RA4W1 Group

Various PHY connection sample application with advertising extension

Introduction

This document describes how to use the advertising extension based on the demo project of the *RA4W1 Group BLE sample application* (R01AN5402).

Target Device

RA4W1 Group

Related Documents

Bluetooth Core Specification (https://www.bluetooth.com)

RA4W1 Group User's Manual: Hardware (R01UH0883)

Renesas Flexible Software Package (FSP) User's Manual
e^2 studio Getting Started Guide (R20UT4204)

EK-RA4W1 – Quick Start Guide (R20QE0015)

EK-RA4W1 User's Manual (R20UT4683)

RA4W1 Group BLE sample application (R01AN5402)

RA4W1 Group Bluetooth Low Energy Profile Developer's Guide (R01AN5428)

QE for BLE[RA,RE] V1.2.0 Release Note (R20UT4951EJ)

Related Environments

Refer to section 2.1.

Contents

1. Overview

This document describes how to modify the sample project provided by the *RA4W1 Group BLE sample application (R01AN5402)* to make the advertising extension available. Advertising Extensions is a feature that was added in Bluetooth Low Energy 5.0. You can establish a connection using LE 1M PHY, LE 2M PHY, or LE Coded PHY without PHY update procedure by using the advertising extension feature.

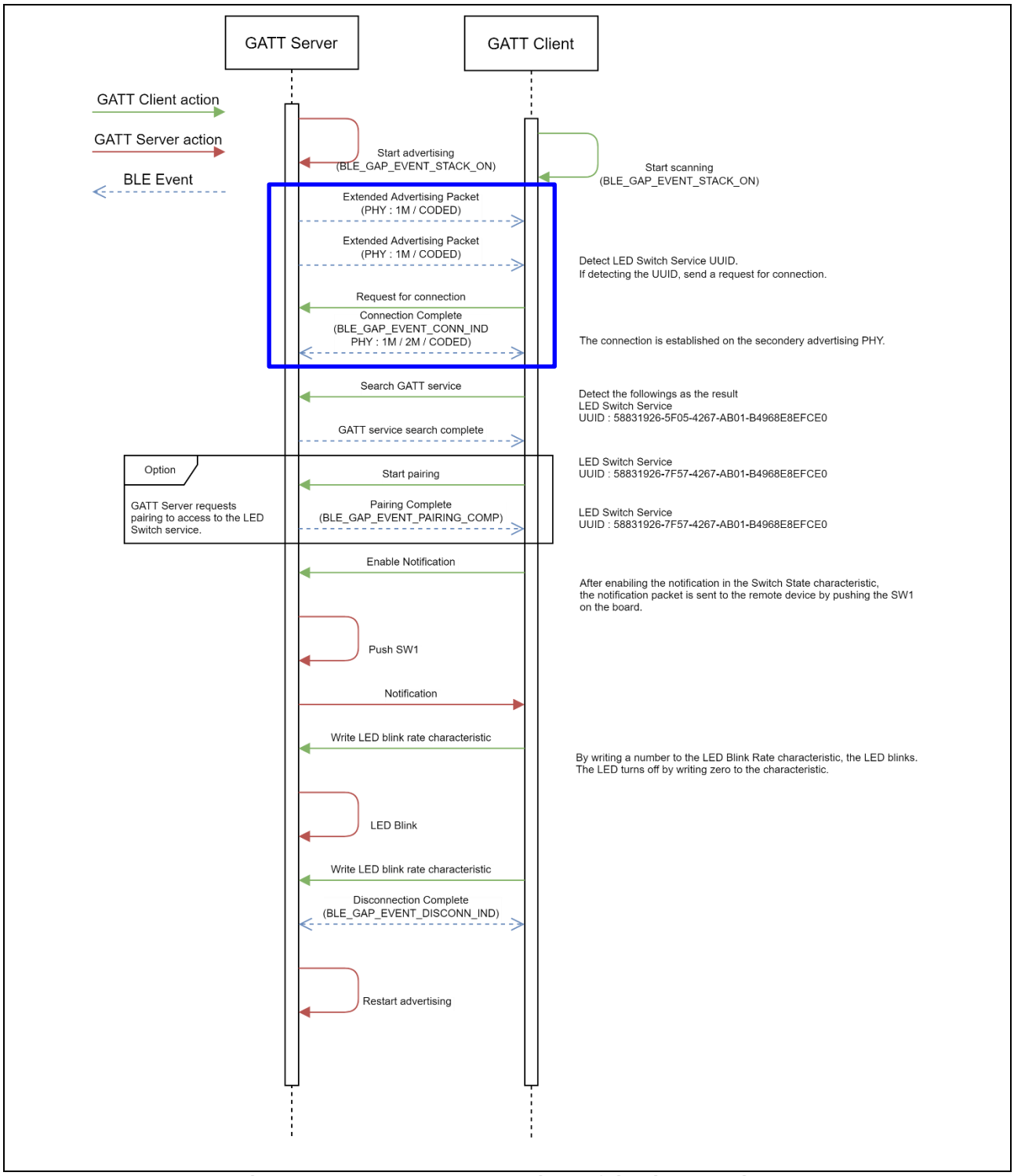

Figure 1. Massage sequence chart of the demo project

Please refer to *RA4W1 Group BLE sample application (R01AN5402)* to get the sample project and learn more about the contents of the sample project.

2. How to use the demo project

This chapter describes how to use the sample project with reference to *RA4W1 Group BLE sample application (R01AN5402)*.

2.1 Operating environment

Table 1 shows the hardware requirements for building and debugging BLE software.

Table 1. Hardware requirements

