

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 ([Core Specifications](#))
- Bluetooth Mesh Networking Specifications ([Mesh Networking Specifications](#))
- CC-RX Compiler User's Manual ([R20UT3248](#))
- e² studio User's Manual: Getting Started Guide ([R20UT4374](#))
- RX Smart Configurator User Guide: e² 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	3
1.1 Features	3
1.2 Software Architecture	4
1.3 File Composition	5
1.4 API Specification	5
1.5 API Header File	5
2. Requirements	6
2.1 Hardware Requirements	6
2.2 Software Requirements	6
2.3 Supported Toolchain	6
2.4 Sections	7
2.5 Program Size	8
3. FIT Module Configurations	9
3.1 Mesh FIT Module	9
3.2 BSP FIT Module	10
3.3 BLE FIT Module	10
4. How to add FIT Modules	11
5. Usage	12
5.1 Create a New Project	12
5.2 Configure Clocks	15
5.3 Add Components	16
5.4 Component Configurations	18
5.4.1 r_bsp	18
5.4.2 r_ble_rx23w	19
5.4.3 r_sci_rx	20
5.4.4 r_irq_rx	22
5.5 Generate Code	24
5.6 Change Code	25
5.6.1 r_ble_rx23w	25
5.7 Configure Link Options	27
5.7.1 Sections	27
5.7.2 Libraries	29
5.8 Configure Debug Configurations	31
5.8.1 Debugger Connection	31
5.9 Build a Project	31
6. How to Implement Mesh Applications	32

1. Overview

1.1 Features

Mesh Fit Module provides many-to-many wireless communication features which are compliant with Bluetooth Mesh Networking Specifications. 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)
- 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 show the software architecture to use Mesh FIT Module.

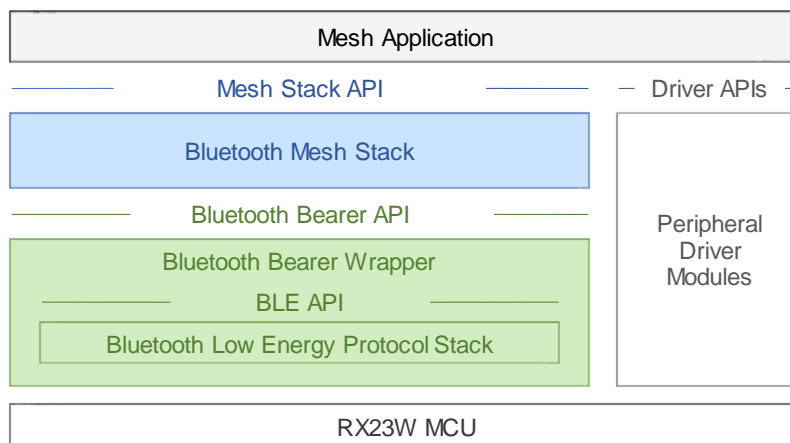


Figure 1 Software Architecture

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 Wrapper**

Bluetooth Bearer Wrapper is the abstraction layer that provides wrapper functions of Bluetooth Protocol Stack.

- **Bluetooth Protocol Stack**

Bluetooth 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 Wrapper are included in Mesh FIT Module ([R01AN4930](#)).

Bluetooth 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\	
r01an4930ej0110-rx23w-blemesh.pdf	Mesh FIT Module Application Note (en)
+---ja\	
r01an4930jj0110-rx23w-blemesh.pdf	Mesh FIT Module Application Note (ja)
+---lib\	Mesh Stack Library
+---ref\	Mesh Stack Default Configuration
+---src\	
+---bearer\	Bluetooth Bearer Wrapper
+---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)
- **r_ble_rx23w**: Bluetooth Low Energy (BLE FIT Module)
- **r_flash_rx**: Data Flash memory (Flash FIT Module)

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 4.50 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 e² studio V7.8.0
- **Compiler**: Renesas Electronics C/C++ Compiler for RX Family (CC-RX) V2.08.00
- **Endian**: Little Endian
- **Board**: Target Board for RX23W (RTK5RX23W0C00000BJ)
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 1.

Table 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 5.7.1.

2.5 Program Size

Table 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 Total Program Size of Mesh FIT Module

Device	Compiler	Category	Size
RX23W Group	CC-RX V2.08.00	ROM	62,667byte
		RAM	9,165byte
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 (-nootimize)		

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

Device	Compiler	Category	Size
RX23W Group	CC-RX V2.08.00	ROM	297,919byte (Mesh FIT Module 58,672byte)
		RAM	43,757byte (Mesh FIT Module 9,165byte)
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 4.

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 4 Configuration Macros of Mesh FIT Module

Configuration Macro	Value Range	Default Value
MESH_CFG_DEFAULT_COMPANY_ID Company ID registered with Bluetooth SIG	0x0000 to 0xFFFFE	0x0036
MESH_CFG_DEFAULT_PID Product ID assigned by vendor	0x0000 to 0xFFFF	0x0001
MESH_CFG_DEFAULT_VID Product Version ID assigned by vendor	0x0000 to 0xFFFF	0x0100
MESH_CFG_ACCESS_ELEMENT_COUNT Maximum number of Elements, 1 or over	1 to 65,535	4
MESH_CFG_ACCESS_MODEL_COUNT Maximum number of Models, 1 or over	1 to 65,535	20
MESH_CFG_HEALTH_SERVER_MAX Maximum number of Health Servers, 1 or over	1 to 65,535	2
MESH_CFG_MAX_SUBNETS Maximum number of subnets, 1 or over	1 to 65,535	4
MESH_CFG_MAX_APPS Maximum number of Application Keys that device stores, 1 or over	1 to 65,535	8
MESH_CFG_MAX_DEV_KEYS Maximum number of Device Key, 1 or over	1 to 65,535	4
MESH_CFG_MAX_VIRTUAL_ADDRS Maximum number of Virtual Address that device stores, 1 or over	1 to 65,535	8
MESH_CFG_MAX_NON_VIRTUAL_ADDRS Maximum number of Non-virtual Address (Unicast Address or Group Address) that device stores, 1 or over	1 to 65,535	8
MESH_CFG_NET_SEQ_NUMBER_BLOCK_SIZE Distance between Network Sequence Numbers for writing to Data Flash memory, 1 or over	1 to 65,535	2048
MESH_CFG_MAX_LPNS Maximum number of peer Low Power Nodes (LPN) with which device establishes friendship as a Friend Node, 1 or over	1 to 65,535	1
MESH_CFG_FRIEND_SUBSCRIPTION_LIST_SIZE Maximum number of Subscription Lists for each LPN, 1 or over	1 to 65,535	8
MESH_CFG_FRIEND_MESSAGEQUEUE_SIZE Size of Message Queues for each LPN, 2 or over	2 to 65,535	15
MESH_CFG_PROXY_FILTER_LIST_SIZE Maximum number of addresses that can be added to each Proxy List, 1 or over	1 to 65,535	2

MESH_CFG_NET_CACHE_SIZE The number of nodes of Network Message Cache, 2 or over	2 to 65,535	10
MESH_CFG_NET_SEQNUM_CACHE_SIZE Size of Network Message Cache per node	32 to 65535	32
MESH_CFG_REPLAY_CACHE_SIZE Size of Replay Protection Cache, 2 or over	2 to 65,535	10
MESH_CFG_LTRN_SAR_CTX_MAX Size of SAR (Segmentation and reassembly) Contexts, 2 or over	2 to 65,535	8
MESH_CFG_REASSEMBLED_CACHE_SIZE Size of SAR(Segmentation and reassembly) Reception Cache, 2 or over	2 to 65,535	8
MESH_CFG_NUM_NETWORK_INTERFACES The number of bearers used for Mesh Network, 1 or over	1 to 65,535	2
MESH_CFG_NUM_PROVISIONING_INTERFACES The number of bearers used for Provisioning, 1 or over	1 to 65,535	2

3.2 BSP FIT Module

The configuration macros listed in Table 5 of BSP FIT Module must be changed to use Mesh FIT Module. Regarding how to change by using Smart Configurator, refer to Subsection 5.4.1.

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

Table 5 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 6 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 6 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 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 e² studio to add FIT modules.
You can add FIT modules to your project automatically by using **Smart Configurator** on e² studio. For more details, refer to "RX Smart Configurator User Guide: e² 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: e² studio" ([R20AN0451](#)).
- (3) If you use **FIT Configurator** on e² studio to add FIT modules.
You can add FIT modules to your project automatically by using **FIT Configurator** on e² 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.

