

RX23W グループ

Bluetooth メッシュモジュール Firmware Integration Technology

要旨

本書は Firmware Integration Technology(FIT)を使用した Bluetooth® メッシュモジュールについて説明します。本モジュールは Bluetooth Mesh Networking 仕様に準拠した多対多の無線通信機能を提供します。

以後、本書では本モジュールをメッシュ FIT モジュールと称します。

対象デバイス

RX23W グループ

関連ドキュメント

- Bluetooth Core 仕様 ([Core Specifications](#))
- Bluetooth Mesh Networking 仕様 ([Mesh Networking Specifications](#))
- CC-RX コンパイラ ユーザーズマニュアル ([R20UT3248](#))
- e² studio ユーザーズマニュアル 入門ガイド ([R20UT4374](#))
- RX スマート・コンフィグレータ ユーザーガイド e² studio 編 ([R20AN0451](#))
- Firmware Integration Technology ユーザーズマニュアル ([R01AN1833](#))
- RX ファミリ e² studio に組み込む方法 Firmware Integration Technology ([R01AN1723](#))
- RX ファミリ CS+に組み込む方法 Firmware Integration Technology ([R01AN1826](#))
- RX ファミリ ボードサポートパッケージモジュール Firmware Integration Technology ([R01AN1685](#))
- RX ファミリ フラッシュモジュール Firmware Integration Technology ([R01AN2184](#))
- RX23W グループ BLE モジュール Firmware Integration Technology ([R01AN4860](#))
- RX23W グループ Bluetooth メッシュスタック スタートアップガイド ([R01AN4874](#))
- RX23W グループ Bluetooth メッシュスタック 開発ガイド ([R01AN4875](#))

目次

1. 概要	3
1.1 提供機能	3
1.2 ソフトウェア構成	4
1.3 ファイル構成	5
1.4 API 仕様	5
1.5 API ヘッダファイル	5
2. 要件	6
2.1 ハードウェア要件	6
2.2 ソフトウェア要件	6
2.3 サポートするツールチェーン	6
2.4 セクション	7
2.5 プログラムサイズ	8
3. FIT モジュール設定	9
3.1 メッシュ FIT モジュール	9
3.2 BSP FIT モジュール	10
3.3 BLE FIT モジュール	10
4. FIT モジュールの追加方法	11
5. 使用方法	12
5.1 新規プロジェクトの作成	12
5.2 クロックの設定	15
5.3 コンポーネントの追加	16
5.4 コンポーネントの設定	18
5.4.1 r_bsp	18
5.4.2 r_ble_rx23w	19
5.4.3 r_sci_rx	20
5.4.4 r_irq_rx	22
5.5 コードの生成	24
5.6 コードの変更	25
5.6.1 r_ble_rx23w	25
5.7 リンカ設定	27
5.7.1 セクション	27
5.7.2 ライブラリ	29
5.8 デバッグ設定	31
5.8.1 デバッガの接続	31
5.9 プロジェクトのビルド	31
6. メッシュアプリケーションの実装方法	32

1. 概要

1.1 提供機能

メッシュ FIT モジュールは Bluetooth Mesh Networking 仕様に準拠した多対多の無線通信機能をアプリケーションに提供します。本モジュールがサポートする機能を以下に示します。

Bluetooth Core Mesh Profile 機能:

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

Bluetooth Mesh Model 機能:

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

1.2 ソフトウェア構成

図 1 にメッシュ FIT モジュールを使用するためのソフトウェア構成を示します。

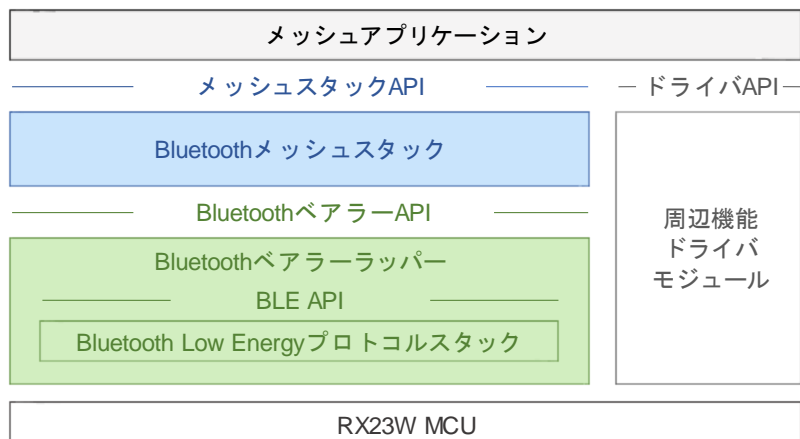


図 1 ソフトウェア構成

メッシュ FIT モジュールを使用するためのソフトウェア構成は下記のソフトウェアで構成されます。

- **メッシュアプリケーション**
メッシュアプリケーションは Bluetooth メッシュスタックが提供する機能を実行するプログラムです。
- **Bluetooth メッシュスタック**
Bluetooth メッシュスタックは Bluetooth Mesh Networking 仕様に準拠した多対多の無線通信機能をアプリケーションに提供するソフトウェアです。
- **Bluetooth ペアラーラッパー**
Bluetooth ペアラーラッパーは Bluetooth プロトコルスタックのラッパー関数をメッシュスタックとアプリケーションに提供する抽象化レイヤーです。
- **Bluetooth プロトコルスタック**
Bluetooth プロトコルスタックは Bluetooth Low Energy 仕様に準拠した無線通信機能を上位レイヤーに提供するソフトウェアです。

メッシュアプリケーションのサンプルプログラムは、メッシュ FIT モジュールパッケージ([R01AN4930](#))のデモプロジェクトに含まれます。

Bluetooth メッシュスタックと Bluetooth ペアラーラッパーは、メッシュ FIT モジュール([R01AN4930](#))に含まれます。

Bluetooth プロトコルスタックは、BLE FIT モジュール([R01AN4860](#))に含まれます。

1.3 ファイル構成

メッシュ FIT モジュール(r_mesh_rx23w)に含まれるファイル構成を以下に示します。

r_mesh_rx23w	
r_mesh_rx23w_if.h	メッシュスタック API ヘッダファイル
readme.txt	メッシュ FIT モジュール情報ファイル
+---doc\ blemesh_api.chm	メッシュスタック API 仕様書
+---en\ r01an4930ej0110-rx23w-blemesh.pdf	メッシュ FIT モジュール説明書(英)
+---ja\ r01an4930jj0110-rx23w-blemesh.pdf	メッシュ FIT モジュール説明書(日)
+---lib\ +---ref\ +---src\ +---bearer\ +---drivers\ +---include\ 	メッシュスタックライブラリ メッシュスタックデフォルト設定 Bluetooth ベアラーラッパー メッシュドライバ メッシュスタックヘッダファイル

メッシュ FIT モジュールが提供する機能を利用するには、メッシュ FIT モジュールをプロジェクトに組み込む必要があります。本モジュールの組み込み方法は、本書の 4 章を参照してください。

1.4 API 仕様

メッシュ FIT モジュールの機能を実行するには、メッシュ FIT モジュールに含まれるメッシュスタックの API を利用します。メッシュスタック API の仕様は、メッシュ API 仕様書(doc\blemesh_api.chm)を参照してください。

1.5 API ヘッダファイル

メッシュ FIT モジュールを利用するには、r_mesh_rx23w_if.h をインクルードしてください。メッシュスタックの API は src\include\に含まれる複数のヘッダファイルで定義されていますが、r_mesh_rx23w_if.h によって全てのヘッダファイルをインクルードすることができます。

2. 要件

メッシュ FIT モジュールを使用したアプリケーション開発の要件について示します。

2.1 ハードウェア要件

ご使用になる MCU は下記のハードウェア機能をサポートしている必要があります。

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

2.2 ソフトウェア要件

メッシュ FIT モジュールは下記の FIT モジュールを必要とします。

- **r_bsp**: Board Support Package (BSP FIT モジュール)
- **r_ble_rx23w**: Bluetooth Low Energy (BLE FIT モジュール)
- **r_flash_rx**: Data Flash memory (フラッシュ FIT モジュール)

BLE FIT モジュールは下記の FIT モジュールを必要とします。

- **r_lpc_rx**: Low Power Control (LPC FIT モジュール)
- **r_cmt_rx**: Compare Match Timer (CMT FIT モジュール バージョン 4.50 以降) 注
- **r_sci_rx**: Serial Communication Interface (SCI FIT モジュール)
- **r_byteq**: Byte Queues/Circular Buffers (BYTEQ FIT モジュール)
- **r_gpio_rx**: General Purpose I/O (GPIO FIT モジュール)
- **r_irq_rx**: Interrupt Request (IRQ FIT モジュール)

注: BLE FIT モジュールは CMT2 と CMT3 を直接使用するため、CMT FIT モジュールは CMT0 と CMT1 のみ使用できます。

2.3 サポートするツールチェーン

メッシュ FIT モジュールは下記のツールチェーンで動作を確認済みです。

- **統合開発環境:** ルネサスエレクトロニクス製 e² studio V7.8.0
- **コンパイラ:** ルネサスエレクトロニクス製 C/C++ Compiler for RX Family (CC-RX) V2.08.00
- **エンディアン:** リトルエンディアン
- **開発ボード:** Target Board for RX23W (RTK5RX23W0C00000BJ)
Renesas Solution Starter Kit (RSSK) for RX23W (RTK5523W8AC00001BJ)

2.4 セクション

メッシュ FIT モジュールに含まれるメッシュスタックは表 1 に示すセクション名で配置されます。

表 1 メッシュスタックのセクション名

プログラム領域の名称	メッシュスタックのセクション		
	名称	属性	アライメント数
プログラム領域	MESH_P	code	1byte
定数領域	MESH_C	romdata	4byte
	MESH_C_2	romdata	2byte
	MESH_C_1	romdata	1byte
初期化データ領域	MESH_D	romdata	4byte
	MESH_D_2	romdata	2byte
	MESH_D_1	romdata	1byte
	MESH_R	data	4byte
	MESH_R_2	data	2byte
	MESH_R_1	data	1byte
未初期化データ領域	MESH_B	data	4byte
	MESH_B_2	data	2byte
	MESH_B_1	data	1byte
switch 文分岐テーブル領域	MESH_W	romdata	4byte
	MESH_W_2	romdata	2byte
	MESH_W_1	romdata	1byte
リテラル領域	MESH_L	romdata	4byte

属性: code 実行命令を格納
 data 変更可能なデータを格納
 romdata 固定データを格納

セクション仕様の詳細は「CC-RX コンパイラ ユーザーズマニュアル」([R20UT3248](#))の 6 章「セクション仕様」を参照してください。

メッシュ FIT モジュールを使用するプログラムは、ROM の初期化データを RAM の初期化データセクションに転送する必要があります。初期化データの転送のための設定は 5.7.1 項を参照してください。

2.5 プログラムサイズ

表 2 にメッシュ FIT モジュールのプログラムサイズを示します。なお使用されない変数や関数がある場合、リンカの最適化処理によって削除されるため、実際にリンクされるメッシュ FIT モジュールの ROM サイズは減少します。またメッシュ FIT モジュールの設定変更により、メッシュ FIT モジュールが必要とする RAM サイズは変動します。

表 2 メッシュ FIT モジュールの全プログラムサイズ

デバイス	コンパイラ	分類	サイズ
RX23W グループ	CC-RX V2.08.00	ROM	62,667byte
		RAM	9,165byte
条件			
メッシュ FIT モジュール			
デフォルト設定 (r_mesh_rx23w\ref\r_mesh_rx23w_config_reference.h)			
コンパイルオプション			
最適化レベル	レベル 2 全体的に最適化を実施する (-optimize=2)		
最適化方法	コード・サイズ重視の最適化を実施する (-size)		
リンクオプション			
最適化方法	すべての最適化を行わない (-nooptimize)		

表 3 にメッシュ FIT モジュールパッケージに含まれるデモプロジェクトのプログラムサイズを示します。デモプロジェクトの詳細は「RX23W グループ Bluetooth メッシュスタック 開発ガイド」([R01AN4875](#))を参照してください。