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 onboard debugger (J-Link OB). Therefore it is not necessary to prepare an emulator.
USB cables	Used to connect to the MCU board. EK-RA4W1: 2 USB A-microB cable

Table 2 shows the software requirements for build and debug BLE software.

Table 2. Software requirements

Software		Version	Description
GCC environment	e² studio	2021-07	Integrated development environment (IDE) for Renesas devices.
	GCC ARM Embedded	V9.3.1	C/C++ Compiler. (download from e² studio installer)
	Renesas Flexible Software Package (FSP)	V3.2.0	Software package for making applications for the RA microcontroller series.
	QE for BLE[RA,RE]	V1.2.0	Generates the source codes (BLE base skeleton program) as a base for the BLE Application and the BLE Profile. Refer to *R20UT4951EJ* about how to download QE for BLE.
	QE utility[RA,RE]	V1.2.1	
	SEGGER J-Flash	V7.50	Tool for programming the on-chip flash memory of microcontrollers.
Header files			All API calls and their supporting interface definitions are located in r_ble_api.h and rm_ble_abs_api.h.
Integer types			It uses ANSI C99 “Exact width integer types”. These types are defined in stdint.h.
Endian			Little-endian
Terminal emulator			VT-100 compatible

2.2 Importing demo project

Refer to *RA4W1 Group BLE sample application (R01AN5402)* about how to import demo projects into e²stidio.

2.3 Building and debugging

Refer to *e² studio Getting Started Guide (R20UT4204)*

3. Implementation

This chapter describes how to modify the sample project provided by the *RA4W1 Group BLE sample application (R01AN5402)* to make the advertising extension available. Note that advertising extension can only be used in the "*extended*" configuration of the *BLE abstraction driver on rm_ble_abs*.

