

# RX ファミリ

## RYZ012 Bluetooth Low Energy モジュール

### Firmware Integration Technology

R01AN6290JJ0101

Rev.1.01

2022.03.31

#### 要旨

本アプリケーションノートは、Firmware Integration Technology (FIT)に準拠した RYZ012 Bluetooth Low Energy モジュール FIT モジュールの使用方法について説明します。

以降、RYZ012 Bluetooth Low Energy モジュール FIT モジュールのソフトウェアを総じて” RYZ012 FIT モジュール”、または”本 FIT モジュール”と称します。

本 FIT モジュールがサポートしている Bluetooth Low Energy モジュールは以下です。

Renesas Electronics 社製 RYZ012 Bluetooth Low Energy モジュール(RYZ012x1)

以降、本モジュールを”BLE モジュール”と称します。

本 FIT モジュールは、以下の FIT モジュールを使用します。

ボードサポートパッケージモジュール(R01AN1685)

RX ファミリ SCI モジュール (R01AN1815)

RX ファミリ バイト型キューバッファ (BYTEQ) モジュール (R01AN1683)

#### 対象デバイス

RX65N グループ

本アプリケーションノートを他のマイコンへ適用する場合、そのマイコンの仕様にあわせて変更し、十分評価してください。

#### 関連ドキュメント

Firmware Integration Technology ユーザーズマニュアル(R01AN1833)

ボードサポートパッケージモジュール Firmware Integration Technology(R01AN1685)

e<sup>2</sup> studio に組み込む方法 Firmware Integration Technology (R01AN1723)

CS+に組み込む方法 Firmware Integration Technology (R01AN1826)

Renesas e<sup>2</sup> studio スマート・コンフィグレータ ユーザーガイド(R20AN0451)

RX ファミリ SCI モジュール Firmware Integration Technology (R01AN1815)

RX ファミリ バイト型キューバッファ (BYTEQ) モジュール Firmware Integration Technology (R01AN1683)

## 目次

1. 概要 .....	4
1.1 RYZ012 FIT モジュールとは.....	4
1.2 RYZ012 Bluetooth Low Energy FIT モジュールの概要.....	4
1.2.1 ソフトウェア構成.....	4
1.2.2 ファイル構成.....	5
2. 要件 .....	6
2.1 ハードウェアの要求 .....	6
2.2 ソフトウェアの要求 .....	6
2.3 サポートされているツールチェーン .....	6
2.4 使用する割り込みベクタ .....	6
2.5 コンフィグレーション設定 .....	7
2.6 FIT モジュールの追加方法 .....	8
3. BLE Interface (Common) .....	9
3.1 R_BLE_Open().....	9
3.2 R_BLE_Close().....	10
3.3 R_BLE_Execute().....	11
3.4 R_BLE_SetEvent().....	12
4. BLE Interface (GAP) .....	13
4.1 R_BLE_GAP_Init() .....	13
4.2 R_BLE_GAP_SetAdvParam().....	14
4.3 R_BLE_GAP_SetAdvSresData() .....	16
4.4 R_BLE_GAP_StartAdv() .....	18
4.5 R_BLE_GAP_StopAdv() .....	20
5. BLE Interface (GATT Common) .....	21
5.1 R_BLE_GATT_GetMtu() .....	21
6. BLE Interface (GATT Server) .....	22
6.1 R_BLE_GATTS_Init().....	22
6.2 R_BLE_GATTS_SetDbInst().....	23
6.3 R_BLE_GATTS_RegisterCb().....	24
6.4 R_BLE_GATTS_Notification().....	25
6.5 R_BLE_GATTS_Indication() .....	26
6.6 R_BLE_GATTS_GetAttr() .....	27
6.7 R_BLE_GATTS_SetAttr().....	29
7. BLE Interface (GATT Client) .....	31
7.1 R_BLE_GATTC_Init().....	31
7.2 R_BLE_GATTC_RegisterCb() .....	32
8. BLE Interface (Vendor Specific) .....	33
8.1 R_BLE_VS_SetTxPower() .....	33

8.2	R_BLE_VS_SetBdAddr() .....	35
9.	Abstraction API for Renesas QE for BLE .....	36
9.1	RM_BLE_ABS_Open().....	36
9.2	RM_BLE_ABS_Close() .....	38
9.3	RM_BLE_ABS_StartLegacyAdvertising() .....	39
10.	QE for BLE を用いたサンプルコード生成.....	41
11.	動作確認環境 .....	55
12.	参考ドキュメント .....	56
	改訂記録.....	58

## 1. 概要

### 1.1 RYZ012 FIT モジュールとは

本 FIT モジュールは、API としてプロジェクトに組み込んで使用します。本 FIT モジュールの組み込み方については、「2.6 章 FIT モジュールの追加方法」および「10 章 QE for BLE を用いたサンプルコード生成」を参照してください。

### 1.2 RYZ012 Bluetooth Low Energy FIT モジュールの概要

#### 1.2.1 ソフトウェア構成

図 1.1 にソフトウェア構成を示します。

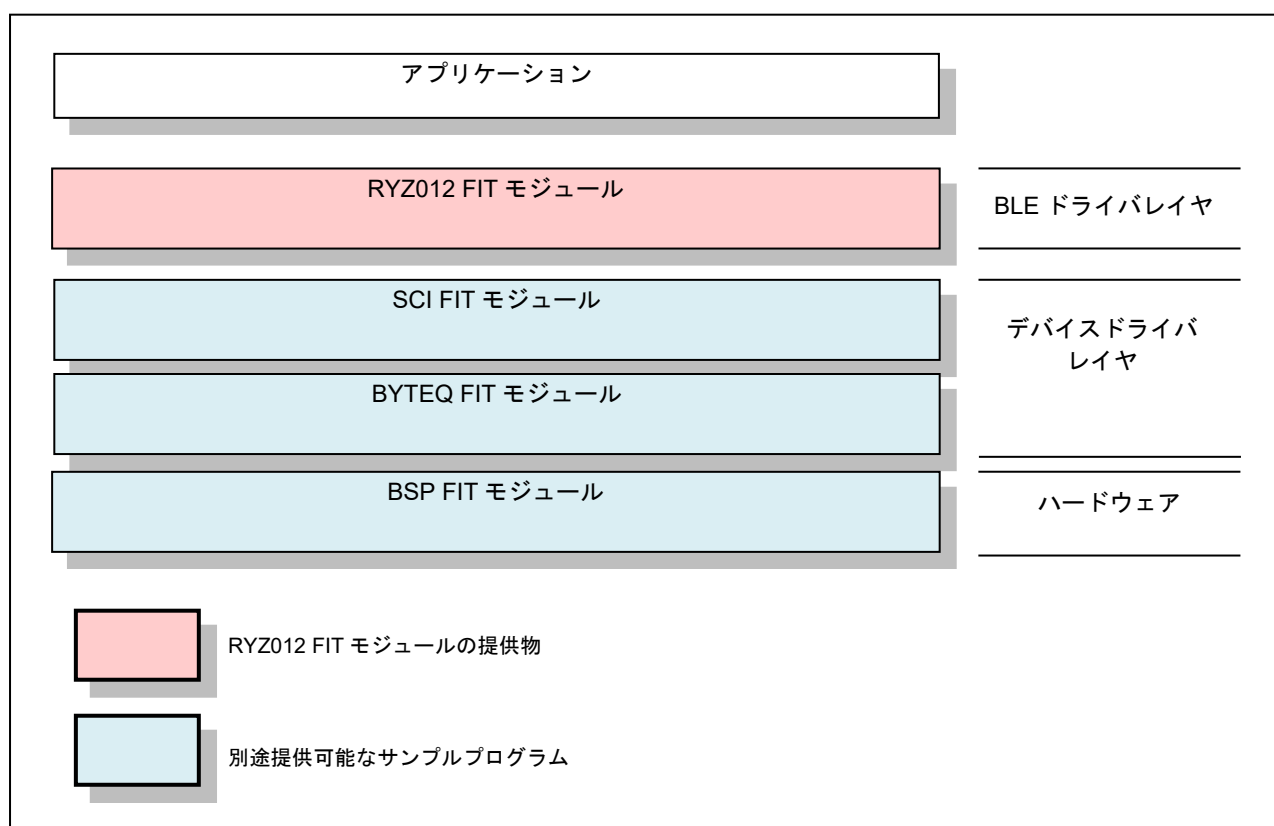


図 1.1 アプリケーション構成図

- (1) RYZ012 FIT モジュール  
本 FIT モジュールです。BLE モジュールを制御するために使用するソフトウェアです。
- (2) SCI FIT モジュール  
BLE モジュールと MCU 間の通信を行います。サンプルプログラムが入手可能です。先頭ページの「関連ドキュメント」を参照し、入手してください。
- (3) BYTEQ FIT モジュール  
シリアルデータのバッファリングを行います。サンプルプログラムが入手可能です。先頭ページの「関連ドキュメント」を参照し、入手してください。

## (4) BSP FIT モジュール

ボードサポートパッケージです。サンプルプログラムが入手可能です。先頭ページの「関連ドキュメント」を参照し、入手してください。

## 1.2.2 ファイル構成

ディレクトリおよびファイルの構成を表 1.1 に示します。

表 1.1 ファイル構成

ファイル/ディレクトリ(太字>名		内容
<b>r_config</b>		
	r_ryz012_rx_config.h	コンフィグレーションヘッダファイル
<b>r_ryz012_rx</b>		
<b>doc</b>		
	<b>英語(en)</b>	アプリケーションノート (英語)格納用フォルダ
	r01an6290ej0101-rx-ryz012-ble.pdf	アプリケーションノート (英語)
	<b>日本語(ja)</b>	アプリケーションノート (日本語)格納用フォルダ
	r01an6290jj0101-rx-ryz012-ble.pdf	アプリケーションノート (日本語)
<b>src</b>		
	hal_data.c	HAL
	hal_data.h	HAL
	r_ble_spp.c	RYZ012 API (通信部)
	r_ble_spp.h	RYZ012 API (ヘッダ)
	r_ble_spp_api.c	RYZ012 API (インタフェース部)
	rm_ble_abs_api.h	QE for BLE 向け API (ヘッダ)
	rm_ble_abs_spp.c	QE for BLE 向け API (処理部)
	wrap_sci.c	SCI ドライバのラッパー
	wrap_sci.h	SCI ドライバのラッパー
	r_ble_api.h	インタフェースヘッダ
	rm_ble_abs.h	QE for BLE 向けのインタフェースヘッダ

## 2. 要件

本 FIT モジュールは、下記条件で動作を確認しています。

---

### 2.1 ハードウェアの要求

---

ご使用になるマイコンが以下の機能をサポートしている必要があります。

- シリアル通信
- I/O ポート

---

### 2.2 ソフトウェアの要求

---

本 FIT モジュールは、以下のパッケージに依存しています。

- r\_bsp
- r\_sci\_rx
- r\_byteq\_rx

---

### 2.3 サポートされているツールチェーン

---

本 FIT モジュールは、11 章 動作確認環境に示すツールチェーンで動作確認を行っています。

---

### 2.4 使用する割り込みベクタ

---

なし

## 2.5 コンフィグレーション設定

本 FIT モジュールのコンフィギュレーション設定は、r\_ryz012\_rx\_config.h と r\_sci\_rx\_config.h で行います。

RYZ012 FIT モジュールに対する設定オプション名および設定値を表 2.1 に、SCI に対する設定オプション名および設定値を表 2.2 に示します。表 2.1 および表 2.2 の設定は、RSKRX65N-2MB を使用する場合の例です。

表 2.1 Configuration options(r\_ryz012\_rx\_config.h)

Configuration options in r_ryz012_rx_config.h	
BLE_CFG_SCI_CHANNEL ※デフォルトは “6”	PMOD-2,3 に割り当てる SCI チャンネル番号を指定します。
BLE_CFG_RESET_PORT ※デフォルトは “F”	PMOD-8 に割り当てる GPIO を設定します。 例) PF5
BLE_CFG_RESET_PIN ※デフォルトは “5”	#define BLE_CFG_RESET_PORT F #define BLE_CFG_RESET_PIN 5
BLE_CFG_SCI_MODE_PORT ※デフォルトは “G”	PMOD-9 に割り当てる GPIO を設定します。 例) PG4
BLE_CFG_SCI_MODE_PIN ※デフォルトは “4”	#define BLE_CFG_SCI_MODE_PORT G #define BLE_CFG_SCI_MODE_PIN 4

表 2.2 Configuration options(r\_sci\_rx\_config.h)

Configuration options in r_sci_rx_config.h	
SCI_CFG_CHx_INCLUDED ※1. CHx = CH0~CH12 ※2. 各デフォルト値は以下のとおり: CH0~CH5、CH7~CH12: 0、CH6: 1	チャンネルごとに送受信バッファ、カウンタ、割り込み、その他のプログラム、RAM などのリソースを持ちます。このオプションを “1” に設定すると、そのチャンネルに関連したリソースが割り当てられます。
SCI_CFG_CHx_TX_BUFSIZ ※1. CHx = CH0~CH12 ※2. 各デフォルト値は (80)	チャンネルごとの送信バッファサイズを指定します。 送信データより十分大きい値を設定してください。
SCI_CFG_CHx_RX_BUFSIZ ※1. CHx = CH0~CH12 ※2. 各デフォルト値は (80)	チャンネルごとの受信バッファサイズを指定します。 受信データより十分大きい値を設定してください。
SCI_CFG_TEI_INCLUDED ※デフォルト値は “0”	シリアル送信の送信完了割り込みを有効にします。 必ず “1” を設定してください。