表 3 メッシュ FIT モジュールパッケージに含まれるデモプロジェクトのプログラムサイズ

デバイス	コンパイラ	分類	サイズ
RX23W グループ	CC-RX V2.08.00	ROM	297,919byte (メッシュ FIT モジュール 58,672byte)
		RAM	43,757byte (メッシュ FIT モジュール 9,165byte)
条件			
プロジェクト			
Target Board for RX23W 向け Server Models プロジェクト (rsskrx23w_mesh_server)			
コンパイルオプション			
最適化レベル レベル 2 全体的に最適化を実施する (-optimize=2)			
最適化方法 コード・サイズ重視の最適化を実施する (-size)			
リンクオプション			
最適化方法 一度も参照のない変数/関数を削除する (-optimize=symbol_delete)			

3. FIT モジュール設定

3.1 メッシュ FIT モジュール

メッシュ FIT モジュールには、メッシュネットワーク規模やノードの要件に応じて設定可能なパラメータがあります。これらのパラメータは表 4 に示す設定マクロとして `r_ble_rx23w_config.h` で定義されます。

スマート・コンフィグレータを使用する場合は、GUI でこれらの設定マクロの値を設定でき、プロジェクトへのメッシュ FIT モジュールの追加時に `r_ble_rx23w_config.h` に反映されます。

表 4 メッシュ FIT モジュールの設定マクロ

設定マクロ	設定範囲	デフォルト値
MESH_CFG_DEFAULT_COMPANY_ID Bluetooth SIG に登録されたカンパニー ID	0x0000～0xFFFFE	0x0036
MESH_CFG_DEFAULT_PID ベンダー独自の製品 ID	0x0000～0xFFFF	0x0001
MESH_CFG_DEFAULT_VID ベンダー独自の製品バージョン ID	0x0000～0xFFFF	0x0100
MESH_CFG_ACCESS_ELEMENT_COUNT エレメントの最大数	1～65,535	4
MESH_CFG_ACCESS_MODEL_COUNT モデルの最大数	1～65,535	20
MESH_CFG_HEALTH_SERVER_MAX Health Servers の最大数	1～65,535	2
MESH_CFG_MAX_SUBNETS サブネットの最大数	1～65,535	4
MESH_CFG_MAX_APPS デバイスが保持するアプリケーションキーの最大数	1～65,535	8
MESH_CFG_MAX_DEV_KEYS デバイスキーの最大数	1～65,535	4
MESH_CFG_MAX_VIRTUAL_ADDRS デバイスが保持するバーチャルアドレスの最大数	1～65,535	8
MESH_CFG_MAX_NON_VIRTUAL_ADDRS デバイスが保持する非バーチャルアドレス(ユニキャストアドレス、グループアドレス)の最大数	1～65,535	8
MESH_CFG_NET_SEQ_NUMBER_BLOCK_SIZE ストレージへの書き込みに使用するシーケンス番号のブロック幅	1～65,535	2048
MESH_CFG_MAX_LPNS フレンドノードとしてフレンドシップを確立する対向ローパワーノード (LPN) の最大数	1～65,535	1
MESH_CFG_FRIEND_SUBSCRIPTION_LIST_SIZE 各ローパワーノードに対するサブスクリプションリストの最大数	1～65,535	8
MESH_CFG_FRIEND_MESSAGEQUEUE_SIZE 各ローパワーノードに対するメッセージキュー数	2～65,535	15
MESH_CFG_PROXY_FILTER_LIST_SIZE 各プロキシリストに登録可能なアドレスの最大数	1～65,535	2
MESH_CFG_NET_CACHE_SIZE ネットワークメッセージキャッシュのノード数	2～65,535	10

MESH_CFG_NET_SEQNUM_CACHE_SIZE ネットワークメッセージキャッシュのノードあたりのシーケンス番号キャッシュ数	32~65,535	32
MESH_CFG_REPLAY_CACHE_SIZE リプレイプロテクション(Replay Protection)キャッシュ数	2~65,535	10
MESH_CFG_LTRN_SAR_CTX_MAX SAR(Segmentation and reassembly)コンテキスト数	2~65,535	8
MESH_CFG_REASSEMBLED_CACHE_SIZE SAR(Segmentation and reassembly)受信キャッシュ数	2~65,535	8
MESH_CFG_NUM_NETWORK_INTERFACES メッシュネットワークに使用するベアラース数	1~65,535	2
MESH_CFG_NUM_PROVISIONING_INTERFACES プロビジョニングに使用するベアラース数	1~65,535	2

3.2 BSP FIT モジュール

BSP FIT モジュールの表 5 に示す設定マクロは、メッシュ FIT モジュールが動作するために変更が必要です。スマート・コンフィグレータでの設定方法は 5.4.1 項を参照してください。

注: メッシュ FIT モジュールを使用する場合、本変更を必ず行ってください。

表 5 BSP FIT モジュールの変更が必要な設定マクロ

設定マクロ	デフォルト値	メッシュ向け設定値
BSP_CFG_HEAP_BYTES	0x400	0x1000
BSP_CFG_CLOCK_SOURCE	4	1
BSP_CFG_USB_CLOCK_SOURCE	1	0
BSP_CFG_PCKB_DIV	2	1
BSP_CFG_FCK_DIV	2	1
BSP_CFG_CONFIGURATOR_SELECT	0	1

3.3 BLE FIT モジュール

BLE FIT モジュールの表 6 に示す設定マクロは、使用するリソースを削減するために変更します。スマート・コンフィグレータでの設定方法は 5.4.2 項を参照してください。

表 6 BLE FIT モジュールの変更すべき設定マクロ

設定マクロ	デフォルト値	メッシュ向け設定値
BLE_CFG_RF_CONN_MAX	7	1
BLE_CFG_RF_ADV_DATA_MAX	1650	31
BLE_CFG_RF_ADV_SET_MAX	4	1
BLE_CFG_RF_SYNC_SET_MAX	2	1
BLE_CFG_CMD_LINE_CH	1	8
BLE_CFG_BOARD_TYPE	0	1: Target Board for RX23W 2: RSSK for RX23W

4. FIT モジュールの追加方法

各 FIT モジュールはプロジェクト毎に追加する必要があります。推奨される FIT モジュールの追加方法は、スマート・コンフィグレータを使用する下記の(1)または(2)です。

- (1) e² studio 上でスマート・コンフィグレータを使用して追加する場合
e² studio のスマート・コンフィグレータを使用して、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「RX スマート・コンフィグレータ ユーザーガイド e² studio 編」([R20AN0451](#))を参照してください。また 5 章も併せて参照してください。
- (2) CS+上でスマート・コンフィグレータを使用して追加する場合
CS+上でスタンドアロン版スマート・コンフィグレータを使用して、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「RX スマート・コンフィグレータ ユーザーガイド e² studio 編」([R20AN0451](#))を参照してください。
- (3) e² studio 上で FIT コンフィグレータを使用して追加する場合
e² studio の FIT コンフィグレータを使用して、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「RX ファミリ e² studio に組み込む方法 Firmware Integration Technology」([R01AN1723](#))を参照してください。
- (4) CS+上で手動追加する場合
CS+上で、手動でユーザプロジェクトに FIT モジュールを追加することができます。詳細は「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。

5. 使用方法

e² studio 上でスマート・コンフィグレータを使用して、新規プロジェクトにメッシュ FIT モジュールを追加する方法について説明します。

5.1 新規プロジェクトの作成

[ファイル]メニューから[新規]→[C/C++ Project]を選択します。[Templates for New C/C++ Project]ダイアログの左側で[Renesas RX]、右側で[Renesas CC-RX C/C++ Executable Project]を選択し、[次へ]ボタンをクリックします。

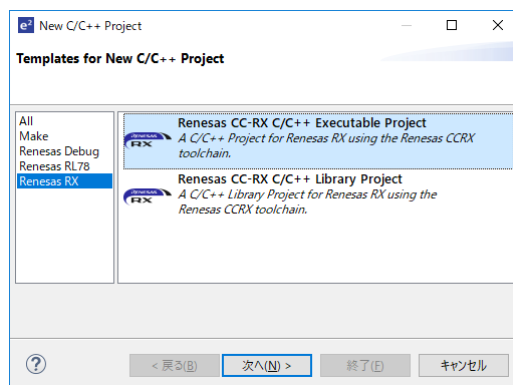


図 2 プロジェクトテンプレートの選択

[New Renesas CC-RX Executable Project]ダイアログでプロジェクト名を入力し、[次へ]ボタンをクリックします。

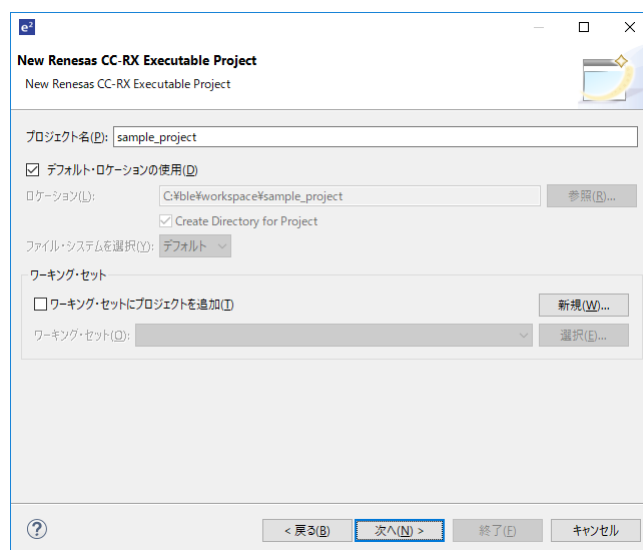


図 3 新規プロジェクトの設定

Device Setting でエンディアンを[Little]に設定します。またターゲット・デバイスで使用する RX23W のデバイス型名を選択し、[次へ]ボタンをクリックします。

Target Board for RX23W を使用する場合は"R5F523W8AxNG"を選択します。RSSK for RX23W を使用する場合、RSSK の型名が"RTK5523W8AC00001BJ"であれば"R5F523W8AxBL"、"RTK5523W8BC00001BJ"であれば"R5F523W8BxBL"を選択します。

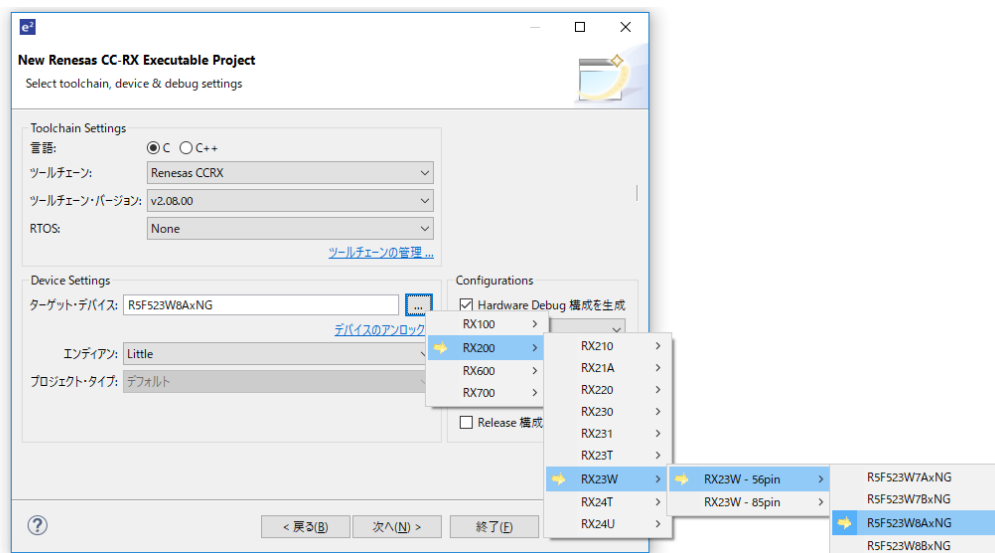


図 4 ツールチェーン、デバイス、デバッグの設定

[コーディング・アシストツールの選択]ダイアログで[スマート・コンフィグレータを使用する]にチェックを入れ、[終了]ボタンをクリックします。

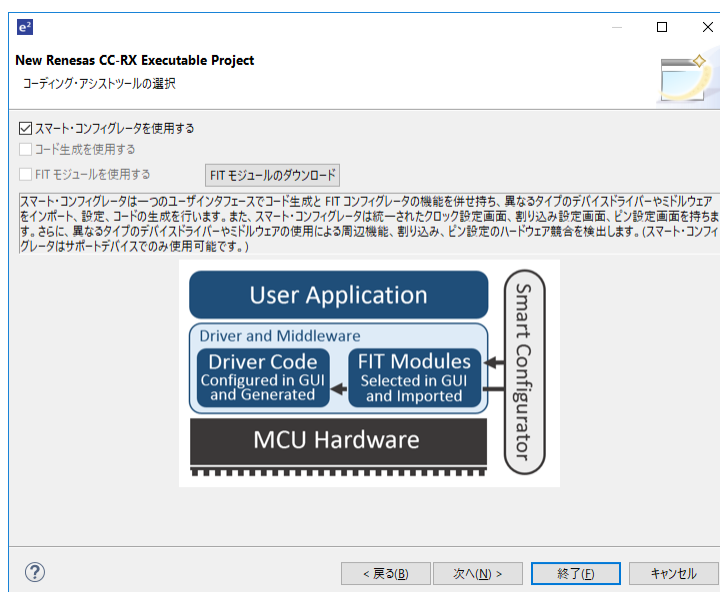


図 5 コーディング・アシストの選択

e² studio に新規のプロジェクトが作成されます。またスマート・コンフィグレータはプロジェクトに含まれる{プロジェクト名}.scfg をクリックすることで使用できます。

