

# Technical Note

TecStar

## Silicon Labs 社 BGM1xx クイックスタートガイド

---

2018 年 10 月

株式会社 **マクニカ**  
テクスター カンパニー

### クイックスタートガイド

#### 目次

1 はじめに .....	4
2 BGM1xx の概要 .....	5
2-1 製品ラインナップ .....	5
2-2 Bluetooth 5.0 への対応状況 .....	5
2-3 モジュールの制御方法 .....	6
3 開発環境のご紹介 .....	7
3-1 ハードウェア .....	7
3-1-1 Blue Gecko Bluetooth Smart Module Wireless Starter Kit .....	7
3-1-2 Simplicity Debug Adaptor Board (SLSDA001A) .....	8
3-2 ソフトウェア .....	9
3-2-1 Simplicity Studio .....	9
4 各種ドキュメントの入手方法 .....	10
4-1 ドキュメントの入手方法 (Simplicity Studio から) .....	10
4-1-1 情報が表示されない場合には? .....	12
4-1-2 欲しい情報が見つからない場合には? .....	12
4-1-3 表示される情報を制限したい場合には? .....	13
4-1-4 いつも使うドキュメントに素早くアクセスしたい場合には? .....	13
4-2 ドキュメントの入手方法 (Web から) .....	14
4-3 最初に読むべきドキュメント .....	15
4-4 API のドキュメントはどれですか? .....	15
4-5 評価基板の回路図・レイアウト図・部品表はどこから入手できますか? .....	15
5 ソフトウェア・インストール .....	16
5-1 Simplicity Studio / Bluetooth SDK のインストール .....	16
5-2 インストールがうまくいかない場合 .....	21
5-2-1 シリコンラボ社アカウントの取得方法 .....	21
5-2-2 企業プロキシサーバーを介して接続している場合 .....	22
5-2-3 プロキシ設定をしてもインストールがうまくいかない場合 .....	24
5-2-4 オフライン・インストーラ .....	25
5-2-5 Install Manager / Install Wizard の画面を閉じてしまいました .....	25
5-3 IAR コンパイラのインストール (オプション) .....	26
6 ハードウェア・セットアップ .....	27
6-1 Wireless Starter Kit のセットアップ .....	27
6-2 Wireless Starter Kit の制御ファームウェアの更新 .....	28

<b>7 使用方法</b> .....	<b>29</b>
7-1 サンプルコードを動かしてみる前に(ブートローダーの更新) .....	29
7-2 サンプルコードを動かしてみる(C言語編) .....	31
7-3 OTA update (over-the-air)を試してみる.....	37
7-4 ユーザ基板のプログラミング・デバッグを行ってみる .....	40
7-4-1 参考資料.....	40
7-4-2 ハードウェア接続.....	41
7-4-3 デバッグ対象の切り替え.....	43
7-5 VCOM を利用した printf デバッグ.....	46
7-6 BGTool を使って評価する (NCP モード).....	54
7-7 RF PHY の特性を評価する .....	58
7-7-1 テストコマンドを使用する .....	58
7-7-2 BGTool を使用する .....	60
7-8 消費電流を測定してみる (Energy Profiler) .....	61
<b>8 ソフトウェア設計</b> .....	<b>64</b>
8-1 ソースコードの追い方 .....	64
8-2 サンプルコードにペリフェラルを実装してみる (外部割込み) .....	65
8-2-1 サンプルコードを理解する (SLSTK3401A_gpio_int_pg1b) .....	66
8-2-2 サンプルコードを理解する (SOC - iBeacon).....	69
8-2-3 ペリフェラル設定を移植する .....	73
8-3 こんなサンプルコードはありませんか? .....	78
8-4 新バージョン SDK への移行手順 .....	79
<b>9 トラブルシューティング</b> .....	<b>80</b>
9-1 動作障害.....	80
9-2 ツール障害.....	80
<b>参考文献</b> .....	<b>81</b>

## 1 はじめに

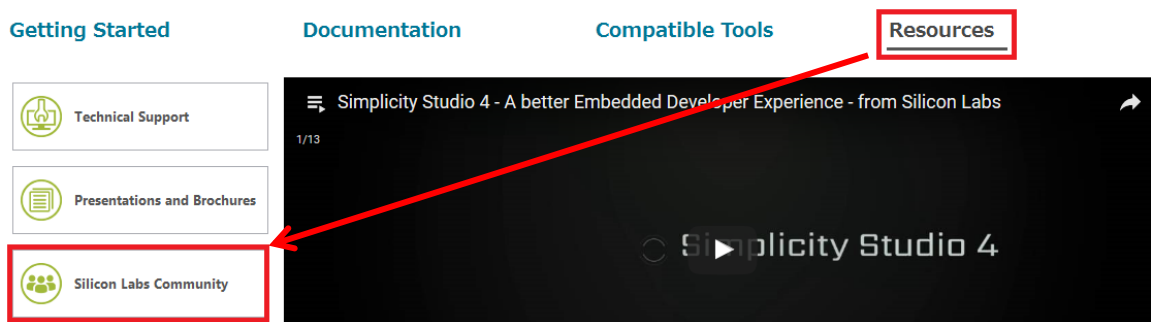
この資料は、Silicon Laboratories(以下、Silicon Labs)社製 Bluetooth®モジュール BGM1xx の開発環境について簡易にまとめたものです。内容に誤りがないよう注意は払っておりますが、もし Silicon Labs 社が提供するドキュメント等と差異がございましたら、メーカー提供のものを優先してご参照ください。

また、Silicon Labs 社の ナレッジベース(FAQ)やコミュニティフォーラム(ユーザ同士で問題解決。Silicon Labs のエンジニアも頻繁にコメントしています)には、本資料で取り上げていない様々な情報が記載されております。

製品をご使用頂く過程で疑問や課題が生じることもあると思いますが、他のユーザが既に解決方法を見つけている場合も多々ございます。非常に有益ですので、ぜひご活用下さい。

### ◆ アクセス方法

Simplicity Studio から



Web Site から

<https://www.silabs.com/community> (Silicon Labs 社製品全般)

<https://www.silabs.com/community/wireless/bluetooth> (Bluetooth に特化)

### ◆ 使用方法



## 2 BGM1xx の概要

BGM1xx は、シリコンラボ社 Blue Gecko (EFR32BG) を使用した、Bluetooth Low Energy 対応の Bluetooth®モジュールです。

モジュールを使用するメリットとして、主に以下が挙げられます。

- ハードウェア設計にかかる時間とコストが最小限で済み、早く市場に製品を投入できます。
- 無線性能を最大限に引き出すことができます。
- 最終製品としての認証 (Bluetooth®認証/各国認証) が最小限で済み、早く市場に製品を投入できます。

シリコンラボ社の BGM1xx ファミリーは、上記に加え、特に出力レベルの高さ、省スペースに特長があります。また ARM Cortex-M4F を搭載していますので、処理性能の高さも特長の 1 つです。

### 2-1 製品ラインナップ

ラインナップは以下の通りです。

型番	出力レベル	受信感度	サイズ	無線チップ	アンテナ
BGM111	+ 8 dBm	- 92 dBm	12.9 x 15 x 2.2 mm	EFR32BG1B232F256GM48	内蔵 or U.FL *1
BGM113	+ 3 dBm	- 92 dBm	9.2 x 15.8 x 1.83 mm	EFR32BG1B132F256GM32	内蔵
BGM121	+ 8 dBm	- 90 dBm	6.5 x 6.5 x 1.4 mm	EFR32BG1B232F256GM56	内蔵 or RF ピン *1
BGM123	+ 2 dBm	- 90 dBm	6.5 x 6.5 x 1.4 mm	EFR32BG1B232F256GM56	内蔵 or RF ピン *1
BGM11S12	+ 2 dBm	- 90 dBm	6.5 x 6.5 x 1.4 mm	EFR32BG1B232F256GM56	内蔵
BGM11S22	+8 dBm	- 90 dBm	6.5 x 6.5 x 1.4 mm	EFR32BG1B232F256GM56	内蔵
BGM13P22	+ 8 dBm	- 94.8 dBm	12.9 x 15.0 x 2.2 mm	EFR32BG13P22F512	内蔵 or U.FL *1
BGM13P32	+19 dBm	- 94.8 dBm	12.9 x 15.0 x 2.2 mm	EFR32BG13P32F512	内蔵 or U.FL *1
<b>NEW</b> <b>NEW</b> BGM13S22	+ 8 dBm	-94.1 dBm	6.5 x 6.5 x 1.4 mm	EFR32BG13P22F512	内蔵 or RF ピン *1
BGM13S32	+ 18 dBm	-94.1 dBm	6.5 x 6.5 x 1.4 mm	EFR32BG13P32F512	内蔵 or RF ピン *1

\*1: 別型番にて提供

### 2-2 Bluetooth 5.0 への対応状況

Bluetooth 5.0 の各オプション機能への対応状況は以下の通りです。

型番	2M PHY	LE long range	アダプタサイズ拡張
BGM111	×	×	○
BGM113	×	×	○
BGM121	×	×	○
BGM123	×	×	○
BGM11Sxx	×	×	○
BGM13Pxx	○	○	○
BGM13Sxx	○	○	○

## 2-3 モジュールの制御方法

BGM1xx の制御方法は大きく分けて 2 通りあります。以前の SDK で対応していた”BGScript”は、**Bluetooth SDK 2.3.2 を最後にサポート終了**しております。

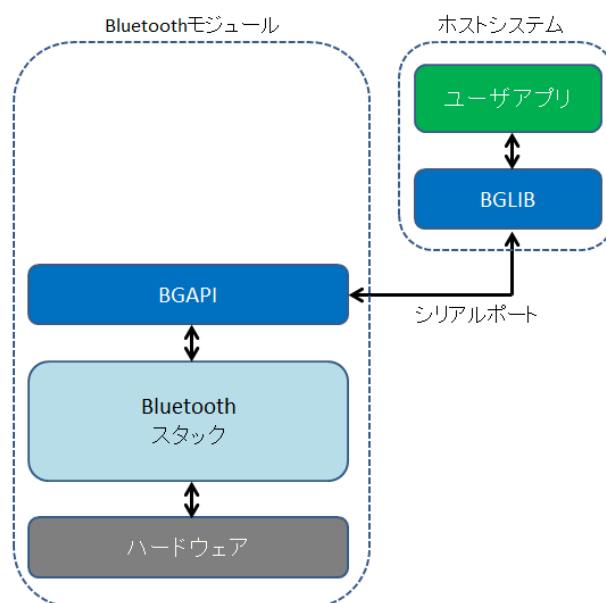
なお、下図はイメージしやすいように簡略化したものです。実際には Bluetooth スタックからも EMLIB を使用しているなど、下図とは相違があります。

### ◆ ネットワーク・コプロセッサ (NCP) モード

外部のホストシステム (マイコンや PC) からの制御により動作するモードです。Silicon Labs 社のドキュメントでは Network Co-Processor (NCP) モードという名称で紹介されています。

モジュールの制御は、ホストシステムからはシリアルポートを介して行い、API (BGAPI) が用意されています。また、ホストシステムのソフト設計を助けるため、ライブラリ (BGLIB) が用意されています。

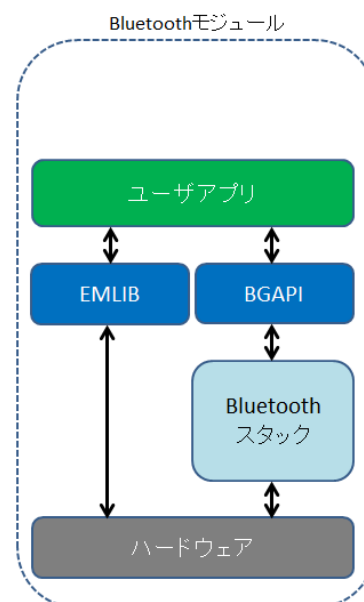
モジュールに搭載したマイコンでは性能不十分の場合や、ユーザアプリをホストシステムに載せた方がシステム設計の観点で都合が良い場合、などにご活用頂けます。



### ◆ C 言語設計

モジュールを単なるハードウェアの器として扱い、ソフトウェアについてはワイヤレスマイコンとして C 言語設計する方法です。柔軟性の高さがメリットで、ワイヤレスマイコンの持つ機能を存分に活用頂けます。

Bluetooth 制御用の API、ペリフェラルの制御用の API (EMLIB) が用意されています。ペリフェラル制御用の API (EMLIB) は、マイコン単品 (EFM32 シリーズ) と共用 API になっています。



## 3 開発環境のご紹介

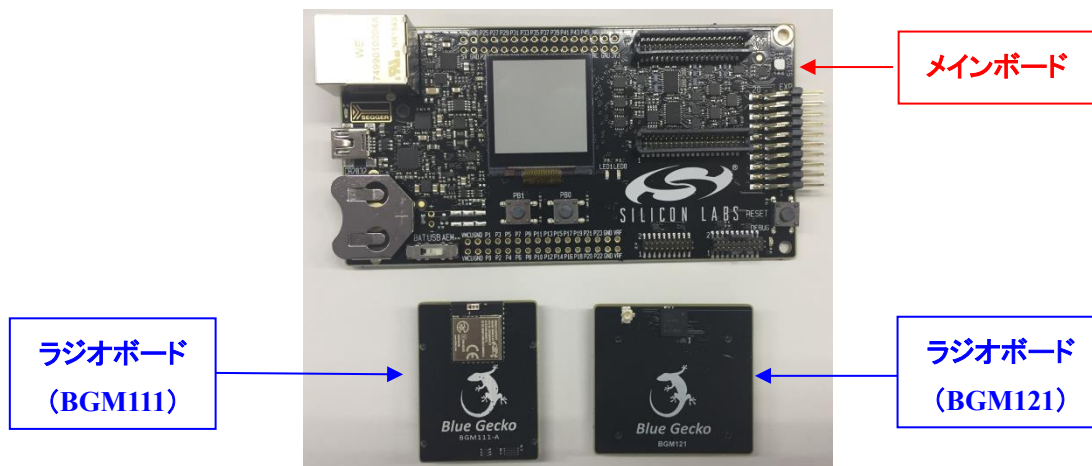
BGM1xx の開発環境について、ハードウェアとソフトウェアに分けてご紹介します。

### 3-1 ハードウェア

開発環境として、Blue Gecko Bluetooth Smart Module Wireless Starter Kit(以後、Wireless Starter Kit)を用意しています。

#### 3-1-1 Blue Gecko Bluetooth Smart Module Wireless Starter Kit

Wireless Starter Kit には、Wireless Starter Kit メインボード(以後、メインボード)と、Wireless Starter Kit ラジオボード(以後、ラジオボード)が含まれています。ラジオボードをメインボードのソケットに装着して使用します。



ラジオボードのラインナップは以下の通りです。

モジュール名	ラジオボード名	出力レベル	コメント
BGM111	SLWRB4300A	+ 8 dBm	
BGM113	SLWRB4301A	+ 3 dBm	
BGM121	SLWRB4302A	+ 8 dBm	
BGM123	N/A	+ 3 dBm	BRD4302A をご利用ください
BGM11S12	SLWRB4303A	+ 8 dBm	
BGM11S22	SLWRB4303A	+ 8 dBm	
BGM13P22	SLWRB4306A	+ 8 dBm	
BGM13P32	SLWRB4306B	+ 19 dBm	
<b>NEW</b> <b>NEW</b> BGM13S22	SLWRB4305C	+ 8 dBm	
BGM13S32	SLWRB4305A	+ 18 dBm	

Wireless Starter Kit に同梱されているラジオボードは以下の通りです。

ファミリ名	メインボード	ラジオボード			
		SLWRB4300A (BGM111)	SLWRB4301A (BGM113)	SLWRB4302A (BGM121)	
SLWSTK6101B(旧品番)	○	○	○		
SLWSTK6101C	○	○		○	

### 3-1-2 Simplicity Debug Adaptor Board (SLSDA001A)

Wireless Starter Kit とユーザ基板とを接続する際に使用するコネクタです。評価基板 Thunderboard React へプログラミングを行う際にも使用します。



SLSDA001A



このように接続して使う

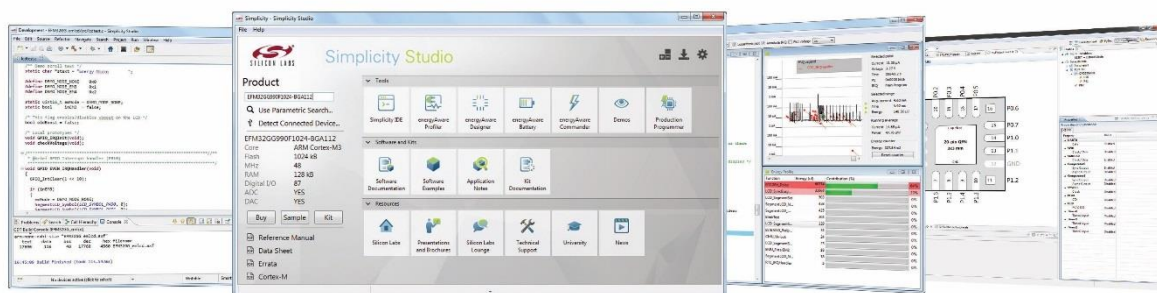


## 3-2 ソフトウェア

BGM1xx の開発環境には Simplicity Studio がご使用頂けます。

### 3-2-1 Simplicity Studio

Simplicity Studio は、BGM1xx をターゲットとしたコンパイル・デバッグ・プログラミングを 1 つのプラットフォームで提供することができるソフトウェアです。統合開発環境 (IDE) を中心に、非常に便利なツール群が充実しています。同社製の 32bit MCU や 8bit MCU も同一プラットフォームで開発が可能です。



注) 画像は Simplicity Studio v3 のものです

PC をネットワーク・コプロセッサ (NCP) モードのホストシステムに見立てた評価が行える BG Tool や、生成したバイナリをダウンロードする際に使う Simplicity Commander や、パケットトレースできる Network Analyzer も、Simplicity Studio 上に統合されています。

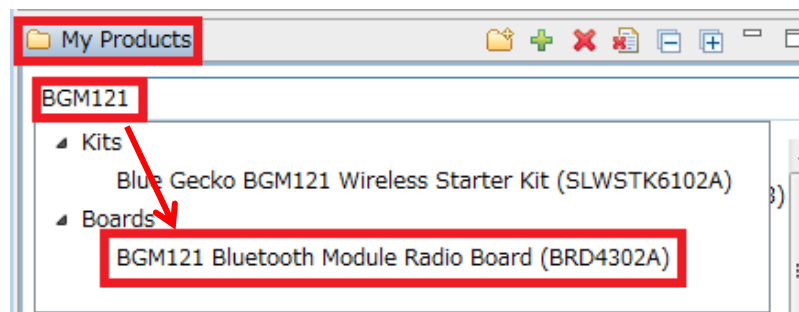
## 4 各種ドキュメントの入手方法

BGM1xx のドキュメントの入手方法について紹介します。

### 4-1 ドキュメントの入手方法（Simplicity Studio から）

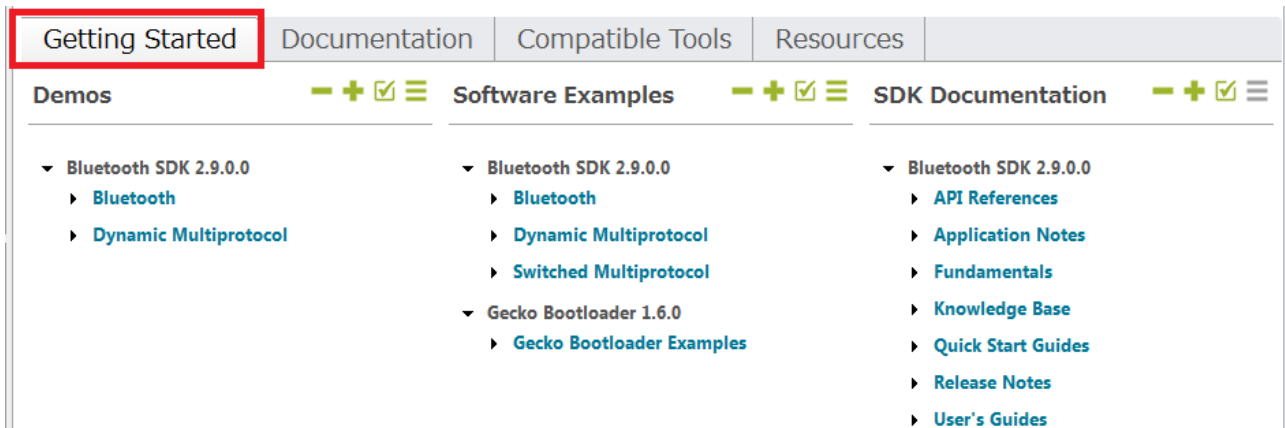
BGM1xx のデータシート、リファレンス・マニュアル、エラッタ、アプリケーションノート および 評価基板 (starter kit) の回路情報などは、Simplicity Studio からご入手頂くことが可能です。

Simplicity Studio を起動し、My Products タブ ⇒ 空欄に使用する製品型番を入力 ⇒ 候補の中から該当する型番を選択します。



製品型番を指定すると、関連するドキュメントやサンプルコードが自動でリストアップされます。情報の種別に応じて、Getting Started、Documentation、Compatible Tools、Resources というタブに分類されています。

#### ◆ Getting Started タブ



#### Demos:

評価基板上で動作するデモンストレーション用のソフトです。Build することなくモジュールに書き込んで、動作を確認することができます。

#### Software Example:

評価ボード上で動作するサンプルコードです。ソフトの実装方法について学んだり、機能について理解したりするのに役立ちます。Bluetooth アプリ (Bluetooth スタック+ユーザコード) と、ブートローダー単体 (Gecko Bootloader) のサンプルコードが用意されています。

## SDK Documentation:

Bluetooth SDK に関するドキュメントがまとめてあります。

- API References ... API の使用方法。
- Application Notes ... 特定の用例について記しています。
- Fundamentals ... Bluetooth の基礎などについてまとめています。
- Quick Start Guides ... ボードやツールの簡易取説。
- Release Notes ... SDK リリース時に追加・修正した機能や既知のバグ情報。
- User's Guide ... 各種ツールや設計手法などについて記しています。

### ◆ Documentation タブ

The screenshot shows a web interface with a navigation bar containing 'Getting Started', 'Documentation' (highlighted with a red box), 'Compatible Tools', and 'Resources'. Below the navigation bar, there are two main sections: 'My Favorite Documents' and 'All Documents'. 'My Favorite Documents' is currently empty, with a message: 'No documents have been favored. Click the 'Favorite' icon to add a document here.' 'All Documents' shows a list of documents for 'Gecko SDK Suite v2.3.0: Bluetooth 2.9.0.0, EmberZNet 6.3.0.0, Flex 2.3.0.0, Kernel, MCU 5.5.0.0, Micrium, OS, Thread 2.7.0.0 BGM121 Bluetooth Module Radio Board (BRD4302A)'. A list of document categories is shown with expandable arrows: API References, Application Notes, Data Sheets, Errata, Fundamentals, Knowledge Base, Quick Start Guides, Reference Manuals, Release Notes, Schematic and Layout Files, and User's Guides.

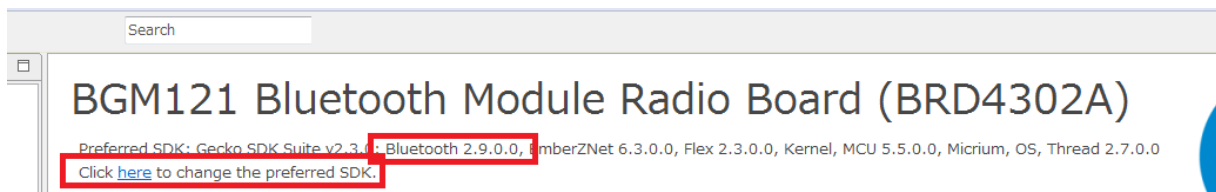
## All Documents:

各種ドキュメントがまとめてあります。前出の SDK Documentations と重複するものもあります。

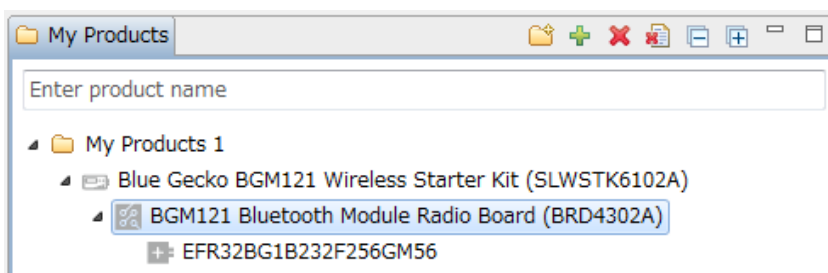
- API References ... API の使用方法。
- Application Notes ... 特定の用例について記しています。各ペリフェラル(ADC やシリアルインタフェースなど)の使用方法に関する情報も用意されています。
- Data Sheets ... モジュール内で使用されている SoC(EFR32BGxx)のデータシート。
- Errata ... モジュール内で使用されている SoC(EFR32BGxx)のバグ情報。
- Fundamentals ... Bluetooth の基礎などについてまとめています。
- Quick Start Guides ... ボードやツールの簡易取説。
- Reference Manual ... BGM1xx のデータシート、モジュール内で使用されている SoC(EFR32BGxx)の動作仕様書。
- Release Notes ... SDK リリース時に追加・修正した機能や既知のバグ情報。
- Schematic and Layout Files ... ラジオボードの回路図・部品表・レイアウト情報。
- User's Guide ... ラジオボードの取説、各種ツールや設計手法などについて記しています。

### 4-1-1 情報が表示されない場合には？


Demos, Documentation などに情報が表示されない場合には、SDK が適正に選択されていない可能性があります。下図を参考に、Bluetooth SDK が選択されているか確認してみてください。SDK が選択されていない場合には、[here](#) から SDK を選択してください。

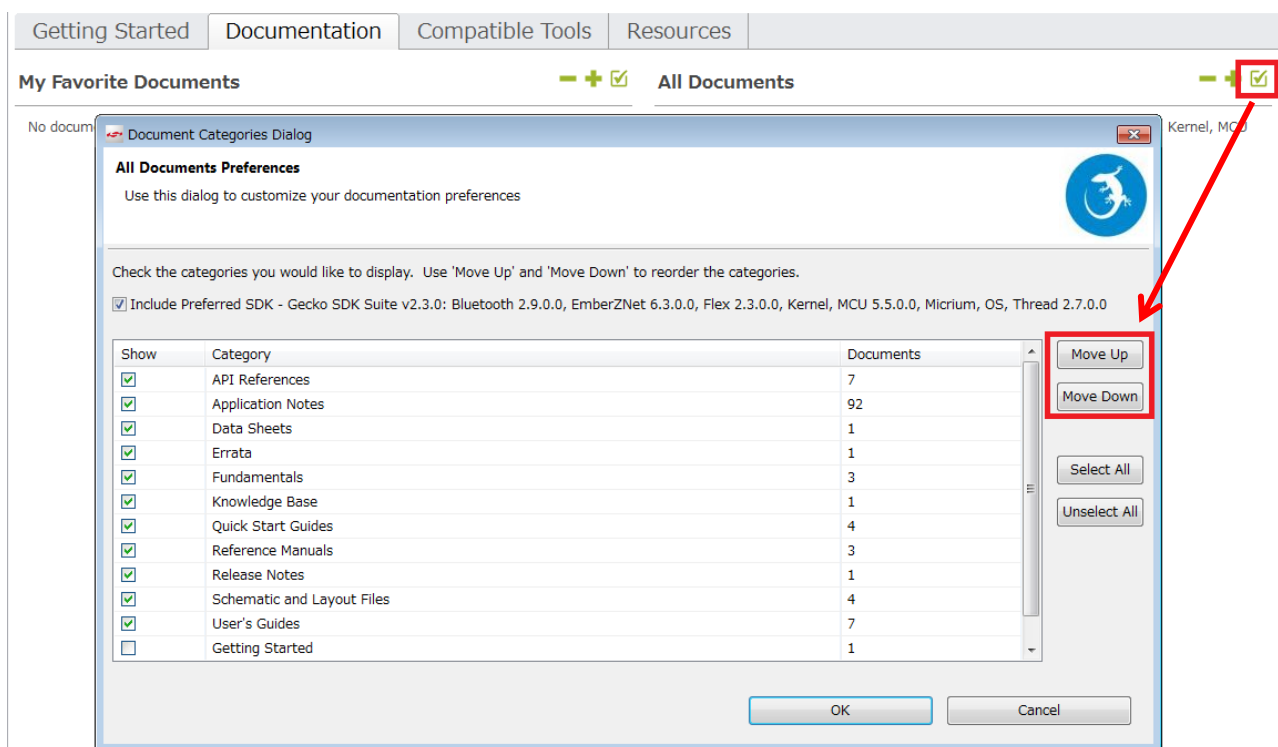


また、My Products タブで何を選ぶかで、表示される情報も変わりますので、その点も確認ください。







### 4-1-2 欲しい情報が見つからない場合には？

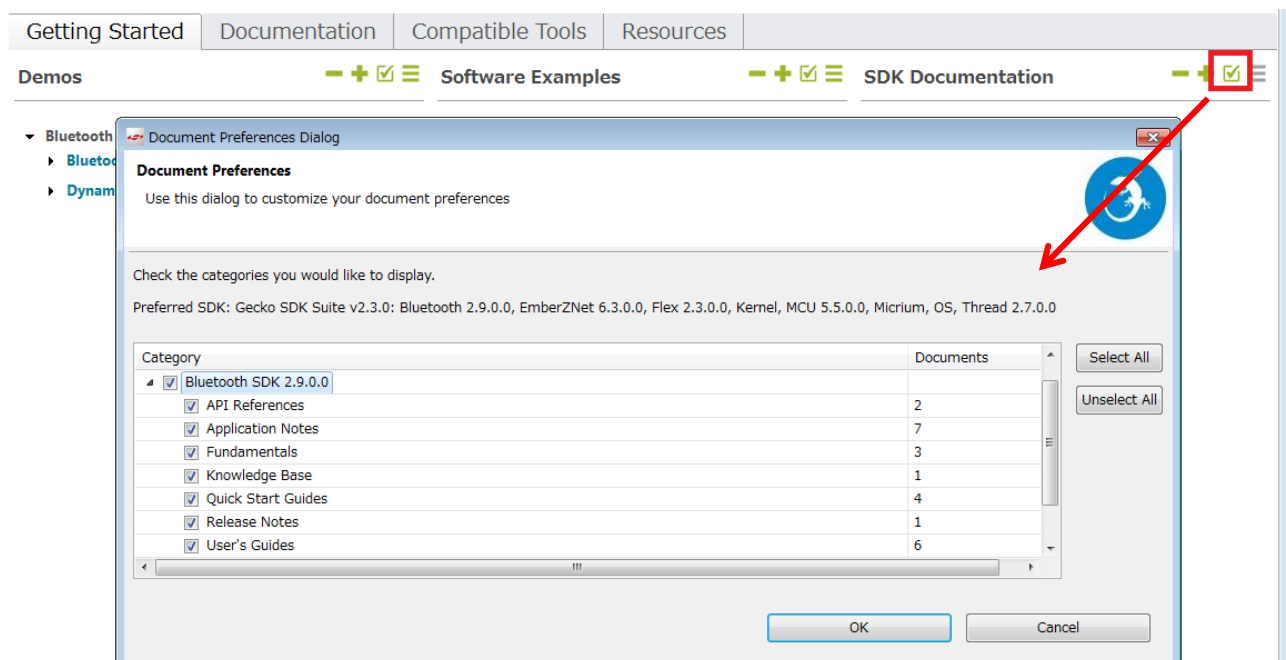
All Documents の右横にある  アイコンで、表示項目の選択や、表示項目の並び替え (Move Up / Move Down) を行うことができます。



