Technical Note



Silicon Labs 社 BGM1xx アドバンストガイド(上級編)

2019 年 4 月





TecStar -

Silicon Labs 社 BGM1xx

アドバンストガイド(上級編)

目次	
1 はじめに	3
2 デバッグする	4
2-1 printf デバッグ	4
2-2 MCU デバッグ機能	7
2-3 ユーザ基板のデバッグ	9
2-3-1 デバッグ対象の切り替え	9
2-3-2 モジュール型番の指定	11
2-4 ユーザ基板との接続	12
3 機能・性能を評価する	15
3-1 RF PHY の特性を評価する	15
3-1-1 テストコマンドを使用する	15
3-1-2 BGToolを使用する	17
3-2 OTA update(over-the-air)を使用する	18
3-3 スループットを測定する	21
3-3-1 シリコンラボ社・サンプルコード (SDK 2.7~2.10)	21
3-3-2 マクニカ・サンプルコード (SDK 2.11)	22
4 消費電流を最適化する	24
4 消費電流を最適化する4-1 消費電流の簡易測定	24 24
 4 消費電流を最適化する 4-1 消費電流の簡易測定 5 ソフトウェア設計する 	24 24 28
 4 消費電流を最適化する 4-1 消費電流の簡易測定	24 24 28 28
 4 消費電流を最適化する 4-1 消費電流の簡易測定 5 ソフトウェア設計する 5-1 ソースコードの追い方 5-2 ペリフェラルの実装 (外部割込み) 	24 24 28 28 28 29
 4 消費電流を最適化する 4-1 消費電流の簡易測定	24 24 28 28 28 29 30
 4 消費電流を最適化する	24 24 28 28 28 29 30 33
 4 消費電流を最適化する 4-1 消費電流の簡易測定	24 24 28 28 28 29 30 33 37
 4 消費電流を最適化する 4-1 消費電流の簡易測定	24 24 28 28 29 30 33 37 42
 4 消費電流を最適化する 4-1 消費電流の簡易測定	24 24 28 28 28 29 30 33 37 42 42
 4 消費電流を最適化する 4-1 消費電流の簡易測定	24 28 28 28 28 29 30 33 37 42 43
 4 消費電流を最適化する 4-1 消費電流の簡易測定	24 24 28 28 28 29 30 33 37 42 42 43 44
 4 消費電流を最適化する 4-1 消費電流の簡易測定	24 24 28 28 28 29 30 33 37 42 42 43 44 46
 4 消費電流を最適化する	24 28 28 28 28 29 30 33 37 42 42 42 43 44 46 47
 4 消費電流を最適化する	24 28 28 28 28 29 30 33 37 42 42 43 44 44 46 47 47
 4 消費電流を最適化する	24 28 28 28 28 29 30 33 37 42 42 43 44 44 46 47 47 47

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 の使用方 法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。

本資料は、基本的な使用方法は理解頂いている方を対象としたアドバンストガイド(上級編)です。初め て BGM をご使用になる方は、まずクイックスタートガイド(初級編)をご参照ください。

2 デバッグする

2-1 printf デバッグ

BGM1xx のソフトウェア開発では、break point や step 実行を活用したデバッグを行うこともできますが、 Bluetooth 通信中に MCU を停止させてしまうと通信が切れてしまいますので、printf デバッグが非常に効 果的です。

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。ここでは、"SOC – Thermometer"を使用します。

② プロジェクトの hardware/kit/common/drivers に、retargetio.c, retargetserial.c, retargetserial.h が含ま れているかを確認します。

含まれていなければ、、これらのファイルをプロジェクトにコピー(ドラッグアンドドロップ)します。これらの ファイルは、"STUDIO_SDK_LOC¥hardware¥kit¥common¥drivers"にあります。



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 の順で確認できます。

🎦 Project Explorer 🔀	📄 🔄 🗢 🔽 🗖 🚺 🚯 soc-thermome	eter.isc 📝 main.c 🛛				
Soc-thermometer [CNU ARM v7.2.	1 - Default] [BGM13 ▲ 1⊕ /******* 2 * @file	***************************************				
Properties for soc-thermom	eter					
P ⊂ type filter text Linked Resources ↔ ▼						
A Resource Path Variables Linked Resources						
Resource Filters	Resource Filters Path variables specify locations in the file system, including other path variables with the syntax "\${VAR}".					
Builders	The locations of linked resources may	be specified relative to these path variables.				
▷ III ▷ C/C++ Build	▷ ▷ ▷ ○					
▷ C/C++ General						
Refactoring History	listory ECLIPSE_HOME C:¥SiliconLabs¥SimplicityStudio¥v4¥					
Run/Debug Settings	PARENT_LOC	C:¥Users¥ ¥SimplicityStudio¥v4_workspace	Loren			
	PROJECT_LOC	C:¥Users¥ .¥SimplicityStudio¥v4_workspace¥soc-thermometer	Remove			
⊳ .c	STUDIO_SDK_LOC	C:¥SiliconLabs¥SimplicityStudio¥v4¥developer¥sdks¥gecko_sdk_suite¥v2.5				
d. d	STUDIO_TOOLCHAIN_LOC	C:¥SiliconLabs¥SimplicityStudio¥v4¥developer¥toolchains¥gnu_arm¥7.2_201				

③ main.c に、stdio.h と retargetserial.h を include します。

```
<記述>
```

#include "stdio.h"
#include "retargetserial.h"

④ 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 initApp() の直後に入れてみます。

<記述>

RETARGET_SerialInit();

145⊝	<pre>int main(void)</pre>
146	{
147	<pre>// Initialize device</pre>
148	initMcu();
149	<pre>// Initialize board</pre>
150	initBoard();
151	<pre>// Initialize application</pre>
152	<pre>initApp();</pre>
155 154 155 156	<pre>/* For printf() */ RETARGET_SerialInit();</pre>
157 158	<pre>// Initialize stack gecko_init(&config);</pre>

⑤ hal-config.h を開き、HAL_VCOM_ENABLE の値を 1 に変更します。

21	<pre>#include "board_features.h"</pre>	
22	<pre>#include "hal-config-board.h"</pre>	
23	<pre>#include "hal-config-app-common.h"</pre>	
24		
25	<pre>#ifndef HAL_VCOM_ENABLE</pre>	
26	#define HAL_VCOM_ENABLE	(1)
27	#endif	
28		

⑥ 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。

<記述>

TecStar =

printf("Now Initializing...¥r¥n");



