependency Warning of FIT Module Version Mismatch occurs, change FIT Module version by right-click menu of FIT Module displayed in Components tab.

5.4.2 r_ble_rx23w

To reduce resources used by BLE FIT Module, select [r_ble_rx23w] in [Components] tab, then set [Maximum number of connections] to "1", set [Maximum advertising data length] to "31", set [Maximum advertising set number] to "1", and set [Maximum periodic sync set number] to "1".

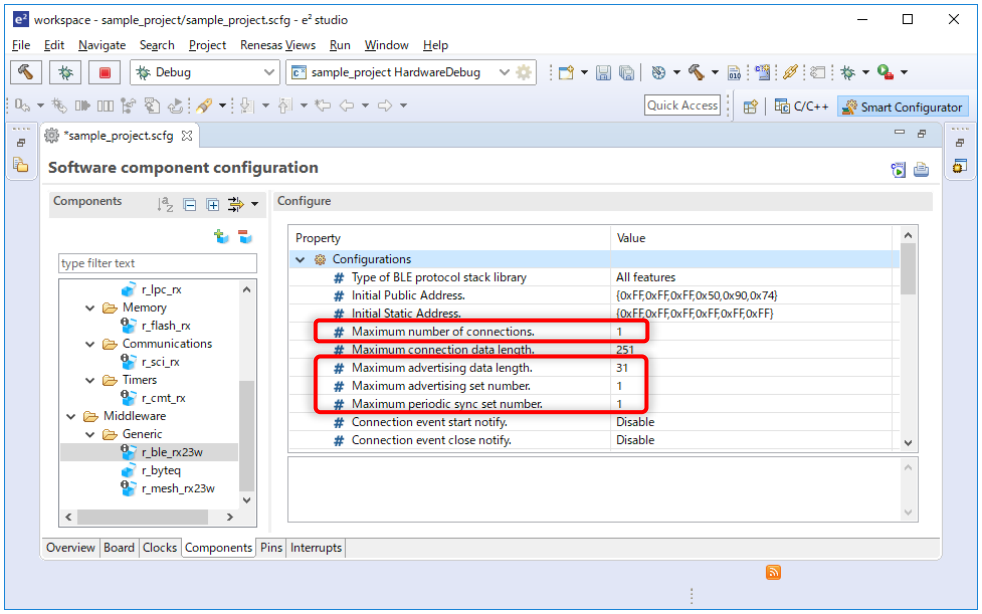

Figure 5-12 r_ble_rx23w Configuration (1)

If you use either Target Board for RX23W / Target Board for RX23W module or RSSK for RX23W, set [Enabled/Disabled command line function.] to "Enable" and set [SCI CH for command line function] to "8". Set [Enabled/Disabled board LED and Switch control support.] to "Enable" an select "Target Board" or "RSSK" as [Board Type].

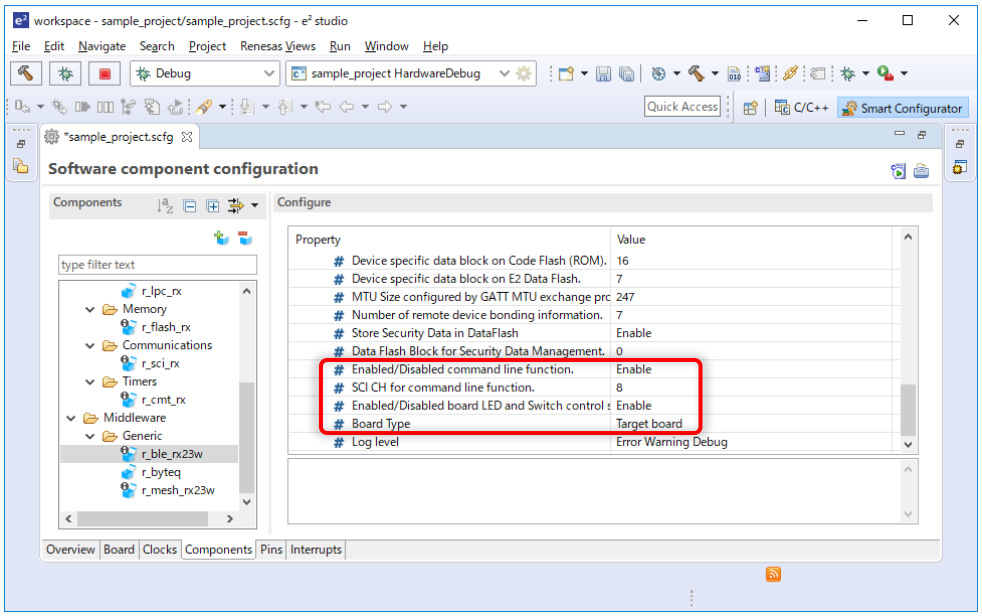

Figure 5-13 r_ble_rx23w Configuration (2)

NOTE: Either "All features" or "Balance" can be selected as [Type of BLE protocol stack library].
"Compact" cannot be selected because it does not support Scan operation.

5.4.3 r_sci_rx

If you use serial communication functionality of either Target Board for RX23W / Target Board for RX23W module or RSSK for RX23W, select [r_sci_rx] in [Components] tab, then set [Include software support for channel 8] to "Include" and set other SCI channels to "Not".

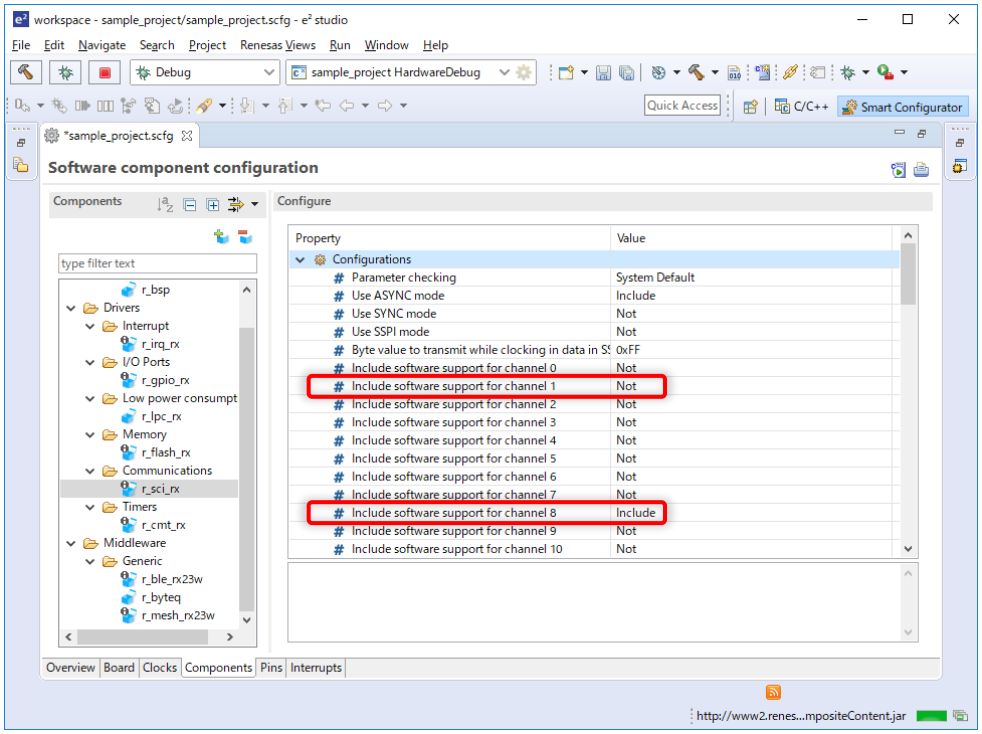

Figure 5-14 r_sci_rx Configuration (1)

Furthermore, select [Resources]→[SCI8] and set both [RXD8/SMISO8/SSCL8 Pin] and set [TXD8/SMOSI8/SSDA8 Pin] to "Used".

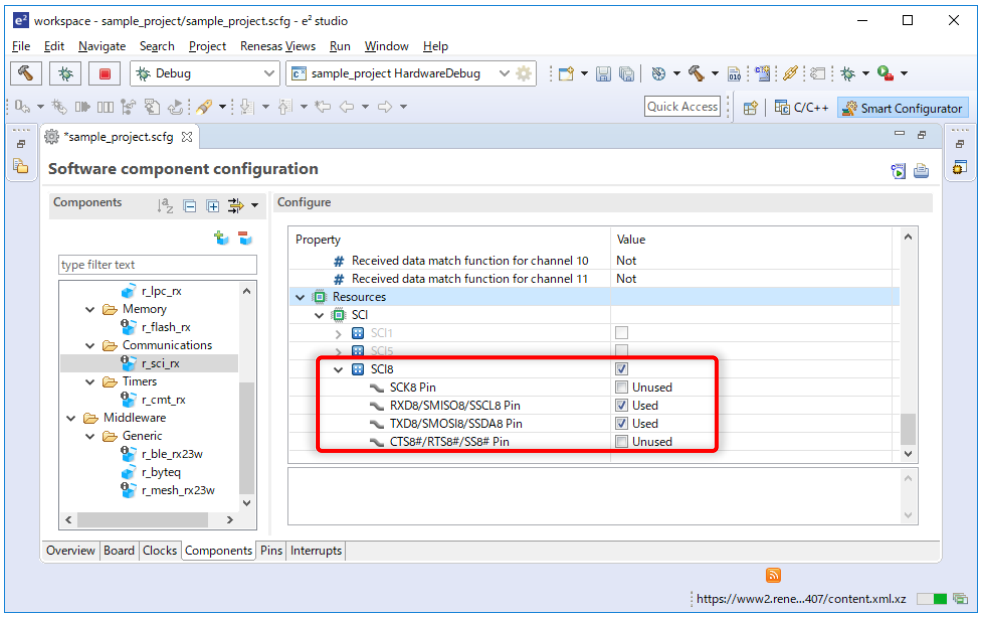

Figure 5-15 r_sci_rx Configuration (2)

BLE FIT Module provides Command Line Interface (CLI) to perform serial communication on RX23W Development Boards.