### 4-1-3 表示される情報を制限したい場合には？

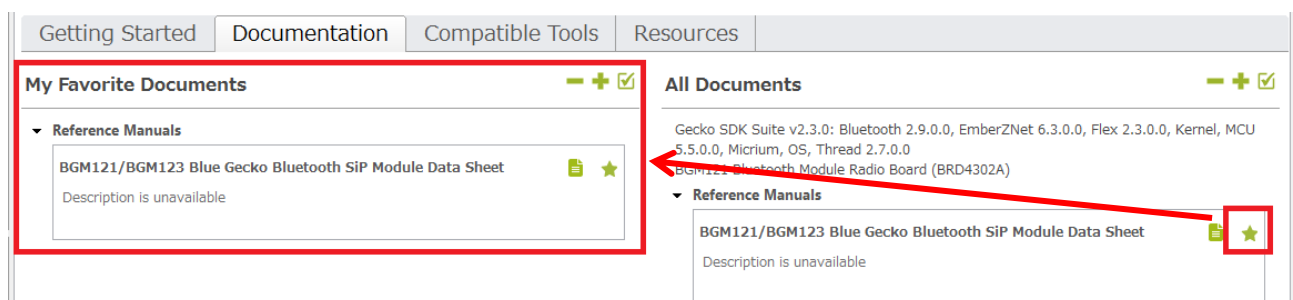
Demos, Software Examples, SDK Documentation の右横に、4つのアイコンが並んでいます。このアイコンを使うことで、表示される情報を制限することができます。

-  リストを折りたたむ
-  リストを展開する
-  表示する大項目を選択する
-  別の一覧表を表示する (Demos, Software Examples のみ)



### 4-1-4 いつも使うドキュメントに素早くアクセスしたい場合には？

各ドキュメントの右横にある☆印をクリックすると、☆の色が変わり、My Favorite Documents に追加されます。良く使うドキュメントを追加しておく便利です。



#### 4-2 ドキュメントの入手方法 (Web から)

BGM1xx のデータシート、リファレンス・マニュアル、エラッタ、アプリケーションノート および 評価基板 (starter kit) の回路情報などは、Silicon Labs 社の Web Site からのご入手可能です。[\(リンク\)](#)

Products や Resource Type で、リストアップする対象を絞り込むこともできます。

BGM1xx ファミリは、Products -> Wireless -> Bluetooth Low Energy -> Blue Gecko Bluetooth Low Energy Modules の下に分類されています。

Silicon Labs » Support » Technical Resource Search

### Technical Resource Search

Expand All / Collapse All    Showing 50 of 65 Results

**Narrow by:**

Products: Blue Gecko Bluetooth Low Energy Modules

Clear All

**Products** -

- Analog
- Audio and Radio
- Voice
- Wireless
  - Bluetooth Classic
  - Bluetooth Low Energy
    - Blue Gecko Bluetooth Low Energy Modules
    - Blue Gecko Bluetooth Low Energy SoCs
    - Bluegiga Bluetooth Low Energy Modules
  - Proprietary
  - Wi-Fi
  - ZigBee and Thread

Apply text filter

Title	Version	Resource Type
<a href="#">AN1036: BLE113 to BGM113 Migration Guide</a>	1.1	Application Notes
<a href="#">AN1037: Apple® HomeKit Over Bluetooth®</a>	0.2	Application Notes
<a href="#">AN1042: Using the Silicon Labs Bluetooth Stack in Network Co-Processor Mode</a>	0.3	Application Notes
⋮		
<a href="#">BGM111 Canada Certification</a>		Miscellaneous
<a href="#">BGM111 CE and Safety Reports</a>		Miscellaneous
<a href="#">BGM111 Japan Certificate and Report</a>		Miscellaneous
<a href="#">BGM113 Blue Gecko Bluetooth Smart Module Data Sheet</a>	1.00	Data Sheets
<a href="#">Blue Gecko Bluetooth Smart Module データシート</a>	1.00	Data Sheets
<a href="#">Blue Gecko Bluetooth Smart Module 数据表</a>	1.00	Data Sheets
<a href="#">and Report</a>		Miscellaneous
<a href="#">rts</a>		Miscellaneous
<a href="#">Report</a>		Miscellaneous
<a href="#">and Report</a>		Miscellaneous
<a href="#">(WSTK) Radio Board Bill</a>	A00	Schematic and Layout Files

**Resource Type** -

- Application Notes
- Data Sheet Addendums
- Data Sheets
- Errata
- Example Code
- Getting Started
- Manuals
- Miscellaneous
- Product Change Notifications (PCN)
- Reference Designs
- Release Notes
- Schematic and Layout Files
- Software

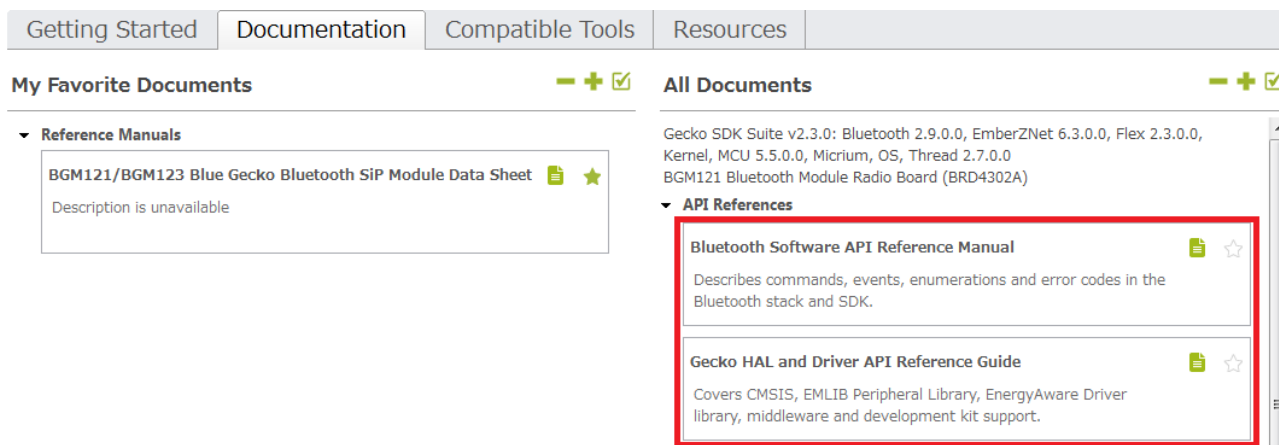
### 4-3 最初に読むべきドキュメント

QSG108「Getting Started with Silicon Labs Bluetooth® Software」([英語版](#), [日本語版](#))を最初にご覧ください。提供される Bluetooth SDK の構成、各種ツール、ドキュメント体系などをご紹介します。その上で、使用する設計手法に応じたドキュメントを読み進めて頂くのが効果的です。

### 4-4 API のドキュメントはどれですか？

BGM1xx のソフト設計には、2 種の API が用意されています。どちらも Getting Started タブ ⇒ SDK Documentation ⇒ API Reference もしくは Documentation タブ ⇒ All Documents ⇒ API References から入手できます。

- Bluetooth Software API Reference Manual  
Bluetooth スタックの制御 API です。アドバタイズ、スキャン、ボンディングなどの接続に関する制御 API や、DFU (Device Firmware Upgrade) 用の API、認証試験用のテスト API などを網羅しています。
- Gecko HAL and Driver API Reference Guide  
MCU やペリフェラル(無線部分を除く)を制御する際に使う API 群です。基本的に、EFM32 (32-bit MCU) と共用の API になっています。



### 4-5 評価基板の回路図・レイアウト図・部品表はどこから入手できますか？

Simplicity Studio からご入手頂けます。以下の情報を参照ください。

<マクニカオンラインサービス FAQ>

- [Wireless Starter Kit のメインボードの回路・レイアウト情報はどこから入手できますか？](#)
- [ラジオボード\(評価ボード\)の回路・レイアウト情報はどこから入手できますか？](#)

## 5 ソフトウェア・インストール

BGM1xx のスタックやサンプルコードは Bluetooth SDK に含まれています。その入手方法について、順を追ってご紹介します。

Bluetooth の開発・評価には、Simplicity Studio (開発環境)、Bluetooth SDK、C コンパイラをインストールする必要があります。Bluetooth のスタックやサンプルコードは Bluetooth SDK に含まれており、Simplicity Studio の一部としてインストールされます。また、ツール類 (IDE や Flash Programmer など) は Simplicity Studio に搭載されています。

ここでは、それらの入手方法について、順を追ってご紹介します。

### 5-1 Simplicity Studio / Bluetooth SDK のインストール

- ① 下記 URL より、「Windows Installer」をダウンロードします。64-bit OS 用となっていますので、32-bit OS 用が必要な場合には「クリックして Windows (32-bit) ...」からダウンロードしてください。

<http://jp.silabs.com/products/mcu/Pages/simplicity-studio.aspx>

ダウンロード時に Silicon Labs 社のアカウントが必要になります。お持ちでない場合には、本資料「5-2-1 シリコンラボ社アカウントの取得方法」を参考にご入手ください。

## Simplicity Studio 4

Simplicity Studio は、Eclipse 4.5 ベースの統合開発環境 (IDE) を使用して、開発者がプロジェクト完了に必要なすべてのものにワンクリックでアクセスできるようにすることで、IoT 開発プロセスを簡略化します。Simplicity Studio には、エネルギー・プロファイリング、構成、ワイヤレス・ネットワーク分析用のパワフルなツール・セットの他に、デモ、ソフトウェアの例、完全版の資料、テクニカル・サポート、コミュニティ・フォーラムが含まれています。これらの統合されたツールと機能を組み合わせて使用することにより、すべてのスキル・レベルの IoT 開発者の組み込み開発がシンプルになり、生産性が高まります。Simplicity Studio は、開発者がプロジェクトを数分で軌道に乗せらせるように、接続されている 8 ビットまたは 32 ビットの MCU またはワイヤレス SoC を自動的に検出し、デバイスをグラフィカルに設定し、サポートされている設定オプションを表示するインテリジェンスを内蔵しています。

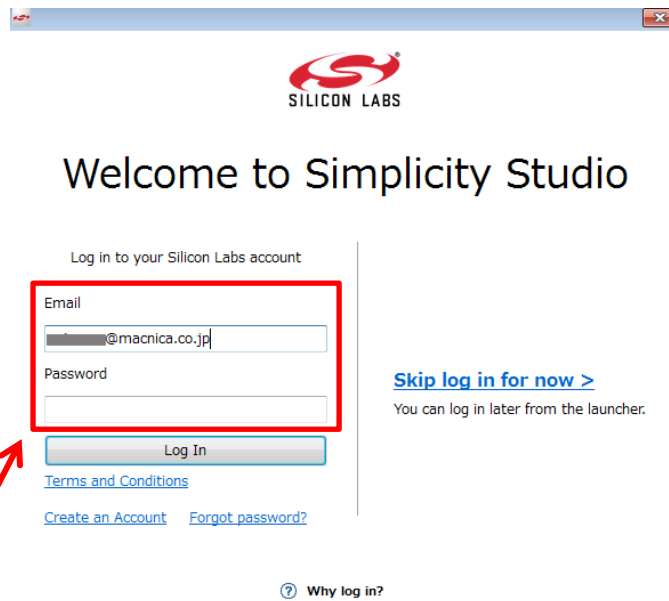


- ② ダウンロード完了後 ”install-studio-v4\_xx.exe” を起動し、インストールを開始してください。License Agreement → インストールフォルダの指定 (Choose Destination Location) → インストール実行の手順で進んでいきます。インストールフォルダを指定する際には、全角文字 (2 バイトコード) が入らない path を指定してください。使用時にエラーが出る場合があります。

インストールが進むと、ログイン画面が表示されます。シリコンラボ社のアカウント情報 (Email とパスワード) を入力し、Log In をクリックします。

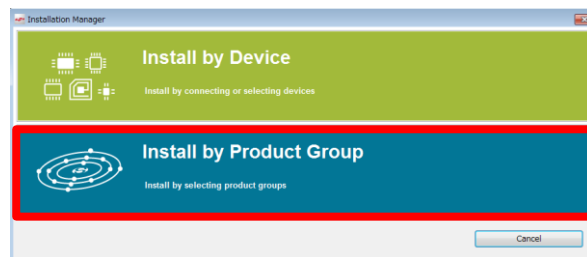
アカウントを持っていない場合には、Skip log in for now から先に進むことはできませんが、Bluetooth SDK をインストールすることができません。



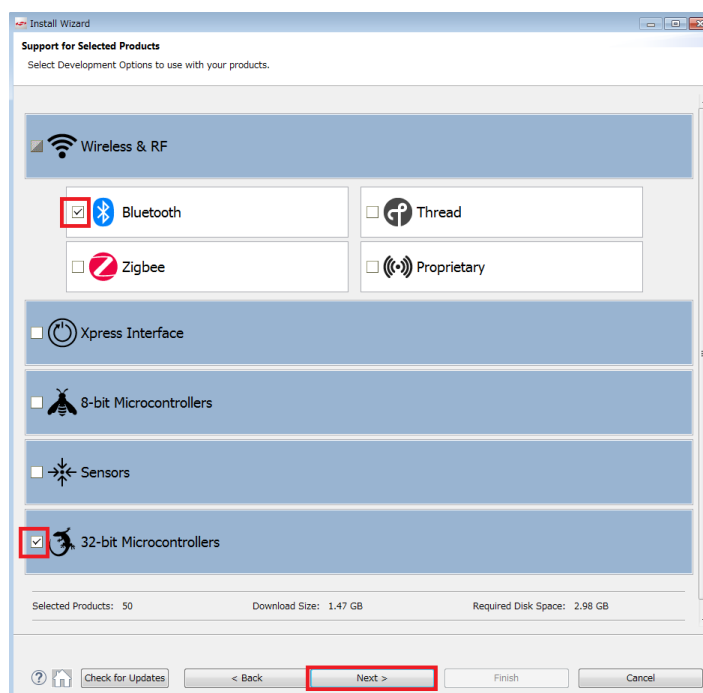


アカウント情報を入力

- ③ Installation Manager が起動しますので、Install by Product Group を選択します。



続いて Install Wizard が起動しますので、“Bluetooth”と“EFM32 32-bit MCU Products”にチェックを付け、Next をクリックします。



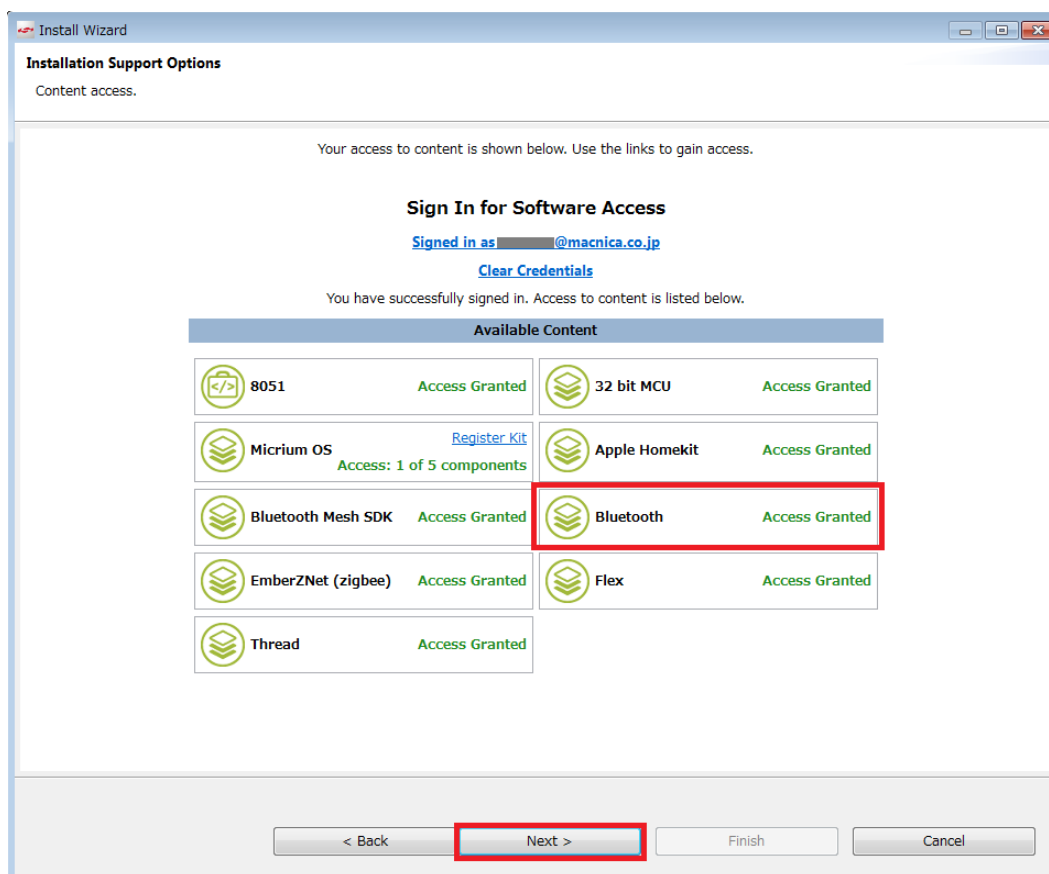
なぜ EFM32 32-bit MCU Products もインストールするの？

BGM1xx は Bluetooth モジュールですが、無線機能以外にも ADC などのアナログペリフェラル、I2C や SPI などのシリアルインタフェース、タイマなどの各種ペリフェラルを搭載しています。










BGM1xx の動作確認用に用意されているサンプルコードは、Bluetooth を使用したアプリケーションの実装例となっており、例えば「I2C の機能だけを確認する」といったシンプルなサンプルコードは用意されていません。

それに対して、EFM32 32-bit MCU ファミリでは、各ペリフェラルだけにフォーカスしたサンプルコードも多く用意されています。BGM1xx 内部では無線マイコン EFR32BGxx ファミリが使用されており、また EFR32BGxx ファミリは マイコン EFM32PGxx ファミリと機能互換(無線部は除く)になっていますので、EFM32PGxx 向けのサンプルコードや情報を、BGM1xx でも活用頂けます。

- ④ インストール可能なコンテンツが表示されます。Bluetooth が“Access Granted”と表示されていることを確認して、Next をクリックします。

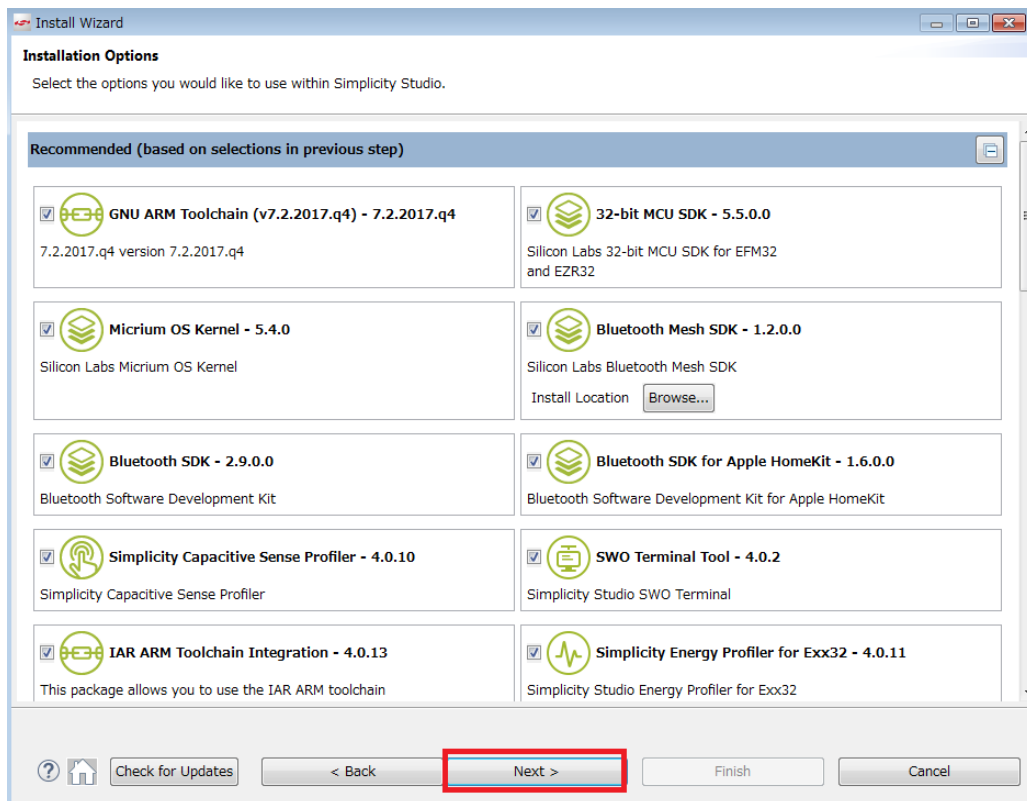


なお、ログインを行っていないと、Bluetooth のコンテンツがインストール不可となっています。

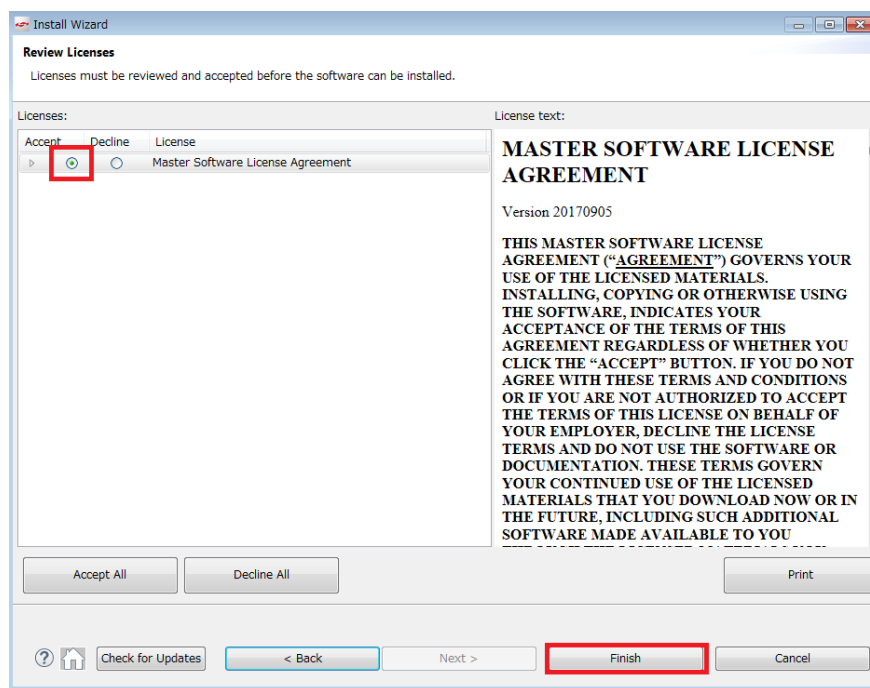
Available Content	
 <b>8051</b> <span style="color: green;">Access Granted</span>	 <b>32 bit MCU</b> <span style="color: green;">Access Granted</span>
 Micrium OS <a href="#">Sign In</a>	 Apple Homekit <a href="#">Sign In</a>
 Bluetooth Mesh SDK <a href="#">Sign In</a>	 Bluetooth <a href="#">Sign In</a>
 EmberZNet (zigbee) <a href="#">Sign In</a>	 Flex <a href="#">Sign In</a>
 Thread <a href="#">Sign In</a>	

⑤ インストールを行うコンテンツがリストアップされます。Recommended でリストアップされているコンテンツは取捨ができますので、不要なものを外すことでインストールに必要な時間と容量を削減することができます。

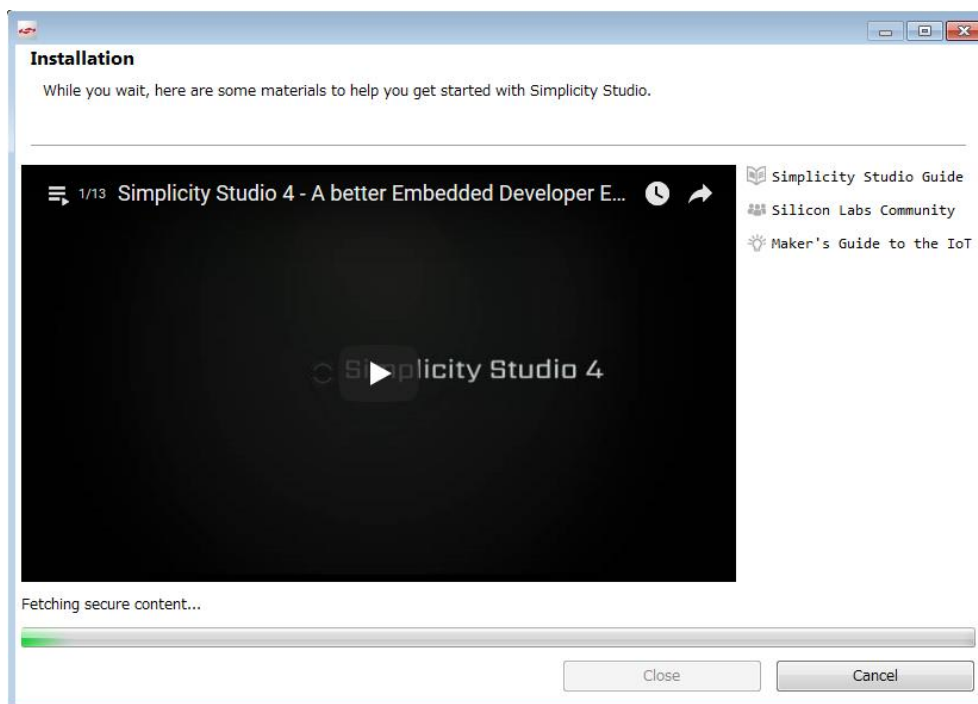
選択が終わったら、Next をクリックします。



- ⑥ Review License でライセンス内容を確認し、Accept にチェックを付け、Finish をクリックします。



コンテンツのインストールが始まります。インストール後に再起動したら、セットアップは完了です。



## 5-2 インストールがうまくいかない場合

### 5-2-1 シリコンラボ社アカウントの取得方法

Bluetooth SDK の入手には、シリコンラボ社 WEB サイトのアカウントが必要になります。お持ちでない場合には、下記の手順でご入手ください。アカウントの作成は無料です。

- ① 下記 URL にアクセスし、右上の Register からアカウント作成に進んでください。

<https://www.silabs.com/>



[简体中文](#) [繁體中文](#) [日本語](#)

[Log In](#) | [Register](#)

[Parametric Search](#) | [Cross-Reference Search](#)

Search silabs.com

GO

[About](#) ▾ [Products](#) ▾ [Solutions](#) ▾ [Community & Support](#) ▾

- ② 必要事項を入力し、Create an Account でアカウントを作成してください。

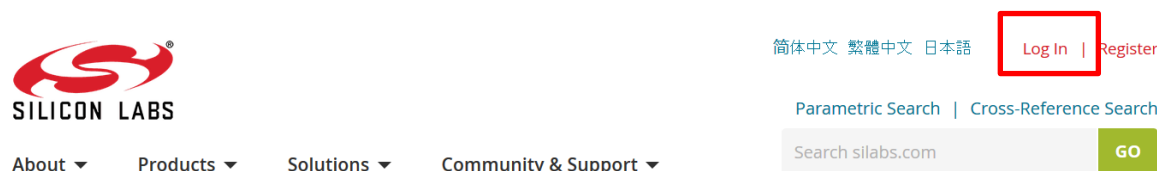
The screenshot shows the 'Create An Account' form on the Silicon Labs website. The form includes the following fields and labels:

- First Name: Taro (名前)
- Last Name/Family Name: Yamada (苗字)
- Company Name: Macnica (会社名)
- Email: xxxxxx@xxxxx.co.jp (メールアドレス)
- Password: [Redacted] (パスワード)
- Confirm Password: [Redacted] (パスワード(再入力))
- Country: Japan (国名)
- State: Kanagawa (県名)
- Zip Code: 2228561 (郵便番号)

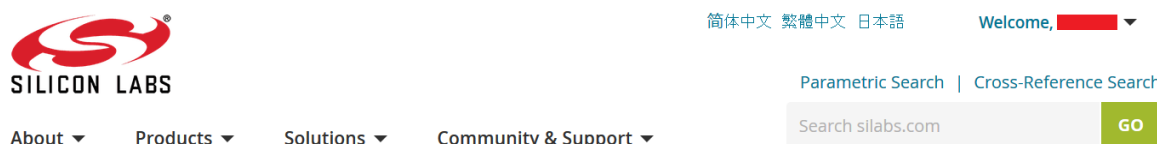
At the bottom of the form, there is a checkbox for "I would like to receive email communications from Silicon Labs" which is checked. Below the checkbox is a red-bordered button labeled "Create an Account". At the very bottom, there is a link for "Log in" for existing users.