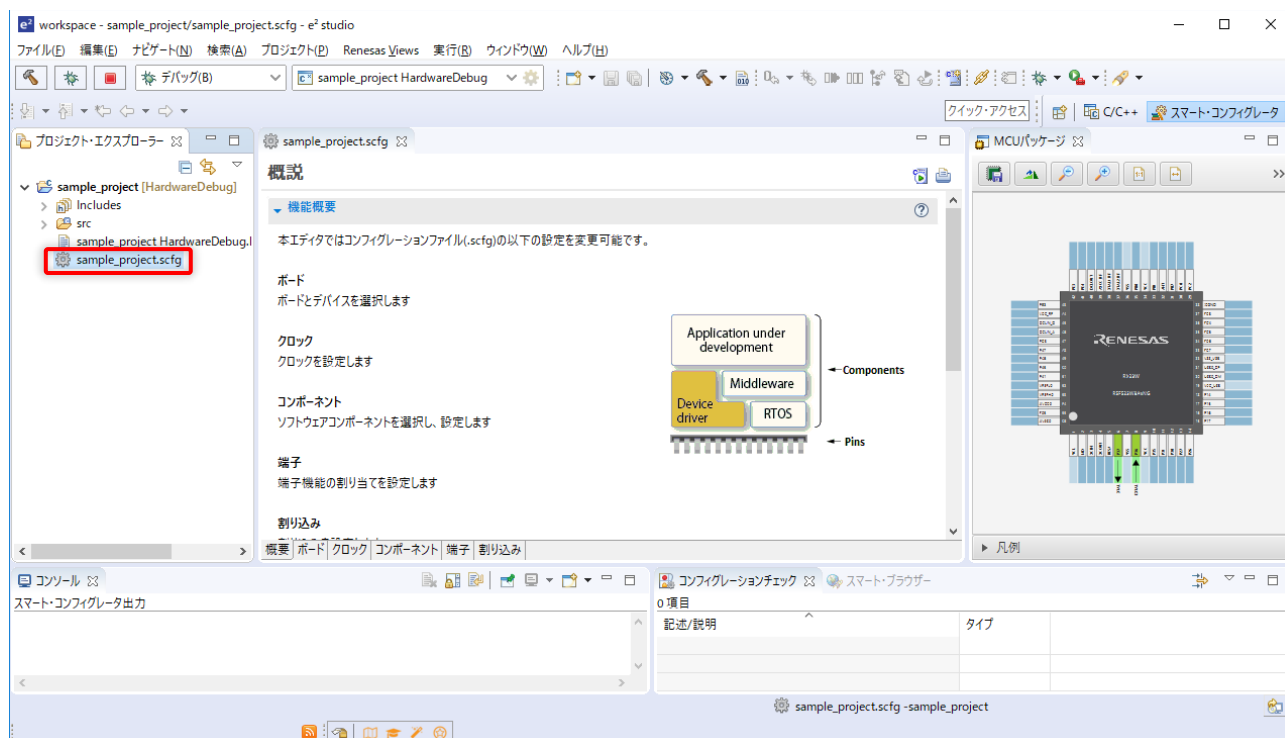


図 6 新規プロジェクトの作成完了

5.2 クロックの設定

スマート・コンフィグレータの[クロック]タブで、クロックの選択と周波数を設定します。メッシュ FIT モジュールを使用する場合、下記の通り設定してください。

- システムクロック (ICLK): 8MHz 以上
- 周辺モジュールクロック B(PCLKB): 8MHz 以上

BLE FIT モジュールに含まれる BLE プロトコルスタックは、ICLK と PCLKB の周波数が 32MHz の場合に最適化されています。このため ICLK と PCLKB の周波数が 32MHz となるように設定することを推奨します。

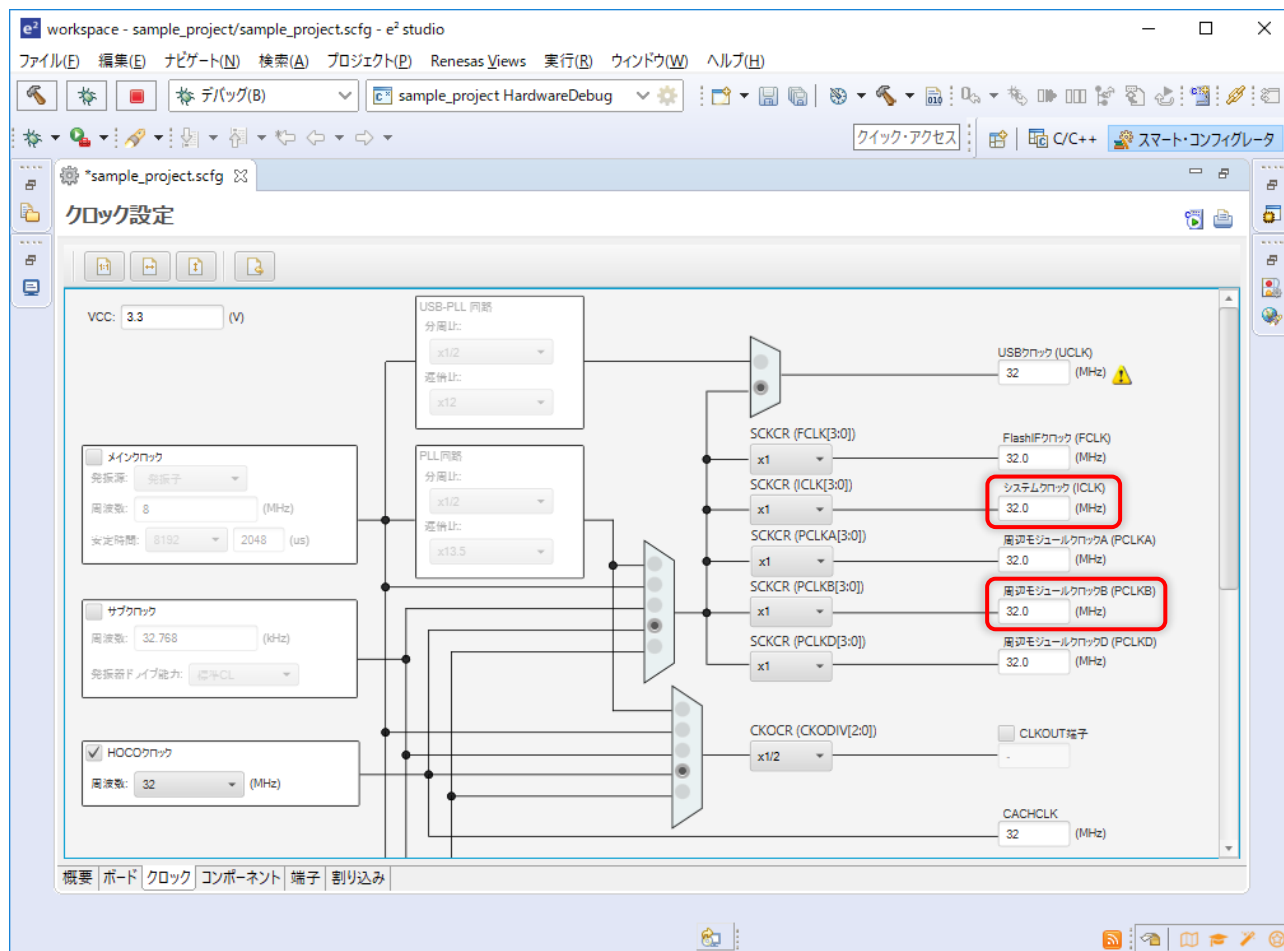



図 7 クロックの設定

5.3 コンポーネントの追加

スマート・コンフィグレータの[コンポーネント]タブで、メッシュ FIT モジュールとその他の必要な FIT モジュールを追加します。必要な FIT モジュールは 2.2 節を参照してください。

[コンポーネントの追加]ボタンをクリックします。[ソフトウェアコンポーネントの選択]ダイアログで、必要な FIT モジュール(r_mesh_rx23w, r_bsp, r_ble_rx23w, r_byteq, r_cmt_rx, r_flash_rx, r_gpio_rx, r_irq_rx, r_lpc_rx, r_sci_rx)を選択し、[終了]ボタンをクリックします。

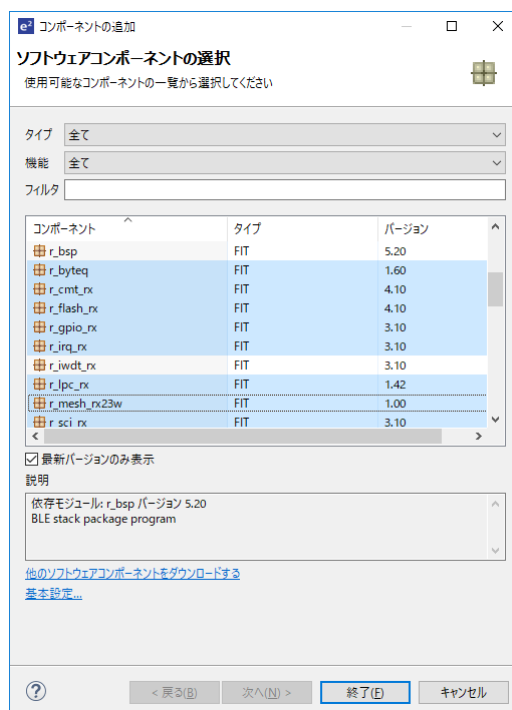


図 8 ソフトウェアコンポーネントの選択

注: 必要な FIT モジュールが見つからない場合は、[他のソフトウェアコンポーネントをダウンロードする]をクリックし、[FIT モジュールのダウンロード]ダイアログの手順に従って FIT モジュールをダウンロードしてください。

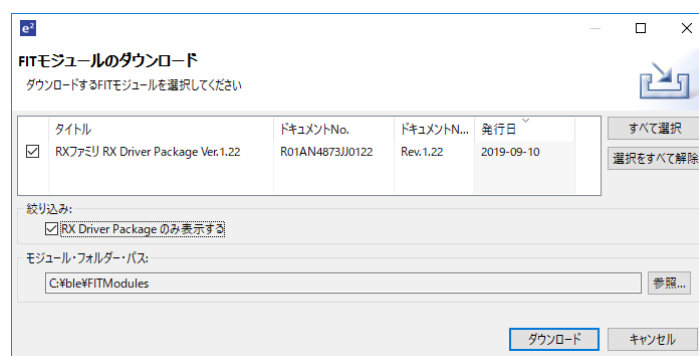


図 9 FIT モジュールのダウンロード

注: ダウンロードした FIT モジュールが見つからない場合は、[基本設定...]をクリックし、[設定]ダイアログの[すべての FIT モジュールを表示する]にチェックを入れてください。