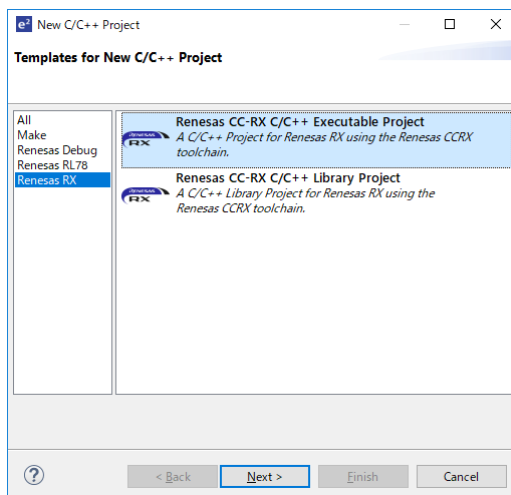


Figure 2 Project Template Selection

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

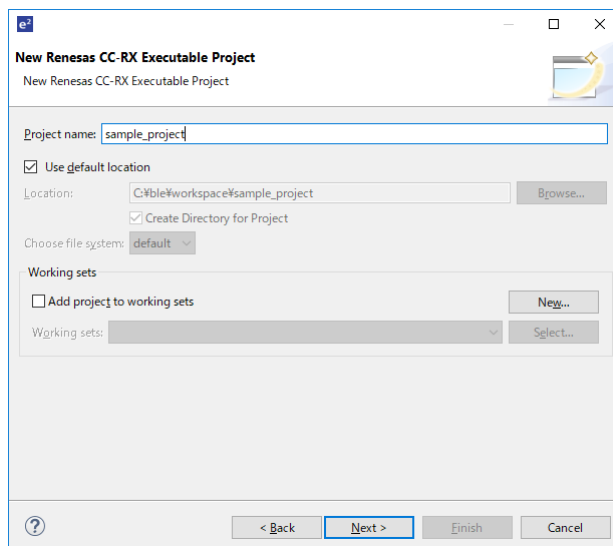


Figure 3 New Project Settings

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 RSSK for RX23W, select "R5F523W8AxBL" when part number of the RSSK is "RTK5523W8AC00001BJ", or select "R5F523W8BxBL" when part number of the RSSK is "RTK5523W8BC00001BJ".

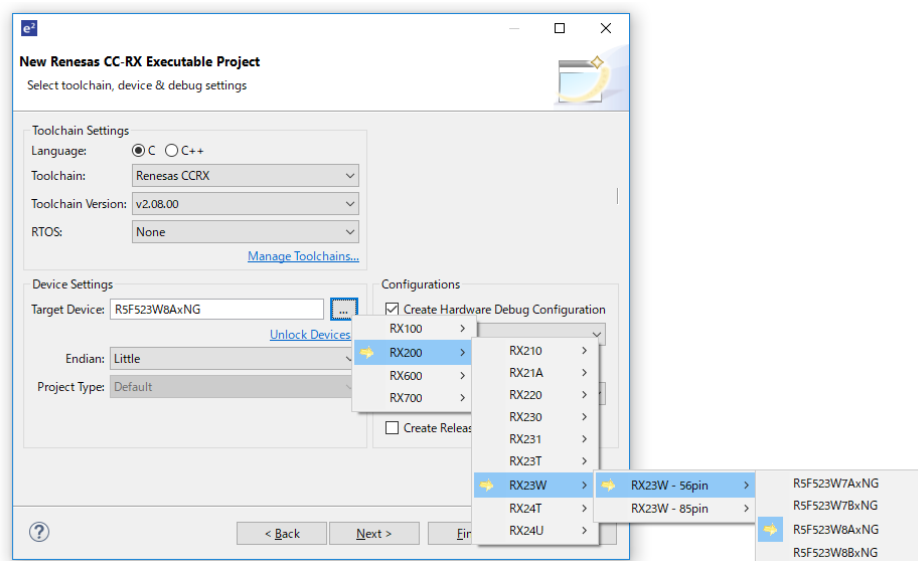


Figure 4 Toolchain, Device, and Debug Settings

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

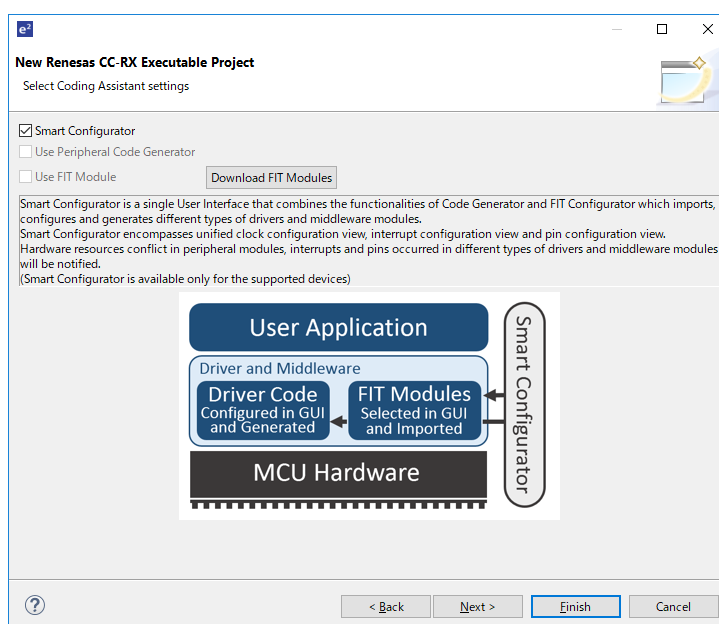


Figure 5 Coding Assistant Selection

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

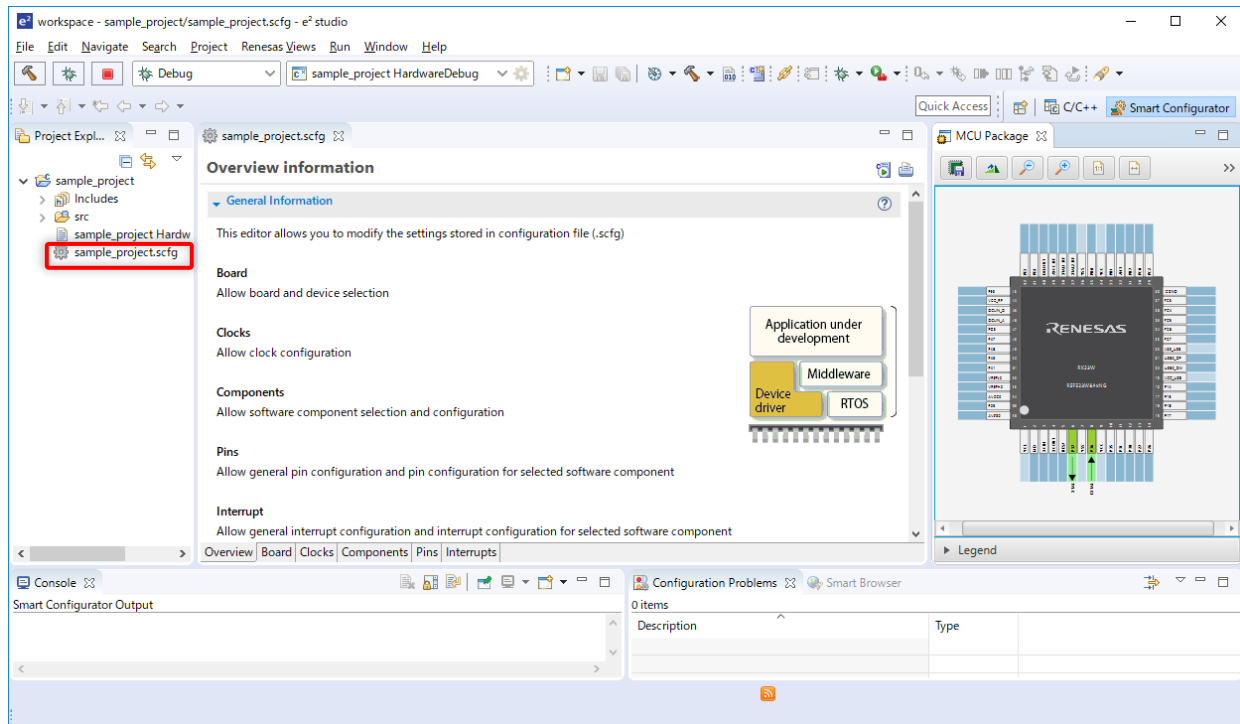


Figure 6 Completion of New Project Creation

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

BLE 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.