## 2.6 FIT モジュールの追加方法

本 FIT モジュールは、使用するプロジェクトごとに追加する必要があります。ルネサスでは、Smart Configurator を使用した(1)、(3)の追加方法を推奨しています。ただし、Smart Configurator は、一部の RX デバイスのみサポートしています。サポートされていない RX デバイスについては(2)、(4)の方法を使用してください。

- (1) e<sup>2</sup> studio 上で Smart Configurator を使用して FIT モジュールを追加する場合  
e<sup>2</sup> studio の Smart Configurator を使用して、自動的にユーザプロジェクトに FIT モジュールを追加します。詳細は、アプリケーションノート「Renesas e<sup>2</sup> studio スマート・コンフィグレータ ユーザーガイド (R20AN0451)」を参照してください。
- (2) e<sup>2</sup> studio 上で FIT Configurator を使用して FIT モジュールを追加する場合  
e<sup>2</sup> studio の FIT Configurator を使用して、自動的にユーザプロジェクトに FIT モジュールを追加することができます。詳細は、アプリケーションノート「RX ファミリ e<sup>2</sup> studio に組み込む方法 Firmware Integration Technology (R01AN1723)」および「10 章 QE for BLE を用いたサンプルコード生成」を参照してください。
- (3) CS+上で Smart Configurator を使用して FIT モジュールを追加する場合  
CS+上で、スタンドアロン版 Smart Configurator を使用して、自動的にユーザプロジェクトに FIT モジュールを追加します。詳細は、アプリケーションノート「Renesas e<sup>2</sup> studio スマート・コンフィグレータ ユーザーガイド (R20AN0451)」を参照してください。
- (4) CS+上で FIT モジュールを追加する場合  
CS+上で、手動でユーザプロジェクトに FIT モジュールを追加します。詳細は、アプリケーションノート「RX ファミリ CS+に組み込む方法 Firmware Integration Technology (R01AN1826)」を参照してください。



### 3. BLE Interface (Common)

---

#### 3.1 R\_BLE\_Open()

---

BLE プロトコルスタックをオープンします。

##### Format

```
ble_status_t R_BLE_Open (  
    void  
)
```

##### Parameters

なし

##### Return Values

*BLE\_SUCCESS(0x0000)*

*Success*

##### Properties

r\_ble\_api.h にプロトタイプ宣言されています。

##### Description

BLE プロトコルスタックを使用する前にこの関数を呼び出す必要があります。

##### Reentrant

不可

##### Examples

```
R_BLE_Open();
```

##### Special Notes:

なし

## 3.2 R\_BLE\_Close()

---

BLE プロトコルスタックをクローズします。

### Format

```
ble_status_t R_BLE_Close (  
    void  
)
```

### Parameters

なし

### Return Values

*BLE\_SUCCESS(0x0000)*

*Success*

### Properties

r\_ble\_api.h にプロトタイプ宣言されています。

### Description

BLE プロトコルスタックを閉じるにはこの関数を呼び出す必要があります。

### Reentrant

不可

### Examples

```
R_BLE_Close();
```

### Special Notes:

なし

---

### 3.3 R\_BLE\_Execute()

---

BLE タスクを実行します。

#### Format

```
ble_status_t R_BLE_Execute (  
    void  
)
```

#### Parameters

なし

#### Return Values

*BLE\_SUCCESS(0x0000)*

*Success*

#### Properties

r\_ble\_api.h にプロトタイプ宣言されています。

#### Description

BLE プロトコルスタックの内部タスクキューにキューイングされたすべてのタスクが処理され戻ります。  
この関数はメインループで繰り返し呼び出す必要があります。

#### Reentrant

不可

#### Examples

```
R_BLE_Open();  
  
while(1)  
{  
    R_BLE_Execute();  
}
```

#### Special Notes:

なし

---

### 3.4 R\_BLE\_SetEvent()

---

BLE プロトコルスタックの内部キューにイベントを追加します。

#### Format

```
ble_status_t R_BLE_SetEvent (  
    ble_event_cb_t cb  
)
```

#### Parameters

[in]    *cb*                      The callback for the event.

#### Return Values

<i>BLE_SUCCESS(0x0000)</i>	<i>Success</i>
<i>BLE_ERR_ALREADY_IN_PROGRESS(0x000A)</i>	<i>The event already registered with the callback.</i>
<i>BLE_ERR_CONTEXT_FULL(0x000B)</i>	<i>No free slot for the event.</i>

#### Properties

r\_ble\_api.h にプロトタイプ宣言されています。

#### Description

イベントは、Bluetooth イベントと同様に R\_BLE\_Execute で処理されます。この関数はハードウェア割り込みコンテキストで呼び出されることを目的としています。コールバック関数 *cb* が呼び出される前に同じ *cb* でこの関数を呼び出した場合でも、登録されるイベントは 1 つだけです。一度に登録できるイベントの最大数は 8 つです。

#### Reentrant

不可

#### Examples

なし

#### Special Notes:

なし

## 4. BLE Interface (GAP)

---

### 4.1 R\_BLE\_GAP\_Init()

---

ホストスタックを初期化します。

#### Format

```
ble_status_t R_BLE_GAP_Init (  
    ble_gap_app_cb_t gap_cb  
)
```

#### Parameters

[in] gap\_cb            A callback function registered with this function.

#### Return Values

BLE_SUCCESS(0x0000)	Success
BLE_ERR_INVALID_PTR(0x0001)	gap_cb is specified as NULL.
BLE_ERR_INVALID_STATE(0x0008)	The reason for this error is as follows: <ul style="list-style-type: none"><li>• Host Stack was already initialized.</li><li>• The task for host stack is not running.</li></ul>
BLE_ERR_MEM_ALLOC_FAILED(0x000C)	Insufficient memory is needed to generate this function.

#### Properties

r\_ble\_api.h にプロトタイプ宣言されています。

#### Description

ホストスタックはこの関数で初期化されます。すべての R\_BLEAPI を使用する前にこの関数を呼び出す必要があります。GAP イベントを受信するには、コールバック関数を登録する必要があります。この API 呼び出しの結果は BLE\_GAP\_EVENT\_STACK\_ON イベントで通知されます。

#### Reentrant

不可

#### Examples

なし

#### Special Notes:

なし

---

## 4.2 R\_BLE\_GAP\_SetAdvParam()

---

アドバタイジングパラメータを設定します。

### Format

```
ble_status_t R_BLE_GAP_SetAdvParam (  
    st_ble_gap_adv_param_t * p_adv_param  
)
```

### Parameters

[in] *p\_adv\_param*                      Advertising parameters.

### Return Values

<i>BLE_SUCCESS(0x0000)</i>	Success
<i>BLE_ERR_INVALID_PTR(0x0001)</i>	<i>p_adv_param</i> is specified as NULL.
<i>BLE_ERR_INVALID_ARG(0x0003)</i>	The below <i>p_adv_param</i> field value is out of range. <ul style="list-style-type: none"><li>• <i>adv_handle</i></li><li>• <i>adv_intv_min/adv_intv_max</i></li><li>• <i>adv_ch_map</i></li><li>• <i>o_addr_type</i></li><li>• <i>p_addr_type</i></li><li>• <i>adv_phy</i></li><li>• <i>sec_adv_phy</i></li><li>• <i>scan_req_ntf_flag</i></li></ul>
<i>BLE_ERR_INVALID_STATE(0x0008)</i>	The task for host stack is not running.
<i>BLE_ERR_MEM_ALLOC_FAILED(0x000C)</i>	Insufficient memory is needed to generate this function.

### Properties

*r\_ble\_api.h* にプロトタイプ宣言されています。

### Description

この関数は、アドバタイジングパラメータを設定します。アドバタイジングセットごとにパラメータが異なるアドバタイジングを行うことができます。コントローラに設定されているアドバタイズメントの数は、*BLE\_MAX\_NO\_OF\_ADV\_SETS\_SUPPORTED* として定義されています。各アドバタイジングセットは、アドバタイジングハンドル（0x00-0x03）で識別されます。アドバタイジング開始前にこの機能を使用してアドバタイジングセットを作成し、定期的なアドバタイジングパラメータを設定し、定期的なアドバタイジングを開始し、アドバタイジングデータ/スキャン応答データ/定期的なアドバタイジングデータを設定します。この API 呼び出しの結果は、*BLE\_GAP\_EVENT\_ADV\_PARAM\_SET\_COMP* イベントで通知されます。

**Reentrant**

不可

**Examples**

なし

**Special Notes:**

なし

---

## 4.3 R\_BLE\_GAP\_SetAdvSresData()

---

アドバタイジングデータ/スキャン応答データ/定期的なアドバタイジングデータを設定します。

### Format

```
ble_status_t R_BLE_GAP_SetAdvSresData (  
    st_ble_gap_adv_data_t * p_adv_srsp_data  
)
```

### Parameters

[in] *p\_adv\_srsp\_data*      Advertising data/scan response data/periodic advertising data.

### Return Values

<i>BLE_SUCCESS(0x0000)</i>	Success
<i>BLE_ERR_INVALID_PTR(0x0001)</i>	The reason for this error is as follows: <ul style="list-style-type: none"><li>• <i>p_adv_srsp_data</i> is specified as NULL.</li><li>• <i>data_length</i> field in <i>p_adv_srsp_data</i> parameter is not 0 and <i>p_data</i> field is specified as NULL.</li></ul>
<i>BLE_ERR_INVALID_ARG(0x0003)</i>	The following field in <i>p_adv_srsp_data</i> parameter is out of range. <ul style="list-style-type: none"><li>• <i>adv_hdl</i></li><li>• <i>data_type</i></li><li>• <i>data_length</i></li><li>• <i>zero_length_flag</i></li></ul>
<i>BLE_ERR_INVALID_STATE(0x0008)</i>	The task for host stack is not running.
<i>BLE_ERR_MEM_ALLOC_FAILED(0x000C)</i>	Insufficient memory is needed to generate this function.

### Properties

*r\_ble\_api.h* にプロトタイプ宣言されています。

### Description

この機能は、アドバタイジングデータ/スキャン応答データ/定期的なアドバタイジングデータをアドバタイジングセットに設定します。この関数を呼び出す前に、*R\_BLE\_GAP\_SetAdvParam()*によってアドバタイジングセットを作成する必要があります。データにメモリを割り当てた後、アドバタイジングデータ/スキャン応答データ/定期的なアドバタイジングデータを設定します。



**Reentrant**

不可

**Examples**

なし

**Special Notes:**

なし

---

## 4.4 R\_BLE\_GAP\_StartAdv()

---

アドバタイジングを開始します。

### Format

```
ble_status_t R_BLE_GAP_StartAdv (  
    uint8_t adv_hdl,  
    uint16_t duration,  
    uint8_t max_extd_adv_evts  
)
```

### Parameters

- [in] *adv\_hdl*      The advertising handle pointing to the advertising set which starts advertising.  
The valid range is 0x00 - 0x03.
- [in] *duration*      The duration for which the advertising set identified by *adv\_hdl* is enabled.  
Time = duration \* 10ms. When the duration expires, BLE\_GAP\_EVENT\_ADV\_OFF event notifies that advertising is stopped. The valid range is 0x0000 - 0xFFFF. The duration parameter is ignored when the value is set to 0x0000.
- [in] *max\_extd\_adv\_evts*      The maximum number of advertising events that be sent during advertising.  
When all the advertising events(*max\_extd\_adv\_evts*) have been sent, BLE\_GAP\_EVENT\_ADV\_OFF event notifies that advertising is stopped.  
The *max\_extd\_adv\_evts* parameter is ignored when the value is set to 0x00.

### Return Values

- |                                  |  |
|----------------------------------|--|
| BLE_SUCCESS(0x0000)              | Success  |
| BLE_ERR_INVALID_ARG(0x0003)      | <i>adv_hdl</i> is out of range.                          |
| BLE_ERR_INVALID_STATE(0x0008)    | The task for host stack is not running.                  |
| BLE_ERR_MEM_ALLOC_FAILED(0x000C) | Insufficient memory is needed to generate this function. |

### Properties

r\_ble\_api.h にプロトタイプ宣言されています。

### Description

この関数を呼び出す前に、R\_BLE\_GAP\_SetAdvParam()によって *adv\_hdl* で指定されたアドバタイズメントセットを作成します。この API の呼び出し結果は BLE\_GAP\_EVENT\_ADV\_ON イベントで通知されます。

**Reentrant**

不可

**Examples**

なし

**Special Notes:**

なし

## 4.5 R\_BLE\_GAP\_StopAdv()

---

アドバタイジングを停止します。

### Format

```
ble_status_t R_BLE_GAP_StopAdv (  
    uint8_t adv_hdl  
)
```

### Parameters

[in] *adv\_hdl*            The advertising handle pointing to the advertising set which stops advertising.  
The valid range is 0x00 - 0x03.

