

RA4W1 Group

Bluetooth Mesh sample application

Introduction

This document describes the sample application which uses the Bluetooth[®] Mesh Stack. Bluetooth[®] Mesh Stack is the software library to which is used to build a mesh network that is compliant with Bluetooth Mesh Networking Specification and to perform many-to-many wireless communication.

In this document, the Bluetooth® Mesh is referred to as the Mesh.

For more details on how to perform the Mesh demonstration which uses this sample application, refer to "RA4W1 Group Bluetooth Mesh Startup Guide" (R01AN5847).

Target Device

RA4W1 Group

Related Documents

- Bluetooth Core Specifications (<u>https://www.bluetooth.com</u>)
- Mesh Profile Specification (Search for "Mesh Profile 1.0.1" in <u>https://www.bluetooth.com</u>)
- Mesh Model Specification (Search for "Mesh Model 1.0.1" in https://www.bluetooth.com)
- Renesas Flexible Software Package (FSP) User's Manual (R11UM0155)
- e² studio User's Manual: Getting Started Guide (R20UT4374)
- RA4W1 Group BLE sample application (R01AN5402)
- RA4W1 Group Bluetooth Mesh Startup Guide (R01AN5847)
- RA4W1 Group Bluetooth Mesh Development Guide (R01AN5849)



Contents

1.	Overview	3
1.1	Demo projects	3
1.2	Mesh Stack features	5
1.3	Software Architecture	6
1.4	File Composition	7
1.5	API Specification	7
1.6	Operating environment	8
2.	How to use demo project	9
2.1	Importing demo project	9
2.2	Building and debugging	11
3.	How to make and configure new project	12
3.1	Create a New Project	12
3.2	Heap and Stack configuration	15
3.3	Clocks configuration	16
3.4	Add and configure MESH Stack	17
3.4.1	Add MESH Stack in BareMetal environment	17
3.4.2	2 Add MESH Stack in FreeRTOS environment	20
3.4.3	3 Configure Mesh Stack	25
3.4.4	Oher Mesh Stack configuration	27
3.5	Add and configure related module	32
3.5.1	r_gpt (g_timer0)	33
3.5.2	2 r_flash_lp	34
3.5.3	3 r_icu (g_external_irq0)	35
3.5.4	r_gpt (g_timer1)	36
3.5.5	5 r_icu (g_ble_sw_irq)	37
3.5.6	6 r_sci_uart	38
3.5.7	′ r_lpm	40
3.6	Generate Code	41
3.7	Building and debugging	41
4.	How to Implement Mesh Applications	42
5.	Appendix	43
5.1	Program Size	43
Rev	ision History	45



1. Overview

1.1 Demo projects

Projects for the sample application accompanying this document are shown in Table 1-1.

Table 1-1 Projects

Project name	Description
ekra4w1_mesh_client_baremetal	Client Models project for EK-RA4W1 not using FreeRTOS.
ekra4w1_mesh_client_freertos	Client Models project for EK-RA4W1 using FreeRTOS.
ekra4w1_mesh_server_baremetal	Server Models project for EK-RA4W1 not using FreeRTOS.
ekra4w1_mesh_server_freertos	Server Models project for EK-RA4W1 using FreeRTOS.
ekra4w1_mesh_cli_client_baremetal	Command Line Interface (CLI) project for EK-RA4W1 not using FreeRTOS.
ekra4w1_mesh_cli_server_baremetal	Command Line Interface (CLI) project for EK-RA4W1 not using FreeRTOS.

These projects can work on an EK-RA4W1 board.

The Server Models projects perform Server model. They receive messages from remote device (e.g. smart phone) performing Client model, then blink the board mounted LED, and display the received strings.

The Client Models projects perform Client model. They support CLI which can be accessed by using a terminal emulator (e.g. Tera Term) on a PC by connecting it with an EK-RA4W1 board via USB cable. They send messages to the remote device performing Server model by pushing the board mounted switch or by using CLI, then blink the board mounted LED, and display strings on the remote device.

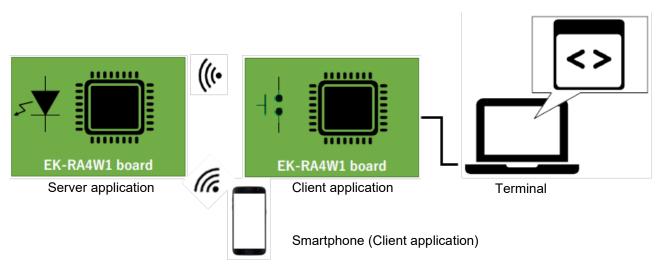
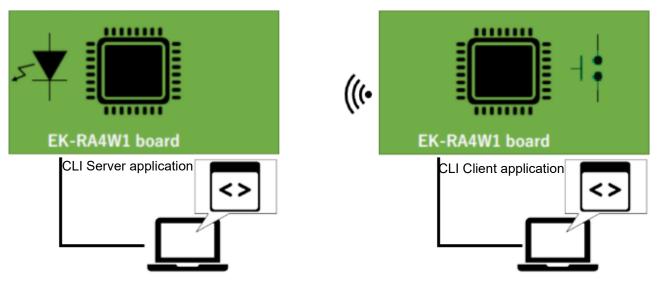


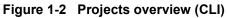
Figure 1-1 Projects overview (Server and Client)



RA4W1 Group

CLI projects can perform all models defined Mesh Specification. They can perform various procedures relating to the Mesh by transmitting and receiving messages using CLI.







1.2 Mesh Stack features

The Mesh Stack 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 stack 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



1.3 Software Architecture

Figure 1-3 shows the software architecture using Mesh Stack.

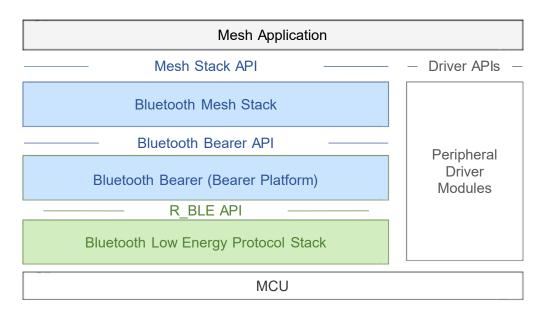


Figure 1-3 Software Architecture

The Mesh Stack software is composed of the followings:

Mesh Application

The Mesh Application is an application which performs features provided by the Bluetooth Mesh Stack.

• Bluetooth Mesh Stack

The Bluetooth Mesh Stack is a software that provides applications with many-to-many wireless communication features which are compliant with Bluetooth Mesh Networking Specifications.

• Bluetooth Bearer (Bearer Platform)

The Bluetooth Bearer is the abstraction layer that provides wrapper functions of Bluetooth Low Energy Protocol Stack.

Bluetooth Low Energy Protocol Stack

The Bluetooth Low Energy Protocol Stack (hereinafter referred to as "Bluetooth LE Stack") is the software that provides the higher layers with wireless communication features which are compliant with the Bluetooth Low Energy specifications.

A Sample program of Mesh Application is included in the demo project included in this document.

Bluetooth Mesh Stack and Bluetooth Bearer are provided as FSP.

Bluetooth LE Stack is provided as <u>FSP</u>.



1.4 File Composition

File composition of demo project is as follows:

```
ekra4w1_mesh_xxx_yyy
                                                          Project folder
  +---ra\fsp\inc\api\
                                                          Bluetooth LE Stack Header File
           r_ble_api.h
           rm_ble_mesh_xxx_api.h
                                                          Mesh Stack API Header File
           rm_mesh_bearer_platform_api.h
                                                          Bluetooth Bearer API Header File
  +---ra\fsp\inc\instances\
                                                          Mesh Stack Header File
           rm_ble_mesh_xxx.h
           rm_mesh_bearer.h
                                                          Bluetooth Bearer Header File
                                                          Mesh Stack Header File (Model Feature)
           rm_mesh_xxx.h
  +---ra\fsp\lib\r ble\
                                                          Bluetooth LE Stack Library
  +---ra\fsp\lib\rm_ble_mesh\
                                                          Mesh Stack Library
  +---ra_cfg\fsp_cfg\
                                                          Bluetooth LE Configuration
           r_ble_cfg.h
                                                          Mesh Stack Configuration
           rm ble mesh cfg.h
  +---src\
                                                          Mesh Application
```

To use the features provided by the Mesh Stack, the Mesh Stack must be added to a project. Regarding how to add the stack to a project, refer to Chapter 3 in this document.

1.5 API Specification

To perform the features provided by the Mesh Stack, it is necessary to use the API of the Mesh Stack. Regarding the specification of Mesh Stack API, refer to "Renesas Flexible Software Package (FSP) User's Manual (R11UM0155)".



1.6 Operating environment

Table 1-2 shows the confirmed operating environment for hardware to build and debug the demo project.

Table 1-2 Hardware environment

Hardware	Description
Host PC	Windows [®] 10 PC with USB interface.
MCU board	The MCU used must support BLE functions.
	EK-RA4W1 [RTK7EKA4W1S00000BJ]
On-chip debugging emulators	The EK-RA4W1 has an on-board debugger (J-Link OB), therefore it is not necessary to prepare an emulator
USB cables	Used to connect to the MCU board.

Table 1-3 shows the confirmed operating environment for software to build and debug the demo project.

Table 1-3 Software environment

Software		Version	Description
GCC environment	e ² studio	2022-10	Integrated development environment (IDE) for Renesas devices.
	GCC ARM Embedded	V10	C/C++ Compiler. (Download from e ² studio installer)
	Renesas Flexible Software Package (FSP)	V4.2.0	Software package for making applications for the RA microcontroller series.
	SEGGER J-Flash	V6.86	Tool for programming the on-chip flash memory of microcontrollers.
Integer types			ANSI C99 "Exact width integer types". These types are defined in stdint.h.
Endian			Little endian



2. How to use demo project

This chapter describes how to use the demo project included in this document.