To use this functionality, set [Transmit end interrupt] to "Enable". If you use either Target Board for RX23W / Target Board for RX23W module or RSSK for RX23W, set [ASYNC mode TX queue buffer size for channel 8] to "180".

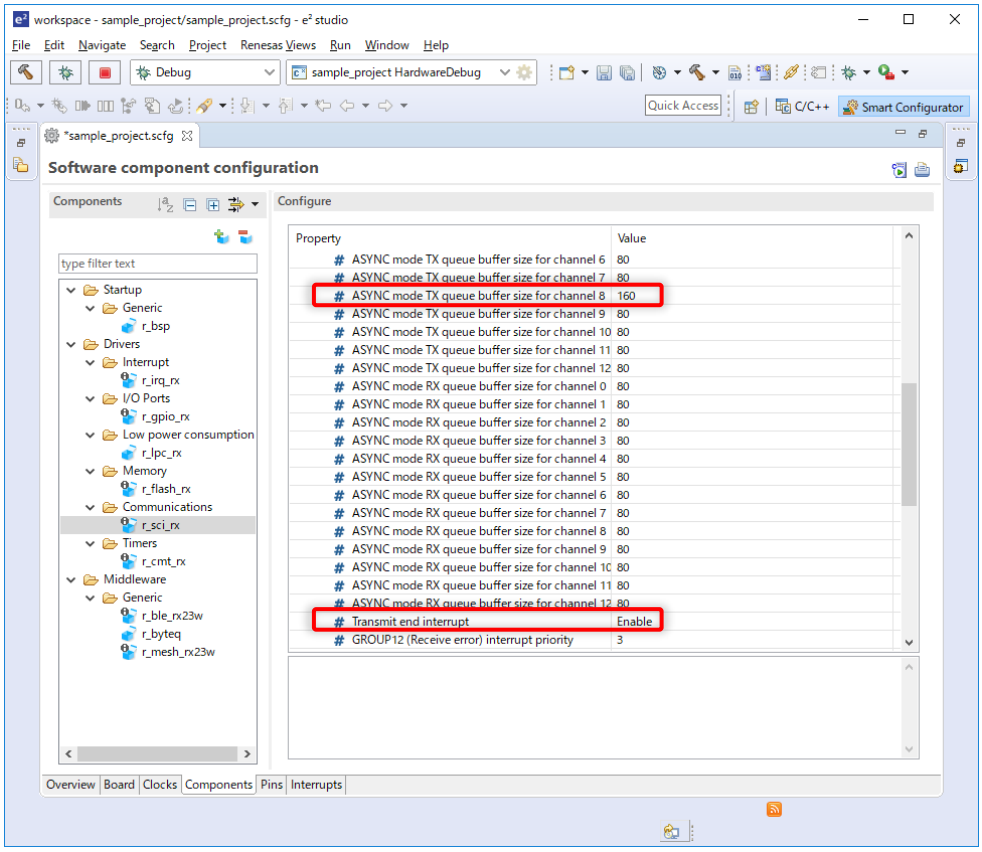

Figure 5-16 r_sci_rx Configuration (3)

5.4.4 r_irq_rx

If you use a switch on Target Board for RX23W or Target Board for RX23W module, set [IRQ5 Pin] to "Used" in [Components] tab.

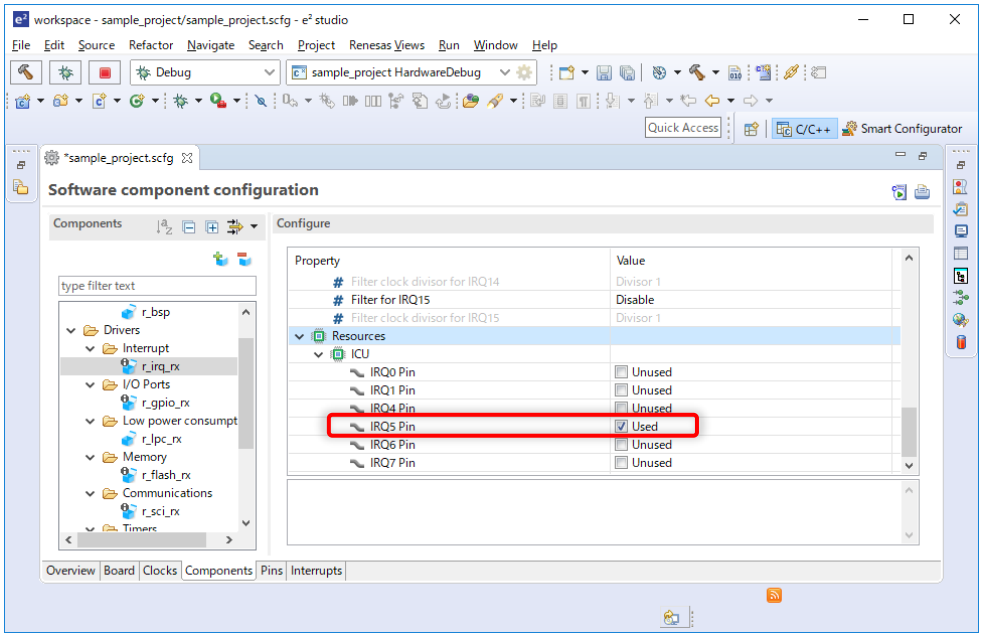

Figure 5-17 r_irq_rx Configuration (1) for Target Board

Furthermore, set [Filter for IRQ5] to "Enable" and set [Filter clock divisor for IRQ5] to "Divisor 1".

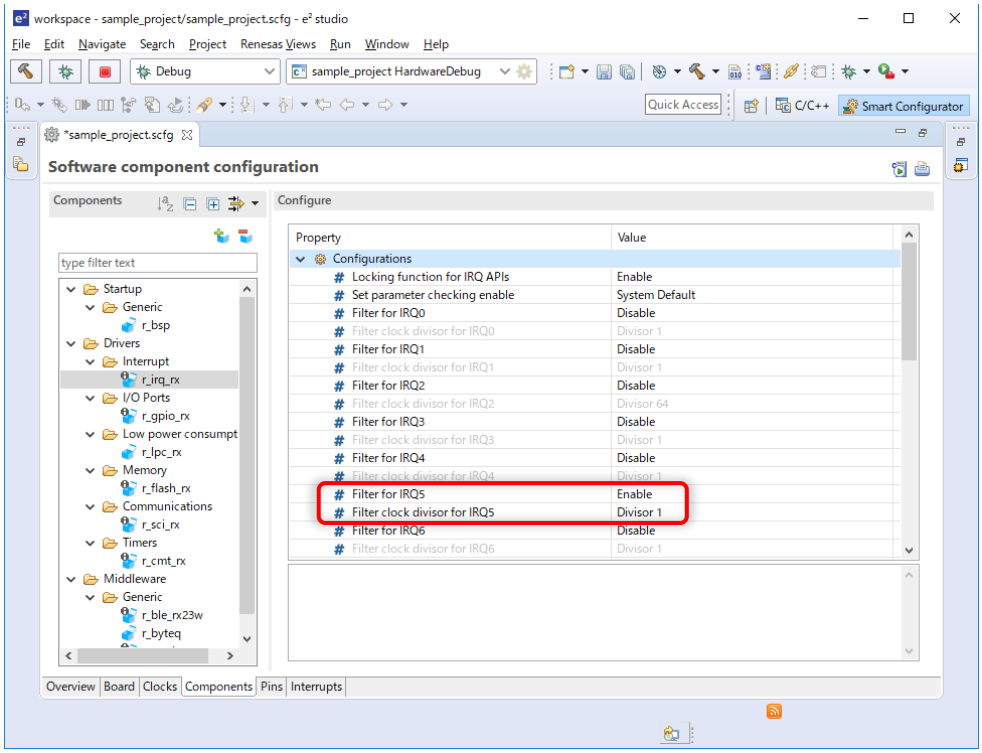

Figure 5-18 r_irq_rx Configuration (2) for Target Board

If you use switches on RSSK for RX23W, set both [IRQ0 Pin] and [IRQ1 Pin] to "Used" in [Components] tab.

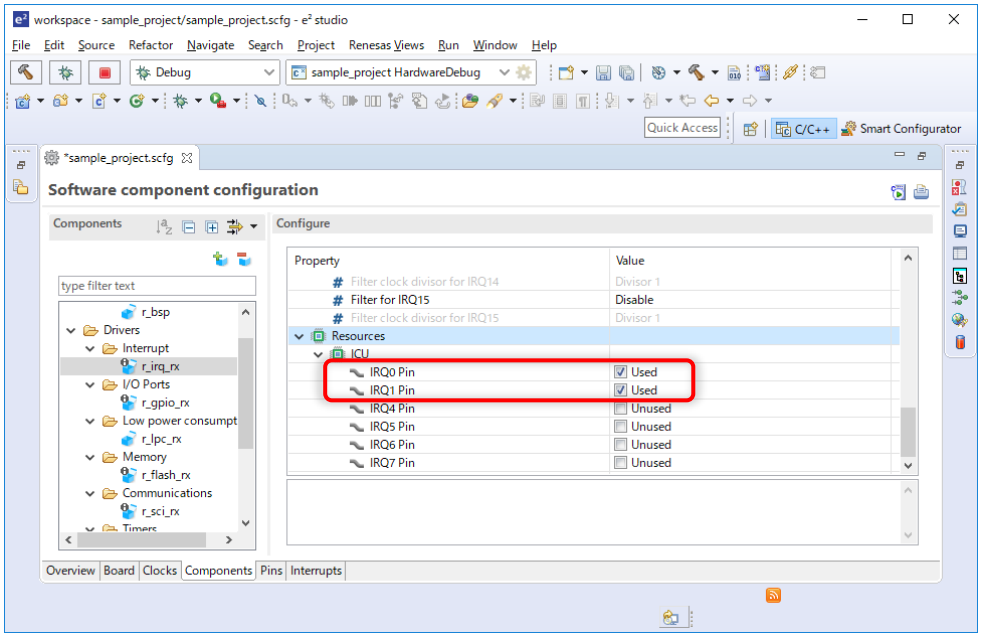

Figure 5-19 r_irq_rx Configuration (1) for RSSK

Furthermore, set both [Filter for IRQ0] and [Filter for IRQ1] to "Enable", then set [Filter clock divisor for IRQ0] and [Filter clock divisor for IRQ1] to "Divisor 1".