### Return Values

*BLE\_SUCCESS(0x0000)*            Success  
*BLE\_ERR\_INVALID\_ARG(0x0003)*    *adv\_hdl* is out of range.  
*BLE\_ERR\_INVALID\_STATE(0x0008)* The task for host stack is not running.  
*BLE\_ERR\_MEM\_ALLOC\_FAILED(0x000C)* Insufficient memory is needed to generate this function.

### Properties

r\_ble\_api.h にプロトタイプ宣言されています。

### Description

この関数はアドバタイジングを停止します。この API の呼び出し結果は、BLE\_GAP\_EVENT\_ADV\_OFF イベントで通知されます。

### Reentrant

不可

### Examples

なし

### Special Notes:

なし

## 5. BLE Interface (GATT Common)

---

### 5.1 R\_BLE\_GATT\_GetMtu()

---

この関数は、GATT 通信で使用されている現在の MTU を取得します。

#### Format

```
ble_status_t R_BLE_GATT_GetMtu(  
    uint16_t conn_hdl,  
    uint16_t * p_mtu  
)
```

#### Parameters

[in] *conn\_hdl*      Connection handle identifying the GATT Server or the GATT Client.  
[in] *p\_mtu*          The Current MTU. Before MTU exchange, this parameter is 23 bytes.  
                      After MTU exchange, this parameter is the negotiated MTU.

#### Return Values

*BLE\_SUCCESS(0x0000)*      Success  
*BLE\_ERR\_INVALID\_PTR(0x0001)*      The mtu parameter is NULL.  
*BLE\_ERR\_INVALID\_HDL(0x000E)*      The GATT Server or the GATT Client specified by *conn\_hdl* was not found.

#### Properties

r\_ble\_api.h にプロトタイプ宣言されています。

#### Description

GATT サーバーと GATT クライアントの両方がこの機能を使用できます。この API 呼び出しの結果は、戻り値によって返されます。

#### Reentrant

不可

#### Examples

なし

#### Special Notes:

なし

## 6. BLE Interface (GATT Server)

---

### 6.1 R\_BLE\_GATTS\_Init()

---

この関数は、GATT サーバーを初期化し、GATT サーバーイベントのコールバックの数を登録します。

#### Format

```
ble_status_t R_BLE_GATTS_Init(  
    uint8_t cb_num  
)
```

#### Parameters

[in]    *cb\_num*                    The number of callbacks to be registered.

#### Return Values

*BLE\_SUCCESS(0x0000)*                    Success  
*BLE\_ERR\_INVALID\_ARG(0x0003)*        The *cb\_num* parameter is out of range.

#### Properties

r\_ble\_api.h にプロトタイプ宣言されています。

#### Description

cb\_num を 1 から BLE\_GATTS\_MAX\_CB までの値に指定し、R\_BLE\_GATTS\_RegisterCb() でコールバックを登録します。この API 呼び出しの結果は、戻り値によって返されます。

#### Reentrant

不可

#### Examples

なし

#### Special Notes:

なし

---

## 6.2 R\_BLE\_GATTS\_SetDbInst()

---

この関数は、GATT データベースをホストスタックに設定します。

### Format

```
ble_status_t R_BLE_GATTS_SetDbInst(  
    st_ble_gatts_db_cfg_t * p_db_inst  
)
```

### Parameters

[in] *p\_db\_inst*      GATT Database to be set.

### Return Values

<i>BLE_SUCCESS(0x0000)</i>	Success
<i>BLE_ERR_INVALID_PTR(0x0001)</i>	The reason for this error is as follows. <ul style="list-style-type: none"><li>• The <i>db_inst</i> parameter is specified as NULL.</li><li>• The array in the <i>db_inst</i> is specified as NULL.</li></ul>

### Properties

*r\_ble\_api.h* にプロトタイプ宣言されています。

### Description

この API 呼び出しの結果は、戻り値によって返されます。

### Reentrant

不可

### Examples

なし

### Special Notes:

なし

---

## 6.3 R\_BLE\_GATTS\_RegisterCb()

---

この関数は、GATT サーバーイベントのコールバックを登録します。

### Format

```
ble_status_t R_BLE_GATTS_RegisterCb(  
    ble_gatts_app_cb_t cb,  
    uint8_t priority  
)
```

### Parameters

<i>[in]</i> <i>cb</i>	<i>Callback function for GATT Server event.</i>
<i>[in]</i> <i>priority</i>	<i>The priority of the callback function.</i> <i>Valid range is 1 &lt;= priority &lt;= BLE_GATTS_MAX_CB.</i> <i>A lower priority number means a higher priority level.</i>

### Return Values

<i>BLE_SUCCESS(0x0000)</i>	<i>Success</i>
<i>BLE_ERR_INVALID_PTR(0x0001)</i>	<i>The cb parameter is specified as NULL.</i>
<i>BLE_ERR_INVALID_ARG(0x0003)</i>	<i>The priority parameter is out of range.</i>
<i>BLE_ERR_CONTEXT_FULL(0x000B)</i>	<i>Host stack has already registered the maximum number of callbacks.</i>

### Properties

r\_ble\_api.h にプロトタイプ宣言されています。

### Description

この関数によって登録される可能性のあるコールバックの番号は、R\_BLE\_GATTS\_Init()によって指定された値です。この API 呼び出しの結果は、戻り値によって返されます。

### Reentrant

不可

### Examples

なし

### Special Notes:

なし



---

## 6.4 R\_BLE\_GATTS\_Notification()

---

アトリビュート値の Notification を送信します。

### Format

```
ble_status_t R_BLE_GATTS_Notification(  
    uint16_t conn_hdl,  
    st_ble_gatt_hdl_value_pair_t * p_ntf_data  
)
```

### Parameters

[in] *conn\_hdl*            Connection handle identifying the remote device to be sent the notification.  
[in] *p\_ntf\_data*        The attribute value to send.

### Return Values

<i>BLE_SUCCESS(0x0000)</i>	Success
<i>BLE_ERR_INVALID_PTR(0x0001)</i>	The <i>p_ntf_data</i> parameter or the value field in the value field in the <i>p_ntf_data</i> parameter is NULL.
<i>BLE_ERR_INVALID_ARG(0x0003)</i>	The <i>value_len</i> field in the value field in the <i>p_ntf_data</i> parameter is 0 or the <i>attr_hdl</i> field in the <i>p_ntf_data</i> parameter is 0.
<i>BLE_ERR_INVALID_OPERATION(0x0009)</i>	This function was called while processing another request.
<i>BLE_ERR_MEM_ALLOC_FAILED(0x000C)</i>	Insufficient memory is needed to generate this function.
<i>BLE_ERR_INVALID_HDL(0x000E)</i>	The remote device specified by <i>conn_hdl</i> was not found.

### Properties

*r\_ble\_api.h* にプロトタイプ宣言されています。

### Description

通知とともに送信できる属性値の最大長は MTU-3 です。この API 呼び出しの結果は、戻り値によって返されます

### Reentrant

不可

### Examples

なし

### Special Notes:

なし

---

## 6.5 R\_BLE\_GATTS\_Indication()

---

アトリビュート値の Indication を送信します。

### Format

```
ble_status_t R_BLE_GATTS_Indication(  
    uint16_t conn_hdl,  
    st_ble_gatt_hdl_value_pair_t * p_ind_data  
)
```

### Parameters

[in] *conn\_hdl*            Connection handle identifying the remote device to be sent the indication.  
[in] *p\_ind\_data*        The attribute value to send.

### Return Values

<i>BLE_SUCCESS(0x0000)</i>	Success
<i>BLE_ERR_INVALID_PTR(0x0001)</i>	The <i>p_ind_data</i> parameter or the value field in the value field in the <i>p_ind_data</i> parameter is NULL.
<i>BLE_ERR_INVALID_ARG(0x0003)</i>	The value_len field in the value field in the <i>p_ind_data</i> parameter is 0 or the attr_hdl field in the <i>p_ind_data</i> parameter is 0.
<i>BLE_ERR_INVALID_OPERATION(0x0009)</i>	This function was called while processing another request.
<i>BLE_ERR_MEM_ALLOC_FAILED(0x000C)</i>	Insufficient memory is needed to generate this function.
<i>BLE_ERR_INVALID_HDL(0x000E)</i>	The remote device specified by <i>conn_hdl</i> was not found.

### Properties

r\_ble\_api.h にプロトタイプ宣言されています。

### Description

表示付きで送信できる属性値の最大長は MTU-3 です。この API 呼び出しの結果は、戻り値によって返されます。指示を受信したリモートデバイスは確認を送信します。BLE\_GATTS\_EVENT\_HDL\_VAL\_CNF イベントは、確認が受信されたことをアプリケーション層に通知します。

### Reentrant

不可

### Examples

なし

### Special Notes:

なし

---

## 6.6 R\_BLE\_GATTS\_GetAttr()

---

アトリビュート値を GATT データベースから取得します。

### Format

```
ble_status_t R_BLE_GATTS_GetAttr(  
    uint16_t conn_hdl,  
    uint16_t attr_hdl,  
    st_ble_gatt_value_t * p_value  
)
```

### Parameters

- [in] *conn\_hdl*      If the attribute value that has information about the remote device is retrieved, specify the remote device with the *conn\_hdl* parameter. When information about the remote device is not required, set the *conn\_hdl* parameter to *BLE\_GAP\_INVALID\_CONN\_HDL*.
- [in] *attr\_hdl*      The attribute handle of the attribute value to be retrieved.
- [out] *p\_value*      The attribute value to be retrieved.

### Return Values

- BLE\_SUCCESS(0x0000)*      Success
- BLE\_ERR\_INVALID\_PTR(0x0001)*      The *p\_value* parameter is specified as NULL.
- BLE\_ERR\_INVALID\_ARG(0x0003)*      The *attr\_hdl* parameter is 0 or larger than the last attribute handle of GATT Database.
- BLE\_ERR\_INVALID\_STATE(0x0008)*      The attribute is not in a state to be read.
- BLE\_ERR\_INVALID\_OPERATION(0x0009)*      The attribute cannot be read.
- BLE\_ERR\_NOT\_FOUND(0x000D)*      The attribute specified by the *attr\_hdl* parameter is not belonging to any services or characteristics.
- BLE\_ERR\_INVALID\_HDL(0x000E)*      The remote device specified by the *conn\_hdl* parameter was not found.

### Properties

*r\_ble\_api.h* にプロトタイプ宣言されています。

### Description

この API 呼び出しの結果は、戻り値によって返されます。

**Reentrant**

不可

**Examples**

なし

**Special Notes:**

なし

---

## 6.7 R\_BLE\_GATTS\_SetAttr()

---

アトリビュート値を GATT データベースに設定します。

### Format

```
ble_status_t R_BLE_GATTS_SetAttr(  
    uint16_t conn_hdl,  
    uint16_t attr_hdl,  
    st_ble_gatt_value_t * p_value  
)
```

### Parameters

- [in] *conn\_hdl*      If the attribute value that has information about the remote device is retrieved, specify the remote device with the *conn\_hdl* parameter. When information about the remote device is not required, set the *conn\_hdl* parameter to *BLE\_GAP\_INVALID\_CONN\_HDL*.
- [in] *attr\_hdl*      The attribute handle of the attribute value to be set.
- [in] *p\_value*      The attribute value to be set.

### Return Values

- BLE\_SUCCESS(0x0000)*      Success
- BLE\_ERR\_INVALID\_PTR(0x0001)*      The *p\_value* parameter is specified as NULL.
- BLE\_ERR\_INVALID\_DATA(0x0002)*      The write size is larger than the length of the attribute value.
- BLE\_ERR\_INVALID\_ARG(0x0003)*      The *attr\_hdl* parameter is 0 or larger than the last attribute handle of GATT Database.
- BLE\_ERR\_INVALID\_STATE(0x0008)*      The attribute is not in a state to be written.
- BLE\_ERR\_INVALID\_OPERATION(0x0009)*      The attribute cannot be written.
- BLE\_ERR\_NOT\_FOUND(0x000D)*      The attribute specified by the *attr\_hdl* parameter is not belonging to any services or characteristics.
- BLE\_ERR\_INVALID\_HDL(0x000E)*      The remote device specified by the *conn\_hdl* parameter was not found.

