Bluetooth® Low Energy プロトコルスタック

Fast Prototyping Board ホストサンプル

要旨

本アプリケーションノートでは、RL78/G14 Fast Prototyping Board と RL78/G1D BLE Module Expansion Board を使用してセンサ(Renesas HS3001 Humidity and Temperature Sensor)情報を Bluetooth 通信で送信する方法を説明します。

ホストサンプルは、Bluetooth Low Energy プロトコルスタックの Modem 構成を使用した RL78/G14 Fast Prototyping Board で動作する Host MCU プログラムで、UART 2 線分岐接続方式で接続した RL78/G1D BLE Module Expansion Board 上の RL78/G1D モジュール(RY7011)を制御します。また、Security Library を使用することでペアリングやペアリング情報の保存を行うことができます。

本書では、ホストサンプルが動作するハードウェア構成、ソフトウェア構成、スマートフォンと通信の確認をする手順や Bluetooth の通信シーケンスについても記載します。

動作確認デバイス

RL78/G14 Fast Prototyping Board (製品型名:RTK5RLG140C00000BJ)

RL78/G1D BLE Module Expansion Board (製品型名: RTKYRLG1D0B00000BJ)

関連資料

資料名	資料番号	
	和文	英文
Bluetooth Low Energy プロトコルスタック		
ユーザーズマニュアル	R01UW0095J	R01UW0095E
API リファレンスマニュアル 基本編	R01UW0088J	R01UW0088E
サンプルプログラムアプリケーションノート	R01AN1375J	R01AN1375E
rBLE コマンド仕様書	R01AN1376J	R01AN1376E
BLE 仮想 UART アプリケーション	R01AN3130J	R01AN3130E
Security Library	R01AN3777J	R01AN3777E
RL78/G1D		
ユーザーズマニュアル　ハードウェア編	R01UH0515J	R01UH0515E
RL78/G1D モジュール		
ユーザーズマニュアル　ハードウェア編	R02UH0004J	R02UH0004E
ファームウェア ユーザーズマニュアル	R01UW0160J	R01UW0160E
モジュール制御ソフトウェア	R01AN3362J	R01AN3362E
RL78/G14		
ユーザーズマニュアル　ハードウェア編	R01UH0186J	R01UH0186E

目次

1. 概要

本アプリケーションノートでは、RL78/G14 Fast Prototyping Board と RL78/G1D BLE Module Expansion Board を使用してセンサ情報を Bluetooth 通信で送信する方法を説明します。センサは Renesas HS3001 Humidity and Temperature Sensor Module (湿度、温度)を使用します。

ホストサンプルは、Bluetooth Low Energy プロトコルスタック(BLE プロトコルスタック)の Modem 構成を使用した RL78/G14 Fast Prototyping Board で動作する Host MCU プログラムです。UART 2 線分岐接続方式[注1]で接続した BLE MCU の RL78/G1D BLE Module Expansion Board を構成する RL78/G1D モジュール(RY7011)を制御します。また、Security Library [注2]を使用することでペアリングの実施やペアリング情報を RL78/G14 Fast Prototyping Board に保存することができます。

RY7011 には出荷時に動作確認用のモジュールファームウェアが書き込まれており、Host MCU プログラムから BLE プロトコルスタックの rBLE API [注3]を用いて制御することができます。

【注】 1. UART のデータ信号線である TxD、RxD に加えて、Host MCU がデータ送信時に BLE MCU を起床させるための WAKEUP 信号線を持ちます。WAKEUP 信号線は Host MCU の TxD を分岐して BLE MCU の WAKEUP と接続します。Host MCU と BLE MCU 間の接続については、「6.4.3 応用回路」参照してください。
2. Security Library の詳細については、「Bluetooth® Low Energy プロトコルスタック Security Library」(R01AN3777)を参照してください。
3. BLE プロトコルスタック API の詳細については、「Bluetooth® Low Energy プロトコルスタック API リファレンスマニュアル 基本編」(R01UW0088)を参照してください。

2. 開発環境

ホストサンプルのビルドと動作確認で使用する開発環境を示します。

2.1 ハードウェア環境

— ホストマシン

- PC/AT™ 互換機
- プロセッサ　：1GHz 以上（ハイパースレッディング，マルチコア CPU に対応）
- メイン・メモリ　：推奨 2G バイト以上
- ディスプレイ　：1024×768 以上の解像度，65536 色以上
- インタフェース　：USB 2.0（RL78/G14 Fast Prototyping Board との接続）

— 開発ボード

- RL78/G14 Fast Prototyping Board (RTK5RLG140C00000BJ)
- RL78/G1D BLE Module Expansion Board (RTKYRLG1D0B00000BJ)
- Renesas HS3001 Humidity and Temperature Sensor Module with I2C Interface

— スマートフォン(タブレット)

- Android デバイス、または iOS デバイス

2.2 ソフトウェア環境

— OS

- Windows 7 以降

— 統合環境/コンパイラ

下記のいずれかの統合環境とコンパイラの組み合わせを使用してください。

- CS+ for CC V8.05.00 / CC-RL V1.10.00
- e2 studio 2021-01 (64 ビット版) / CC-RL V1.10.00
- e^2 studio V7.8.0 (32 ビット版) / CC-RL V1.10.00

3. ホストサンプル構成

3.1 デバイス構成

「図 3-1　デバイス構成」に本アプリケーションノートで使用するデバイス構成図を示します。

Local Device(Peripheral)は、Host MCU にあたる RL78/G14 Fast Prototyping Board と、BLE MCU にあたる RL78/G1D BLE Module Expansion Board で構成されます。2 つのボードは PMOD1 で接続され、UART 2 線分岐式接続で通信します。また、RL78/G1D BLE Module からセンサデータを送信するために、RL78/G14 Fast Prototyping Board の PMOD2 注 1 に Renesas HS3001 Humidity and Temperature Sensor Module（湿度、温度）を接続します。PMOD2 は IICA で通信します。

Remote Device(Central)は、Android デバイスまたは iOS デバイスのスマートフォンを使用します。

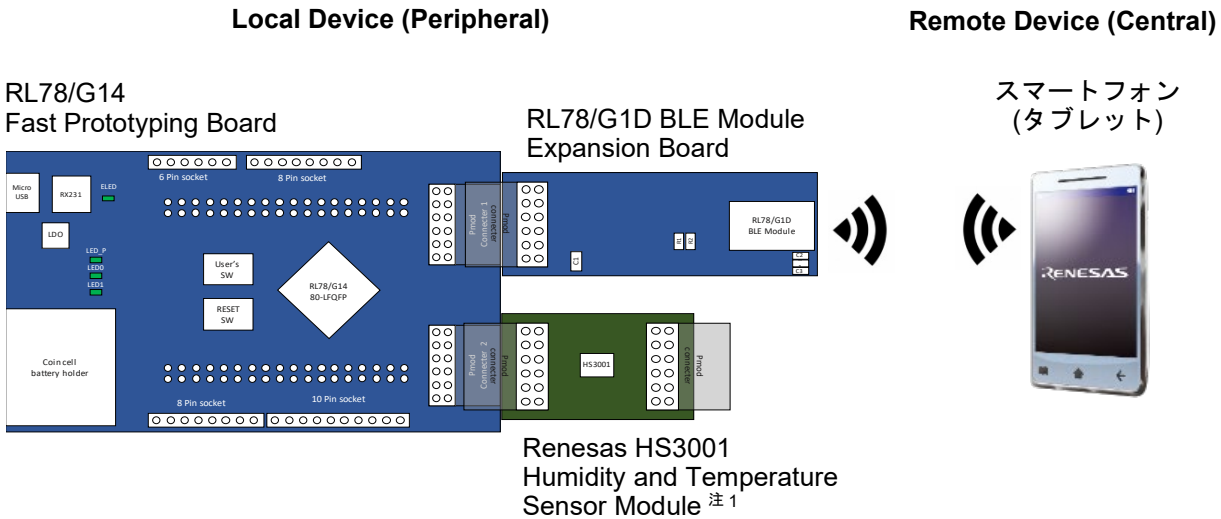

図 3-1　デバイス構成

【注】 1. I2C 通信方式の Pmod™ モジュールを使用する際は PMOD2 に接続します。そして RL78/G14 の周辺 I/O リダイレクト機能でシリアル・インタフェース IICA の機能を割り当てるピンを変更します。ピンの割り当ては「表 3-2　PMOD2 コネクタピンアサイン」を参照してください。