2.1 Importing demo project

Follow the steps bellow:

1. Launch the e^2 studio and select the workspace directory.

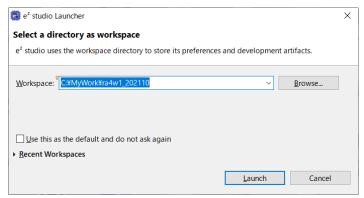


Figure 2-1 Select workspace

2. Select File \rightarrow Import.

File	Edit	Navigate	Search	Project	Renesas Views	R
۵,	New Open Open Recen	Projects fro	m File Sy	/stem	Alt+Shift+N	>
	Close	Lancon			Ctrl+W	
	Save	All Editors			Ctrl+Shift+W Ctrl+S	
	Save A Save A Revert	AII			Ctrl+Shift+S	
	Move.				50	
~	Renan				F2 F5	
5		n rt Line Deli	miters To		F3	>
Ð	Print				Ctrl+P	
	Impor	t		mport 📔		
4	Export	t				
	Prope	rties			Alt+Enter	
	Switch Restar	Workspace	e			>

Figure 2-2 File menu



3. Select **Existing Projects into Workspace** and click **Next** button.

Import		×
Select Create new projects from an archive file or directory.	[2
<u>S</u> elect an import wizard:		
type filter text		
 Seneral Archive File CMSIS Pack Existing Projects into Workspace File System Preferences Projects from Folder or Archive Rename & Import Existing C/C++ Project into Workspace 		< >
? < <u>B</u> ack <u>Next ></u> Einish	Canc	el

Figure 2-3 Select an import wizard

4. Select **Select root directory**, click **Browse...** button and select the demo project folder. Click **Finish** button to import the demo project.

Import				
Import Projects				
Select a directory to sea				
			04000100 4001	
Select roo <u>t</u> directory:		w1_202110¥r01an5		B <u>r</u> owse
Select <u>archive</u> file:	C:¥MyWork¥ra4	w1_202110¥new2.z	cip ~	B <u>r</u> owse
<u>P</u> rojects:				
ekra4w1_mesh_se	rver_baremetal (C:¥MyWork¥ra4w1	_202110¥r01an5848jj	Select All
				Deselect All
				R <u>e</u> fresh
<			>	
Options				
Search for nested pro	-			
Copy projects into w				
Close newly importe		-		
Working sets	,			
Add project to work	rina sets		[Ne <u>w</u>
	ang sets			
Working sets:				S <u>e</u> lect
?	. Da ala	N and a	Finish	Cancel
	< <u>B</u> ack	<u>N</u> ext >	<u>F</u> inish	Cancel

Figure 2-4 Import Project



2.2 Building and debugging

Follow the steps bellow:

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

1. Select [Build Project] in [Project] menu or click the Build icon solution to build the project. In [Console] tab, if you can see "Build Finished" message that follows build log, the build is successful.

You can see and change the current project with Launch Configuration Rever_barer > * After building, the firmware (.srec file) is generated in the "Debug\" in the project directory.

- 2. Connect EK-RA4W1 to a PC.
- 3. Click the Debug icon to launch the project in debug mode. After launching the project, the firmware is downloaded to EK-RA4W1.
- 4. Click the Resume icon Perspective to run the project.
- 5. After debugging the project, click the Terminate icon eo on Debug Perspective. Firmware of the project remains on the flash memory of RA4W1 even after termination and power off.

To perform the demonstration, it is recommended to use at least two EK-RA4W1; one board works as a Client and the other works as a Server.

NOTE: When the error indicating "No toolchain set or toolchain not integrated." occurs and building fails, open [Project]→[C/C++ Project Settings] and move to [C/C++ Build]→ [Settings]→[Toolchain] tag, then set the toolchain (9.3.1.20200408 or later).

pe filter text	Settings		🤤 🖛 🖙 🖛
Resource			
Builders	Configuration: Debug [Active	1	Manage Configurations
C/C++ Build Build Variables	Configuration: Debug [Active	.1	wanage configurations
Environment Logging	Tool Settings Toolchain	🎤 Build Steps 😤 Build Artif	act 🗟 Binary Parsers 🖪
Settings Tool Chain Editor	Name	GNU Tools for ARM Emb	bedded Processors (arm $ \sim $
C/C++ General	Architecture	ARM (AArch32) V	
MCU	Prefix	arm-none-eabi-	
Project Natures Project References	Suffix		
Renesas QE	C compiler	gcc	
Run/Debug Settings	C++ compiler	g++	
Task Tags Validation	Archiver	ar	
	Hex/Bin converter	objcopy	
	Listing generator	objdump	
	Size command	size	
	Build command	make	
	Remove command	rm	
	Create flash image		
	Create extended listing		
	Use integrated toolchain ve	rsion 0 3 1 20200408	
		55.1.20200400	D
	Per project path		Browse
		Restor	e Defaults Apply



3. How to make and configure new project

This chapter describes how to add Mesh Stack to a new project by using FSP Configuration in the e² studio.

3.1 Create a New Project

1. Launch e² studio and select [File]→[New]→[Renesas C/C++ Project]→[Renesas RA]. In New C/C++ Project dialog, select Renesas RA C/C++ Project and click on the Next button.

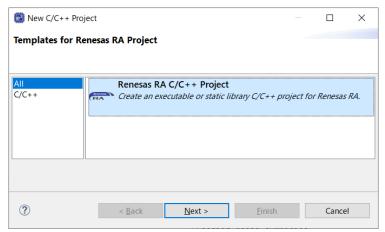


Figure 3-1 Templates for New C/C++ Project

2. Enter the project name and click on **Next** button. The project is named **sample_appl** in this document.

Renesas RA C/C++ Project					
Renesas RA C/C++ Project Project Name and Location					
Project name					
sample_appl					
Use default location					
Location: C:¥MyWork¥ra4w1_202110¥sample_appl				Browse	
Choose file system: default \sim					
You can download more Renesas packs here					
0	< <u>B</u> ack	<u>N</u> ext >	Einish	Cancel	

Figure 3-2 New Renesas Executable Project



3. Select the Custom User Board (Any Device) from Board, R7FA4W1AD2CNG from Device.

Renesas RA C/C++ Project			—		×
Renesas RA C/C++ Project Device and Tools Selection					Ź
Device Selection FSP Version: 3.6.0-alpha0+20211230.d166da01 Board: Custom User Board (Any Device) Device: R7FA4W1AD2CNG Language: C C C++	Board Description				
	Device Details				
	TrustZone Pins Processor	No 56 Cortex-M4			
Toolchains	Debugger				
GNU ARM Embedded GCC ARM A-Profile (AArch64 bare-metal) 9.3.1.20200408	J-Link ARM				~
0	< <u>B</u> ack <u>N</u> e	ext >	<u>F</u> inish	Cance	4

Figure 3-3 Project Configuration (Board and Device)

4. When making the MESH application on a BareMetal environment, choose **No RTOS**. When making the application on a FreeRTOS environment, choose **FreeRTOS**.

Renesas RA C/C++ Project		- 🗆 X	Renesas RA C/C++ Project	– 🗆 X
Renesas RA C/C++ Project			Renesas RA C/C++ Project	
Build Artifact and RTOS Selection			Build Artifact and RTOS Selection	1
Build Artifact Selection	RTOS Selection		Build Artifact Selection	RTOS Selection
Executable Project builds to an executable file	No RTOS] ~	Executable Project builds to an executable file	FreeRTOS (v10.4.3-LTS.Patch.2+fsp.3.6.0.alpha0.20211230.d166da01)
Static Library Project builds to a static library file			Static Library Project builds to a static library file	
Executable Using an RA Static Library Project builds to an executable file Project uses an existing RA static library project			Executable Using an RA Static Library Project builds to an executable file Project uses an existing RA static library project	
0	< Back Next >	Einish Cancel	0	< gack Next > Finish Cancel

Figure 3-4 Project Configuration

5. Click **Next** button.



RA4W1 Group

6. When making the MESH application on a BareMetal environment, choose **BareMetal -Minimal**. When making the application on a FreeRTOS environment, choose **FreeRTOS -Minimal- Static Allocation**.

Renesas RA C/C++ Project	×	🐻 Renesas RA C/C++ Project - 🗆 🗙
Renesas RA C/C++ Project Project Template Selection	Ď	Renesas RA C/C++ Project Project Template Selection
Project Template Selection Project Template Selection Bare Metal - Minimal Bare metal FSP project that includes BSP This project will initialize clocks, pins, stacks, and the C runtime environment. [Renesas:RA.3.6.0-alpha0+20211230.d166da01.pack] Code Generation Settings Use: Renesas Code Formatter		Project Remplate Selection Image: Selection Image: Selection Empty FreeRTOS - Minimal - Static Allocation Empty FreeRTOS Far project with no threads. FreeRTOS is pre-configured for static memory allocation. This project will initialize the MCU using the SBR [Reneas RA.3.6.0-alpha0+20211230.d166da01.pack] Code Generation Settings Outse Renease Code Formatter
⑦ <back next=""> Brish Car</back>	ncel	(2) Cancel

Figure 3-5 Project Configuration (Select Template)

7. Click Finish button. After a while, the project will be created

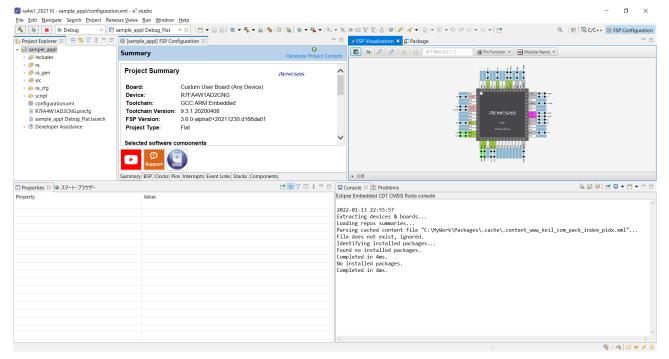


Figure 3-6 Project Overview



3.2 Heap and Stack configuration

To allocate enough memory size to use as the Mesh Stack, set heap and stack configuration as following in [Properties] of [BSP] tab on FSP configuration.