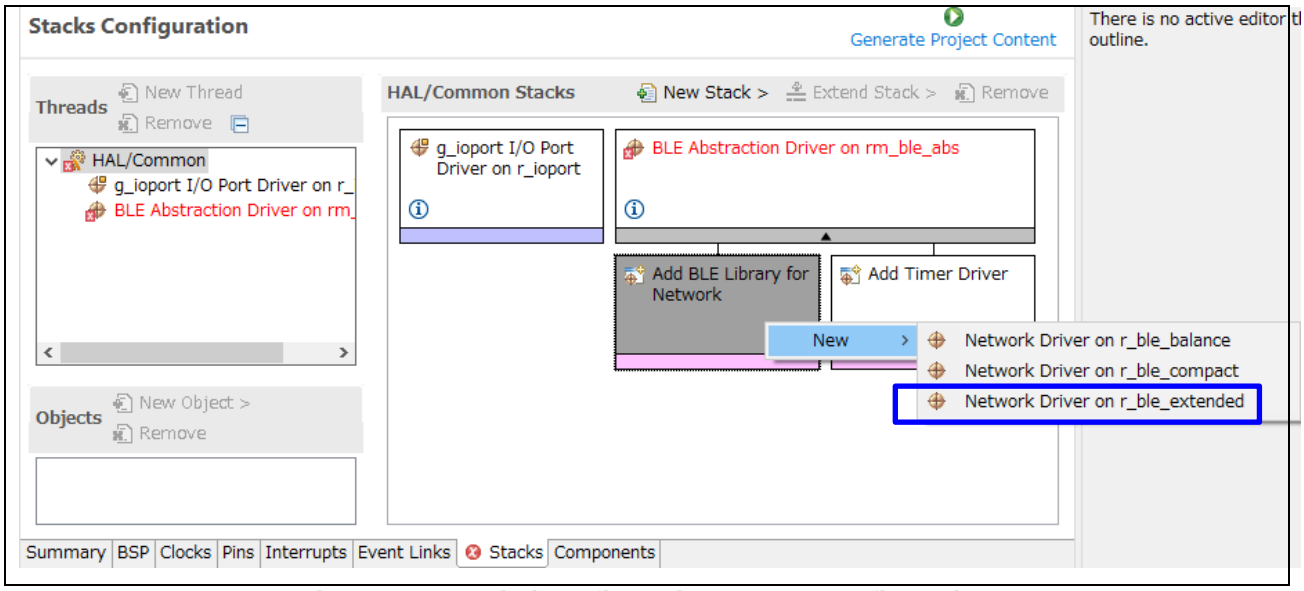

Figure 2. Extended configuration on FSP configuration

After applying all the changes in this chapter to sample application with *BLE sample application (R01AN5402)*, refer to the *e² studio Startup Guide (R20UT4204)* to build and write the program to the EK-RA4W1 board.

3.1 Server-side

Existing server side sample application in *RA4W1 Group BLE sample application (R01AN5402)* performs legacy advertising and waits for a connection request from the client. This section describes how to change legacy advertising to extended advertising. Items to be changed as follows.

1. Replace legacy advertising API with extended advertising API
2.
3. Add the LED Switch Service UUID to gs_advertising_data

3.1.1 Replace legacy advertising API with extended advertising API

To use extended advertising, it is necessary to replace *RM_BLE_ABS_StartLegacyAdvertising* API in app_main.c with *RM_BLE_ABS_StartExtendedAdvertising* API as shown in Code 1 and Code 2.

```
void vs_cb(uint16_t type, ble_status_t result, st_ble_vs_evt_data_t *p_data)
{
/* some code is omitted */
    switch(type)
    {
        case BLE_VS_EVENT_GET_ADDR_COMP:
        {
            /* Start advertising when BD address is ready */
            st_ble_vs_get_bd_addr_comp_evt_t * get_address =
            (st_ble_vs_get_bd_addr_comp_evt_t *)p_data->p_param;

            memcpy(g_ble_abs_extend_advertising_parameter.own_bluetooth_address,
            get_address->addr.addr, BLE_BD_ADDR_LEN);
```

Change API from *RM_BLE_ABS_StartLegacyAdvertising* to *RM_BLE_ABS_StartExtendedAdvertising*.

```
            RM_BLE_ABS_StartExtendedAdvertising(&g_ble_abs0_ctrl,
            &g_ble_abs_extend_advertising_parameter);
        } break;
/* some code is omitted */
    }
}
```

Code 1. Replace with RM_BLE_ABS_StartExtendedAdvertising API in vs_cb (app_main.c)

```
void gap_cb(uint16_t type, ble_status_t result, st_ble_evt_data_t *p_data)
{
 /* some code is omitted */
    switch(type)
    {
/* some code is omitted */
        case BLE_GAP_EVENT_CONN_IND:
        {
            if (BLE_SUCCESS == result)
            {
/* some code is omitted */
            }
            else
            {
                /* Restart advertising when connection failed */
```

Change API from *RM_BLE_ABS_StartLegacyAdvertising* to *RM_BLE_ABS_StartExtendedAdvertising*.

```
                RM_BLE_ABS_StartExtendedAdvertising(&g_ble_abs0_ctrl,
                &g_ble_abs_extend_advertising_parameter);
            }
        } break;

        case BLE_GAP_EVENT_DISCONN_IND:
        {
            /* LED OFF */
            g_ioport.p_api->pinWrite(g_ioport.p_ctrl, BSP_IO_PORT_04_PIN_04,
                BSP_IO_LEVEL_HIGH);

            /* Restart advertising when disconnected */
            g_conn_hdl = BLE_GAP_INVALID_CONN_HDL;
```

Change API from *RM_BLE_ABS_StartLegacyAdvertising* to *RM_BLE_ABS_StartExtendedAdvertising*.

```
            RM_BLE_ABS_StartExtendedAdvertising(&g_ble_abs0_ctrl,
            &g_ble_abs_extend_advertising_parameter);
        } break;
/* some code is omitted */
```

Code 2. Replace with RM_BLE_ABS_StartExtendedAdvertising API in gap_cb (app_main.c)