⑦ ビルドし、BGM にダウンロードします。

⑧ ターミナルソフト(Tera Term など)を使用し、シリアルポート(JLink CDC)をオープンします。UART の 設定は、ボーレート 115200, data 8bit, non parity, 1 stop bit としてください。

⑨ メインボードをリセットすると、ターミナルソフト上に文字が表示されます。



hal-config.h が含まれていない古い SDK をご使用の場合には、幾つかの追加手順が必要です。詳しく はクイックスタートガイド v1.9 をご参照ください。

また、シリコンラボ社のナレッジベースにも情報がございますので、こちらも参照ください。(リンク)

2-2 MCU デバッグ機能

TecStar =

プロジェクトをビルドした後、Flash Programmer ではなく Debug アイコンを使ってダウンロードすると、ブレ ークポイント、ステップ実行などのソフトウェア・デバッグの機能がご使用になれます。



ビルド用の画面(Simplicity IDE)と、デバッグ用の画面(Debug)は右上のアイコンで切り替えます。



◆ コードの実行・停止

コードの実行には Resume ボタン、停止には Suspend ボタンを使用します。

i 🗙 🕪 III 🙌 🌫	ര ¢i⇒ 🕹 •	× 🕨 💷	N 3. 73. 18 i⇒ 🕹 -
上一一 実行(Rea	sume)		停止(Suspend)
◆ ハードウェア・リセット			
MCU にハードウェア・リセットを	かけます。		
	🔌 🕩 II 🕅 🕉	. ∿. ¢ i⇒	.
	リセット(Rese	et the device)	_
◆ デバッグ経路の切断			
デバッグモード経路を切断して、	デバッグ用の画面を	終了します。ビ	ルド用の画面に切り替わります。



◆ ブレークポイント

ブレークポイントを設定するには、停止させたい行の左横をダブルクリックします。設定されると、水色の 小さな〇印が表示されます。再度ダブルクリックすれば解除されます。



TecStar

◆ ステップ実行

各種ステップ実行に対応しています。



実機で実際に動作を見て頂くのが、判りやすいです。



◆ レジスタ値の閲覧・変更

レジスタ・変数の閲覧や変更は、下のウィンドウ(Register ウィンドウなど)で行うことができます。前回の 停止から、値が変化した場合には黄色で表示されます。

∝)= Variables	⁰ Breakp	IIII Registers 🛛	& Express	- E			
	·		* = = 1	ľĽí			
Name	Va	alue (Description		w Variables	Brook	
High General		C	General Purpose Registe	ers	(x)= Valiables	• Бтеакр	
▷ 🛗 DMA		[MA				
AES		A	ES		Name		
⊳ 👬 USB		L	JSB		4 🛗 General		
⊳ 👬 MSC		N	1SC		1919 RO		
EMU		E	MU		1919 R1		
RMU		F	MU		1919 R2		
⊳ 👬 CMU		C	MU		1919 R3		
ILESENSE		L	ESENSE		1919 R4		
▷ ₩ RTC		F	tτC		1919 R5		
LETIMERO		L	ETIMER0		1919 R6		
▷ 👬 EBI		E	BI		1919 R7		
▷ M USART0		ι	JSART0		1919 R8		

*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。

2-3 ユーザ基板のデバッグ

メインボードを使用することで、ラジオボード上の BGM1xx だけでなく、ユーザ基板上の BGM1xx に対し てもプログラミングやデバッグを行うことが可能です。メインボードとユーザ基板の接続には、Simplicity Debug Adaptor Board (SLSDA001A)が便利です。



Simplicity Debug Adapter Board (SLSDA001A)

2-3-1 デバッグ対象の切り替え

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



この Debug Mode は、MCU、OUT、IN、OFF の4設定があり、ラジオボードをデバッグする際には MCU を、ユーザ基板をデバッグする際には OUT を選択します。





Change をクリックし、Adapter Configuration タブの Debug Mode で、OUT を選択します。

💥 J-Link Silicon Labs (440090374)	• 🔀
Configuration of device: J-Link Silicon Labs (440090374)	
Application images Device hardware J-Link Configuration Scratchpad Packet Trace Adapter Configuration CTU	INE
Update Adapter Installation Package Debug Mode: MCU V OFF MCU IN	:kage
OK Cancel	

デバッグ対象がラジオボードから外部(ユーザ基板)に切り替わると、メインボード右下の DEBUG OUT という LED が点灯します。



2-3-2 モジュール型番の指定

ラジオボードを使用している場合には、使用しているモジュール型番を自動で認識してくれます。しか し、ユーザ基板上のモジュール型番は自動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 でモジュ ール型番(BGM13S など)を選択し、OK をクリックします。以下は注意点です。

- Target part を選んだ場合には、ラジオボード用のデモは表示されません。
- Boards に使用しないボードが登録されている場合には、×をクリックして消去してください。
- Boards でリストアップされる候補が少ない場合には、Target part を None にしてお試しください。

All detected inform Parts:	nation:				
Parts:					
Boards:					
Wireless Starte	r Kit Mainboard (BRD	4001A Rev A01)			
V					
Target part:					
Search					-
None					
_					
Boards:					
bgm13s					•
BGM13S32	netooth Module Rad	io Board (BRD4305A)			
BGM13S32 E	luetooth Module Rad	io Board (BRD4305A F	Rev A02)		
BGM13S22 E	luetooth Module Rad	io Board (BRD4305C)			
BGM13S22 E	luetooth Module Red	io Board (BRD4305C F	Rev A02)		
BGM13S22 E	luetooth Module And	io Board (BRD4305C F	Rev A02)		

登録が完了すると、Device Adapter タブに指定したモジュール(或いはラジオボード)が追加されます。 あとは、Debug Mode = MCU 時のラジオボードと同じ要領で、ご使用頂けます。下図は、Target part で BGM13S を選択した場合です。

Debug Adapters % X (2) % X (2) <th>J-Link Silicon Labs (Preferred SDK: Gecko SDK Suite v2.5.2: Bluetooth Click <u>here</u> to change the preferred SDK.</th>	J-Link Silicon Labs (Preferred SDK: Gecko SDK Suite v2.5.2: Bluetooth Click <u>here</u> to change the preferred SDK.
	Debug Mode: OUT <u>Change</u>

TecStar

2-4 ユーザ基板との接続

メインボードとユーザ基板の接続方法については、AN958「Debugging and Programming Interfaces for Custom Designs」の中で詳しく説明されています。(リンク)