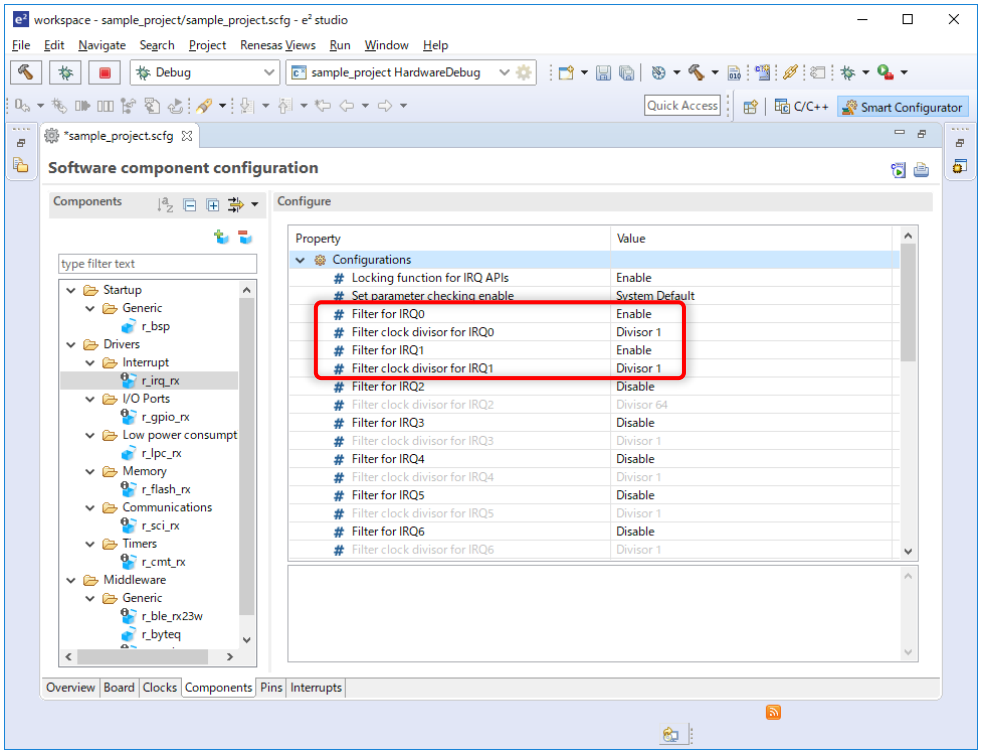

Figure 5-20 r_irq_rx Configuration (2) for RSSK

5.5 Generate Code

In [Components] tab on Smart Configurator, click [Generate] button .
Code of FIT modules are generated in "smc_gen" folder of the project.

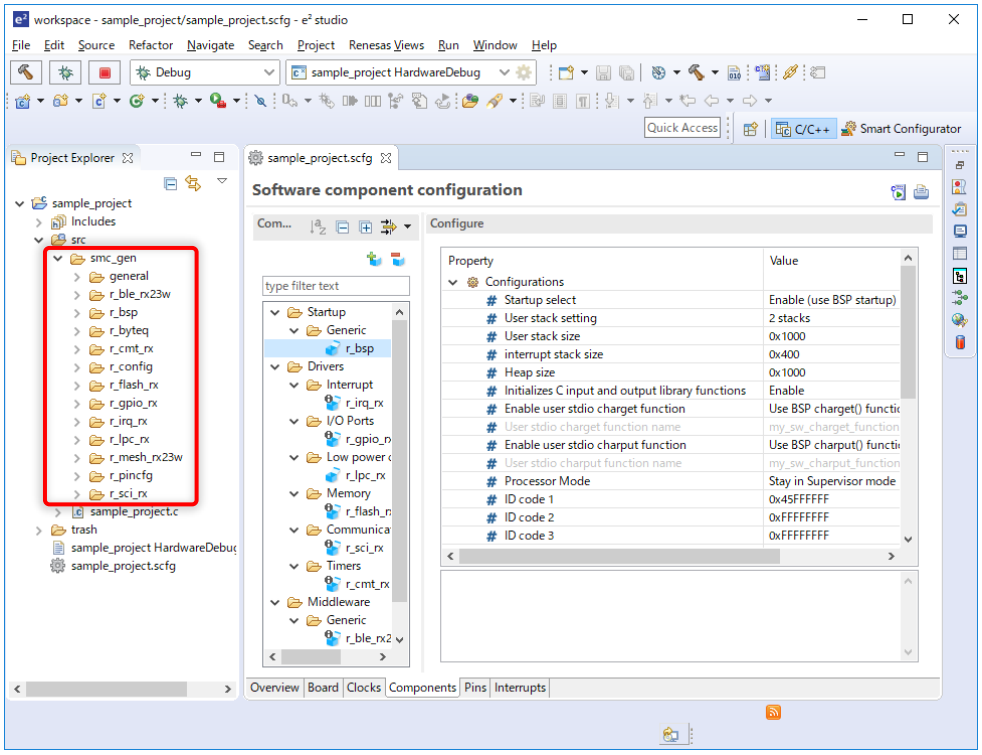

Figure 5-21 Result of Code Generation

5.6 Configure Link Options

5.6.1 Sections

Project which uses Mesh FIT Module and BLE FIT Module must add sections of each module to Link Option and set Link Option to transfer Initialization Data of ROM to Initialized Data Section of RAM.

(1) Adding Sections

Select [Properties] in [Project] menu, then select [C/C++ Build]→[Settings] in the left side of [Properties] and [Linker]→[Section] in the right side of [Tool Settings] tab. Click [...] button to show Section Viewer, then add the following sections.

[RAM]

BLE FIT Module Sections	BLE_B*, BLE_R*
Mesh FIT Module Sections	MESH_B*, MESH_R*

[ROM]

BLE FIT Module Sections	BLE_C*, BLE_D*, BLE_W*, BLE_L*, BLE_P*
Mesh FIT Module Sections	MESH_C*, MESH_D*, MESH_W*, MESH_L*, MESH_P*

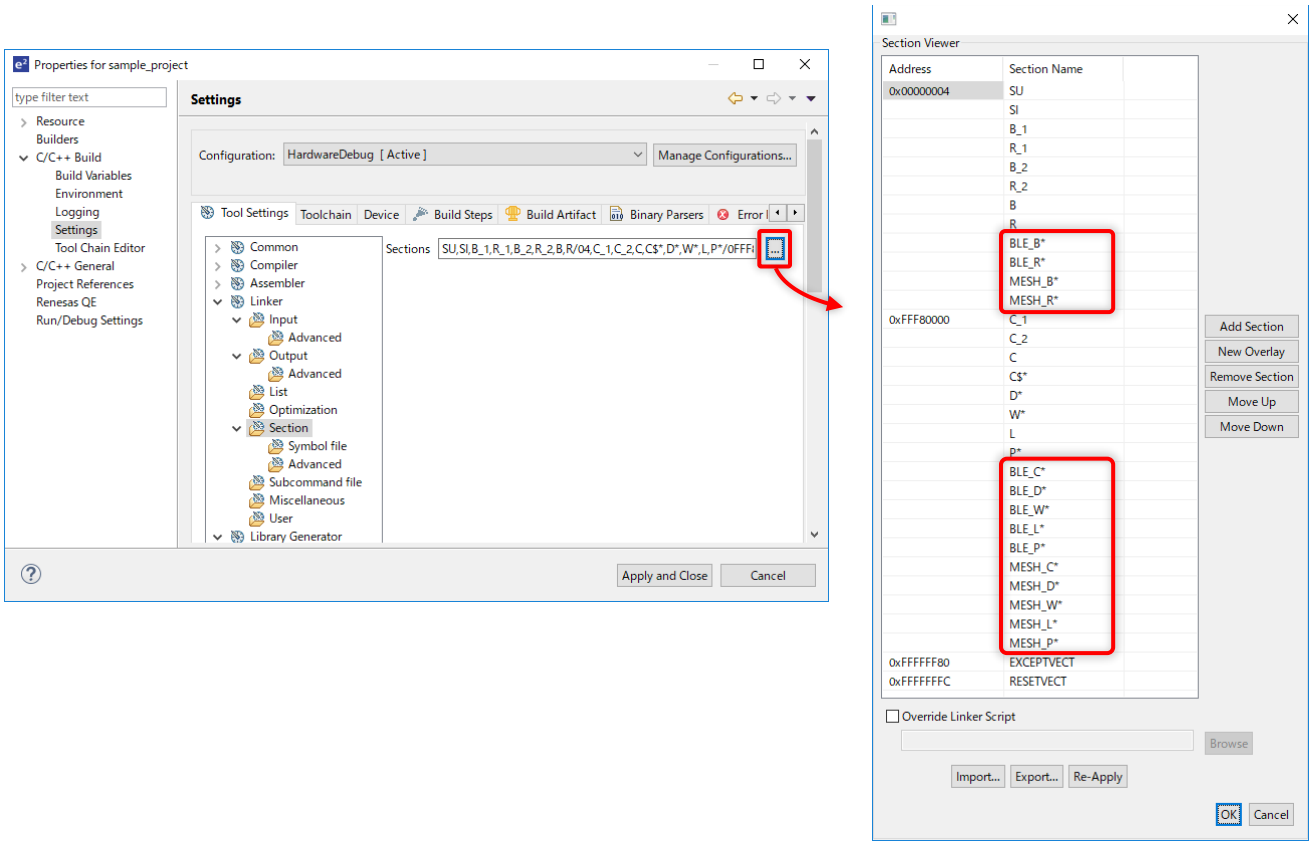

Figure 5-22 Section Configuration

(2) Mapping ROM to RAM Sections

Select [Linker]→[Symbol file] in [Tool Settings] tab of [Properties] dialog, then add the following settings to [ROM to RAM mapped section].