③ アカウントが生成できたら、念のため発行されたアカウントでログインできることを確認してください。下記 URL にアクセスし、右上の Log In からログインを行ってください。

<https://www.silabs.com/>



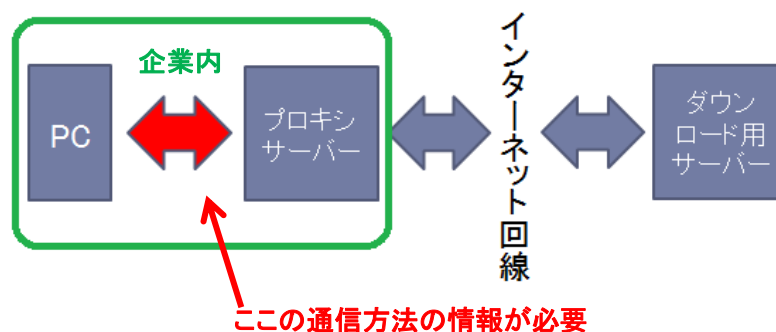
④ ログインに成功すると、画面右上に「Welcome, 名前」が表示されます。



## 5-2-2 企業プロキシサーバーを介して接続している場合

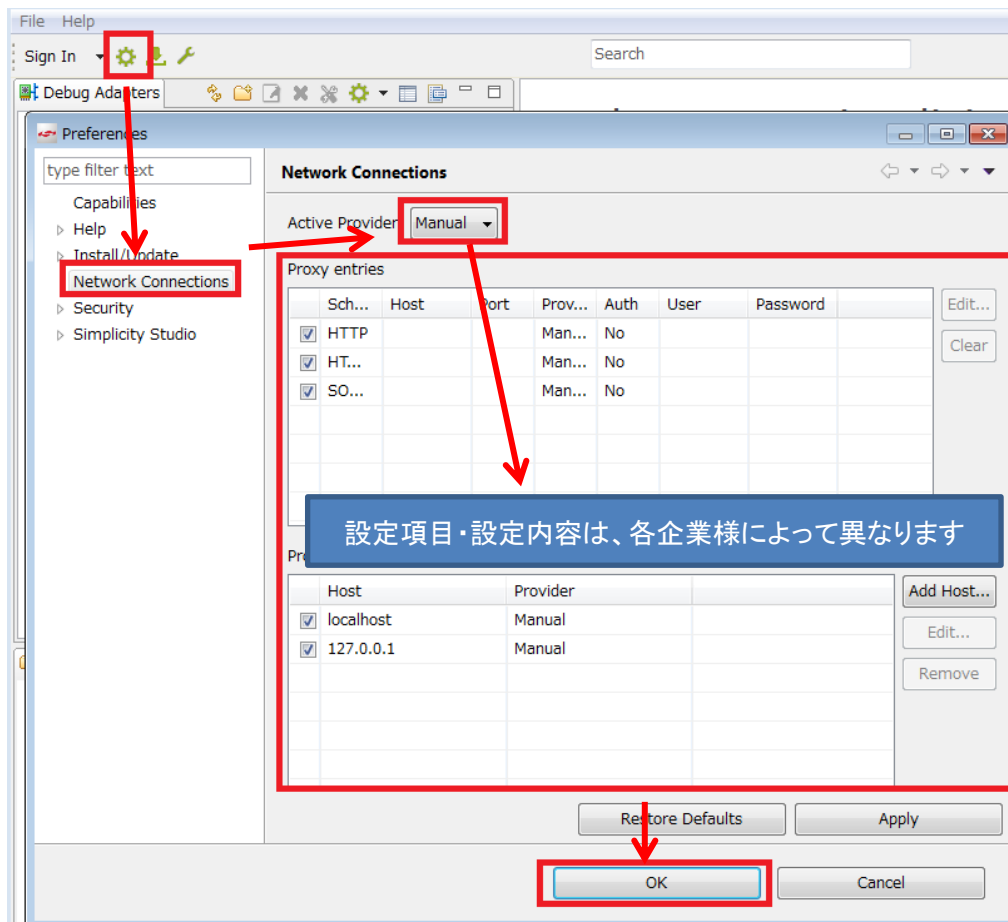
インストールにはインターネット接続が必要になりますが、プロキシサーバーを導入している企業ユーザ様の場合にはプロキシ設定が必要になる場合があります。設定内容については、自社のネットワーク管理者にご相談下さい。プロキシを介さずにインターネット回線に接続できる環境が構築できる場合には、そちらをご利用頂くのが簡単です。(WiFi ルーターや自宅など)

Simplicity Studio がアクセスする先については、シリコンラボ社のコミュニティフォーラムに関連情報があります。(リンク) 企業プロキシサーバーのセキュリティオプション(ホワイトリスト)で回避するような場合にご利用ください。

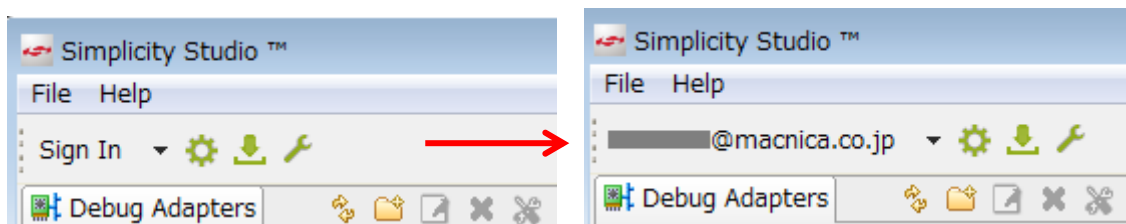


プロキシサーバーの設定は、以下の手順で行います。

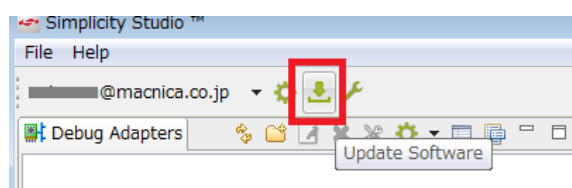
Simplicity Studio の Settings アイコンを選択し、Network Connections を選択します。プロキシ設定の画面が表示されますので、Active Provider を Manual に設定変更し、Proxy entries に必要な設定を入力してください。



設定が終わったらログイン(Sign In)を行います。画面左上の Sign In をクリックし、シリコンラボ社 WEB サイトのアカウントを入力します。ログインに成功すると、画面左上にメールアドレスが表示されます。



ログインに成功したら、Update Software アイコンをクリックし、Install Manager からインストールが継続できます。



**設定例： PC とプロキシサーバー間の通信に HTTP のみを使用している場合**

Active Provider:

Proxy entries

	Schema	Host	Port	Provider	Auth	User	Password	
<input checked="" type="checkbox"/>	HTTP	██████████	██████	Manual	No			<input type="button" value="Edit..."/>
<input checked="" type="checkbox"/>	HTTPS			Manual	No			<input type="button" value="Clear"/>
<input checked="" type="checkbox"/>	SOCKS			Manual	No			
<input type="checkbox"/>	HTTP	Dynamic	Dynamic	Native	No			

Proxy bypass

	Host	Provider	
<input checked="" type="checkbox"/>	localhost	Manual	<input type="button" value="Add Host..."/>
<input checked="" type="checkbox"/>	127.0.0.1	Manual	<input type="button" value="Edit..."/>
			<input type="button" value="Remove"/>

**5-2-3 プロキシ設定をしてもインストールがうまくいかない場合**

強固なセキュリティを施している企業様の場合には、適当なプロキシ設定を行ったとしても、サインインやインストールが阻害される場合があります。(WEB ではサインインできるが、Simplicity Studio ではサインインできない、などの症状が出ます)

その場合には、Simplicity Studio がアクセスする下記アドレスを、プロキシサーバーのホワイトリストに追加して頂くことで、サインインやインストールが可能になると考えられます。

- <https://developer.silabs.com>
- <https://devtools.silabs.com>
- <https://siliconlabs.force.com>
- <https://gecko-resources.silabs.com>

もし、ポートを指定してのホワイトリスト追加を行う場合には、以下の接続先・ポートをご使用ください。

- <https://developer.silabs.com> (port 443)
- <https://siliconlabs.force.com> (port 443)

Silicon Labs 社の WEB サイトでも情報公開されております。( [リンク](#) )



#### 5-2-4 オフライン・インストーラ

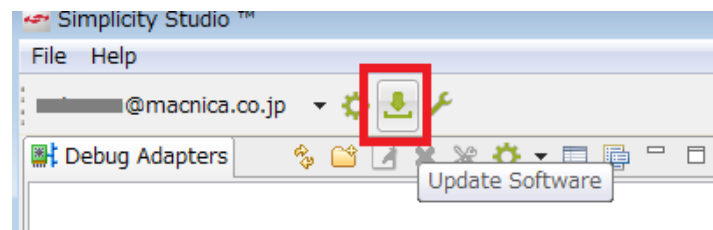
オンラインでインストールすることが望ましいですが、どうしてもプロキシの設定がうまくいかない場合には、オフライン・インストーラも活用頂けます。

<マクニカオンラインサービス FAQ>

- [Simplicity Studio のオフライン・インストーラはありますか？](#)

#### 5-2-5 Install Manager／Install Wizard の画面を閉じてしまいました

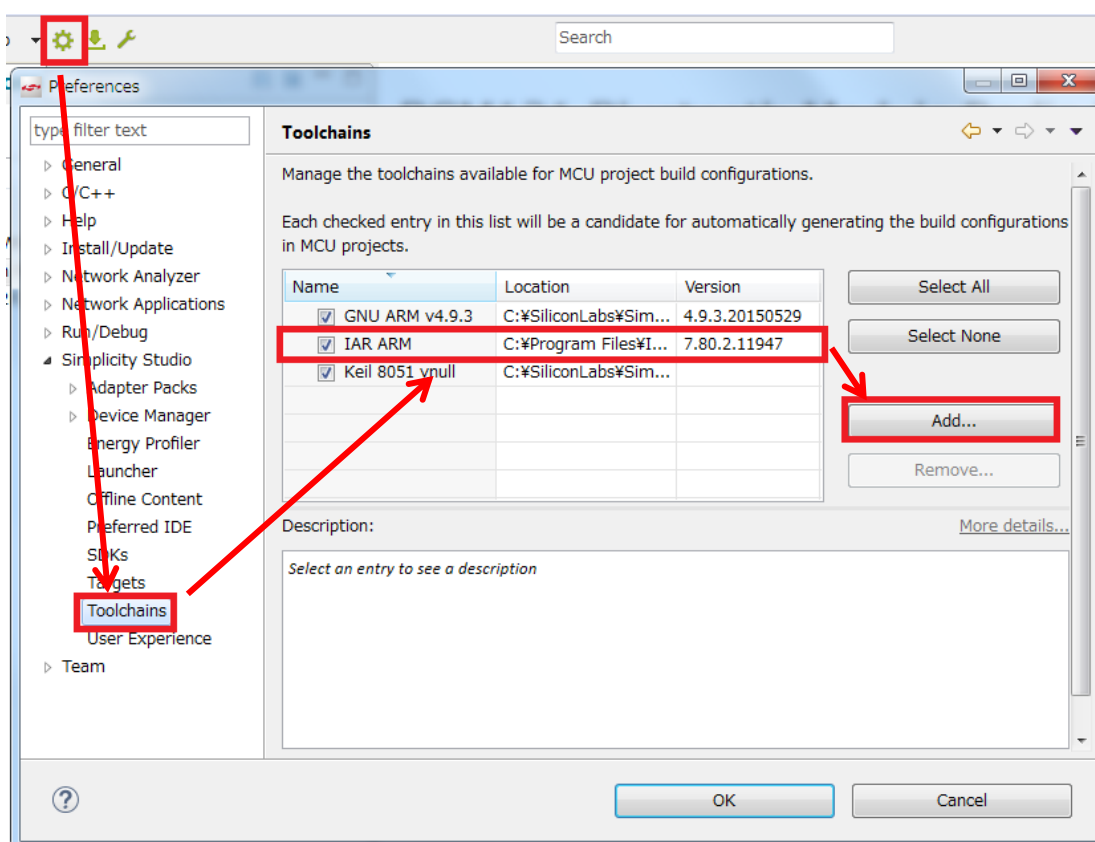
画面左上の Update Software アイコンをクリックすると、Install Manager を起動することができます。



### 5-3 IAR コンパイラのインストール（オプション）

BGM1xx を C 言語設計する場合には、C コンパイラが必要になります。Bluetooth SDK 2.4.0 以降の SDK では、Simplicity Studio に標準インストールされる GCC（無償コンパイラ）をご使用頂くことができます。有償になりますが、GCC に比べてコード効率の良い 3rd party 製コンパイラ（IAR システムズ社）もご使用になれます。

IAR コンパイラのインストールが完了すると、Simplicity Studio は Toolchain として自動認識します。念のため、歯車アイコン（Preference）→Simplicity Studio→Toolchains で IAR コンパイラが認識されていることを確認してください。もし自動認識されていないようであれば、Add ボタンから追加登録を行うことができます。



Bluetooth SDK では、ご使用頂ける IAR バージョンを指定させて頂いております。

- SDK 2.0~2.8.2 → IAR v.7.80.2
- SDK 2.9.0~2.9.2 → IAR v.7.80.4
- SDK 2.10.0~ → IAR v.8.30.1

#### <マクニカオンラインサービス FAQ>

- [BGM1xx の C 言語設計で IAR コンパイラを使用しますが、バージョンの指定はありますか？](#)
- [IAR EWARM v.7.80.2 の入手先を教えてください](#)

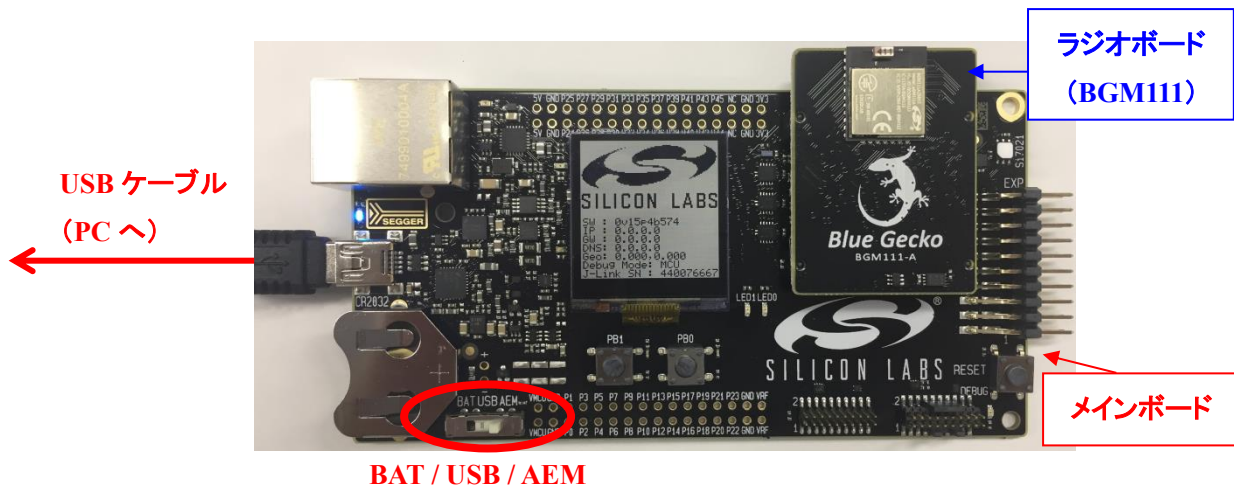
## 6 ハードウェア・セットアップ

BGM1xx の評価に必要なハードウェアの設定を行います。

### 6-1 Wireless Starter Kit のセットアップ

以下の手順で設定していきます。

1. メインボードにラジオボードを装着します。ラジオボードの向きは下図を参照ください。
2. BAT, USB, AEM の中から、基板に給電する方法を選びます。スイッチを **AEM** に切り替えます。
3. 基板左の USB コネクタと PC を USB ケーブルで接続します



なお、給電スイッチは、

- BAT: 電池からの給電（電池では供給電流が足りず、動作しない場合が多々あります）
- USB: 未使用
- AEM: PC から USB 経由での給電

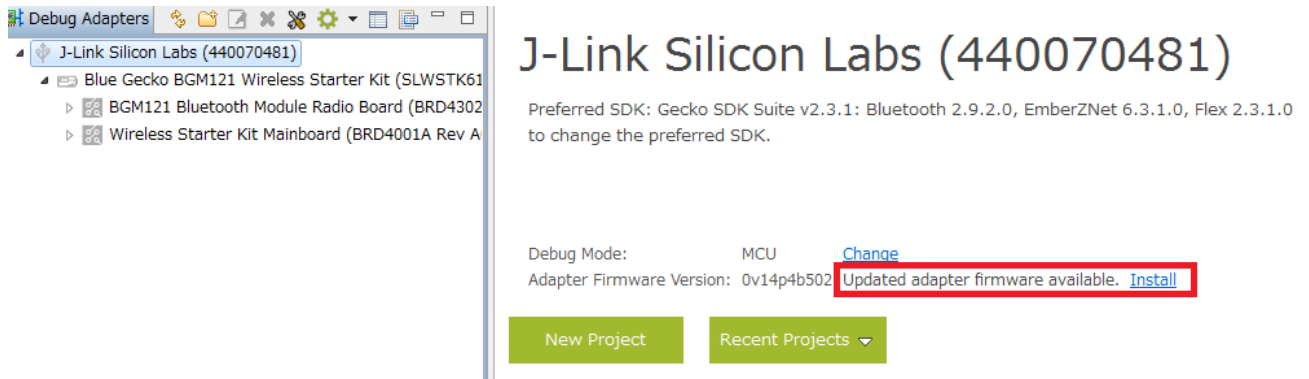
となっています。

## 6-2 Wireless Starter Kit の制御ファームウェアの更新

Simplicity Studio を起動し、Wireless Starter Kit を接続します。

画面左上の Debug Adapters ウィンドウに J-Link が表示されますので、それをクリックすると、Debug Mode(7-4-3 で紹介)および Adapter Firmware version が表示されます。

Adapter Firmware version の横に、Install という表示があった場合は、Wireless Starter Kit 上の制御ファームウェアが最新状態ではありません。Install をクリックして最新状態にしてください。

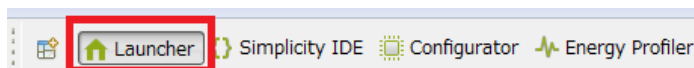


更新が完了すると、表示が Change に変わります。これが最新状態です。



## 7 使用方法

サンプルコードを評価キットと Simplicity Studio を使用した評価手順をご紹介します。ここでは BGM121 を使用しておりますが、他のモジュールでも手順は同じです。なお、各ツールから Simplicity Studio のトップ画面に戻るには、画面右上の Launcher アイコンを使用します。



### 7-1 サンプルコードを動かしてみる前に(ブートローダーの更新)

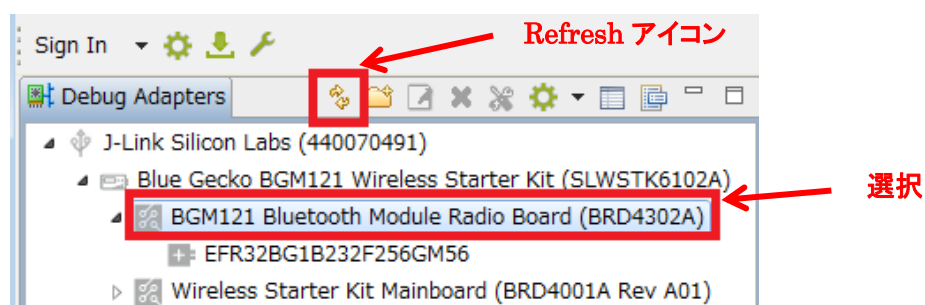
Bluetooth SDK 2.7.0 以降のサンプルコードは、Gecko Bootloader(新しいブートローダー)上での動作を前提としています。しかし、購入したてのラジオボードや、Bluetooth SDK 2.6.2 以前の SDK で使用していたラジオボードには Legacy Bootloader(古いブートローダー)が書き込まれています。そのため、初めて Bluetooth SDK 2.7.0 以降のサンプルコードを使用する場合には、ブートローダーの更新を行う必要があります。詳しくは下記を参照ください。


<マクニカオンラインサービス FAQ>

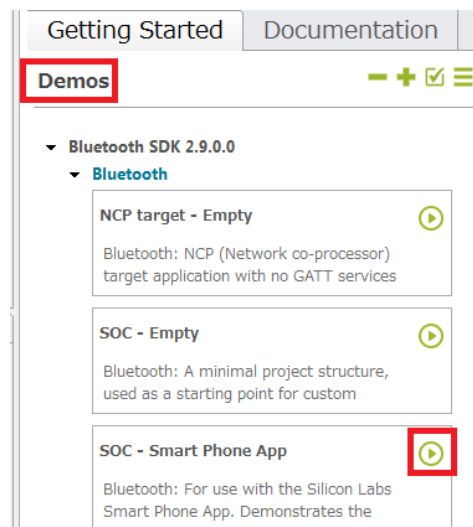
- [Bluetooth Smart SDK 2.7.x のサンプルコードをダウンロードしましたが正常動作しません。対処方法を教えてください](#)

更新手順は以下の通りです。

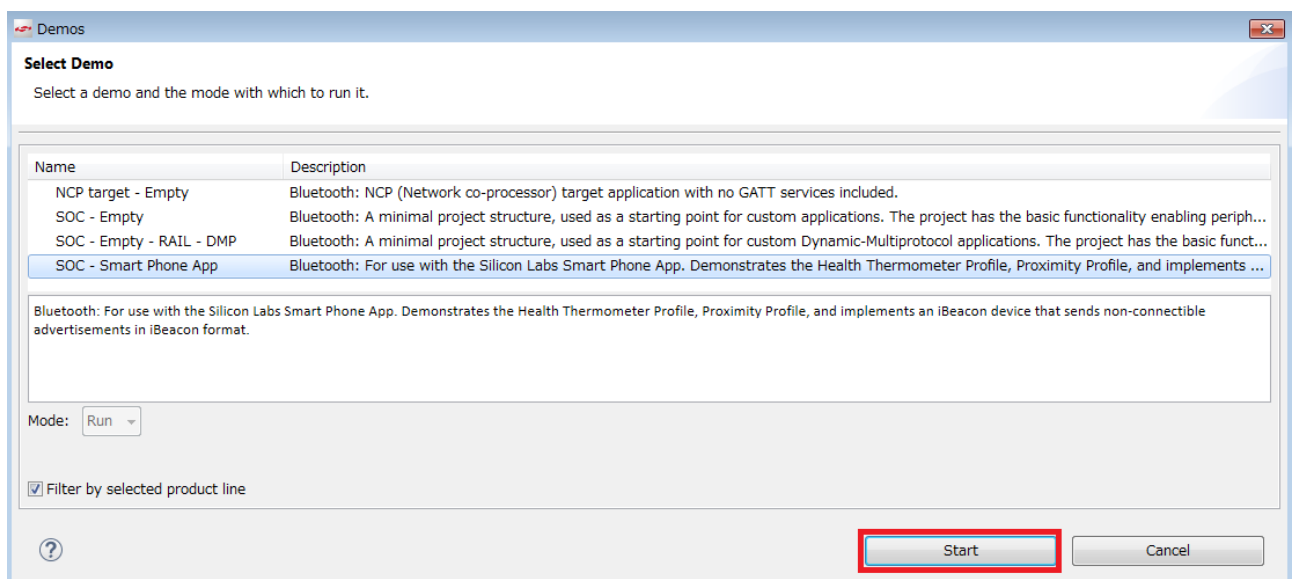
Wireless Starter Kit を PC に接続すると、Simplicity Studio が Wireless Starter Kit とラジオボードを自動認識します。Device タブに表示されたラジオボードを選択してください。うまく認識してくれない場合には、Refresh アイコンを押してみてください。



次に、Demos の“SOC -Smart Phone App” (SOC -xxx であればどれでも良いです)の右横の  をクリックします。



Demos というウィンドウが開きますので、そのまま Start ボタンをクリックすると、ラジオボードへダウンロードされます。



この作業は一度行えば良いですが、Bluetooth SDK 2.6.2 以前のコードをダウンロードすると、古いブートローダーに書き戻されてしまう場合がありますので、その際には再度ブートローダーの更新を行ってください。

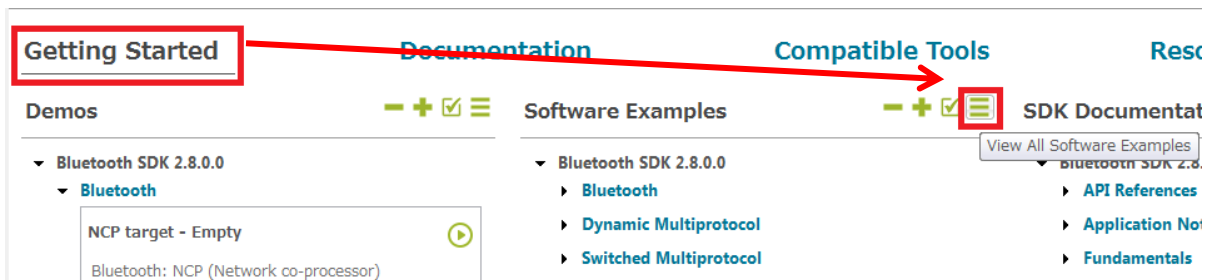
**注意** EFR32BG1 ベースのモジュール(BGM111, 113, 121, 123, 11S)では、Flash メモリをイレースするとブートローダーもイレースされます。Flash メモリをイレースした場合や、Debug Unlock (Flash メモリをイレースします)をした場合には、再度ブートローダーを書いてください。

## 7-2 サンプルコードを動かしてみる(C言語編)

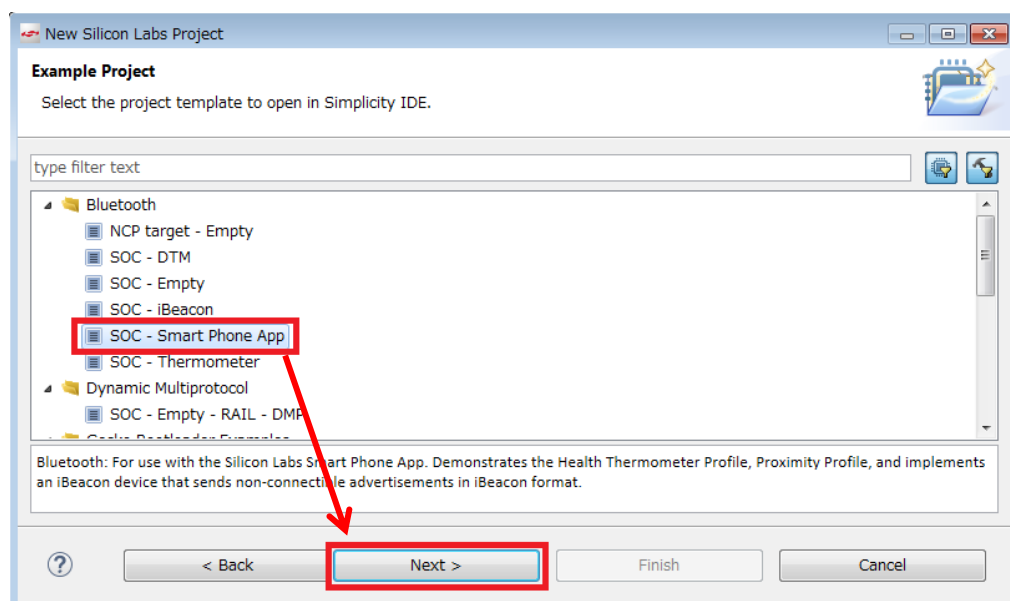
BGM1xx にサンプルコードをダウンロードして、スマホアプリと接続するところまで行ってみましょう。

PC に Wireless Starter Kit を接続し、Device タブでラジオボードを選択します。

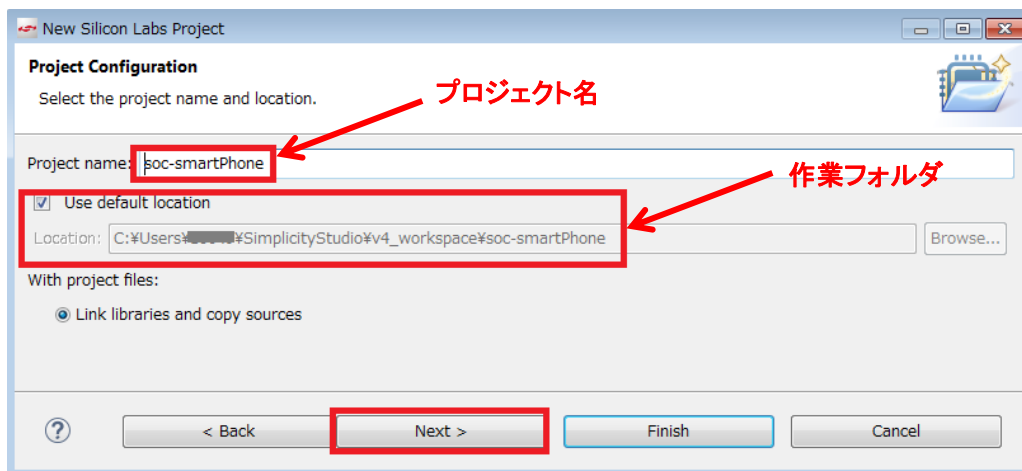
次に、Getting Started タブ ⇒ Software Examples 横の View All Software Examples を選択します。