何通りか方法はありますが、Mini Simplicity Connectorを介して接続するのがお勧めです。

メインボードの右下に、In/Out Debug Header と Simplicity Connector の 2 つのコネクタがあります。



In/Out Debug Header は、arm CPU 用の標準的なデバッグインタフェースです。ダウンロードやデバッグ に必要な SWCLK、SWDIO などが割り当てられています。

一方、Simplicity Connector には、printf デバッグなどで使用する仮想 COM ポートや、パケットトレース 機能が割り当てられています。

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



Simplicity Debug Adaptor Board 上にも 10 ピンコネクタが 3 つ並んでいますが、中央の Mini Simplicity Connector に接続します。2 つのコネクタは、ダウンロードに必要な SW インタフェース、printf デバッグに必要な VCOM などを含む 10 ピンに変換されます。



Figure 4.2. Mini Simplicity Connector Pin-Out

Pin #	Pin Name	Pin Function	EFR32 Functionality
1	VAEM	Target Advanced Energy Monitor Voltage Net	VDD
2	GND	Target Ground	VSS
3	RST	Target Reset (Active Low)	RESETn
4	VCOM_RX	Target Pass-through UART/Virtual COM Port Receive	US0_RX
5	VCOM_TX	Target Pass-through UART/Virtual COM Port Transmit	US0_TX
6	SWO	Target Serial Wire Output	SWO
7	SWDIO	Target Serial Wire Data Input/Output	SWDIO
8	SWCLK	Target Serial Wire Clock	SWCLK
9	PTI_FRAME	Target Packet Trace Interface Frame Signal	FRC_DFRAME
10	PTI_DATA	Target Packet Trace Interface Data Signal	FRC_DOUT

TecStar _____

信号名称	補足コメント
VAEM	メインボードとユーザ基板とで信号レベルが異なることを想定し、信号線にはレベルシフタが入っています。そのレベルシ
	フタに、ユーザ基板の信号レベル(電源電圧)を伝えるためのピンです。通常はユーザ基板の電源に接続します。
GND	メインボードとユーザ基板の GND を同一レベルにするためのピンです。ユーザ基板の GND に接続してください。
RST	リセットピンです。デバッグ経路が遮断されてしまった場合の復旧手段として使用します。BGM1xxのRESETnに接続し
	てください。
VCOM_RX	メインボードには USB-UART 変換機能が付いており、その UART が VCOM ピンから外部に出ています。 printf デバッグ
	や、NCP モードの評価に使用します。BGM1xx の UART ピン(RX)に接続してください。
VCOM_TX	メインボードには USB-UART 変換機能が付いており、その UART が VCOM ピンから外部に出ています。 printf デバッグ
	や、NCP モードの評価に使用します。BGM1xx の UART ピン(TX)に接続してください。
SWO	デバッグ用のインタフェースです。Energy Profiler でコード連携機能を使用する場合には、BGM1xxのSWOに接続してく
	ださい。使用は必須ではありません。
SWDIO	ダウンロード・デバッグ用のインタフェースです。BGM1xx の SWDIO に接続してください。
SWCLK	ダウンロード・デバッグ用のインタフェースです。BGM1xx の SWCLK に接続してください。
PTI_FRAME	パケットトレース機能を使用する際に接続ください。
PTI_DATA	パケットトレース機能を使用する際に接続ください。

3 機能・性能を評価する

3-1 RF PHY の特性を評価する

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。ここでは手順を 簡単にご紹介します。

詳しくはドキュメントが用意されておりますので、AN1046をご覧ください(リンク)。本章で扱わない DTM (direct test mode)を使用した手順も紹介されています。

3-1-1 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・マニュアルに記載されています。

SILICON LABS docs.silabs.com	You are viewing documentation for version: 2.11 (latest) 2.10 Version History					
Bluetooth						
Getting Started Overview	test cor	test commands				
API Reference coex cte-receiver cte-transmitter dfu	test_dtm_end This command can be used to end a transmitter or a receiver test. When the command is processed by the radio and the test has ended, a test_dtm_completed event is triggered. Command					
flash >	Byte	Туре	Name	Description		
gatt >	0	0x20	hilen	Message type: Command		
gatt-server >	1	0x00	lolen	Minimum payload length		
hardware >	2	0x0e	class	Message class:testing commands		
le-connection >	3	0x02	method	Message ID		
le-gap > sm >	Response	Response				
system	Byte	Туре	Name	Description		
teet	0	0x20	hilen	Message type: Response		
Commanda	1	0x02	lolen	Minimum payload length		
Commands 2	2	0x0e	class	Message class:testing commands		
Events 2	3	0x02	method	Message ID		
Enumerations	4-5	UINTIG	result	Command result		
user >	API					
Error Codes						

1例として送信コマンドをご紹介します。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。

使用するチャネルや 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	Туре	Name	Description
0	0x20	hilen	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
			Range: 0-255
6	uint8	channel	Bluetooth channel
			Range: 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	0000000 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

[●] 送信コマンド(cmd_test_dtm_tx)

3-1-2 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 も使用できます。BGTool を使用するための手順についてはクイックスタートガイドをご参照ください。 BGTool で 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。

🛥 BGTool		- • •
File Kit Connection	View Help	
SILICON LABS	S	
Interactive view	RF regulatory test view	
Connection	Connect with: JLink CDC UART Port (COM6) Baud Rate: 115200	
	Open	Cancel

GUI 上で各種設定を行うことができます。必要な設定を行い、Start test で動作開始します。

RF test mode for regulatory testing
Bluetooth Low Energy Transmit power:
• Low energy transmit ● PRBS9 (GFSK) ● 10101010 ● 10101010 ● 10101010 ● 00000000 ● パケットタイプの指定 ● 0000111 ● 01010101 ● PN9 continuously modulated output ● Unmodulated carrier は無変調 Packet length:
○Low energy receive RX テストを実施 Stop test Start test

3-2 OTA update (over-the-air)を使用する

"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。

soc-smartPhone.isc 🔀	
Bluetooth SDK, version:2.8.0.0	
A General	
BLE GATT Configurator	
Source filters	Custom BLE GATT
SIG Silicon Labs	Generic Access
Profiles Convices Characteristics Descriptors	C Device Name
Promes Services Characteristics Descriptors	Appearance
type filter text	S Device Information
P Alert Notification	Manufacturer Name String
P Automation IO	Generation Set
P Blood Pressure	C Temperature Measurement
P Continuous Glucose Monitoring	C Temperature Type
Cycling Power	Intermediate Temperature
P Cycling Speed and Cadence	C Measurement Interval
P Environmental Sensing	▲ S Immediate Alert
Find Me	Alert Level
▷ P Glucose	S Silicon Labs OTA
Health Thermometer	Silicon Labs OTA Control
P Heart Rate	