ホストサンプルの動作概要は以下のとおりです。

- rBLE API を使用し以下の動作を実行。
  - プログラム実行後、自動的にアドバタイジングを開始。
  - Remote Device からの接続要求により接続を実行。
  - Remote Device から要求があれば、ペアリング/暗号化を実行し接続。
  - Remote Device からの Indication 許可により、
    汎用双方向通信(General Purpose Communication Profile : GPCP)を有効化。
  - Remote Device からの測定開始ワード('S'または's')により、
    センサで測定した結果を BLE で繰り返し送信。
  - USER SW でデータ・フラッシュに保存されたセキュリティ情報を消去。
- プログラム実行のための簡易的なスケジューラのみ実装。
- 実行すべき処理がない期間は、RL78/G14 を STOP モードに遷移。
- 対向デバイスは、スマートフォン（Android デバイスまたは iOS デバイス）を想定。

3.1.1　Pmod™ インタフェース

「図 3-1　デバイス構成」の Pmod™ インタフェース・コネクタピンアサインを示します。各ピンで使用している機能を青字で示します。

(1)　PMOD1

表 3-1　PMOD1 コネクタピンアサイン

RL78/G14 Fast Prototyping Board		Pmod™	RL78/G1D BLE Module Expansion Board	
機能名	ピン番号	ピン番号	ピン番号	機能名
P74/KR4/INTP8	33	1	2	P30/**INTP3**/RTC1HZ
P51/INTP2/SO00/**TxD0**/TOOLTxD/TRGIOB	42	2	8	P11/SI00/**RxD0**/TOOLRxD/SDA00/(TI06)/(TO06)
P50/INTP1/SI00/**RxD0**/TOOLRxD/SDA00/TRGIOA/(TRJO0)	41	3	7	P12/SO00/**TxD0**/TOOLTxD/(TI05)/(TO05)
P30/INTP3/RTC1HZ/SCK00/SCL00/TRJO0	40	4	9	P10/SCK00/SCL00/(TI07)/(TO07)
GND	-	5	-	GND
VCC	-	6	-	V_{DD}
P140/PCLBUZ0/INTP6	2	7	-	N.C
P130	72	8	24	**RESET#**
P147/ANI18/VCOUT1	58	9	-	N.C
P146	57	10	23	P40/TOOL0
GND	-	11	-	GND
VCC	-	12	-	V_{DD}

(2)　PMOD2

表 3-2　PMOD2 コネクタピンアサイン

RL78/G14 Fast Prototyping Board		Pmod™	Renesas HS3001 Humidity and Temperature Sensor Module
機能名	ピン番号	ピン番号	機能名
P16/TI01/TO01/INTP5/TRDIOC0/IVREF0/(SI00)/(RxD0)	48	1	NC
P13/TxD2/SO20/TRDIOA1/IVCMP1	51	2	NC
(注 1)P14/RxD2/SI20/SDA20/TRDIOD0/(**SCLA0**)	50	3	**SCL**
(注 1)P15/SCK20/SCL20/TRDIOB0/(**SDAA0**)	49	4	**SDA**
GND	-	5	VSS
VCC	-	6	VDD
P141/PCLBUZ1/INTP7	1	7	NC
P110/(INTP11)	55	8	NC
P17/TI02/TO02/TRDIOA0/TRDCLK/IVCMP0/(SO00)/(TxD0)	47	9	NC
P111	56	10	NC
GND	-	11	VSS
VCC	-	12	VDD

【注】　1. 周辺 I/O リダイレクト機能で、RL78/G14 の機能割り当てを変更しています。I2C 通信方式の Pmod™ モジュールを使用する際は、PMOD2 に接続してください。

3.1.2　ユーザスイッチ(SW_USR)

RL78/G14 データ・フラッシュのセキュリティ情報を消去します。

セキュリティ情報の詳細については、「Bluetooth® Low Energy プロトコルスタック Security Library」(R01AN3777)を参照してください。

3.2　ソフトウェア構成

Host MCU である RL78/G14 と、BLE MCU である RY7011 のソフトウェア構成図を示します。RY7011 には出荷時に動作確認用ファームウェアが書き込まれており複数のプロファイルに対応しています。本書では汎用双方向通信(General Purpose Communication Profile : GPCP)を使用します。その他のプロファイルについては、「RL78/G1D モジュール ファームウェア ユーザーズマニュアル」(R01UW0160)の「7. プロファイル」を参照してください。

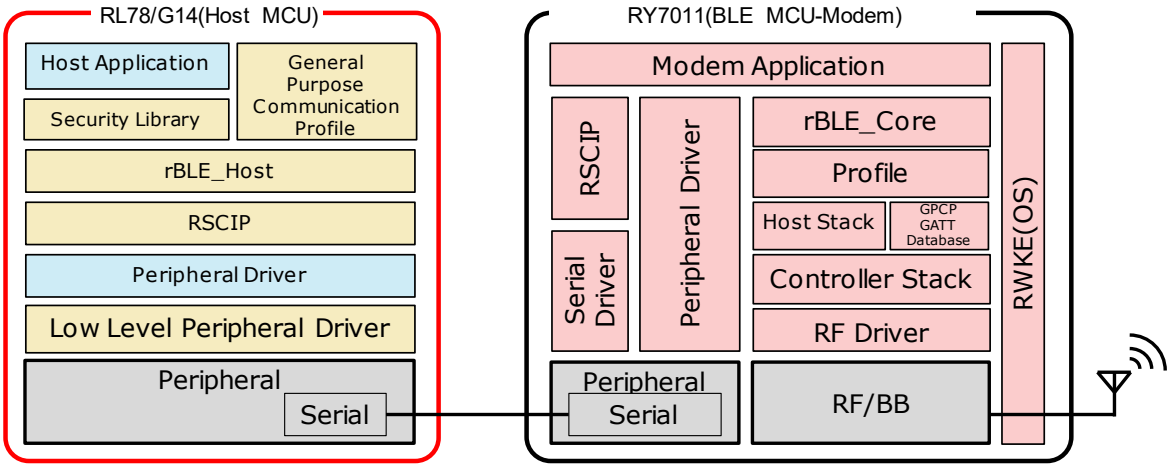

図 3-2　ソフトウェア構成

Host MCU は、MCU 周辺機能の制御と BLE MCU との通信を実行するための低レベル周辺ドライバ、周辺ドライバ、RSCIP（Renesas Serial Communication Interface Protocol）と、rBLE API をアプリケーションに提供するための rBLE_Host と、システムを制御するためのホストアプリケーション、GATT API を使用した General Purpose Communication Profile(GPCP)、ペアリング等のセキュリティ機能を提供する Security Library で構成されます。

低レベル周辺ドライバは、コード生成ツールが自動生成します。RSCIP、rBLE_Host は BLE プロトコルスタックに含まれており、ソースコードが提供されます。ソフトウェア開発時は、BLE プロトコルスタックが提供する最新のソースコードをご使用ください。

表 3-3　Host MCU ソフトウェア構成

ソフトウェア	機能	ソフトウェア開発
Host Application	rBLE の初期化 rBLE コマンドの実行スケジューリング rBLE イベントコールバックの登録	コーディングが必要
Security Library	ペアリング、ボンディング 暗号化、プライバシー	コーディングが不要 (ソースコード提供) 注 1
General Purpose Communication Profile (GPCP)	GATT API を使用した独自プロファイル	コーディングが不要 (ソースコード提供) 注 1
rBLE_Host	rBLE API 提供 イベントコールバックの実行	コーディングが不要 (ソースコード提供) 注 1
RSCIP	シリアル通信プロトコルの制御	コーディングが不要 (ソースコード提供) 注 1
Peripheral Driver	Host MCU 周辺機能の制御	コーディングが必要
Low Level Peripheral Driver	Host MCU 周辺機能のプリミティブな制御	コーディングが不要 (ツール自動生成) 注 2

【注】 1. ソフトウェア開発用コードファイルは BLE プロトコルスタックが提供。
2. ソフトウェア開発用コードファイルはコード生成ツールが自動生成。

BLE MCU は、RF/BB を制御するための RF ドライバ、Sample Custom Profile の GATT Database、Host/Controller スタック、Profile、rBLE_Core と、Host MCU と通信するためのシリアル通信ドライバ、RSCIP と、システムを制御するための RWKE（Renesas Wireless Kernel Extension）、Modem アプリケーションで構成されます。これらは「RL78/G1D モジュール モジュール制御ソフトウェア」(R01AN3362)でビルド環境が提供されます。