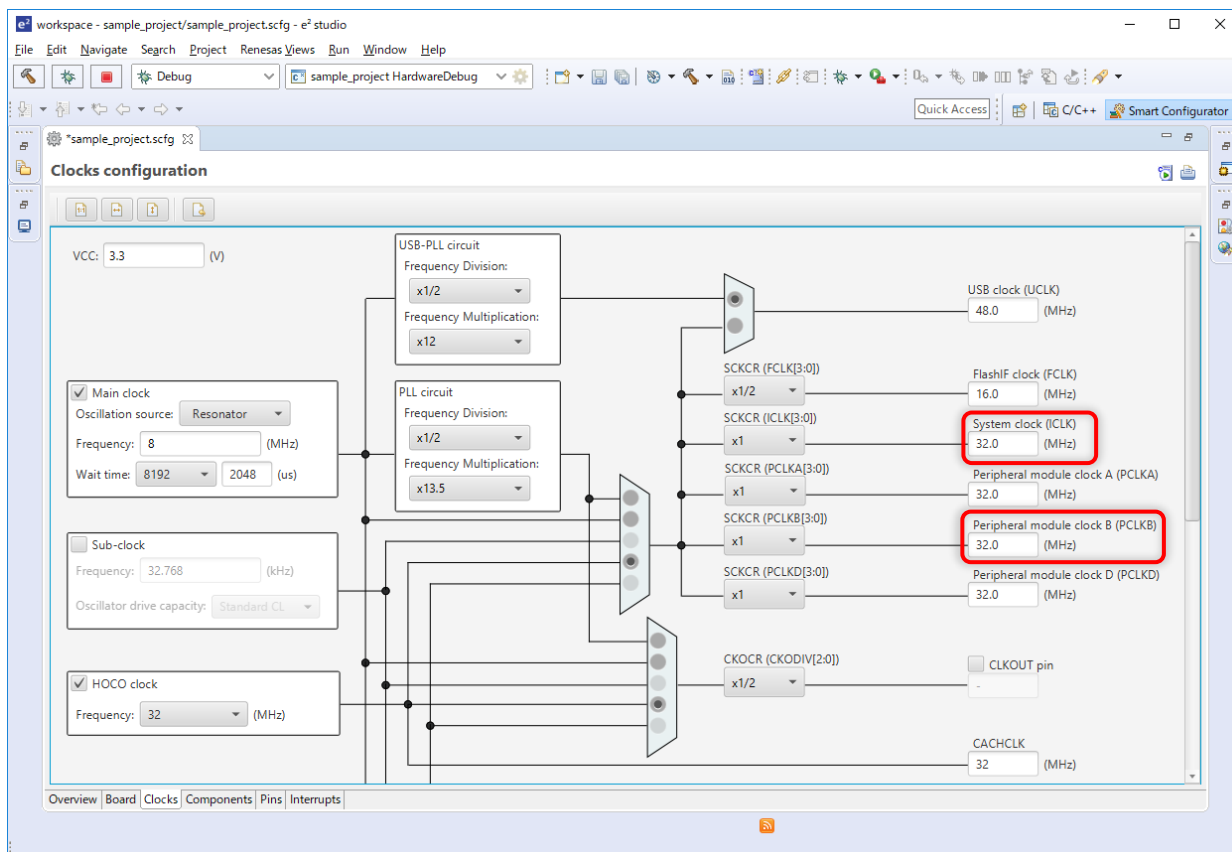



Figure 7 Clock Configuration

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 click [Finish] button.

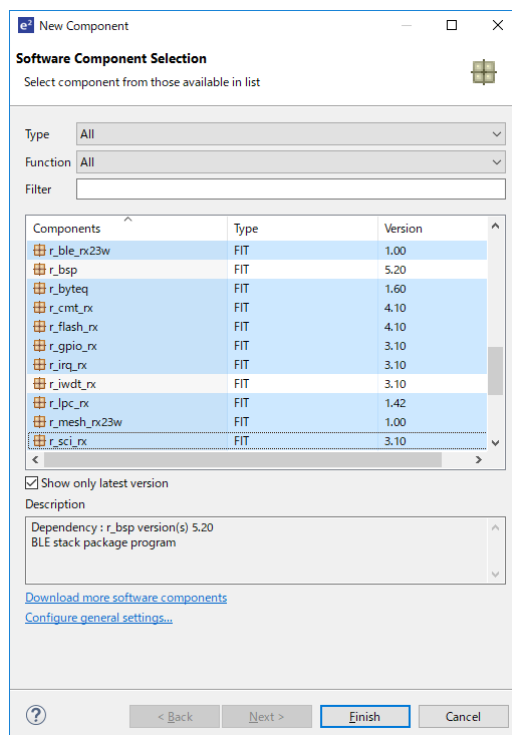


Figure 8 Software Components Selection

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.

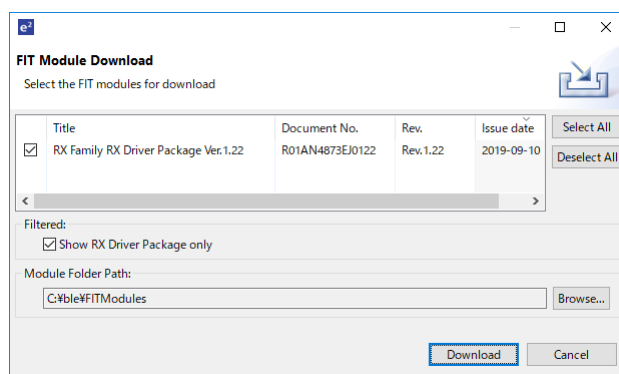


Figure 9 FIT Module Download

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.

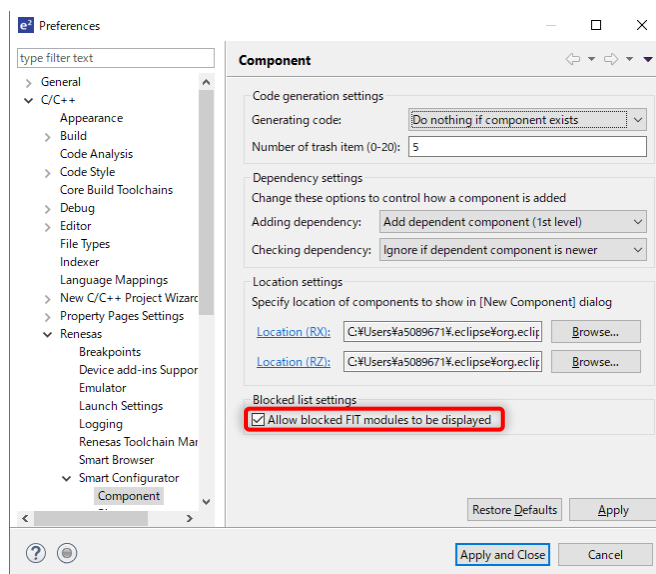


Figure 10 Display All FIT Modules

Selected FIT Modules are added on [Components] tab.

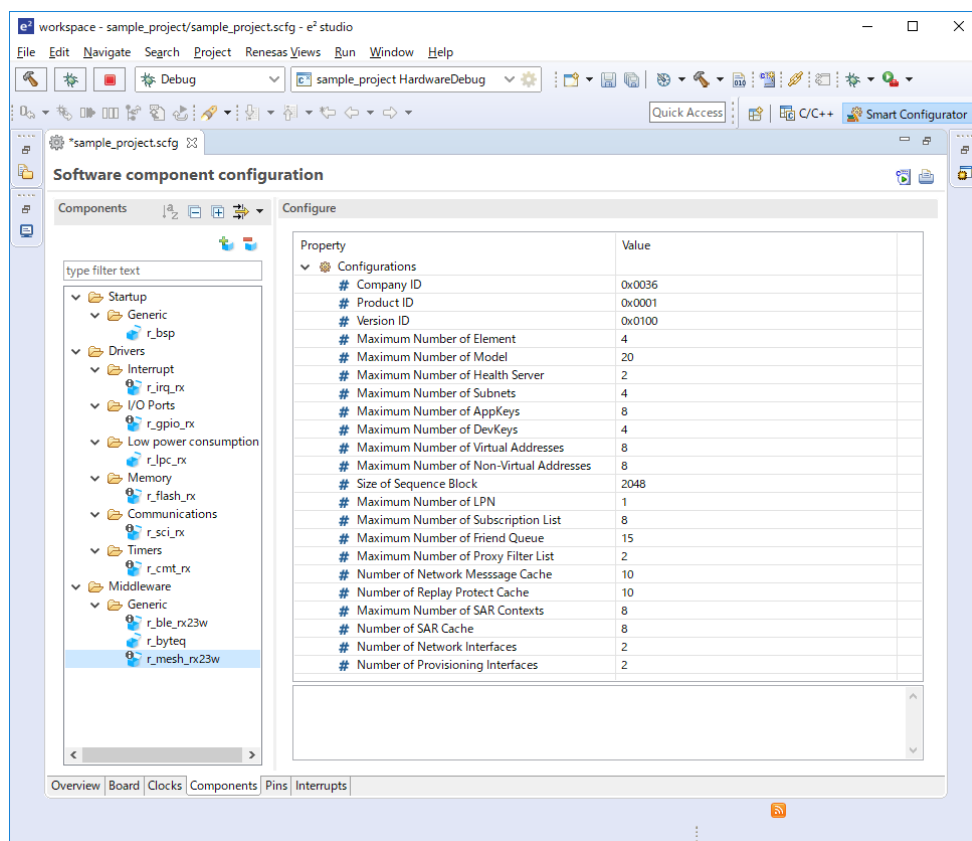


Figure 11 Added Software Components

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 "0x1000" to [Heap size] in [Components] tab.