- [RA Common]→[Main stack size (bytes)] : 0x1400
- [RA Common]→[Heap size (bytes)] : 0x1000

Custom	User Board (Any Device)		
Settings	Property V RA Common	Value	1
	Main stack size (bytes) Heap size (bytes)	0x1400 0x1000	
	MCU Vcc (mV) Parameter checking	3300 Disabled	
	Assert Failures Error Log	Return FSP_ERR_ASSERTION No Error Log	
	Clock Registers not Reset Values during Start	Disabled	-
	Main Oscillator Populated PFS Protect	Populated Enabled	
	C Runtime Initialization Early BSP Initialization	Enabled Disabled	
	Main Oscillator Clock Source Subclock Populated	Crystal or Resonator Populated	_
	Subclock Drive (Drive capacitance availabilit	Standard/Normal mode	
	Subclock Stabilization Time (ms)	1000	

Figure 3-7 BSP Configuration

The configuration macros listed in Table 3-1 of BSP are changed by the above configuration.

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

Table 3-1	BSP Configuration and Macro
-----------	------------------------------------

Configuration and Macro	Default Value	Value for Mesh
RA Common > Main stack size (bytes)	0x400	0x1400
(BSP_CFG_STACK_MAIN_BYTES)		
RA Common > Heap size (bytes)	0	0x1000
(BSP_CFG_HEAP_BYTES)		



3.3 Clocks configuration

In [Clocks] tab of the FSP Configuration, select the clocks and set their clock frequency. To use the Mesh Stack, the following settings are required.

- System Clock (ICLK): 8MHz or over
- Peripheral module Clock A (PCLKA): 8MHz or over

Bluetooth LE Stack is optimized for the case that the clock frequency of both the ICLK and the PCLKA is 32MHz. Thus, it is recommended to set the clock configuration in which the clock frequency of both the ICLK and the PCLKA become 32MHz.

*[sample_appl] FSP Configuration ×		
Clocks Configuration		O Generate Project Content
		Restore Defaults
XTAL 8MHz	→ ICLK Div /1 v -	ICLK 32MHz
> PLL Src: XTAL	PCLKA Div /1 ∨ -	PCLKA 32MHz
PLL Div /2 ~	→ PCLKB Div /1 ~ -	→ PCLKB 32MHz
PLL Mul x12 ~	PCLKC Div /1	→ PCLKC 32MHz
PLL 48MHz	↔ PCLKD Div /1 ~	→ PCLKD 32MHz
	⇒BCLK Div /64 ~	→ BCLK 500kHz
HOCO 32MHz V CLKOUT Disabled V	v →No output v	→ BCLKout 0Hz
LOCO 32768Hz	→ FCLK Div /2 ~ -	→ FCLK 16MHz
MOCO 8MHz		
SUBCLK 32768Hz		→ USBCLK 48MHz
Summary BSP Clocks Pins Interrupts Event Links Stacks Components		

Figure 3-8 Clocks Configuration



3.4 Add and configure MESH Stack

This section describes how to add / configure the MESH Stack into the MESH application. Click **configuration.xml** in the project and add / configure the MESH Stack in the [**Stacks**] tab on the FSP Configuration. The procedure about adding the MESH Stack is different for the BareMetal and the FreeRTOS environment. Section 3.4.1 describes the procedure for the BareMetal environment. Section 3.4.2 describes the procedure for the FreeRTOS environment. The MESH Stack configuration is common to the 7BareMetal and the FreeRTOS environment. The configuration is described in detail in section 3.4.3.

3.4.1 Add MESH Stack in BareMetal environment

Click New Stack and add Networking → BLE Mesh Bearer Platform
 (rm_ble_mesh_bearer_platform) to HAL/Common. This driver includes some peripheral driver. The
 configuration for these peripherals are described in section 3.5.

*[sample_appl] FSP Configuration	n ×	•	BLE Mesh Access (rm_ble_mesh_access)		- [
		•	BLE Mesh Bearer (rm_ble_mesh_bearer)		0
Stacks Configuration		\oplus	BLE Mesh Bearer Platform (rm_ble_mesh_bearer_platform)	Generate Pro	
		÷	BLE Mesh Bearer Platform (rm_mesh_bearer_platform)		
Threads	🔊 New Thread 😰 Remove 😑 🛛		BLE Mesh Config Client (rm_mesh_config_clt)	Stack > 🐣 Extend Stack >	Remove
Threads			Bit Meis Colling, Liener, Ching, Louis, Guo Bit Meis Colling, Liener, Ching, Guo Bit Meis Lower, Tans (im. bit-meist-Jower, Irans) Bit Meish Nodel Generic Admin Poperty Server (im. meist-generic, Jatriery, ct) Bit Meish Model Generic Battery Server (im. meist-generic, Dattery, ct) Bit Meish Model Generic Diether Streation (im. meist-generic, Calient, prop. srv) Bit Meish Model Generic Diether Streation (im. meist-generic, Calient, prop. srv) Bit Meish Model Generic Default Transition Time Client (im. meist-generic, ctl, ct) Bit Meish Model Generic Default Transition Time Server (im. meist-generic, ctl, ct) Bit Meish Model Generic Level Server (im. meist-generic, level, srv) Bit Meish Model Generic Level Server (im. meist-generic, level, srv) Bit Meish Model Generic Level Server (im. meist-generic, level, srv) Bit Meish Model Generic Level Server (im. meist-generic, level, srv) Bit Meish Model Generic Level Server (im. meist-generic, level, srv) Bit Meish Model Generic Location Client (im. meist-generic, level, srv) Bit Meish Model Generic Location Client (im. meist-generic, level, srv) Bit Meish Model Generic Level Server (im. meist-generic, level, srv) Bit Meish Model Generic Level Server (im. meist-generic, level, srv) Bit Meish Model Generic Level Server (im. meist-generic, level, srv) Bit Meish Model Generic Location Client (im. generic, meric, level, srv) Bit Meish Model Generic Level Server (im. meist-generic, level, srv) Bit Meish Model Generic Location Client (im. generic, meric, generic, meric, srv) Bit Meish Model Generic Location Client (im. generic, meric, generic, meric, srv) Bit Meish Model Generic Location Client (im. generic, generic, level, srv)	Stack = Extend Stack - Analog Antificial Intelligence Audio Bootloader CapTouch Connectivity DSP Graphics Input Montoring Motor Networking Power	KC Remove
			BLE Mesh Model Generic On Off Server (m. mesh.generic.on.off.srv) BLE Mesh Model Generic Power Level Client (m. mesh.generic.pl.ct) BLE Mesh Model Generic Power Level Server (m. mesh.generic.pl.srv) BLE Mesh Model Generic Power On Off Client (m. mesh.generic.poo.clt)	Security Sensor Storage	>
Objects	New Object > Remove	$\oplus \oplus \oplus \oplus \oplus \oplus \oplus \oplus \oplus \oplus$	BLE Mesh Model Generic Rower On Off Server (m_mneth_generic_poo_sm) BLE Mesh Model Generic Roperty (liner (m_mneth_generic_prog_c1) BLE Mesh Model Health Client (m_meth_generic_user_prop_sr) BLE Mesh Model Health Client (m_meth_health_sr) BLE Mesh Model Light Control Generic (m_mneth_glight_st]. BLE Mesh Model Light Control Server (m_mneth_glight_st]. BLE Mesh Model Light Control Server (m_meth_glight_st]. BLE Mesh Model Light HSL Client (m_meth_glight_st]. BLE Mesh Model Light HSL Server (m_meth_glight_st]. BLE Mesh Model Light HSL Server (m_mth_st]. BLE Mesh Model Light HSL Server (m_mth	System Timers Transfer Search	> >
Summary BSP Clocks Pins Interrup	ts Event Links Stacks Components		BLE Mesh Model Light Lightness Controller Client (rm_mesh_light_lc_clt)		

Figure 3-9 Add Bluetooth Bearer

2. Click Add BLE Mesh OS Module box and select New→BLE Mesh OS on Baremetal (rm_mesh_os_baremetal).

*[sample_appl] FSP Configuration ×	
Stacks Configuration	O Generate Project Content
Threads 🗟 New Thread 🗟 Remove 🕒	HAL/Common Stacks 🚯 New Stack > 🖄 Extend Stack > 🔊 Remove
 	BLE Mesh Bearer Platform (m_mesh_bearer_platform)
	BLE Mesh Bearer (m_ble_mesh_bearer)
	BLE Mesh (m_ble_mesh)
	Add BLE Mesh OS Module Să Add BLE Mesh Timer Module Să Add Timer Driver Module # BLE Driver (r_ble_ext (r_flash_lp)
Objects 🐑 New Object > 🛍 Remove	0
	New > BLE Mesh OS on Baremetal (m_mesh_os_baremetal) BLE Mesh OS on FreeRTOS (m_mesh_os_freertos) G
Summary BSP Clocks Pins Interrupts Event Links @ Stacks Compon	