表 3-4　BLE MCU ソフトウェア構成

ソフトウェア	機能
Modem Application	RSCIP と rBLE の制御
RWKE	システム全体のスケジューリングとメモリ資源の管理
RSCIP	シリアル通信プロトコルの制御
Peripheral Driver/Serial Driver	BLE MCU 周辺機能の制御
rBLE_Core	rBLE_API 提供
Profile	プロファイル機能の提供
Host Stack	GAP、GATT、SM、L2CAP 機能の提供
GPCP GATT Database	General Purpose Communication Profile の GATT Database
Controller Stack	LL 機能の提供

3.3　周辺機能構成

RL78/G14 Fast Prototyping Board で使用する RL78/G14 の周辺機能を以下に示します。

表 3-5　周辺機能

周辺機能	用途	必要性(注)
端子割り当て設定 (周辺 I/O リダイレクト機能)	RL78/G14 Fast Prototyping Board と Renesas HS3001 Humidity and Temperature Sensor Module の IICA 通信で使用します。 RL78/G14 のリセット時、IICA0 端子は SCLA0=P60、SDAA0=P61 に割り当てられています。RL78/G14 Fast Prototyping Board の PMOD2 I/F に周辺 I/O リダイレクト機能で SCLA0=P14、SDAA0=P15 として割り当てることにより IICA0 で通信できるようになります。	任意
クロック発生回路	RL78/G14 の動作周波数設定に使用します	必須
ポート	P74: RL78/G1D BLE Module Expansion Board との UART 2 線分岐式接続の WAKEUP 端子として使用します。 P130: RL78/G1D BLE Module Expansion Board のリセット解除端子として使用します。	必須
	P43: RL78/G14 Fast Prototyping Board の LED0 で使用します。 P44: RL78/G14 Fast Prototyping Board の LED1 で使用します。	任意
シリアル-UART0	RL78/G14 Fast Prototyping Board と RL78/G1D BLE Module Expansion Board の UART 通信に使用します。	必須
シリアル-IICA0	RL78/G14 Fast Prototyping Board と Renesas HS3001 Humidity and Temperature Sensor Module の IICA 通信に使用します。	任意
タイマ-TAU0	Renesas HS3001 Humidity and Temperature Sensor Module の A/D 変換時間のウエイトで使用します。	任意
12 ビット・インターバル・タイマ	rBLE のタイマ機能に使用します。	必須
割り込み機能-INTP0	ユーザスイッチ(SW_USR)の割り込みに使用します。	任意

【注】 Host MCU が rBLE を使用するために最低限必要とする周辺機能を「必須」、その他を「任意」とします。

表 3-6　周辺機能設定

周辺機能	設定項目	設定
端子割り当て設定 (周辺 I/O リダイレクト機能)	PIOR2 ビット	1 を設定 (P14 に SCLA0 を割り当て) (P15 に SDAA0 を割り当て)
クロック発生回路	動作モード	高速メインモード 2.7(V)≦V_{DD}≦5.5(V)
	EV_{DD}	1.8(V)≦EV_{DD}≦5.5(V)
	高速オンチップオシレータクロック	64(MHz)
	低速内蔵発振クロック	15kHz
	RTC,インターバル・タイマ動作クロック	15(f_{IL})kHz
	CPU と周辺クロック	32000(f_{IH})kHz
ポート	P43	出力 (RL78/G14 Fast Prototyping Board LED0)
	P44	出力 (RL78/G14 Fast Prototyping Board LED1)
	P74	出力 (RL78/G1D BLE Module Expansion Board WAKEUP 端子)
	P130	出力 (RL78/G1D BLE Module Expansion Board RESET 端子)
シリアル SAU0 - UART0 - 受信	データ・ビット長	8 ビット
	データ転送方向	LSB
	パリティ	パリティなし
	ストップ・ビット長	1 ビット固定
	受信データ・レベル	標準
	転送レート	115200(bps)
	受信完了割り込み(INTSR0)	高
	コールバック機能	受信完了 エラー
シリアル SAU0 - UART0 - 送信	転送モード	単発モード
	データ・ビット長	8 ビット
	データ転送方向	LSB
	パリティ	パリティなし
	ストップ・ビット長	1 ビット
	送信データ・レベル	標準
	転送レート	115200(bps)

	送信完了割り込み(INTST0)	低
	コールバック機能	送信完了
シリアル IICA0 - シングルマスタ	カウント・クロック	f_{CLK}/2
	自局アドレス	16
	動作モード	標準
	転送クロック	100000(bps)
	通信完了割り込み (INTIICA0)	低
	コールバック機能	マスタ送信完了 マスタ受信完了 マスタ・エラー
	コールバック拡張機能	マスタ送信/受信完了コールバック時にストップ・コンディションを生成
タイマ TAU0 - チャネル 0 - インターバル・タイマ	インターバル時間	1ms
	割り込み設定	タイマ・チャネル 0 のカウント完了で割り込み発生(INTTM00)
	優先順位	低
12 ビット・インターバル・タイマ	インターバル・タイマ動作	使用する
	インターバル時間	10ms
	割り込み	インターバル信号検出(INTIT)
	優先順位	低
割り込み INTP0 - 外部割り込み	有効エッジ	立下りエッジ
	優先順位	低