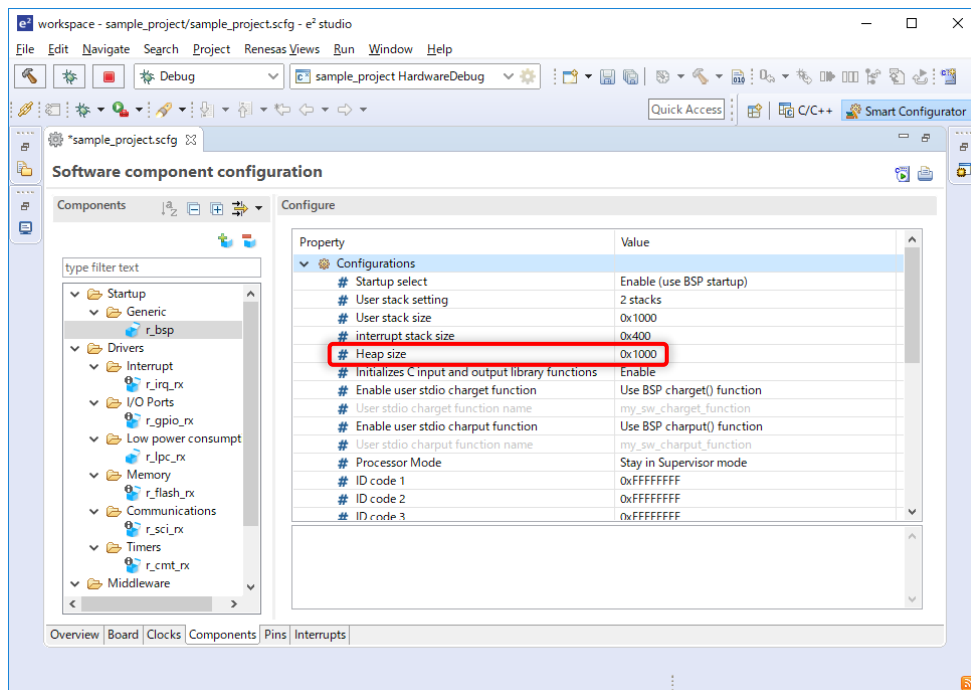


Figure 12 r_bsp Configuration

5.4.2 r_ble_rx23w

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

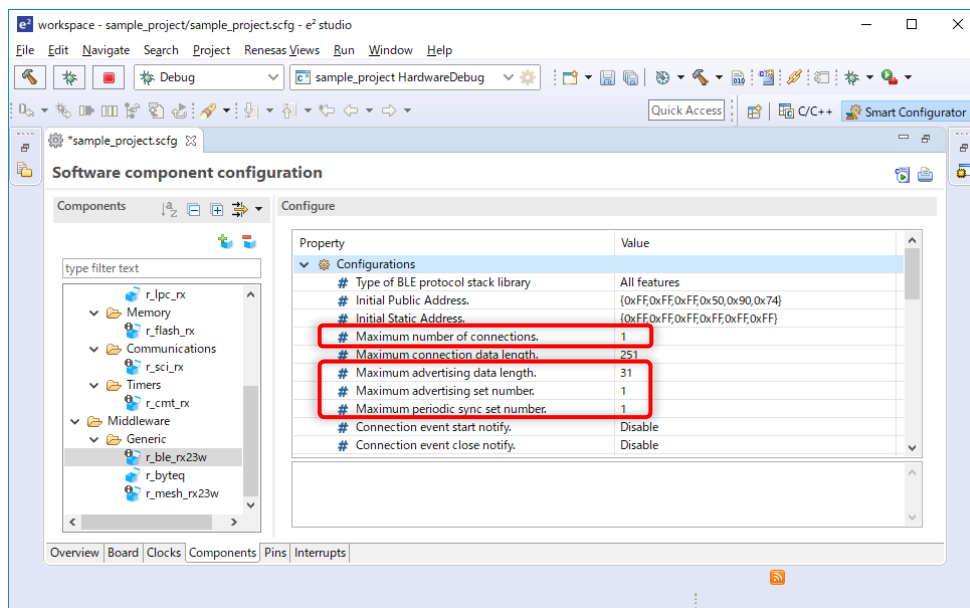


Figure 13 r_ble_rx23w Configuration (1)

If you use either Target Board for RX23W or RSSK for RX23W, set "8" to [SCI CH for command line function], then select "Target Board" or "RSSK" as [Board Type].

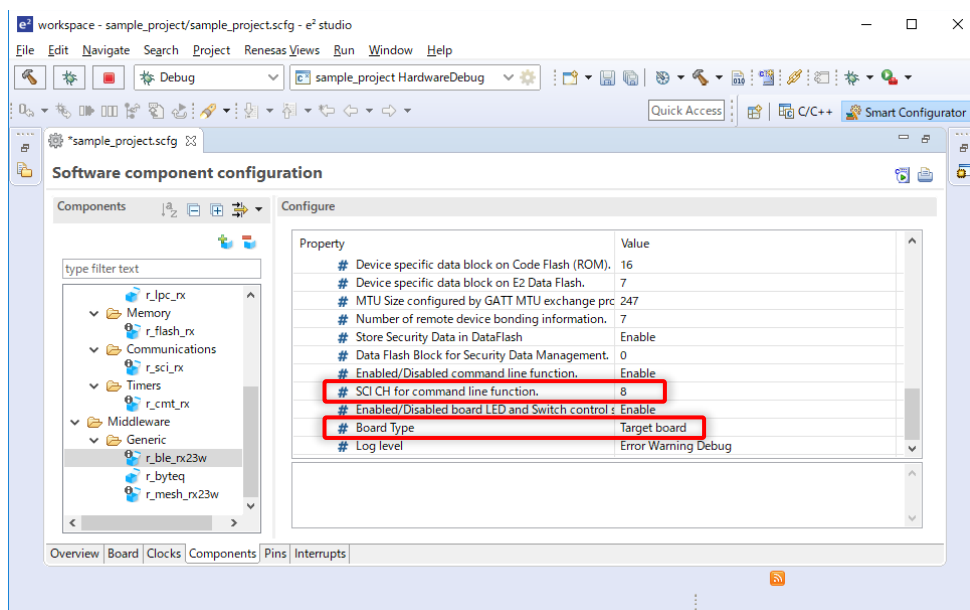


Figure 14 r_ble_rx23w Configuration (2)

5.4.3 r_sci_rx

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

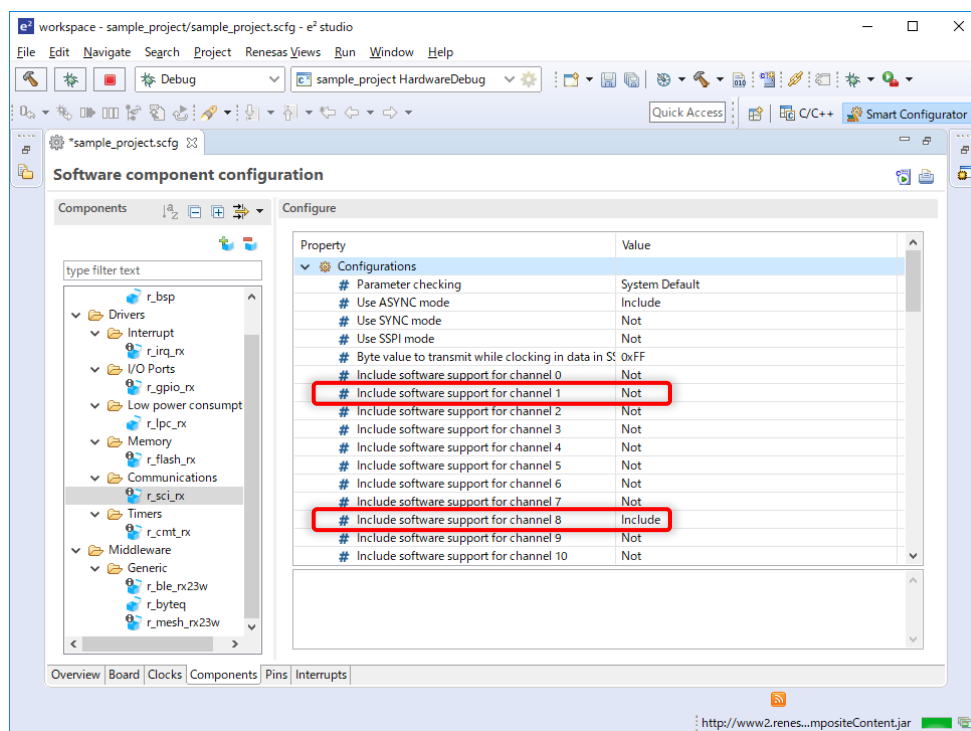


Figure 15 r_sci_rx Configuration (1)

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

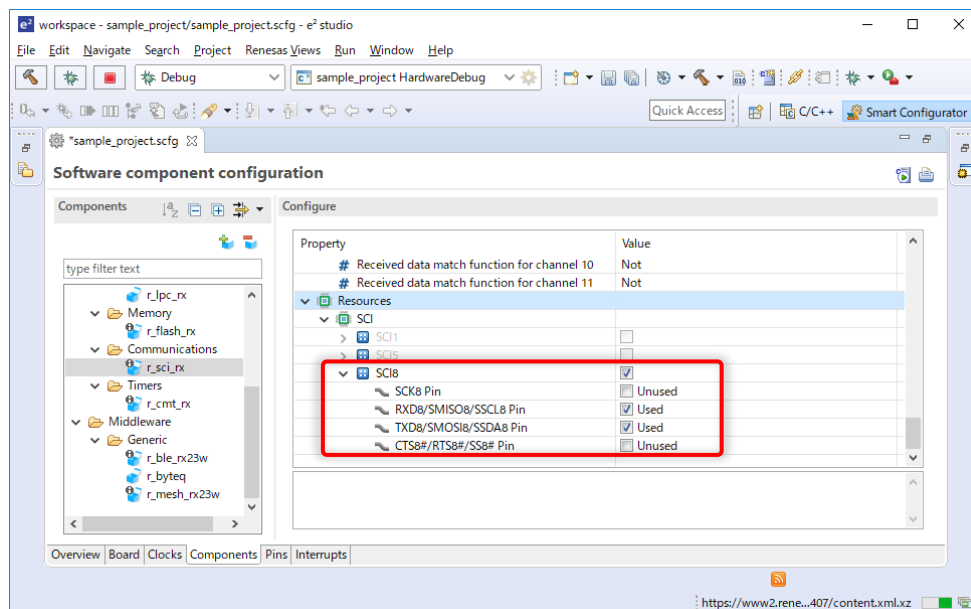


Figure 16 r_sci_rx Configuration (2)

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

To use this functionality, set "Enable" to [Transmit end interrupt]. If you use either Target Board for RX23W or RSSK for RX23W, set "160" to [ASYNC mode TX queue buffer size for channel 8].