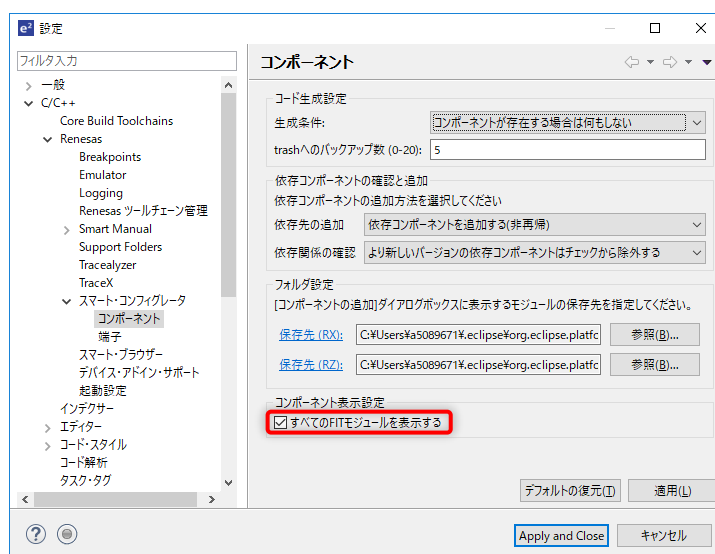


図 10 すべての FIT モジュールを表示

選択した FIT モジュールが[コンポーネント]タブに追加されます。

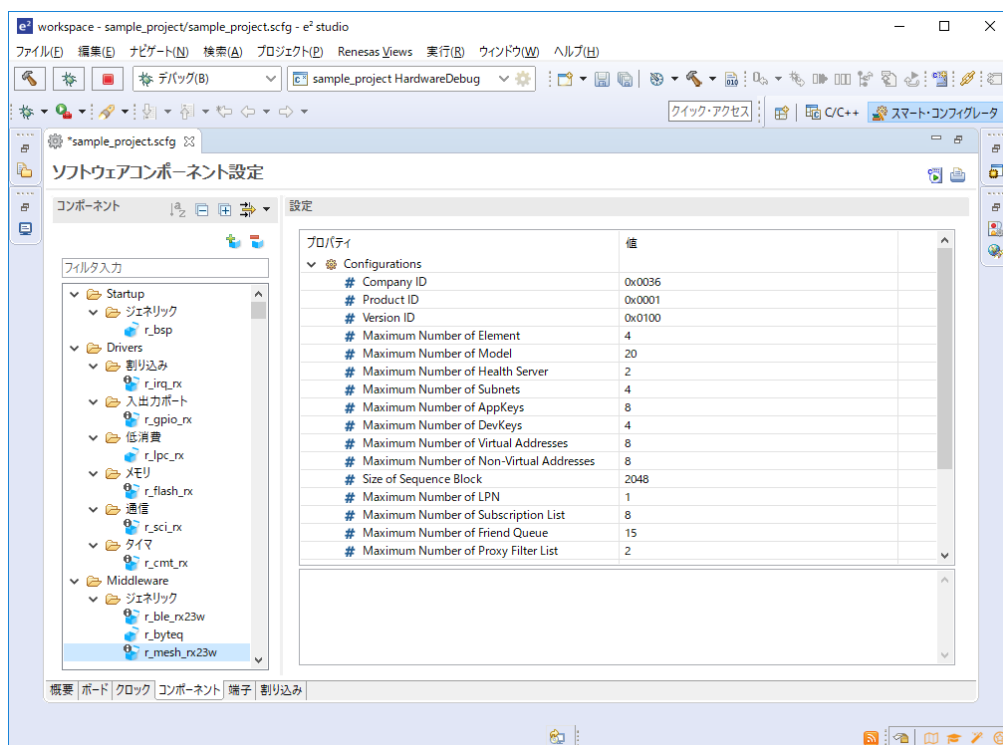


図 11 ソフトウェアコンポーネント

5.4 コンポーネントの設定

スマート・コンフィグレータの[コンポーネント]タブで、メッシュ FIT モジュールが必要とする FIT モジュールの設定を変更します。

5.4.1 r_bsp

メッシュ FIT モジュールの動作に必要なヒープ領域を確保するため、[コンポーネント]タブで[r_bsp]を選択して[Heap size]を"0x1000"に設定します。

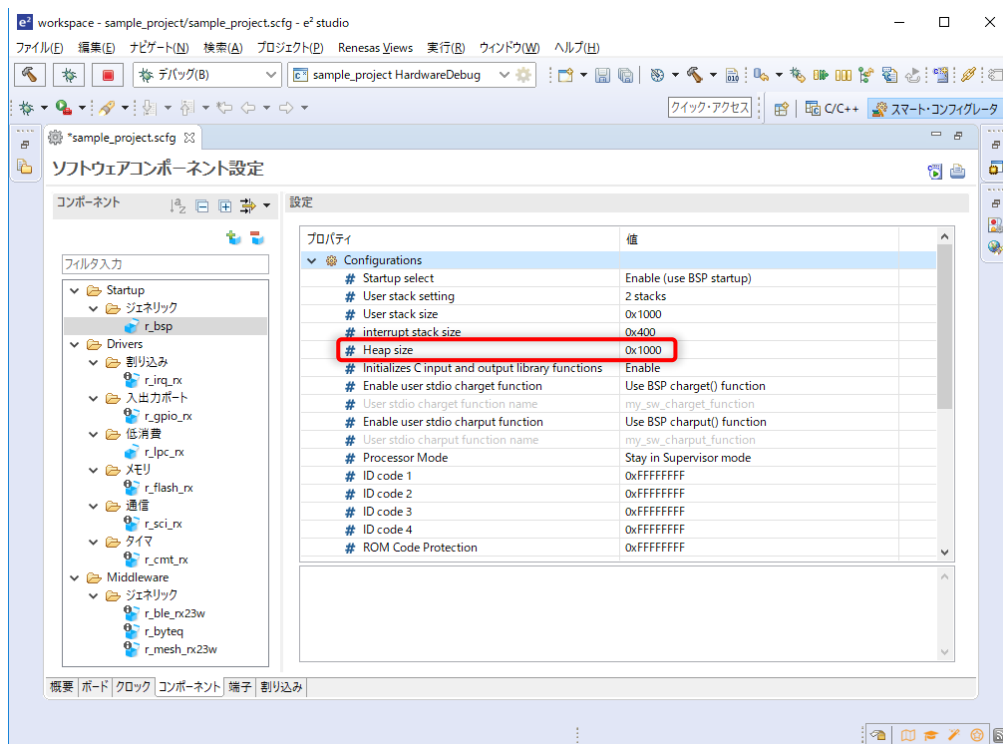


図 12 r_bsp 設定

5.4.2 r_ble_rx23w

BLE FIT モジュールが使用するリソースを削減するため、[コンポーネント]タブで[r_ble_rx23w]を選択して[Maximum number of connections]を"1"に、[Maximum advertising data length]を"31"に、[Maximum advertising set number]を"1"に、[Maximum periodic sync set number]を"1"に設定します。

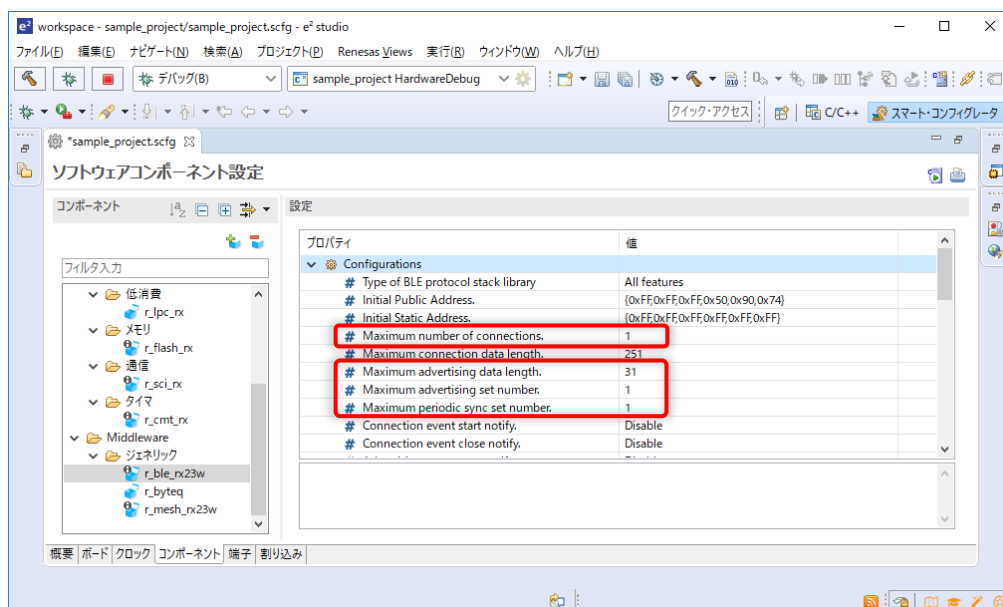


図 13 r_ble_rx23w 設定(1)

また Target Board for RX23W または RSSK for RX23W を使用の場合は、[SCI CH for command line function]を"8"に設定し、[Board Type]を"Target Board"または"RSSK"に設定します。

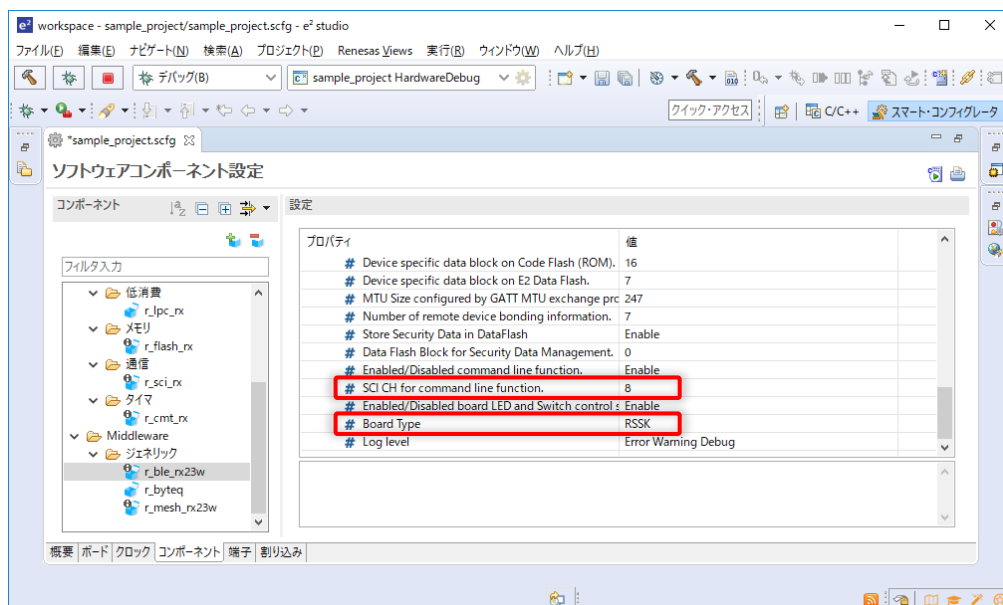


図 14 r_ble_rx23w 設定(2)

5.4.3 r_sci_rx

Target Board for RX23W または RSSK for RX23W でシリアル通信機能を使用する場合は、[コンポーネント]タブで[r_sci_rx]を選択して[Include software support for channel 8]を"Include"に設定します。またその他の SCI チャンネルを"Not"に設定します。

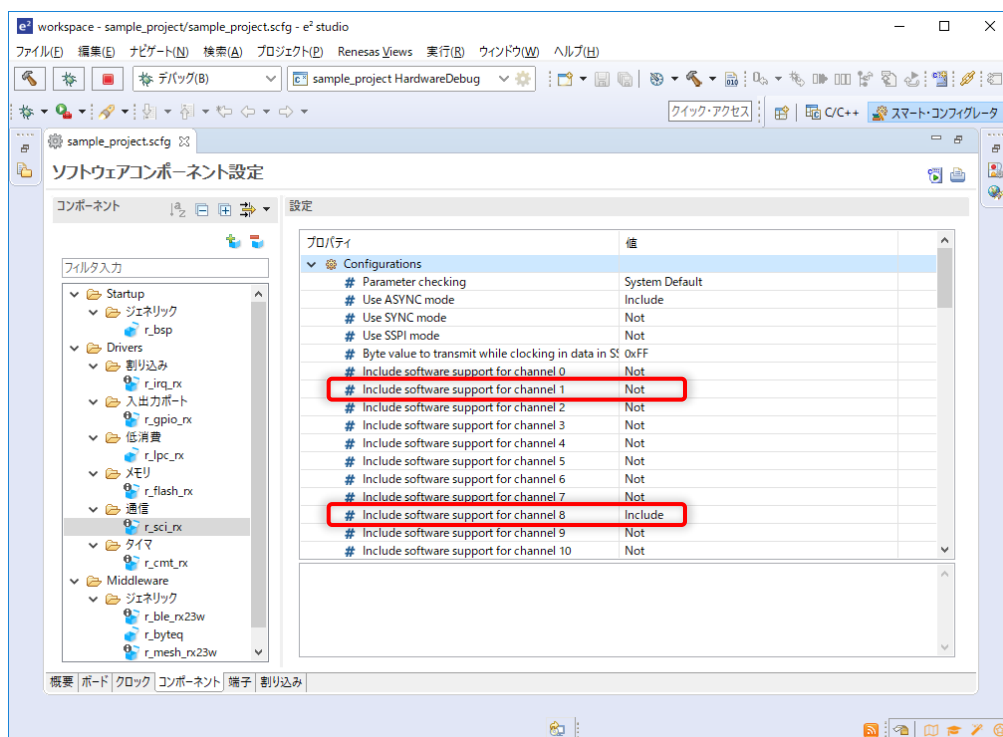


図 15 r_sci_rx 設定(1)