3.4　ファイル構成

本サンプルプログラムのファイル構成を示します。

ファイル構成の(R)表記は、BLE プロトコルスタックに含まれているファイルであることを示します。ソフトウェア開発時は、BLE プロトコルスタックが提供する最新のコードファイルをご使用ください。

```
RL78G14_Fast_Prototyping_Board_HostSample
├──HostSample
│   ├──Platform
│   │   ├──driver
│   │   │   ├──hs3001
│   │   │   │       r_hs3001.c                          Renesas HS3001 ドライバ・コードファイル
│   │   │   │       r_hs3001.h                          Renesas HS3001 ドライバ・ヘッダファイル
│   │   │   │
│   │   │   ├──serial
│   │   │   │       uart.c                              UART ドライバ・コードファイル
│   │   │   │       uart.h                              UART ドライバ・コードファイル
│   │   │   │
│   │   │   ├──timer
│   │   │   │       timer.c                             タイマドライバ・コードファイル
│   │   │   │       timer.h                             タイマドライバ・ヘッダファイル
│   │   │   │
│   │   │   └──dataflash
│   │   │       │   dataflash.c                        データ・フラッシュドライバ・コードファイル
│   │   │       │   dataflash.h                        データ・フラッシュドライバ・ヘッダファイル
│   │   │       │   eel_descriptor_t02.c               EEPROM Emulation Library コードファイル
│   │   │       │   eel_descriptor_t02.h               EEPROM Emulation Library ヘッダファイル
│   │   │       │   fdl_descriptor_t02.c               Data Flash Access Library コードファイル
│   │   │       │   fdl_descriptor_t02.h               Data Flash Access Library ヘッダファイル
│   │   │       │
│   │   │       └──cc_rl
│   │   │           eel.h                              EEPROM Emulation Library API 定義ヘッダファイル
│   │   │           eel.lib                            EEPROM Emulation Library
│   │   │           eel_types.h                        EEPROM Emulation Library タイプ定義ヘッダファイル
│   │   │           fdl.h                              Data Flash Access Library API 定義ヘッダファイル
│   │   │           fdl.lib                            Data Flash Access Library
│   │   │           fdl_types.h                        Data Flash Access Library タイプ定義ヘッダファイル
│   │   │
│   │   └──include
│   │           arch.h                          (R)   アーキテクチャ・ヘッダファイル
│   │           compiler.h                      (R)   コンパイラ・ヘッダファイル
│   │           ll.h                            (R)   低レベルマクロ・ヘッダファイル
│   │           rscip_api.h                     (R)   RSCIP コールバック・ヘッダファイル
│   │           types.h                         (R)   タイプ定義・ヘッダファイル
│   │
│   └──rBLE
│       ├──host
│       │   │   rble_host.c                     (R)   rBLE_Host コードファイル
│       │   │   rble_if_api_cb.c                (R)   rBLE API コールバック・コードファイル
│       │   │
│       │   ├──gap
│       │   │       rble_api_gap.c              (R)   GAP API コードファイル
│       │   │
│       │   ├──gatt
│       │   │       rble_api_gatt.c             (R)   GATT API コードファイル
│       │   │
│       │   ├──sm
│       │   │       rble_api_sm.c               (R)   SM API コードファイル
│       │   │
│       │   └──vs
│       │           rble_api_vs.c               (R)   VS API コードファイル
```

```
|       |
|       ├──include
|       |   |   db_handle.h                                  (R)  データベースハンドル・ヘッダファイル
|       |   |   prf_sel.h                                    (R)  プロファイル選択・ヘッダファイル
|       |   |   rble.h                                       (R)  rBLE マクロ定義・ヘッダファイル
|       |   |   rble_api.h                                   (R)  rBLE API・ヘッダファイル
|       |   |   rble_app.h                                   (R)  rBLE SCP API・ヘッダファイル
|       |   |   rble_trans.h                                 (R)  rBLE 通信・ヘッダファイル
|       |   |
|       |   └──host
|       |           rble_host.h                              (R)  rBLE_Host ヘッダファイル
|       |
|       ├──rscip
|       |       rscip.c                                      (R)  RSCIP コードファイル
|       |       rscip.h                                      (R)  RSCIP ヘッダファイル
|       |       rscip_cntl.c                                 (R)  RSCIP コントロール・コードファイル
|       |       rscip_cntl.h                                 (R)  RSCIP コントロール・ヘッダファイル
|       |       rscip_ext.h                                  (R)  RSCIP 外部コールバック・ヘッダファイル
|       |       rscip_uart.c                                 (R)  RSCIP シリアル制御・コードファイル
|       |       rscip_uart.h                                 (R)  RSCIP シリアル制御・ヘッダファイル
|       |
|       ├──sample_app
|       |   |   app.c                                            ホストアプリケーション・コードファイル
|       |   |   r_hs3001_app.c                                   Renesas HS3001 アプリケーション・コードファイル
|       |   |   r_hs3001_app.h                                   Renesas HS3001 アプリケーション・ヘッダファイル
|       |   |
|       |   └──seclib
|       |           seclib.c                                     Security Library コードファイル
|       |           seclib.h                                     Security Library ヘッダファイル
|       |           secdb.c                                      Security Database コードファイル
|       |           secdb.h                                      Security Database ヘッダファイル
|       |
|       └──sample_profile
|           └──vuart
|                   vuart.h                                    汎用双方向通信共通ヘッダファイル
|                   vuarts.c                                   汎用双方向通信共通 Server ファイル
|                   vuarts.h                                   汎用双方向通信共通 Server ヘッダファイル
|
└──project
    ├──cs_cc                                                   スタートアップ・ルーチン
    |   |   cstart.asm                                         I/O ヘッダファイル
    |   |   iodefine.h                                         レジスタアクセス用マクロ定義・ヘッダファイル
    |   |   RL78G14_Fast_Prototyping_Board_HostSample.mtpj     CS+ for CC プロジェクトファイル
    |   |   RL78G14_Fast_Prototyping_Board_HostSample.rcpe     CS+ for CC プロジェクトファイル
    |   |   stkinit.asm                                        スタック領域初期化ルーチン
    |   |
    |   └──src
    |           r_cg_cgc.c                                     クロック生成ドライバ・コードファイル
    |           r_cg_cgc.h                                     クロック生成ドライバ・ヘッダファイル
    |           r_cg_cgc_user.c                                クロック生成ドライバ・ユーザコードファイル
    |           r_cg_it.c                                      12 ビットインターバルタイマドライバ・コードファイル
    |           r_cg_it.h                                      12 ビットインターバルタイマドライバ・ヘッダファイル
    |           r_cg_it_user.c                                 12 ビットインターバルタイマドライバ・ユーザコードファイル
    |           r_cg_macrodriver.h                             汎用マクロ・ヘッダファイル
    |           r_cg_macrodriver_hostsample.h                  汎用マクロ・ヘッダファイル(バックアップファイル)
    |           r_cg_port.c                                    ポートドライバ・コードファイル
    |           r_cg_port.h                                    ポートドライバ・ヘッダファイル
    |           r_cg_port_user.c                               ポートドライバ・ユーザコードファイル
    |           r_cg_serial.c                                  シリアルドライバ・コードファイル
    |           r_cg_serial.h                                  シリアルドライバ・ヘッダファイル
    |           r_cg_serial_user.c                             シリアルドライバ・ユーザコードファイル
    |           r_cg_timer.c                                   タイマアレイユニットドライバ・コードファイル
    |           r_cg_timer.h                                   タイマアレイユニットドライバ・ヘッダファイル
```

```
│           r_cg_timer_user.c                    タイマアレイユニットドライバ・ユーザコードファイル
│           r_cg_intc.c                          割り込みドライバ・コードファイル
│           r_cg_intc.h                          割り込みドライバ・ヘッダファイル
│           r_cg_intc_user.c                     割り込みドライバ・ユーザコードファイル
│           r_cg_userdefine.h                    ユーザ定義マクロ・ヘッダファイル
│           r_main.c                             メインループ・コードファイル
│           r_systeminit.c                       周辺機能初期化・コードファイル
│
└──e2studio
    │   .cproject                                e2studio プロジェクトファイル
    │   .project                                 e2studio プロジェクトファイル
    │   RL78G14_Fast_Prototyping_Board_HostSample- e2studio プロジェクトファイル
    │   HardwareDebug.launch
    │
    ├──.settings
    │   │   Dependency_Scan_Preferences.prefs    e2studio プロジェクトファイル
    │   │   e2studio_project.prefs               e2studio プロジェクトファイル
    │   │   org.eclipse.cdt.managedbuilder.core.prefs e2studio プロジェクトファイル
    │   │   renesasPGModel.xml                   e2studio プロジェクトファイル
    │   │
    │   └──CodeGenerator                         e2studio プロジェクトファイル
    │           cgproject.cgp                    e2studio プロジェクトファイル
    │           cgprojectDatas.datas
    │
    ├──generate
    │       cstart.asm                           スタートアップ・ルーチン
    │       iodefine.h                           I/O ヘッダファイル
    │       stkinit.asm                          スタック領域初期化ルーチン
    │
    └──src
            r_cg_cgc.c                           クロック生成ドライバ・コードファイル
            r_cg_cgc.h                           クロック生成ドライバ・ヘッダファイル
            r_cg_cgc_user.c                      クロック生成ドライバ・ユーザコードファイル
            r_cg_it.c                            12 ビットインターバルタイマドライバ・コードファイル
            r_cg_it.h                            12 ビットインターバルタイマドライバ・ヘッダファイル
            r_cg_it_user.c                       12 ビットインターバルタイマドライバ・ユーザコードファイル
            r_cg_macrodriver.h                   汎用マクロ・ヘッダファイル
            r_cg_port.c                          ポートドライバ・コードファイル
            r_cg_port.h                          ポートドライバ・ヘッダファイル
            r_cg_port_user.c                     ポートドライバ・ユーザコードファイル
            r_cg_serial.c                        シリアルドライバ・コードファイル
            r_cg_serial.h                        シリアルドライバ・ヘッダファイル
            r_cg_serial_user.c                   シリアルドライバ・ユーザコードファイル
            r_cg_timer.c                         タイマアレイユニットドライバ・コードファイル
            r_cg_timer.h                         タイマアレイユニットドライバ・ヘッダファイル
            r_cg_timer_user.c                    タイマアレイユニットドライバ・ユーザコードファイル
            r_cg_intc.c                          割り込みドライバ・コードファイル
            r_cg_intc.h                          割り込みドライバ・ヘッダファイル
            r_cg_intc_user.c                     割り込みドライバ・ユーザコードファイル
            r_cg_userdefine.h                    ユーザ定義マクロ・ヘッダファイル
            r_main.c                             メインループ・コードファイル
            r_systeminit.c                       周辺機能初期化・コードファイル
```

4. ビルド手順

RL78/G14 Fast Prototyping Board のホストサンプルをビルドするためのプロジェクトとビルド手順を以下に示します。

表 4-1　RL78/G14 Fast Prototyping Board プロジェクト

CS+ for CC		
	Project File	RL78G14_Fast_Prototyping_Board_HostSample¥project¥cs_cc¥ RL78G14_Fast_Prototyping_Board_HostSample.mtpj
	HEX File	RL78G14_Fast_Prototyping_Board_HostSample¥project¥cs_cc¥DefaultBuild¥ RL78G14_Fast_Prototyping_Board_HostSample.hex
e² studio		
	Project Folder	RL78G14_Fast_Prototyping_Board_HostSample¥project¥e2studio
	HEX File	RL78G14_Fast_Prototyping_Board_HostSample¥project¥e2studio¥HardwareDebug¥ RL78G14_Fast_Prototyping_Board_HostSample.hex