BLE_D=BLE_R

BLE_D_1=BLE_R_1

BLE_D_2=BLE_R_2

MESH_D=MESH_R

MESH_D_1=MESH_R_1

MESH_D_2=MESH_R_2

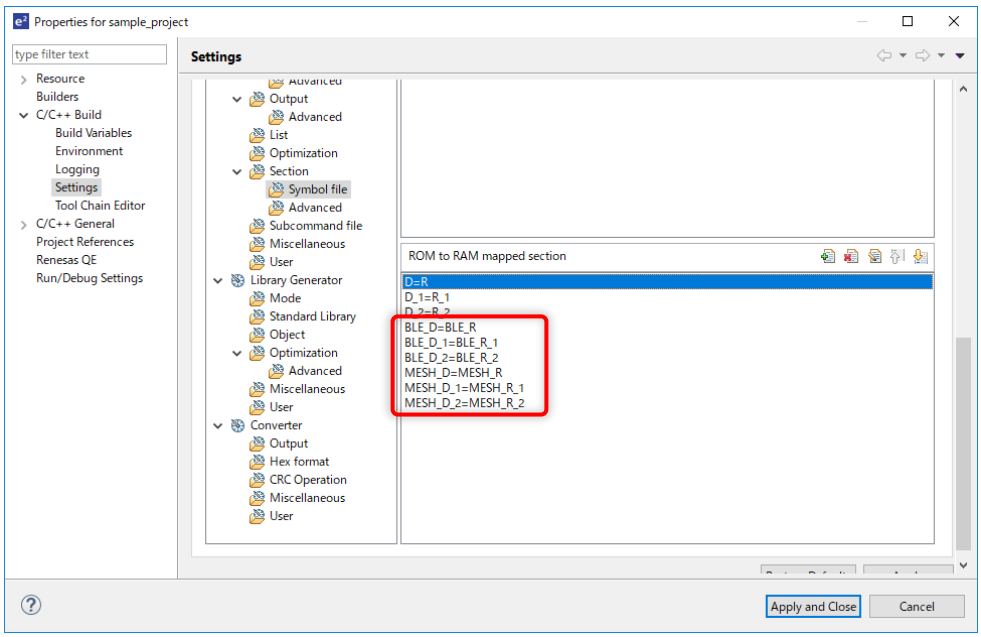

Figure 5-23 ROM to RAM mapped Section Setting

5.6.2 Libraries

Both Mesh Stack Library of Mesh FIT Module and Bluetooth Low Energy Protocol Stack Library of BLE FIT Module must be added to Link Option.

(1) Adding Libraries

Select [Linker]→[Input] in [Tool Settings] tab of [Properties] dialog. Check if the following library files are added to [Relocatable files, object files and library files].

"${workspace_loc: /${ProjName}/src/smc_gen/r_mesh_rx23w/lib/lib_ble_ms_ccrx.lib}"

"${workspace_loc: /${ProjName}/src/smc_gen/r_ble_rx23w/lib/lib_ble_ps_ccrx.lib}"

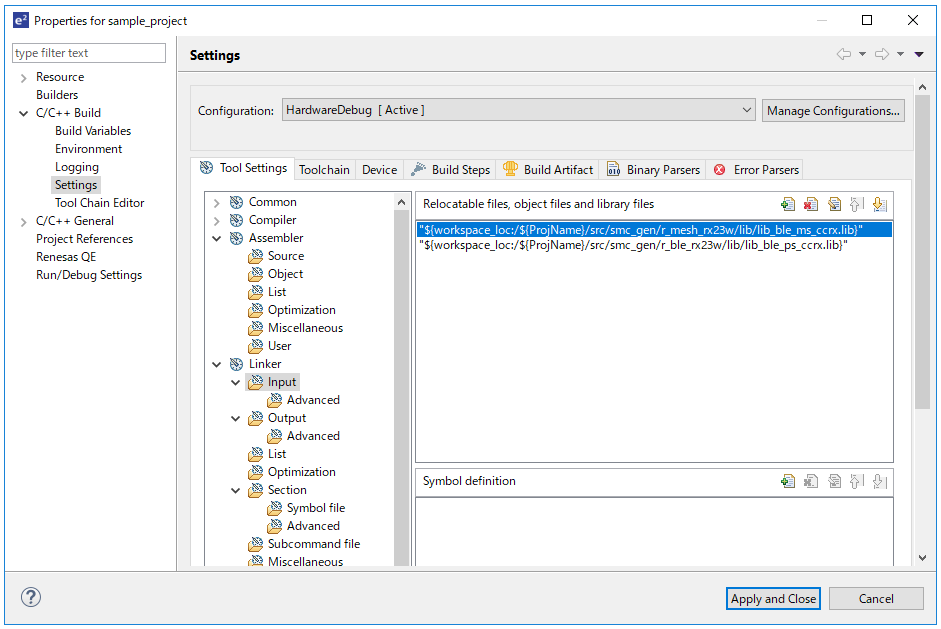

Figure 5-24 Library Files Setting

(2) Adding Prebuild Command

To use Bluetooth Low Energy Protocol Stack included in BLE FIT Module, add the following command to [Pre-build steps]→[Command] in [Build Steps] tab in [Properties] dialog.

..\src\smc_gen\r_ble_rx23w\lib\ble_fit_lib_selector.bat

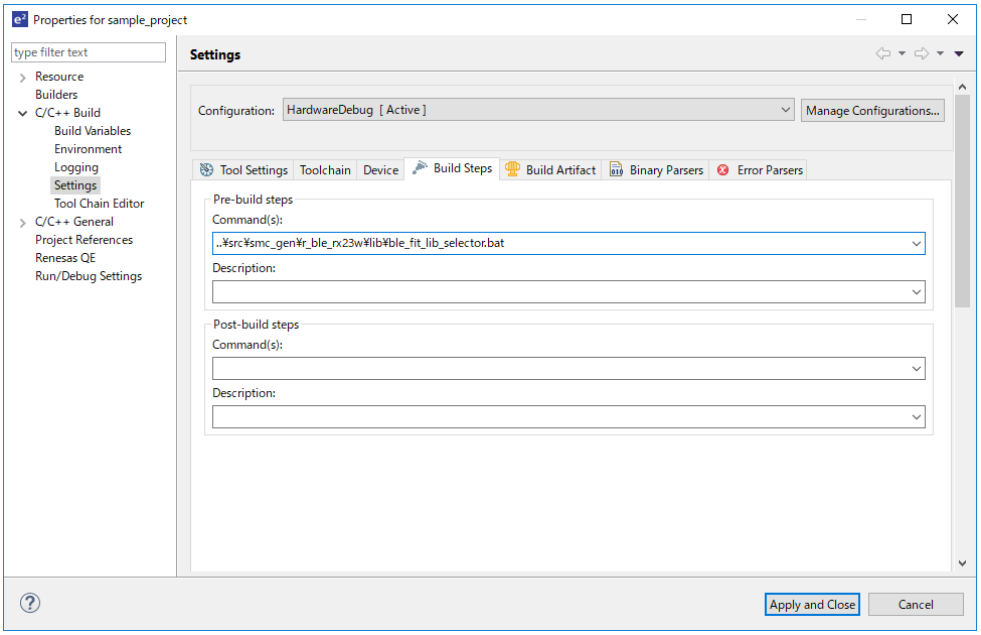

Figure 5-25 Prebuild Command Setting

After configuring the Link Options, click [Apply and Close] button in [Properties] dialog.

5.7 Configure Debug Configurations

Select [Debug Configurations] in [Run] menu. Select "{Project Name} HardwareDebug" in [Renesas GDB Hardware Debugging] and configure to debug software on RX23W.

5.7.1 Debugger Connection

In [Debug hardware], select a debugger you use. If you use Target Board for RX23W / Target Board for RX23W module or RSSK for RX23W, select "E2 Lite (RX)".

In [Target Device], select a device you use. If you use Target Board for RX23W / Target Board for RX23W module or RSSK for RX23W, select "RX"→"RX23W"→"R5F523W8".

If you use Target Board for RX23W / Target Board for RX23W module or RSSK for RX23W, set [Clock]→[Main Clock Source] to "HOCO" and set [Power]→[Power Target From The Emulator] to "No".

NOTE: In the factory setting of Target Board for RX23W, ID Code Protection of RX23W is enabled. When the message "ID code authentication failed" is displayed, set [Flash]→[ID Code] to"45FFFFFFFFFFFFFFFFFFFFFFFFFFFFFF" in [Connection Settings] tab of [Debugger] tab.