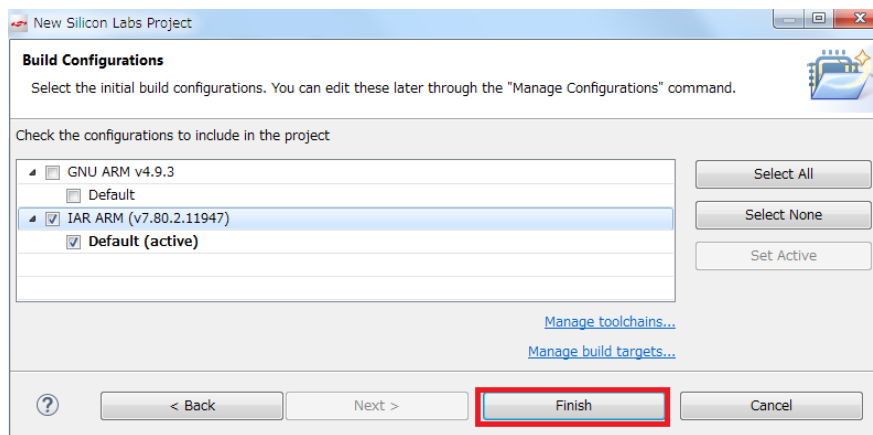
Example Project で SOC – Smart Phone App を選択し、Next をクリックします。



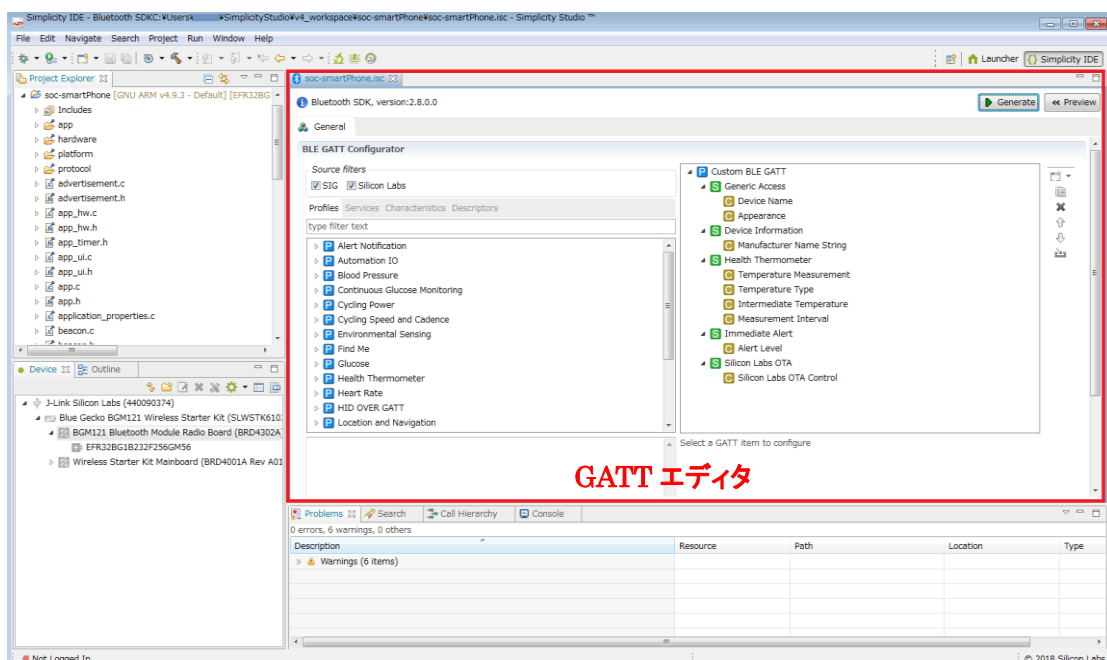
プロジェクト名を入力し、作業フォルダを指定します。With project files では、サンプルコードをローカルにコピーして使うかどうかを指定します。指定が終わったら、Next をクリックします。



使用するコンパイラを選択し、Finish をクリックします。

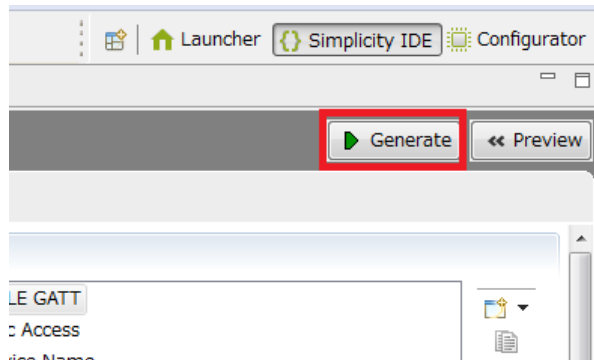


サンプルコードの準備が整うと、Simplicity IDE が起動します。画面右に表示されているのが GATT エディタで、Profiles/Services/Characteristics/Descriptors を設定することができます。

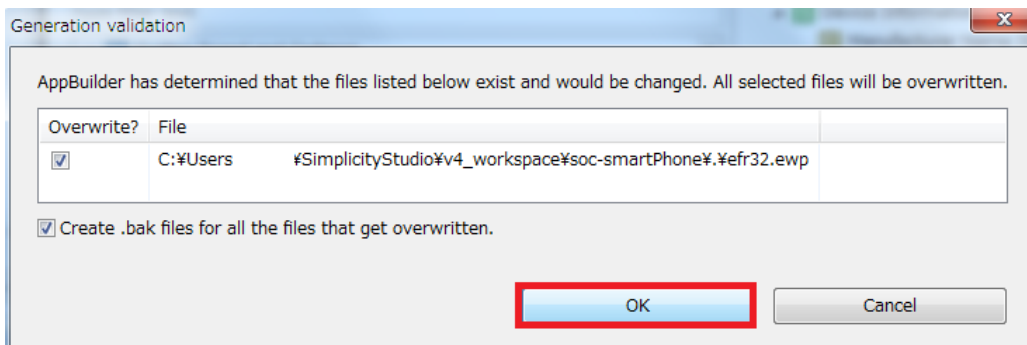




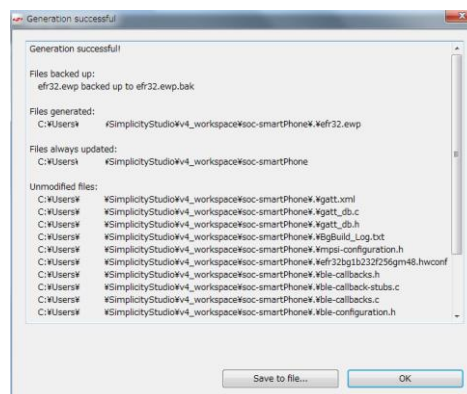
GATT の設定をソースコードに反映するために、Generate をクリックします。



ファイルの上書きが生じる場合には確認が行われます。上書きしたくないものがあればチェックを外してください。ここではそのまま OK をクリックします。

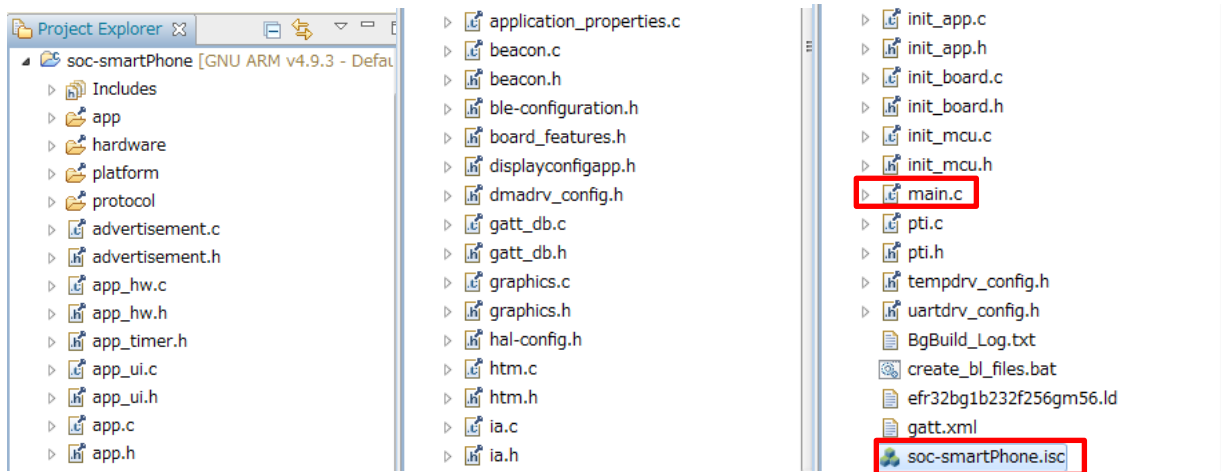


ファイルが生成されます。

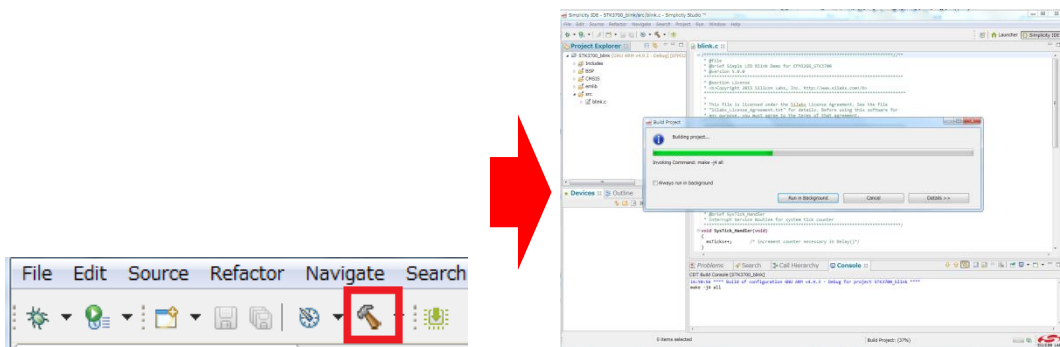


生成されたプロジェクトツリーは以下のようになっています。main.c がプログラム本体です。また、isc ファイルは GATT エディタのプロジェクトファイルです。なお、Bluetooth SDK 2.6.2 以前で生成されていた Hardware Configurator 用のプロジェクトファイルは生成されなくなっています。

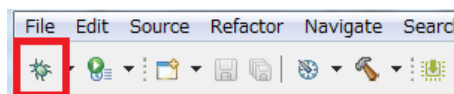
各ファイルの役割などについては、「UG136: Silicon Labs Bluetooth C Application Developer's Guide」(英語版, 日本語版)に記載がありますのでご参照ください。日本語版は英語版と比べてバージョンが古い場合もありますので、その際には英語版の記述を優先してください。



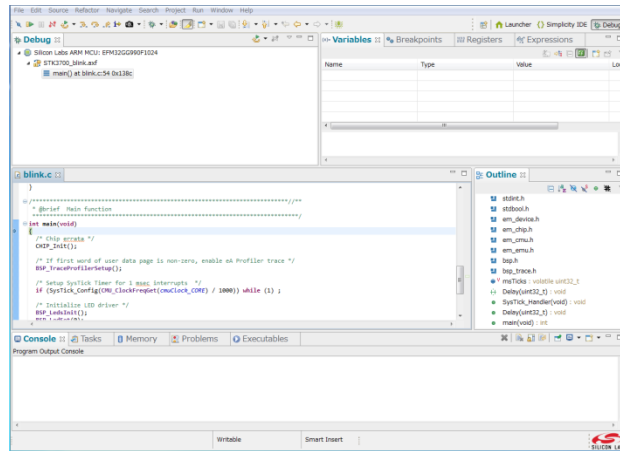
サンプルコードをビルドし、Starter Kit にダウンロードします。まずはトンカチのアイコン (Build) をクリックします。コンパイラが走り、サンプルコードがビルドされます。



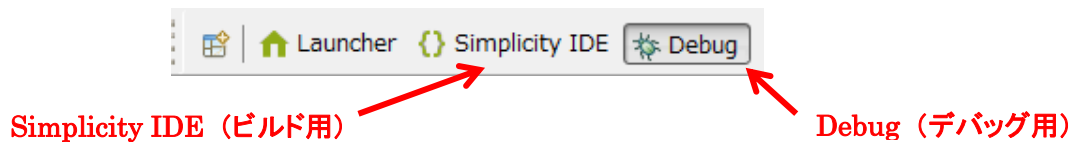
ビルドが完了したら、次に虫のアイコン (Debug) をクリックし、Starter Kit にダウンロードします。



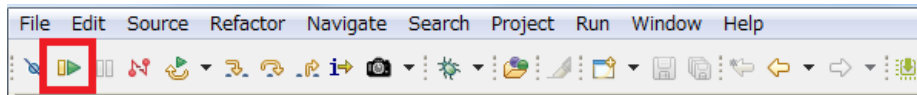
ダウンロードが完了すると、デバッグ用の画面に切り替わります。



なお、ビルド用の画面と、デバッグ用の画面の切り替えは、ウィンドウ右上のアイコンで行います。



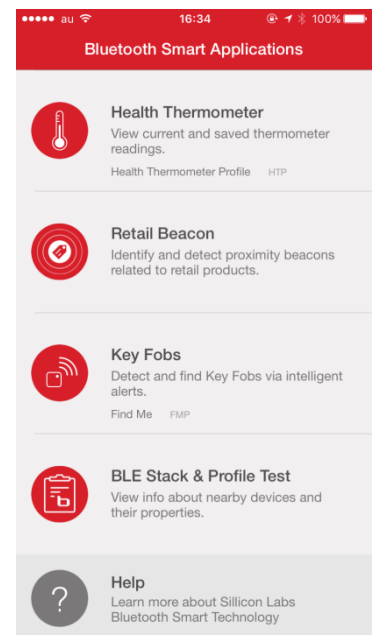
サンプルコードを実行します。下図の実行のアイコン (Resume) をクリックしてください。



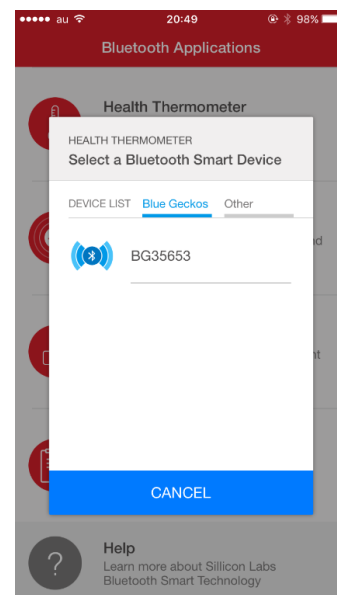
これで BGM121 側の準備は完了です。

次にスマホ側です。シリコンラボ社が提供するスマートフォン用のアプリ「Silicon Labs Blue Gecko WSTK App」を、スマートフォンにインストールしてください。

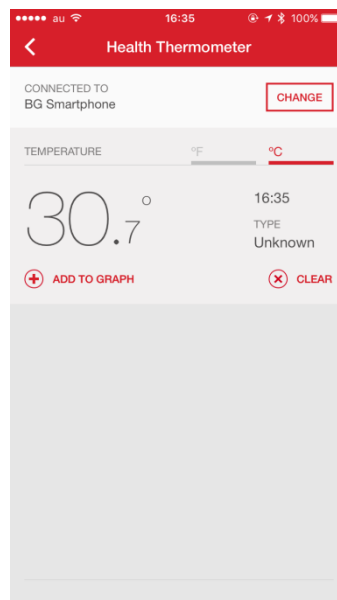
起動してみると、このような画面です。



Health Thermometer を選択してみると、BG Smartphone が見つかりました。これが BGM121 です。

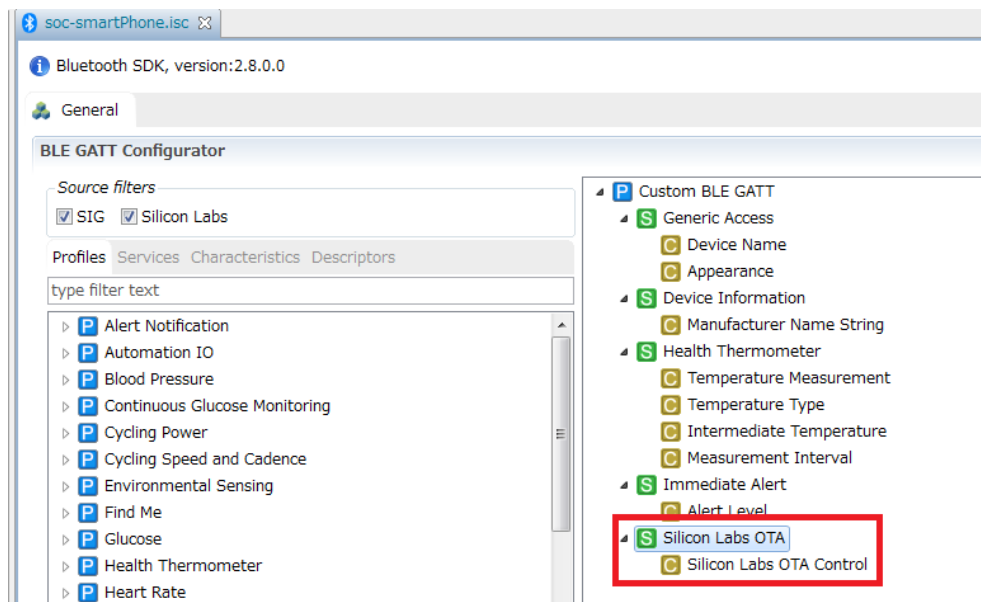


BG Smartphone を選択すると Connect します。アプリ上に温度情報が表示されました。これは Wireless Starter Kit 上の温度情報をスマホに送り、アプリで表示を行っています。



### 7-3 OTA update (over-the-air)を試してみる

サンプルコード SOC - Smart Phone App には、OTA update のサービスが実装されていますので、この機能を使ってアップデートを実践してみましょう。OTA update を使って、SOC - Smart Phone App を SOC - iBeacon に書き換える手順を紹介します。

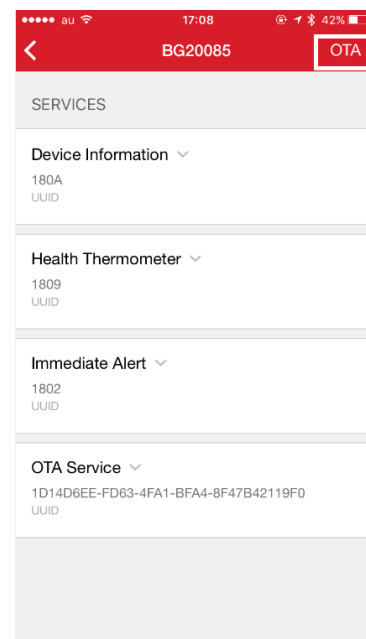


① Simplicity Studio で、SOC - iBeacon のプロジェクトを生成し、Build を実行します。手順は 7-2 を参照ください。

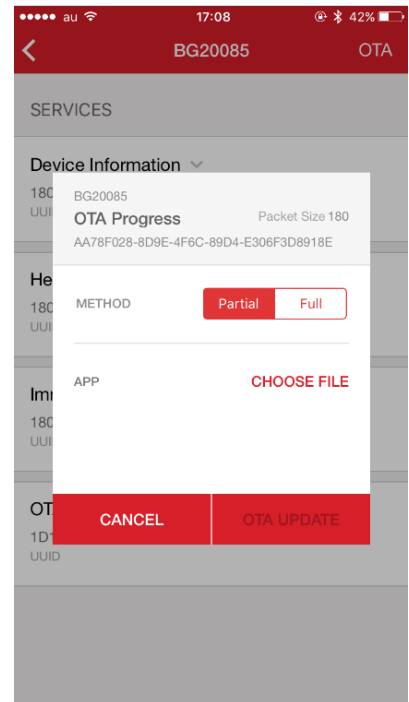
② ¥¥v4\_workspace¥¥soc-ibeacon に create\_bl\_files.bat が生成されるので、実行して OTA 用のバイナリを作成します。作成したバイナリ(application.gbl)は、¥¥v4\_workspace¥¥soc-ibeacon¥¥output\_gbl に格納されます。

③ スマートフォンからアクセスできるフォルダ(dropbox など)に、application.gbl をコピーします。

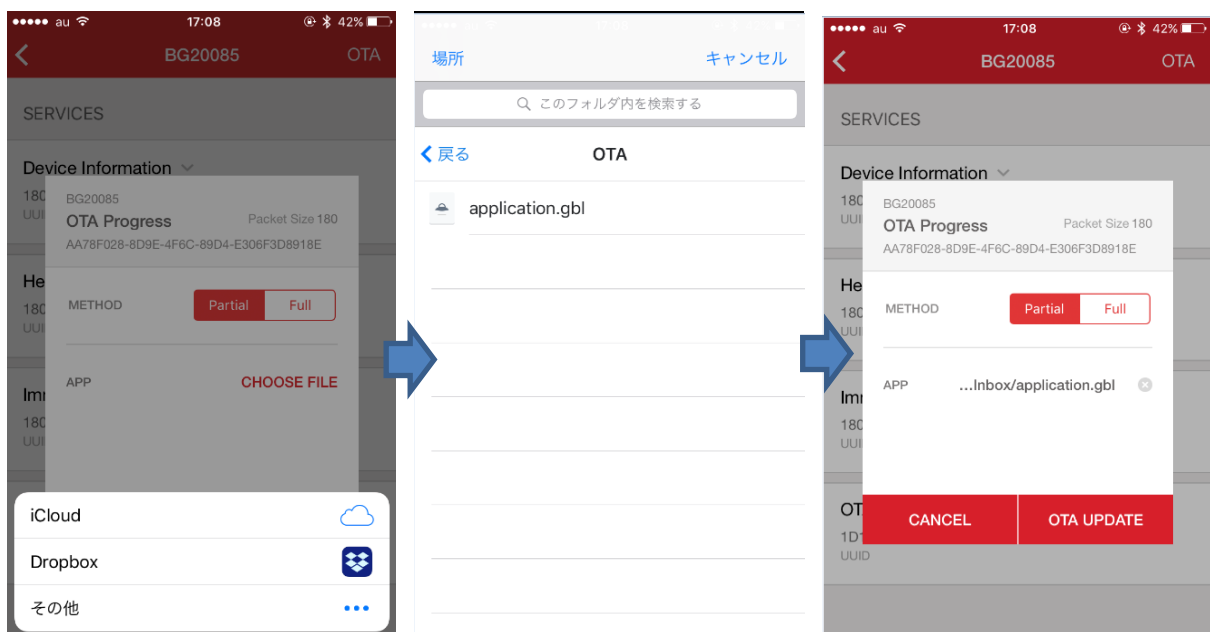
④ スマホアプリを起動して BGM121 に接続し、右上の OTA ボタンを押します。(右図)



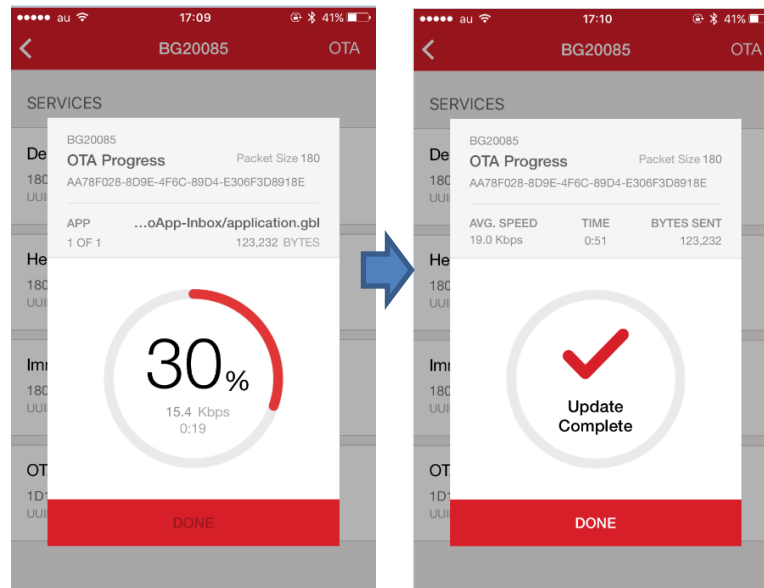
⑤ APP の CHOOSE FILE ボタンを押します。(右図)



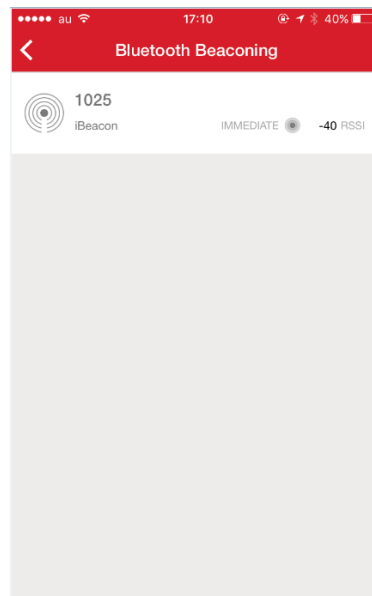
⑥ dropbox から、application.gbl を選択します。(下図)



⑦ OTA update を開始し(左下図)、完了する(右下図)。

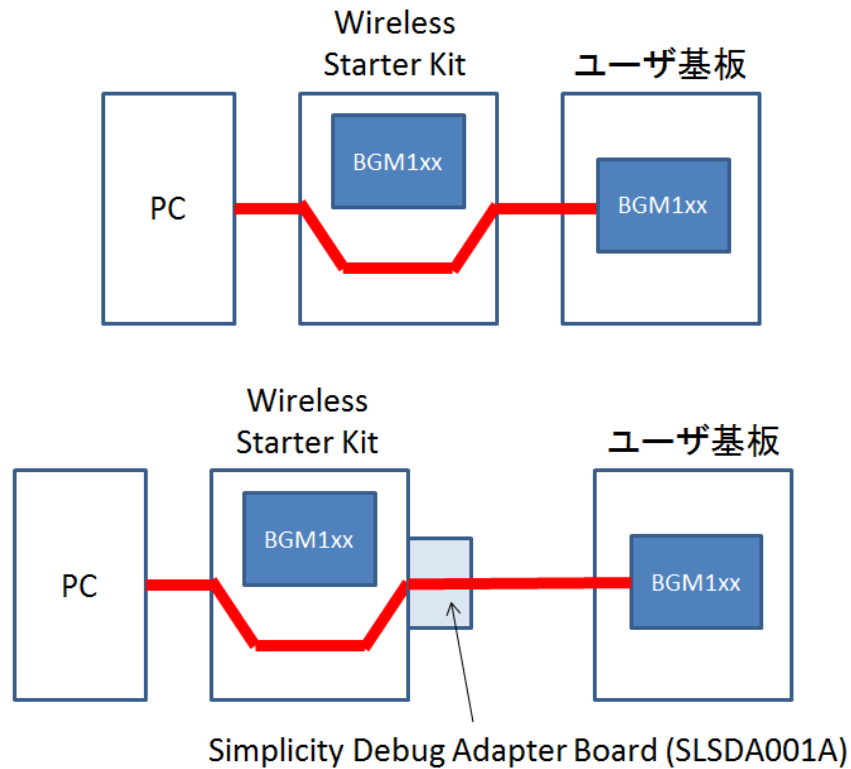


⑧ スマホアプリで、iBeacon として動作開始していることが確認できます。(下図)



## 7-4 ユーザ基板のプログラミング・デバッグを行ってみる

Wireless Starter Kit を使用することで、ユーザ基板上の BGM1xx に対して、プログラミング或いはデバッグを行うことが可能です。また、Simplicity Debug Adaptor Board (SLSDA001A) を使用すると、より簡単にユーザ基板と接続頂けます。



### 7-4-1 参考資料

- ・AN958: Debugging and Programming Interfaces for Custom Designs

<http://www.silabs.com/documents/public/application-notes/an958-mcu-stk-wstk-guide.pdf>

- ・Wireless Starter Kit ユーザガイド

BGM111: <http://www.silabs.com/documents/login/user-guides/ug122-brd4300a-user-guide.pdf>