4.1 CS+ for CC

1. 「表 4-1　RL78/G14 Fast Prototyping Board プロジェクト」の Project File に示すプロジェクトファイルをダブルクリックします。これにより CS+が起動します。
2. 「プロジェクト・ツリー」内の「RL78G14_Fast_Prototyping_Board_HostSample（プロジェクト）」を右クリックし、ドロップダウンメニューから「RL78G14_Fast_Prototyping_Board_HostSample をビルド」を選択して、ビルドを開始します。
3. 「表 4-1　RL78/G14 Fast Prototyping Board プロジェクト」の CS+ for CC の HEX File の欄で示すパスに HEX ファイルが生成されます。

4.2 e² studio

1. e² studio を起動します。
2. 「プロジェクト・エクスプローラー」上で右クリックし、表示されたメニューから「インポート」を選択します。
3. 「インポート」ウインドウが表示されるので、「既存プロジェクトをワークスペースへ」を選択し、「次」をクリックします。
4. 「ルートディレクトリの選択」フォームに、「表 4-1　RL78/G14 Fast Prototyping Board プロジェクト」の e² studio の Project Folder に示すプロジェクトフォルダを選択します。選択後、「プロジェクト」内に指定したプロジェクトが表示されていることを確認し、「終了」をクリックします。すると、「インポート」ウインドウが閉じられます。
5. 「プロジェクト・エクスプローラー」上に表示されたプロジェクト上で右クリックし、「プロジェクトのビルド」を選択し、ビルドを開始します。
6. 「表 4-1　RL78/G14 Fast Prototyping Board プロジェクト」の e² studio の HEX File に示すパスに HEX ファイルが生成されます。

4.3　Renesas Flash Programmer を使用した HEX ファイルの書き込み

本アプリケーションノートに添付されているビルド済みの HEX ファイルの書き込み方法について記します。

ビルド済み HEX ファイルを書き込むためには Fast Prototyping Board が単体動作できるようにヘッダ部品を搭載する必要があります。詳細は「RL78/G14 Fast Prototyping Board ユーザーズマニュアル」(R20UT4573)の「5.12 エミュレータリセットヘッダ」を参照してください。

表 4-2　ビルド済み HEX ファイル

HEX File		
	RFP Project File	RL78G14_Fast_Prototyping_Board_HostSample¥ROM_File¥ RL78G14 Fast Prototyping Board.rpj
	HEX File	RL78G14_Fast_Prototyping_Board_HostSample¥ROM_File¥ RL78G14_Fast_Prototyping_Board_HostSample.hex

1. Renesas Flash Programmer を起動します。
2. メニューの「ファイル(F)」-「プロジェクトを開く(O)...」を選択し、「表 4-2　ビルド済み HEX ファイル」の RFP Project File を開きます。
3. 「操作」タブの「プログラムファイル」の「参照...(B)」ボタンを押し、「表 4-2　ビルド済み HEX ファイル」の HEX File を開きます。
4. 「スタート(S)」ボタンを押して書き込みます。

5. 通信動作確認手順

「図 3-1　デバイス構成」で示す構成で BLE 通信の動作確認を行ないます。

5.1 GATTBrowser のインストール

スマートフォン(タブレット)に BLE の動作確認を行なうことができる GATTBrowser をインストールしてください。

- Android 版 GATTBrowser

  https://play.google.com/store/apps/details?id=com.renesas.ble.gattbrowser

- iOS 版 GATTBrowser

  https://itunes.apple.com/jp/app/gattbrowser/id1163057977?mt=8

5.2 ホストサンプルの実行

CS+ for CC または e^2 studio を使用してビルドしたホストサンプルを RL78/G14 Fast Prototyping Board にダウンロードして実行します。

最初に「4.ビルド手順」に従ってプログラムをビルドしてください。RL78/G14 Fast Prototyping Board と PC を USB ケーブルで接続し、プログラムをダウンロードします。プログラムを実行すると、RL78/G1D BLE Module Expansion Board からアドバタイジングパケットの送信が開始されます。

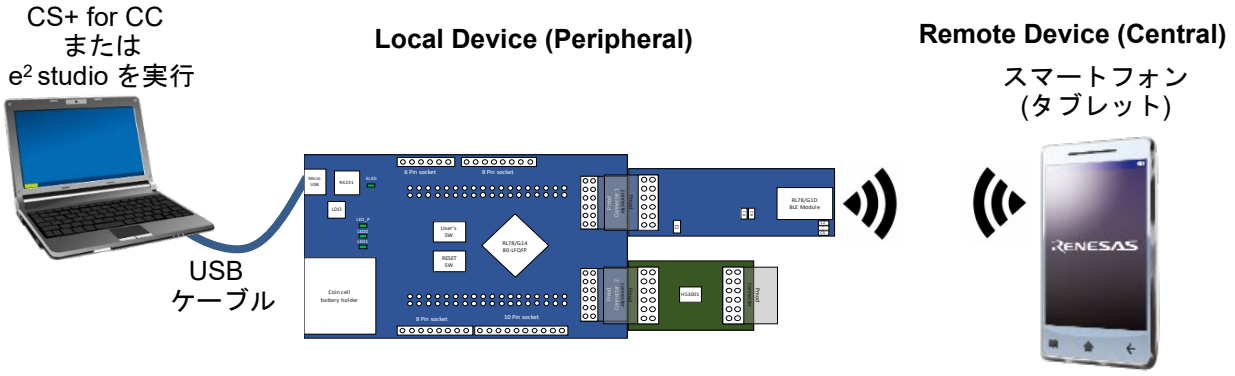

図 5-1　通信動作確認のデバイス構成

5.2.1 CS+ for CC

1. 「4.1 CS+ for CC」を参照してプログラムをビルドしてください。
2. メニューの「デバッグ(D)」-「デバッグ・ツールへダウンロード(D)」を選択し、RL78/G14 Fast Prototyping Board へプログラムをダウンロードします。
3. CS+の内部エディターで"Platform¥driver¥hs3001¥r_hs3001.c"を開きます。
4. 135 行目の"humidity"の上で右クリックし、「アクション・イベントの登録(A)...」を選択します。
5. 「Printf イベント」タブ -「変数式(V)」のテキストボックスに、"humidity"を入力し、「OK」を押します。
6. 143 行目の"temperature"の上で右クリックし、「アクション・イベントの登録(A)...」を選択します。
7. 「Printf イベント」タブ -「変数式(V)」のテキストボックスに、"temperature"を入力し、「OK」を押します。
8. メニューの「デバッグ(D)」-「実行(G)」を選択、または「F5」キーを押し、プログラムを実行します。
9. RL78/G1D BLE Module Expansion Board からアドバタイジングパケットの送信が開始されます。

5.2.2 e² studio

1. 「4.2 e² studio」を参照してプログラムをビルドしてください。
2. メニューの「実行(R)」-「デバッグ(D)」を選択、または「F11」を押し、RL78/G14 Fast Prototyping Board へプログラムをダウンロードします。
3. e² studio のエディターで"src¥HostSample¥Platform¥driver¥hs3001¥r_hs3001.c"を開きます。
4. 135 行目の行番号の上で右クリックし、「Add Dynamic Printf...」を選択します。
5. 「printf」のテキストボックスに、""humidity=%d¥n", humidity"を入力し、「Apply and Close」を押します。
6. 143 行目の行番号の上で右クリックし、「Add Dynamic Printf...」を選択します。
7. 「printf」のテキストボックスに、""temperature=%d¥n", temperature"を入力し、「Apply and Close」を押します。
8. メニューの「実行(R)」-「再開(M)」を選択、または「F8」を押し、プログラムを実行します。
9. RL78/G1D BLE Module Expansion Board からアドバタイジングパケットの送信が開始されます。

5.3　スマートフォンを使用した通信動作確認

5.3.1　Android デバイスでの通信動作確認

Remote Device(Central)に Android デバイスを使用して通信動作の確認を行ないます。「図 5-2　Android デバイスでの通信動作確認」も合わせて参照してください。