3.1.2 Declare a new structure for extended advertising

RM_BLE_ABS_StartExtendedAdvertising API needs "*ble_abs_extend_advertising_parameter_t*" structure containing advertising parameter (e.g. advertising interval, advertising PHY, etc.) as an argument. To use the API, it is necessary to newly declare the structure in app_main.c as following. Refer to *Renesas Flexible Software Package (FSP) User's Manual* about details of "*ble_abs_extend_advertising_parameter_t*" structure.

```
ble_abs_extend_advertising_parameter_t g_ble_abs_extend_advertising_parameter =
{
 .p_peer_address             = NULL,
 .slow_advertising_interval  = 0x0000003E, 1,000.0(ms)
 .slow_advertising_period    = 0x0000,
 .p_advertising_data         = gs_advertising_data,
 .advertising_data_length    = ARRAY_SIZE(gs_advertising_data),
 .advertising_filter_policy  = BLE_ABS_ADVERTISING_FILTER_ALLOW_ANY,
 .advertising_channel_map    = ( BLE_GAP_ADV_CH_37 | BLE_GAP_ADV_CH_38 |
                               BLE_GAP_ADV_CH_39 ),
 .own_bluetooth_address_type = BLE_GAP_ADDR_RAND,
 .own_bluetooth_address      = { 0 },
 .primary_advertising_phy    = BLE_GAP_ADV_PHY_CD,
 .secondary_advertising_phy  = BLE_GAP_ADV_PHY_1M
};
```

Code 3. ble_abs_extend_advertising_parameter_t structure

The structure has members that specify the primary PHY and secondary PHY to be used in the advertising extension as "*primary_advertising_phy*" and "*secondary_advrtising_phy*". The combinations of PHYs that can be used for primary and secondary advertising PHYs are shown in .

Table 3. Combinations of PHYs that can be used for primary and secondary advertising PHYs

		Secondary advertising PHY		
		LE 1M PHY	LE 2M PHY	LE Coded PHY
Primary advertising PHY	LE 1M PHY	Yes	Yes	Yes
	LE 2M PHY	No	No	No
	LE Coded PHY	Yes	Yes	Yes

The example shown in uses a Coded PHY for primary advertising PHY and 1M PHY for secondary advertising PHY.

3.1.3 Add the LED Switch Service UUID to gs_advertising_data

Client will send a connection request when client finds advertising PDU with LED switch service UUID (58831926-5F05-4267-AB01-B4968E8EFCE0) with advertising data. Add LED switch service UUID to "*gs_advertising_data*" in app_main.c so that client can find advertising PDU from server-side as following.

```
/* Advertising Data */
static uint8_t gs_advertising_data[] =
{
    /* Flags */
    0x02, /**< Data Size */
    0x01, /**< Data Type */
    ( 0x06 ), /**< Data Value */

    /* Complete List of 128-bit Service Class UUIDs */
    0x11, /**< Data Size */
    0x07, /**< Data Type */
    0xe0, 0xfc, 0x8e, 0x8e, 0x96, 0xb4, 0x01, 0xab, 0x67, 0x42, 0x05, 0x5f, 0x26,
    0x19, 0x83, 0x58, /**< Data Value */

    /* Shortened Local Name */
    0x05, /**< Data Size */
    0x08, /**< Data Type */
    0x52, 0x42, 0x4c, 0x45, /**< Data Value */
};
```

Add Service Class UUID to advertising data.

Code 4. Add LED switch service UUID

You can also use QE for BLE to add it. Please refer to *RA4W1 Group Bluetooth Low Energy Profile Developer's Guide (R01AN5428)* about how to add the advertising data by using QE for BLE.

3.2 Client-side

Existing client side sample application in *RA4W1 Group BLE sample application (R01AN5402)* performs legacy scan and sends connection request to the server. This section describes how to change legacy scan and connection to extended scan and connection. Items to be changed as follows.

1. Enable scan and connection API

2. Modify scan parameters
3. Add scan filter parameter
4. Modify create connection parameters

3.2.1 Enable scan and connection API

1) **Scan**

You can use *RM_BLE_ABS_StartScanning* API to perform extended scan. In existing client side sample application in *RA4W1 Group BLE sample application (R01AN5402)*, the scan API call is commented out. Therefore, it is necessary to enable the API call as follows.

```
void gap_cb(uint16_t type, ble_status_t result, st_ble_evt_data_t *p_data){
/* some code is omitted */
    switch(type)
    {
        case BLE_GAP_EVENT_STACK_ON:
        {
/* some code is omitted */
            RM_BLE_ABS_StartScanning(&g_ble_abs0_ctrl, &gs_scan_parameter);
        } break;
        case BLE_GAP_EVENT_CONN_IND:
        {
            if (BLE_SUCCESS == result)
            {
/* some code is omitted */
            }
            else
            {
                RM_BLE_ABS_StartScanning(&g_ble_abs0_ctrl, &gs_scan_parameter);
            }
        } break;
        case BLE_GAP_EVENT_DISCONN_IND:
        {
/* some code is omitted */
            RM_BLE_ABS_StartScanning(&g_ble_abs0_ctrl, &gs_scan_parameter);
        } break;
/* some code is omitted */
```