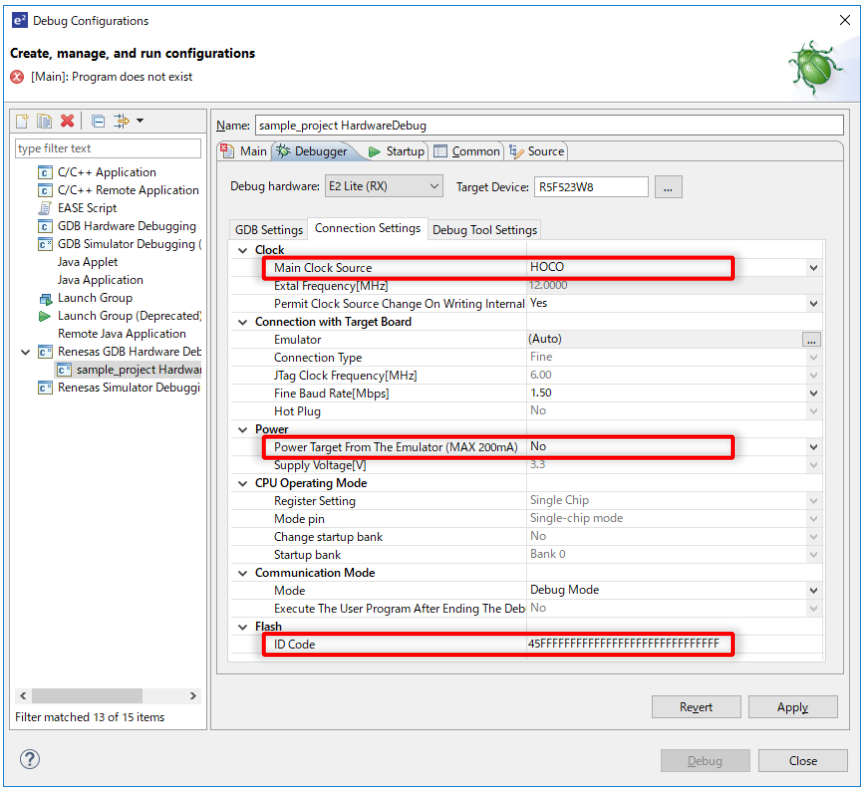

Figure 5-26 Debugger Connection Settings

5.8 Build a Project

To build a project, select [Build Project] in [Project] menu. In [Console] tab, if you can see "Build Finished" message that follows build log, building a project is successful.

For more information on debugging with e² studio, refer to Chapter 5 in "e² studio User's Manual: Getting Started Guide" (R20UT4374).

6. How to Implement Mesh Applications

Regarding how to implement mesh applications using Mesh FIT Module, refer to "RX23W Group Bluetooth Mesh Stack Development Guide" (R01AN4875).

Also, demo projects using Mesh FIT Module are included in the package of Mesh FIT Module (R01AN4930). Regarding how to run the demo projects, refer to "RX23W Group Bluetooth Mesh Stack Startup Guide" (R01AN4874).

Trademark and Copyright

RX23W Group Bluetooth Mesh Stack uses the following open source software.

- crackle; AES-CCM, AES-128bit functionality
  BSD 2-Clause License

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  * Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

  * Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Known Limitations

Bluetooth Mesh Stack	− When a node acts as Low Power Node, MCU cannot transit to Software Standby mode of low power consumption modes because Mesh Stack uses CMT (Compare Match Timer).
	− When a node acts as Sensor Setup Server, its callback function may not notify the parameters of Sensor Cadence Set / Sensor Cadence Set Unacknowledged message received. Refer to the parameters of Sensor Cadence Set / Sensor Cadence Set Unacknowledged message from the argument `msg_raw` of Sensor Setup Server's callback function.
Bluetooth Bearer	− When a node acts as Proxy Server and connects to multiple Proxy Clients, big payload size message addressed to multicast address from Proxy Server may not reach part of Proxy Clients. e.g., When a Proxy Server connects to seven Proxy Clients and transmits Vendor Set message including 95 characters, only one or two Proxy Clients can receive this message.
Mesh Sample Program	− When a node acts as Generic OnOff server or Vendor Server, STATUS message is transmitted even if Model State is not changed.

Program Updates (MESH FIT Module)

- **Rev1.01**

Bluetooth Bearer	**blebrr.c** – Added randomized delay to Advertising transmission timing.

- **Rev1.10**

Bluetooth Mesh Stack	**Common** – Added the macro below to MESH FIT Module Configuration. For details, refer to Section 3.1 MESH_CFG_NET_SEQNUM_CACHE_SIZE – Added the function below to register system time timer. MS_systemtime_init_pl() – Added the function below to enable Mesh Monitoring Functionality. MS_monitor_register_pl() **Provisioning** – Added prefix "MS_" to function names. MS_prov_set_device_oob_pubkey_pl() MS_prov_set_static_oob_auth_pl() MS_prov_clear_device_oob_pubkey_pl() MS_prov_clear_static_oob_auth_pl() MS_mempool_init_pl() MS_storage_register_pl() – Modified order of implementation of role and bearer arguments of MS_prov_setup(). **Network** – Improved processing of Network Message Cache. – Added the function below to check transmission state of Network layer. MS_net_register_tx_state_access() – Added the function below to disable 96 hours check during IV Update. MS_access_set_iv_update_test_mode() – Added processing to check Network layer and Lower Transport layer transmission state during IV Update. – Added processing to reset SEQ number after IV Update. – Added processing to resume Secure Network Beacon transmission after MCU reset. – Modified processing of MS_proxy_server_adv_stop() to stop Proxy Advertisement immediately. – Modified processing of Proxy Filter List management when Proxy connection is terminated and reestablished. **Lower Transport** – Added the function below to check transmission state of Lower Transport layer. MS_ltrn_register_tx_state_access() – Modified processing in order not to transmit Acknowledgement message for segmented message addressed to multicast address. **Access** – Added processing of checking validity of Publication Address to MS_access_cm_set_model_publication().

	− Added processing of returning some error codes to MS_access_cm_set_iv_index(). ACCESS_IV_VAL_NOT_PERMITTED ACCESS_IV_UPDATE_TOO_SOON ACCESS_IV_INCORRECT_STATE ACCESS_IV_UPDATE_DEFERRED_IN_BUSY − Modified processing of MS_access_cm_reset() to reset configuration of the following features. Relay Proxy Friend Low Power Secure Network Beacon **Model** − Changed type of the first argument of callback functions of the following Client Models to MS_ACCESS_MODEL_REQ_MSG_CONTEXT. MS_CONFIG_MODEL_CB MS_HEALTH_CLIENT_CB MS_GENERIC_ONOFF_CLIENT_CB MS_GENERIC_LEVEL_CLIENT_CB MS_GENERIC_DEFAULT_TRANSITION_TIME_CLIENT_CB MS_GENERIC_POWER_ONOFF_CLIENT_CB MS_GENERIC_POWER_LEVEL_CLIENT_CB MS_GENERIC_BATTERY_CLIENT_CB MS_GENERIC_LOCATION_CLIENT_CB MS_GENERIC_PROPERTY_CLIENT_CB MS_SENSOR_CLIENT_CB MS_TIME_CLIENT_CB MS_SCENE_CLIENT_CB MS_SCHEDULER_CLIENT_CB MS_LIGHT_LIGHTNESS_CLIENT_CB MS_LIGHT_CTL_CLIENT_CB MS_LIGHT_HSL_CLIENT_CB MS_LIGHT_XYL_CLIENT_CB MS_LIGHT_LC_CLIENT_CB − Added processing of checking registration of processing for Periodic Message Publication. − Added appl_cb argument to register callback function to MS_config_server_init().
Bluetooth Bearer	**blebrr.c/h** − Changed timer used for controlling Advertising transmission to BLE software timer (R_BLE_TIMER) from internal timer of Bluetooth Low Energy Protocol Stack. − Changed Transmission Queue Size (BLEBRR_QUEUE_SIZE) to 64 from 32. − Changed Advertising Transmission Count (BLEBRR_ADVREPEAT_COUNT) to 3 from 5. − Changed Advertising Transmission Randomized Delay (BLEBRR_ADVREPEAT_RAND_DELAY) to 10msec from 7msec. − Changed processing to manage Advertising and Scan state independently. − Fixed the issue where Transmission Queue leaks in the case that malloc() fails to allocate memory for data buffer. **blebrr_pl.c/h** − Added processing to get Static Random Address and changed default Device Address Type (BLEBRR_VS_ADDR_TYPE) to Static Random Address from Public Address. − Added processing to support simultaneous multiple connection with devices having GATT Interface. − Added processing to start Service Discovery after establishing a connection as a Mesh GATT Client and enable Notification if peer device has Mesh GATT Service. − Added the function to scan devices having Mesh GATT Service. R_MS_BRR_Scan_GattBearer() − Added prefix to function names.

	R_MS_BRR_Init() R_MS_BRR_Setup() R_MS_BRR_Register_GattIfaceCallback() R_MS_BRR_Set_GattMode() R_MS_BRR_Get_GattMode() R_MS_BRR_Disconnect() R_MS_BRR_Set_ScanRspData() R_MS_BRR_Create_Connection() R_MS_BRR_Discover_Service() R_MS_BRR_Config_Notification() − Added GATT Interface Events. BLEBRR_GATT_IFACE_NOT_FOUND BLEBRR_GATT_IFACE_SCAN − Added conn_hdl and peer_addr arguments to GATT Interface callback (BLEBRR_GATT_IFACE_CB). **gatt_db_prov.c/h, gatt_db_proxy.c/h** − Replace GATT database with one generated by QE for BLE and consisting of the following services. GAP Service GATT Service Mesh Provisioning Service or Mesh Proxy Service **gatt_clients.c** − Added processing to discover Client Characteristic Configuration Descriptor (CCCD) in Mesh GATT Service of peer device. − Added processing to check validity of Attribute Handle found from Mesh GATT Service of peer device.
Drivers	**mesh_resource.h** − Updated macro definition of Memory Pool Size (MESH_MEMPOOL_SIZE). − Added macro definition of Storage Size (MESH_STORAGE_SIZE). **mesh_dataflash.h** − Changed default configuration of compilation switch of Data Flash Enable (DATAFLASH_EN) to 1 from 0. **mesh_systemtime.c/h** − Added driver that generates 32bit system time in units of 1msec with 8-bit Timer (TMR).