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⊐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)

••••• au ᅙ	17:08	@ 🕇 🎖 42% 🔲
<	BG20085	OTA
SERVICES		
Device Informa 180A UUID	ation ~	
Health Thermo 1809 UUID	ometer ~	
Immediate Ale 1802 UUID	rt ~	
OTA Service	-4FA1-BFA4-8F47B	42119F0

TSC-01019

⑤ APP の CHOOSE FILE ボタンを押します。(右図)

TecStar —



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

•••• a	au ᅙ 17:08	@ 🖇 42% 📼 >	••••• au 🖘	17:08	@ \$ 42% 💷 🕞	•••• au ᅙ	17:08	@ 🖇 42% 💷
		OTA	場所		キャンセル	<	BG20085	OTA
SER\	/ICES			2 このフォルダ内を検索	する	SERVICE	S	
Devid	ce Information 🗸		< 戻る	ΟΤΑ		Device In	formation ~	
180 UUI	BG20085 OTA Progress Pa AA78F028-8D9E-4F6C-89D4-E306	acket Size 180 SF3D8918E	applica	ation.gbl		180 BG20 UUI OTA AA78	0085 • Progress Pa F028-8D9E-4F6C-89D4-E306	cket Size 180 F3D8918E
He 180 UUI	METHOD Partial	Full				He 180 MET UUI	HOD Partial	Full
Im: 180 001	app CH	IOOSE FILE	[APP 180 UUI	Inbox/applicatio	n.gbl 📀
iClo Droj	ud pbox					OT 1D ¹ UUID	CANCEL OTA	UPDATE
その	他							

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)

••••• au 穼	17:10	• 1	∦ 40% 💶						
Bluetooth Beaconing									
iBeacon	IMMEDIA	ate 💿	-40 RSSI						

3-3 スループットを測定する

通信性能を測る指標の一つにスループットがあります。使用する PHY を 2M に設定したからといって、 2Mbps のスループットが得られる訳ではありません。様々な要因によって、スループットは低下します。

BLE におけるスループットの考え方については、シリコンラボ社のナレッジベースに参考になる情報があります。(リンク:スループットについて)。コネクション・インターバルや MTU サイズ、ATT 動作種別などがスループットに影響します。

スループットを測定するには、測定したい条件に合せてユーザコードをご設計頂く必要があります。 BGM⇔BGM 間のスループット測定については、サンプルコードをご用意していますので、こちらをベース にして、ご使用条件に合せてカスタマイズするのが容易です。

本章では、用意されているサンプルコードについてご紹介します。

3-3-1 シリコンラボ社・サンプルコード (SDK 2.7~2.10)

シリコンラボ社のナレッジベースに、サンプルコードと実装方法が用意されています。(<u>リンク:サンプ</u> <u>ルコード</u>)

実装の流れは以下の通りです。

- SOC-Empty プロジェクトを作成する
- サンプルコード(gatt.xml)をプロジェクトにインポート/Generate する
- LCD インタフェースを追加する
- main.cをサンプルコード(main_2_7_0.c)に差し替える
- プロジェクトをビルドし、2 つの BGM1xx にダウンロードする

基本的な使用方法は、次章のマクニカ・サンプルコードと同じです。測定したスループットは、スタータ ーキット上の LCD に表示されます。

3-3-2 マクニカ・サンプルコード (SDK 2.11)

SDK 2.7 と 2.11 とで、main.c の実装が少し変更されていますので、シリコンラボ社・サンプルコードに 一部手を加えたものをご用意しています。BGM13P および BGM13S 向けです。

主な変更点は以下の通りです。

- SDK 2.11 への対応
- Coded-PHY S2(500kbps)を追加
- 測定したスループットを UART に出力

プロジェクトをビルドし、2つの BGM13P/S にダウンロードしてください。手っ取り早く評価したい場合には、GNUフォルダの下にhexファイルが入っていますので、こちらを使用ください。その後の評価手順は以下の通りです。

- 基板1にて、PB0を押したままリセットボタンを押す(または電源をONにする)と、Materモードで 起動して Virtual COM に次のメッセージを出力します。
 Booted as Master
- 基板2にて、何も押さずにリセットボタンを押す(または電源をONにする)と、Slaveモードで起動 して Virtual COM に次のメッセージを出力します。
 Booted as Slave
- 3. Master と Slave の両方を起動すると、自動的に接続して Master と Slave の双方の Virtual COM に次のメッセージを出力します。

Connected Phy: 1M

- 4. Slave の PB0 を押している間 Notification の通信を行い、PB0 を放すと Slave の LCD と Virtual COM に転送レートを出力します。 Throughput: 710400 bps (1M/Notification)
- 5. SlaveのPB1を押している間 Indicationの通信を行い、PB1を放すとSlaveのLCDとVirtual COM に転送レートを出力します。 Throughput: 19519 bps (1M/Indication)
- Master の PB0 を押すと Phy を切り替えることができます。 Phy は PB0 を押すごとに、 1M → 2M → S8 (125k) → S2 (500k) → 1M の順に切り替わり、 Virtual COM に切り替え後の Phyを出力します。 以下は Master の Virtual COM の出力例です。

Phy: 1M

Phy: 2M Phy: S8 Phy: S2 Phy: 1M

7. Phy を切り替えながら Notification と Indication を測定すると、Slave の Virtual COM に次のよう に出力されます。

Phy: 1M Throughput: 710400 bps (1M/Notification) Throughput: 19519 bps (1M/Indication) Phy: 2M Throughput: 1250657 bps (2M/Notification) Throughput: 38648 bps (2M/Indication) Phy: S8 Throughput: 94848 bps (S8/Notification) Throughput: 4684 bps (S8/Indication) Phy: S2 Throughput: 333696 bps (S2/Notification) Throughput: 4684 bps (S2/Indication)

8. LCDの表示には時間が掛かるため、LCDの表示が更新されたことを確認してから次の操作をしてください。特に Phy の切り替え(S8 や S2 への切り替え)には時間が掛かります。Master と Slave の両方が切り替わったことを確認してからスループットを測定してください。

4 消費電流を最適化する

4-1 消費電流の簡易測定

Wireless Starter Kit main board (以後、WSTK)と Simplicity Studioを使用することで、簡単に消費電流 を測定することができます。