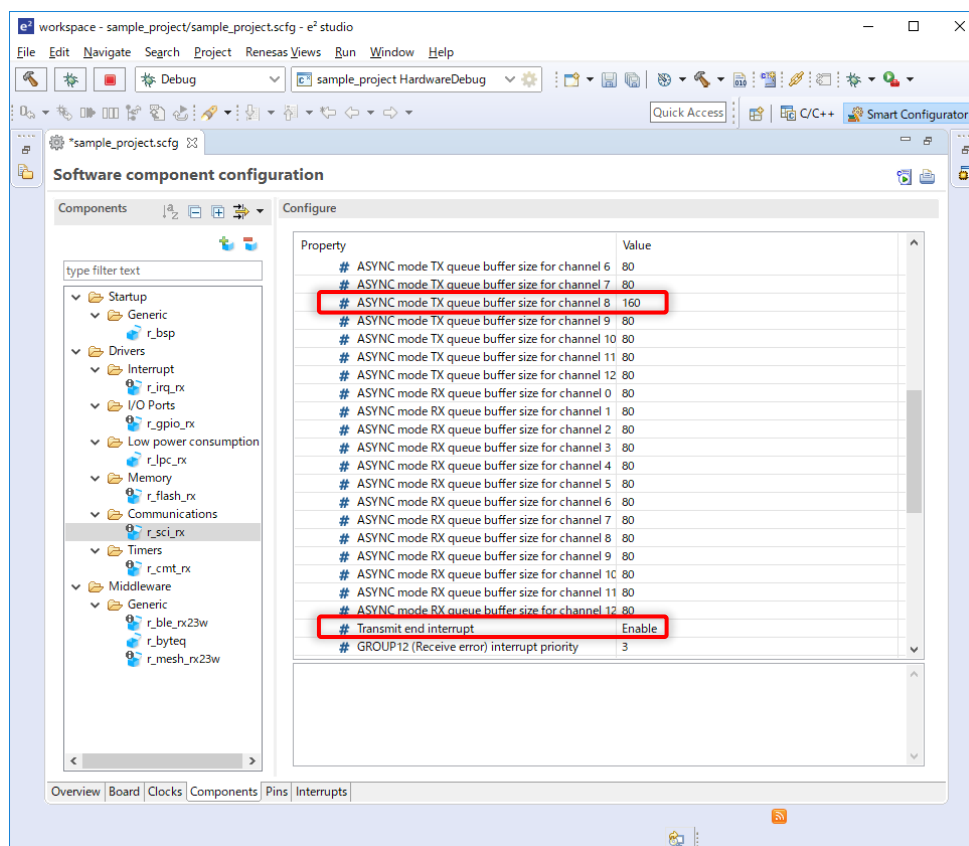


Figure 17 r_sci_rx Configuration (3)