Figure 3-10 Add OS



3. Click Add BLE Mesh Timer Module box and select New→BLE Mesh Timer on Baremetal (rm_mesh_timer_baremetal).

*[sample_appl] FSP Configuration ×		• •
Stacks Configuration		Generate Project Content
Threads 🛞 New Thr	ead 🔊 Remove 🕒	HAL/Common Stacks 🚯 New Stack > 🚖 Extend Stack > 🛞 Remove
 ✓ BHAL/Common ④ g.joport I/O Port (r_joport) BLE Mesh Bearer Platform (rm_mesh_b) 	earer_platform)	BLE Mesh Bearer Platform (m_mesh_bearer_platform)
		BLE Mesh Bearer (m_ble_mesh_bearer)
		BLE Mesh (m_ble_mesh)
Objects	bject > 🔊 Remove	Image: Constraint of the second sec
	_	a) New > # BLE Mesh Timer on Baremetal (m_mesh_timer_baremetal) BLE Mesh Timer on FreeRTOS (m_mesh_timer_freetto))
		(r_flash.jp)
Common DCD Clocks Diss Interrupts Const Links		

Figure 3-11 Add Timer

4. Click Add BLE Network Driver box and select New→BLE Driver (r_ble_extended).

🌻 *[new] FSP Configuration ×	- 0
Stacks Configuration	O Generate Project Content
Threads New Thread Remove	BLE Mesh Bearer Platform (m.,mesh_bearer_platform) Stacks 🚯 New Stack > 🖄 Extend Stack > 🔊 Remove
I g_ioport I/O Port (r_ioport)	BLE Mesh Bearer Platform (rm.,mesh,bearer_platform)
	BLE Mesh Bearer (rm_ble_mesh_bearer) Add BLE Network Driver
	⊕ BLE Driver (r_ble_balance) ● BLE Driver (r_ble_compact)
	BLE Mesh (rm_ble_mesh) BLE Driver (r_ble_extended)
Objects	0
Upjecs I new upject > ist nemove	BLE Mesh OS on Baremetal Baremetal Baremetal Baremetal
	(m_mesh_os_baremet () al) (0 metal) (0 metal)
	(
Summary BSP Clocks Pins Interrupts Event Links @ Stacks Components	

Figure 3-12 Add Bluetooth LE Stack



5. Click **New Stack** and add the required model for your mesh application to **HAL/Common**. For example, if you want to use the Generic On Off Server model, choose **Networking→BLE Mesh Model Generc On Off Server**.

🔅 [sample_appl] FSP Co	nfiguration ×	0	BLE Mesh Bearer Platform (rm_ble_mesh_bearer_platform)		
		+	BLE Mesh Config Client (rm_mesh_config_clt)	0	
Stacks Configura	tion	+	BLE Mesh Config Server (rm_mesh_config_srv)	Generate Pro	ect Content
		\$	BLE Mesh Lower Trans (rm_ble_mesh_lower_trans)		
Threads	🕢 New Thread 🚯 Remove 📄	н/ 🕈	BLE Mesh Model Generic Admin Property Server (rm_mesh_generic_admin_prop_srv)	tack s Pyteod Stack s	Remove
✓ → HAL/Common		•	BLE Mesh Model Generic Battery Client (rm_mesh_generic_battery_clt)	Analog	
	2	. +	BLE Mesh Model Generic Battery Server (rm_mesh_generic_battery_srv)	Artificial Intelligence	> 1
∉ g_ioport I/O P	rort (r_loport) arer Platform (rm mesh bearer platform)	+	BLE Mesh Model Generic Client Property Server (rm_mesh_generic_client_prop_srv)	Audio	>
BLE Mesh Bea	arer Platform (rm_mesn_bearer_platform)	\$	BLE Mesh Model Generic Default Transition Time Client (rm_mesh_generic_dtt_clt)	Bootloader	>
		•	BLE Mesh Model Generic Default Transition Time Server (rm_mesh_generic_dtt_srv)	CapTouch	>
		•	BLE Mesh Model Generic Level Client (rm_mesh_generic_level_clt)	Connectivity	, E
			BLE Mesh Model Generic Level Server (rm_mesh_generic_level_srv)	DSP	, F
		+	BLE Mesh Model Generic Location Client (rm_mesh_generic_loc_clt)		
		+	BLE Mesh Model Generic Location Server (rm_mesh_generic_loc_srv)	Graphics	,
		•	BLE Mesh Model Generic Manufacturer Property Server (rm_mesh_generic_mfr_prop_srv)	Input	>
		E 🚸	BLE Mesh Model Generic On Off Client (rm_mesh_generic_on_off_clt)	Monitoring	> =
		+	BLE Mesh Model Generic On Off Server (rm_mesh_generic_on_off_srv)	Motor	>
		•	BLE Mesh Model Generic Power Level Client (rm_mesh_generic_pl_clt)	Networking	>
		•	BLE Mesh Model Generic Power Level Server (rm_mesh_generic_pl_srv)	Power	>
		•	BLE Mesh Model Generic Power On Off Client (rm_mesh_generic_poo_clt)	Security	>
		E 🕸	BLE Mesh Model Generic Power On Off Server (rm_mesh_generic_poo_srv)	Sensor	, -
		_ [. ⊕	BLE Mesh Model Generic Property Client (rm_mesh_generic_prop_clt)	Storage	> .
		+	BLE Mesh Model Generic User Property Server (rm_mesh_generic_user_prop_srv)	System	· · ·
Objects	New Object > Remove	+	BLE Mesh Model Health Client (rm_mesh_health_clt)	Timers	
		•	BLE Mesh Model Health Server (rm_mesh_health_srv)		
		E 🚸	BLE Mesh Model Light Control Client (rm_mesh_light_ctl_clt)	Transfer	
		+	BLE Mesh Model Light Control Server (rm_mesh_light_ctl_srv)	🛷 Search	F
		+	BLE Mesh Model Light HSL Client (rm_mesh_light_hsl_clt)	(r flash I	p)
		+	BLE Mesh Model Light HSL Server (rm_rm_mesh_light_hsl_srv)		
		•	BLE Mesh Model Light Lightness Client (rm_mesh_light_lightness_clt)	1	
		+	BLE Mesh Model Light Lightness Controller Client (rm_mesh_light_lc_clt)		
		< 🕈	BLE Mesh Model Light Lightness Controller Server (rm_rm_mesh_light_lc_srv)		>
DCD Classic D		+	BLE Mesh Model Light Lightness Server (rm_mesh_light_lightness_srv)		
ummary BSP Clocks Pi	ins Interrupts Event Links 3 Stacks Compone	nts 🕀	BLE Mesh Model Liaht XYL Client (rm mesh liaht xvl clt)		

Figure 3-13. Add Mesh Model

NOTE: If you are adding a second or subsequent model to the same element, click the Add BLE Mesh Access Module box and select Use \rightarrow g_rm_ble_access0 BLE Mesh Access (rm_ble_mesh_access).

# *[new] FSP Configuration ×		- 0
Stacks Configuration		Generate Project Content
Threads 🔊 New Thread 🔊 Remove 🕒	BLE Mesh Config Server (rm_mesh_config_srv) Stacks	🕥 New Stack > 🔺 Extend Stack > 🖄 Remove
 	BLE Mesh Config Server (m,meh_config.srv) Add BLE Mesh Access Module New Use g.m.ble_mesh_access0 BLE Mesh	Access (m_ble_mesh_access)
Objects 🐑 New Object > 🐑 Remove		
Summary BSP Clocks Pins Interrupts Event Links @ Stacks Components	8	



3.4.2 Add MESH Stack in FreeRTOS environment

1. Click **New Thread** on the Threads area and add a New Thread. In this example, the New Thread is named BLE_CORE_TASK.

	De_appl] FSP Configuration ⊠ Configuration		Generate Project Content
Threads	🔄 New Thread 🕫 Remove 😑	BLE_CORE_TASK Stacks	New Stack > A Extend Stack > Remove
4	AL/Common g_ioport I/O Port (r_ioport) FreeRTOS Port (rm_freertos_port) LE_CORE_TASK	Add stacks to the selected the pasting here from the clipbo	hread by using the 'New Stack >' toolbar button (above), or by pard.
Objects	New Object > 紀 Remove		
		Stacks Components	
ummary	BSP Clocks Pins Interrupts Event Links	Stacks Components	
ummary I Proper	BSP Clocks Pins Interrupts Event Links ties × 争 スマート・ブラウザー	Stacks Components	
ummary Proper LE_COI	BSP Clocks Pins Interrupts Event Links t ties × 粂スマート・ブラウザー RE_TASK		
Proper	BSP Clocks Pins Interrupts Event Links ties × 傘 スマート・ブラウザー RE_TASK Property	Stacks Components	
ummary Proper LE_COI	BSP Clocks Pins Interrupts Event Links ties ×	Value	
ummary Proper LE_COI	BSP Clocks Pins Interrupts Event Links ties × 傘 スマート・ブラウザー RE_TASK Property		
ummary Proper LE_COI	BSP Clocks Pins Interrupts Event Links ties × 〜 スマート・ブラウザー RE_TASK Property × Thread Symbol	Value ble_core_task	
ummary Proper LE_COI	BSP Clocks Pins Interrupts Event Links ties × ゆ スマート・ブラウザー RE_TASK Property × Thread Symbol Name	Value ble_core_task BLE_CORE_TASK	
ummary Proper LE_COI	BSP Clocks Pins Interrupts Event Links : ties × ゆ スマート・ブラウザー RE_TASK Property > Thread Symbol Name Stack size (bytes)	Value ble_core_task BLE_CORE_TASK 1024	
ummary Proper LE_COI	BSP Clocks Pins Interrupts Event Links ties × 粂スマート・ブラウザー RE_TASK Property × Thread Symbol Name Stack size (bytes) Priority	Value ble_core_task BLE_CORE_TASK 1024 1	

Figure 3-14 Add BLE_CORE_TASK

2. Change Stack size to 0x3000[bytes]. Change Priority to 2.

BLE_COF	RE_TASK	
Settings	Property ✓ Thread	Value
	Symbol	ble_core_task
	Name	BLE CORE TASK
	Stack size (bytes)	0x3000
	Priority	2
	Thread Context	NULL
	Memory Allocation	Static
	Allocate Secure Context	Enable

Number bytes in the stack memor...st be an integer multiple of 8.