さらに[リソース]→[SCI8]を選択後、[RXD8/SMISO8/SSCL8 端子]と[TXD8/SMOSI8/SSDA8 端子]を"使用する"に設定します。

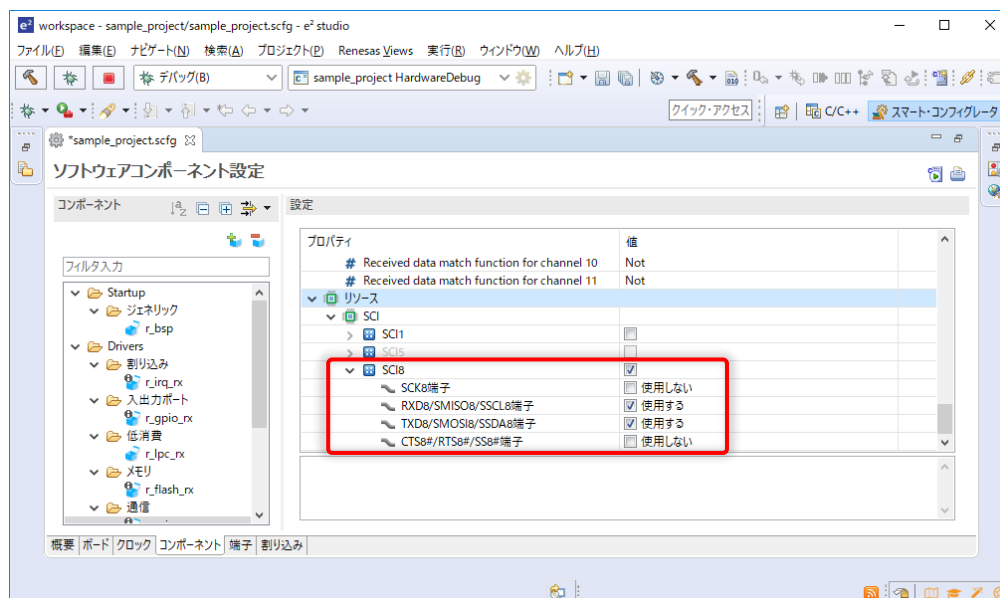


図 16 r_sci_rx 設定(2)

BLE FIT モジュールは RX23W の開発ボードでシリアル通信するためのコマンドラインインタフェース (CLI)を提供します。

本機能を使用する場合は、[Transmit end interrupt]を"Enable"に設定します。また本機能を Target Board for RX23W または RSSK for RX23W で使用する場合は、[ASYNC mode TX queue buffer size for channel 8]を"160"に設定します。

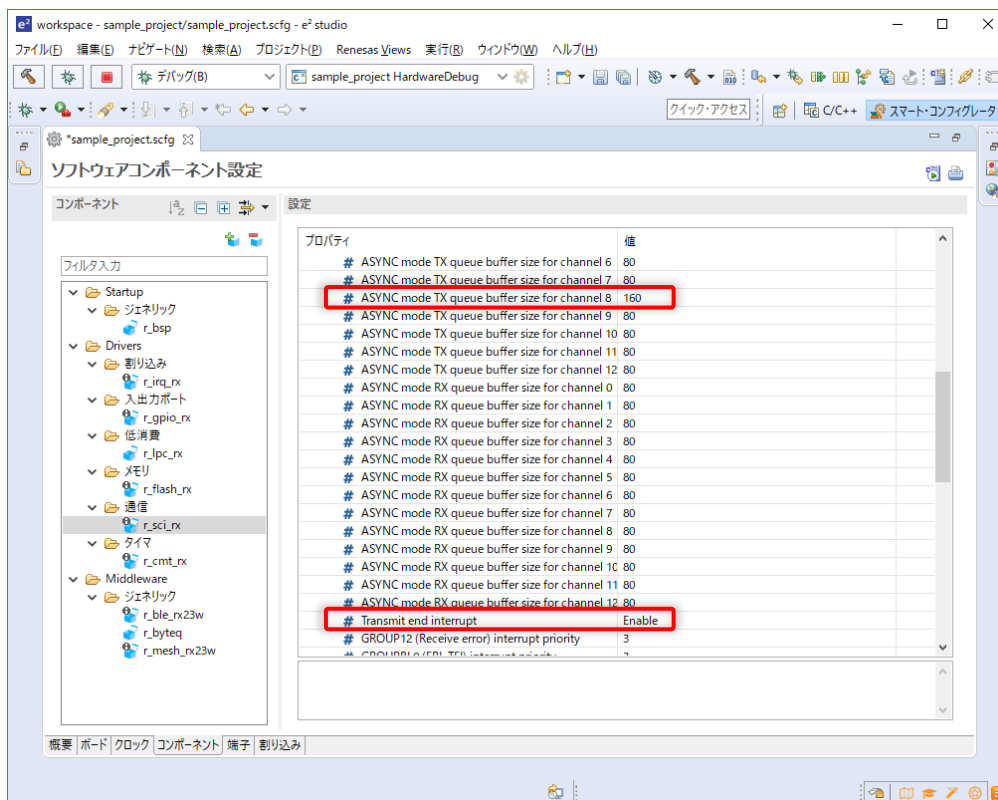


図 17 r_sci_rx 設定(3)

5.4.4 r_irq_rx

Target Board for RX23W に実装されたスイッチを使用する場合は、[コンポーネント]タブで[r_irq_rx]を選択し、[IRQ5 端子]を"使用する"に設定します。

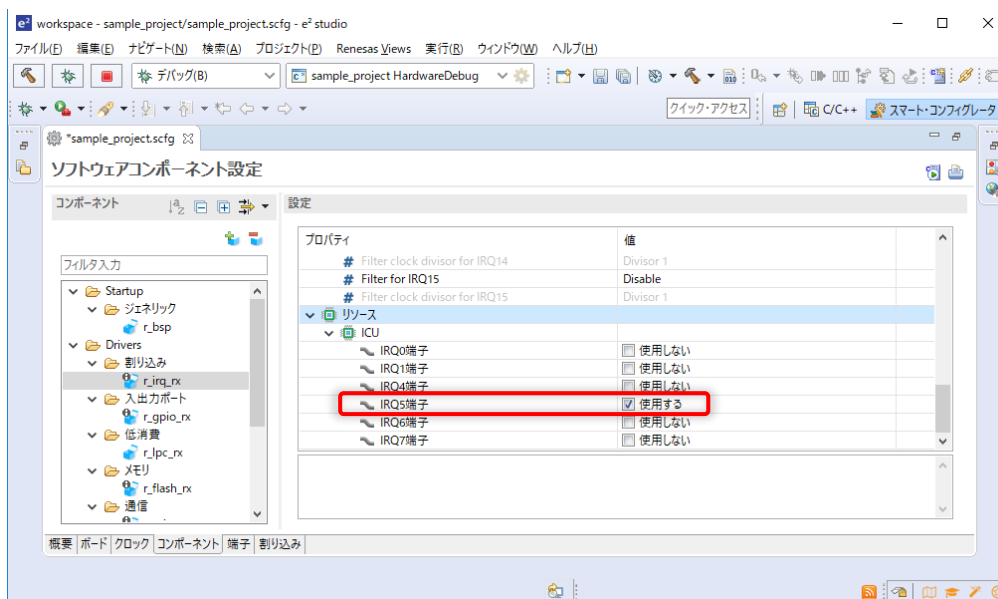


図 18 Target Board 向け r_irq_rx 設定(1)

さらに[Filter for IRQ5]を"Enable"に設定し、[Filter clock divisor for IRQ5]を"Divisor 1"に設定します。

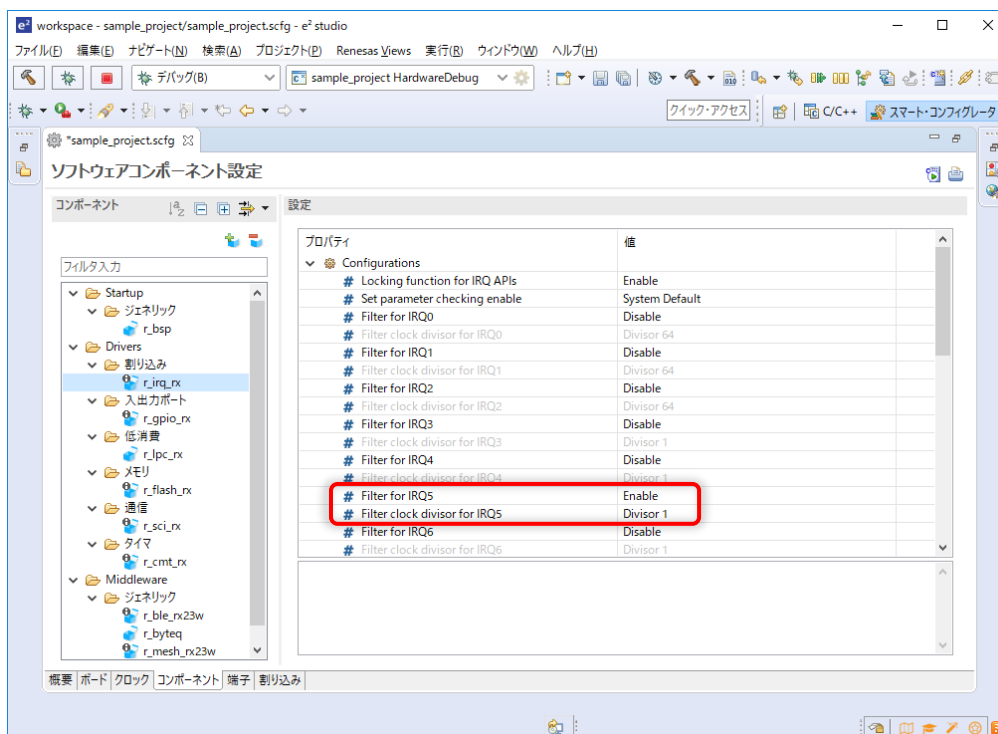


図 19 Target Board 向け r_irq_rx 設定(2)

RSSK for RX23W に実装されたスイッチを使用する場合は、[コンポーネント]タブで[r_irq_rx]を選択し、[IRQ0 端子]と[IRQ1 端子]を"使用する"に設定します。

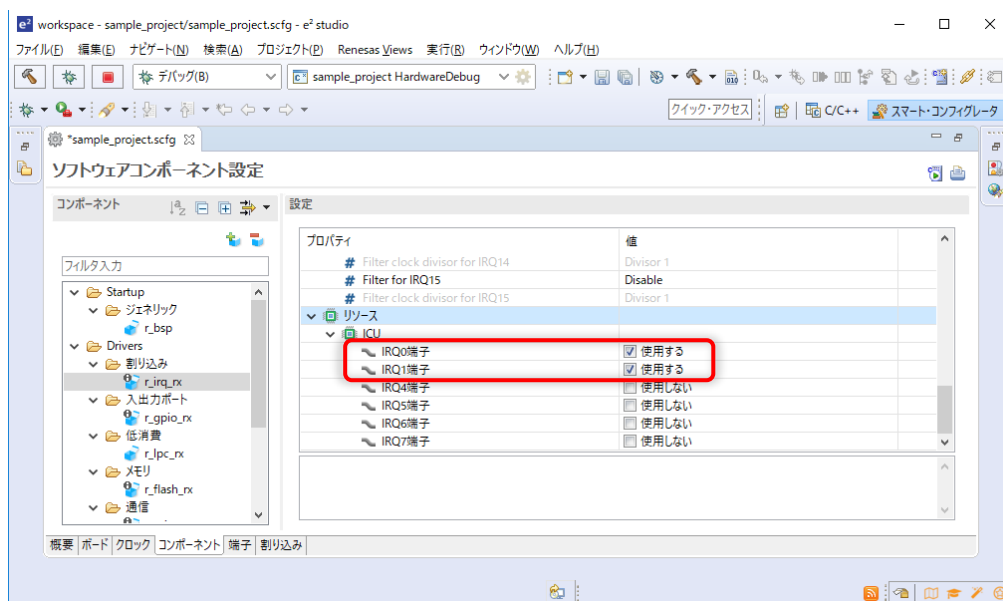


図 20 RSSK 向け r_irq_rx 設定(1)