5.4.4 r_irq_rx

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

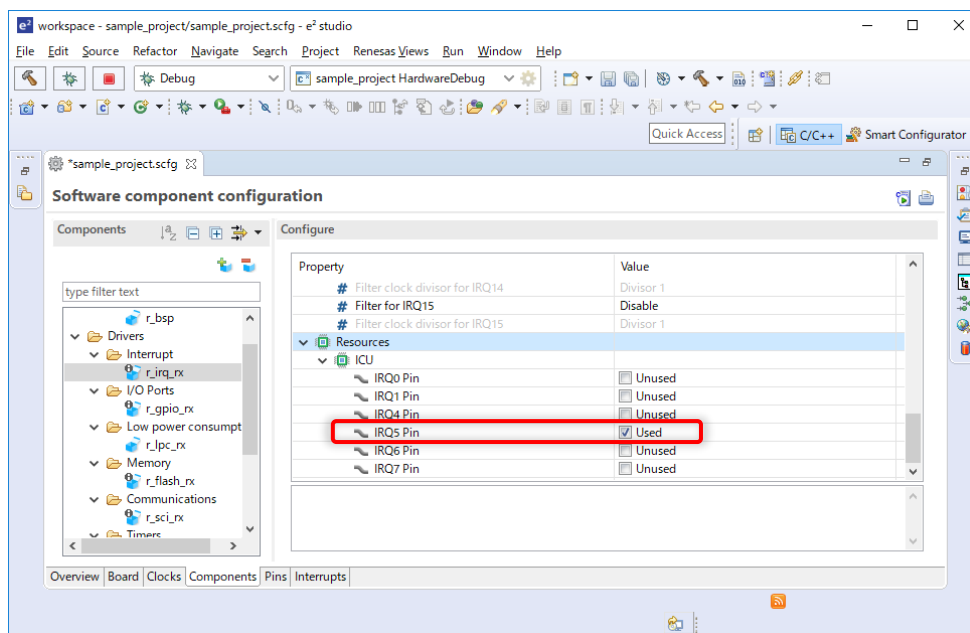


Figure 18 r_irq_rx Configuration (1) for Target Board

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

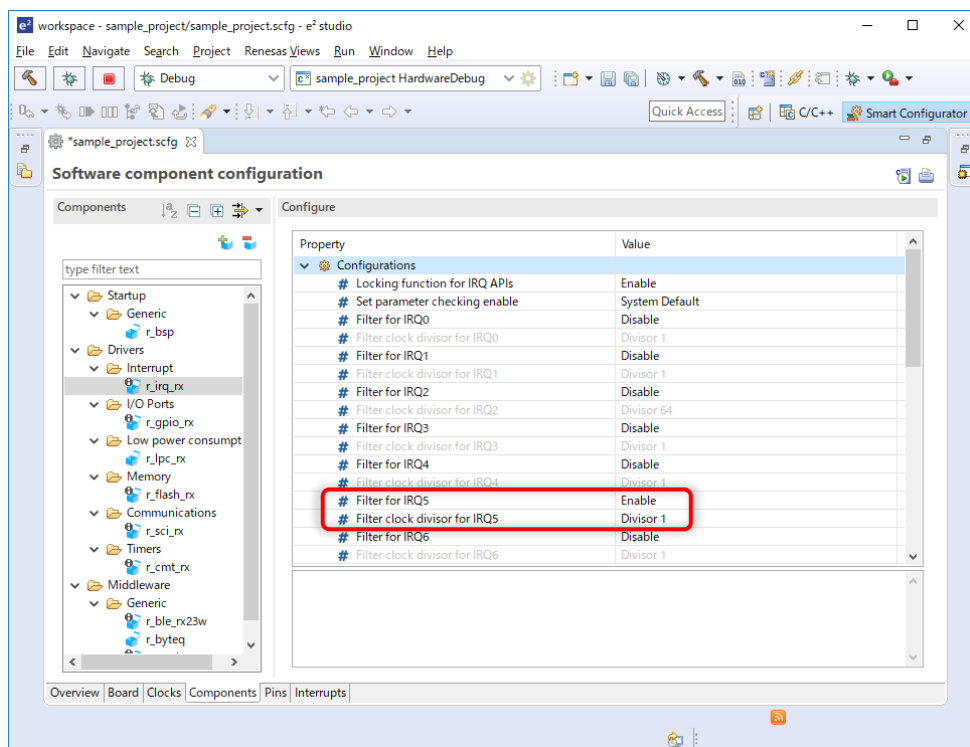


Figure 19 r_irq_rx Configuration (2) for Target Board

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

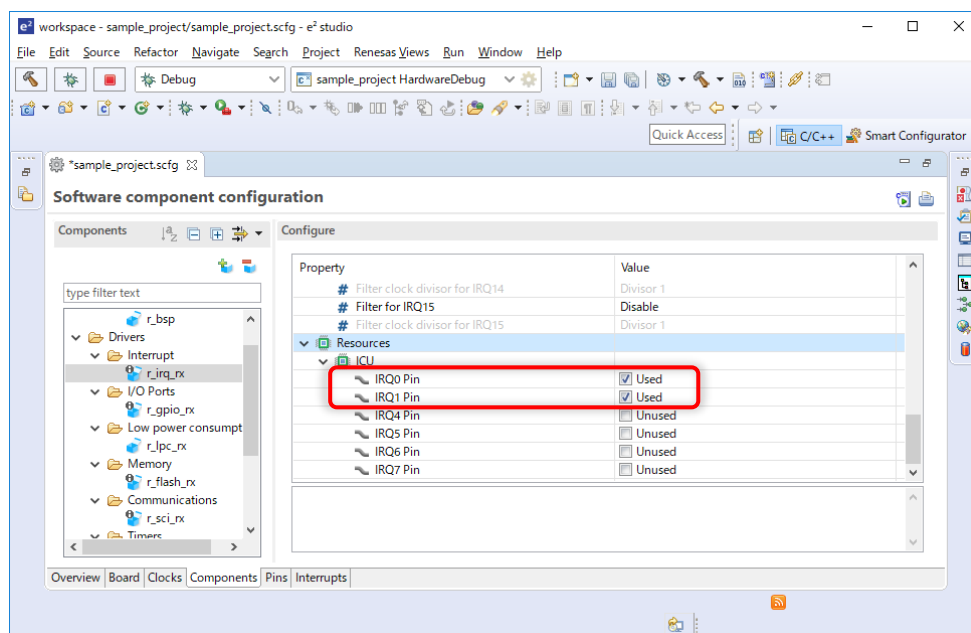


Figure 20 r_irq_rx Configuration (1) for RSSK

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

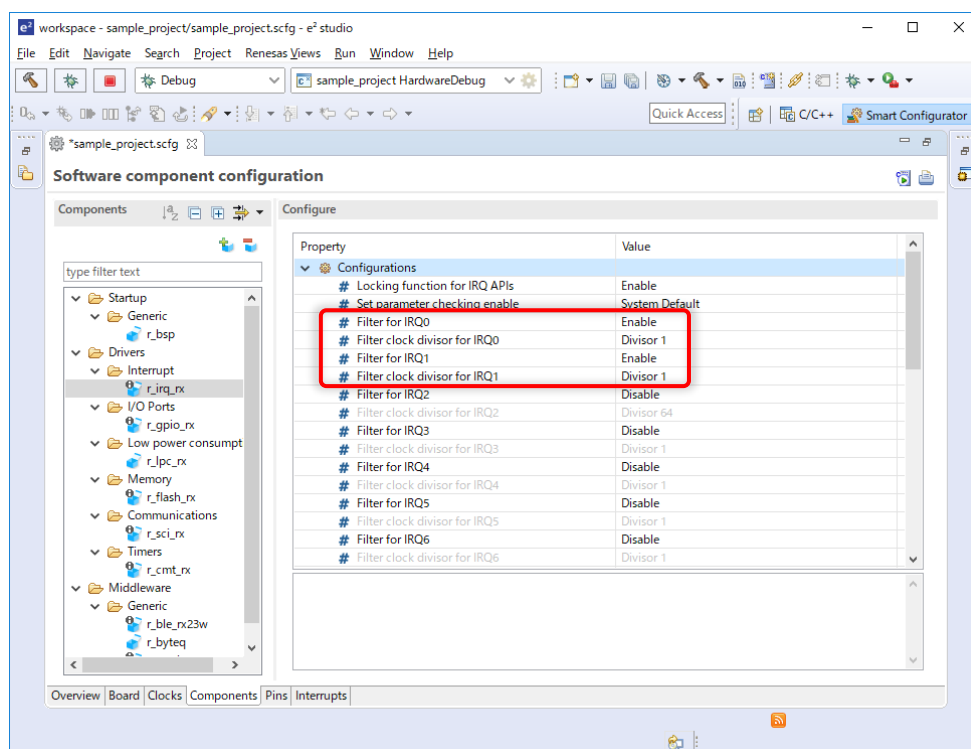



Figure 21 r_irq_rx Configuration (2) for RSSK

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.

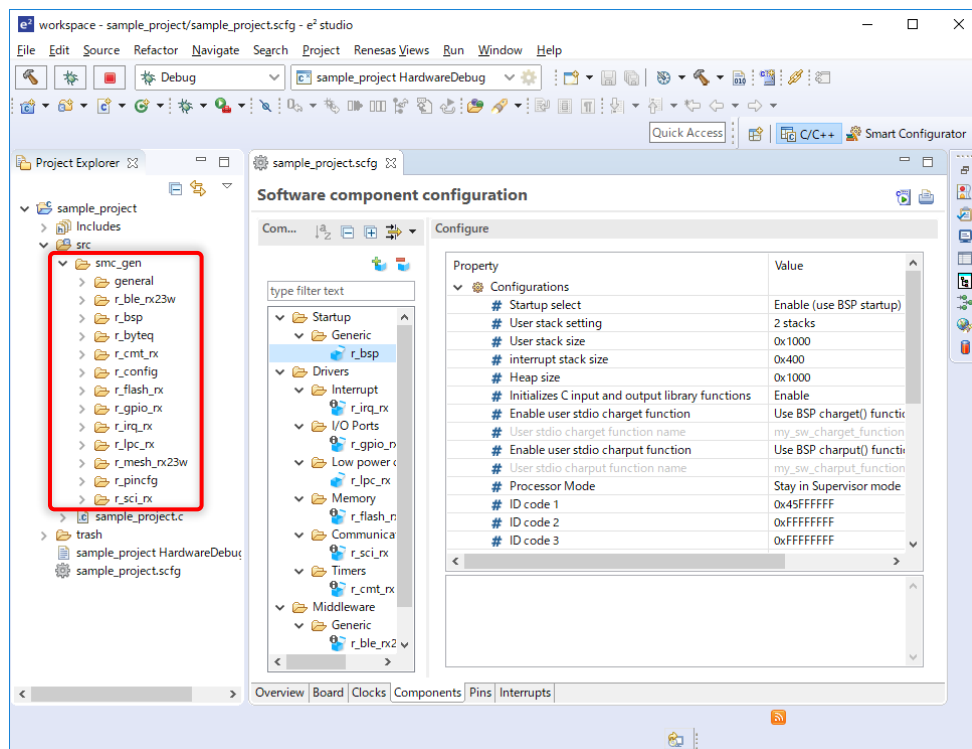


Figure 22 Result of Code Generation

5.6 Change Code

After Smart Configurator generates codes, it is necessary to change codes to use Mesh FIT Module.

5.6.1 r_ble_rx23w

To avoid unexpected behavior of BLE FIT Module, apply the following changes.

(1) Software Timer

To solve the issue that timeout is not informed in the case that API of Software Timer is invoked at the specified timing, apply the following change to the code generated by r_ble_rx23w.

File: src\smc_gen\r_ble_rx23w\src\app_lib\timer\r_ble_timer.c

Note: Apply this change to Rev2.00 or earlier of r_ble_rx23w. This change will be included in the code generated by the revisions later than Rev2.00 of r_ble_rx23w.

Refer to the following code and update the implementation of update_remaining_time_ms() and process_timer_expires() for (BSP_CFG_RTOS_USED == 0).

1. update_remaining_time_ms(): Add the code indicated with red.

```
static void event_cb(void); // prototype declaration

static void update_remaining_time_ms(bool expired)
{
    R_BSP_InterruptsDisable();

    uint8_t set_event = false;
    uint32_t elapsed_time_ms = pl_get_elapsed_time_ms(expired);

    for (uint32_t i = 0; i < BLE_TIMER_NUM_OF_SLOT; i++)
    {
        if (BLE_TIMER_STATUS_STARTED == gs_timer[i].status)
        {
            if (gs_timer[i].remaining_time_ms > elapsed_time_ms)
            {
                gs_timer[i].remaining_time_ms -= elapsed_time_ms;
            }
            else
            {
                gs_timer[i].remaining_time_ms = 0;
                gs_timer[i].status = BLE_TIMER_STATUS_EXPIRED;
                set_event = true;
            }
        }
    }

    if (false != set_event)
    {
        R_BLE_SetEvent(event_cb);
    }

    R_BSP_InterruptsEnable();
}
```

2. `update_remaining_time_ms()`: Delete the code indicated with gray.

```
void process_timer_expire(void)
{
    update_remaining_time_ms(true);

    for (uint32_t i = 0; i < BLE_TIMER_NUM_OF_SLOT; i++)
    {
        if ((BLE_TIMER_STATUS_STARTED == gs_timer[i].status) &&
            (0 == gs_timer[i].remaining_time_ms))
        {
            gs_timer[i].status = BLE_TIMER_STATUS_EXPIRED;
            R_BLE_SetEvent(event_cb);
        }
    }

    start_timer();
}
```

(2) RF Sleep

To solve the issue that BLE Protocol Stack included in BLE FIT Module stops Scan operation unexpectedly in the case Advertising operation is performed continuously, apply the following change to the code generated by `r_ble_rx23w`.

File: `src\smc_gen\r_ble_rx23w\src\platform\r_ble_pf_functions.c`

Note: Apply this change to Rev2.00 or earlier of `r_ble_rx23w`. This change will not be necessary in the code generated by the revisions later than Rev2.00 of `r_ble_rx23w`.