Figure 3-15 Stack size and Priority of BLE_CORE_TASK

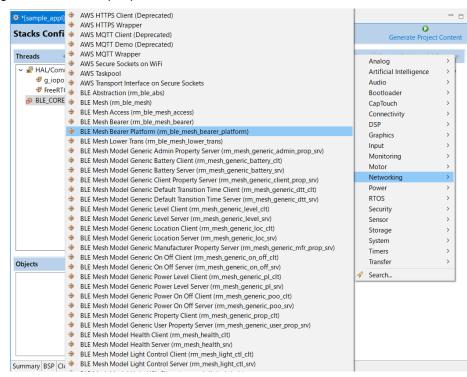


3. Change FreeRTOS configurations as the following in BLE_CORE_TASK **Properties** tab.

Table 3-2 FreeRTOS Configuration and Macro

Configuration and Macro	Changed Value	Default Value
Common > General > Minimal Stack Size (configMINIMAL_STACK_SIZE)	1024	128
Common > General > Use Mutexes (configUSE_MUTEXES)	Enabled	Disabled
Common > General > Use Recursive Mutexes (configUSE_RECURSIVE_MUTEXES)	Enabled	Disabled
Common > General > Enable Backward Compatibility (configENABLE_BACKWARD_COMPATIBILITY)	Enabled	Disabled
Common > Memory Allocation > Support Dynamic Allocation (configSUPPORT_DYNAMIC_ALLOCATION)	Enabled	Disabled
Common > Memory Allocation > Total Heap Size (configTOTAL_HEAP_SIZE)	11264	0
Common > Timers > Timer Queue Length (configTIMER_QUEUE_LENGTH)	32	10
Common > Optional Functions > <i>xTimerPendFunctoinCall()</i> Function (INCLUDE_xTimerPendFunctionCall)	Enabled	Disabled

 Click New Stack and add Networking → BLE Mesh Bearer Platform (rm_ble_mesh_bearer_platform) to BLE_CORE_TASK. This driver includes some peripheral driver. The configuration for these peripherals are described in section 3.5.







6. Click Add BLE Mesh OS Module box and select New→BLE Mesh OS on FreeRTOS (rm_mesh_os_freertos).

*[sample_freertos] FSP Configuration ×	- 8
Stacks Configuration	Generate Project Content
Threads 💿 New Thread 🔞 Remove 😑	BLE_CORE_TASK Stacks 🕢 New Stack > 😤 Extend Stack > 🛞 Remove
 ✓ Mather Alexand Antipart of the second seco	BLE Mesh Bearer Platform (rm_mesh_bearer_platform)
BLE Mesh Bearer Platform (rm_mesh_bearer_platform)	
	BLE Mesh Bearer (rm_ble_mesh_bearer) BLE Mesh (rm_ble_mesh)
Objects 🔄 New Object > 🏠 Remove	
	Add BLE Mesh OS Add BLE Mesh Timer Module Add Timer Driver Module
Summary BSP Clocks Pins Interrupts Event Links Stacks Comp	

Figure 3-17 Add OS

7. Click Add BLE Mesh Timer Module box and select New→BLE Mesh Timer on FreeRTOS (rm_mesh_timer_freertos).

*[sample_freertos] FSP Configuration ×	
Stacks Configuration	Generate Project Content
Threads 💿 New Thread 🔹 Remove 🕞	BLE_CORE_TASK Stacks 🔊 New Stack > 🏦 Extend Stack > 🖈 Remove
 ✓ ✓ IAL/Common <i>I</i> g_ioport I/O Port (r_ioport) <i>I</i> FreeRTOS Port (rm_freertos_port) ✓ <i>P</i> BLE_CORE_TASK 	BLE Mesh Bearer Platform (rm_mesh_bearer_platform)
BLE Mesh Bearer Platform (rm_mesh_bearer_platform)	
	BLE Mesh Bearer (rm_ble_mesh_bearer) BLE Mesh (rm_ble_mesh)
Objects 💿 New Object > 🛍 Remove	
	New Image: BLE Mesh Timer on Baremetal (rm_mesh_timer_baremetal) Image: BLE Mesh Timer on FreeRTOS (rm mesh timer freertos)
	<
Summary BSP Clocks Pins Interrupts Event Links @ Stacks Comp	oonents

Figure 3-18 Add Timer



7. Click Add BLE Network Driver box and select New→BLE Driver (r_ble_extended_freertos).

Stacks Configuration Threads New Thread Remove BLE CORE TASK Stacks	Generate Project Content New Stack > ≜ Extend Stack > ℝ Remove
Threads New Thread Premove DIE CORE TASK Stacks	A New Grades & Entered Creater B Deserves
	Mew Stack > == Extend Stack > M. Remove
✓ Instruction	
BLE Mesh Bearer Platform (rm_mesh_bearer_platform)	
sh Bearer (rm_ble_mesh_bearer)	
Objects New Object > 🔊 Remove	
sh OS on DS sh_os_freetos)	

Figure 3-19 Add Bluetooth LE Stack

8. Click **New Stack** and add the required model for your mesh application to **BLE_CORE_TASK**. For example, if you want to use Generic On Off Server model, choose **Networking→BLE Mesh Model Generic On Off Server**.

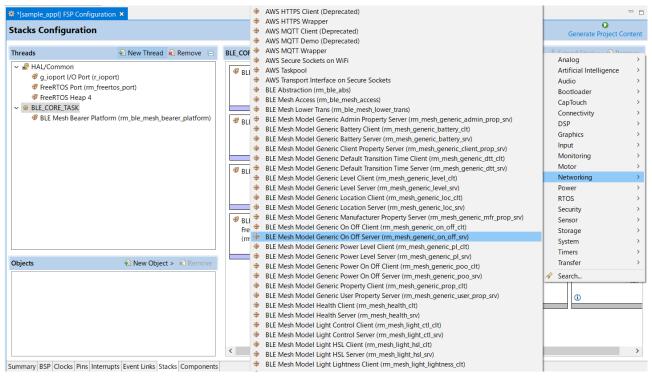


Figure 3-20 Add Mesh Model

NOTE: If you are adding a second or subsequent model to the same element, click the Add BLE Mesh Access Module box and select Use \rightarrow g_rm_ble_access0 BLE Mesh Access (rm_ble_mesh_access).



9. Add **RTOS→FreeRTOS Heap4** to **HAL/Common**.

🌣 *[sample_appl] FSP Configuration 🗙		c
Stacks Configuration		Generate Project Con
Threads New Thread Remove Image: Comparison of the second	HAL/Common Stacks Gioport I/O Port (r_ioport) G G G HereRTOS Port (rm_freertos_port) G	New Crack Carlo Charles Permeters Analog Artificial Intelligence Audio Bootloader CapTouch Connectivity DSP Graphics Input Monitoring Motor Networking Power Networking
Objects New Object > Remove	 ➡ FreeRTOS Buffer Alloc ➡ FreeRTOS Heap 1 ➡ FreeRTOS Heap 2 ➡ FreeRTOS Heap 3 ➡ FreeRTOS Heap 4 ➡ FreeRTOS Heap 5 	ation 2 RTOS Security Sensor Storage System Timers Transfer P Search

Figure 3-21 Add Heap4



3.4.3 Configure Mesh Stack

This section describes the configurations required for the e BLE Mesh Network Middleware on rm_ble_mesh module.

Select [BLE Mesh Bearer Platform (rm_mesh_bearer_platform)] on the [Stacks] tab and change the properties as shown in Table 3-3. For details on each property value, refer to "Renesas Flexible Software Package (FSP) User's Manual (R11UM0155)".

Table 3-3 Mesh Stack Configuration and Macro

Configuration and Macro	Default Value	Value for Mesh
Maximum number of connections	7	1
(MESH_BEARER_PLATFORM_CFG_RF_CONNECTION_MAXIMUM)		
Maximum advertising data length	1650	31
(MESH_BEARER_PLATFORM_CFG_RF_ADVERTISING_DATA_MAXIMUM)		
Maximum advertising set number	4	1
(MESH_BEARER_PLATFORM_CFG_RF_ADVERTISING_SET_MAXIMUM)		
Maximum periodic sync set number	2	1
(MESH_BEARER_PLATFORM_CFG_RF_SYNC_SET_MAXIMUM)		

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.

BLE Mes	h Bearer Platform (rm_mesh_bearer_j	olatform)
Settinas	Property	Value
	✓ Common	
	Debug Public Address	FF:FF:FF:50:90:74
	Debug Random Address	FF:FF:FF:FF:FF:FF
	Maximum number of connections	1
	Maximum connection data length	251
	Maximum advertising data length	31
	Maximum advertising set number	1
	Maximum periodic sync set number.	1
	Store Security Data	Disable
	Data Flash Block for Security Data	0
	Remote Device Bonding Number	7
	Connection Event Start Notify	Disable
	Connection Event Close Notify	Disable
	Advertising Event Start Notify	Disable
	Advertising Event Close Notify	Disable
	Scanning Event Start Notify	Disable
	Scanning Event Close Notify	Disable
	Initiating Event Start Notify	Disable
	Initiating Event Close Notify	Disable
	RF Deep Sleep Start Notify	Disable
	RF Deep Sleep Wakeup Notify	Disable
	Bluetooth dedicated clock	6

Figure 3-22 Mesh Stack Configuration (1)

NOTE: Set the same values in the properties of the BLE Driver box as well.