さらに[Filter for IRQ0]と[Filter for IRQ1]を"Enable"に設定し、[Filter clock divisor for IRQ0]と[Filter clock divisor for IRQ1]を"Divisor 1"に設定します。

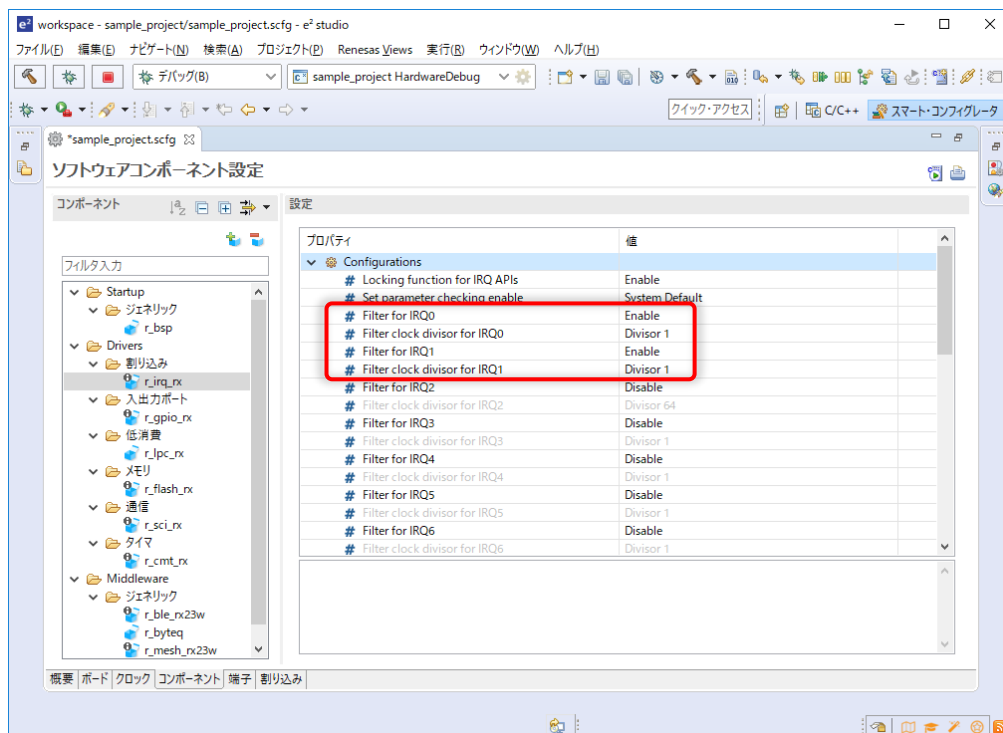



図 21 RSSK 向け r_irq_rx 設定(2)

5.5 コードの生成

スマート・コンフィグレータの[コンポーネント]タブで、[コードの生成]ボタン  をクリックします。プロジェクトの smc_gen フォルダに FIT モジュールのコードが生成されます。

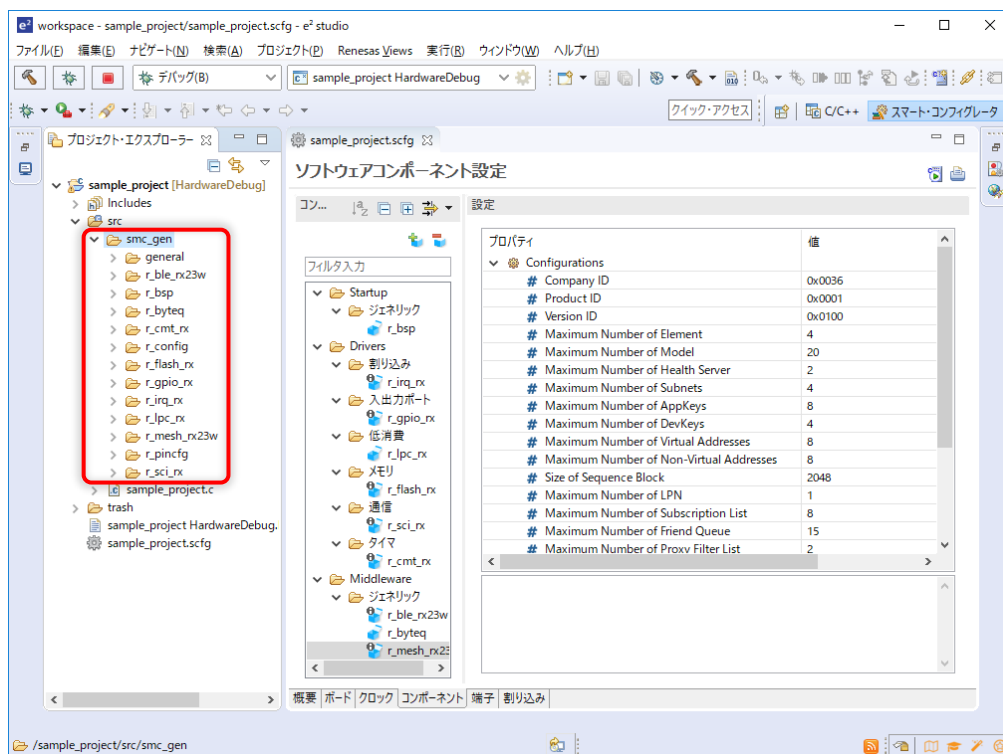


図 22 コードの生成結果

5.6 コードの変更

スマート・コンフィグレータのコード生成後、メッシュ FIT モジュールを使用するために必要なコードの変更を行います。

5.6.1 r_ble_rx23w

BLE FIT モジュールの意図しない動作を回避するため、下記の変更を適用します。

(1) ソフトウェアタイマ

BLE FIT モジュールに含まれるソフトウェアタイマの API を特定のタイミングで実行した場合、タイムアウトが通知されない問題を解消するため、r_ble_rx23w が生成したコードに下記の変更を適用します。

ファイル: src\smc_gen\r_ble_rx23w\src\app_lib\timer\r_ble_timer.c

注: 本変更は Rev2.00 以前の r_ble_rx23w に適用してください。Rev2.00 より後の r_ble_rx23w で生成したコードには本変更が含まれます。

下記のソースコードを参照し、(BSP_CFG_RTOS_USED == 0)の場合の update_remaining_time_ms() と process_timer_expire()の実装を更新してください。

1. update_remaining_time_ms(): 赤で示すコードを追加してください。

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

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

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

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

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

    R_BSP_InterruptsEnable();
}
```

2. process_timer_expire(): グレーで示すコードを削除してください。

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

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

    start_timer();
}
```

(2) RF スリープ機能

BLE FIT モジュールに含まれる BLE プロトコルスタックの Advertising 動作を連続実行した場合、Scan 動作が意図せず停止する問題を解消するため、r_ble_rx23w が生成したコードに下記の変更を適用します。

ファイル: src\smc_gen\r_ble_rx23w\src\platform\r_ble_pf_functions.c

注: 本変更は Rev2.00 以前の r_ble_rx23w に適用してください。Rev2.00 より後の r_ble_rx23w では本変更は不要となります。