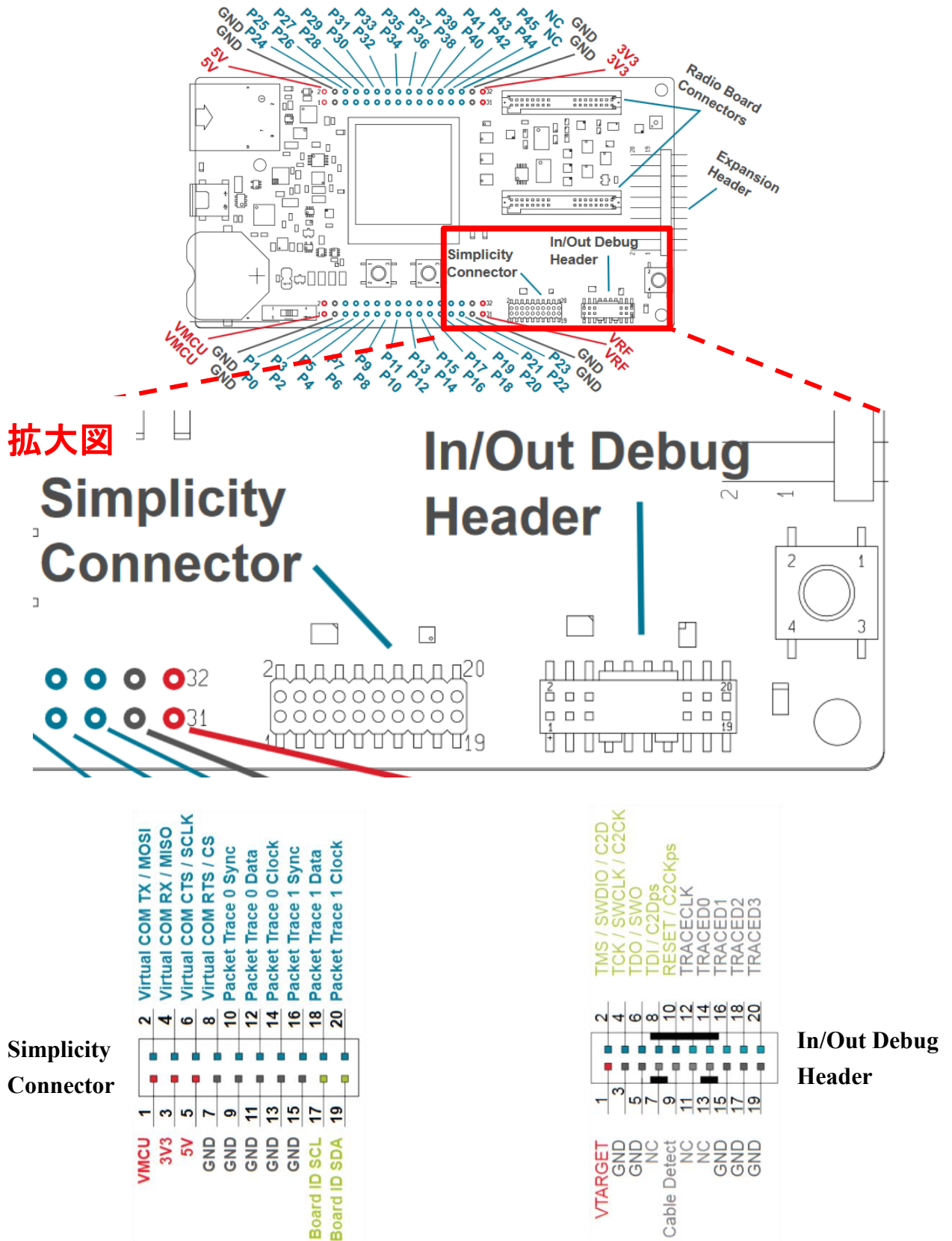
BGM113: <http://www.silabs.com/documents/login/user-guides/ug187-brd4301a-user-guide.pdf>

BGM12x: <http://www.silabs.com/documents/login/user-guides/ug234-brd4302a-user-guide.pdf>



## 7-4-2 ハードウェア接続

Wireless Starter Kit の右下にある In/Out Debug Header および Simplicity Connector を介して、ユーザ基板に接続します。下図は UG122 (BGM111 ユーザガイド) からの抜粋です。

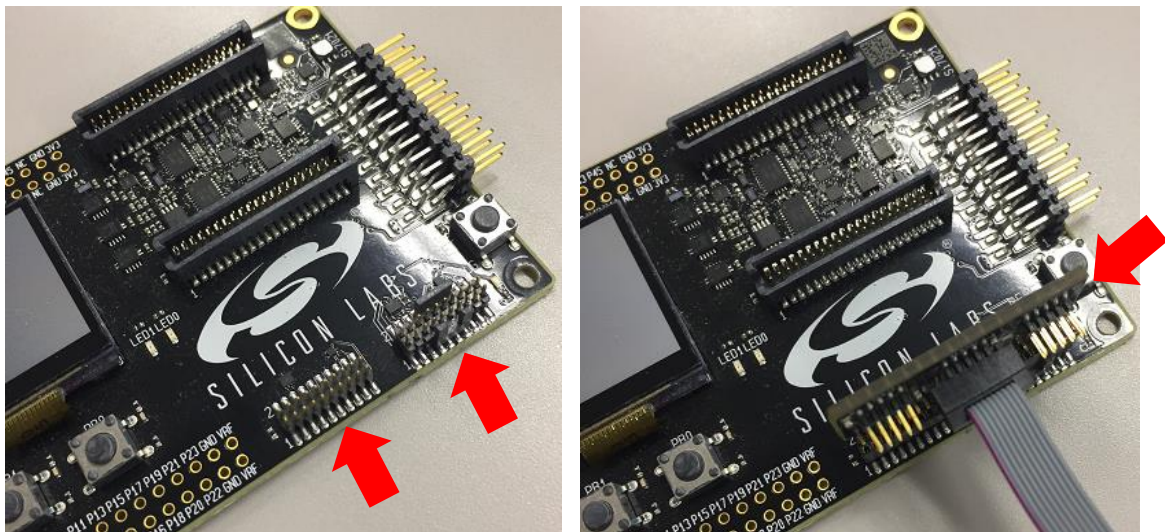


プログラミングについては、In/Out Debug Header にある SWCLK、SWDIO、RESET、VTARGET、GND の計 5 ピンを使用します。

SWCLK	BGM1xx(ユーザ基板上)の該当ピンに接続してください。直結で結構です。
SWDIO	BGM1xx(ユーザ基板上)の該当ピンに接続してください。直結で結構です。
RESET	BGM1xx(ユーザ基板上)の該当ピンに接続してください。直結で結構です。
VTARGET	BGM1xx(ユーザ基板上)への供給電源に接続してください。Wireless Starter Kitとユーザ基板の信号レベルを合わせるために使用します。接続し忘れると、Wireless Starter Kit からユーザ基板を認識できませんので、ご注意ください。
GND	Wireless Starter Kit の GND と、ユーザ基板の GND を接続してください。

BGTool 等を使って UART デバッグするには、Simplicity Connector にある Virtual COM を BGM1xx の UART に接続します。プログラミング用と UART デバッグ用とで、使用するコネクタが 2 つになりますので、その不便さを解消するために Simplicity Debug Adaptor Board (SLSDA001A) が用意されています。

2 つのコネクタに跨るように、Simplicity Debug Adaptor Board (SLSDA001A) を挿入します。



下図は AN958 からの抜粋ですが、このように 2 つのコネクタが 10 ピンに変換されます。

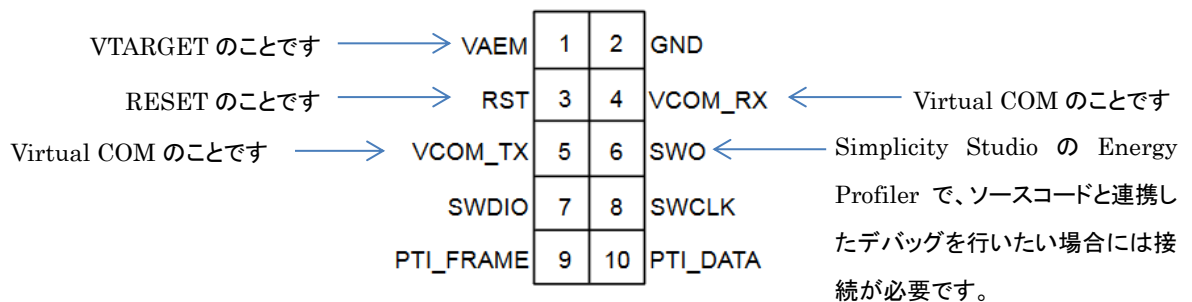
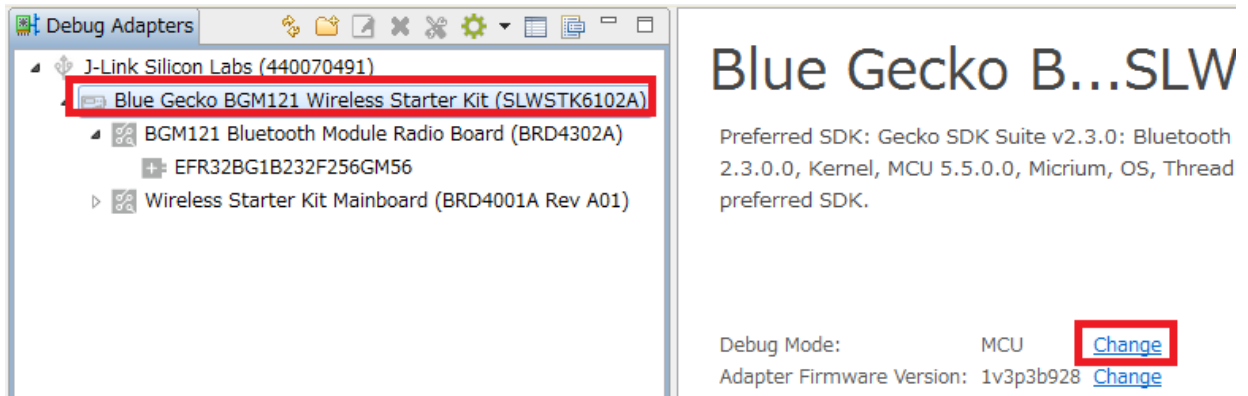


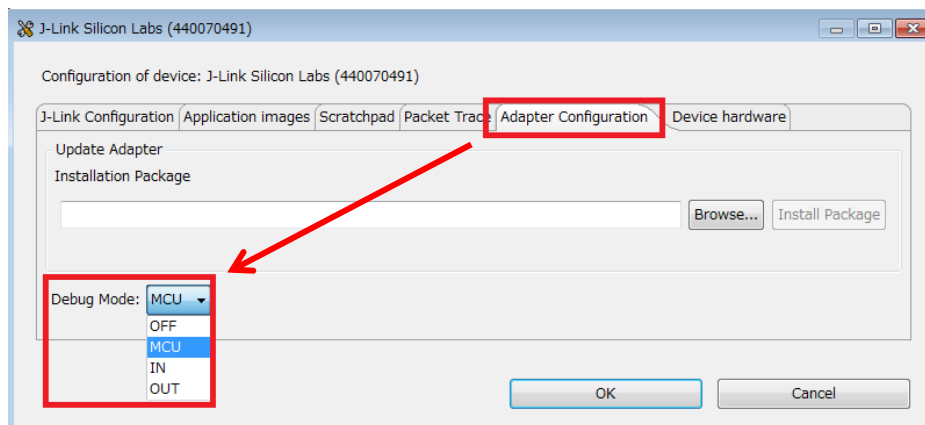
Figure 4.2. Mini Simplicity Connector Pin-Out

### 7-4-3 デバッグ対象の切り替え

デバッグ対象を、Wireless Starter Kit 上の BGM1xx から、ユーザ基板上の BGM1xx に切り替えます。Simplicity Studio の Device タブで Wireless Starter Kit を選択すると、画面右に現在の Debug Mode について表示されます。下図では MCU の設定になっています。



Adapter Configuration タブの Debug Mode で、OUT を選択します。



デバッグ対象が Wireless Starter Kit 外部に切り替わると、Wireless Starter Kit 右下の DEBUG OUT という LED が点灯します。

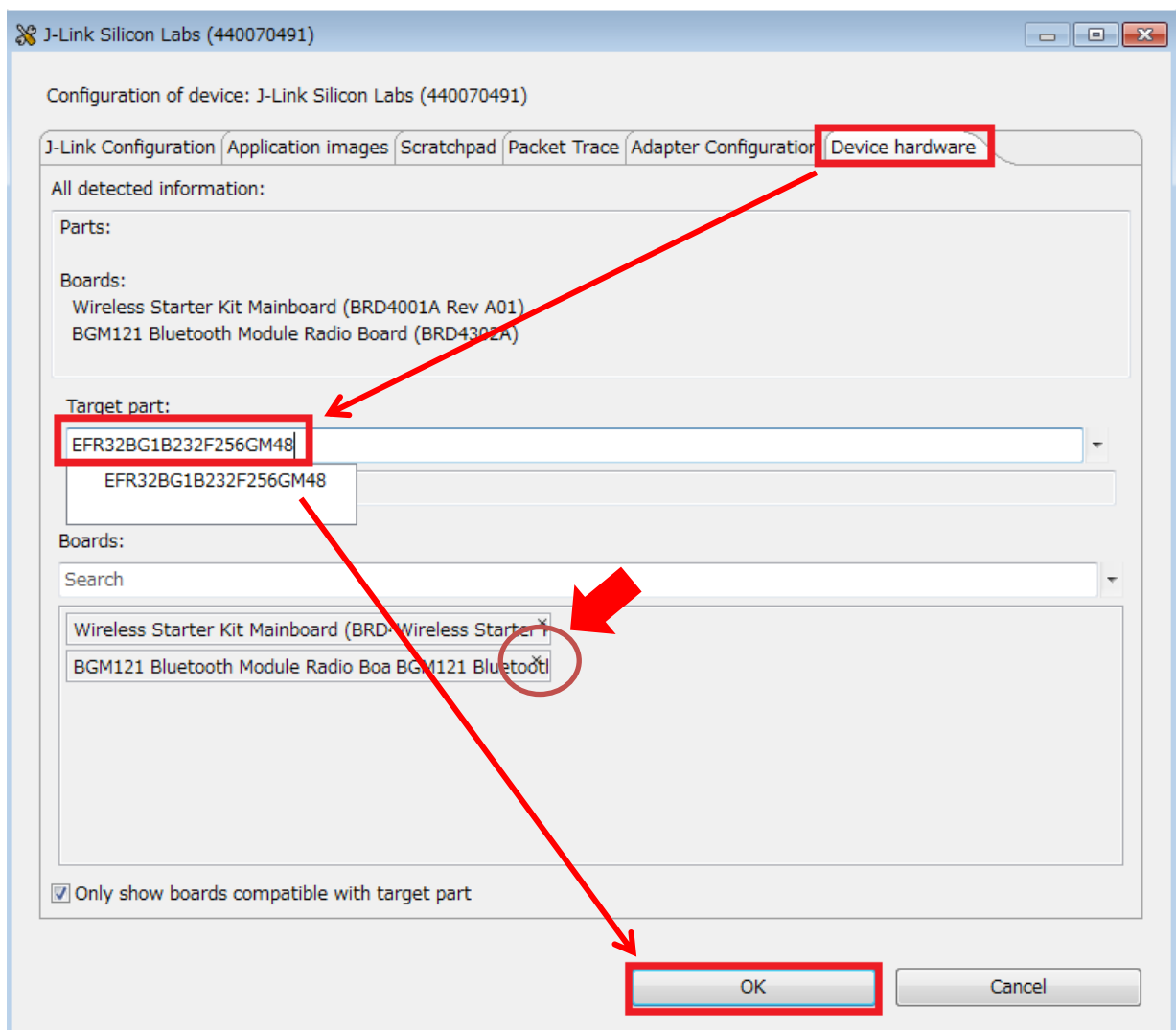


ユーザ基板上のデバイス(モジュール)型番は自動では認識されませんので、指定する必要があります。Device hardware タブの Boards で BGM1xx のラジオボードを選択するか、或いは Target part で SOC 型番を選択し、OK をクリックします。BGM1xx で使用している SOC 品番については、2-1「製品ラインナップ」にある無線チップ型番をご入力下さい。

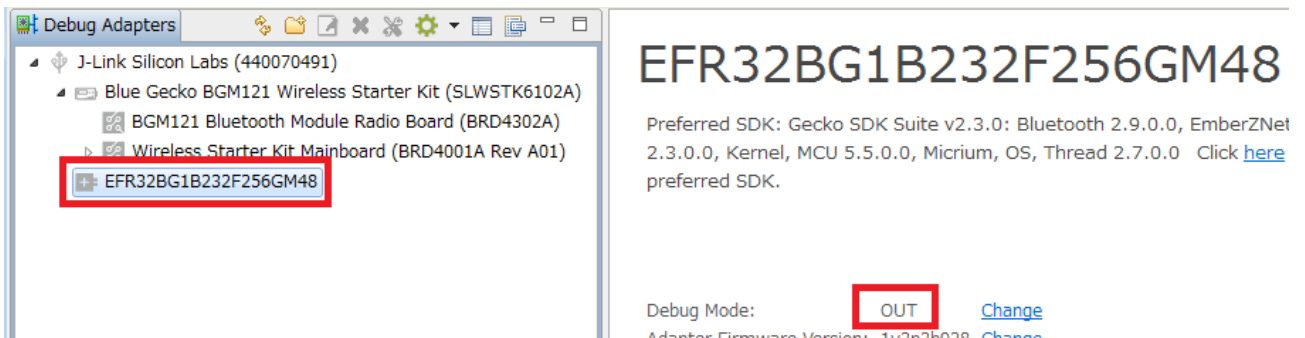
Boards に、使用しないデバイスやモジュールが載ったボードが登録されている場合には、× をクリックして情報を消去してください。

また、Boards で BGM1xx がリストアップされない場合には、Target part を None にしてからお試しください。

下図は BGM111 (で使われている EFR32BG1B232F256GM48) を追加するところです。Boards に BGM121 (違うモジュール) が表示されているので、情報消去する必要があります。



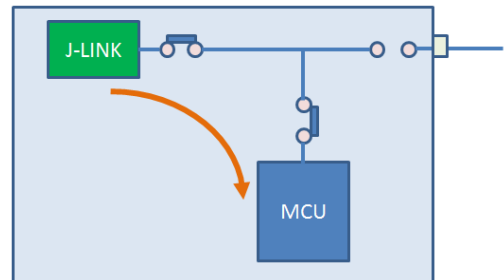
登録が完了すると、Device タブにユーザ基板上の EFR32 が追加されます。あとは、Wireless Starter Kit 上の EFR32 (BGM1xx) と同様に使用できます。



デバッグ対象を Wireless Starter Kit 上の BGM1xx に戻す場合には、Adapter Configuration タブの Debug Mode で、MCU を選択し、Device hardware タブの Target part で Wireless Starter Kit 上で使用している型番を選択してください。MCU に切り替えただけでは自動認識しませんので、ご注意ください。  
 なお、Debug Mode の MCU、IN、OUT の違いは以下の通りです。

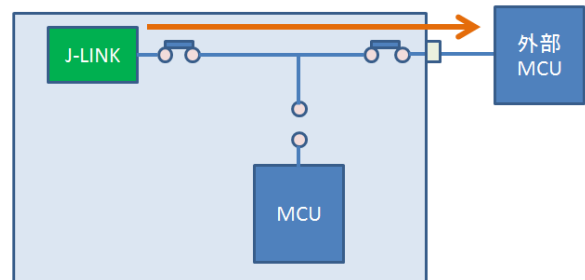
**MCU:**

Starter Kit 上のラジオボードがデバッグ対象



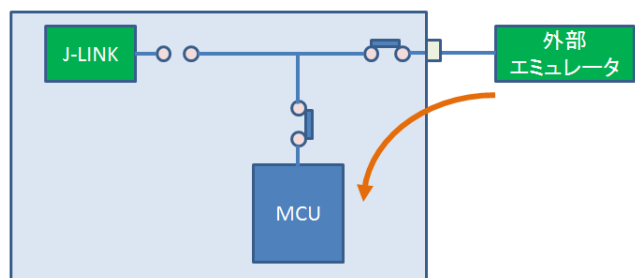
**OUT:**

外部のユーザボードがデバッグ対象



**IN:**

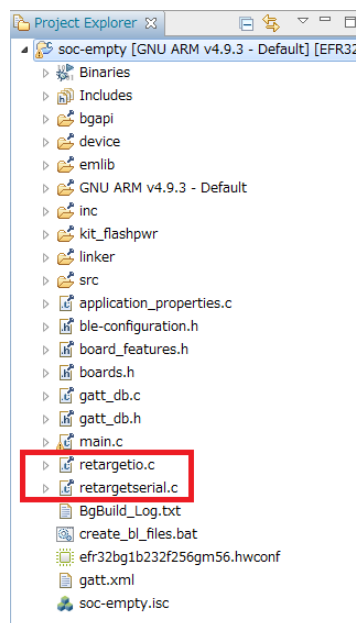
外部エミュレータを使って、StarterKit 上のラジオボードをデバッグ



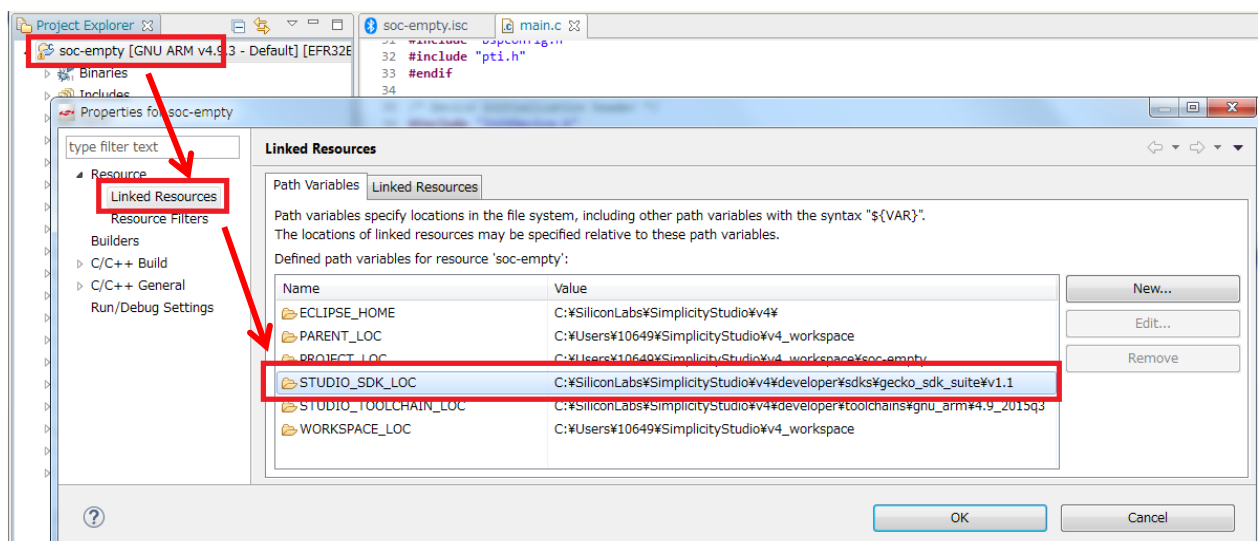
## 7-5 VCOM を利用した printf デバッグ

Wireless Starter Kit を使用して評価を行う際に、printf デバッグが使えると何かと便利です。ここでは、サンプルコードを例に、printf を実装する手順をご紹介します。

1. プロジェクトを作成します。この後の手順では、ハードウェアに「BGM121」を、サンプルプロジェクトに「SOC – Empty」を使用して説明します。
2. “STUDIO\_SDK\_LOC¥hardware¥kit¥common¥drivers” にある retargetserial.c と retargetio.c をプロジェクトにコピーします。ドラッグアンドドロップすれば良いです。



なお、STUDIO\_SDK\_LOC の位置は、Project Explorer でプロジェクトを選択して右クリック→Property→Resource→Linked Resources の順で確認頂けます。



- main.c に、stdio.h と retargetserial.h を include します。

<記述>

```
#include "stdio.h"
```

```
#include "retargetserial.h"
```

```

27 /* Libraries containing default Gecko configuration values */
28 #include "em_emu.h"
29 #include "em_cmu.h"
30 #ifdef FEATURE_BOARD_DETECTED
31 #include "bspconfig.h"
32 #include "pti.h"
33 #endif
34
35 /* Device initialization header */
36 #include "InitDevice.h"
37
38 #ifdef FEATURE_SPI_FLASH
39 #include "em_usart.h"
40 #include "mx25flash_spi.h"
41 #endif /* FEATURE_SPI_FLASH */
42
43 /* For printf */
44 #include "stdio.h"
45 #include "retargetserial.h"

```

- RETARGET\_SerialInit();を追加します。初期化の関数ですので、Printfなどを使用する前に実施が必要です。ここでは enter\_DefaultMode\_from\_RESET() の直後に入れてみます。

<記述>

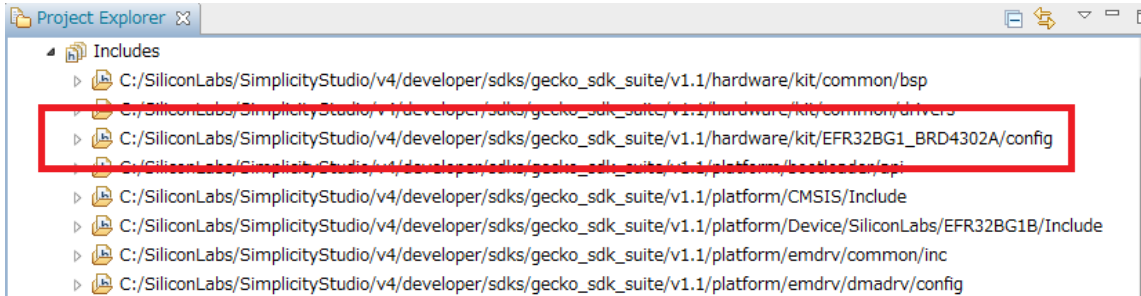
```
RETARGET_SerialInit();
```

```

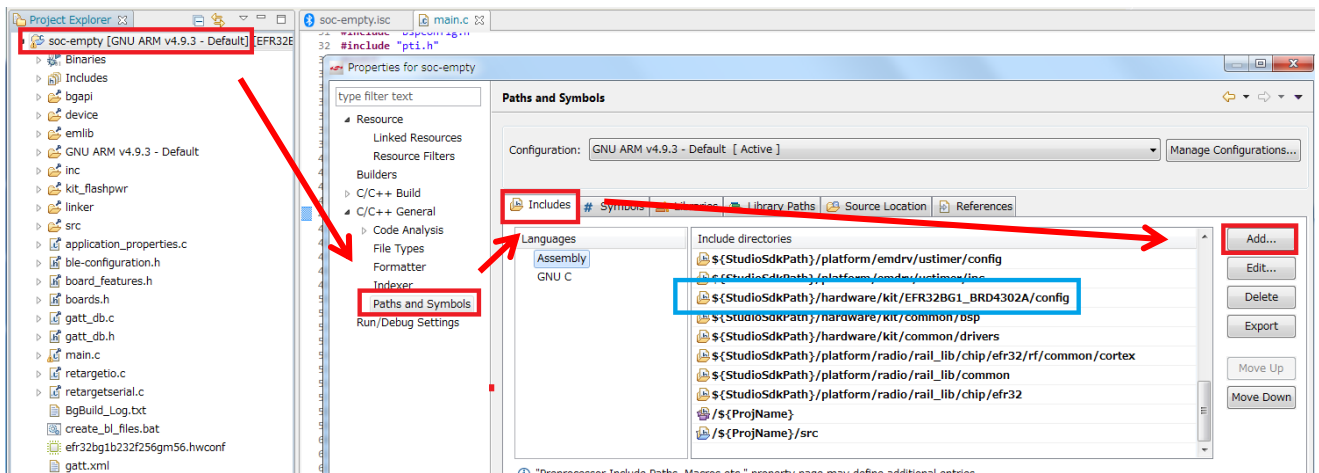
97
98 #endif /* FEATURE_SPI_FLASH */
99
100 /* Initialize peripherals */
101 enter_DefaultMode_from_RESET();
102
103 /* Initialize printf */
104 RETARGET_SerialInit();
105
106 /* Initialize stack */
107 gecko_init(&config);
108
109 while (1) {
110     /* Event pointer for handling
111     struct gecko cmd packet* evt;

```

- Project Explorer の Include で、“STUDIO\_SDK\_LOC¥hardware¥kit¥ラジオボード名¥config” (評価ボード用 header ファイル)へ path が通っているか確認します。下図は BGM121 のラジオボード (BRD4302A) の場合です。SDK のバージョンやラジオボード種別によって path が通っていない場合があります。



path が通っていなかった場合には、retargetserialconfig.h と bspconfig.h をプロジェクトにコピーして使うか、或いは path を通してください。path を通すには、Project Explorer でプロジェクトを選択して右クリック→Property→C/C++ Build→Paths and Symbols→Includes→Add ボタン から行ってください。