RA4W1 Group

Select [BLE Mesh (rm_ble_mesh)] on the [Stacks] tab and change the properties as shown in Table 3-4. For details on each property value, refer to "Renesas Flexible Software Package (FSP) User's Manual (R11UM0155)".

Table 3-4 Mesh Stack Configuration

Configuration	Default Value	Value for Mesh
Storage→Block Number	1	6
Memory Pool→Memory Pool Size	0x4000	0x3000
The number of Data Flash Blocks used for storing mesh information		
MIN: 1		
MAX: 8		

Set "6" to [Block number], "0x3000" to [Memory pool size].

🔲 Prope	erties ×	🎋 Debug	🔗 Search	📱 Stack Analysis	🔋 Memory Usage
BLE Me	esh (rm	_ble_mes	h)		
Settings	Prop	erty			Value
	> Co	ommon			
	~ M	odule BLE M	/lesh (rm_b	le_mesh)	
	>	General			
	~	Storage			
		Block N	lumber		6
	~	Memory P	ool		
		Memor	y Pool Size		0x3000
	>	Logging			
		Channel N	umber		0
		Network Ir	nterface Nu	mber	2
		Provisionir	ng Interface	Number	2
		Network C	ache Size		10
		Network S	equence N	umber Cache Size	32
		Maximum	Number of	Subnet	4
		Maximum	Number of	Device Key	4
		Proxy Filte	r List Size		2
		Maximum	Number of	LPN	1
		Reassemb	led Cache S	lize	8
		Maximum	Number of	Segmentation an	8
		Maximum	Number of	Friend Message C	15
		Maximum	Number of	Friend Subscriptio	8
		Maximum	Number of	Element	4
		Block num	ber.		

Figure 3-23 Mesh Stack Configuration (2)



3.4.4 Oher Mesh Stack configuration

The Mesh Stack has parameters that can be changed depends on each mesh network scale and each requirement for node. These parameters can be set with FSP Configuration and are reflected in "common_data.c" when generating code.

Configuration and Variable name	Description
Bearer→Network Interface Number	The number of bearers used for Mesh Network
(network_interfaces_num)	MIN: 1
*Default: 2	MAX: (1 +
	MESH BEARER PLATFORM CFG RF CONNECTIO
	N_MAXIMUM)
	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.
Bearer→Provisioning Interface Number	The number of bearers used for Provisioning
(provisioning_interfaces_num)	MIN: 1
*Default: 2	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.
Provisioning \rightarrow Unprovisioned Device Beacon Timeout in	Transmission interval of Unprovisioned Device Beacon
Milliseconds	[msec]
(unprov_device_beacon_timeout)	MIN: 20
*Default: 200	
	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.
Network→Network Cache Size	The maximum number of nodes that Network Message
(network_cache_size)	Cache can store
*Default: 10	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.
Network → Network Sequence Number Cache Size	The number of SEQ number that Network Message
(network_sequence_num_cache_size)	Cache can store for each node
*Default: 32	MIN: 32
Network→Maximum Number of Subnet	Maximum number of subnet information such as
(maximum_subnets)	Network Key and NID
*Default: 4	MIN: 1
Network→Maximum Number of Device Key	Maximum number of Device Key
(maximum_device_keys)	MIN: 1
*Default: 4	
	When Configuration Client Model is not used, it is
	enough to set this configuration to 1.

Table 3-5 BLE Mesh (rm_ble_mesh) Configuration and Variable name



Network → Proxy Filter List Size	Maximum number of addresses that can be added to
(proxy_filter_list_size)	each Proxy List
*Default: 2	MIN: 1
Network → Network Sequence Number Block Size	Distance between SEQ number for writing to Data
(net sequence number block size)	Flash memory
*Default: 2048	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.
Network → Network Transmit Count for Network Packets	Default value of Network Transmit Count state
(net_tx_count)	MIN: 0
*Default: 1	MAX: 7
Network → Network Interval Steps for Network Packets	Default value of Network Transmit Interval Steps state
(net_tx_interval_steps)	MIN: 0
*Default: 4	MAX: 31
Network→Network Transmit Count for Relayed Packets	Default value of Relay Retransmit Count state
(net_relay_tx_count)	MIN: 0
*Default: 0	MAX: 7
Network→Network Interval Steps for Relayed Packets	Default value of Relay Retransmit Interval Steps state
(net_relay_tx_interval_steps)	MIN: 0
*Default: 9	MAX: 31
Network→Proxy ADV Network ID Timeout for Each Subnet	Transmission interval of Proxy Advertisement with
in Milliseconds	Network ID [msec]
(proxy_subnet_netid_adv_timeout)	MIN: 20
*Default: 100	
Network→Proxy ADV Node Identity Timeout for Each	Transmission interval of Proxy Advertisement with
Subnet in Milliseconds	Node Identity [msec]
(proxy_subnet_nodeid_adv_timeout)	MIN: 20
*Default: 300	
Network→Proxy ADV Node Identity Overall Time Period in	Transmission period of Proxy Advertisement with Node
Milliseconds	Identity [sec]
(proxy_nodeid_adv_timeout)	MIN: 1
*Default: 60	
Network→Maximum Number of Queued Messages for	Size of transmission queue for Network PDUs
Transmission	MIN: 2
(net_tx_queue_size)	
*Default: 64	
Transport→Maximum Number of LPN	Maximum number of Low Power Nodes that Friend
(maximum_lpn)	Node can establish Friendship with
*Default: 1	MIN: 1
Transport→Replay Protection Cache Size	Size of Replay Protection Cache
(replay_cache_size)	MIN: 2
*Default: 10	
Transport→Reassembled Cache Size	Size of reception message cache of Segmentation and
(reassembled_cache_size)	Reassembly (SAR)
*Default: 8	MIN: 2



Transport→Friend Poll Retry Count (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
Transport→Maximum Number of Segmentation and Reassembly (maximum_ltrn_sar_context) *Default: 8	The number of contexts of Segmentation and Reassembly (SAR) mechanism used for transmitting and receiving Segmented Message MIN: 2
Transport→Lower Transport Segment Transmission Timeout in Milliseconds (ltrn_rtx_timeout) *Default: 300	Retransmission interval of segmented message [msec] MIN: 200
Transport→Lower Transport Segment Re-Transmission Count (ltrn_rtx_count) *Default: 2	The number of times to retransmit segmented message MIN: 2 MAX: 255
Transport→Lower Transport Acknowledgement Timeout in Milliseconds (ltrn_ack_timeout) *Default: 200	Transmission interval of Segmented Acknowledgement message [msec] MIN: 200
Transport→Lower Transport Incomplete Timeout in Milliseconds (ltrn_incomplete_timeout) *Default: 20	Cancel timeout time of receiving segmented message [sec] MIN: 10
Transport→Friendship Receive Window (frnd_receive_window) *Default: 100	Reception windows size of Low Power Node [msec] MIN: 100 MAX: 255
Transport→Maximum Number of Friend Message Queue (maximum_friend_message_queue) *Default: 15	Size of Message Queues for each Low Power Node MIN: 2
Transport→Maximum Number of Friend Subscription List (maximum_friend_subscription_list) *Default: 8	Maximum number of Friend Subscription Lists for each Low Power Node MIN: 1
Transport→Friend Clear Confirmation Timeout in Milliseconds (lpn_clear_retry_timeout_initial) *Default: 1000	Retransmission interval of Friend Clear message when Low Power Node does not receive Friend Clear Confirmation message MIN: 1000
Transport→Friend Clear Retry Count (lpn_clear_retry_count) *Default: 5	The number of times to retry Friend Clear message when Low Power Node does not receive Friend Clear Confirmation message MIN: 1
Transport→Friendship Retry Timeout in Milliseconds (trn_frndreq_retry_timeout) *Default: 1200	Transmission period of Friend Request message [msec] MIN: 1100
Access→Maximum Number of Element (maximum_access_element_num) *Default: 4	Maximum number of Elements MIN: 1
Access-→Maximum Number of Model (maximum_access_model_num) *Default: 60	Maximum number of Models MIN: 1
Access-→Maximum Number of Application (maximum_application) *Default: 8	Maximum number of Application Keys MIN: 1
Access→Maximum Number of Virtual Address (maximum_virtual_address) *Default: 8	Maximum number of Virtual Address MIN: 1