`r_ble_rf_power_save_mode()`: Add the code indicated with red.

```
/* This API is called within BLE driver and specifies the feedback for power saving operation of
BLE driver. */
BLE_SECTION_P uint8_t r_ble_rf_power_save_mode(void)
{
    uint8_t ret = BLE_CFG_RF_DEEP_SLEEP_EN;

    /*
    * When this function returns 0x0, RF does not enter power saving mode.
    * When this function returns 0x1, RF enter power saving mode.
    *
    * When the application layer occupies more time than the RF event time,
    * RF communication can be maintained by returning 0x0 with this function.
    */

    #if BLE_CFG_RF_DEEP_SLEEP_EN
    /* Disable RF power saving mode during Scan is active */
    extern uint8_t blebrr_scanstate;
    ret = (blebrr_scanstate == 0) ? 1 : 0;
    #endif /* BLE_CFG_RF_DEEP_SLEEP_EN */

    return ret;
}
```

5.7 Configure Link Options

5.7.1 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*

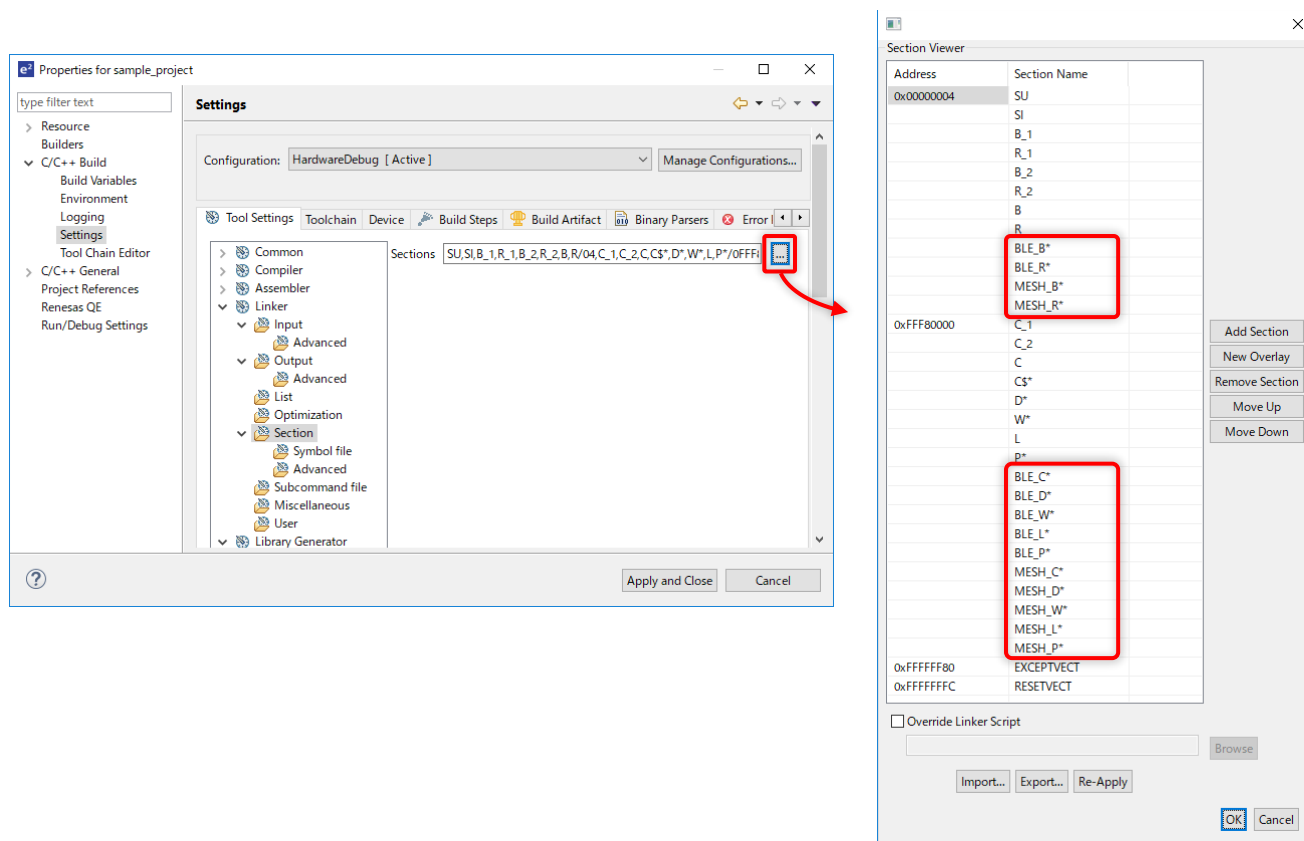


Figure 23 Section Configuration

(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

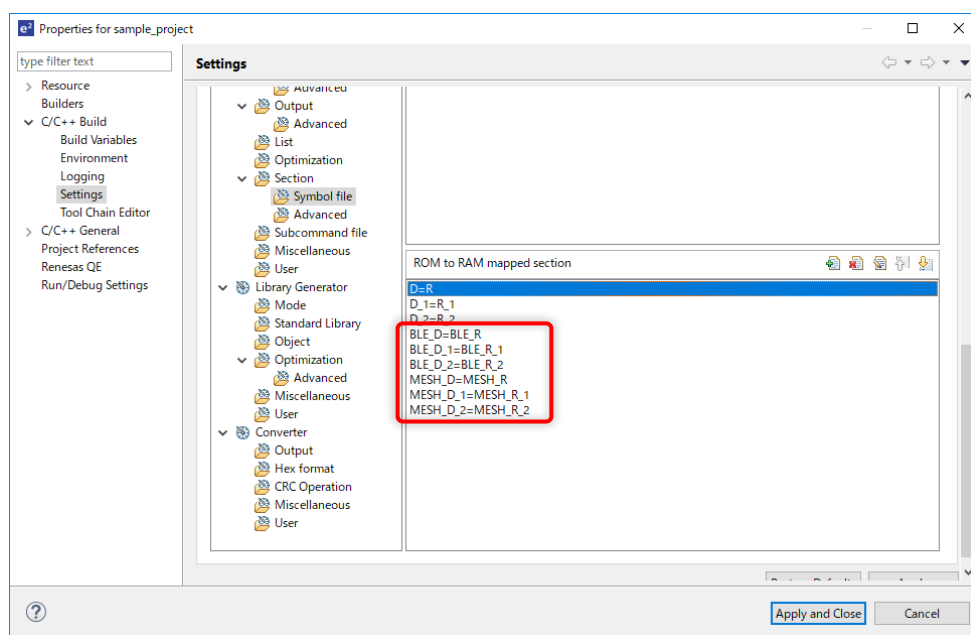


Figure 24 ROM to RAM mapped Section Setting

5.7.2 Libraries

Both Mesh Stack Library of Mesh FIT Module and BLE 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"