### Properties

*r\_ble\_api.h* にプロトタイプ宣言されています。

### Description

この API 呼び出しの結果は、戻り値によって返されます。

**Reentrant**

不可

**Examples**

なし

**Special Notes:**

なし

## 7. BLE Interface (GATT Client)

---

### 7.1 R\_BLE\_GATTC\_Init()

---

GATT クライアントを初期化し、GATT クライアントイベントのコールバック数を登録します。

#### Format

```
ble_status_t R_BLE_GATTC_Init(  
    uint8_t cb_num  
)
```

#### Parameters

[in] *cb\_num*            *The number of callbacks to be registered.*

#### Return Values

*BLE\_SUCCESS(0x0000)*            *Success*  
*BLE\_ERR\_INVALID\_ARG(0x0003)*    *The cb\_num parameter is out of range.*

#### Properties

r\_ble\_api.h にプロトタイプ宣言されています。

#### Description

cb\_num パラメータを 1 から BLE\_GATTC\_MAX\_CB までの値に指定します。  
R\_BLE\_GATTC\_RegisterCb()はコールバックを登録します。  
この API 呼び出しの結果は、戻り値によって返されます。

#### Reentrant

不可

#### Examples

なし

#### Special Notes:

なし

---

## 7.2 R\_BLE\_GATTC\_RegisterCb()

---

GATT クライアントイベントのコールバック関数を登録します。

### Format

```
ble_status_t R_BLE_GATTC_RegisterCb(  
    ble_gattc_app_cb_t cb,  
    uint8_t priority  
)
```

### Parameters

<i>[in]</i> <i>cb</i>	<i>Callback function for GATT Client event.</i>
<i>[in]</i> <i>priority</i>	<i>The priority of the callback function.</i> <i>Valid range is 1 &lt;= priority &lt;= BLE_GATTC_MAX_CB.</i> <i>A lower priority number means a higher priority level.</i>

### Return Values

<i>BLE_SUCCESS(0x0000)</i>	<i>Success</i>
<i>BLE_ERR_INVALID_PTR(0x0001)</i>	<i>The cb parameter is specified as NULL.</i>
<i>BLE_ERR_INVALID_ARG(0x0003)</i>	<i>The priority parameter is out of range.</i>
<i>BLE_ERR_CONTEXT_FULL(0x000B)</i>	<i>Host stack has already registered the maximum number of callbacks.</i>

### Properties

r\_ble\_api.h にプロトタイプ宣言されています。

### Description

この関数によって登録される可能性のあるコールバックの番号は、R\_BLE\_GATTC\_Init()によって指定された値です。この API 呼び出しの結果は、戻り値によって返されます。

### Reentrant

不可

### Examples

なし

### Special Notes:

なし



## 8. BLE Interface (Vendor Specific)

---

### 8.1 R\_BLE\_VS\_SetTxPower()

---

この関数は送信電力を設定します。

#### Format

```
ble_status_t R_BLE_VS_SetTxPower(  
    uint16_t conn_hdl,  
    uint8_t tx_power  
)
```

#### Parameters

- [in] *conn\_hdl*      Connection handle identifying the link whose transmit power to be configured.  
If non connected state, set BLE\_GAP\_INIT\_CONN\_HDL (0xFFFF).
- [in] *tx\_power*      Transmission power. Select one of the following.
- BLE\_VS\_TX\_POWER\_HIGH (0x00)
  - BLE\_VS\_TX\_POWER\_MID (0x01)
  - BLE\_VS\_TX\_POWER\_LOW (0x02)

#### Return Values

- |                                   |  |
|-----------------------------------|--|
| BLE_SUCCESS (0x0000)              | Success  |
| BLE_ERR_INVALID_STATE (0x0008)    | The task for host stack is not running.            |
| BLE_ERR_MEM_ALLOC_FAILED (0x000C) | There are no memories for Vendor Specific Command. |

#### Properties

r\_ble\_api.h にプロトタイプ宣言されています。

#### Description

この関数は次の送信電力を構成します。

- ・ アドバタイジング PDU、スキャン要求 PDU、接続要求 PDU（未接続状態）
- ・ PDU を送信する際に使用される送信電力。未接続状態で送信電力を構成する場合は、conn\_hdl パラメータを BLE\_GAP\_INIT\_CONN\_HDL（0xFFFF）に設定します。

接続状態で使用する送信電力を設定する場合は、conn\_hdl パラメータをリンクの接続ハンドルに設定してください。次の送信電力レベルのいずれかを選択します。（High・Middle・Low）

**Reentrant**

不可

**Examples**

なし

**Special Notes:**

なし

## 8.2 R\_BLE\_VS\_SetBdAddr()

ローカルデバイスのパブリック/ランダムアドレスをパラメータで指定された領域に設定します。

### Format

```
ble_status_t R_BLE_VS_SetBdAddr(  
    uint8_t area,  
    st_ble_dev_addr_t * p_addr  
)
```

### Parameters

[in] area	The area that the address is to be written in.
[in] p_addr	The address to be set to the area. Set BLE_GAP_ADDR_PUBLIC(0x00) or BLE_GAP_ADDR_RAND(0x01) to the type field in the p_addr parameter.

### Return Values

BLE_SUCCESS(0x0000)	Success
BLE_ERR_INVALID_PTR(0x0001)	The p_addr parameter is specified as NULL.
BLE_ERR_INVALID_STATE(0x0008)	The task for host stack is not running.
BLE_ERR_MEM_ALLOC_FAILED(0x000C)	There are no memories for Vendor Specific Command.

### Properties

r\_ble\_api.h にプロトタイプ宣言されています。

### Description

アドレスが不揮発性領域に書き込まれている場合、そのアドレスは次の MCU リセットでデフォルトアドレスとして使用されます。ランダムアドレスの詳細については、コア仕様 Vol 6, PartB, “1.3.2 Random Device Address”を参照してください。この API 呼び出しの結果は、BLE\_VS\_EVENT\_SET\_ADDR\_COMP イベントで通知されます。

### Reentrant

不可

### Examples

なし

### Special Notes:

なし

## 9. Abstraction API for Renesas QE for BLE

---

### 9.1 RM\_BLE\_ABS\_Open()

---

ホストスタックを初期化します。

#### Format

```
fsp_err_t RM_BLE_ABS_Open (  
    ble_abs_ctrl_t *const    p_ctrl,  
    ble_abs_cfg_t const *const p_cfg  
)
```

#### Parameters

[in] *p\_ctrl*                      *Pointer to control structure.*  
[in] *p\_cfg*                        *Pointer to the configuration structure for this instance.*

#### Return Values

<i>FSP_SUCCESS</i>	<i>Channel opened successfully.</i>
<i>FSP_ERR_ASSERTION</i>	<i>Null pointer presented.</i>
<i>FSP_ERR_INVALID_CHANNEL</i>	<i>The channel number is invalid.</i>
<i>FSP_ERR_ALREADY_OPEN</i>	<i>Requested channel is already open in a different configuration.</i>
<i>FSP_ERR_INVALID_ARGUMENT</i>	<i>Invalid input parameter.</i>

#### Properties

rm\_ble\_abs.h にプロトタイプ宣言されています。

#### Description

すべての R\_BLEAPI を使用する前に、この関数を呼び出す必要があります。コールバック関数はこの関数に登録されています。GAP、GATT、ベンダー固有のイベントを受信するには、コールバック関数に登録する必要があります。この API 呼び出しの結果は、BLE\_GAP\_EVENT\_STACK\_ON イベントで通知されます。ble\_abs\_api\_t::open を実装します。

## Reentrant

不可

## Examples

```
/* Open the module. */  
err = RM_BLE_ABS_Open(&g_ble_abs0_ctrl, &g_ble_abs0_cfg);
```

## Special Notes:

なし

---

## 9.2 RM\_BLE\_ABS\_Close()

---

BLE チャンネルを閉じます。

### Format

```
fsp_err_t RM_BLE_ABS_Close (  
    ble_abs_ctrl_t * const p_ctrl  
)
```

### Parameters

[in] *p\_ctrl*                      *Pointer to control structure.*

### Return Values

<i>FSP_SUCCESS</i>	<i>Channel closed successfully.</i>
<i>FSP_ERR_ASSERTION</i>	<i>Null pointer presented.</i>
<i>FSP_ERR_NOT_OPEN</i>	<i>Control block not open.</i>

### Properties

rm\_ble\_abs.h にプロトタイプ宣言されています。

### Description

ble\_abs\_api\_t::close を実装します。

### Reentrant

不可

### Examples

```
/* Close BLE driver */  
err = RM_BLE_ABS_Close(&g_ble_abs0_ctrl);
```

### Special Notes:

なし

### 9.3 RM\_BLE\_ABS\_StartLegacyAdvertising()

アドバタイジングパラメータ、アドバタイジングデータ、スキャン応答データを設定した後、レガシーアドバタイジングを開始します。

#### Format

```
fsp_err_t RM_BLE_ABS_StartLegacyAdvertising (
    ble_abs_ctrl_t * const      p_ctrl,
    ble_abs_legacy_advertising_parameter_t const * const p_advertising_parameter
)
```

#### Parameters

<i>[in]</i> <i>p_ctrl</i>	<i>Pointer to control structure.</i>
<i>[in]</i> <i>p_advertising_parameter</i>	<i>Pointer to Advertising parameters for Legacy Advertising.</i>

#### Return Values

<i>FSP_SUCCESS</i>	<i>Operation succeeded</i>
<i>FSP_ERR_ASSERTION</i>	<i>p_instance_ctrl is specified as NULL.</i>
<i>FSP_ERR_NOT_OPEN</i>	<i>Control block not open.</i>
<i>FSP_ERR_INVALID_STATE</i>	<i>Host stack hasn't been initialized.</i>
<i>FSP_ERR_INVALID_POINTER</i>	<i>p_advertising_parameter is specified as NULL.</i>
<i>FSP_ERR_INVALID_ARGUMENT</i>	<i>The advertising parameter is out of range.</i>

#### Properties

rm\_ble\_abs.h にプロトタイプ宣言されています。

#### Description

従来のアドバタイズメントは、アドバタイジングハンドルが 0 のアドバタイズメントセットを使用します。アドバタイズメントタイプは接続可能およびスキャン可能 (ADV\_IND) です。ローカルデバイスのアドレスタイプは、パブリック ID アドレスまたは RPA です (解決リストに一致するエントリが含まれていない場合は、パブリックアドレスを使用してください)。スキャン要求イベント

(BLE\_GAP\_EVENT\_SCAN\_REQ\_RECV) は通知されません。ble\_abs\_api\_t :: startLegacyAdvertising を実装します。

## Reentrant

不可

## Examples

```
/* Start advertising. */  
err = RM_BLE_ABS_StartLegacyAdvertising(&g_ble_abs0_ctrl,  
&legacy_advertising_parameter);
```

## Special Notes:

なし



### 10. QE for BLE を用いたサンプルコード生成

e<sup>2</sup> studio で新規プロジェクトを作成し、Smart Configurator を用いて本 FIT モジュールを組み込む方法、および QE for BLE を用いてサンプルコードを生成する方法を以下に示します。本章で示す手順における設定値は、RSKRX65N-2MB を用いる場合の例です。

1. 本 FIT モジュールを以下のフォルダにコピーし、Smart Configurator でコンポーネント追加可能にする。

コピーするファイル :

r\_ryz012\_rx\_v1.00.xml

r\_ryz012\_rx\_v1.00.zip

r\_ryz012\_rx\_v1.00\_extend.mdf

コピー先 :

C:\¥Users¥<ユーザ名>¥.eclipse¥com.renesas.platform\_download¥FITModules¥

## 2. e<sup>2</sup> studio で新規プロジェクトを作成する。

- ・ Renesas CC-RX C/C++ Executable Project
- ・ プロジェクト名：任意
- ・ RTOS：None（なし）
- ・ Target Board：RSKR65N-2MB ※TSIP 搭載（R5F565NEHxFC）の場合も同様です

**New Renesas CC-RX Executable Project**  
Select toolchain, device debug settings

**Toolchain Settings**

言語: ☒ C ☐ C++

ツールチェーン: Renesas CCRX

ツールチェーン・バージョン: v3.03.00

RTOS: None

RTOS Version:

**Device Settings**

Target Board: RSKRX65N-2MB

ターゲット・デバイス: R5F565NEDxFC

エンディアン: Little

プロジェクト・タイプ: デフォルト

**Configurations**

☒ Hardware Debug 構成を生成

☐ Debug 構成を生成

☐ Release 構成を生成

E1 (RX)