RA4W1 Group

Access→Maximum Number of Non-Virtual Address	Maximum number of Non-virtual Address (Unicast	
(maximum_non_virtual_address)	Address or Group Address) MIN: 1	
*Default: 8	The number of State Transition Timers for models	
Access→Maximum Number of Transition Timers		
(max_num_transition_timers)	MIN: 1	
*Default: 5		
Access→Maximum Number of Periodic Step Timers	The number of Periodic Publication Timers for models	
(max_num_periodic_step_timers)	MIN: 1	
*Default: 5		
Foundation→Config Server Secure Network Beacon	Transmission Interval of Secure Network Beacon [sec]	
Interval	MIN: 10	
(config_server_snb_timeout)	MAX: 600	
*Default: 10		
Foundation→Maximum Number of Health Server Instance	Maximum number of Health Server Model	
(maximum_health_server_num)	MIN: 1	
*Default: 2		
Model→Maximum Number of Light Lightness Controller	Maximum number of Light Lightness Controller Server	
Server Instance	Model	
(maximum_light_lc_server_num)	MIN: 1	
*Default: 1		
ID→Company ID	Company ID registered with Bluetooth SIG	
(default_company_id)	MIN: 0x0000	
*Default: 0x0036	MAX: 0xFFFF	
ID→Product ID	Product ID assigned by vendor	
(default_product_id)	MIN: 0x0000	
*Default: 0x0001	MAX: 0xFFFF	
ID→Vendor ID	Product Version ID assigned by vendor	
(default vendor id)	MIN: 0x0000	
*Default: 0x0100	MAX: 0xFFFF	
logging→Packet Bitfield	Get packet-related log	
(p_logging_cfg->packet_bitfield)	-	
*Default: 0		
logging→Module Info Bitfield	Get module-related log	
(p_logging_cfg->module_info_bitfield)		
*Default: 0		
logging→Generic Log Bitfield	Get generic log	
(p logging cfg->generic log bitfield)		
*Default: 0		
logging→function	Callback when logging	
(p_logging_cfg->p_logging_func)		
*Default: logging function		
	1	

Table 3-6 BLE Mesh Provision (rm_ble_mesh_provision) Configuration and Variable name

Configuration and Variable name	Description
Provision Capabilities→Number of Elements	For detail, refer to Mesh Profile Specification.
(num_elements)	
*Default: 1	
Provision Capabilities→Supported Algorithms	For detail, refer to Mesh Profile Specification.
(supported_algorithms)	
*Default: 1	
Provision Capabilities→Public Key Type	For detail, refer to Mesh Profile Specification.
(supported_pubkey)	
*Default: 1	
Provision Capabilities→Static OOB Type	For detail, refer to Mesh Profile Specification.
(supported_soob)	
*Default: 1	



RA4W1 Group

Provision Capabilities→Output OOB Action	For detail, refer to Mesh Profile Specification.
(output_oob.action)	
*Default: 0x1F	
Provision Capabilities→Output OOB Size	For detail, refer to Mesh Profile Specification.
(output_oob.size)	
*Default: 0x08	
Provision Capabilities→Input OOB Action	For detail, refer to Mesh Profile Specification.
(input_oob.action)	
*Default: 0x0F	
Provision Capabilities→Input OOB Size	For detail, refer to Mesh Profile Specification.
(input_oob.size)	
*Default: 0x04	
Provision Callback	Callback from Provisioning process
(p_callback)	
*Default: NULL	

Table 3-7 BLE Mesh Network (rm_ble_mesh_network) Configuration and Variable name

Configuration and Variable name	Description
Callback	Callback from Network process
(p_callback)	
*Default: 1	

Table 3-8 BLE Mesh Upper Trans (rm_ble_mesh_upper_trans) Configuration and Variable name

Configuration and Variable name	Description	
Callback (p_callback) *Default: NULL	Callback Upper Transport process	

Table 3-9 BLE Mesh Access (rm_ble_mesh_access) Configuration and Variable name

Configuration and Variable name	Description
Location Descriptor	where the element is placed
(p_element_descriptor->loc)	
*Default: 0	
Element Number	The number to identify the element
(element_number)	
*Default: 0	

Table 3-10 BLE Mesh Bearer Platform (rm_mesh_bearer_platform) Configuration and Variable name

Configuration and Variable name	Description	
Device Address Type	BD address type	
(device_address_type)	0: Public address	
*Default: 1	1: Random address	
GATT Server Callback Number	The number of callbacks to be registered	
(gatt_server_callback_num)	MIN: 1	
*Default: 15	MAX: 15	
GATT Client Callback Number	The number of callbacks to be registered	
(gatt_client_callback_num)	MIN: 1	
*Default: 15	MAX: 15	
Vender Specific Callback	Callback of completing BD address setting with open	
(vender_specific_callback)	API of Bearer Platform	
*Default: NULL		



3.5 Add and configure related module

The MESH Stack uses the following peripherals to perform MESH communication.

Table 3-11 Related peripherals

Item	Usage
Bluetooth Low Energy Driver (r_ble_extended or r_ble_extended_freertos)	Bluetooth Low Energy communication
General PWM Timer Driver (r_gpt)	Timer (g_timer0) for Bluetooth Mesh Stack Timer (g_timer1) for Bluetooth LE Stack
Low-Power Flash Driver (r_flash_lp)	Store Bonding information etc.
Interrupt Controller Unit Driver (r_icu)	Interrupt (g_external_irq0) from BLE(H/W) Interrupt (g_ble_sw_irq) from switch(H/W)
Serial Communication Interface Driver (r_sci_uart)	Serial communication
Low Power Modes Driver (r_lpm)	Low Power Modes of MCU

This section describes how to configure related peripherals (timers, interrupt) for MESH Stack which added in previous section. Procedure describes in this section is common to BareMetal and FreeRTOS environment.



3.5.1 r_gpt (g_timer0)

1. Click Add Timer Driver box and select New→Timer, General PWM (r_gpt).

*[sample_appl] FSP Configuration ×	•
Stacks Configuration	Generate Project Content
Threads 🐑 New Thread 🔊 Remove 🖃	HAL/Common Stacks 🚯 New Stack > 🏦 Extend Stack > 🛞 Remove
 	BLE Mesh Bearer Platform (m_mesh_bearer_platform)
	BLE Mesh Bearer (rm_ble_mesh_bearer)
	BLE Mesh (m_ble_mesh)
Objects	# BLE Mesh OS on Baremetal (m_mesh_timer_bare # Add Timer Driver # g_flash/D Flash (r,flash,lp)
	al) metal) ↓ ↑ Timer, General PWM (r.gpt)
	Timer, Low-Power (r.agt) (r.flash_lp)
Summary BSP Clocks Pins Interrupts Event Links @ Stacks Compor	sere l

Figure 3-24 Add GPT for Bluetooth Mesh Stack

2. Set Overflow/Crest Interrupt Priority of g_timer0 Timer, General PWM (r_gpt) as Priority 3 on Properties tab.

her) Timer, General PWM (r_gpt)	
qs	Property	Value
fo	✓ Common	
10	Parameter Checking	Default (BSP)
	Pin Output Support	Disabled
	Write Protect Enable	Disabled
	Clock Source	PCLKD
	 Module g_timer0 Timer, General PWM (r_gpt) 	
	> General	
	> Output	
	> Input	
	✓ Interrupts	
	Callback	NUU
	Overflow/Crest Interrupt Priority	Priority 3
	Capture A Interrupt Priority	Disabled
	Capture B Interrupt Priority	Disabled
	Underflow/Trough Interrupt Priority	Disabled
	<	

Figure 3-25 GPT configuration for Bluetooth Mesh Stack



3.5.2 r_flash_lp

Configure as following.

1. Set the **Data Flash Background Operation** of **g_flash0 Flash (r_flash_lp)** as **Disabled** on **Properties** tab.

h0 Flash (r_flash_lp)	
Property	Value
✓ Common	
Parameter Checking	Default (BSP)
Code Flash Programming	Disabled
Data Flash Programming	Enabled
 Module g_flash0 Flash (r_flash_lp) 	
Name	g_flash0
Data Flash Background Operatio	n Disabled
Callback	NULL
Flash Ready Interrupt Priority	Disabled

Figure 3-26 Flash configuration



3.5.3 r_icu (g_external_irq0)

- 1. Set **Pin Interrupt Priority of g_external_irq0 External IRQ (r_icu)** as the followings.
 - **BareMetal environment** Priority 0 on Priority.

■ Properties × 🦗 スマート・ブラウザー				
g_extern	g_external_irq0 External IRQ (r_icu)			
Settings API Info	Property ✔ Common	Value		
7411110	Parameter Checking	Default (BSP)		
	 Module g_external_irq0 External IRQ (r_icu) 			
	Name	g_external_irq0		
	Channel	8		
	Trigger	🔒 Falling		
	Digital Filtering	Disabled		
	Digital Filtering Sample Clock (Only valid whe	PCLK / 64		
	Callback	🔒 r rf ble interrupt		
	Pin Interrupt Priority	Priority 0 (highest)		
	✓ Pins			
	IRQ08	<unavailable></unavailable>		
	Select the PIN interrupt priority.			

Figure 3-27 ICU configuration (BareMetal Environment)

• FreeRTOS environment

Priority 1 on Priority. Because it is the highest priority used by the FreeRTOS kernel.

Propert	ies × 🛞 スマート・ブラウザー			
g_external_irq0 External IRQ (r_icu)				
Settings API Info	Property ✔ Common	Value		
	Parameter Checking Module g_external_irq0 External IRQ (r_icu) 	Default (BSP)		
	Name Channel	g_external_irq0		
	Trigger Digital Filtering	Falling Disabled		
	Digital Filtering Sample Clock (Only valid whe Callback	PCLK / 64		
	Pin Interrupt Priority	Priority 1		
	V Pins IRQ08	<unavailable></unavailable>		
	Select the PIN interrupt priority.			