- hal-config.h を開き、HAL\_VCOM\_ENABLE の値を 1 に変更します。

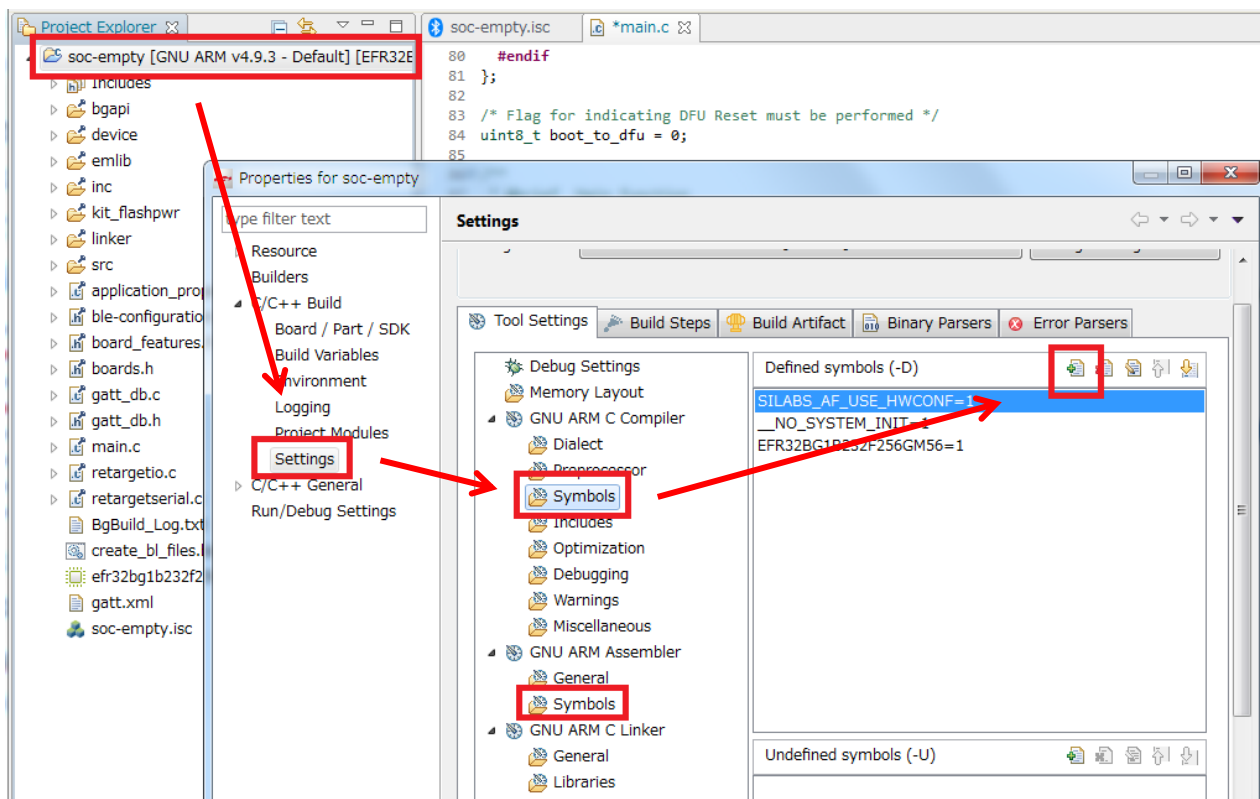
```

.h *hal-config.h
1 #ifndef HAL_CONFIG_H
2 #define HAL_CONFIG_H
3
4 #include "board_features.h"
5 #include "hal-config-board.h"
6 #include "hal-config-app-common.h"
7
8 #define HAL_VCOM_ENABLE (1)
9 #define HAL_I2CSENSOR_ENABLE (0)
10 #define HAL_SPIDISPLAY_ENABLE (0)
11
12 #endif
13
    
```

- hal-config.h が含まれていない古い SDK の場合には、RETARGET\_VCOM を define します。  
RETARGET\_VCOM は retargetserialconfig.h で使用しますので、retargetserialconfig.h そのものに追記するか、或いは下記手順でプロジェクトに登録してください。

<GCC の場合>

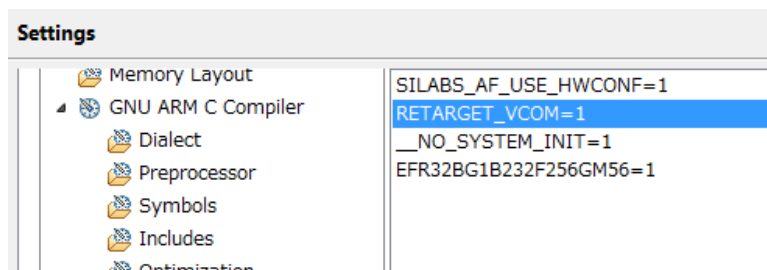
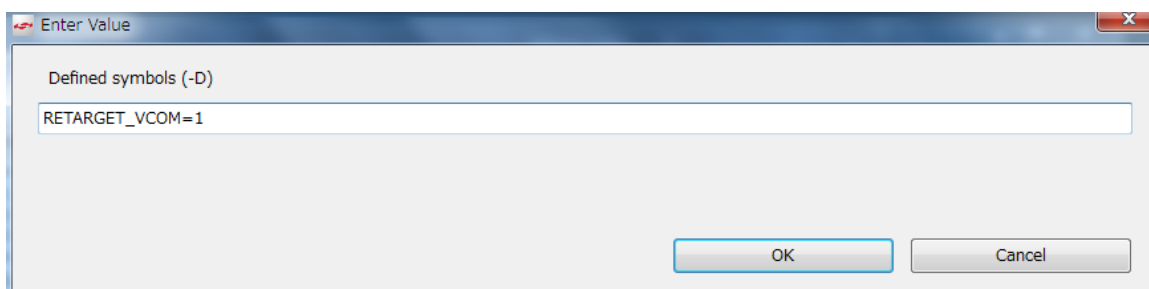
Project Explorer でプロジェクトを選択して右クリック→Property→C/C++ Build→Settings→GNU ARM C Compiler→Symbols→追加ボタンの順に進みます。



Enter Value ウィンドウにて、

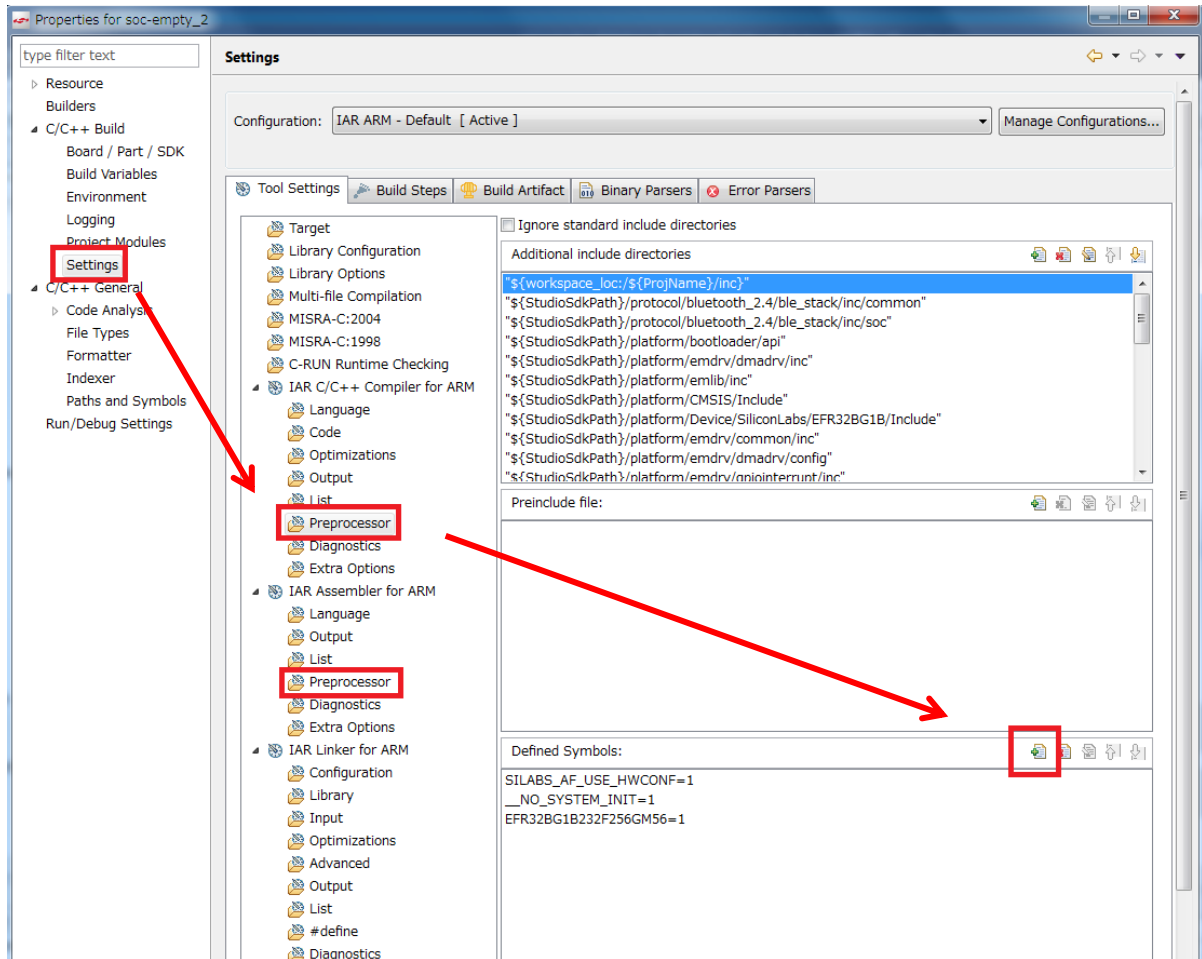
**RETARGET\_VCOM=1**

と記入し OK を押します。同じ手順を GNU ARM Assembler→Symbols でも行います。



<IAR コンパイラの場合>

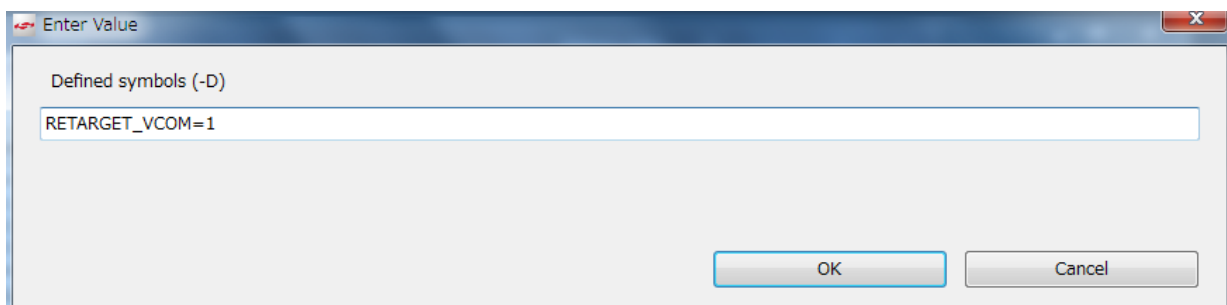
Project Explorer でプロジェクトを選択して右クリック→Property→C/C++ Build→Settings→IAR C/C+ Compiler for ARM→Preprocessor→Define Symbols→追加ボタンの順に進みます。

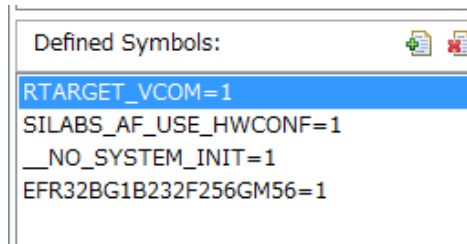


GCC と同様に、Enter Value ウィンドウにて、

**RETARGET\_VCOM=1**

と記入し OK を押します。同じ手順を IAR Assembler for ARM →Preprocessor でも行います。



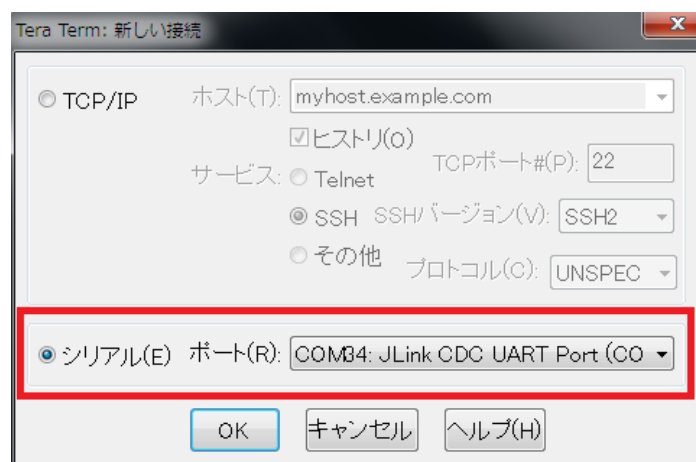


8. printfを使った記述を行います。RETARGET\_SerialInit();より後に行ってください。

```

105  /* Initialize printf */
106  RETARGET_SerialInit();
107
108  printf("Now Initializing...\r\n");
109
110  /* Initialize stack */
111  gecko_init(&config);
112
113  while (1) {
114      /* Event pointer for handling events */
115      struct gecko_cmd_packet* evt;
    
```

9. プロジェクトを Build し、BGM121 にダウンロードします。
10. Tera Term から、シリアルポート(JLink CDC)をオープンします。

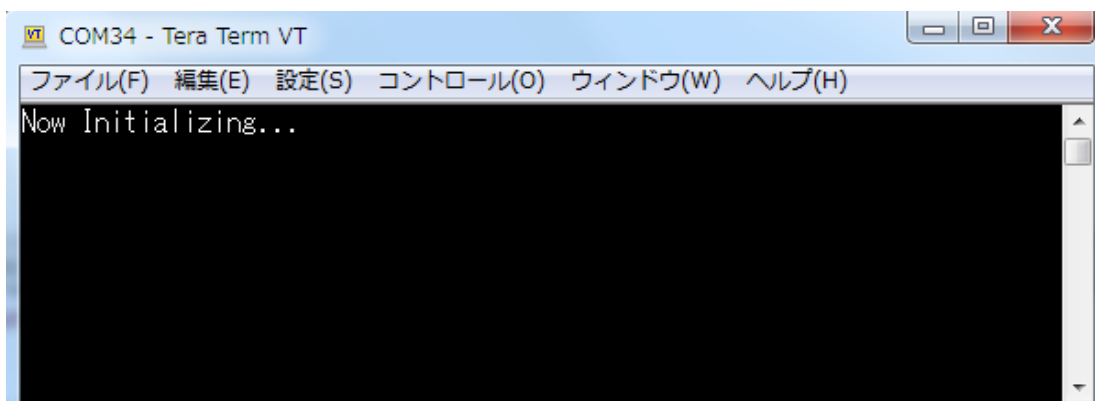


UART の設定は、デフォルト設定 (ボーレート 115200, data 8bit, non parity, 1 stopbit) が使われています。

```

em_usart.h
334 #define USART_INITASYNC_DEFAULT
335 {
336     usartEnable,    /* Enable RX/TX when init completed. */
337     0,              /* Use current configured reference clock for configuring baudrate. */
338     115200,         /* 115200 bits/s. */
339     usartOVS16,     /* 16x oversampling. */
340     usartDatabits8, /* 8 databits. */
341     usartNoParity,  /* No parity. */
342     usartStopbits1, /* 1 stopbit. */
343     false,          /* Do not disable majority vote. */
344     false,          /* Not USART PRS input mode. */
345     usartPrsRxCh0, /* PRS channel 0. */
346     false,          /* Auto CS functionality enable/disable switch */
347     0,              /* Auto CS Hold cycles */
348     0               /* Auto CS Setup cycles */
349 }
    
```

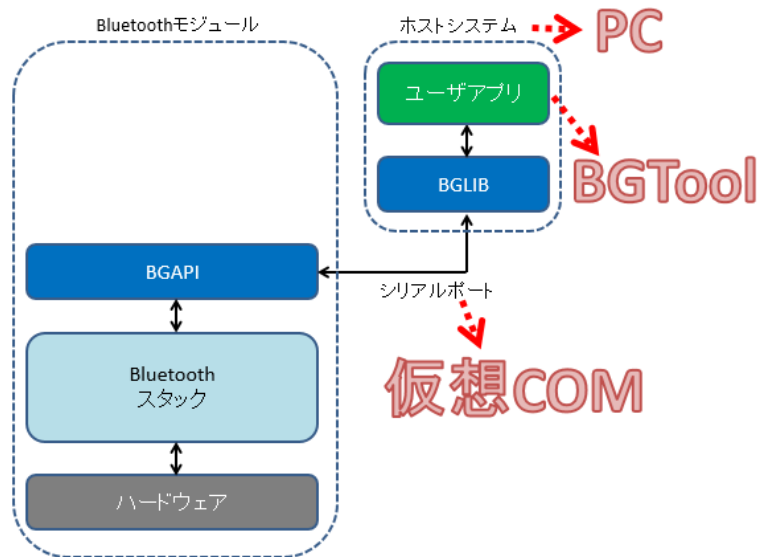
11. Wireless Starter Kit をリセットすると、文字が表示されます。



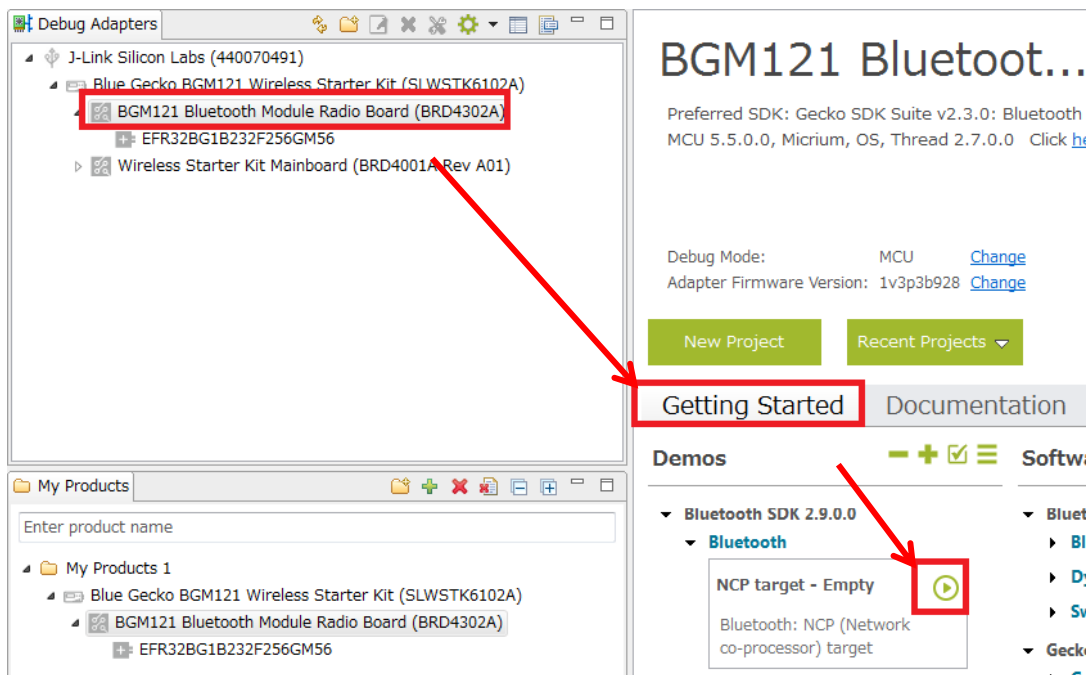
シリコンラボ社のコミュニティにも情報がございましたので、こちらも参照ください。([リンク](#))

7-6 BGTool を使って評価する (NCP モード)

BGM1xx にネットワーク・コプロセッサ(NCP)モードのファームウェアを書き込むと、BGTool から制御することができるようになります。下図は、2-3 章で紹介した NCP モードの制御図ですが、ホストシステムを PC が、ユーザアプリを BGTool が役割を担うこととなります。Bluetooth モジュールとホストシステムを繋ぐシリアルポートは仮想 COM ポート(Wireless Starter Kit 上の J-LINK に CDC が実装されている)を使用します。



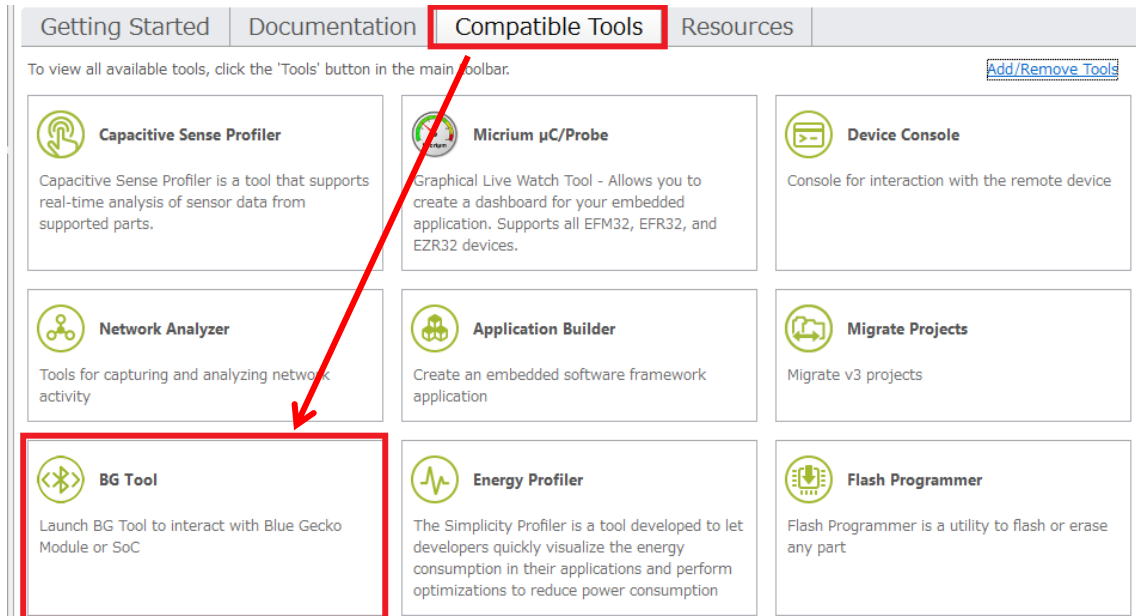
まず、BGM1xx に、NCP モードのファームウェアを書き込みます。Wireless Starter Kit を接続し、Getting Started タブの Demos の中から、「NCP target - Empty」をダウンロードします。



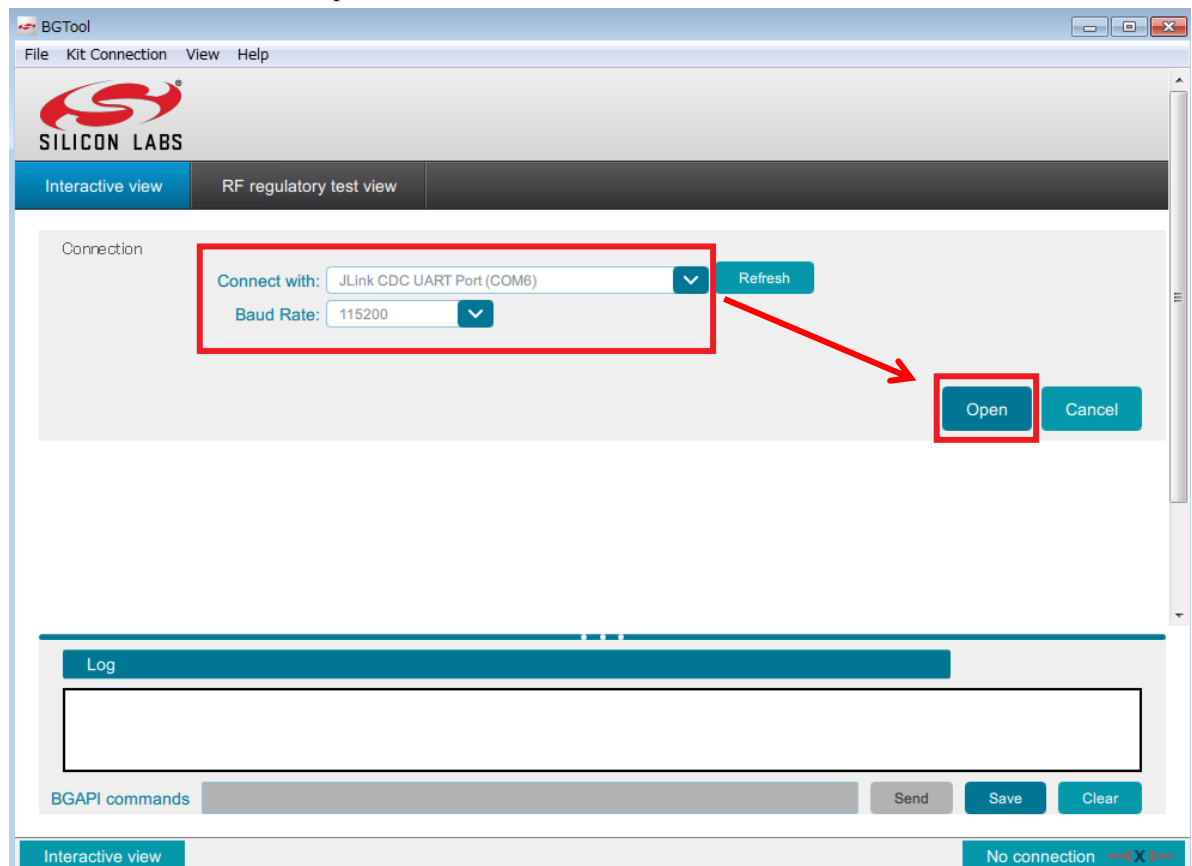
ここでは Demos を使用しましたが、Sample Examples の中に用意された NCP target をビルドし、ダウン

ロードしても良いです。また、NCP target – Empty の詳細仕様は AN1042 をご参照ください。(リンク)

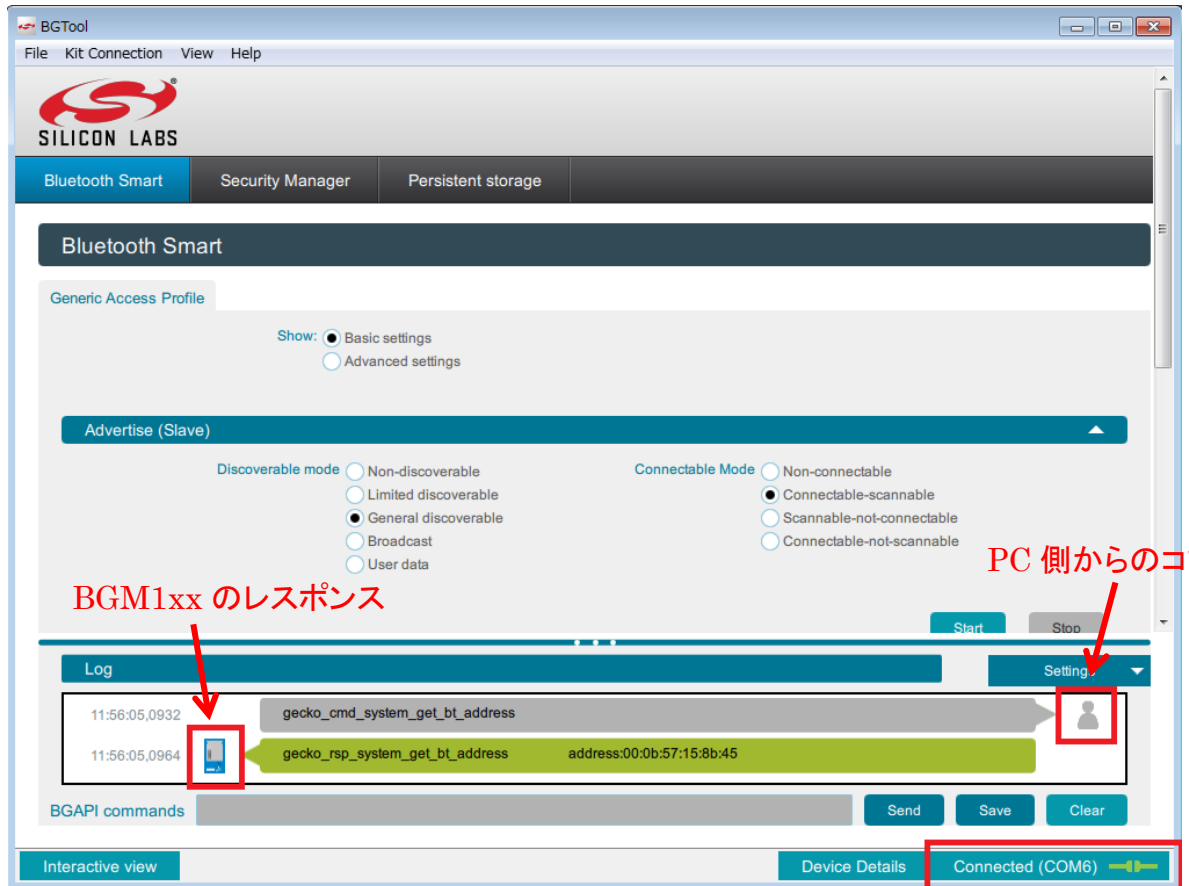
ダウンロードが完了したら、Compatible Tools タブから BG Tool を選択します。



BG Tool が起動します。Connect with で JLink CDC UART Port (仮想 COM です) を選択し、ポートは 115200 に設定して、Open をクリックします。



右下の接続状態を示すメッセージが“Connected”に変わり、接続できたことが確認できます。画面上に制御用ウィンドウがあり、アダプタイズを開始(送信)とスキャン(受信)が行えます。画面下にはLogが表示されます。Logでは、BG Tool(PC)側からのコマンドと、それに対するBGM1xxのレスポンスが確認できます。下図の例では、BG Tool から bt\_address を調べるコマンドを送り、BGM1xx がアドレス情報を返しています。



- Advertise (Slave)

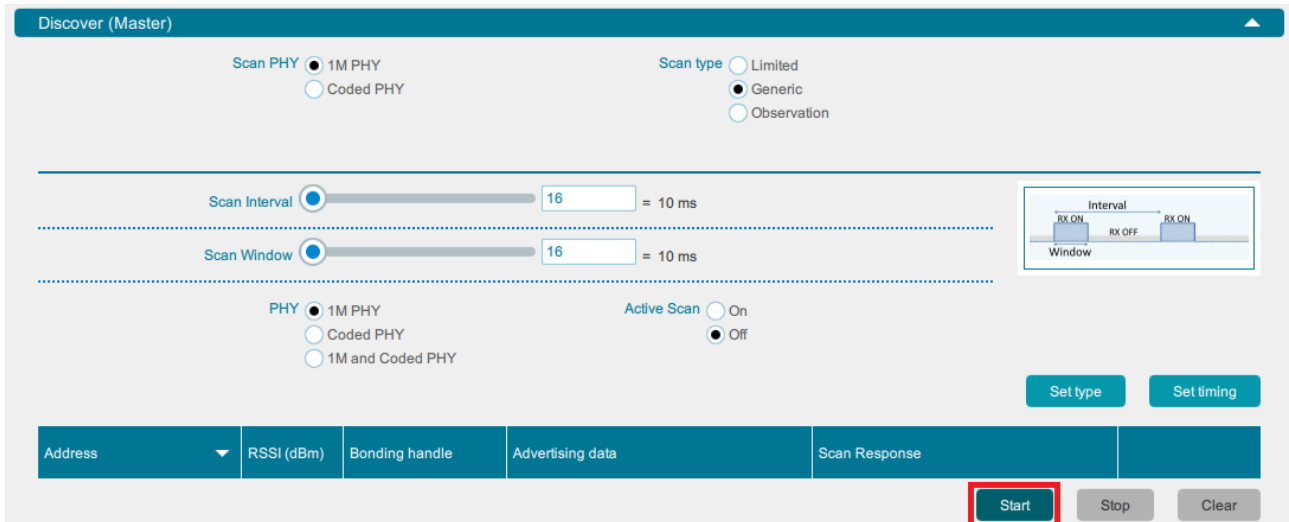
アダプタイズのモードを設定して Start ボタンをクリックすると、アダプタイズが開始されます。“Show”で“Advanced settings”を選択すると、インターバルの設定などが行えます。