メインボードには電流計が実装されています。USB 給電(5V)から LDO で 3.3V を生成し、その 3.3V (VMCU)に流れる電流を測定しています。ですから、VMCU を BGM1xx への給電に限定するか、あるい は周辺回路への給電にも利用するか、によって測定対象が変わってきます。



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、ラジオボードを使用して評価する手順を紹介しますが、ユーザ基板への給電にVMCUを使用 すれば、ユーザ基板の消費電流を測定することができます。

手順は以下の通りです。

- ① メインボードにラジオボードを挿入し、PC に接続します。給電スイッチは AEM に設定ください。
- ② "SOC Smart Phone App"デモをダウンロードします。
- ③ Compatible Tools タブから Energy Profiler を選択します。

Gettin	ig Started	Documentation	Compat	ible Tools	Resources					
To view all available tools, click the 'Tools' button in the main toolbar.								Add/Remove Tools		
Capacitive Sense Profiler Capacitive Sense Profiler (Capacitive Sense Profiler is a tool that supports real-time available of concerned as form supports real-time			e Graph	Wicrium µC/Probe Graphica Live Watch Tool - Allows you to create a dashbaar for your embedded application. Supports all Console for interaction with				the remote device		
analysis of sensor data from supported parts.			EFM32	EFM32, EFR32, and EZR32 devices.						
Network Analyzer				A Ucation Builder			Migrate Projects			
Tools for capturing and analyzing network activity			Create	Create an embedded software framework application			Migrate v3 projects			
(*) I	3G Tool		A	Energy Profile	er		Flash Programmer			
Launch B	G Tool to interact	with Blue Gecko Module or So	C The Si quickly applica consur	mplicity Profiler i v visualize the en- itions and perforr nption	s a tool developed to ergy consumption in t m optimizations to rec	let developers heir duce power	Flash Programmer is a utilit	y to flash or erase any part		

④ Profiler ⇒ Start Energy Capture を選択します。Quick Access から選択しても良いです。



⑤ 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 をクリック。

🛃 Start Energy Profiler Captu	re	×					
Please select one or more devices that supports energy profiling.							
▲							
		Select All					
		Select None					
Remember my decision	V						
	ОК Са	ncel					



⑥ 電流測定がスタートします。



⑦ Paused ボタンを押し、測定をいったん停止します。

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。



⑧ Running ボタンを押し、測定を再開します。

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。



TecStar

5 ソフトウェア設計する

ソフトウェア設計に役立つ情報をご紹介します。

5-1 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



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。

5-2 ペリフェラルの実装(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 どかと思いますので、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(5-2-1、5-2-2)

● 「SLSTK3401A_gpio_int_pg1b」のペリフェラル設定を、「SOC – iBeacon」に移植する(5-2-3) です。

5-2-1 サンプルコードを理解する (SLSTK3401A_gpio_int_pg1b)

TecStar —

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。

🗀 My Products	Ċ	÷	×	8	Ē	Ŧ	
EFM32 Pea							
 Kits EFM32 Pearl Gecko Starter Kit (SLSTK Boards EFM32 Pearl Gecko Starter Kit board (Kit) 	3401A BRD25) 00A)				

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

Getting Started	Documentation	Compatible Tools	Resources		
My Favorite Documents -+ 🕑			ocuments		-+6
		AN Thi ger	0012: General Purpo is application note desc neral-purpose input/ou	se Input Output rribes usage of the EFM32 tput (GPIO) subsystem. This	∎ ☆

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られない場合には、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。

🕶 Simplicity		×
Application Notes		
Open an application note	ペリフェラル名や機能名を入力	
Name	(interrupt など)	
type filter text		
AN0011: I2C Master and Slave Operation		*
AN0012: General Purpose Input Output		_
AN0013: Direct Memory Access		
AN0014: EFM32 Timers		
AN0015.0: EFM32 and EZR31 Wireless MCU Series	0 Watchdog	
AN0015.1: EFM32 and EFR32 Series 1 Watchdog		
AN0016: Oscillator Design Considerations		_
AMOD17-1Example LLADT		
This application note describes usage of the EFM32 general pin values, peripheral function routing, external interrupt c projects that illustrate these concepts can be run on the Sta document Source files (zip) Example C source code Multiple	I-purpose input/output (GPIO) subsystem. This document discusses configuration, read and wr apability, and use of GPIO pins as producers for the Peripheral Reflex System (PRS). Example arter Kit boards for many different EFM32 derivatives. This application note includes: This PDF e IDE projects	iting
Filter by selected product line		
? Import Project	Open Folder Open Close	

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。

13	Project Selection							×
	Choose a project							
	SLSTK3400A_gpio_conf.slsproj							^
	SLSTK3400A_gpio_int.slsproj							
	SLSTK3400A_gpio_periph.slsproj							=
	SLSTK3400A_gpio_prs.slsproj							
	SLSTK3401A_gpio_conf_pg1b.slsproj	_						
	SLSTK3401A_gpio_int_pg1b.slsproj							
	SLSTK3401A_gpio_periph_pg1b.slsproj							
	SLSTK3401A_gpio_prs_pg1b.slsproj	1						
	SLSTK3401A_slew_rate_pg1b.slsproj							
	SLSTK3402A_gpio_conf_pg12b.slsproj	- \						
	SLSTK3402A_gpio_int_pg12b.slsproj							
	SLSTK3402A_gpio_periph_pg12b.slsproj							
	SLSTK3402A_gpio_prs_pg12b.slsproj							-
			V					
	٢			01/		0	_	
	(!)			UK		Cancel		