■ **Rev1.20**

Bluetooth Mesh Stack	**Common** – Added the macros below to MESH FIT Module Configuration. For details, refer to Section 3.1. MESH_CFG_NET_TX_COUNT MESH_CFG_NET_TX_INTERVAL_STEPS MESH_CFG_NET_RELAY_TX_COUNT MESH_CFG_NET_RELAY_TX_INTERVAL_STEPS MESH_CFG_CONFIG_SERVER_SNB_TIMEOUT MESH_CFG_PROXY_SUBNET_NETID_ADV_TIMEOUT MESH_CFG_PROXY_SUBNET_NODEID_ADV_TIMEOUT MESH_CFG_PROXY_NODEID_ADV_TIMEOUT MESH_CFG_FRND_POLL_RETRY_COUNT MESH_CFG_LTRN_RTX_TIMEOUT MESH_CFG_LTRN_RTX_COUNT MESH_CFG_LTRN_ACK_TIMEOUT MESH_CFG_LTRN_INCOMPLETE_TIMEOUT MESH_CFG_FRND_RECEIVE_WINDOW MESH_CFG_LPN_CLEAR_RETRY_TIMEOUT_INITIAL MESH_CFG_TRN_FRNDREQ_RETRY_TIMEOUT MESH_CFG_UNPROV_DEVICE_BEACON_TIMEOUT MESH_CFG_NET_TX_QUEUE_SIZE MESH_CFG_MAX_NUM_TRANSITION_TIMERS MESH_CFG_MAX_NUM_PERIODIC_STEP_TIMERS – Added the function below to convert from Transition Time value to msec. MS_common_get_transition_time_in_ms() – Updated log structures of Mesh Monitor (MS_logger.h) **Bearer** – Improve the transmission sequence of Mesh beacon and remove MS_brr_bcast_end() **Provisioning** – Added a processing to reject invalid Public Key and invalid Confirmation Value during Provisioning (Vulnerability CVE-2020-26560) – Added the function below to get and set Public Key. MS_prov_access_local_public_key() **Network** – Modified Network Message Cache to fix the issue where message was discarded unexpectedly – Fixed the issue where Proxy Server could not transmit message to Proxy Client in a certain condition – Fixed the issue where Proxy Advertising with Node Identity transmission did not stop after the time specified by MESH_CFG_NODEID_ADV_TIMEOUT macro elapsed – Improved the implementation to transmit message in accordance with Network Transmit state and Relay Retransmit state **Transport** – Fixed the issue where Low Power Node could not reestablish a Friendship when Friend Clear Confirm was not received – Fixed the issue where Low Power Node did not resume Scan in a certain condition after Friendship was terminated – Improve transmission and reception processing of Low Power Node and Friend Node

	Model − Added the parameters to Status message transmission function of Server Models, to improve the behavior when Publish Address was set reply: flag to determine if to be transmitted to Unicast Address publish: flag to determine if to be transmitted to Publish Address MS_generic_battery_server_state_update() MS_generic_level_server_state_update() MS_generic_location_server_state_update() MS_generic_location_setup_server_state_update() MS_generic_onoff_server_state_update() MS_generic_power_level_server_state_update() MS_generic_power_level_setup_server_state_update() MS_generic_power_level_server_state_update() MS_generic_power_level_setup_server_state_update() MS_generic_power_onoff_server_state_update() MS_generic_power_onoff_setup_server_state_update() MS_generic_user_property_server_state_update() MS_generic_admin_property_server_state_update() MS_generic_manufacturer_property_server_state_update() MS_generic_client_property_server_state_update() MS_light_ctl_server_state_update() MS_light_ctl_temperature_server_state_update() MS_light_hsl_server_state_update() MS_light_hsl_hue_server_state_update() MS_light_hsl_saturation_server_state_update() MS_light_lc_server_state_update() MS_light_lightness_server_state_update() MS_light_lightness_setup_server_state_update() MS_light_xyl_server_state_update() MS_scheduler_server_state_update() MS_sensor_server_state_update() MS_sensor_setup_server_state_update() MS_time_server_state_update() − Added the functions below to change Mesh Model instance when multiple elements use the same Client Model MS_generic_battery_client_set_model_handle() MS_generic_default_transition_time_client_set_model_handle() MS_generic_level_client_set_model_handle() MS_generic_onoff_client_set_model_handle() MS_generic_power_level_client_set_model_handle() MS_generic_power_onoff_client_set_model_handle() MS_generic_property_client_set_model_handle() MS_light_ctl_client_set_model_handle() MS_light_hsl_client_set_model_handle() MS_light_lc_client_set_model_handle() MS_light_lightness_client_set_model_handle() MS_light_xyl_client_set_model_handle() MS_scene_client_set_model_handle() MS_scheduler_client_set_model_handle() MS_sensor_client_set_model_handle() MS_time_client_set_model_handle()
Bluetooth Bearer	**blebrr.c/h** − Expanded the time between BLE_GAP_EVENT_ADV_ON event and calling R_BLE_GAP_StartAdv() from 4sec to 10sec, to fix the issue where Advertising transmission was stopped in the middle (BLEBRR_ADV_TIMEOUT) − Removed the processing to repeat Advertising transmission by Bluetooth Bearer. (BLEBRR_ADVREPEAT_COUNT) − Added Advertising transmission timeout processing (BLEBRR_ADV_ABORT_TIMEOUT, blebrr_advscan_timeout_handler) − Added a function to disable ADV Bearer (blebrr_adv_disable) − Removed the processing to repeat Mesh Beacon transmission (blebrr_clear_bcon, blebrr_get_next_beacon, blebrr_bcon_send, blebrr_update_advdata)

	− Updated Advertising transmission processing (blebrr_adv_setup, blebrr_pl_advertise_setup, blebrr_advscan_timeout_handler) − Fixed the issue where ADV Bearer was not resumed in a certain timing. (blebrr_adv_sleep, blebrr_adv_wakeup) − Added an event to notify that Mesh GATT Service is found by Service Discovery (BLEBRR_GATT_IFACE_FOUND) − Added an event to notify that GATT Service Changed is received (BLEBRR_GATT_IFACE_CHANGED) − Added an event to notify that Create Connection is canceled (BLEBRR_GATT_IFACE_CANCEL) **blebrr_gatt.c** − Fixed the issue where Scan was not resumed after GATT Client terminated a connection (blebrr_pl_gatt_disconnection) **blebrr.c** − Updated the configuration to be able to transmit data continuously with More Data bit (BLEBRR_CONN_MIN_CE_LEN, BLEBRR_CONN_MAX_CE_LEN) − Added a processing to clear Scan Response data, to fix the issue where Non-connectable Advertising was not transmitted when Scan Response data had been set (blebrr_advertise_data_pl, blebrr_adv_param_set_comp_hdlr, blebrr_adv_data_upd_comp_hdlr, blebrr_gap_cb) − Added a processing to notify that GATT Service Changed is received (BLEBRR_GATT_IFACE_CHANGED) − Added an API function to cancel Create Connection (R_MS_BRR_Cancel_CreateConnection) − Added an API function to configure CCCD value of GATT Service Changed (R_MS_BRR_Config_ServChanged) − Added a processing to transmit GATT Service Changed when GATT database is changed (blebrr_set_gattmode_pl) − Added a processing to expand MTU size after a connection is established as a GATT Client (blebrr_gap_cb) **gatt_client.c** − Added a function to perform MTU Exchange (mesh_client_expand_mtu) − Added a function to configure CCCD value of GATT Service Changed (mesh_client_config_serv_changed) − Added a processing to find GATT Service Changed by Service Discovery (mesh_client_common_disc_cb) **gatt_services.c/h** − Added a processing to respond to MTU Exchange Request (mesh_serv_gatts_cb) − Added a processing to get MTU Size with R_BLE_GATT_GetMtu() (mesh_serv_get_mtu) − Added a processing to switch between Mesh Provisioning Service and Mesh Proxy Service in GATT database (mesh_serv_prov_init, mesh_serv_proxy_init, mesh_serv_prov_deinit, mesh_serv_proxy_deinit) − Added a function to transmit GATT Service Changed (mesh_indicate_serv_changed) **gatt_db.c** − Merged Mesh Provisioning Service and Mesh Proxy Service into one GATT database (gatt_db_prov.c/h, gatt_db_proxy.c/h)
Drivers	**mesh_resources.h** − Updated the definition of memory pool size macro (MESH_MEMPOOL_SIZE) − Updated the definition of storage size macro (MESH_STORAGE_SIZE) **mesh_systemtime.c/h** − Updated the calculation for the register value of 8-bit Timer (TMR)

	− Changed the default value of Systemtime Enable macro from (1) to (0) (SYSTEMTIME_EN)

■ **Rev1.30**

Bluetooth Mesh Stack	**Common** − Added the macros below to MESH FIT Module Configuration. For details, refer to Section 3.1. MESH_CFG_LPN_CLEAR_RETRY_COUNT MESH_CFG_LIGHT_LC_SERVER_MAX − Added the function below to initialize Mesh Stack and return a status code. MS_init_ext() − Fixed the issue where Data Flash driver remains Open mode after Mesh Stack initialization on the condition that there is not Provisioning information in Data Flash. − Fixed the issue where received Model message become unable to be notified to application by changing ROM section mapping. − Enhanced error checking of each Mesh Stack API. **Provisioning** − Added the function below to generate ECDH key used for Provisioning and return its Public Key. MS_prov_generate_ecdh_key_pl() **Network** − Added the function below to update Config Node Identity state when Proxy Advertisement with Node Identity stops. − Removed the processing to set Config Node Identity state to 0x02(not supported) when Proxy feature is disabled. **Lower Transport** − Added the function below to send Friend Poll message in any timing. MS_trn_lpn_poll() **Access** − Added the function below to enable any element. MS_access_register_element_ext() − Added the function below to get the number of Models registered. MS_access_get_model_count() MS_access_get_total_model_count() **Model** − Added the functions below that integrates Server Model and Setup Server Model initialization. MS_generic_location_server_init_ext() MS_generic_power_level_server_init_ext() MS_generic_power_onoff_server_init_ext() MS_light_ctl_server_init_ext() MS_light_hsl_server_init_ext() MS_light_lc_server_init_ext() MS_light_lightness_server_init_ext() MS_light_xyl_server_init_ext() MS_scheduler_server_init_ext() MS_sensor_server_init_ext() MS_time_server_init_ext() − Added the functions below that integrates Status message transmission of Server Model and Setup Server Model.