1. Android デバイスにインストールした GATTBrowser を起動します。
2. デバイスの検索結果から、RTK5RL140C と表示されたデバイスと接続を開始します。（図 A1 の矢印 1）
3. 接続すると Service の一覧が表示されます。一番下までスクロールさせ Renesas Virtual UART Service の Indication Characteristic を選択します。（図 A2 の矢印 2）
4. [Indication Off]をタップし、[Indication On]にします。（図 A3 の矢印 3）
5. Service の一覧に戻り、Write Characteristic を選択します。（図 A4 の矢印 4）
6. String を選択し's'を入力した後、Write をタップすると、Android デバイスから's'が送信されます。's'を受信した Local Device(Peripheral)は、センサで測定した情報を Indication で送信します。（図 A5 の矢印 5）
7. Service の一覧に戻り、Indication Characteristic を選択します。Local Device(Peripheral)から 5 秒ごとに送信されるセンサ情報が表示されます。（図 A6 の矢印 6）

(1)　CS+でのセンサ測定結果の表示

1. CS+の「表示(V)」-「出力(O)」を選択し、出力パネルを開きます。
2. センサでの測定結果(湿度、温度)が表示されます。

(2)　e² studio でのセンサ測定結果の表示

1. e² studio の「ウインドウ(W)」-「ビューの表示(V)」-「Debugger Console」を選択し、Debugger Console を開きます。
2. センサでの測定結果(湿度、温度)が表示されます。

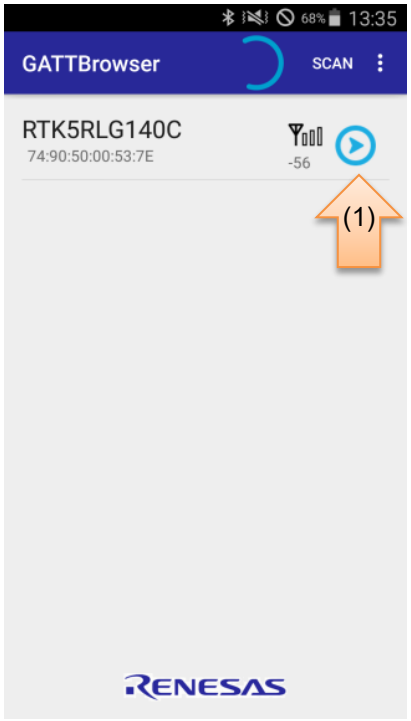

図 A1

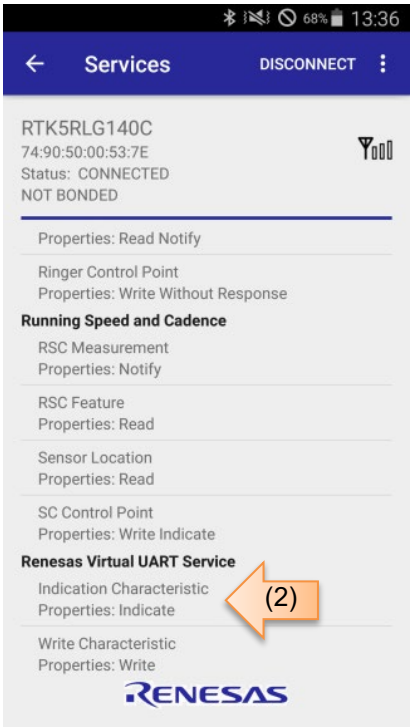

図 A2

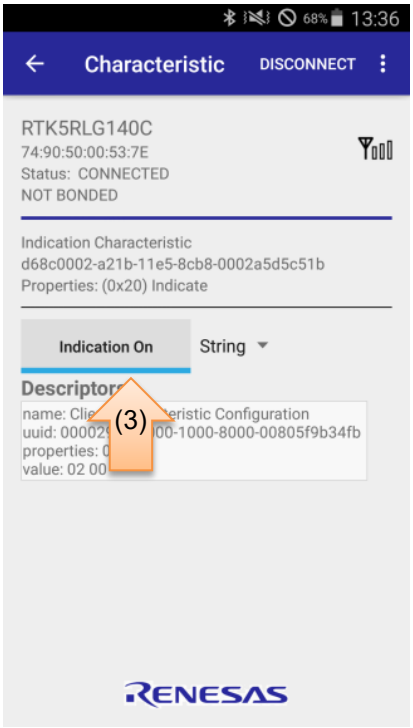

図 A3

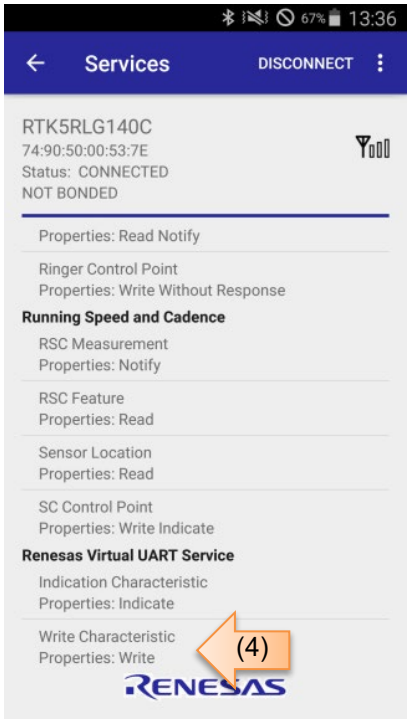

図 A4

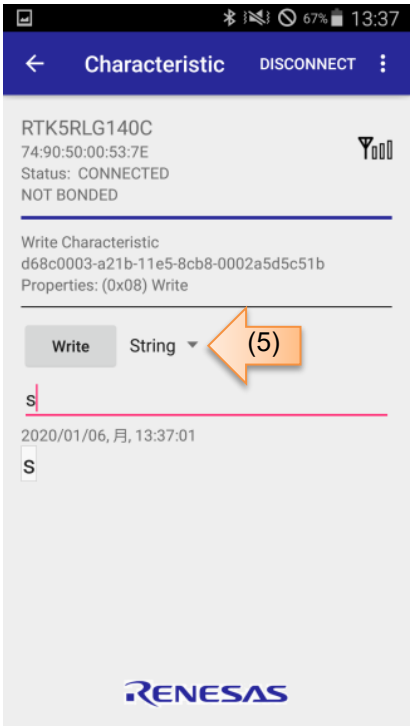

図 A5

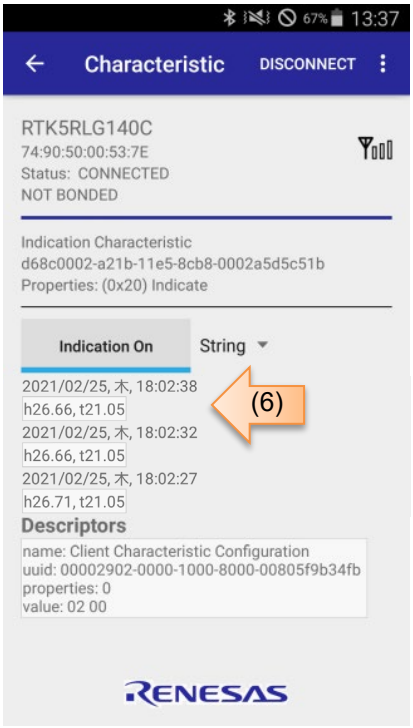

図 A6

図 5-2　Android デバイスでの通信動作確認

5.3.2　iOS デバイスでの通信動作確認

Remote Device(Central)に iOS デバイスを使用して通信動作の確認を行ないます。「図 5-3　iOS デバイスでの通信動作確認」も合わせて参照してください。

1. iOS デバイスにインストールした GATTBrowser を起動します。
2. デバイスの検索結果から、RTK5RL140C と表示されたデバイスと接続を開始します。（図 B1 の矢印 1）
3. 接続すると Service の一覧が表示されます。一番下までスクロールさせ Renesas Virtual UART Service の Indication Characteristic を選択します。（図 B2 の矢印 2）
4. [Enable Indication]をタップし、[Disable Indication]にします。（図 B3 の矢印 3）
5. Service の一覧に戻り、Write Characteristic を選択します。（図 B4 の矢印 4）
6. String を選択し's'を入力した後、Write をタップすると、Android デバイスから's'が送信されます。's'を受信した Local Device(Peripheral)は、センサで測定した情報を Indication で送信します。（図 B5 の矢印 5）
7. Service の一覧に戻り、Indication Characteristic を選択します。Local Device(Peripheral)から 5 秒ごとに送信されるセンサ情報が表示されます。（図 B6 の矢印 6）