r_ble_rf_power_save_mode(): 赤で示すコードを追加してください。

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

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

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

    return ret;
}
```

5.7 リンカ設定

5.7.1 セクション

メッシュ FIT モジュール、BLE FIT モジュールを使用するプロジェクトでは、リンクオプションに各モジュールのセクションを追加し、ROM の初期化データを RAM の初期化データセクションに転送するための設定を行う必要があります。

(1) セクションの追加

[プロジェクト]メニューで[プロパティ]を選択します。[プロパティ]ダイアログの左側で[C/C++ ビルド]→[設定]を選択後、右側の[ツール設定]タブで[Linker]→[セクション]を選択します。[...]ボタンをクリックすると表示されるセクション・ビューアで、下記のセクションを追加します。

[RAM]

BLE FIT モジュールセクション BLE_B*, BLE_R*

メッシュ FIT モジュールセクション MESH_B*, MESH_R*

[ROM]

BLE FIT モジュールセクション BLE_C*, BLE_D*, BLE_W*, BLE_L*, BLE_P*

メッシュ FIT モジュールセクション MESH_C*, MESH_D*, MESH_W*, MESH_L*, MESH_P*

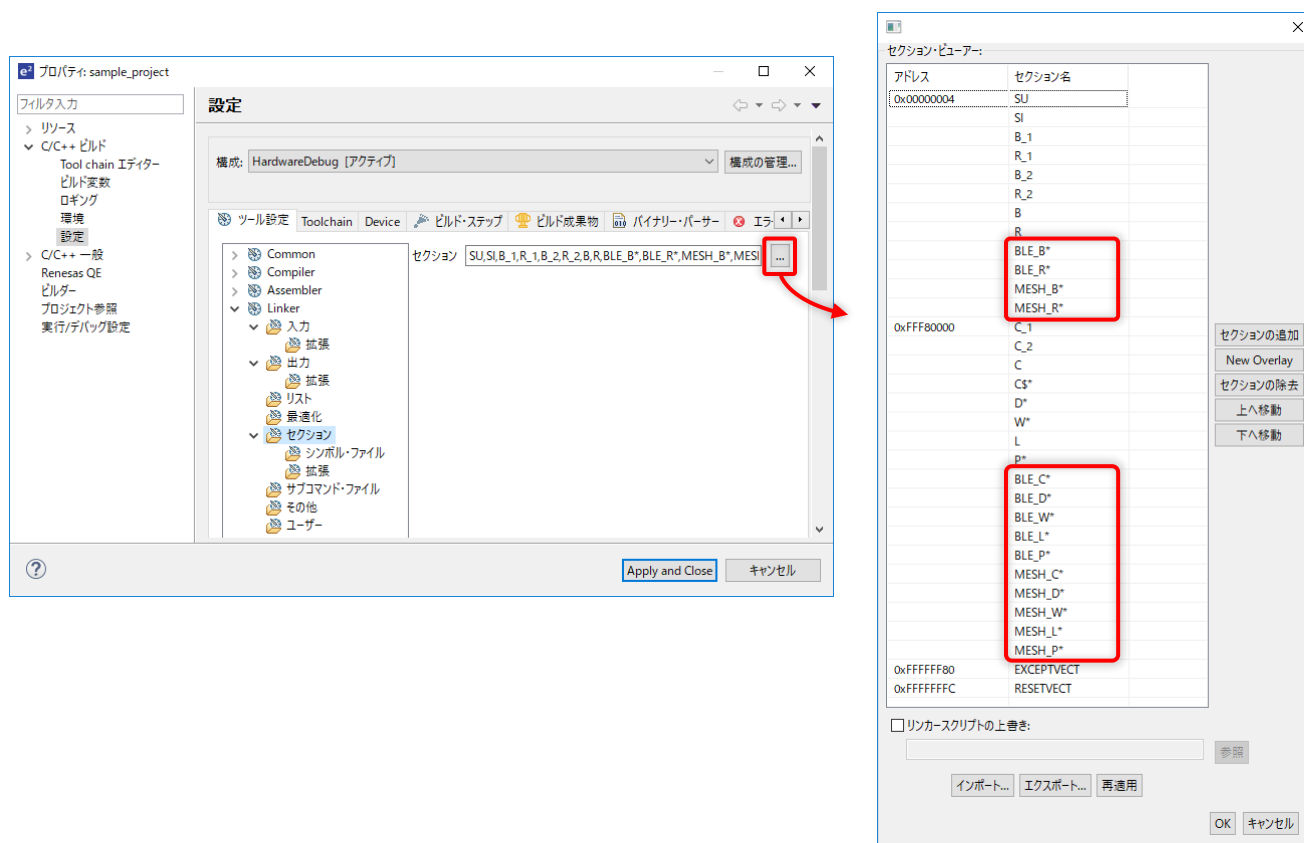


図 23 セクションの追加

(2) ROM から RAM へのマッピング

[プロパティ]ダイアログの[ツール設定]タブで[Linker]→[シンボル・ファイル]を選択します。[ROM から RAM へマップするセクション]に下記を追加します。

BLE_D=BLE_R

BLE_D_1=BLE_R_1

BLE_D_2=BLE_R_2

MESH_D=MESH_R

MESH_D_1=MESH_R_1

MESH_D_2=MESH_R_2

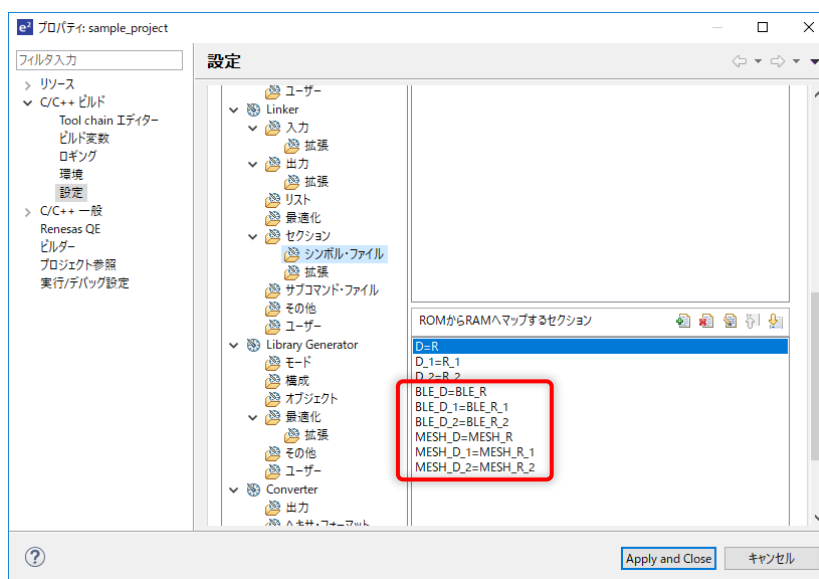


図 24 ROM から RAM へマップするセクションの設定

5.7.2 ライブラリ

メッシュ FIT モジュールのメッシュスタックライブラリ、BLE FIT モジュールの BLE プロトコルスタックライブラリをリンクオプションに追加する必要があります。

(1) ライブラリの追加

[プロパティ]ダイアログの[ツール設定]タブで[Linker]→[入力]を選択します。[リンクするリロケートابل・ファイル、ライブラリ・ファイルおよびバイナリ・ファイル]に下記が追加されていることを確認します。

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

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

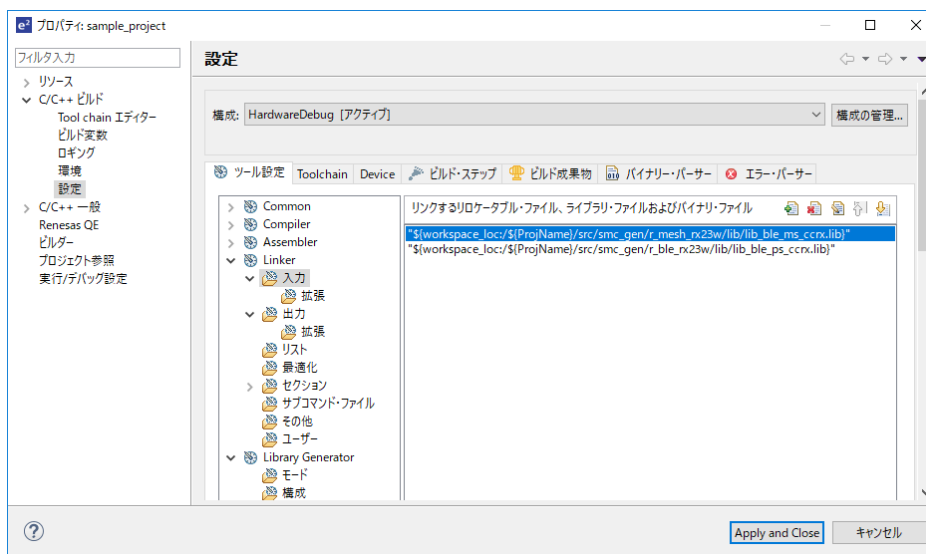


図 25 ライブラリ・ファイルの設定

(2) ビルド前コマンドの追加

BLE FIT モジュールの BLE プロトスタックライブラリを使用するため、[プロパティ]ダイアログの [ビルド・ステップ] タブで、[ビルド前のステップ] → [コマンド] に下記のコマンドを追加します。

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

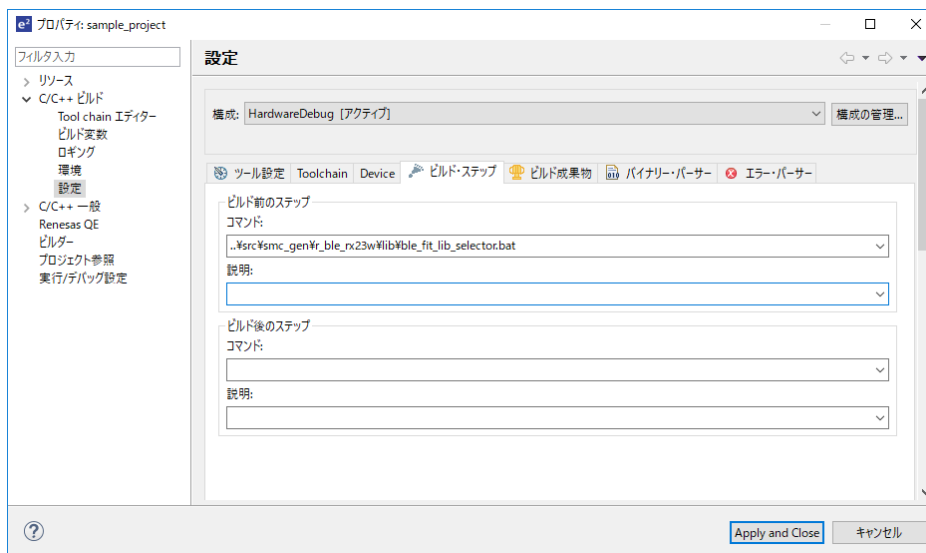


図 26 ビルド前コマンドの追加

リンカ設定の完了後は、[プロパティ]ダイアログの[Apply and Close]ボタンをクリックします。

5.8 デバッグ設定

[実行]メニューの[デバッグの構成]を選択します。[Renesas GDB Hardware Debugging]で{プロジェクト名} HardwareDebug を選択し、RX23W でのソフトウェアデバッグに必要な設定を行います。

5.8.1 デバッグの接続

[Debug hardware]で使用するデバッグガを選択します。Target Board for RX23W のオンボードエミュレータを使用する場合は"E2 Lite (RX)"を選択してください。

[Target Device]で使用するデバイスを選択します。Target Board for RX23W または RSSK for RX23W を使用する場合は、"RX"→"RX23W"→"R5F523W8"を選択してください。

Target Board for RX23W または RSSK for RX23W を使用する場合は、[クロック]→[メイン・クロック・ソース]を"HOCO"に設定します。また[電源]→[エミュレーターから電源を供給する]に"いいえ"を設定します。

注: Target Board for RX23W は出荷時、RX23W の ID コードプロテクト機能が有効になっています。ファームウェアの書き込み時に「id コードの設定に失敗しました」と表示される場合は、[Debugger] タブの[Connection Settings]タブで[フラッシュ]→[ID コード]に"45FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF"を設定してください。

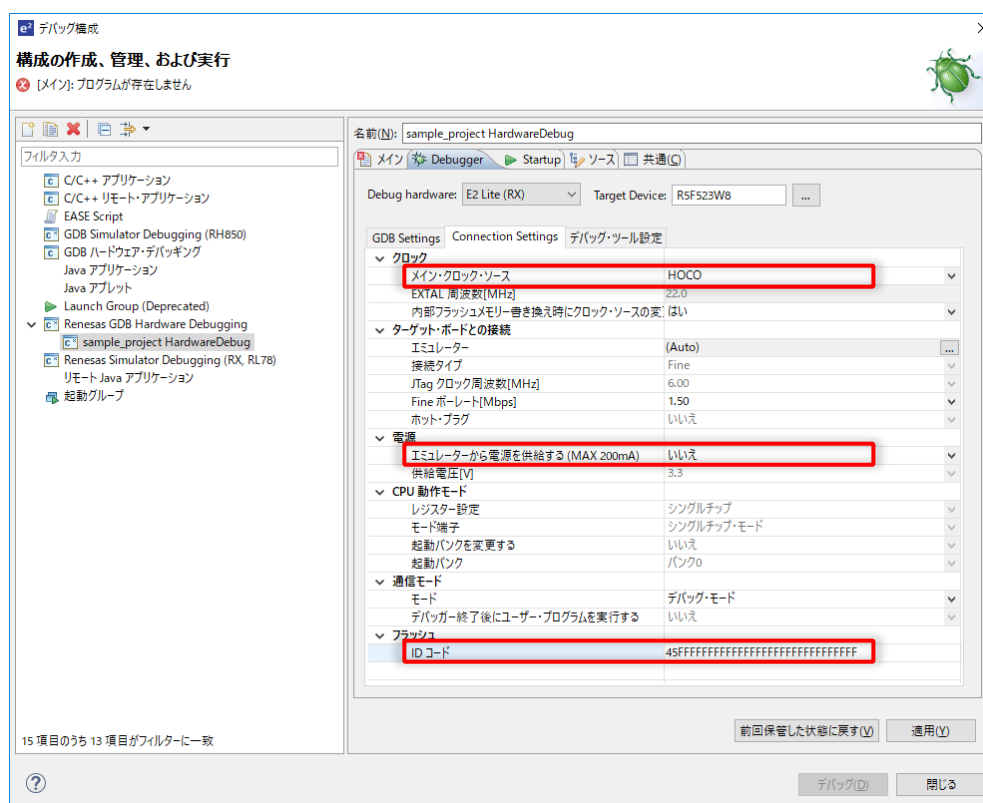


図 27 デバッグ接続の設定

5.9 プロジェクトのビルド

プロジェクトをビルドするには、[プロジェクト]メニューの[プロジェクトのビルド]を選択します。[コンソール]タブで、ビルドログに続いて"Build Finished"と表示されれば、プロジェクトのビルドは成功です。

e² studio でのデバッグについては「e² studio ユーザーズマニュアル 入門ガイド」([R20UT4374](#))の 5 章「デバッグ」を参照してください。

6. メッシュアプリケーションの実装方法

メッシュ FIT モジュールを使用するメッシュアプリケーションの実装方法は「RX23W グループ Bluetooth メッシュスタック 開発ガイド」([R01AN4875](#))を参照してください。

またメッシュ FIT モジュールパッケージ([R01AN4930](#))にはメッシュ FIT モジュールを使用したデモプロジェクトが含まれます。デモプロジェクトの実行方法は「RX23W グループ Bluetooth メッシュスタック スタートアップガイド」([R01AN4874](#))を参照してください。

商標権および著作権

Bluetooth® のワードマークおよびロゴは Bluetooth SIG, Inc が所有する登録商標であり、ルネサスエレクトロニクス株式会社はこれらのマークをライセンスに基づいて使用しています。その他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。

RX23W グループ Bluetooth メッシュスタックは次のオープンソースソフトウェアを使用します。

- [crackle](#); AES-CCM, AES-128bit 機能
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.

プログラム更新内容 (MESH FIT モジュール)

■ Rev1.01

- Bluetooth ペアラーラッパー

blebrr.c

- アドバタイジング送信タイミングにランダムディレイを追加

■ Rev1.10

- Bluetooth メッシュスタック

Common

- システムタイム関数を登録する MS_systemtime_init_pl() を追加。
- メッシュモニタ機能を追加し、MS_monitor_register_pl() を追加。

Provisioning

- 関数名にプレフィックスを追加。

prov_set_device_oob_pubkey_pl()	→ MS_prov_set_device_oob_pubkey_pl()
prov_set_static_oob_auth_pl()	→ MS_prov_set_static_oob_auth_pl()
prov_clear_device_oob_pubkey_pl()	→ MS_prov_clear_device_oob_pubkey_pl()
prov_clear_static_oob_auth_pl()	→ MS_prov_clear_static_oob_auth_pl()
mempool_init_pl()	→ MS_mempool_init_pl()
storage_init_pl()	→ MS_storage_register_pl()
- MS_prov_setup() の引数 role と bearer の実装順序を修正。

Network

- Network メッセージキャッシュの実装を改善し、MS_CONFIG 構造体にノードあたりのキャッシュ数を指定するパラメータ config_MS_NET_SEQNUM_CACHE_SIZE を追加。
- Network レイヤーの送信状態を確認する MS_net_register_tx_state_access() を追加。
- IV Update の 96 時間チェックを無効化する MS_access_set_iv_update_test_mode() を追加。
- IV Index 更新時に Network レイヤーと Lower Transport レイヤーの送信状態チェックを追加。
- IV Update 完了時に SEQ 番号をリセットする処理を追加。
- MCU リセット時に Secure Network Beacon 送信を復帰する処理を追加。
- Proxy Advertisement を即時停止するように MS_proxy_server_adv_stop() の実装を修正。
- Proxy 接続の切断と再確立時の Proxy フィルタリスト管理の実装を修正。

Lower Transport

- Lower Transport レイヤーの送信状態を確認する MS_ltrn_register_tx_state_access() を追加。
- マルチキャストアドレス向けセグメントメッセージ受信に対して Acknowledgement メッセージを送信しないよう修正。

Access

- MS_access_cm_set_model_publication() に Publication アドレスの有効チェックを追加。
- MS_access_cm_set_iv_index() が返却するエラーコードを追加。
 - ・ ACCESS_IV_VAL_NOT_PERMITTED
 - ・ ACCESS_IV_UPDATE_TOO_SOON
 - ・ ACCESS_IV_INCORRECT_STATE
 - ・ ACCESS_IV_UPDATE_DEFERRED_IN_BUSY
- MS_access_cm_reset() の実行で下記機能の設定をクリアするように修正。
 - ・ Relay
 - ・ Proxy
 - ・ Friend
 - ・ Low Power
 - ・ Secure Network Beacon

Model

- 下記 Client モデルコールバックの第 1 引数型を MS_ACCESS_MODEL_REQ_MSG_CONTEXT に変更。
 - MS_CONFIG_MODEL_CB : Configuration Client モデル
 - MS_HEALTH_CLIENT_CB : Health Client モデル
 - MS_GENERIC_ONOFF_CLIENT_CB : Generic OnOff Client モデル
 - MS_GENERIC_LEVEL_CLIENT_CB : Generic Level Client モデル
 - MS_GENERIC_DEFAULT_TRANSITION_TIME_CLIENT_CB : Generic Default Transition Time Client モデル
 - MS_GENERIC_POWER_ONOFF_CLIENT_CB : Generic Power OnOff Client モデル
 - MS_GENERIC_POWER_LEVEL_CLIENT_CB : Generic Power Level Client モデル
 - MS_GENERIC_BATTERY_CLIENT_CB : Generic Battery Client モデル
 - MS_GENERIC_LOCATION_CLIENT_CB : Generic Location Client モデル
 - MS_GENERIC_PROPERTY_CLIENT_CB : Generic Property Client モデル
 - MS_SENSOR_CLIENT_CB : Sensor Client モデル
 - MS_TIME_CLIENT_CB : Time Client モデル
 - MS_SCENE_CLIENT_CB : Scene Client モデル
 - MS_SCHEDULER_CLIENT_CB : Scheduler Client モデル
 - MS_LIGHT_LIGHTNESS_CLIENT_CB : Light Lightness Client モデル
 - MS_LIGHT_CTL_CLIENT_CB : Light CTL Client モデル
 - MS_LIGHT_HSL_CLIENT_CB : Light HSL Client モデル
 - MS_LIGHT_XYL_CLIENT_CB : Light xyl Client モデル
 - MS_LIGHT_LC_CLIENT_CB : Light LC Client モデル