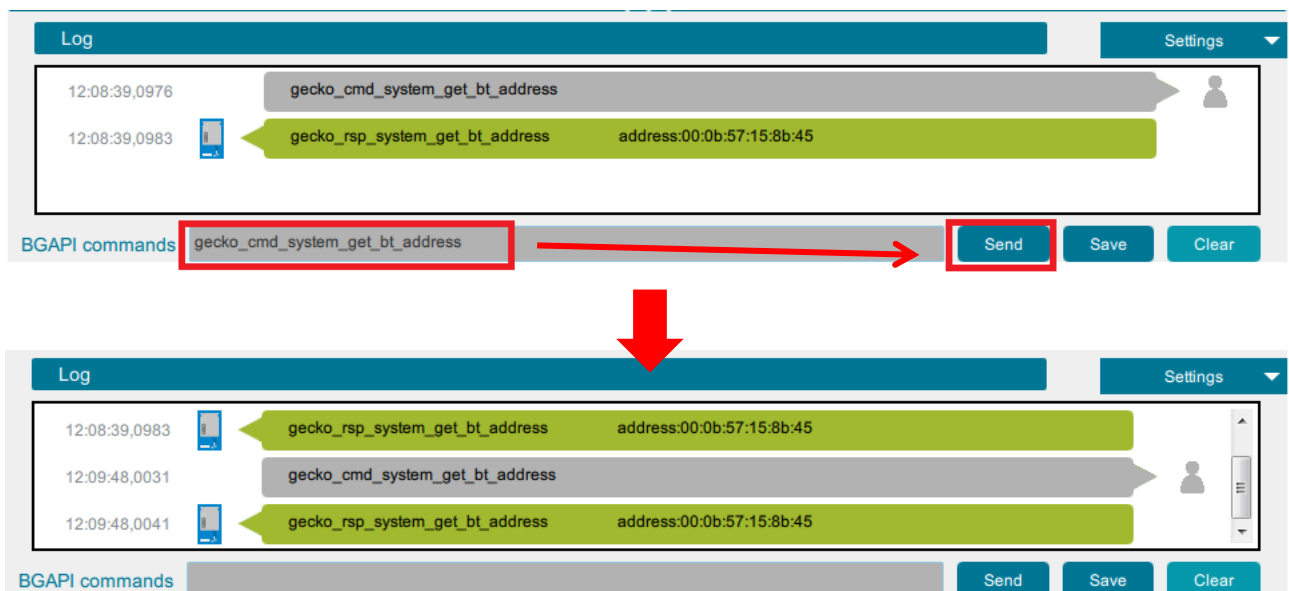
- Advertise (Master)

スキャンを実行することができます。Start ボタンをクリックするとスキャンを開始し、受信したアドレス、RSSI(信号強度)、アドバタイズデータなどを閲覧できます。“Advanced settings”を選択すると、スキャン・インターバルやスキャン・ウィンドウの設定が行えます。



- API を直接実行する

Log の下に、コマンドを直接入力できる欄も用意されています。コマンドを入力し、Send ボタンをクリックします。下図の例では、もう一度 `gecko_cmd_system_get_bt_address` を送っています。



- 注意点

BGToolを使って評価するには、NCP モードのファームウェアのダウンロードが必須です。それ以外のファームウェア(例えば Smart Phone App)をダウンロードした状態では、BGM1xx に接続することができませんので、ご注意ください。(Connected の状態にはなりますが、そこから先の通信が行えません)

### 7-7 RF PHY の特性を評価する

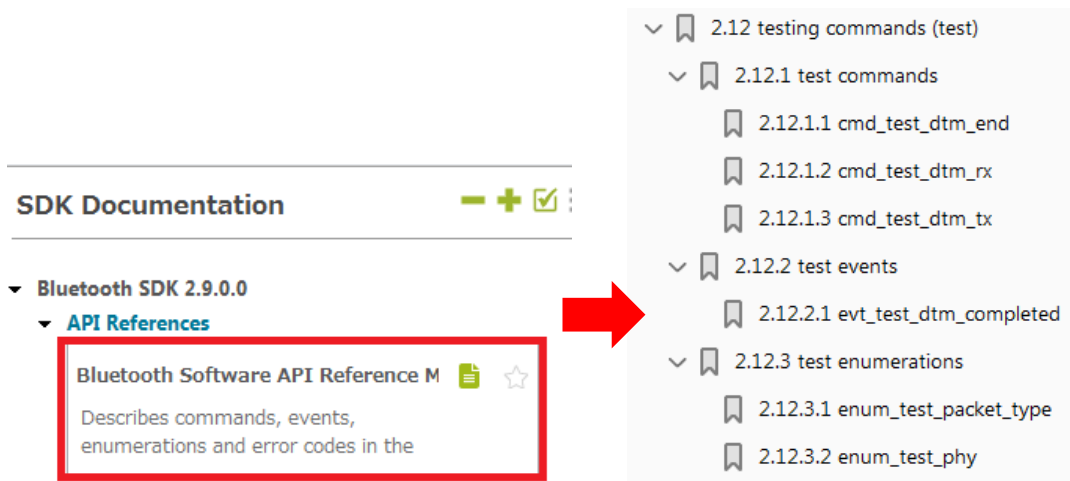
BGM1xx を海外で使用するには、現地国の電波法を順守する必要があります。BGM1xx は、あらかじめ幾つかの国の電波法認証を取得していますが、取得していない国についてはお客様による認証取得が必要になります。また、最終製品でなければ認証を取得できない(モジュールではフル認証は取得できない)国もあり、やはり同様に認証取得が必要になります。

BGM1xx および Bluetooth スタックには、無線試験を想定した機能が実装されています。本章ではその手順を簡単にご紹介します。詳細は AN1046 をご参照ください。(リンク) 本章で紹介しない DTM (direct test mode) を使用した手順も紹介されています。

#### 7-7-1 テストコマンドを使用する

Bluetooth スタックには、テスト用のコマンドが用意されています。ユーザコードからコマンド実行することで、BGM1xx に特定のテスト用動作をさせることができます。

使用できるコマンドについては、API リファレンス・マニュアル(Bluetooth Software API Reference Manual)の 2.12 章に記載されています。



- 受信コマンド (cmd\_test\_dtm\_rx)

使用するチャンネルを指定して、受信モードに移行することができます。

Byte	Type	Name	Description
0	0x20	hlen	Message type: Command
1	0x02	lolen	Minimum payload length
2	0x0e	class	Message class: testing commands
3	0x01	method	Message ID
4	uint8	channel	Bluetooth channel <b>Range:</b> 0-39 Channel is (F - 2402) / 2, where F is frequency in MHz
5	uint8	phy	PHY to use

- 送信コマンド (cmd\_test\_dtm\_tx)

使用するチャンネルや PHY タイプ (1M PHY, 2M PHY など)、送信するパケットタイプ (無変調, 特定データパターンなど)、パケット長を指定して、送信することができます。

Byte	Type	Name	Description
0	0x20	hlen	Message type: Command
1	0x04	lolen	Minimum payload length
2	0x0e	class	Message class: testing commands
3	0x00	method	Message ID
4	uint8	packet_type	Packet type to transmit
5	uint8	length	Packet length in bytes <b>Range:</b> 0-255
6	uint8	channel	Bluetooth channel <b>Range:</b> 0-39 Channel is $(F - 2402) / 2$ , where F is frequency in MHz
7	uint8	phy	PHY to use

packet\_type:

Value	Name	Description
0	test_pkt_prbs9	PRBS9 packet payload
1	test_pkt_11110000	11110000 packet payload
2	test_pkt_10101010	10101010 packet payload
3	test_pkt_carrier_deprecated	Unmodulated carrier - deprecated
4	test_pkt_11111111	11111111 packet payload
5	test_pkt_00000000	00000000 packet payload
6	test_pkt_00001111	00001111 packet payload
7	test_pkt_01010101	01010101 packet payload
253	test_pkt_pn9	PN9 continuously modulated output
254	test_pkt_carrier	Unmodulated carrier

Phy:

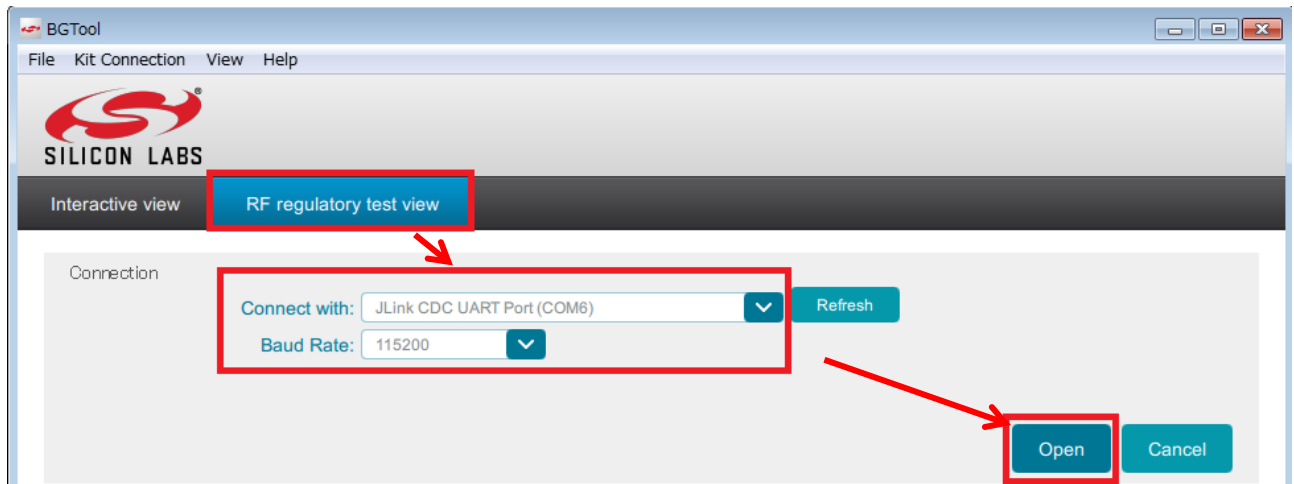
Value	Name	Description
1	test_phy_1m	1M PHY
2	test_phy_2m	2M PHY
3	test_phy_125k	125k Coded PHY
4	test_phy_500k	500k Coded PHY

各コマンドに対するレスポンスや、その他コマンドについては、API リファレンス・マニュアルをご参照ください。

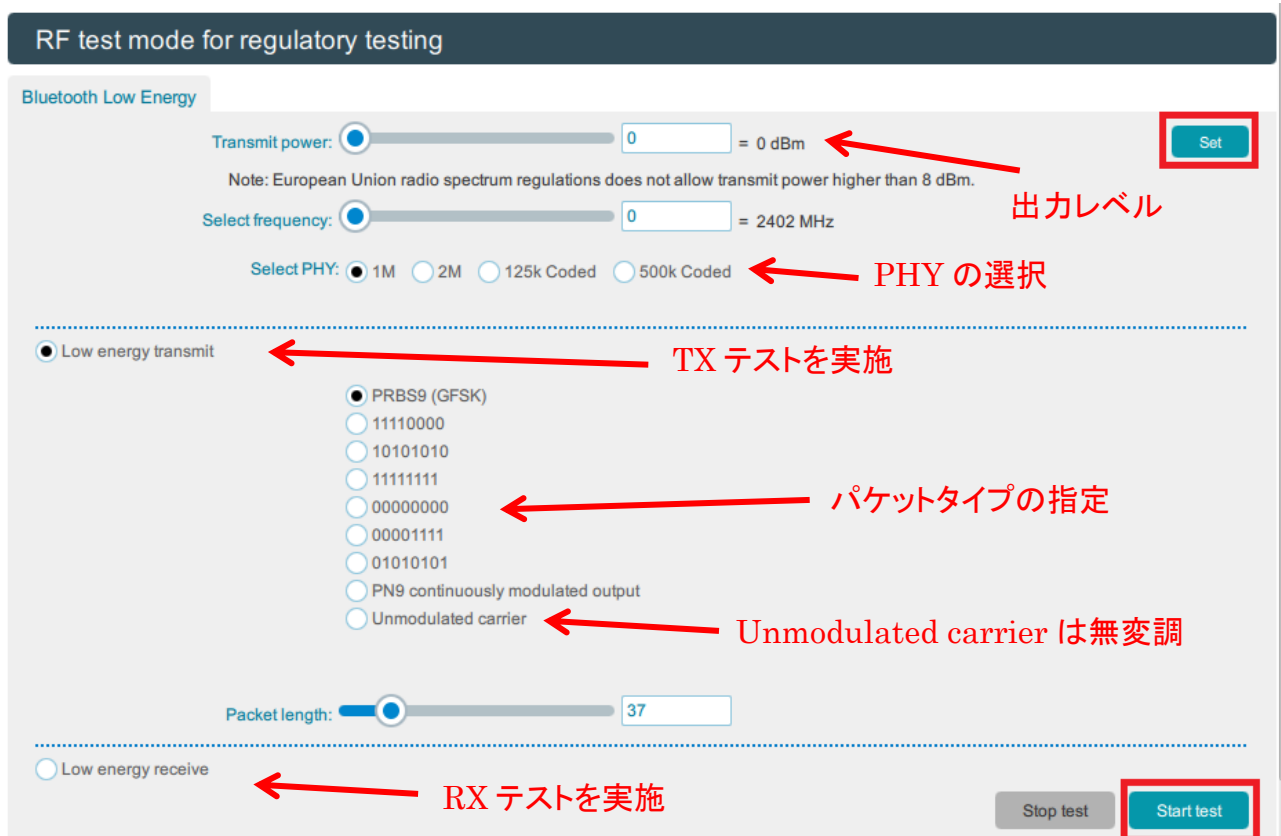
## 7-7-2 BGTool を使用する

Bluetooth スタックに用意されたテストコマンドは、ユーザコードから実行するだけでなく、BGTool から使用できます。

「7-6 BGTool を使って評価する (NCP モード)」を参考に、BGM1xx に NCP モードのサンプルコードをダウンロードし、それから BGTool を起動してください。次に、RF regulatory test view タブを選び、Connect with で JLink CDC UART Port (仮想 COM です) を選択し、ボーレートは 115200 に設定して、Open をクリックします。



「7-7-1 テストコマンドを使用する」で紹介した内容が、GUI 上で操作できます。必要な設定を行い、Start test で動作開始します。



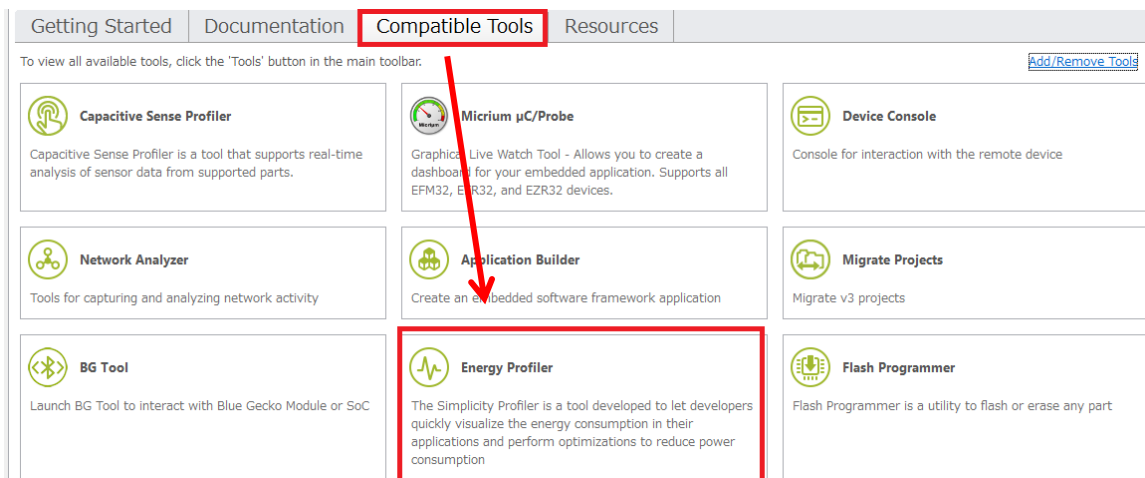
### 7-8 消費電流を測定してみる (Energy Profiler)

Wireless Starter Kit には電流センサが搭載されており、消費電流測定ツール (Energy Profiler) と組み合わせることで電流測定が可能です。ただし、サンプリング周波数はそれほど速くありませんので、厳密な測定にはオシロスコープが必要になります。詳しくは AN969 をご参照ください。(リンク)

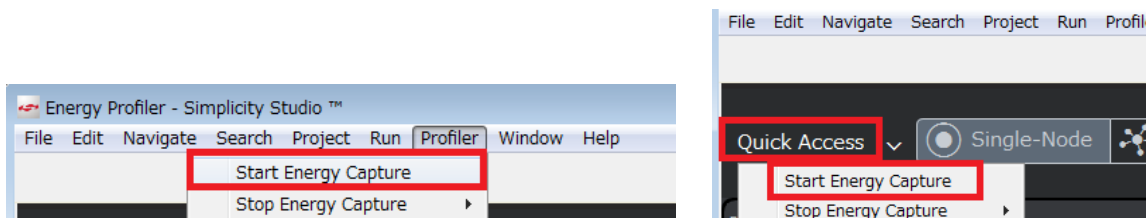
ここではサンプルコードを使用して、消費電流測定ツール (Energy Profiler) を使った簡易評価の方法をご紹介します。

BGM121 には、あらかじめ、Demos「SoC – Smart Phone App」を書き込んでおきます。

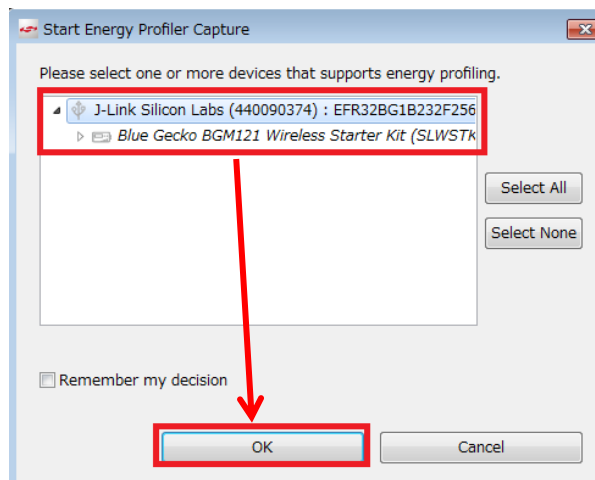
次に、Compatible Tools の中にある Energy Profiler を起動します。



Profiler ⇒ Start Energy Capture を選択します。画面左上の Quick Access から選択しても良いです。



測定対象を聞かれますので、接続している Wireless Starter Kit を選択し、OK します。



電流測定がスタートします。



スタート/ストップ (Running/Paused) の切り替え、対数/リニア表示の切り替え、X 軸・Y 軸の拡大/縮小などの機能が付いていますので、操作性を体感ください。平均電流も表示されています。



波形上でクリックするとカーソルが出ます。電流のピーク間を測定すると間隔はおよそ 100ms で、その周期でアドバタイズしていることが判ります。



スマートフォンから接続すると、その挙動が電流波形からも見て取れます。



## 8 ソフトウェア設計

ソフトウェア設計に役立つ情報をご紹介します。

### 8-1 ソースコードの追い方

Simplicity IDE でソースコードを追うための方法を紹介します。

- ◆ 変数や関数を定義している記述を探す

```
while (1)
{
  if(i2c_rxInProgress){
    /* Receiving data */
    receiveI2CData();
  }else if (i2c_startTx){
    /* Transmitting data */
    performI2CTransfer();
    /* Transmission complete */
    i2c_startTx = false;
  }
}
```

①調べたい変数  
や関数をクリック  
(色がグレーに変わる)

```
while (1)
{
  if(i2c_rxInProgress){
    /* Receiving data */
    receiveI2CData();
  }else if (i2c_startTx){
    /* Transmitting data */
    performI2CTransfer();
    /* Transmission complete */
    i2c_startTx = false;
  }
}
```

```
void performI2CTransfer(void)
{
  /* Transfer structure */
  I2C_TransferSeq_TypeDef i2cTransfer;

  /* Setting pin to indicate transfer */
  GPIO_PinOutSet(gpioPortC, 0);
}
```

②右クリックして、  
Open Declaration を選択  
(F3 キーでも OK)

Open Declaration	F3
Open Type Hierarchy	F4

③変数や関数の定義にジャンプ

\* 上記の説明では、EFM32 向けのコードを使用しています。BGM1xx でも手順は同じです。



## 8-2 サンプルコードにペリフェラルを実装してみる（外部割込み）

BGM1xx には、Smart Phone App や iBeacon といった、Bluetooth 制御を学ぶためのサンプルコードが用意されていますが、各ペリフェラルのサンプルコードはありません。

一方で、EFM32 には非常に多くのサンプルコードが用意されています。実際のアプリケーションをイメージしたサンプルコードもありますが、ADC のサンプルコード、UART のサンプルコード、外部割込みのサンプルコード…といった具合に 1 つのペリフェラルにスポットを当てたサンプルコードも多く、ペリフェラルの機能・動作を学ぶのに非常に役立ちます。

実際のアプリケーション設計では、Bluetooth 機能だけでなく、他のペリフェラルも使用することがほとんどかと思しますので、BGM1xx のサンプルコードに別のペリフェラルを追加する手順を学ぶことは非常に有益です。

この章では、2 つのサンプルコードを migration していく手順について紹介します。

題材として、「SOC – iBeacon」と「SLSTK3401A\_gpio\_int\_pg1b」という 2 つのサンプルコードを使用します。「SOC – iBeacon」は BGM1xx 用のサンプルコードで、今回は BGM121 向けに実装されたものを使います。「SLSTK3401A\_gpio\_int\_pg1b」は AN0012「General Purpose Input Output」に付属したサンプルコードで、EFM32PG starter kit (SLSTK3401A) 向けです。

「SOC – iBeacon」は、ビーコン送信するだけのサンプルコードです。

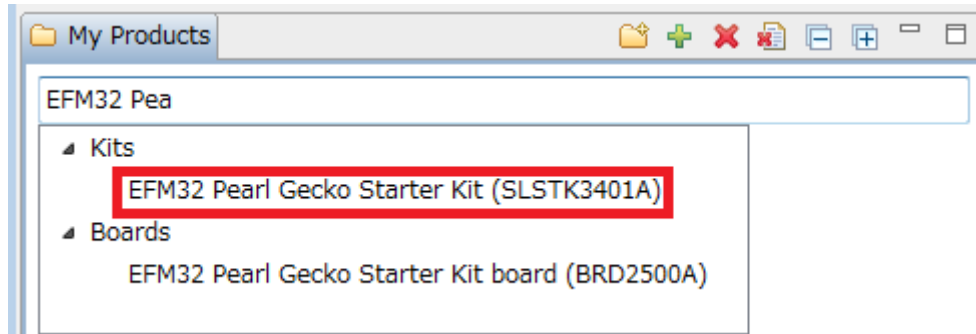
「SLSTK3401A\_gpio\_int\_pg1b」は、ボタンを押すと外部割込みが生じ、LED を反転 (ON→OFF 或いは OFF→ON) するサンプルコードです。

大まかな流れとしては、

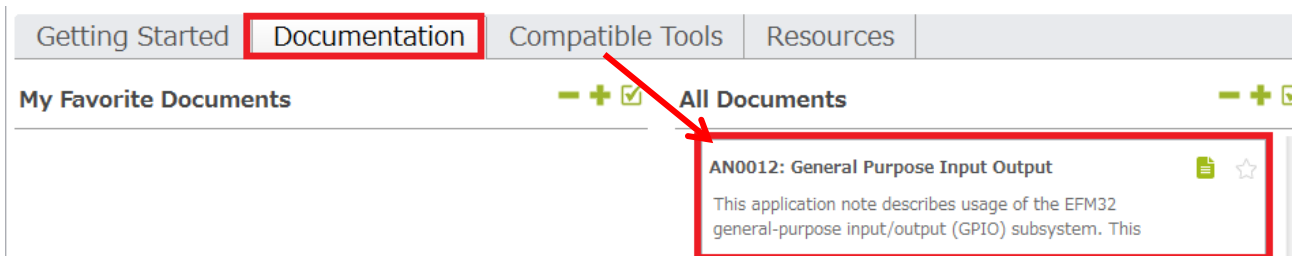
- サンプルコードを理解する (8-2-1、8-2-2)
- 「SLSTK3401A\_gpio\_int\_pg1b」のペリフェラル設定を、「SOC – iBeacon」に移植する (8-2-3) です。

## 8-2-1 サンプルコードを理解する (SLSTK3401A\_gpio\_int\_pg1b)

まずはサンプルコードをロードします。My Products タブで EFM32 Pearl とタイプし、EFM32 Pearl Gecko Starter Kit を選びます。

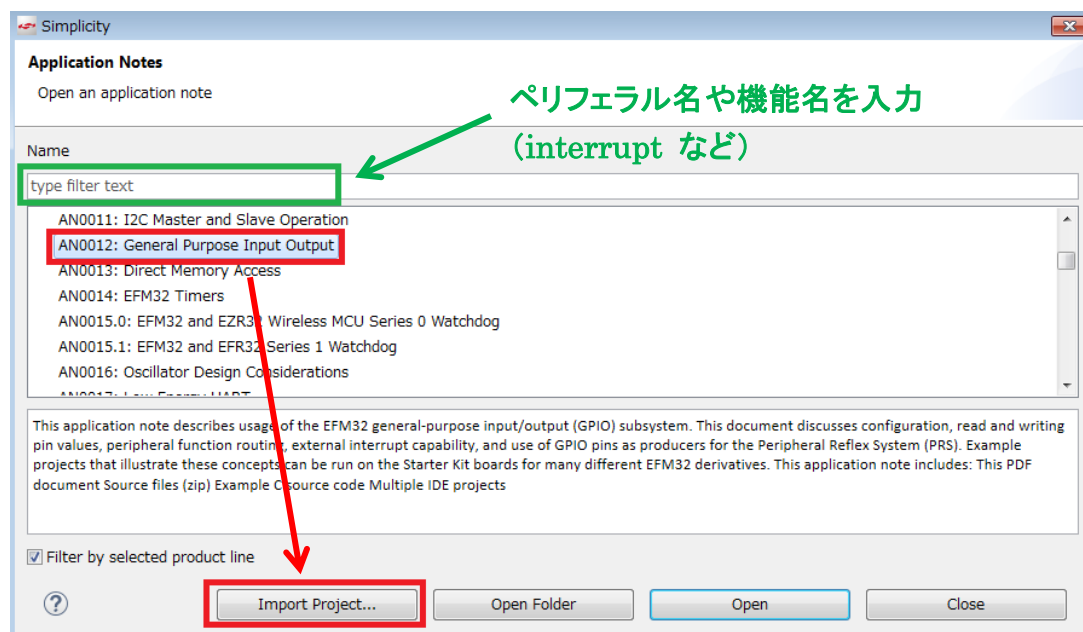


次に、Documentation タブ ⇒ Application Notes ⇒ AN0012 を選択します。

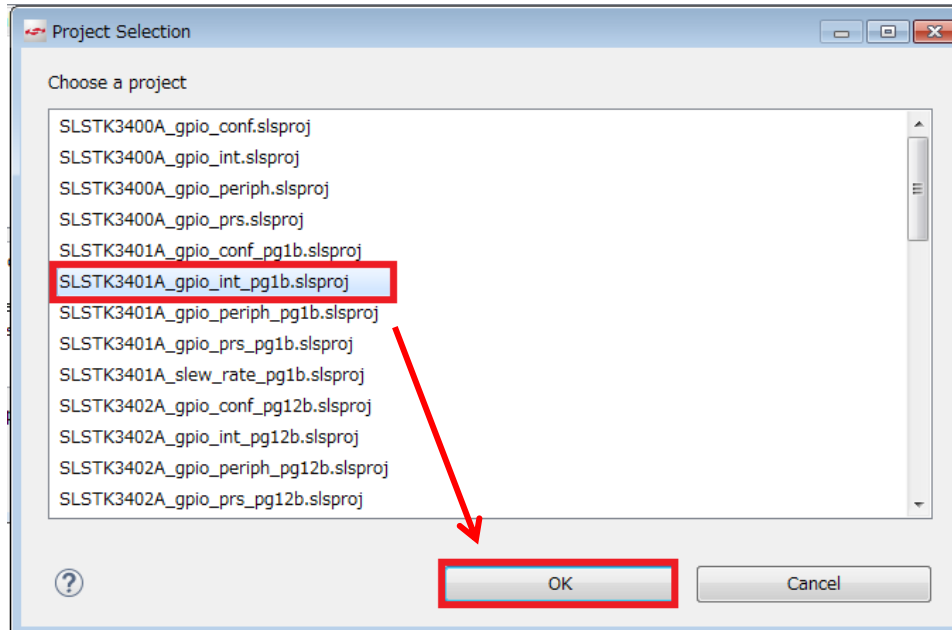


AN0012 が選択されていることを確認して、Import Project をクリックします。

なお、必要なペリフェラルのサンプルコードを見つけれない場合には、下図の緑枠 (type filter text) の部分に、ADC や interrupt など のペリフェラル名を入力してください。候補がリストアップされます。