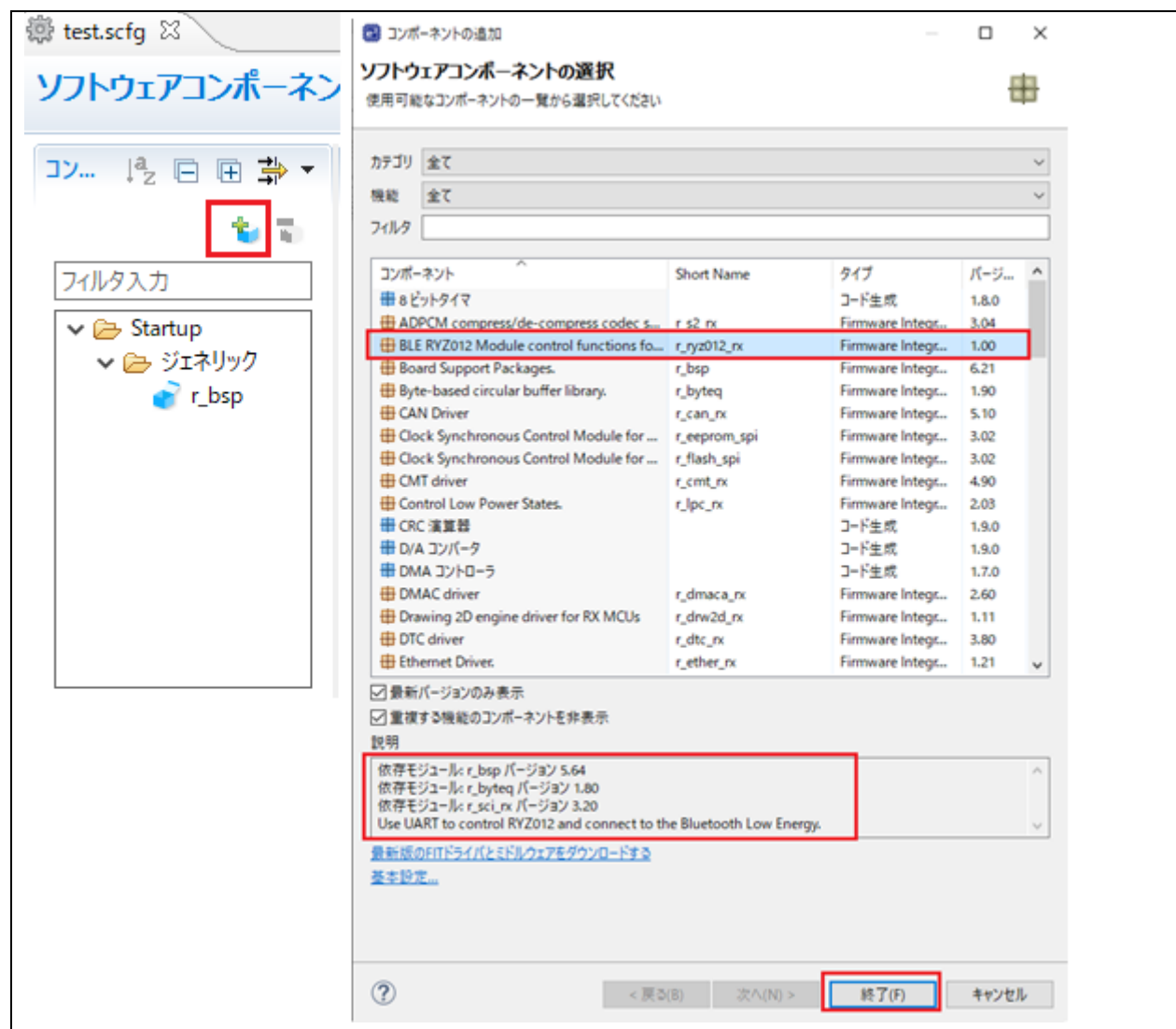
RX Simulator

< 戻る(B) **次へ(N) >** 終了(F) キャンセル

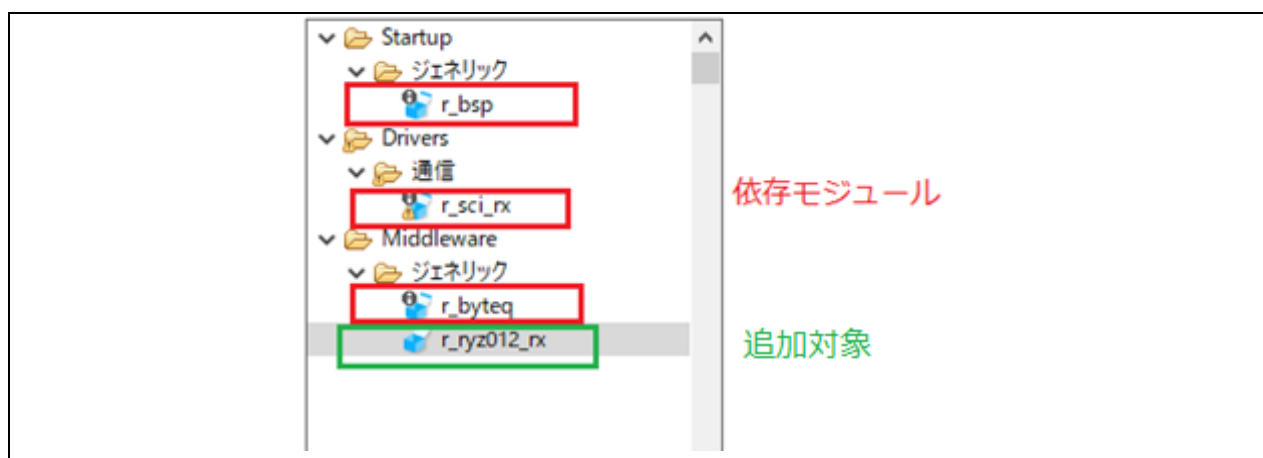
3. 「Use Smart Configurator」をチェックし「終了」を押す。



4. Smart Configurator パースペクティブの「コンポーネント」タブの「コンポーネントの追加」ボタンを押し、本 FIT モジュールが一覧に表示されていることを確認する。表示されていれば選択して「終了」を押す。

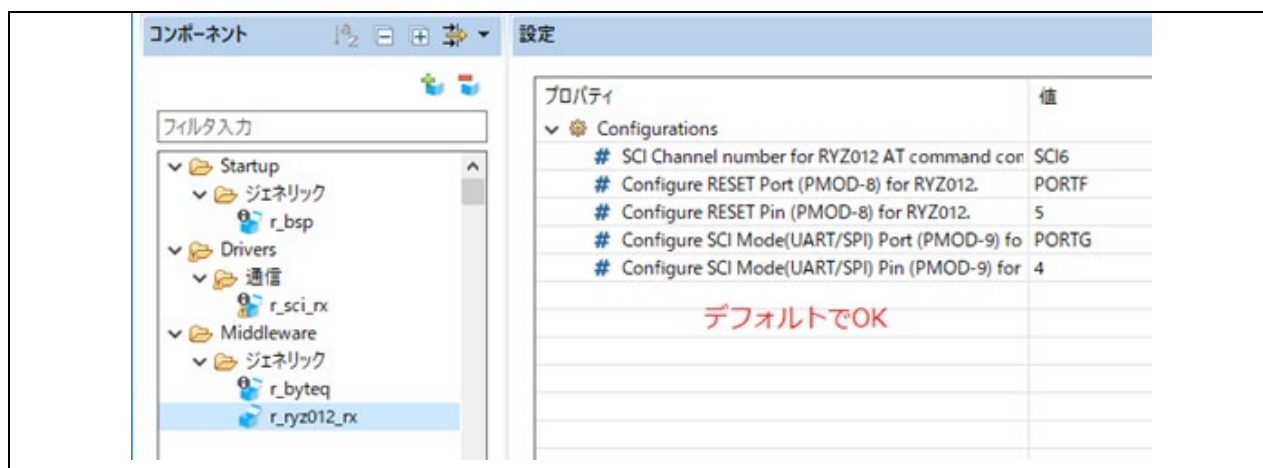


モジュールが追加されていることを確認する。

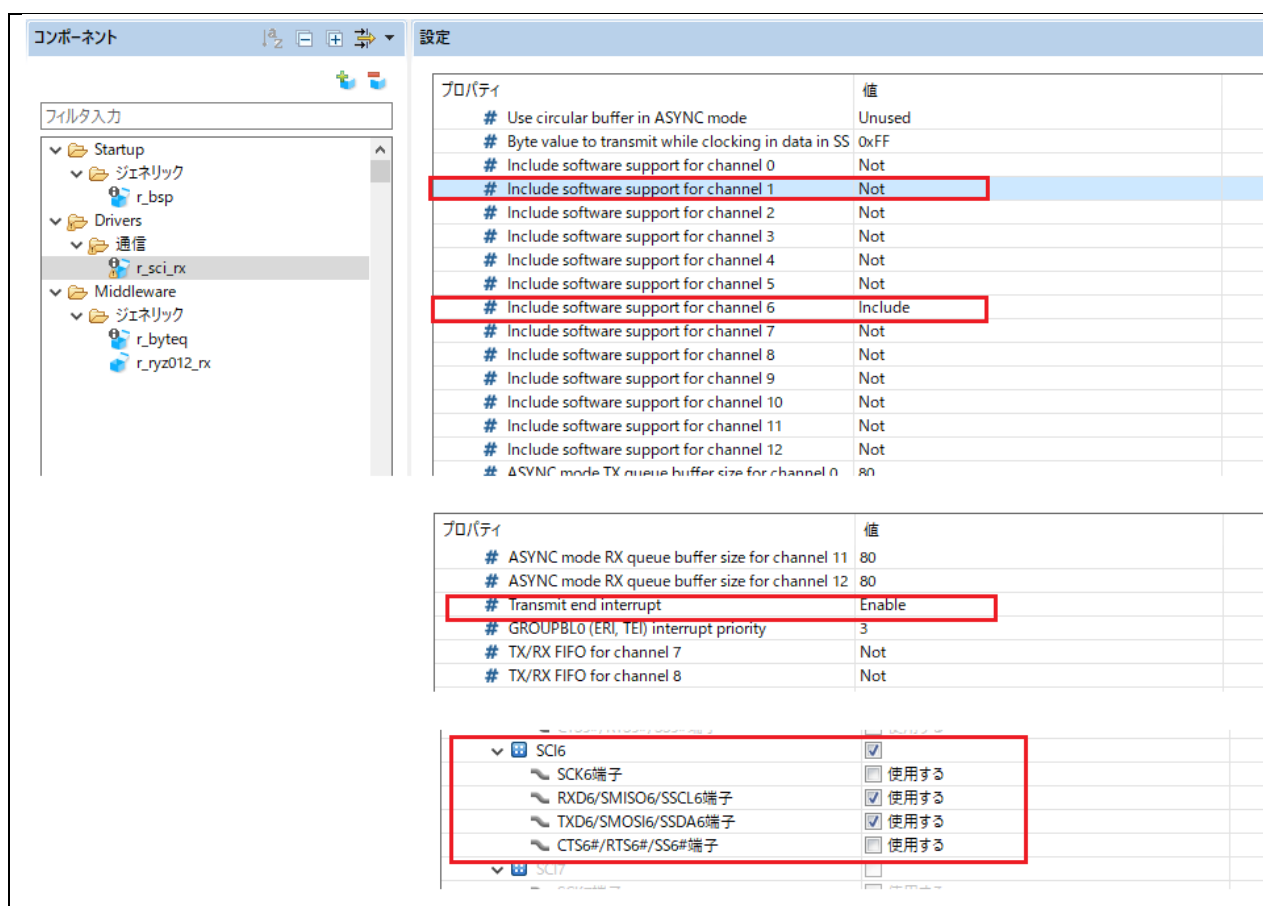


## 5. コンフィグを設定する

r\_ryz012\_rx



r\_sci\_rx (RSKR65N-2MB を使用する場合)



## 端子設定

端子機能

フィルタ入力 (\* = any string, ? = any character)

使用する	機能	端子割り当て	端子番号	方向
<input type="checkbox"/>	CTS6#	/ 設定されていません	/ 設定されていま	なし
<input type="checkbox"/>	RTS6#	/ 設定されていません	/ 設定されていま	なし
<input type="checkbox"/>	RXD6	/ 設定されていません	/ 設定されていま	なし
<input type="checkbox"/>	SCK6	/ 設定されていません	/ 設定されていま	なし
<input checked="" type="checkbox"/>	SMISO6	/ P01/TMCI0/RXD6/SMISO6/SSCL6/IRQ9/AN119	/ 7	IO
<input checked="" type="checkbox"/>	SMOSI6	/ P00/TMRI0/TXD6/SMOSI6/SSDA6/IRQ8/AN118	/ 8	IO
<input type="checkbox"/>	SS6#	/ 設定されていません	/ 設定されていま	なし
<input type="checkbox"/>	SSCL6	/ 設定されていません	/ 設定されていま	なし
<input type="checkbox"/>	SSDA6	/ 設定されていません	/ 設定されていま	なし
<input type="checkbox"/>	TXD6	/ 設定されていません	/ 設定されていま	なし

端子機能 端子番号

概要 | ボード | クロック | システム | コンポーネント | 端子 | 割り込み

6. 「コードの生成」を押す。¥smc\_gen 以下に追加した FIT モジュールが生成される。

コードの生成 レポートの生成

any character) すべて

端子割り当て	端子番号	方向
/ 設定されていません	/ 設定されていま	なし
/ 設定されていません	/ 設定されていま	なし
/ P01/TMCI0/RXD6/SMISO6/SSCL6/IRQ9/AN119	/ 7	I
/ 設定されていません	/ 設定されていま	なし
/ 設定されていません	/ 設定されていま	なし
/ 設定されていません	/ 設定されていま	なし
/ 設定されていません	/ 設定されていま	なし
/ 設定されていません	/ 設定されていま	なし
/ 設定されていません	/ 設定されていま	なし
/ P00/TMRI0/TXD6/SMOSI6/SSDA6/IRQ8/AN118	/ 8	O

Project Structure:

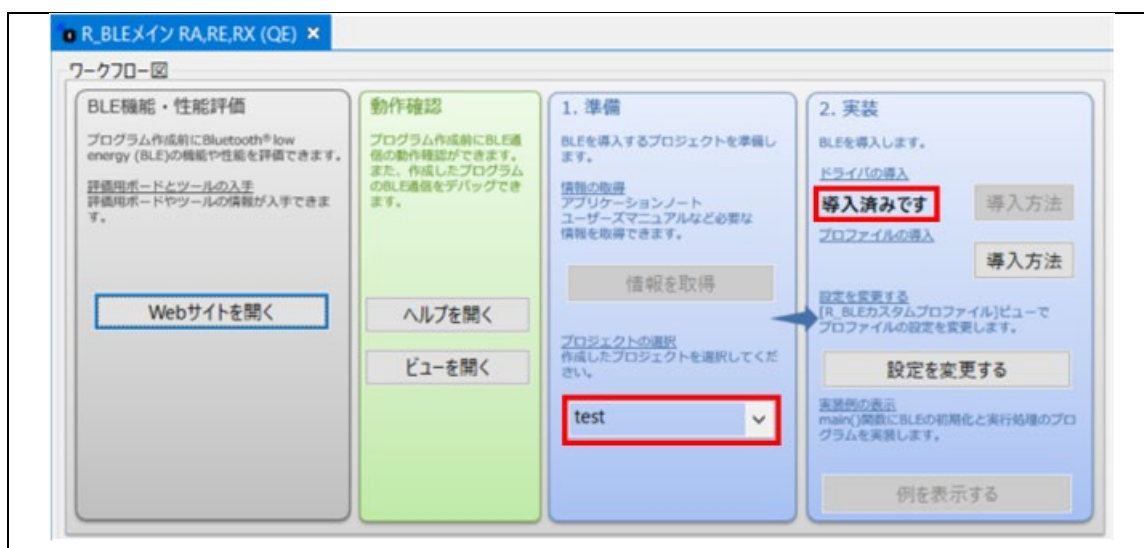
- ra6m5a\_base
  - rx65n\_ck\_ryz012
  - rx65n\_rsk\_fsp350\_2
  - test
    - Includes
    - src
      - smc\_gen
        - general
        - r\_bsp
        - r\_byteq**
        - r\_config
        - r\_pincfg
        - r\_ryz012\_rx**
        - r\_sci\_rx**
      - test.c
    - trash
    - test.scfg
    - test HardwareDebug.launch

# 7. QE for BLE を起動し、コードを生成する。

## (1) R\_BLE メインタブを開く。



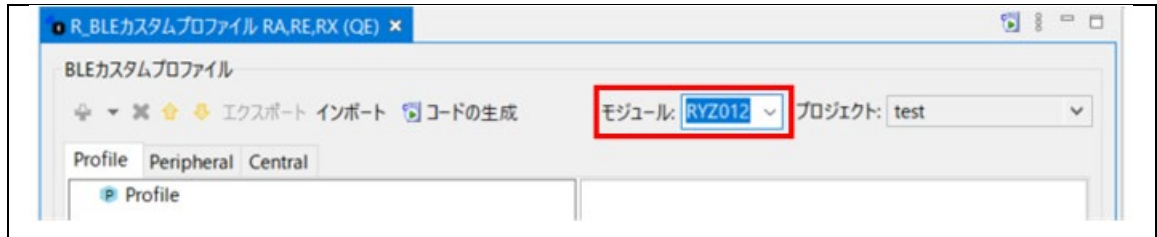
## (2) プロジェクトを選択する。 ドライバが導入済みとなっていることを確認する。



## (3) R\_BLE カスタムプロファイルタブを開く。



(4) 「モジュール」 ドロップダウンリストで「RYZ012」を選択する。

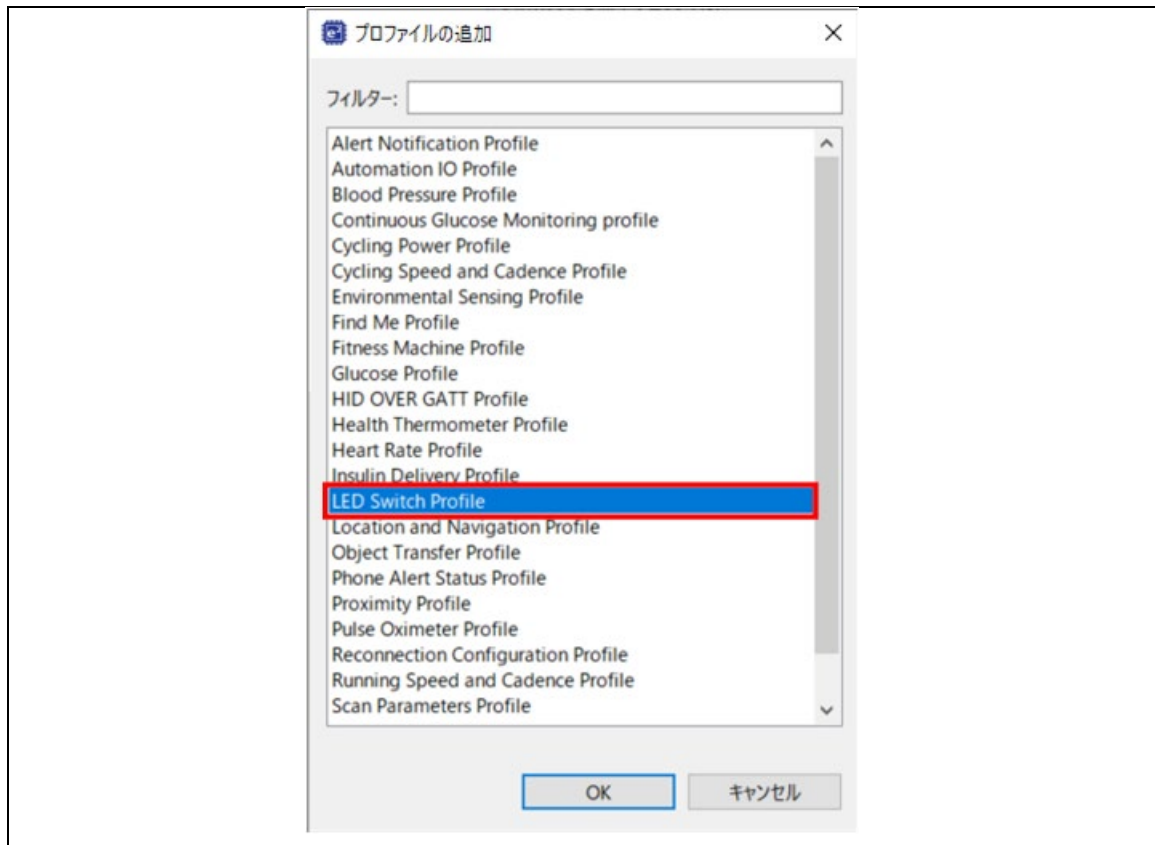


(5) 「プロファイルを追加」 ウィンドウを開く。

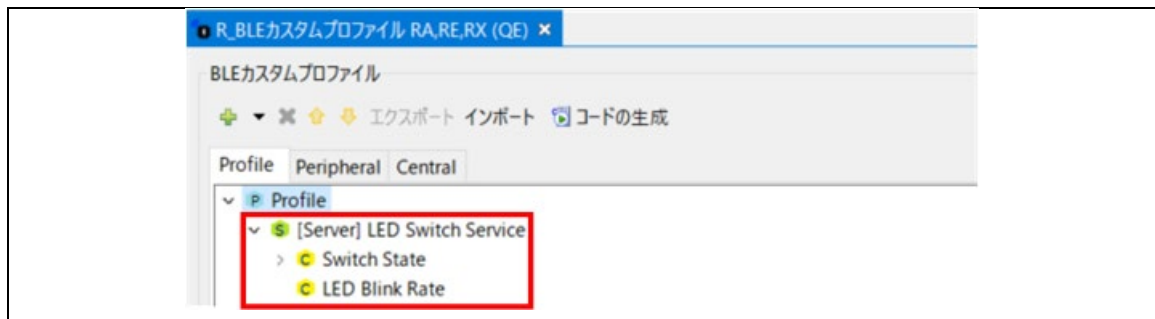




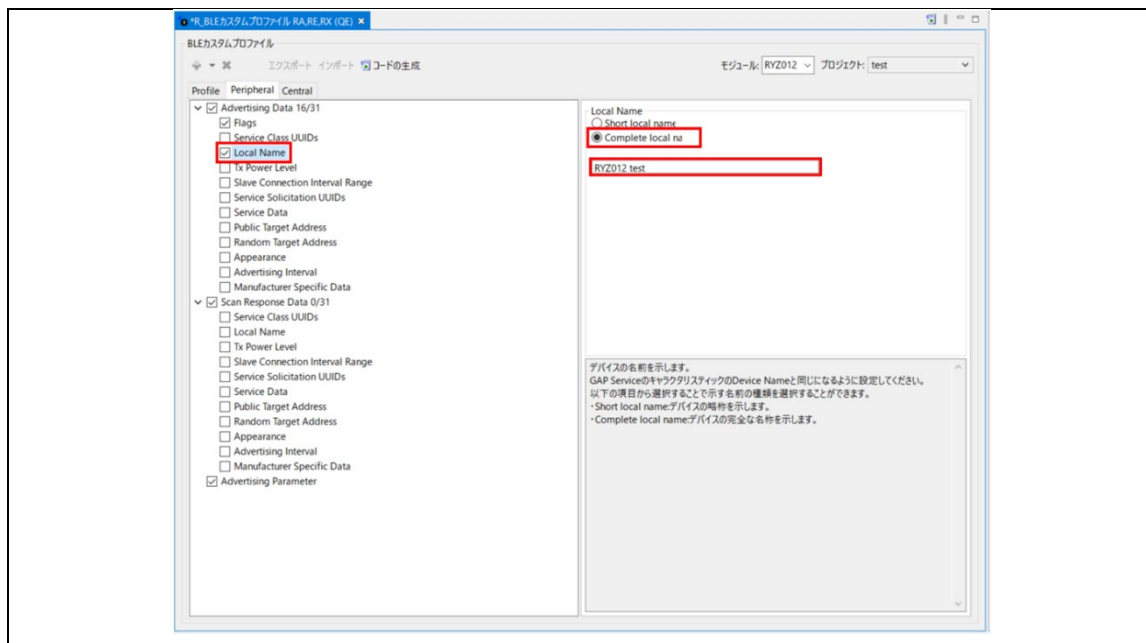
(6) 「LED Switch Profile」を選択する。



Profile が追加される。



- (7) 「Peripheral」タブで「Local Name」チェックボックスをチェックし、「Complete local name」を選択、テキストボックスへデバイス名となる文字列を入力する。(例では「RYZ012 test」と入力)



- (8) 「コードの生成」をクリックし、QE のコード生成を実行する。



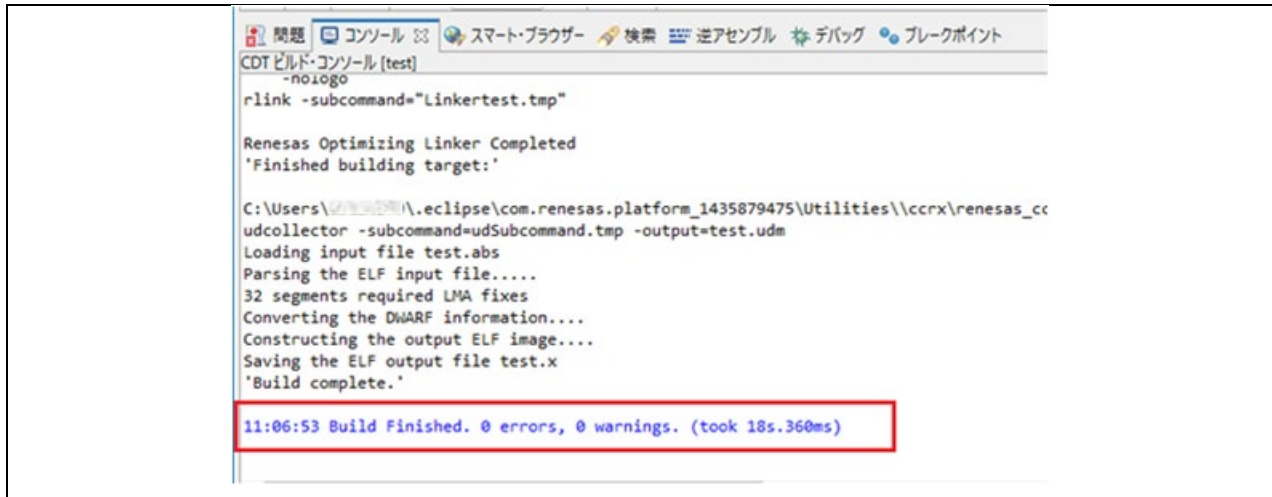
8. main 関数のコードを、以下の通り修正する。

```
#include "r_smc_entry.h"

void main(void);
extern void app_main(void); ※追加

void main(void)
{
    app_main(); ※追加
}
```