Figure 3-28 ICU configuration (FreeRTOS Environment)



3.5.4 r_gpt (g_timer1)

1. Click Add GPT Driver box and select New→Timer, General PWM (r_gpt).

*[sample_appl] FSP Conf	igurution of		
Stacks Configuration	on		Generate Project Conte
hreads	🗟 New Thread 🔞 Remove 🕒	BLE Mesh Bearer Platform (rm_ble_mesh_bearer_platform) Stacks	🔁 New Stack > 😩 Extend Stack > 💼 Remove
 ✓ AL/Common ♥ g_ioport I/O Port (r_ioport) ♥ FreeRTOS Port (m_freertos_port) ♥ FreeRTOS Heap 4 		E Mesh Bearer (rm_ble_mesh_bearer)	
V @ BLE_CORE_TASK			
4 BLE Mesh Bearer	Platform (rm_ble_mesh_bearer_platform)	E Mesh (rm_ble_mesh)	
Objects	New Object > Remove	(r_flash_lp) Exte	tetmal_irq0 rmal IRQ (r_icu) mer, General PWM (r_gpt) New mer, Low-Power (r_agt)

Figure 3-29 Add GPT for Bluetooth LE Stack

2. Set Overflow/Crest Interrupt Priority of g_timer1 Timer, General PWM (r_gpt) as Priority 2 on Properties tab.

-	0 Timer, General PWM (r_gpt)	
Settings	Property	Value
API Info	> Common	
Arrino	 Module g_timer0 Timer, General PWM (r_gpt) 	
	> General	
	> Output	
	> Input	
	✓ Interrupts	
	Callback	🔒 r rf host timer interrup
	Overflow/Crest Interrupt Priority	Priority 2
	Capture A Interrupt Priority	Disabled
	Capture B Interrupt Priority	Disabled
	Underflow/Trough Interrupt Priority	Disabled
	> Extra Features	
	✓ Pins	
	GTIOCA	<unavailable></unavailable>
	GTIOCB	<unavailable></unavailable>

Figure 3-30 GPT configuration for Bluetooth LE Stack



3.5.5 r_icu (g_ble_sw_irq)

If you use a switch on EK-RA4W1, set the configuration as the followings.

1. Click **New Stack** and add **Input→External IRQ Driver (r_icu)** to **HAL/Common**.

🔅 *[sample_appl] FSP Configuration 😕		
Stacks Configuration		Generate Project Content
Threads New Thread Remove # HAL/Common @ gioport I/O Port (r_joport) # FreeRTOS Port (m_freertos_port) # FreeRTOS Heap 4 # BLE_CORE_TASK # BLE_Mesh Bearer Platform (m_ble_mesh_bearer_platform) # BLE Mesh Bearer Platform (on Off Server (m_mesh_generic) > Diplects New Object > Remove 	g_ioport I/O Port (r_joport) FreeRTOS Port (rm_freertos_port) 1 1	♦ New Stack ♦ Extendel Stack > ● Demonstere Analog > Artificial Intelligence > Audio > Bootloader > CapTouch > Connectivity > DSP > Graphics > ♦ Key Matrix (r_kint) Monitoring Motor > Networking > Sensor > Storage > System > Timers > Y Search



2. Set g_external_irq1 External IRQ (r_icu) as the following.

:

- [Name] : g_ble_sw_irq
- [Channel] : 4
- [Trigger] : Falling
- [Callback]
- Callback_ble_sw_irq

Priority 2

[Pin Interrupt Priority] :

exterr	nal_irq1 External IRQ (r_icu)			
ettings	Property Value			
PI Info	✓ Common			
111110	Parameter Checking	Default (BSP)		
	 Module g_ble_sw_irq External IRQ (r_icu) 			
	Name	g_ble_sw_irq		
	Channel	4		
	Trigger	Falling		
	Digital Filtering	Disabled		
	Digital Filtering Sample Clock (Only valid whe	PCLK / 64		
	Callback	Callback_ble_sw_irq		
	Pin Interrupt Priority	Priority 2		
	✓ Pins			
	IRQ04	<unavailable></unavailable>		
	Module name.			

Figure 3-32 ICU Driver configuration



3.5.6 r_sci_uart

If you use the Command Line Interface (CLI) to perform serial communication with the EK-RA4W1, set the configuration as the following.

3.5.6.1 Related source files

Source files related to the CLI are installed under ./src/app_lib in this demo project. The user can add the CLI functionality by copying the app_lib directory from this demo project to another project.

3.5.6.2 Configurations of SCI

Open the FSP configuration of user's project and select the **Stacks** tab. Add **New Stack**->Connectivity->UART (r_sci_uart) to HAL/Common. Modify configuration of the added r_sci_uart as the following.

Priority 2

P205

P206

4

- [General]→[Channel]
- [Interrupts]→[Callback] : user_uart_callback_ble_cli

2

- [Interrupts]→[xxx Interrupt Priority] :
- [Pins]→[TXD] :
- [Pins]→[RXD] :

UART (r_sci_uart)	
Property	Value
> Common	
 Module g_uart0 UART (r_sci_uart) 	
✓ General	
Name	g_uart0
Channel	4
Data Bits	8bits
Parity	None
Stop Bits	1bit
> Baud	
> Flow Control	
> Extra	
✓ Interrupts	
Callback	user_uart_callback_ble_cl
Receive Interrupt Priority	Priority 2
Transmit Data Empty Interrupt Pr	ority Priority 2
Transmit End Interrupt Priority	Priority 2
Error Interrupt Priority	Priority 2
✓ Pins	
TXD	P205
RXD	P206
CTS	<unavailable></unavailable>

Figure 3-33 UART configuration



3.5.6.3 Designating module name

Edit the value of BLE_UART_INSTANCE macro in app_lib / r_ble_console.c if the module name of the r_sci_uart has been changed from g_uart0.

/*************************************			
#define BLE_TX_BUFSIZ	(180)		
#define BLE_UART_INSTANCE	(g_uart0)		

Code 1. BLE_UART_INSTANCE macro

3.5.6.4 Serial data output of UART

The serial data output of UART can be invoked by using the *R_BLE_CLI_Printf()* function. *R_BLE_CLI_Printf()* function can generate formatted character lines similar to the *printf()* function.

Table 3-12 Syntax of R_BLE_CLI_Printf()

Function Name	R_BLE_CLI_Printf		
Format	void R_BLE_CLI_Printf(const char *format,);		
Return	void -		
Arguments	const char *format Designate a constant character line including formats		
	Variable number of arguments represented by formats can be designated.		



3.5.7 r_lpm

If you use the low power mode of the MCU, configure as following.

2. Click **New Stack** and add **Power→Low Power Modes (r_lpm)** to **HAL/Common**.

*[sample_appl] FSP Configuration ×				
Stacks Configuration		Generate Project Content		
Stacks Configuration Threads New Thread Remove G g_ioport I/O Port (r_ioport) FreeRTOS Port (m_freertos_port) FreeRTOS Heap 4 g_ble_sw_irq External IRQ (r_icu) g_uart0 UART (r_sci_uart) BLE_COERTASK BLE_DERE_Mash Bearer Platform (m_ble_mesh_bearer_platform) BLE_Mesh Model Generic On Off Server (m_mesh_generic_c) Dipiects New Object > Remove	HAL/Common Stacks FreeRTOS Port (m_freertos_port)	V Stark S Extand Ctark S P Demoue Analog Analog Analog Artificial Intelligence Audio Bootloader CapTouch Connectivity Aconnectivity DSP Andreader Aconnectivity		
	<	Storage > System > Timers > Transfer >		
Summary BSP Clocks Pins Interrupts Event Links @ Stacks Component				

Figure 3-34 Add LPM



3.6 Generate Code

Click [Generate] button on the FSP Configuration. API header, library, code, data, and the configuration files of each modules are generated in "ra", "ra_gen", "ra_cfg" folders of the project.

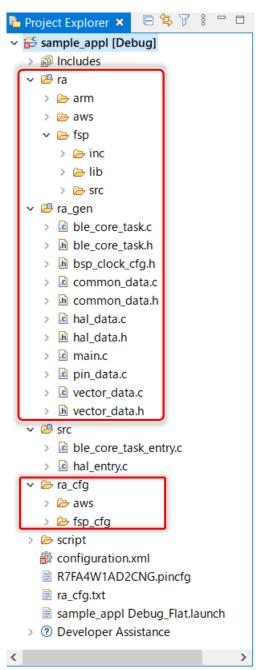


Figure 3-35 Result of Code Generation

3.7 Building and debugging

Refer to section 2.2.



4. How to Implement Mesh Applications

Regarding how to implement mesh applications using the Mesh Stack, refer to "RA4W1 Group Bluetooth Mesh Development Guide" (R01AN5849).



5. Appendix

5.1 Program Size

Table 5 1 shows the program size of demo project.

Table 5-1 Program Size

Project	ROM Size	RAM Size
ekra4w1_mesh_cli_client_baremetal	510KB	52KB
ekra4w1_mesh_cli_server_baremetal	477KB	53KB
ekra4w1_mesh_client_baremetal	334KB	48KB
ekra4w1_mesh_client_freertos	348KB	82KB
ekra4w1_mesh_server_baremetal	335KB	48KB
ekra4w1_mesh_server_freertos	349KB	82KB



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.

RA4W1 Group Bluetooth Mesh Stack uses the following open source software.

 <u>crackle</u>; 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.

Revision History

Rev.	Date	Description	
1.00	Feb. 25, 2022	-	First edition
1.01	Apr. 27, 2022	P.21	Changed Total Heap Size from 10240 to 11264.
		-	Updated attached demo project.
1.03	Aug. 29, 2022	P. 8	Updated software requirements.
		P.25	Added a note to set the same value to the properties of the BLE Mesh Bearer
			Platform and BLE Driver box.
		-	Updated attached demo project.
1.04	Oct. 26, 2022	P. 8	Updated software environment.
		-	Updated attached demo project.
1.05	Dec. 16, 2022	P. 8	Updated software environment.
		P. 27	Updated FSP Configuration parameters.
		-	Updated attached demo project.

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 is supplied until the power is supplied until the power reaches the level at which reseting 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 systemevaluation test for the given product.

Notice

- 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 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.
- This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
 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/.