- 周期的なメッセージパブリケーション処理の登録チェックを追加。
- MS_config_server_init() にコールバック関数を登録するための引数 appl_cb を追加。

Bluetooth ペアラーラッパー

blebrr.c, blebrr.h

- Advertising 送信制御に使用するタイマを BLE プロトコルスタックの内部タイマから BLE ソフトウェアタイマ(R_BLE_TIMER)に変更。
- 送信キューサイズ(BLEBRR_QUEUE_SIZE)を 32 から 64 に変更。
- Advertising 送信回数(BLEBRR_ADVREPEAT_COUNT)を 5 回から 3 回に変更。
- Advertising 送信ランダムディレイ幅(BLEBRR_ADVREPEAT_RAND_DELAY)を 7msec から 10msec に変更。
- Advertising と Scan の動作状態を独立管理に変更。
- malloc() によるデータバッファの確保に失敗した場合、送信キューがリークする問題を修正。

blebrr_pl.c, blebrr.h

- Static Random アドレスの取得処理を追加し、デフォルトのデバイスアドレスタイプ (BLEBRR_VS_ADDR_TYPE) を Public アドレスから Static Random アドレスに変更。
- GATT インタフェースを持つデバイスとの複数同時接続に対応するための実装を追加。
- Mesh GATT Client として接続確立後に Service Discovery を開始して、対向デバイスが Mesh GATT サービスを持つ場合、Notification を有効化する機能を追加。
- Advertising データに Mesh GATT サービスの UUID を含むデバイスを探索する機能を追加し、R_MS_BRR_Scan_GattBearer() を追加。
- 関数名にプレフィックスを追加。

blebrr_init()	→ R_MS_BRR_Init()
blebrr_register()	→ R_MS_BRR_Setup()
blebrr_register_gatt_iface_event_pl()	→ R_MS_BRR_Register_GattifaceCallback()
blebrr_gatt_mode_set()	→ R_MS_BRR_Set_GattMode()
blebrr_gatt_mode_get()	→ R_MS_BRR_Get_GattMode()
blebrr_disconnect_pl()	→ R_MS_BRR_Disconnect()
blebrr_set_adv_scanrsp_data_pl()	→ R_MS_BRR_Set_ScanRspData()
blebrr_create_conn_pl()	→ R_MS_BRR_Create_Connection()
blebrr_discover_service_pl()	→ R_MS_BRR_Discover_Service()
blebrr_config_ntf_pl()	→ R_MS_BRR_Config_Notification()

- GATT インタフェースイベントを追加。
 - ・ BLEBRR_GATT_IFACE_NOT_FOUND
 - ・ BLEBRR_GATT_IFACE_SCAN
- GATT インタフェースコールバック(BLEBRR_GATT_IFACE_CB)に引数 conn_hdl と引数 peer_addr を追加。

gatt_db_prov.c/h, gatt_db_proxy.c/h

- QE for BLE で生成された以下のサービスを持つ GATT データベースに置換。
 - ・ GAP サービス
 - ・ GATT サービス
 - ・ Mesh Provisioning サービスまたは Mesh Proxy サービス

gatt_clients.c

- 対向デバイスの Mesh GATT サービスから Client Characteristic Configuration ディスクリプタ(CCCD)を探索する処理を追加。
- 対向デバイスの Mesh GATT サービスから取得した Attribute Handle の有効性を確認する処理を追加。

- ドライバ

mesh_resource.h

- メモリプールサイズマクロ(MESH_MEMPOOL_SIZE)の定義を更新。
- ストレージサイズマクロ(MESH_STORAGE_SIZE)を追加。

mesh_dataflash.h

- データフラッシュ有効化コンパイルスイッチ(DATAFLASH_EN)のデフォルト設定を 0 から 1 に変更。

mesh_systemtime.c/h

- 8 ビットタイマ(TMR)を使用して 1msec 単位の 32bit システムタイムを生成するドライバを追加。

プログラム更新内容 (MESH FIT モジュールパッケージのメッシュサンプルプログラム)

■ Rev1.01

mesh_cli フォルダ

- Command Line Interface(CLI)プログラムを追加。

mesh_model.c

- ステート操作関数にエラーチェックを追加。
 - ・ mesh_model_onoff_server_state_get()
 - ・ mesh_model_onoff_server_state_set()

■ Rev1.10

vendor_model フォルダ

- 可変長のデータを送受信する Vendor モデルを追加。

mesh_core.c

- Bluetooth ペアラーラッパーの関数の仕様変更に伴って実装を更新。
 - ・ R_MS_BRR_Set_GattMode()
 - ・ R_MS_BRR_Get_GattMode()
 - ・ R_MS_BRR_Register_GattIfcCallback()
- Bluetooth ペアラーラッパーのコールバック関数の仕様変更に伴って実装を更新。
 - ・ BLEBRR_GATT_IFACE_CB
- BLEBRR_GATT_IFACE_CB GATT インタフェースを持つデバイスとの複数同時接続に対応。
- GATT インタフェースイベントの BLEBRR_GATT_IFACE_NOT_FOUND で接続を切断する処理を追加。
- GATT インタフェースイベントの BLEBRR_GATT_IFACE_SCAN で Mesh GATT サービスを持つデバイスをログ表示する処理を追加。
- Provisioning の完了後、Proxy Advertisement の Identification Type が Node Identity となるように修正。
- 送受信するメッセージの SEQ が閾値(CORE_NET_IV_UPDATE_INIT_THRESHOLD)を超えると、IV Update プロシージャを開始する処理を追加。
- Low Power ノードとして Friend ノードと Friendship を確立し、Friend Subscription List を追加する処理を追加。
- Proxy Client として接続確立後、Subscription アドレスを対向 Proxy Server の Proxy Filter List に登録する処理を追加。
- メッシュスタックの全レイヤーの送受信 PDU と SNB をログ出力する処理を追加。
- LED 点滅のタイミングをプロビジョニング中から GATT インタフェースの接続中に変更。

mesh_model.c

- Composition Data ステートの Element Location を設定するための ELEMENT_DESC_LOCATION マクロを追加。
- Generic OnOff Set メッセージ送信関数の引数に TID(Transaction ID)を追加。
- Vendor Client モデルと Vendor Server モデルのメッセージ送信関数、コールバック関数を追加。
- Bluetooth メッシュスタックの関数の仕様変更に対応。
 - ・ MS_config_server_init()
- Bluetooth メッシュスタックのコールバック関数型の仕様変更に伴って実装を更新。
 - ・ MS_GENERIC_ONOFF_CLIENT_CB

main.c

- Bluetooth ペアラーラッパーの関数の仕様変更に伴って実装を更新。
 - ・ R_MS_BRR_Init()
 - ・ R_MS_BRR_Setup()
- Bluetooth ペアラーラッパーのコールバック関数型の仕様変更に伴って実装を更新。
 - ・ BLEBRR_INIT_CB
- Vendor Client モデルを使用し、コンソールで入力された文字列を送信する処理を追加

- Vendor Server モデルを使用し、受信した文字列をコンソールに表示する処理を追加。
- MCU リセット直後のオンボードスイッチ(SW1)の押下で、ノードコンフィグレーションをリセットする処理を追加。
- Config Node Reset メッセージの受信で MCU をリセットする処理を追加。

mesh_appl.h

- コンパイルスイッチを追加。

• IV_UPDATE_INITIATION_EN	IV Update プロシージャ
• LOW_POWER_FEATURE_EN	Low Power 機能
• CONSOLE_MONITOR_CFG	メッシュモニタ機能
• ANSI_CSI_EN	ANSI エスケープシーケンス
• CPU_USAGE_EN	CPU 使用率測定
- メッシュモデルのメッセージ受信コールバック関数を追加。

• onoff_server_set_cb	Generic OnOff Set メッセージ
• onoff_client_status_cb	Generic OnOff Status メッセージ
• vendor_server_set_cb	Vendor Set メッセージ
• vendor_client_status_cb	Vendor Status メッセージ
• config_server_node_reset_cb	Config Node Reset メッセージ
- コンソールへのログ出力マクロを追加。
- CPU 使用率測定マクロを追加。

改訂記録

Rev.	発行日	改定内容	
1.00	2019.10.11	-	初版発行
1.01	2019.11.29	P.8	表 2、表 3 のプログラムサイズを更新
		P.10	表 5 から BSP_CFG_ID_CODE_LONG_1 を削除
		P.18	5.4.1 項から ID コードの設定手順を削除
		P.31	5.8.1 項に ID コードの設定に関する注記を追加
1.10	2020.09.30	P.6	ハードウェア要件に8-Bit Timerを追加
		P.6	ソフトウェア要件のr_cmt_rxバージョンを4.5以降に変更
		P.8	表 2、表 3のプログラムサイズを更新
		P.9	表 4にMESH_CFG_NET_SEQNUM_CACHE_SIZEマクロを追加
		P.25	5.6節から「r_cmt_rx」項を削除
		P.25	5.6.1項「r_ble_rx23w」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イやマガジンケース、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、入力信号や入出力プルアップ電源を入れないでください。入力信号や入出力プ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えば、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。お客様の機器・システムの設計において、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には、お客様の責任において行ってください。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、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、全部または一部を問わず、改造、改変、複製、リパ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、リパ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。
5. 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）、交通管制（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。

6. 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。
7. 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。
8. 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。
9. 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。
10. 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。
11. 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。
12. 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.4.0-1 2017.11)

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

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/

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。