Remove comment out

Remove comment out

Remove comment out

Code 5. RM_BLE_ABS_StartScanning API in gap_cb (app_main.c)

2) **Connection**

You can use *RM_BLE_ABS_CreateConnection* API to perform extended create connection. In existing client side sample application in *RA4W1 Group BLE sample application (R01AN5402)*, the connection API call is commented out. Therefore, it is necessary to enable the API call as following.

```
void gap_cb(uint16_t type, ble_status_t result, st_ble_evt_data_t *p_data)
{
```

```
/* some code is omitted */
    switch(type)
    {
/* some code is omitted */
        case BLE_GAP_EVENT_SCAN_OFF:
        {
            /* Send connection request after scan stopped */
            RM_BLE_ABS_CreateConnection(&g_ble_abs0_ctrl, &gs_connection_parameter);
        } break;
/* some code is omitted */
```

Remove comment out

Code 6. RM_BLE_ABS_CreateConnection API in gap_cb (app_main.c)

3.2.2 Modify scan parameters

RM_BLE_ABS_StartScanning API needs "*ble_abs_scan_parameter_t*" and "*ble_abs_scan_phy_parameter_t*" structure containing scan parameter (e.g. scan interval, scan PHY, etc.) as an argument. The structure is already declared in existing sample application as following. Refer to *Renesas Flexible Software Package (FSP) User's Manual* about details of these structures.

```
/* Scan phy parameters */
static ble_abs_scan_phy_parameter_t gs_scan_phy_parameter =
{
    .fast_scan_interval      = 0x0000, /* 0.0(ms) */
    .fast_scan_window        = 0x0000, /* 0.0(ms) */
    .slow_scan_interval      = 0x0800, /* 1,280.0(ms) */
    .slow_scan_window        = 0x0012, /* 11.25(ms) */
    .scan_type               = BLE_GAP_SCAN_PASSIVE
};
/* Scan parameters */
static ble_abs_scan_parameter_t gs_scan_parameter =
{
    .p_phy_parameter_1M        = &gs_scan_phy_parameter,
    .p_phy_parameter_coded     = &gs_scan_phy_parameter,
    .fast_scan_period          = 0x0BB8, /* 30,000(ms) */
    .slow_scan_period          = 0x0000,
    .p_filter_data             = gs_filter_data,
    .filter_data_length        = ARRAY_SIZE(gs_filter_data),
    .filter_ad_type            = 0x07,
    .device_scan_filter_policy = BLE_GAP_SCAN_ALLOW_ADV_ALL,
    .filter_duplicate          = BLE_GAP_SCAN_FILT_DUPLIC_ENABLE,
};
```

Code 7. ble_abs_scan_phy_parameter_t and ble_abs_scan_phy_parameter_t structure

The "*ble_abs_scan_parameter_t*" has members that specify the scanning PHY to be used in the extended scan as "*p_phy_parameter_1M*" and "*p_phy_parameter_coded*" member. The combinations of PHYs that can be used for scanning PHYs are shown in Table 4.

Table 4. Combinations of PHYs that can be used for extended scan

PHY to be scanned	p_phy_parameter_1M	p_phy_parameter_coded
LE 1M PHY	&gs_scan_phy_parameter	NULL
LE Coded PHY	NULL	&gs_scan_phy_parameter
LE 1M PHY + LE coded PHY	&gs_scan_phy_parameter	&gs_scan_phy_parameter

When scanning by LE 1M PHY and LE Coded PHY simultaneously, the sum of each PHY's scan window must be less than or equal to scan interval.

3.2.3 Add filter parameter

Client will wait for advertising PDU included LED switch service UUID (58831926-5F05-4267-AB01-B4968E8EFCE0) as section 3.1.3 mentioned. To perform such a filter procedure, it is necessary to newly add filter parameter to app_main.c as following.

```
/* Scan filter data (data type: List of 128-bit Service Solicitation UUIDs ) */
static uint8_t gs_filter_data[] =
{
   0xe0, 0xfc, 0x8e, 0x8e, 0x96, 0xb4, 0x01, 0xab, 0x67, 0x42, 0x05, 0x5f, 0x26, 0x19, 0x83, 0x58
};
```

Code 8. Add LED switch service UUID as scan filter parameter (app_main.c)

You can also use QE for BLE to add it. Please refer to *RA4W1 Group Bluetooth Low Energy Profile Developer's Guide (R01AN5428)* about how to add the scan filter data by using QE for BLE.