(1)　CS+センサ測定結果の表示

1. CS+の「表示(V)」-「出力(O)」を選択し、出力パネルを開きます。
2. センサでの測定結果(湿度、温度)が表示されます。

(2)　e² studio センサ測定結果の表示

1. e² studio の「ウインドウ(W)」-「ビューの表示(V)」-「Debugger Console」を選択し、Debugger Console を開きます。
2. センサでの測定結果(湿度、温度)が表示されます。

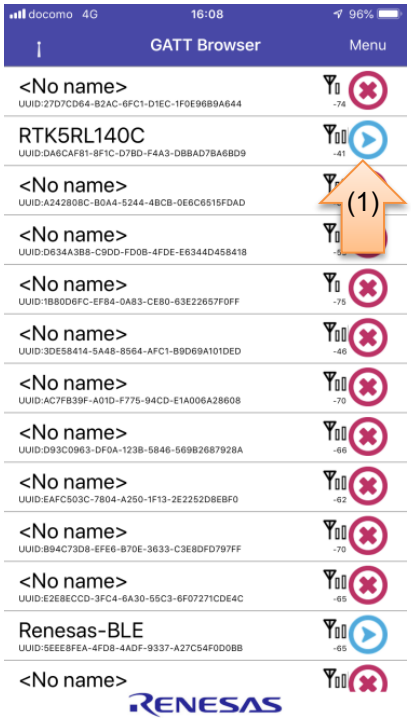

図 B1

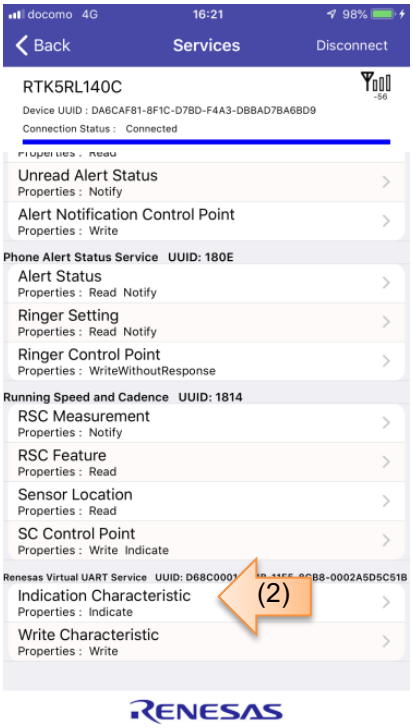

図 B2

図 B3

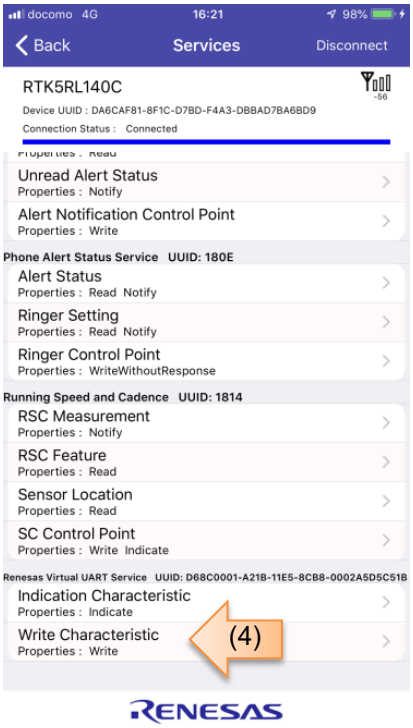

図 B4

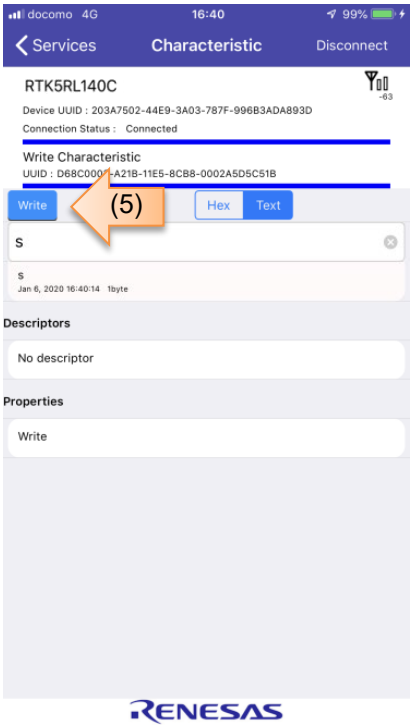

図 B5

図 B6

図 5-3　iOS デバイスでの通信動作確認

6. 動作

「図 3-2　ソフトウェア構成」で示される Host Application(以降、APP)と rBLE_Host を中心に、Host MCU プログラムの動作概要を示します。

6.1 rBLE コマンドと rBLE イベントの動作

「図 6-1　rBLE コマンドと rBLE イベントの動作概要」に rBLE コマンドと rBLE イベントの動作を示します。

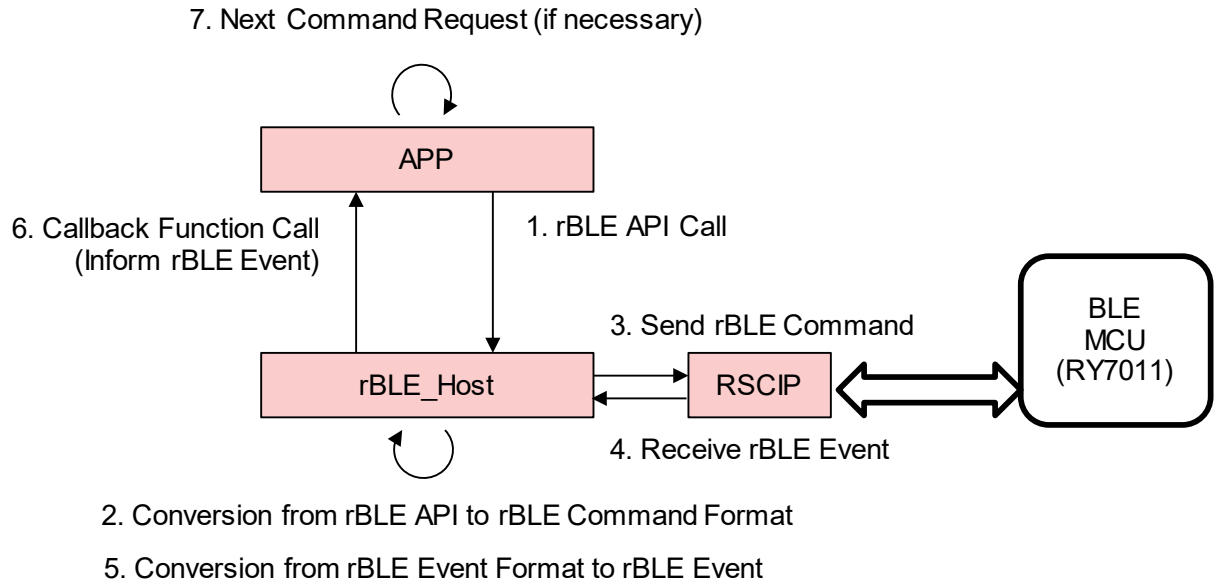

図 6-1　rBLE コマンドと rBLE イベントの動作概要

1. APP は rBLE API をコールします。
2. rBLE_Host は rBLE API とパラメータを rBLE コマンドフォーマットへ変換します。
3. rBLE コマンドは RSCIP で BLE MCU へ送信されます。
4. BLE MCU は rBLE コマンドを実行した後、rBLE イベントを RSCIP で Host MCU へ送信します。
5. rBLE_Host は rBLE イベントフォーマットから rBLE イベントに変換します。
6. rBLE_Host は APP のコールバック関数を呼び出して rBLE イベントを通知します。
7. APP は次の rBLE API をコールします。

rBLE コマンド、rBLE イベントについては、「Bluetooth Low Energy® プロトコルスタック rBLE コマンド仕様書」(R01AN1376)を参照してください。

6.2　メインループの動作

「図 6-2　メインループ動作」にホストサンプルのメインループ動作を示します。

メインループは、APP のシーケンス処理と rBLE の API コールやイベント動作を行う APP スケジューラと、rBLE API コマンドフォーマットの送信や rBLE Event フォーマットの受信を行う rBLE スケジューラ、センサの測定を行うセンサスケジューラ、MCU を STOP モードに遷移させる MCU モード管理処理を実行します。