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。



TSC-01019





5-2-2 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。

m	nain.c	
76 *	*/	と日本たちも
77 vo	oid bcnSetupAdvBeaconing(void)	谷関数や変数の正義
79⊖ `	/* This function sets up a custom advertisement package acco	ording to iBeacon specifications.
80	* The advertisement package is 30 bytes long. See the iBea	con specification for further details.
81	*/	
83	<pre>static struct {</pre>	
84	<pre>uint8_t flagsLen; /* Length of the Flags field. */</pre>	
85	<pre>uint8_t flagsType; /* Type of the Flags field. */</pre>	
86	uint8_t flags; /* Flags field. */	Hold */
88	uint8 t mandataType: /* Type of the Manufacturer Data fi	eld. */
89	<pre>uint8_t compId[2]; /* Company ID field. */</pre>	
90	<pre>uint8_t beacType[2]; /* Beacon Type field. */</pre>	
	Ŏ	
152 🤅	/**	
153	* @brief Main function	
154	*/	nain 閏数
156		
157	// Initialize device	
158	<pre>initMcu();</pre>	
159	// Initialize board	
161	// Initialize application	
162	initApp(); 初期化局	月数
163		
164	// Initialize stack	
165	gecko_init(&config);	
167	while (1) {	
168	<pre>struct gecko_cmd_packet* evt;</pre>	
169		
170	// Check for stack event.	
172	eve = geeko_ware_event();	
173	<pre>// Run application and event handler.</pre>	
174	<pre>switch (BGLIB_MSG_ID(evt->header)) {</pre>	C 1
175	// This boot event is generated when the system boot // Do not call any stack commands before receiving t	s up atter reset. he hoot event
177	case gecko evt system boot id:	ne boot event.
178	// Initialize iBeacon ADV data	
179	<pre>bcnSetupAdvBeaconing();</pre>	
180	break;	
182	default:	
183	break;	
184	}	
185	}	
185	1	
1889	<pre>/** @} (end addtogroup app) */</pre>	
189 /	<pre>/** @} (end addtogroup Application) */</pre>	
190		

TecStar

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
     // Set up DC-DC converter
39
40
     EMU_DCDCInit_TypeDef dcdcInit = BSP_DCDC_INIT;
    #if HAL_DCDC_BYPASS
41
42
     dcdcInit.dcdcMode = emuDcdcMode_Bypass;
43 #endif
44
     EMU_DCDCInit(&dcdcInit);
45
46
     // Set up clocks
47
    initMcu_clocks();
48
    RTCC_Init_TypeDef rtccInit = RTCC_INIT_DEFAULT;
49
50 rtccInit.enable
                                  = true;
   rtccInit.debugRun
51
                                  = false;
52
    rtccInit.precntWrapOnCCV0
                                  = false;
                                 = false;
   rtccInit.cntWrapOnCCV1
53
54
   rtccInit.prescMode
                                  = rtccCntTickPresc;
55
    rtccInit.presc
                                   = rtccCntPresc_1;
   rtccInit.enaOSCFailDetect
56
                                  = false;
57
    rtccInit.cntMode
                                   = rtccCntModeNormal;
58
    RTCC Init(&rtccInit);
59
60 #if defined(_EMU_CMD_EM01VSCALE0_MASK)
   // Set up EM0, EM1 energy mode configuration
61
62
    EMU_EM01Init_TypeDef em01Init = EMU_EM01INIT_DEFAULT;
   EMU EM01Init(&em01Init);
63
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;
   EMU_EM23Init(&em23init);
70
71 #endif //_EMU_CTRL_EM23VSCALE_MASK
72
     TEMPDRV_Init();
73
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。

TecStar

• 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);
    // Enable clock for PRS
33
34
     CMU_ClockEnable(cmuClock_PRS, true);
35 #ifdef FEATURE_EXP_HEADER_USART3
     // Enable clock for USART3
36
37
    CMU_ClockEnable(cmuClock_USART3, true);
38 #else
    // Enable clock for USART0
39
40
    CMU_ClockEnable(cmuClock_USART0, true);
41 #endif
42
     // Enable GPIO clock source
43
    CMU_ClockEnable(cmuClock_GPIO, true);
44
     // Put the SPI flash into Deep Power Down mode for those radio boards where it is available
45
46
    MX25_init();
47
    MX25_DP();
     // We must disable SPI communication
48
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)
     // Enable VCOM if requested
31
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
     I2CSPM_Init_TypeDef i2cInit = I2CSPM_INIT_DEFAULT;
37
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

TecStar

```
51 #ifndef MAX CONNECTIONS
   #define MAX_CONNECTIONS 4
52
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 = {
    .config_flags = 0,
.sleep.flags = SLEEP_FLAGS_DEEP_SLEEP_ENABLE,
58
59
60
     .bluetooth.max_connections = MAX_CONNECTIONS,
61
     .bluetooth.heap = bluetooth_stack_heap,
     .bluetooth.sleep_clock_accuracy = 100, // ppm
62
63
     .bluetooth.heap_size = sizeof(bluetooth_stack_heap),
   .gattdb = &bg_gattdb_data,
#if (HAL_PA_ENABLE) && defined(FEATURE_PA_HIGH_POWER)
64
65
66
    .pa.config_enable = 1, // Enable high power
67
     .pa.input = GECKO_RADIO_PA_INPUT_VBAT, // Configure PA input to VBAT
   #endif // (HAL_PA_ENABLE) && defined(FEATURE_PA_HIGH_POWER)
68
69 };
```

TecStar

5-2-3 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 /* Start advertising in user mode and enable connections */ 148 149 gecko cmd le gap start advertising(0, *le gap user data*, *le gap non connectable*); 150 } このあたりにコピー 151 152 /** * @brief Main function 153 42 */ 154 43 155 void main(void) 44 void GPIO_EVEN_IRQHandler(void) 156 { 45 { // Clear all even pin interrupt flags 157 // Initialize device 46 GPI0_IntClear(0x5555); 47 158 initMcu(); 48 // Initialize board 159 49 // Toggle LED0 initBoard(); GPI0_PinOutToggle(BSP_GPI0_LED0_PORT, BSP_GPI0_LED0_PIN); 160 50 51 } 161 // Initialize application 52 initApp(); 162 53 /** ******* 163 * @brief GPIO Odd IRQ for pushbuttons on odd-numbered pins 54 * ********** 164 // Initialize stack 55 56 void GPIO_ODD_IRQHandler(void) 165 gecko_init(&config); 57 { 166 58 // Clear all odd pin interrupt flags while (1) J 167 59 GPI0_IntClear(0xAAAA); 60 Toggle LED01 61 GPI0_PinOutToggle(BSP_GPI0_LED0_PORT, BSP_GPI0_LED0_PIN); 62 63 } 64 66 * @brief GPIO initialization 67 68 void initGPIO(void) 69 { 70 // Configure GPIO pins 71 CMU_ClockEnable(cmuClock_GPI0, true); 72 // Configure PB0 and PB1 as input with glitch filter enabled GPI0_PinModeSet(BSP_GPI0_PB0_PORT, BSP_GPI0_PB0_PIN, gpioModeInputPullFilter, 1); GPI0_PinModeSet(BSP_GPI0_PB1_PORT, BSP_GPI0_PB1_PIN, gpioModeInputPullFilter, 1); 73 74 75 76 // Configure LED0 and LED1 as output
GPI0_PinModeSet(BSP_GPI0_LED0_PORT, BSP_GPI0_LED0_PIN, gpioModePushPull, 0); 77 78 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(GPI0_ODD_IRQn); 85 86 // Enable falling-edge interrupts for PB pins GPI0_IntConfig(BSP_GPI0_PB0_PORT, BSP_GPI0_PB0_PIN, 0, 1, true); 87 88 GPI0_IntConfig(BSP_GPI0_PB1_PORT, BSP_GPI0_PB1_PIN, 0, 1, true); } 89 90 **a**1