3.2.4 Modify create connection parameters

RM_BLE_ABS_CreateConnection API needs "*ble_abs_connection_parameter_t*" and "*ble_abs_connection_phy_parameter_t*" structure containing connection parameter (e.g. connection interval, connection PHY, etc.) as an argument. The structure is already declared in existing sample application as following. Refer to *Renesas Flexible Software Package (FSP) User's Manual* about details of these structures.

```
/* Connection phy parameters */
static ble_abs_connection_phy_parameter_t gs_connection_phy_parameter_1M =
{
    .connection_interval      = 0x0008, /* 10.0(ms) */
    .supervision_timeout      = 0x0200, /* 5,120(ms) */
    .connection_slave_latency = 0x0000,
};

static ble_abs_connection_phy_parameter_t gs_connection_phy_parameter_2M =
{
    .connection_interval      = 0x0010, /* 20.0(ms) */
    .supervision_timeout      = 0x0200, /* 5,120(ms) */
    .connection_slave_latency = 0x0000,
};

static ble_abs_connection_phy_parameter_t gs_connection_phy_parameter_coded =
{
    .connection_interval      = 0x0018, /* 30.0(ms) */
    .supervision_timeout      = 0x0200, /* 5,120(ms) */
    .connection_slave_latency = 0x0000,
};

/* Connection parameters */
static ble_abs_connection_parameter_t gs_connection_parameter =
{
    .p_connection_phy_parameter_1M = &gs_connection_phy_parameter_1M,
    .p_connection_phy_parameter_2M = &gs_connection_phy_parameter_2M,
    .p_connection_phy_parameter_coded = &gs_connection_phy_parameter_coded,
    .p_device_address              = &gs_connection_device_address,
    .filter_parameter              = BLE_GAP_INIT_FILT_USE_ADDR,
    .connection_timeout            = 0x05, /* 5(s) */
};
```

Code 9. Connection parameters settings (app_main.c)

The structure has members that specify the connection PHY to be used in the establishment connection as "*p_connection_phy_parameter _1M*", "*p_connection_phy_parameter_2M*" and "*p_phy_parameter_coded*" member. The combinations of the connection PHY should include the PHY that server side is using to send advertising PDU. For example, when server side send advertising PDU that the primary advertising PHY is Coded PHY and the secondary advertising PHY is LE 1M PHY, set "*ble_abs_connection_phy_parameter_t"* structure to "*p_connection_phy_parameter_1M*" and "*p_connection_phy_parameter_coded*" member as connection parameter.

4. Sample project behavior confirmation environment

This chapter describes the behavior of server and client after applying this document implementation.

4.1 Environment

To confrm actual behavior, it is necessary to make following environment.