	MS_generic_location_server_state_update_ext() MS_generic_power_level_server_state_update_ext() MS_generic_power_onoff_server_state_update_ext() MS_light_ctl_server_state_update_ext() MS_light_hsl_server_state_update_ext() MS_light_lc_server_state_update_ext() MS_light_lightness_server_state_update_ext() MS_light_xyl_server_state_update_ext() MS_scheduler_server_state_update_ext() MS_sensor_server_state_update_ext() MS_time_server_state_update_ext() − Updated the Server Models so that callback function notifies the timeout of Periodic Publication. − Added the function below to finalize Health Server Models. MS_health_server_deinit() − Added the function below to set a model handle of Health Client Model used. MS_health_client_set_model_handle() − Added the function below to send Scene Status message of Scene Server Model. MS_scene_server_state_update() − Implemented a Light LC State Machine in Light LC Server Model and added the callback function to the function below to receive state transition events. MS_light_lc_server_init_ext() − Added the function below to finalize Light LC Server Models. MS_light_lc_server_deinit() − Added the functions below to set Light LC Property states of Light LC Server Model. MS_light_lc_server_set_time_property() MS_light_lc_server_set_fade_time() MS_light_lc_server_set_fade_on_time() MS_light_lc_server_set_fade_standby_auto_time() MS_light_lc_server_set_fade_standby_manual_time() MS_light_lc_server_set_occupancy_delay_time() MS_light_lc_server_set_prolong_time() MS_light_lc_server_set_run_on_time() − Added the functions below to access Light LC State Machine states of Light LC Server Model. MS_light_lc_server_set_lc_state_info() MS_light_lc_server_get_lc_state_info() MS_light_lc_server_report_occupancy() MS_light_lc_server_report_light_onoff() MS_light_lc_server_report_light_on() MS_light_lc_server_report_light_off()
Bluetooth Bearer	**blebrr.c** − Added the function below to release the resource allocated by. R_MS_BRR_Setup() R_MS_RBB_Close() − Added the processing to transmit Encoded URI that is set by MS_prov_bind().
Drivers	**mesh_resources.h** − Updated the definition of memory pool size macro (MESH_MEMPOOL_SIZE) − Updated the definition of storage size macro (MESH_STORAGE_SIZE)

Program Updates (Mesh Sample Program)

- **Rev1.01**

mesh_cli	– Added Command Line Interface (CLI) program.
mesh_model.c	– Added error check to state operation functions. mesh_model_onoff_server_state_get() mesh_model_onoff_server_state_set()

- **Rev1.10**

vendor_model	– Added Vendor Model to transmit and receive variable length data.
mesh_core.c	– Updated implementation in accordance with the specification change of Bluetooth Bearer functions. R_MS_BRR_Set_GattMode() R_MS_BRR_Get_GattMode() R_MS_BRR_Register_GattIfaceCallback() – Updated implementation in accordance with the specification change of Bluetooth Bearer callback. BLEBRR_GATT_IFACE_CB – Updated implementation to support simultaneous multiple connection with device having GATT Interface. – Added processing to terminate a connection by BLEBRR_GATT_IFACE_NOT_FOUND of GATT Interface Event. – Added processing to display log indicating devices that having Mesh GATT Service by BLEBRR_GATT_IFACE_SCAN of GATT Interface Event. – Modified processing to set Node Identity to Identification Type of Proxy Advertisement after Provisioning. – Added processing to initiate IV Update procedure when SEQ of incoming and outgoing message is over than threshold (CORE_NET_IV_UPDATE_INIT_THRESHOLD). – Added processing to establish a friendship with Friend Node as a Low Power Node and update Friend Subscription List. – Added processing to register Subscription Addresses with Proxy Filter List of peer Proxy Server after establishing a connection as a Proxy Client. – Added processing to output log of incoming and outgoing PDU and SNB from all layers of Bluetooth Mesh Stack. – Changed timing of LED blinking to the timing when connection is established from the timing when Provisioning is performed.
mesh_model.c	– Added the macro below to configure Element Location of Composition Data state. ELEMENT_DESC_LOCATION – Added argument to set TID (Transaction ID) to the function to send Generic OnOff Set message. – Added message functions and callback functions for Vendor Client model and Vendor Server model. – Updated implementation in accordance with the specification change of Bluetooth Mesh Stack API. MS_config_server_init() – Updated implementation in accordance with the specification change of Bluetooth Mesh Stack callback. MS_GENERIC_ONOFF_CLIENT_CB

main.c	− Updated implementation in accordance with the specification change of Bluetooth Bearer functions. R_MS_BRR_Init() R_MS_BRR_Setup() − Updated implementation in accordance with the specification change of Bluetooth Bearer callback type. BLEBRR_INIT_CB − Added processing to send characteristic string that is entered in console by using Vendor Client model. − Added processing to display characteristic string that is received by Vendor Server model. − Added processing to reset Node configuration when on-board switch (SW1) is pushed just after MCU reset. − Added processing to reset MCU by receiving Config Node Reset message.
mesh_appl.h	− Added compilation switches. IV_UPDATE_INITIATION_EN — IV Update procedure LOW_POWER_FEATURE_EN — Low Power feature CONSOLE_MONITOR_CFG — Mesh Monitoring functionality ANSI_CSI_EN — ANSI Escape Sequence CPU_USAGE_EN — CPU Usage Measurement − Added callbacks for Mesh Model message reception. onoff_server_set_cb — Generic OnOff Set message onoff_client_status_cb — Generic OnOff Status message vendor_server_set_cb — Vendor Set message vendor_client_status_cb — Vendor Status message config_server_node_reset_cb — Config Node Reset message − Added macros to output log to console. − Added macros to measure CPU usage.

■ **Rev1.20**

mesh_cli	− Added a processing to get Static OOB AuthValue from command parameter (root->core->provision->auth_act) − Added a processing to get OOB Public Key from command parameter (root->core->provision->dev_pkey) − Added a command to change Unicast Address to be configured during Provisioning (root->core->provision->next_addr) − Updated Log Output of Mesh Monitor (root->pkt_log) − Added Log Output for BLEBRR_GATT_IFACE_FOUND event, BLEBRR_GATT_IFACE_CHANGED event, and BLEBRR_GATT_IFACE_CANCEL event (cli_gatt_bearer_iface_event_pl_cb) − Added a command to set CCCD value of GATT Service Changed (root->brr->config_servchg) − Added a command to cancel Create Connection (root->brr->cancel) − Added a processing to get Local Name to be set to Scan Response Data from command parameter (root->brr->srsp_set) − Updated implementation of Server Model callback functions in accordance with specification change of STATUS message transmission function (appl_model_server_callback.c) − Added processing to include Generic OnOff Server Model and Generic Power OnOff Server Model to the initialization function of Light LC Server Model (main_light_lc_server_operations) − Change the setting of Provisioning Capabilities to enable OOB Public Key Exchange and OOB Authentication (appl_prov_capab)

	− Added a processing to terminate a connection when Provisioning completes over GATT Bearer (appl_prov_callback) − Improved the implementation to be able to work as Provisioner after Provisioning procedure completes (appl_prov_callback) − Added a test command to change the behavior of Network Message Cache (root->core->network->cache_wrap)
main.c	− Defined Health Model Test IDs e_mesh_health_test_id_t − Added a processing to set Scan Response data to Bluetooth Bearer. blebrr_init_cb() − Added a processing to transmit Proxy Advertisement with Node Identity by long press of a switch. sw_long_press_timer_cb()
mesh_appl.h	− Added a function macro to dump memory (CONSOLE_DUMP_BYTES) − Added a function macro to generate NetKey (NETKEY_GEN) − Added a function macro to generate AppKey (APPKEY_GEN)
mesh_core.c	− Added a processing to store connection handles of connected devices. mesh_core_gatt_iface_cb() − Updated the Proxy Advertisement transmission processing. mesh_core_prov_setup() mesh_core_prov_bind() mesh_core_proxy_setup() mesh_core_proxy_start() − Added a processing to terminate connections with connected devices. mesh_core_proxy_disconnect() − Added a processing to transmit Secure Network Beacon for all subnets when Proxy connection is established. mesh_core_proxy_cb() − Added a function to add addresses to Friend Subscription List. mesh_core_lpn_add_addresses() − Updated IV Update Initiation processing. mesh_core_monitor_net_pdu() − Updated Log Output processing of Mesh Monitor. mesh_core_monitor_access_pdu() mesh_core_monitor_trans_pdu() mesh_core_monitor_ltrans_pdu() mesh_core_monitor_net_pdu() mesh_core_monitor_generic_log)
mesh_model.c	− Updated the processing in accordance with the specification change of MS_generic_onoff_server_state_update(). mesh_model_onoff_server_cb() − Updated the processing in accordance with the specification change of MS_vendor_server_state_update().

	mesh_model_vendor_server_cb() – Added a processing to terminate all connections of GATT Bearer when Friend Feature is disabled by Configuration Model. mesh_model_config_server_cb() – Added a processing to add Subscription Addresses, added by Configuration Model, to Friend Subscription List. mesh_model_config_server_cb() – Added a function to transmit Health Fault Status message. mesh_model_health_server_fault_status() – Added a function to resume Periodic Publishing after MCU reset. mesh_model_trigger_publishing()
vendor_server.c	– Added arguments of "reply" and "publish" to MS_vendor_server_state_update() – Supported Periodic Publishing. vendor_server_publish_timout_cb()

■ **Rev1.30**

FITDemos	– Added the demo projects for Target Board for RX23W module. tbrx23wmodule_mesh_cli tbrx23wmodule_mesh_client tbrx23wmodule_mesh_server
mesh_mobile	– Updated the MeshMobile project.
rsskrx23w_mesh_cli **tbrx23w_mesh_cli**	– Added the "frndpoll" command to send Friend Poll message. – Added the "scene_update" command to send Scene Status message. – Added the "lcprop_set" command to set LC Time Properties collectively. – Supported Mesh Stack API updated, updated comments, etc.
rsskrx23w_mesh_client **rsskrx23w_mesh_server** **tbrx23w_mesh_client** **tbrx23w_mesh_server**	– Changed to use a device-specific Device UUID generated from UIDR register. – Supported Provisioning Procedure with OOB Public Key and/or Static/Output/Input Authentication. – Removed the processing to terminate all GATT connection when Proxy feature is disabled. – Changed to restart Provisioning without MCU resetting after receiving Config Node Reset message. – Supported Mesh Stack API updated, updated console log, updated comments, etc.