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。

•							۲.		
🛐 Problems 🔀 🛷	Search	😫 Call Hierarchy	📮 Console				\bigtriangledown		3
14 errors, 0 warnings	s, 0 others								
Description		*			Resource	Path		į.	*
4 🔇 Errors (14 iter	ms)								-
😘 Symbol 'BSP_GPIO_LED0_PIN' could not be resolved				main.c	/soc-ibeacon				
9 Symbol 'BSP_GPIO_LED0_PIN' could not be resolved			main.c	/soc-ibeacon					
😘 Symbol 'B	SP_GPIO_LE	ED0_PIN' could not be	resolved		main.c	/soc-ibeacon			÷
								•	

「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

TecStar

(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。



次に、定義した GPIO 初期化関数を、main()の中で呼び出します。

TecStar

```
209 void main(void)
210 {
211
      // Initialize device
      initMcu();
212
      // Initialize board
213
214
      initBoard();
215
      // Initialize application
216
      initApp();
217
218
      // Initialize stack
219
      gecko_init(&config);
221
       // 初期化 GPIO
                                            GPIO 初期化関数
222
      initGPIO();
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.
        switch (BGLIB_MSG_ID(evt->header)) {
231
          // This boot event is generated when the system boots up after reset.
232
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
        3
242
      }
243 }
```

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
        <sup>e</sup> The first two parameters are minimum and maximum advertising interval,
      * both in units of (milliseconds * 1.6). */
145
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
                                                移植 ここから
152 // 追加(SLSTK3401A gpio int pg1b から)
153
155
     * @brief GPIO Even IRQ for pushbuttons on even-numbered pins
                                                        .
**************************/
                                  ------
156
157 void GPIO_EVEN_IRQHandler(void)
158 {
```

TecStar