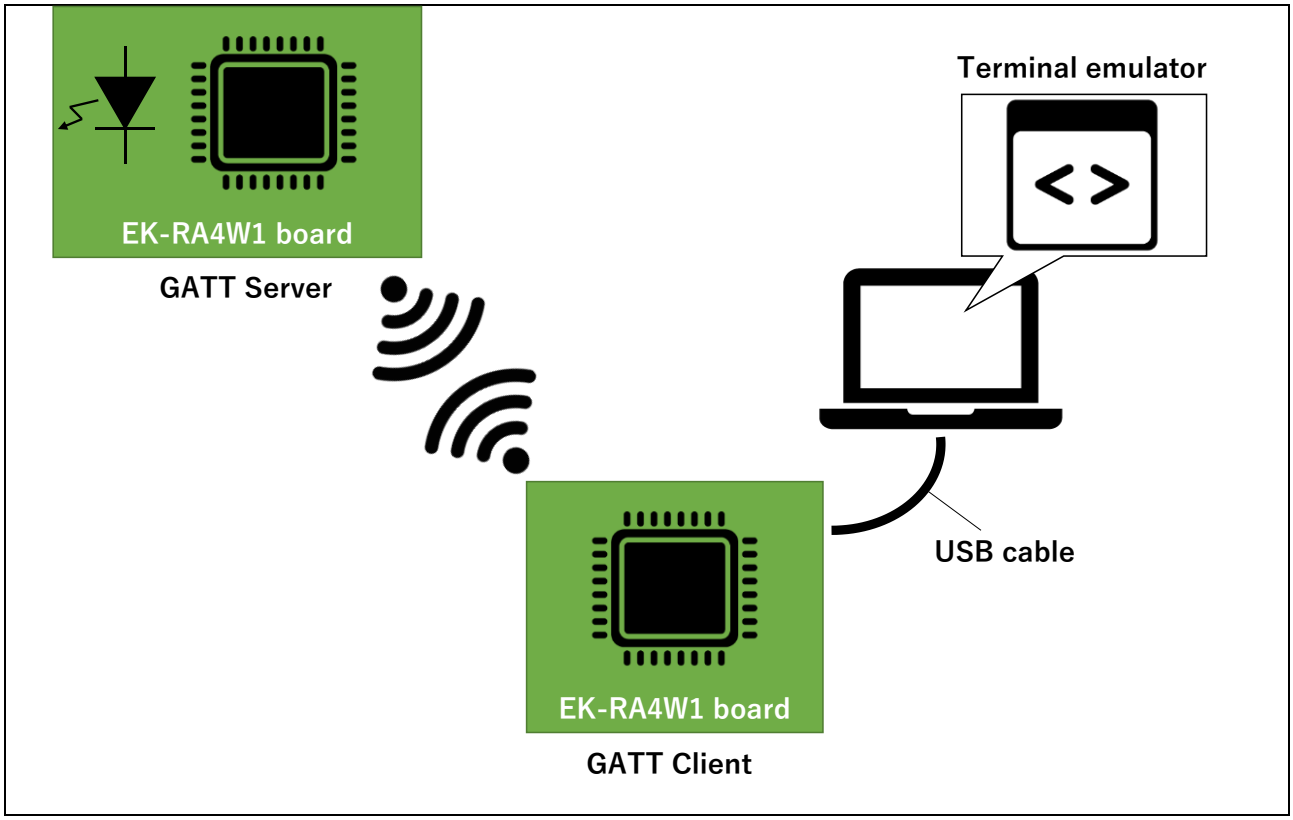

Figure 3. Sample project behavior confirmation environment

4.2 Establish connection between server and client

Server will automatically start extended advertising after power on EK-RA4W1 board. And client will automatically start scan and waiting for advertising PDU from server. After establish connection, you can see following characters on terminal emulator.

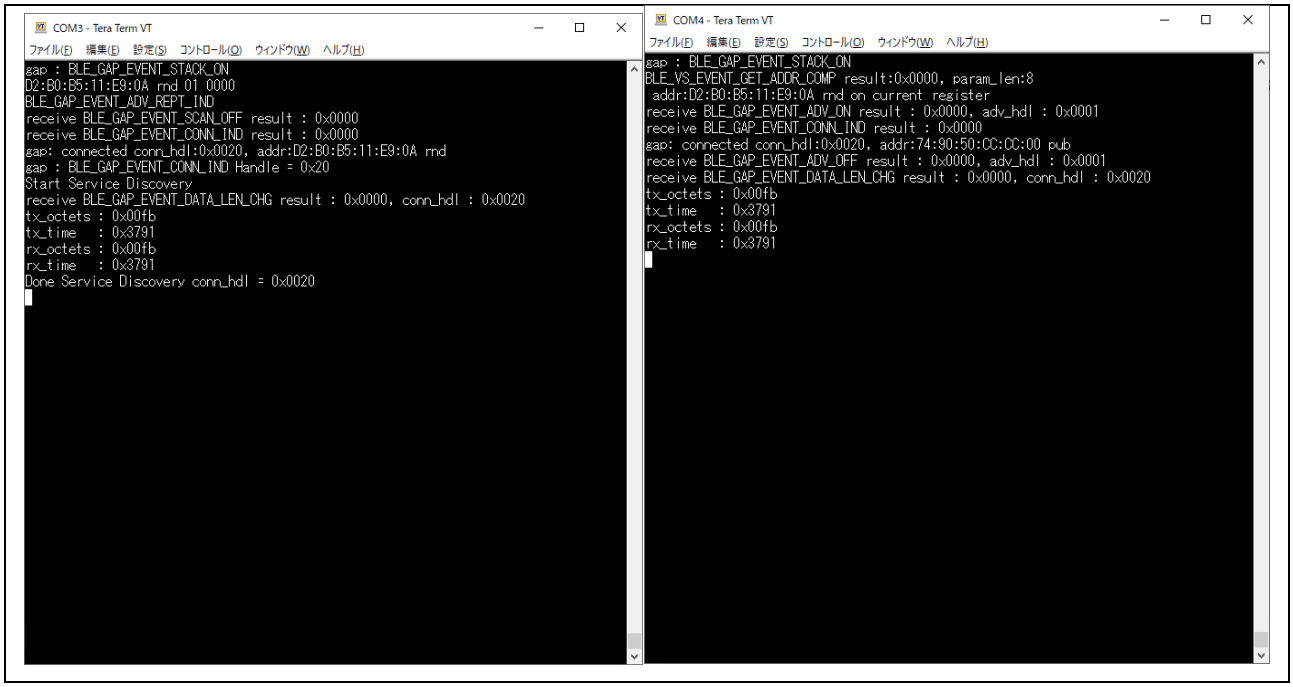

Figure 4. Console after the connection is established between the client and the server (left:client, right:server)