AN0012 に含まれたサンプルコードがリストアップされますので、「SLSTK3401A\_gpio\_int\_pg1b」を選択し、OK をクリックします。



プロジェクトがロードされます。コードが 1 つだけあり、main\_gpio\_int.c がユーザコードです。割込みハンドラ、GPIO の初期化関数、main 関数 から構成されています。

### main\_gpio\_int.c

```

41- /******//**
42  * @brief GPIO Even IRQ for pushbuttons on even-numbered pins
43  * *****//**
44- void GPIO_EVEN_IRQHandler(void)
45  {
46  // Clear all even pin interrupt flags
47  GPIO_IntClear(0x5555);
48
49  // Toggle LED0
50  GPIO_PinOutToggle(BSP_GPIO_LED0_PORT, BSP_GPIO_LED0_PIN);
51  }
52
53- /******//**
54  * @brief GPIO Odd IRQ for pushbuttons on odd-numbered pins
55  * *****//**
56- void GPIO_ODD_IRQHandler(void)
57  {
58  // Clear all odd pin interrupt flags
59  GPIO_IntClear(0xAAAA);
60
61  // Toggle LED01
62  GPIO_PinOutToggle(BSP_GPIO_LED0_PORT, BSP_GPIO_LED0_PIN);
63  }
--
    
```

割込み  
クリア

偶数番号ピンの割込みハンドラ

LED をトグル(反転)

奇数番号ピンの割込みハンドラ

```

65 //*****
66 * @brief GPIO initialization
67 *****
68 void initGPIO(void)
69 {
70     // Configure GPIO pins
71     CMU_ClockEnable(cmuClock_GPIO, true);
72
73     // Configure PB0 and PB1 as input with glitch filter enabled
74     GPIO_PinModeSet(BSP_GPIO_PB0_PORT, BSP_GPIO_PB0_PIN, gpioModeInputPullFilter, 1);
75     GPIO_PinModeSet(BSP_GPIO_PB1_PORT, BSP_GPIO_PB1_PIN, gpioModeInputPullFilter, 1);
76
77     // Configure LED0 and LED1 as output
78     GPIO_PinModeSet(BSP_GPIO_LED0_PORT, BSP_GPIO_LED0_PIN, gpioModePushPull, 0);
79
80     // Enable IRQ for even numbered GPIO pins
81     NVIC_EnableIRQ(GPIO_EVEN_IRQn);
82
83     // Enable IRQ for odd numbered GPIO pins
84     NVIC_EnableIRQ(GPIO_ODD_IRQn);
85
86     // Enable falling-edge interrupts for PB pins
87     GPIO_IntConfig(BSP_GPIO_PB0_PORT, BSP_GPIO_PB0_PIN, 0, 1, true);
88     GPIO_IntConfig(BSP_GPIO_PB1_PORT, BSP_GPIO_PB1_PIN, 0, 1, true);
89 }

91 //*****
92 * @brief Main function
93 *****
94 int main(void)
95 {
96     // Chip errata
97     CHIP_Init();
98
99     // Initializations
100    initGPIO();
101
102    while (1){
103        // Enter Low Energy Mode until an interrupt occurs
104        EMU_EnterEM3(false);
105    }
106 }

```

← GPIO の初期化関数

GPIO\_PinModeSet(BSP\_GPIO\_PB0\_PORT, BSP\_GPIO\_PB0\_PIN, gpioModeInputPullFilter, 1);  
 GPIO\_PinModeSet(BSP\_GPIO\_PB1\_PORT, BSP\_GPIO\_PB1\_PIN, gpioModeInputPullFilter, 1);  
 GPIO\_PinModeSet(BSP\_GPIO\_LED0\_PORT, BSP\_GPIO\_LED0\_PIN, gpioModePushPull, 0);

← ピン設定 (ボタン、LED 用)

NVIC\_EnableIRQ(GPIO\_EVEN\_IRQn);  
 NVIC\_EnableIRQ(GPIO\_ODD\_IRQn);

← 外部割込み有効

GPIO\_IntConfig(BSP\_GPIO\_PB0\_PORT, BSP\_GPIO\_PB0\_PIN, 0, 1, true);  
 GPIO\_IntConfig(BSP\_GPIO\_PB1\_PORT, BSP\_GPIO\_PB1\_PIN, 0, 1, true);

← ピンと外部割込みの紐付け

← main 関数

initGPIO();

while ループに入る前に、  
GPIO 初期化関数を呼んでいる

## 8-2-2 サンプルコードを理解する (SOC - iBeacon)

Device 或いは Solution タブで BGM121 Wireless Starter Kit を選択して、SOC-iBeacon のサンプルコードをロードします。main.c がユーザコードです。

使用する各関数の定義と、main 関数から構成されています。

```

main.c
76  */
77 void bcnSetupAdvBeaconing(void)
78 {
79  /* This function sets up a custom advertisement package according to iBeacon specifications.
80   * The advertisement package is 30 bytes long. See the iBeacon specification for further details.
81   */
82
83   static struct {
84     uint8_t flagsLen;    /* Length of the Flags field. */
85     uint8_t flagsType;  /* Type of the Flags field. */
86     uint8_t flags;      /* Flags field. */
87     uint8_t mandataLen; /* Length of the Manufacturer Data field. */
88     uint8_t mandataType; /* Type of the Manufacturer Data field. */
89     uint8_t compId[2];  /* Company ID field. */
90     uint8_t beacType[2]; /* Beacon Type field. */
91
92     ...
93
94   };
95
96   ...
97
98   ...
99
100  ...
101
102  ...
103
104  ...
105
106  ...
107
108  ...
109
110  ...
111
112  ...
113
114  ...
115
116  ...
117
118  ...
119
120  ...
121
122  ...
123
124  ...
125
126  ...
127
128  ...
129
130  ...
131
132  ...
133
134  ...
135
136  ...
137
138  ...
139
140  ...
141
142  ...
143
144  ...
145
146  ...
147
148  ...
149
150  ...
151
152  /**
153   * @brief Main function
154   */
155  void main(void)
156  {
157    // Initialize device
158    initMcu();
159    // Initialize board
160    initBoard();
161    // Initialize application
162    initApp();
163
164    // Initialize stack
165    gecko_init(&config);
166
167    while (1) {
168      struct gecko_cmd_packet* evt;
169
170      // Check for stack event.
171      evt = gecko_wait_event();
172
173      // Run application and event handler.
174      switch (BGLIB_MSG_ID(evt->header)) {
175        // This boot event is generated when the system boots up after reset.
176        // Do not call any stack commands before receiving the boot event.
177        case gecko_evt_system_boot_id:
178          // Initialize iBeacon ADV data
179          bcnSetupAdvBeaconing();
180          break;
181
182        default:
183          break;
184      }
185    }
186  }
187
188  /** @} (end addtogroup app) */
189  /** @} (end addtogroup Application) */
190

```

← 各関数や変数の定義

← main 関数

← 初期化関数

初期化関数として、initMCU(); initBoard(); initApp(); gecko\_init(&config); の4つが呼ばれています。少し詳しく見ていきます。詳しくは UG136 を参照ください。

- initMCU()

initMCU() は、init\_MCU.c の中で定義されています。

```

init_MCU.c
34 void initMcu(void)
35 {
36     // Device errata
37     CHIP_Init();
38
39     // Set up DC-DC converter
40     EMU_DCDCInit_TypeDef dcdcInit = BSP_DCDC_INIT;
41     #if HAL_DCDC_BYPASS
42     dcdcInit.dcdcMode = emuDcdcMode_Bypass;
43     #endif
44     EMU_DCDCInit(&dcdcInit);
45
46     // Set up clocks
47     initMcu_clocks();
48
49     RTCC_Init_TypeDef rtccInit = RTCC_INIT_DEFAULT;
50     rtccInit.enable           = true;
51     rtccInit.debugRun        = false;
52     rtccInit.precntWrapOnCCV0 = false;
53     rtccInit.cntWrapOnCCV1    = false;
54     rtccInit.prescMode        = rtccCntTickPresc;
55     rtccInit.presc            = rtccCntPresc_1;
56     rtccInit.enaOSCFailDetect = false;
57     rtccInit.cntMode          = rtccCntModeNormal;
58     RTCC_Init(&rtccInit);
59
60     #if defined(_EMU_CMD_EM01VSCALE0_MASK)
61     // Set up EM0, EM1 energy mode configuration
62     EMU_EM01Init_TypeDef em01Init = EMU_EM01INIT_DEFAULT;
63     EMU_EM01Init(&em01Init);
64     #endif // _EMU_CMD_EM01VSCALE0_MASK
65
66     #if defined(_EMU_CTRL_EM23VSCALE_MASK)
67     // Set up EM2, EM3 energy mode configuration
68     EMU_EM23Init_TypeDef em23init = EMU_EM23INIT_DEFAULT;
69     em23init.vScaleEM23Voltage = emuVScaleEM23_LowPower;
70     EMU_EM23Init(&em23init);
71     #endif // _EMU_CTRL_EM23VSCALE_MASK
72
73     TEMPDRV_Init();
74 }

```

UG136 の中では、init\_mcu.c と init\_mcu.h の役割は「These files include the device initialization function, which initializes internal settings of the MCU like clocks and power management.」と説明されています。DCDC や RTCC(リアルタイムクロック)の初期設定を行っています。

- initBoard()

initBoard() は、init\_board.c の中で定義されています。

#### init\_board.c

```
29 void initBoard(void)
30 {
31     // Enable clock for CRYOTIMER
32     CMU_ClockEnable(cmuClock_CRYOTIMER, true);
33     // Enable clock for PRS
34     CMU_ClockEnable(cmuClock_PRS, true);
35 #ifndef FEATURE_EXP_HEADER_USART3
36     // Enable clock for USART3
37     CMU_ClockEnable(cmuClock_USART3, true);
38 #else
39     // Enable clock for USART0
40     CMU_ClockEnable(cmuClock_USART0, true);
41 #endif
42     // Enable GPIO clock source
43     CMU_ClockEnable(cmuClock_GPIO, true);
44
45     // Put the SPI flash into Deep Power Down mode for those radio boards where it is available
46     MX25_init();
47     MX25_DP();
48     // We must disable SPI communication
49     USART_Reset(MX25_USART);
50 }
```

UG136 の中では、init\_board.c と init\_board.h の役割は「These files include the board initialization function, which initializes external parts on the board. For example, it enables GPIOs, and initializes external flash on the radio board.」と説明されています。Starter Kit 上に実装された SPI フラッシュメモリなどの外部部品に関する初期化などを行っています。

- initApp()

initApp() は、init\_app.c の中で定義されています。

#### init\_app.c

```
25 void initApp(void)
26 {
27     // Enable PTI
28     configEnablePti();
29
30     #if defined(HAL_VCOM_ENABLE)
31         // Enable VCOM if requested
32         GPIO_PinModeSet(BSP_VCOM_ENABLE_PORT, BSP_VCOM_ENABLE_PIN, gpioModePushPull, HAL_VCOM_ENABLE);
33     #endif // HAL_VCOM_ENABLE
34
35     #if (HAL_I2CSENSOR_ENABLE)
36         // Initialize I2C peripheral
37         I2CSPM_InitTypeDef i2cInit = I2CSPM_INIT_DEFAULT;
38         I2CSPM_Init(&i2cInit);
39     #endif // HAL_I2CSENSOR_ENABLE
40
41     #if defined(HAL_I2CSENSOR_ENABLE)
42         // Enable I2C sensor if requested
43         GPIO_PinModeSet(BSP_I2CSENSOR_ENABLE_PORT, BSP_I2CSENSOR_ENABLE_PIN, gpioModePushPull, HAL_I2CSENSOR_ENABLE);
44     #endif // HAL_I2CSENSOR_ENABLE
45
46     #if defined(HAL_SPIDISPLAY_ENABLE)
47         // Enable SPI display if requested
48         GPIO_PinModeSet(BSP_SPIDISPLAY_ENABLE_PORT, BSP_SPIDISPLAY_ENABLE_PIN, gpioModePushPull, HAL_SPIDISPLAY_ENABLE);
49     #endif // HAL_SPIDISPLAY_ENABLE
50 }
```

UG136 の中では、init\_app.c と init\_app.h の役割は「These files include the app initialization function, which initializes external parts on the WSTK according to the application. For example, it enables VCOM, sensors, and LCD display on the WSTK.」と説明されています。Starter Kit 上に実装された VCOM(仮想 COM)やセンサ、LCD ディスプレイなどの外部部品に関する初期化などを行っています。

- gecko\_init(&config)

gecko\_init()は、native\_gecko.h の中で定義されています。native\_gecko.h は、GATT エディタ(7-2 参照)が自動生成するファイルで、編集不可です。“config” は main.c の中で定義されており、最大接続数やヒープ領域のサイズなどを指定しています。

#### main.c

```
51 #ifndef MAX_CONNECTIONS
52 #define MAX_CONNECTIONS 4
53 #endif
54 uint8_t bluetooth_stack_heap[DEFAULT_BLUETOOTH_HEAP(MAX_CONNECTIONS)];
55
56 /* Gecko configuration parameters (see gecko_configuration.h) */
57 static const gecko_configuration_t config = {
58     .config_flags = 0,
59     .sleep.flags = SLEEP_FLAGS_DEEP_SLEEP_ENABLE,
60     .bluetooth.max_connections = MAX_CONNECTIONS,
61     .bluetooth.heap = bluetooth_stack_heap,
62     .bluetooth.sleep_clock_accuracy = 100, // ppm
63     .bluetooth.heap_size = sizeof(bluetooth_stack_heap),
64     .gattdb = &bg_gattdb_data,
65     #if (HAL_PA_ENABLE) && defined(FEATURE_PA_HIGH_POWER)
66     .pa.config_enable = 1, // Enable high power PA
67     .pa.input = GECKO_RADIO_PA_INPUT_VBAT, // Configure PA input to VBAT
68     #endif // (HAL_PA_ENABLE) && defined(FEATURE_PA_HIGH_POWER)
69 };
```



8-2-3 ペリフェラル設定を移植する

「SLSTK3401A\_gpio\_int\_pg1b」の**割込みハンドラ**と**GPIO 初期化関数**を、そのまま「SOC-iBeacon」の main.c にコピーします。場所は main()の前です。

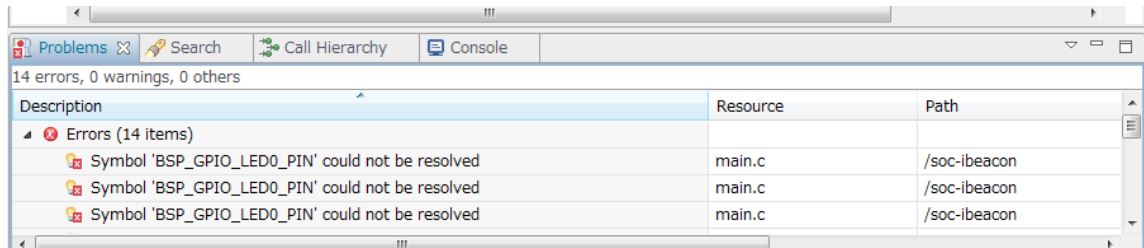
```

146 gecko_cmd_le_gap_set_advertise_timing(0, 160, 160, 0, 0);
147
148 /* Start advertising in user mode and enable connections */
149 gecko_cmd_le_gap_start_advertising(0, le_gap_user_data, le_gap_non_connectable);
150 }
151
152 /**
153  * @brief Main function
154  */
155 void main(void)
156 {
157     // Initialize device
158     initMcu();
159     // Initialize board
160     initBoard();
161     // Initialize application
162     initApp();
163
164     // Initialize stack
165     gecko_init(&config);
166
167     while (1) {
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000

```

← このあたりにコピー

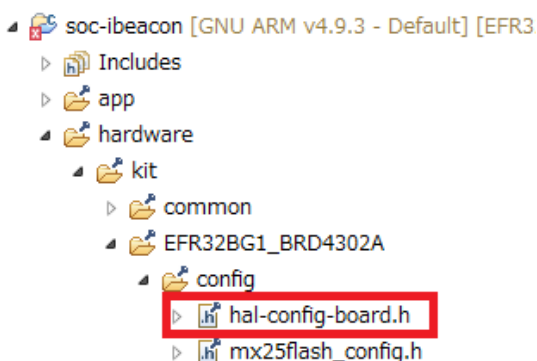
単にコピーしただけだと、画面下の Problem ウィンドウにエラーが多数出ます。



「SLSTK3401A\_gpio\_int\_pg1b」では、ボタンや LED のピン番号を直接指定せずに、変数を定義して (BSP\_GPIO\_PB0\_PORT や BSP\_GPIO\_PB0\_PIN など)、間接的に指定をしています。その変数と同じ名称の定義が「SOC-iBeacon」にないため、エラーが出ています。

BGM121 wireless starter kit を使用しますので、wireless starter kit のボタンや LED のピン番号を代わりに指定します。

BGM121 wireless starter kit にも、ボタンや LED のピン番号を変数定義したファイルがありますので、ご紹介します。hal-config-board.h というファイル内で定義されています。



## hal-config-board.h

```

40 // [BUTTON]
41 #define BSP_BUTTON_PRESENT          (1)
42
43 #define BSP_BUTTON0_PIN             (6U)
44 #define BSP_BUTTON0_PORT           (gpioPortF)
45
46 #define BSP_BUTTON1_PIN             (7U)
47 #define BSP_BUTTON1_PORT           (gpioPortF)
48
49 #define BSP_BUTTON_COUNT            (2U)
50 #define BSP_BUTTON_INIT             { { BSP_BUTTON0_PORT, BSP_BUTTON0_PIN }, { BSP_BUTTON1_PORT, BSP_BUTTON1_PIN } }
51 #define BSP_BUTTON_GPIO_DOUT        (HAL_GPIO_DOUT_LOW)
52 #define BSP_BUTTON_GPIO_MODE        (HAL_GPIO_MODE_INPUT)
53 // [BUTTON]$

157 // [LED]
158 #define BSP_LED_PRESENT              (1)
159
160 #define BSP_LED0_PIN                 (4U)
161 #define BSP_LED0_PORT                (gpioPortF)
162
163 #define BSP_LED1_PIN                 (5U)
164 #define BSP_LED1_PORT                (gpioPortF)
165
166 #define BSP_LED_COUNT                (2U)
167 #define BSP_LED_INIT                 { { BSP_LED0_PORT, BSP_LED0_PIN }, { BSP_LED1_PORT, BSP_LED1_PIN } }
168 // [LED]$
    
```

← ボタンのピン配置を定義

← LED のピン配置を定義

次に、定義した GPIO 初期化関数を、main()の中で呼び出します。

```

209 void main(void)
210 {
211     // Initialize device
212     initMcu();
213     // Initialize board
214     initBoard();
215     // Initialize application
216     initApp();
217
218     // Initialize stack
219     gecko_init(&config);
220
221     // 初期化 GPIO
222     initGPIO();
223
224     while (1) {
225         struct gecko_cmd_packet* evt;
226
227         // Check for stack event.
228         evt = gecko_wait_event();
229
230         // Run application and event handler.
231         switch (BGLIB_MSG_ID(evt->header)) {
232             // This boot event is generated when the system boots up after reset.
233             // Do not call any stack commands before receiving the boot event.
234             case gecko_evt_system_boot_id:
235                 // Initialize iBeacon ADV data
236                 bcnSetupAdvBeaconing();
237                 break;
238
239             default:
240                 break;
241         }
242     }
243 }

```

← GPIO 初期化関数

最終的に、このようなコードになりました。

```

137 /* Set 0 dBm Transmit Power */
138 gecko_cmd_system_set_tx_power(0);
139
140 /* Set custom advertising data */
141 gecko_cmd_le_gap_bt5_set_adv_data(0, 0, len, pData);
142
143 /* Set advertising parameters. 100ms advertisement interval.
144  * The first two parameters are minimum and maximum advertising interval,
145  * both in units of (milliseconds * 1.6). */
146 gecko_cmd_le_gap_set_advertise_timing(0, 160, 160, 0, 0);
147
148 /* Start advertising in user mode and enable connections */
149 gecko_cmd_le_gap_start_advertising(0, le_gap_user_data, le_gap_non_connectable);
150 }
151
152 // 追加 (SLSTK3401A_gpio_int_pg1b から)
153
154 /******//**
155  * @brief GPIO Even IRQ for pushbuttons on even-numbered pins
156  * *****/
157 void GPIO_EVENT_IRQHandler(void)
158 {

```

← 移植 ここから

```
159 // Clear all even pin interrupt flags
160 GPIO_IntClear(0x5555);
161
162 // Toggle LED0
163 GPIO_PinOutToggle(BSP_LED0_PORT, BSP_LED0_PIN);
164 }
165
166 Ⓞ /*****
167  * @brief GPIO Odd IRQ for pushbuttons on odd-numbered pins
168  *****/
169 Ⓞ void GPIO_ODD_IRQHandler(void)
170 {
171     // Clear all odd pin interrupt flags
172     GPIO_IntClear(0xAAAA);
173
174     // Toggle LED01
175     GPIO_PinOutToggle(BSP_LED1_PORT, BSP_LED1_PIN);
176 }
177
178 Ⓞ /*****
179  * @brief GPIO initialization
180  *****/
181 Ⓞ void initGPIO(void)
182 {
183     // Configure GPIO pins
184     CMU_ClockEnable(cmuClock_GPIO, true);
185
186     // Configure PB0 and PB1 as input with glitch filter enabled
187     GPIO_PinModeSet(BSP_BUTTON0_PORT, BSP_BUTTON0_PIN, gpioModeInputPullFilter, 1);
188     GPIO_PinModeSet(BSP_BUTTON1_PORT, BSP_BUTTON1_PIN, gpioModeInputPullFilter, 1);
189
190     // Configure LED0 and LED1 as output
191     GPIO_PinModeSet(BSP_LED0_PORT, BSP_LED0_PIN, gpioModePushPull, 0);
192     GPIO_PinModeSet(BSP_LED1_PORT, BSP_LED1_PIN, gpioModePushPull, 0);
193
194     // Enable IRQ for even numbered GPIO pins
195     NVIC_EnableIRQ(GPIO_EVEN_IRQn);
196
197     // Enable IRQ for odd numbered GPIO pins
198     NVIC_EnableIRQ(GPIO_ODD_IRQn);
199
200     // Enable falling-edge interrupts for PB pins
201     GPIO_IntConfig(BSP_BUTTON0_PORT, BSP_BUTTON0_PIN, 0, 1, true);
202     GPIO_IntConfig(BSP_BUTTON1_PORT, BSP_BUTTON1_PIN, 0, 1, true);
203 }
204
```



変数名を変更

```
205
206 /**
207  * @brief Main function
208  */
209 void main(void)
210 {
211     // Initialize device
212     initMcu();
213     // Initialize board
214     initBoard();
215     // Initialize application
216     initApp();
217
218     // Initialize stack
219     gecko_init(&config);
220
221     // 初期化 GPIO
222     initGPIO();
223
224     while (1) {
225         struct gecko_cmd_packet* evt;
226
227         // Check for stack event.
228         evt = gecko_wait_event();
229
230         // Run application and event handler.
231         switch (BGLIB_MSG_ID(evt->header)) {
232             // This boot event is generated when the system boots up after reset.
233             // Do not call any stack commands before receiving the boot event.
234             case gecko_evt_system_boot_id:
235                 // Initialize iBeacon ADV data
236                 bcnSetupAdvBeaconing();
237                 break;
238
239             default:
240                 break;
241         }
242     }
243 }
244
245 /** @} (end addtogroup app) */
246 /** @} (end addtogroup Application) */
```

 GPIO 初期化関数を実行

こうして実装したコードを、ビルドして実行すると、外部割込みを実装できたことが確認できます。

今回は例として外部割込みの実装手順を紹介しましたが、UG136 に割り込み実装についての注意点が書かれていますので、必ず参照ください。[\(リンク\)](#)

割り込み処理中は、他の処理が行えなくなります。通常のアプリ設計であれば、それでも良いですが、Bluetooth スタック動作も並行して行う必要がありますので、割り込みハンドラ内で重い処理を行うのは好ましくありません。割り込みハンドラ内では単にフラグを立てておき、通常ループ内で処理させるといったコーディングが求められます。

### 8-3 こんなサンプルコードはありませんか？

- SPP 通信 (SPP over BLE)
  - (1) [https://www.silabs.com/community/wireless/bluetooth/knowledge-base.entry.html/2017/04/13/spp-over-ble\\_c\\_examp-mnoe](https://www.silabs.com/community/wireless/bluetooth/knowledge-base.entry.html/2017/04/13/spp-over-ble_c_examp-mnoe)
  
- ホスト側のサンプルコード
  - (1) SPP 通信 (SPP over BLE) のサンプルコード (上記参照) に、サーバー・クライアント双方のサンプルコードが用意されています。
  - (2) Thermometer のサンプルコードが用意されています。  
[https://www.silabs.com/community/wireless/bluetooth/forum.topic.html/example\\_of\\_a\\_bgm121-Oc5c](https://www.silabs.com/community/wireless/bluetooth/forum.topic.html/example_of_a_bgm121-Oc5c)
  
- マルチマスタ/マルチスレーブ (Dual Topology) のサンプルコード
  - (1) [https://www.silabs.com/community/wireless/bluetooth/knowledge-base.topic.10.10.html/multi-slave\\_multi-ma-HjAO](https://www.silabs.com/community/wireless/bluetooth/knowledge-base.topic.10.10.html/multi-slave_multi-ma-HjAO)
  
- BGM1xx 間通信のサンプルコード
  - (1) SPP 通信 (SPP over BLE) のサンプルコード (上記参照) で、BGM1xx 間通信を行っています。
  
- ペリフェラル (ADC, I2C, SPI など) のサンプルコード
  - (1) EFM32PG 用のサンプルコードをご利用ください。手順は「8-2 サンプルコードにペリフェラルを実装してみる」をご参照ください。

#### 8-4 新バージョン SDK への移行手順

SDK 2.3～2.6 を他のバージョンに移行する場合には、下記を参照ください。

<マクニカオンラインサービス FAQ>

- [Bluetooth Smart SDK 2.3.x から 2.4.x へ移行する手順を教えてください](#)
- [Bluetooth Smart SDK 2.4.2 から 2.6.x へ移行する手順を教えてください](#)
- [Bluetooth Smart SDK 2.6.x から 2.7.x へ移行する手順を教えてください](#)

SDK 2.7～2.10 については、単に使用する SDK を選び直してください。

## 9 トラブルシューティング

[Macnica Online Service](#) で紹介した FAQ の中から、特に注意頂きたい点についてご紹介します。

### 9-1 動作障害

- **コードをダウンロードしましたが、正常動作しません (SDK 2.7 以降)**
  - SDK 2.62 以前と SDK 2.7 以降とでブートローダーの仕様が異なっており、SDK 2.7 以降を使用する場合にはブートローダーの書き換えが必要です。
  - 詳細: <https://service.macnica.co.jp/support/faq/126929>
- **NCP モード時の消費電流が高い**
  - NCP モード時は EM2 が無効になっています。Wakeup ピンを用意することで低消費電力化が可能です。
  - 詳細: <https://service.macnica.co.jp/support/faq/128533>

### 9-2 ツール障害

- **ラジオボードが認識されません (Simplicity Studio)**
  - 給電スイッチが AEM になっていますか？(本資料 7-4-3 参照) デバッグ経路が OUT になっていませんか？(本資料 6-1 参照)
- **デバイスにアクセスできなくなりました (Simplicity Studio)**
  - デバッグ経路が遮断(Lock)されると、デバイスへのアクセスが行えなくなります。Unlock することでアクセスできるようになります。
  - 詳細: <https://service.macnica.co.jp/support/faq/108137>



## 改版履歴

Version	改定日	改定内容
1.0	2017年01月	・新規作成。マクニカオンラインで公開
1.1	2017年03月	・C言語設計に関して追記。最新のSimplicity Studioに合わせて説明を一部変更
1.2	2017年05月	・ユーザ基板のプログラム・デバッグについて追記
1.3	2017年09月	・VCOMを利用したprintfデバッグについて追記
1.4	2018年02月	・Bluetooth SDK 2.7.0以降を対象に改版
1.5	2018年03月	・BGToolを使って評価する（NCPモード）について追記
1.6	2018年03月	・消費電流の測定、RF PHYの評価、ソフトウェア設計について追記
1.7	2018年06月	・新 SDK への移行、トラブルシューティングについて追記。最新のSimplicity Studioに合わせて説明を一部変更
1.8	2018年08月	・WSTK 制御ファームの更新方法。プロキシ設定をしてもインストール失敗する際の対策追加。
1.9	2018年10月	・printfデバッグの手順更新。BGM13S追加。

## 参考文献

- Silicon Labs 社 各種ドキュメント
- Silicon Labs 社 ナレッジベース、コミュニティフォーラム

### 免責、及び、ご利用上の注意

弊社より資料を入手されましたお客様におかれましては、下記の使用上の注意を一読いただいた上でご使用ください。

1. 本資料は非売品です。許可無く転売することや無断複製することを禁じます。
2. 本資料は予告なく変更することがあります。
3. 本資料の作成には万全を期していますが、万一ご不審な点や誤り、記載漏れなどお気づきの点がありましたら、弊社までご一報いただければ幸いです。
4. 本資料で取り扱っている回路、技術、プログラムに関して運用した結果の影響については、責任を負いかねますのであらかじめご了承ください。
5. 本資料は製品を利用する際の補助的なものとしてかかれたものです。製品をご使用になる場合は、メーカーリリースの資料もあわせてご利用ください。

本社

〒222-8561 横浜市港北区新横浜 1-6-3 TEL 045-470-9841 FAX 045-470-9844