## 9. ビルドしてエラーがないことを確認する。



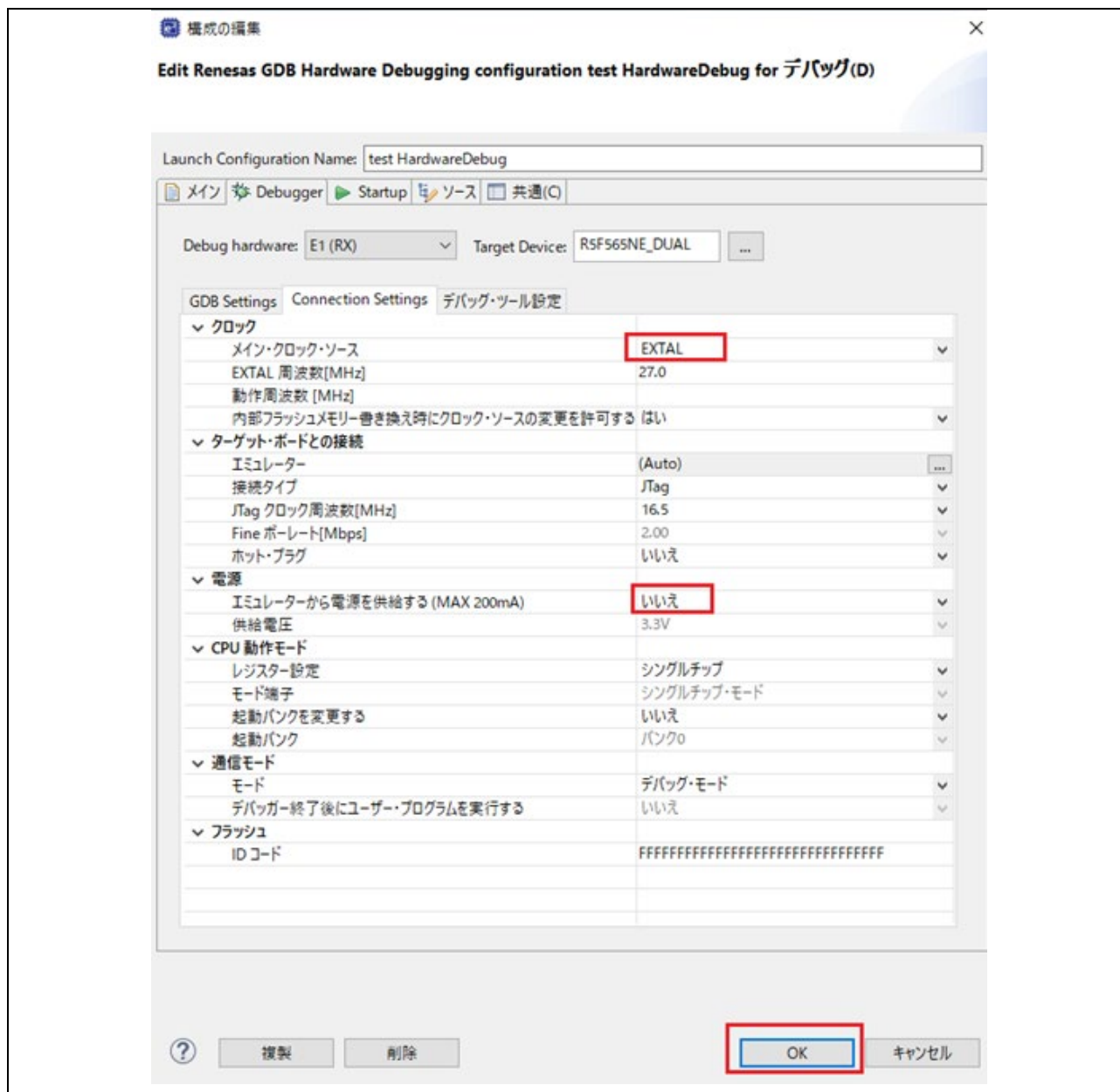
```
CDTビルド・コンソール [test]
-noiogo
rlink -subcommand="Linkertest.tmp"

Renesas Optimizing Linker Completed
'Finished building target:'

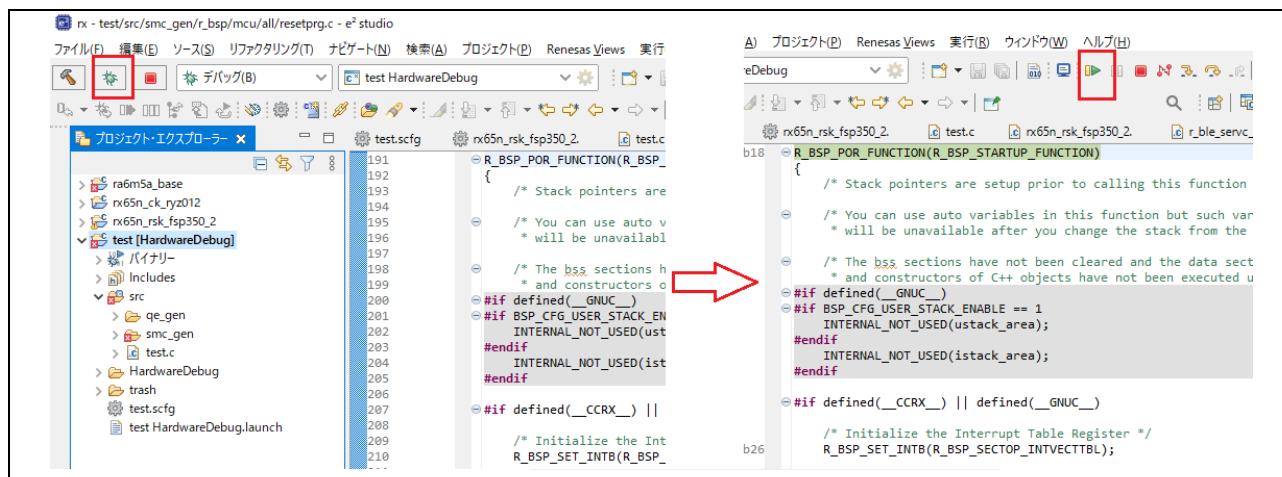
C:\Users\...\eclipse\com.renesas.platform_1435879475\Utilities\ccrx\renesas_cc
udcollector -subcommand=udSubcommand.tmp -output=test.udm
Loading input file test.abs
Parsing the ELF input file.....
32 segments required LMA fixes
Converting the DWARF information....
Constructing the output ELF image....
Saving the ELF output file test.x
'Build complete.'
```

11:06:53 Build Finished. 0 errors, 0 warnings. (took 18s.360ms)

10. デバッグ設定を以下の通りに変更し「OK」を押す。

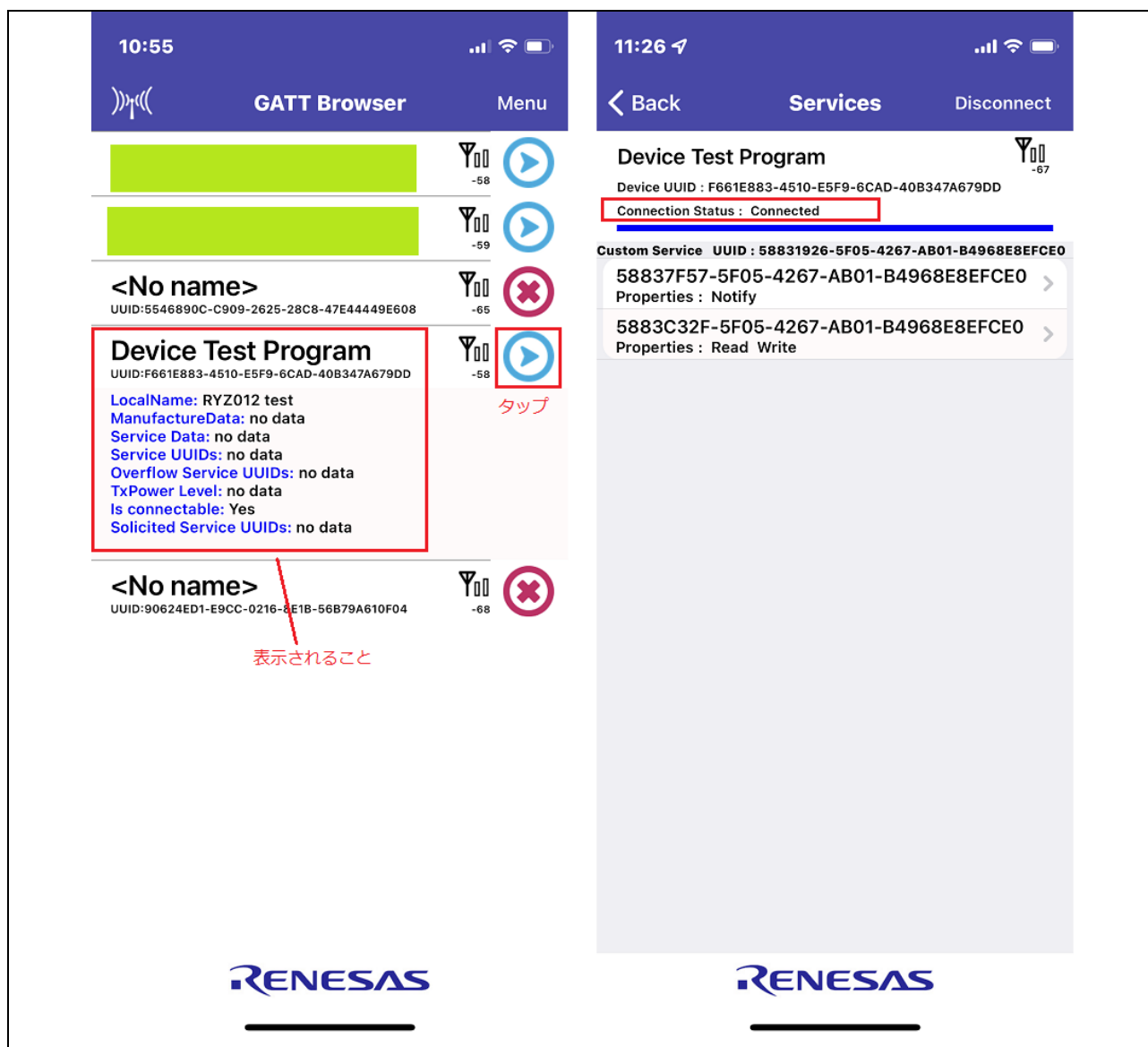


## 11. プログラムを実行。

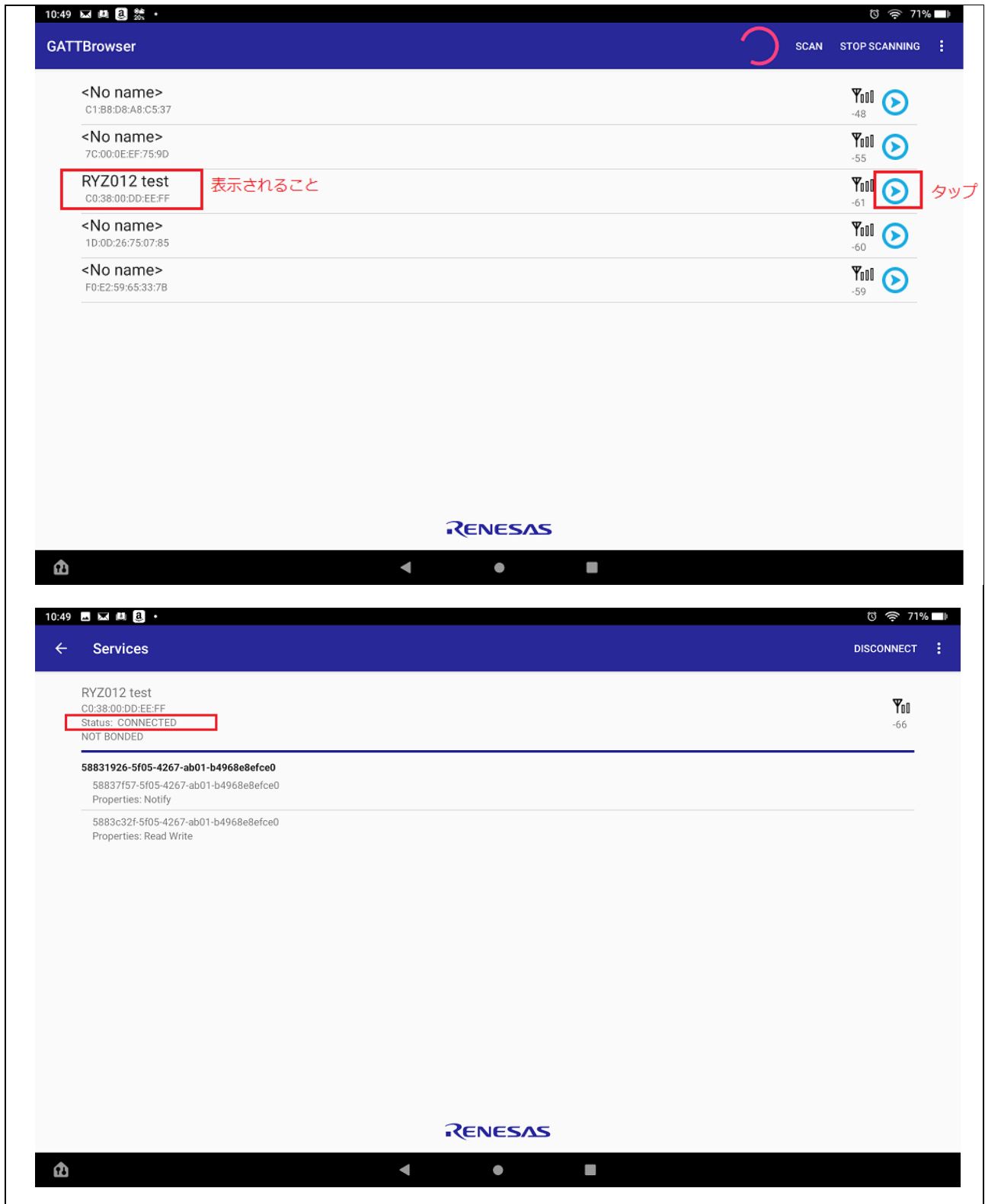


## 12. スマートフォンのアプリ（Renesas GATT Browser）から接続する。

iOS 版



## Android 版



## 11. 動作確認環境

本 FIT モジュールの動作確認環境を以下に示します。

表 11.1 動作確認環境

項目	内容
統合開発環境	ルネサスエレクトロニクス製 e <sup>2</sup> studio 2021.10
C コンパイラ	ルネサスエレクトロニクス製 C/C++ Compiler for RX Family V3.03.00
	コンパイルオプション：統合開発環境のデフォルト設定に以下のオプションを追加 -lang = c99
エンディアン	リトルエンディアン
モジュールのリビジョン	Rev1.00
使用ボード	Renesas Starter Kit+ for RX65N-2MB（型名：RTK50565N2CxxxxxBR）