```
// Clear all even pin interrupt flags
159
   GPI0_IntClear(0x5555);
160
161
     // Toggle LED0
162
    GPIO PinOutToggle(BSP_LED0_PORT, BSP_LED0_PIN);
163
164 }
165
*******************//**
   * @brief GPIO Odd IRQ for pushbuttons on odd-numbered
167
                                                    pins
   ******
                                                         ************************/
168
169 void GPIO ODD IRQHandler(void)
170 {
     // Clear all odd pin interrupt flags
171
172
    GPIO IntClear(0xAAAA);
                                                          変数名を変更
173
174
     // Toggle LED01
   GPI0_PinOutToggle(BSP_LED1_PORT, BSP_LED1 PIN);
175
176 }
177
179 * @brief GPIO initialization
                             180
181<sup>©</sup> void initGPIO(void)
182 {
183
     // Configure GPIO pins
184
     CMU ClockEnable(cmuClock GPIO, true);
185
     // Configure PB0 and PB1 as input with glitch filter enabled
186
     GPIO_PinModeSet(BSP_BUTTON0_PORT, BSP_BUTTON0_PIN, gpioModeInputPullFilter, 1);
187
     GPIO_PinModeSet(BSP_BUTTON1_PORT, BSP_BUTTON1_PIN, gpioModeInputPullFilter, 1);
188
189
     // Configure LED0 and LED1 as output
190
     GPIO PinModeSet(BSP_LED0_PORT, BSP_LED0_PIN, gpioModePushPull, 0);
191
     GPI0_PinModeSet(BSP_LED1_PORT, BSP_LED1_PIN, gpioModePushPull, 0);
192
193
194
     // Enable IRQ for even numbered GPIO pins
195
     NVIC_EnableIRQ(GPIO_EVEN_IRQn);
196
     // Enable IRQ for odd numbered GPIO pins
197
198
     NVIC_EnableIRQ(GPIO_ODD_IRQn);
199
     // Enable falling-edge interrupts for PB pins
200
     GPIO IntConfig(BSP BUTTON0 PORT, BSP BUTTON0 PIN, 0, 1, true);
201
     GPIO_IntConfig(BSP_BUTTON1_PORT, BSP_BUTTON1_PIN, 0, 1, true);
202
203 }
204
```

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

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。

5-3 ブートローダーの作成

ブートローダーは、BGM 起動後に最初に動作するアプリケーションです。ブートローダーで必要な処理 を行い、それからユーザアプリケーションが起動する、という仕組みになっています。

Silicon Labs 社が提供するブートローダーには Gecko Bootloader(新しい)と legacy bootloader(古い)の 2 種があり、SDK 2.7 以降のアプリケーションは Gecko Bootloader 上でのみ動作するようになっています。 詳細は、UG266「Silicon Labs Gecko Bootloader User's Guide」(リンク)をご覧ください。 ここでは Gecko Bootloader に限定して説明します。

5-3-1 ブートローダーの配置アドレス

Gecko Bootloader は 1st ステージブートローダーとメインブートローダーの2 段構成になっていて、ま ずは 1st ステージブートローダーが起動し、処理終了後にメインブートローダーが起動します。その後、 ユーザアプリケーションが起動する、という流れです。

下図は、BGM1xxのメモリマップ上のどこにブートローダーおよびアプリケーションが配置されるかを示したものです。



5-3-2 生成した HEX に含まれる範囲

ブートローダーをビルドすると、

• xxx.hex

TecStar

- xxx.s37
- xxx-combined.s37

という HEX ファイルが生成されます。(.hex はインテル HEX、.s37 はモトローラ HEX です) .bin も生成されますが、基本的にはアドレス情報が入った HEX ファイルを使用します。

末尾に-combined が付いたファイルには、1st ステージブートローダーとメインブートローダーが含まれています。-combined が付いていないファイルには、メインブートローダーのみが含まれています。 また、Bluetooth SDK に含まれるCサンプルコードにはブートローダーが含まれず、デモには含まれて

います。図示すると以下の通りです。



5-3-3 ブートローダーの作成手順

ブートローダーの作成手順は以下の通りです。

① New Project から Gecko Bootloader を選択します。

Applications	
Select an application type	
Select an application you are building:	
🚷 Bluetooth Mesh SDK	-
Bluetooth SDK 2.11.2.0	E
Gecko Bootloader 1.8.2, Gecko Bootloader 1.7.0, Silicon Labs Bootloader 0.0.0, Gecko Bootloader 1.6.0, Gecko Boot Gecko Bootloader 1.4.1, Gecko Bootloader 1.5.0, Gecko Bootloader 1.4.0 (9 stacks) Silicon Labs Flex SDK Flex SDK 2.1.0.0, Flex SDK 2.0.1.0, Flex SDK 2.5.1.0, Flex SDK 2.3.1.0 (4 stacks)	tloader 0.0.0,
Show internal demo stacks	Manage SDKs
? < Back Next > Finish	Cancel

② 複数の Gecko Bootloader がインストールされている場合には、使用するバージョンを指定します。

③ 多数のサンプルプロジェクトが表示されるので、ベースにするプロジェクトを選択します。使用する デバイスやプロトコルによって、適切なプロジェクトが違ってきます。

	tion
Select either	a blank application or a sample application.
ample Applica	lion
Internal S Application images red supports sl configurati storage sh	torage Bootloader (single image on 1MB device) Bootloader for EFR32xG12, EFR32xG21 and EFM32GG1x devices, using the internal flash memory to store upgrade elved by the application in an application specific way, such as OTA, USB, Ethernet, etc. This sample configuration oring a single firmware update image at a time, and is plug-and-play compatible with the "Local Storage Bootloader" on of Ember2Net, Silicon Labs Thread and Flex sample applications on 1024 kB EFR32xG12 devices. The layout of the ould be modified before being run on other devices.
Application by the app single firm EmberZNe modified b	torage Bootloader (single image on SL2KB device) Bootloader for all EFR32 and EFM32 Series 1 devices, using the internal flash memory to store upgrade images receive lication in an application specific way, such as OTA, USB, Ethernet, etc. This sample configuration supports storing a ware update image at a time, and is plug-and-play compatible with the "Local Storage Bootloader" configuration of t, Silicon Labs Thread and Fiex sample applications on S12 kB EFR32xG13 devices. The layout of the storage should be efore being run on other devices.
Internal S Application images rec supports sl configurati storage sh	torage Bootloader (single image with LZMA compression, 1MB flash) Bootloader for EFR32xG12, EFR32xG21 and EFM32GG1x devices, using the internal flash memory to store upgrade elved by the application in an application specific way, such as OTA, USB, Ethernet, etc. This sample configuration coring a single firmware update image at a time, and is plug-and-play compatible with the "Local Storage Bootloader" an of EmberZNet, Silicon Labs Thread and Flex sample applications on 1024 kB EFR32xG12 devices. The layout of the ould be modified before being run on other devices.
	in-place OTA DFU Bootloader
Sluetooth Application application	Bootloader for in-place OTA DFU with the Bluetooth stack on EFR32 devices with 256kB flash parts. Thie sample is using internal flash to store stack and app upgrade images received OTA in a two-stage process.

標準的なブートローダーは以下の通りです。

モジュール	動作モード	プロジェクト名
BGM111, 113, 12x, 11S (EFR32BG1-based)	SOC	Bluetooth in-place OTA DFU Bootloader
BGM13P, 13S (EFR32BG13-based)	SOC	Internal Storage Bootloader (single image on 512kB device)
BGM1xx	NCP	BGAPI UART DFU Bootloader

④ プロジェクト名、作業フォルダ、使用コンパイラなどを指定して、プロジェクトを生成します。

⑤ 必要に応じて、各種設定を行ってください。詳しくは UG266 を参照ください。設定が終わったら Generate ボタンを押し、ソースコードに反映します。

🐊 bootloader-storage-internal-single-512k_2.isc 🔀		
Gecko Bootloader, version:1.8.2	Generate	« Preview
🚓 General 👍 Plugins 👍 Storage 🛱 Callbacks 歳 Other		
- Application configuration		
Generation directory: Relative to ISC file C:¥Users¥10649¥SimplicityStudio¥v4_workspace¥bootloader-storage-inter	nal-single-512k_2	
Select architecture for this application: Board: BGM13522 Bluetooth Module Radio Board (BRD4305C)		

⑥ C 言語のサンプルコードと同様に、ビルドして、ダウンロードします。xxx.hex/.s37 と xxx-combined.s37 とで含まれる範囲が異なることにご注意ください。

5-4 新しい SDK への移行

SDK 2.3~2.6を次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
- <u>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</u>
- 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 2.7~2.11 については、単に使用する SDK を選び直してください。

6 FAQ

BGM について、よく頂く質問をまとめました。

6-1 開発環境・ツール

- オフラインインストーラがアップデートされたことを知る方法はありますか?
 - シリコンラボ社のナレッジベースに最新バージョンの情報が掲載されていますので、定期的に こちらをチェックください。(リンク)
- コードをダウンロードするための簡易ツールはありますか?
 - スタンドアロンで使用できるツールとして Simplicity Commander が提供されています。(リンク)
 Simplicity Studio に搭載されている Simplicity Commander と同じものです。
- 量産工程で使用できるようなライターやプログラマはありますか?
 - パッケージが特殊なので、オンボード書き込みが現実的です。J-Link や、Computex 社 FP-10
 (リンク)などがご使用頂けます。

6-2 トラブルシューティング

- コードをダウンロードしましたが、main に飛びません(動作しません)
 - ブートローダーが書かれていない可能性があります。ブートローダーをビルドしてダウンロード するか、或いはデモ(ブートローダーを含んでいる)をダウンロードしてください。
 - SDK 2.62 以前から SDK 2.7 移行に変更した場合には、ブートローダーの書き換えが必要です。
 ブートローダーの仕様が変更になっています。
 https://garriag.magning.ag.ip/guppart/fog/126020

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。
 https://service.macnica.co.jp/support/fag/128533

nttps://service.macnica.co.jp/support/faq/12853

- SOC モード時の消費電流が高い
 - Simplicity Studio の Debug モードで使用していないでしょうか? Debug モードで使用する際には、クロックの都合上 EM2 に移行しないようになっています。 Simplicity Studio との通信を切断してからお試しください。
- ラジオボードが認識されません (Simplicity Studio)
 - 給電スイッチが AEM になっていないか、デバッグ経路が OUT になっていないか、お確かめく ださい。

- 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。
 https://service.macnica.co.jp/support/fag/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
2.0	2019年03月	・クイックスタートガイドとアドバンストガイドに分割
2.1	2019年04月	・スループット測定を追記。FAQ 更新

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