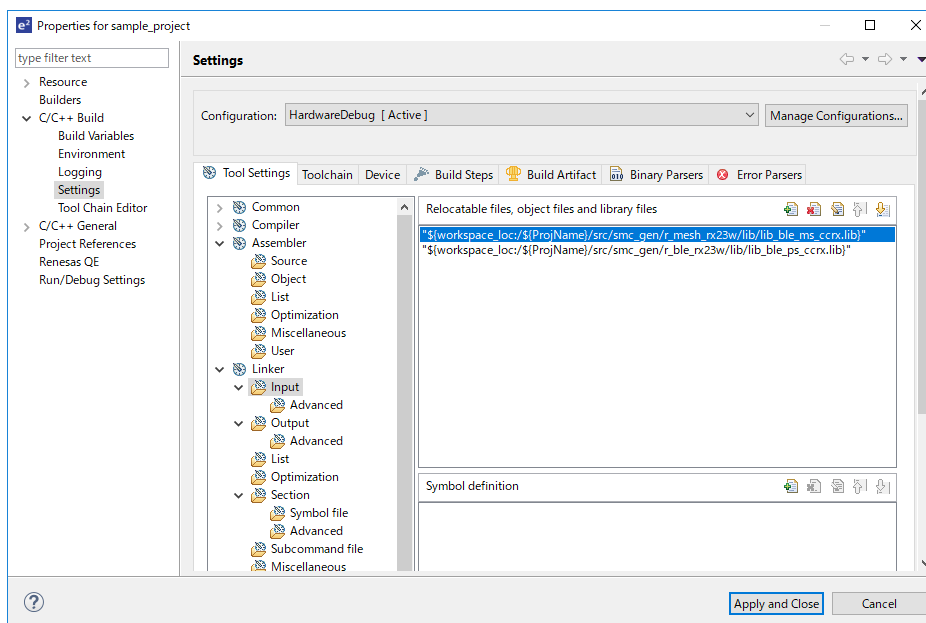


Figure 25 Library Files Setting

(2) Adding Prebuild Command

To use BLE 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
```

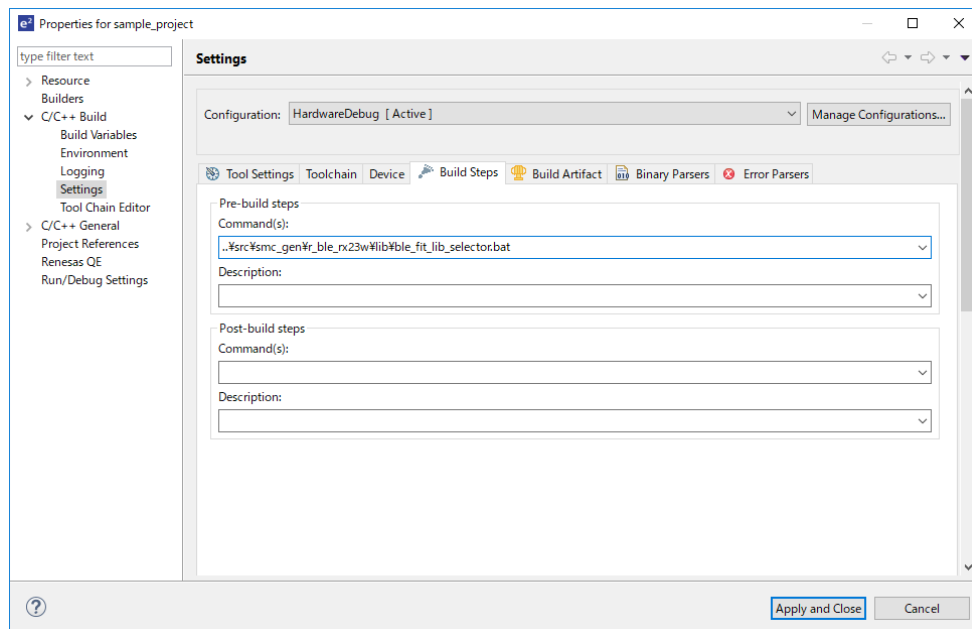


Figure 26 Prebuild Command Setting

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

5.8 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.

5.8.1 Debugger Connection

In [Debug hardware], select a debugger you use. If you use either Target Board for RX23W or RSSK for RX23W, set "HOCO" to [Clock]→[Main Clock Source] and "No" to [Power]→[Power Target From The Emulator] respectively.

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 "45FFFFFFFFFFFFFFFFFFFFFFFF" to [Flash]→[ID Code] in [Connection Settings] tab of [Debugger] tab.

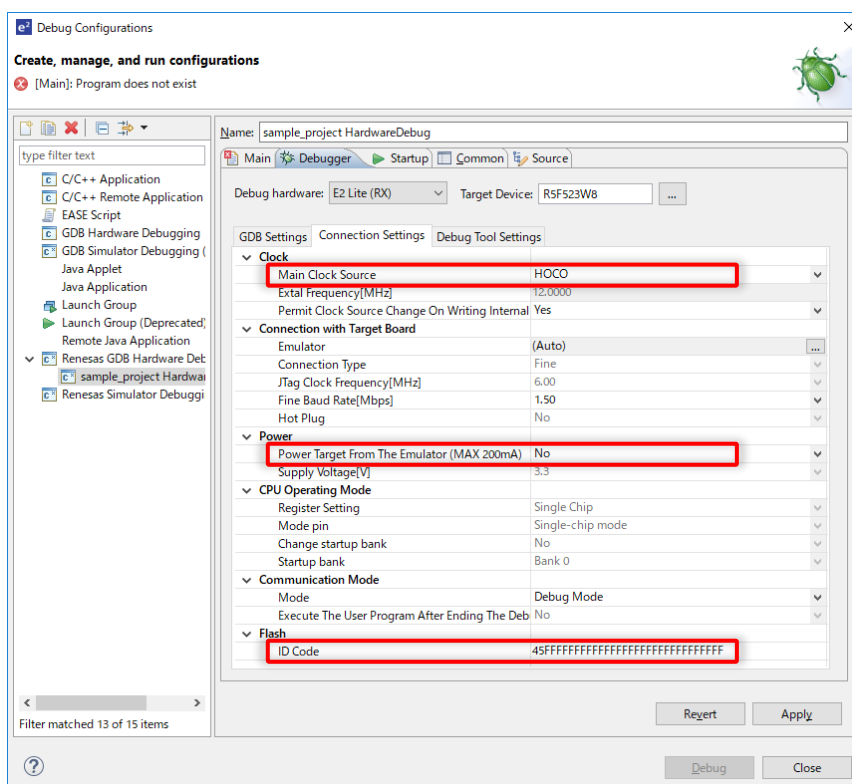


Figure 27 Debugger Connection Settings

5.9 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

The *Bluetooth*® word mark and logos are registered trademarks owned by Bluetooth SIG, Inc. and any use of such marks by Renesas Electronics Corporation is under license. Other trademarks and registered trademarks are the property of their respective owners.

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

- [crackle](#); AES-CCM, AES-128bit functionality
BSD 2-Clause License

Copyright (c) 2013-2018, Mike Ryan
All rights reserved.

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.

Program Updates (MESH FIT Module)

■ Rev1.01

- Bluetooth Bearer Wrapper

blebrr.c

- Added randomized delay to Advertising transmission timing.

■ Rev1.10

- Bluetooth Mesh Stack

Common

- Added MS_systemtime_init_pl() for registering system time function.
- Added Mesh Monitoring Functionality and added MS_monitor_register_pl().

Provisioning

- Added prefix to function names.

prov_set_device_oob_pubkey_pl()	→ MS_prov_set_device_oob_pubkey_pl()
prov_set_static_oob_auth_pl()	→ MS_prov_set_static_oob_auth_pl()
prov_clear_device_oob_pubkey_pl()	→ MS_prov_clear_device_oob_pubkey_pl()
prov_clear_static_oob_auth_pl()	→ MS_prov_clear_static_oob_auth_pl()
mempool_init_pl()	→ MS_mempool_init_pl()
storage_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 and added config_MS_NET_SEQNUM_CACHE_SIZE parameter to MS_CONFIG structure to configure size of the cache per node.
- Added MS_net_register_tx_state_access() to check transmission state of Network layer.
- Added MS_access_set_iv_update_test_mode() to disable 96 hours check during IV Update.
- 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 MS_ltrn_register_tx_state_access() to check transmission state of Lower Transport layer.
- 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 : Configuration Client model
 - MS_HEALTH_CLIENT_CB : Health Client model
 - MS_GENERIC_ONOFF_CLIENT_CB : Generic OnOff Client model
 - MS_GENERIC_LEVEL_CLIENT_CB : Generic Level Client model
 - MS_GENERIC_DEFAULT_TRANSITION_TIME_CLIENT_CB : Generic Default Transition Time Client model
 - MS_GENERIC_POWER_ONOFF_CLIENT_CB : Generic Power OnOff Client model
 - MS_GENERIC_POWER_LEVEL_CLIENT_CB : Generic Power Level Client model
 - MS_GENERIC_BATTERY_CLIENT_CB : Generic Battery Client model
 - MS_GENERIC_LOCATION_CLIENT_CB : Generic Location Client model
 - MS_GENERIC_PROPERTY_CLIENT_CB : Generic Property Client model
 - MS_SENSOR_CLIENT_CB : Sensor Client model
 - MS_TIME_CLIENT_CB : Time Client model
 - MS_SCENE_CLIENT_CB : Scene Client model
 - MS_SCHEDULER_CLIENT_CB : Scheduler Client model
 - MS_LIGHT_LIGHTNESS_CLIENT_CB : Light Lightness Client model
 - MS_LIGHT_CTL_CLIENT_CB : Light CTL Client model
 - MS_LIGHT_HSL_CLIENT_CB : Light HSL Client model
 - MS_LIGHT_XYL_CLIENT_CB : Light xyl Client model
 - MS_LIGHT_LC_CLIENT_CB : Light LC Client model
 - 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 Wrapper
 - blebrr.c, blebrr.h
 - Changed timer used for controlling Advertising transmission to BLE software timer (R_BLE_TIMER) from internal timer of BLE 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.
 - Solved the issue that Transmission Queue leaks in the case that malloc() fails to allocate memory for data buffer.
 - blebrr_pl.c, blebrr.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 R_MS_BRR_Scan_GattBearer() to find devices having Mesh GATT Service.
 - Added functionality to find devices whose Advertising data includes UUID of Mesh GATT Service and added R_MS_BRR_Scan_GattBearer().
 - Added prefix to function names.

blebrr_init()	→ R_MS_BRR_Init()
blebrr_register()	→ R_MS_BRR_Setup()
blebrr_register_gatt_iface_event_pl()	→ R_MS_BRR_Register_GattIfaceCallback()
blebrr_gatt_mode_set()	→ R_MS_BRR_Set_GattMode()
blebrr_gatt_mode_get()	→ R_MS_BRR_Get_GattMode()
blebrr_disconnect_pl()	→ R_MS_BRR_Disconnect()
blebrr_set_adv_scanrsp_data_pl()	→ R_MS_BRR_Set_ScanRspData()
blebrr_create_conn_pl()	→ R_MS_BRR_Create_Connection()
blebrr_discover_service_pl()	→ R_MS_BRR_Discover_Service()
blebrr_config_ntf_pl()	→ 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 by using 8-bit Timer (TMR).

Program Updates (Mesh Sample Program of MESH FIT Module Package)

■ Rev1.01

mesh_cli folder

- 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 folder

- 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_GattIfcCallback()
- 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 layer of 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 ELEMENT_DESC_LOCATION macro to configure Element Location of Composition Data state.
- 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 Wrapper functions.
 - R_MS_BRR_Init()
 - R_MS_BRR_Setup()
- Updated implementation in accordance with the specification change of Bluetooth Bearer Wrapper 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.

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"

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 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.
5. 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.

6. 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.
7. 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.
8. 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.
9. 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.
10. 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.
11. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products.

(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.4.0-1 November 2017)

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/.