Revision History

Rev.	Date	Description	
1.00	Oct. 11, 2019	-	First edition
1.01	Nov. 29, 2019	P.8	Updated the program size in Table 2 and Table 3
		P.10	Removed BSP_CFG_ID_CODE_LONG_1 from Table 5
		P.18	Deleted ID code setting in Subsection 5.4.1
		P.31	Added the note about ID node setting to Subsection 5.8.1
1.10	Sep. 30, 2020	P.6	Added 8-Bit Timer as Hardware Requirement
		P.6	Changed r_cmt_rx version in Software Requirement to 4.5 or later
		P.8	Updated the program size in Table 2 and Table 3
		P.9	Added MESH_CFG_NET_SEQNUM_CACHE_SIZE macro to Table 4
		P.25	Removed "r_cmt_rx" Subsection from Section 5.6
		P.25	Added Subsection 5.6.1 "r_ble_rx23w"
1.20	Sep. 30, 2021	P.6	Updated the file composition in Section 1.3 "File Composition"
		P.7	Updated the FIT module version in Section 2.2 "Software Requirements"
		P.9	Updated the program size in Table 2 2 and Table 2 3
		P.10	Added the Configuration Macros to Table 3 1
		P.31	Section 5.6 "Change Code"
1.30	Dec. 22, 2022	P.6	Added "json" folder in Section 1.3 "File Composition"
		P.7	Updated the e^2 studio version and added Target Board for RX23W module in 2.3 "Supported Toolchain"
		P.9	Updated the program size in Table 2-2 and Table 2-3
		P.11	Changed MESH_CFG_PROXY_SUBNET_NODEID_ADV_TIMEOUT macro's default values and MESH_CFG_PROXY_SUBNET_NETID_ADV_TIMEOUT macro's default values from 100 to 300 in Table 3-1
		P.12 to P.13	Added MESH_CFG_LPN_CLEAR_RETRY_COUNT macro and MESH_CFG_LIGHT_LC_SERVER_MAX macro in Table 3-1
		P.14	Added the setting for Target Board for RX23W module in Table 3-3
		P.16 to P.33	Added the setting for Target Board for RX23W module in Sections 5.1, 5.4, and 5.7

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

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

1. Precaution against Electrostatic Discharge (ESD)

   A strong electrical field, when exposed to a CMOS device, can cause destruction of the gate oxide and ultimately degrade the device operation. Steps must be taken to stop the generation of static electricity as much as possible, and quickly dissipate it when it occurs. Environmental control must be adequate. When it is dry, a humidifier should be used. This is recommended to avoid using insulators that can easily build up static electricity. Semiconductor devices must be stored and transported in an anti-static container, static shielding bag or conductive material. All test and measurement tools including work benches and floors must be grounded. The operator must also be grounded using a wrist strap. Semiconductor devices must not be touched with bare hands. Similar precautions must be taken for printed circuit boards with mounted semiconductor devices.
2. Processing at power-on

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

   Do not input signals or an I/O pull-up power supply while the device is powered off. The current injection that results from input of such a signal or I/O pull-up power supply may cause malfunction and the abnormal current that passes in the device at this time may cause degradation of internal elements. Follow the guideline for input signal during power-off state as described in your product documentation.
4. Handling of unused pins

   Handle unused pins in accordance with the directions given under handling of unused pins in the manual. The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of the LSI, an associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible.
5. Clock signals

   After applying a reset, only release the reset line after the operating clock signal becomes stable. When switching the clock signal during program execution, wait until the target clock signal is stabilized. When the clock signal is generated with an external resonator or from an external oscillator during a reset, ensure that the reset line is only released after full stabilization of the clock signal. Additionally, when switching to a clock signal produced with an external resonator or by an external oscillator while program execution is in progress, wait until the target clock signal is stable.
6. Voltage application waveform at input pin

   Waveform distortion due to input noise or a reflected wave may cause malfunction. If the input of the CMOS device stays in the area between V_{IL} (Max.) and V_{IH} (Min.) due to noise, for example, the device may malfunction. Take care to prevent chattering noise from entering the device when the input level is fixed, and also in the transition period when the input level passes through the area between V_{IL} (Max.) and V_{IH} (Min.).
7. Prohibition of access to reserved addresses

   Access to reserved addresses is prohibited. The reserved addresses are provided for possible future expansion of functions. Do not access these addresses as the correct operation of the LSI is not guaranteed.
8. Differences between products

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

Notice

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 or any other use of the circuits, software, and information in the design of your product or system. Renesas Electronics disclaims any and all liability for any losses and damages incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics hereby expressly disclaims any warranties against and liability for infringement or any other claims involving 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, including but not limited to, the product data, drawings, charts, programs, algorithms, and application examples.
3. 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 shall be responsible for determining what licenses are required from any third parties, and obtaining such licenses for the lawful import, export, manufacture, sales, utilization, distribution or other disposal of any products incorporating Renesas Electronics products, if required.
5. You shall not alter, modify, copy, or reverse engineer any Renesas Electronics product, whether in whole or in part. Renesas Electronics disclaims any and all liability for any losses or damages incurred by you or third parties arising from such alteration, modification, copying or reverse engineering.
6. Renesas Electronics products are classified according to the following two quality grades: “Standard” and “High Quality”. The intended 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; industrial robots; etc.
   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control (traffic lights); large-scale communication equipment; key financial terminal systems; safety control equipment; etc.
   Unless expressly designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not intended or 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 damage (space system; undersea repeaters; nuclear power control systems; aircraft control systems; key plant systems; military equipment; etc.). Renesas Electronics disclaims any and all liability for any damages or losses incurred by you or any third parties arising from the use of any Renesas Electronics product that is inconsistent with any Renesas Electronics data sheet, user’s manual or other Renesas Electronics document.
7. No semiconductor product is absolutely secure. Notwithstanding any security measures or features that may be implemented in Renesas Electronics hardware or software products, Renesas Electronics shall have absolutely no liability arising out of any vulnerability or security breach, including but not limited to any unauthorized access to or use of a Renesas Electronics product or a system that uses a Renesas Electronics product. RENESAS ELECTRONICS DOES NOT WARRANT OR GUARANTEE THAT RENESAS ELECTRONICS PRODUCTS, OR ANY SYSTEMS CREATED USING RENESAS ELECTRONICS PRODUCTS WILL BE INVULNERABLE OR FREE FROM CORRUPTION, ATTACK, VIRUSES, INTERFERENCE, HACKING, DATA LOSS OR THEFT, OR OTHER SECURITY INTRUSION (“Vulnerability Issues”). RENESAS ELECTRONICS DISCLAIMS ANY AND ALL RESPONSIBILITY OR LIABILITY ARISING FROM OR RELATED TO ANY VULNERABILITY ISSUES. FURTHERMORE, TO THE EXTENT PERMITTED BY APPLICABLE LAW, RENESAS ELECTRONICS DISCLAIMS ANY AND ALL WARRANTIES, EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT AND ANY RELATED OR ACCOMPANYING SOFTWARE OR HARDWARE, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE.
8. When using Renesas Electronics products, refer to the latest product information (data sheets, user’s manuals, application notes, “General Notes for Handling and Using Semiconductor Devices” in the reliability handbook, etc.), and ensure that usage conditions are within the ranges specified by Renesas Electronics with respect to maximum ratings, operating power supply voltage range, heat dissipation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions, failure or accident arising out of the use of Renesas Electronics products outside of such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of Renesas Electronics products, semiconductor products have specific characteristics, such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Unless designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not subject to radiation resistance design. You are responsible for implementing safety measures to guard against the possibility of bodily injury, injury or damage caused by fire, and/or danger to the public in the event of a failure or malfunction of Renesas Electronics products, 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 and impractical, you are responsible for evaluating the safety of the final products or systems manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. You are responsible for carefully and sufficiently investigating applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive, and using Renesas Electronics products in compliance with all these applicable laws and regulations. Renesas Electronics disclaims any and all liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. Renesas Electronics products and technologies shall 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 shall comply with any applicable export control laws and regulations promulgated and administered by the governments of any countries asserting jurisdiction over the parties or transactions.
12. It is the responsibility of the buyer or distributor of Renesas Electronics products, or any other party who distributes, disposes of, or otherwise sells or transfers the product to a third party, to notify such third party in advance of the contents and conditions set forth in this document.
13. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
14. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products.

(Note1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its directly or indirectly controlled subsidiaries.
(Note2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.

(Rev.5.0-1 October 2020)

Corporate Headquarters

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

Trademarks

Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners.

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/.