## 12. 参考ドキュメント

ユーザーズマニュアル：ハードウェア

（最新版をルネサス エレクトロニクスホームページから入手してください。）

テクニカルアップデート／テクニカルニュース

（最新の情報をルネサス エレクトロニクスホームページから入手してください。）

ユーザーズマニュアル：開発環境

RX ファミリ CC-RX コンパイラ ユーザーズマニュアル（R20UT3248）

（最新版をルネサス エレクトロニクスホームページから入手してください。）



ホームページとサポート窓口

ルネサス エレクトロニクスホームページ

<http://japan.renesas.com>

お問合せ先

<http://japan.renesas.com/contact/>

すべての商標および登録商標は、それぞれの所有者に帰属します。

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	2022/01/28	-	新規作成
1.01	2022/03/31	5	アプリケーションノート英語版を追加

## 製品ご使用上の注意事項

ここでは、マイコン製品全体に適用する「使用上の注意事項」について説明します。個別の使用上の注意事項については、本ドキュメントおよびテクニカルアップデートを参照してください。

### 1. 静電気対策

CMOS 製品の取り扱いの際は静電気防止を心がけてください。CMOS 製品は強い静電気によってゲート絶縁破壊を生じることがあります。運搬や保存の際には、当社が出荷梱包に使用している導電性のトレーやマガジンケース、導電性の緩衝材、金属ケースなどを利用し、組み立て工程にはアースを施してください。プラスチック板上に放置したり、端子を触ったりしないでください。また、CMOS 製品を実装したボードについても同様の扱いをしてください。

### 2. 電源投入時の処置

電源投入時は、製品の状態は不定です。電源投入時には、LSI の内部回路の状態は不確定であり、レジスタの設定や各端子の状態は不定です。外部リセット端子でリセットする製品の場合、電源投入からリセットが有効になるまでの期間、端子の状態は保証できません。同様に、内蔵パワーオンリセット機能を使用してリセットする製品の場合、電源投入からリセットのかかる一定電圧に達するまでの期間、端子の状態は保証できません。

### 3. 電源オフ時における入力信号

当該製品の電源がオフ状態のときに、入力信号や入出力ブルアップ電源を入れないでください。入力信号や入出力ブルアップ電源からの電流注入により、誤動作を引き起こしたり、異常電流が流れ内部素子を劣化させたりする場合があります。資料中に「電源オフ時における入力信号」についての記載のある製品は、その内容を守ってください。

### 4. 未使用端子の処理

未使用端子は、「未使用端子の処理」に従って処理してください。CMOS 製品の入力端子のインピーダンスは、一般に、ハイインピーダンスとなっています。未使用端子を開放状態で動作させると、誘導現象により、LSI 周辺のノイズが印加され、LSI 内部で貫通電流が流れたり、入力信号と認識されて誤動作を起こす恐れがあります。

### 5. クロックについて

リセット時は、クロックが安定した後、リセットを解除してください。プログラム実行中のクロック切り替え時は、切り替え先クロックが安定した後、に切り替えてください。リセット時、外部発振子（または外部発振回路）を用いたクロックで動作を開始するシステムでは、クロックが十分安定した後、リセットを解除してください。また、プログラムの途中で外部発振子（または外部発振回路）を用いたクロックに切り替える場合は、切り替え先のクロックが十分安定してから切り替えてください。

### 6. 入力端子の印加波形

入力ノイズや反射波による波形歪みは誤動作の原因になりますので注意してください。CMOS 製品の入力がノイズなどに起因して、 $V_{IL}$  (Max.) から  $V_{IH}$  (Min.) までの領域にとどまるような場合は、誤動作を引き起こす恐れがあります。入力レベルが固定の場合はもちろん、 $V_{IL}$  (Max.) から  $V_{IH}$  (Min.) までの領域を通過する遷移期間中にチャタリングノイズなどが入らないように使用してください。

### 7. リザーブアドレス（予約領域）のアクセス禁止

リザーブアドレス（予約領域）のアクセスを禁止します。アドレス領域には、将来の拡張機能用に割り付けられている リザーブアドレス（予約領域）があります。これらのアドレスをアクセスしたときの動作については、保証できませんので、アクセスしないようにしてください。

### 8. 製品間の相違について

型名の異なる製品に変更する場合は、製品型名ごとにシステム評価試験を実施してください。同じグループのマイコンでも型名が違くと、フラッシュメモリ、レイアウトパターンの相違などにより、電気的特性の範囲で、特性値、動作マージン、ノイズ耐量、ノイズ輻射量などが異なる場合があります。型名が違う製品に変更する場合は、個々の製品ごとにシステム評価試験を実施してください。

## ご注意書き

1. 本資料に記載された回路、ソフトウェアおよびこれらに関連する情報は、半導体製品の動作例、応用例を説明するものです。回路、ソフトウェアおよびこれらに関連する情報を使用する場合、お客様の責任において、お客様の機器・システムを設計ください。これらの使用に起因して生じた損害（お客様または第三者いずれに生じた損害も含みます。以下同じです。）に関し、当社は、一切その責任を負いません。
2. 当社製品または本資料に記載された製品データ、図、表、プログラム、アルゴリズム、応用回路例等の情報の使用に起因して発生した第三者の特許権、著作権その他の知的財産権に対する侵害またはこれらに関する紛争について、当社は、何らの保証を行うものではなく、また責任を負うものではありません。
3. 当社は、本資料に基づき当社または第三者の特許権、著作権その他の知的財産権を何ら許諾するものではありません。
4. 当社製品を組み込んだ製品の輸出入、製造、販売、利用、配布その他の行為を行うにあたり、第三者保有の技術の利用に関するライセンスが必要となる場合、当該ライセンス取得の判断および取得はお客様の責任において行ってください。
5. 当社製品を、全部または一部を問わず、改造、改変、複製、リバースエンジニアリング、その他、不適切に使用しないでください。かかる改造、改変、複製、リバースエンジニアリング等により生じた損害に関し、当社は、一切その責任を負いません。
6. 当社は、当社製品の品質水準を「標準水準」および「高品質水準」に分類しており、各品質水準は、以下に示す用途に製品が使用されることを意図しております。

標準水準： コンピュータ、OA 機器、通信機器、計測機器、AV 機器、家電、工作機械、パーソナル機器、産業用ロボット等

高品質水準： 輸送機器（自動車、電車、船舶等）、交通制御（信号）、大規模通信機器、金融端末基幹システム、各種安全制御装置等

当社製品は、データシート等により高信頼性、Harsh environment 向け製品と定義しているものを除き、直接生命・身体に危害を及ぼす可能性のある機器・システム（生命維持装置、人体に埋め込み使用するもの等）、もしくは多大な物的損害を発生させるおそれのある機器・システム（宇宙機器と、海底中継器、原子力制御システム、航空機制御システム、プラント基幹システム、軍事機器等）に使用されることを意図しておらず、これらの用途に使用することは想定していません。たとえ、当社が想定していない用途に当社製品を使用したことにより損害が生じても、当社は一切その責任を負いません。

7. あらゆる半導体製品は、外部攻撃からの安全性を 100%保証されているわけではありません。当社ハードウェア／ソフトウェア製品にはセキュリティ対策が組み込まれているものもありますが、これによって、当社は、セキュリティ脆弱性または侵害（当社製品または当社製品が使用されているシステムに対する不正アクセス・不正使用を含みますが、これに限りません。）から生じる責任を負うものではありません。当社は、当社製品または当社製品が使用されたあらゆるシステムが、不正な改変、攻撃、ウイルス、干渉、ハッキング、データの破壊または窃盗その他の不正な侵入行為（「脆弱性問題」といいます。）によって影響を受けないことを保証しません。当社は、脆弱性問題に起因したまたはこれに関連して生じた損害について、一切責任を負いません。また、法令において認められる限りにおいて、本資料および当社ハードウェア／ソフトウェア製品について、商品性および特定目的との合致に関する保証ならびに第三者の権利を侵害しないことの保証を含め、明示または黙示のいかなる保証も行いません。
8. 当社製品をご使用の際は、最新の製品情報（データシート、ユーザーズマニュアル、アプリケーションノート、信頼性ハンドブックに記載の「半導体デバイスの使用上の一般的な注意事項」等）をご確認の上、当社が指定する最大定格、動作電源電圧範囲、放熱特性、実装条件その他指定条件の範囲内でご使用ください。指定条件の範囲を超えて当社製品をご使用された場合の故障、誤動作の不具合および事故につきましては、当社は、一切その責任を負いません。
9. 当社は、当社製品の品質および信頼性の向上に努めていますが、半導体製品はある確率で故障が発生したり、使用条件によっては誤動作したりする場合があります。また、当社製品は、データシート等において高信頼性、Harsh environment 向け製品と定義しているものを除き、耐放射線設計を行っておりません。仮に当社製品の故障または誤動作が生じた場合であっても、人身事故、火災事故その他社会的損害等を生じさせないよう、お客様の責任において、冗長設計、延焼対策設計、誤動作防止設計等の安全設計およびエージング処理等、お客様の機器・システムとしての出荷保証を行ってください。特に、マイコンソフトウェアは、単独での検証は困難なため、お客様の機器・システムとしての安全検証をお客様の責任で行ってください。
10. 当社製品の環境適合性等の詳細につきましては、製品個別に必ず当社営業窓口までお問合せください。ご使用に際しては、特定の物質の含有・使用を規制する RoHS 指令等、適用される環境関連法令を十分調査のうえ、かかる法令に適合するようご使用ください。かかる法令を遵守しないことにより生じた損害に関して、当社は、一切その責任を負いません。
11. 当社製品および技術を国内外の法令および規則により製造・使用・販売を禁止されている機器・システムに使用することはできません。当社製品および技術を輸出、販売または移転等する場合は、「外国為替及び外国貿易法」その他日本国および適用される外国の輸出管理関連法規を遵守し、それらの定めるところに従い必要な手続きを行ってください。
12. お客様が当社製品を第三者に転売等される場合には、事前に当該第三者に対して、本ご注意書き記載の諸条件を通知する責任を負うものいたします。
13. 本資料の全部または一部を当社の文書による事前の承諾を得ることなく転載または複製することを禁じます。
14. 本資料に記載されている内容または当社製品についてご不明な点がございましたら、当社の営業担当者までお問合せください。

注 1. 本資料において使用されている「当社」とは、ルネサス エレクトロニクス株式会社およびルネサス エレクトロニクス株式会社が直接的、間接的に支配する会社をいいます。

注 2. 本資料において使用されている「当社製品」とは、注 1 において定義された当社の開発、製造製品をいいます。

(Rev.5.0-1 2020.10)

## 本社所在地

〒135-0061 東京都江東区豊洲 3-2-24（豊洲フォレシア）

[www.renesas.com](http://www.renesas.com)

## お問合せ窓口

弊社の製品や技術、ドキュメントの最新情報、最寄の営業お問合せ窓口に関する情報などは、弊社ウェブサイトをご覧ください。

[www.renesas.com/contact/](http://www.renesas.com/contact/)

## 商標について

ルネサスおよびルネサスロゴはルネサス エレクトロニクス株式会社の商標です。すべての商標および登録商標は、それぞれの所有者に帰属します。