4.3 How to confirm current PHY

You can use *R_BLE_GAP_ReadPhy* API after establishing a connection to confirm current communication PHY. *BLE_GAP_RD_PHY_COMP* event will notify after calling the API.
You can implement the event handling function in gap_cb as following.

```
case BLE_GAP_RD_PHY_COMP:
{
    st_ble_gap_phy_rd_evt_t *p_conn_phy
        = (st_ble_gap_phy_rd_event_t *)p_data->p_param;
    R_BLE_CLI_Printf("tx_phy = 0x%X, rx_phy=0x%X\r\n",
        p_conn_phy->tx_phy, p_conn_phy->rx_phy);
} break;
```

Code 10. Event handling function of BLE_GAP_RD_PHY_COMP event in gap_cb (app_main.c)

Refer to *Renesas Flexible software package documentation* about usage of *R_BLE_GAP_ReadPhy* API and detail of "*st_ble_gap_phy_rd_event_t*" structure. When you have implemented above code, you can see following characters on terminal emulator.

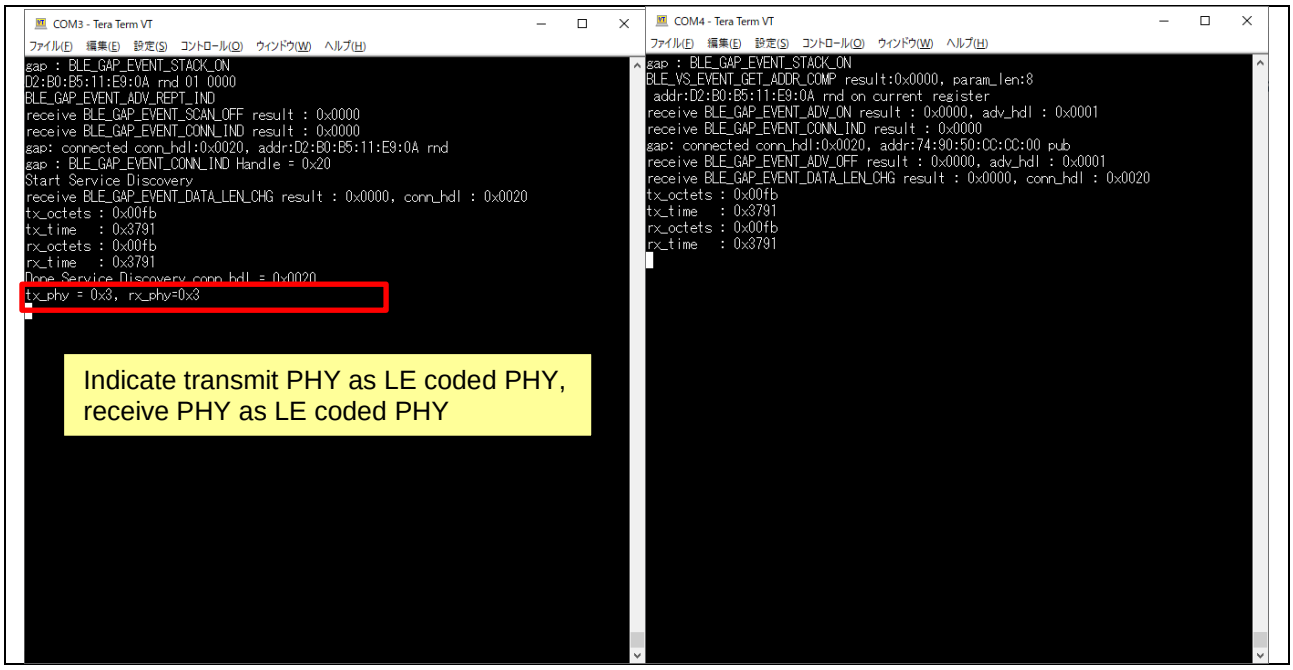

Figure 5. Current PHY read result (left:client, right:server)

Revision History

Rev.	Date	Description	
		Page	Summary
1.00	Sep.08.2021	—	The first edition issued

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

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

1. Precaution against Electrostatic Discharge (ESD)

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

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

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

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

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

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

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

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

Notice

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

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

(Rev.5.0-1 October 2020)

Corporate Headquarters

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

Trademarks

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

Contact information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit:
www.renesas.com/contact/.