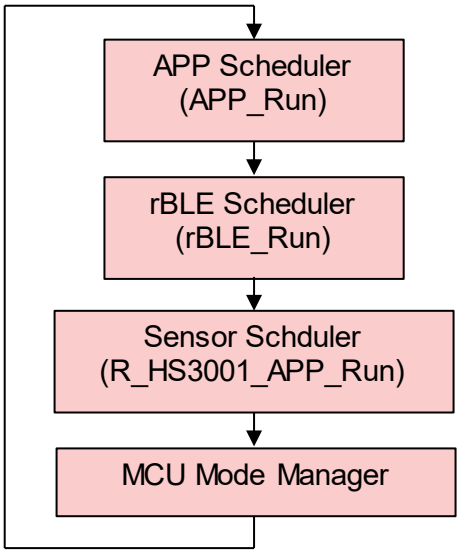

図 6-2　メインループ動作

APP スケジューラは、シーケンス処理で使用するコマンド要求キューを持ち、コマンド要求キューにコマンド要求がセットされているならば rBLE API をコールします。rBLE スケジューラからコールされたコールバック関数は、次のコマンド要求をキューにセットします。

rBLE スケジューラは、rBLE コマンドフォーマットや rBLE イベントフォーマットで BLE MCU と通信します。また、イベントキューを持ち、BLE MCU から受信した rBLE イベントがイベントキューにセットされているならば、APP のコールバック関数をコールします。

センサスケジューラは、センサ初期化やセンサ測定を行います。測定結果は汎用双方向通信を使用してスマートフォンへ送信します。

MCU モード管理処理は、コマンド要求キューとイベントキューに何もセットされていなければ、MCU を STOP に遷移させます。MCU は割り込みによって STOP から復帰します。

6.3　GPCP 送受信関数

Bluetooth でセンサの測定開始指示や、測定データを通信する際に使用する関数について説明します。

データ通信のシーケンスについては「7.10 Step9. Profile Communication シーケンス」を参照してください。

6.3.1　送信関数

センサで測定したデータをスマートフォンへ送信する関数を示します。

void R_HS3001_APP_Send_Data(void)
– HostSample¥rBLE¥sample_app¥r_hs3001_app.c : line.224 GPCP の Indication 送信関数(RBLE_VUART_Server_Send_Indication()[注 1])を呼び出し、センサの測定データをスマートフォンへ送信します。本関数は rBLE のタイマ機能を使用して定期的に実行されます。 【注】 1. 関数詳細は「Bluetooth® low energy プロトコルスタック BLE 仮想 UART アプリケーション」(R01AN3130)の「8.5.2.3 RBLE_VUART_Server_Send_Indication」を参照してください。
Parameters:
none
Return:
nono

6.3.2　受信関数

スマートフォンからのデータを受信する関数を示します。

void R_HS3001_APP_Receive_Data_Callback(uint8_t *rbuf, uint8_t len)

– HostSample¥rBLE¥sample_app¥r_hs3001_app.c : line.278
スマートフォンのデータを受信したことで発生する GPCP のイベント (RBLE_VUART_EVENT_SERVER_WRITE_REQ 注 1)から呼び出され、センサの測定開始や停止を行います。本関数をコールバック関数として設定することで、GPCP イベントから呼び出されるようになります。
コールバック関数の設定は R_HS3001_APP_Init()関数の中で行います。下記に設定例を示します。
– HostSample¥rBLE¥sample_app¥r_hs3001_app.c : line.118
cb_func_t cbf;
cbf.cb_confirmation = R_HS3001_APP_Confirmation_Callback;
cbf.cb_receive_data = R_HS3001_APP_Receive_Data_Callback;
rBLEAPI_Register_CB(&cbf);
 	
【注】 1. イベントの詳細は「Bluetooth® low energy プロトコルスタック BLE 仮想 UART アプリケーション」(R01AN3130)の「8.5.3.2 RBLE_VUART_EVENT_SERVER_WRITE_REQ」を参照してください。

Parameters:

uint8_t *rbuf	受信データが格納されているメモリのアドレス
uint8_t len	データ長

Return:
none

6.4　UART 2 線分岐接続方式

RL78/G1D BLE Module Expansion Board とのシリアル通信で使われる UART 2 線分岐接続方式の通信方法について示します。本アプリケーションノートで使用する RL78/G14 Fast Prototyping Board と RL78/G1D BLE Module Expansion Board との UART 2 線分岐接続方式での接続については、「6.4.3 応用回路」を参照してください。

6.4.1　送信動作

Host MCU から BLE MCU への送信を行うには、ハンドシェイクを行う必要があります。ハンドシェイクは Host MCU から送信する REQ バイト(0xC0)と、BLE MCU から送信される ACK バイト(0x88)または RSCIP パケットによって行われます。また、ハンドシェイクを行う時にはタイマによる監視を行い、タイムアウト発生時にはハンドシェイクを再実行します。Host MCU の UART ドライバは、このハンドシェイクを行うため、送信状況によって 5 つの状態を持ちます。

表 6-1　UART ドライバ送信状態

状態	説明
T_IDLE	UART ドライバ初期化、RSCIP パケット送信完了
T_REQUESTING	REQ バイト送信中
T_RCV_BF_REQUESTED	ACK バイトの代わりに BLE MCU から RSCIP パケットを受信
T_REQUESTED	REQ バイト送信完了(BLE MCU からの ACK バイト待ち)
T_ACTIVE	RSCIP パケット送信中

Host MCU から BLE MCU への送信は必ず REQ バイトから開始されます。REQ バイトを送信した後、Host MCU は受信状態により次の動作のいずれかに分岐します。

(a)　Host MCU が BLE MCU からの RSCIP パケットを受信していない(図 6-3)

(b)　Host MCU が BLE MCU からの RSCIP パケットを受信中(図 6-4)

(c)　ACK バイト受信タイムアウト(図 6-5)

(1)　Host MCU が BLE MCU からの RSCIP パケットを受信していない

この状態は、BLE MCU から RSCIP パケットが送信されておらず、Host MCU から REQ バイトを送信した後、Host MCU が ACK バイトの受信を待っている状態です。BLE MCU は REQ バイトを受信し ACK バイトを送信します。ACK バイトを受信した Host MCU は、BLE MCU に RSCIP パケットを送信します。

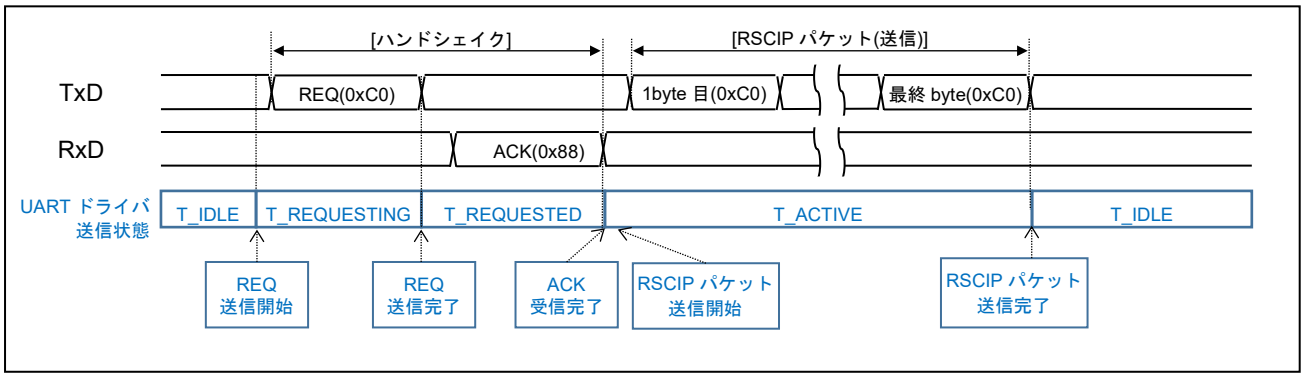

図 6-3　Host MCU が BLE MCU からの rBLE パケットを受信していない場合の動作

(2)　Host MCU が BLE MCU からの RSCIP パケットを受信中

この状態は BLE MCU が RSCIP パケットを送信しており、Host MCU は RSCIP パケットを受信している状態です。BLE MCU は REQ を受信しても ACK バイトを返さず、送信している RSCIP パケットを ACK バイトの代わりとします。ホストは BLE MCU からの RSCIP パケットを ACK バイトの代わりとし、BLE MCU に RSCIP パケットを送信します。

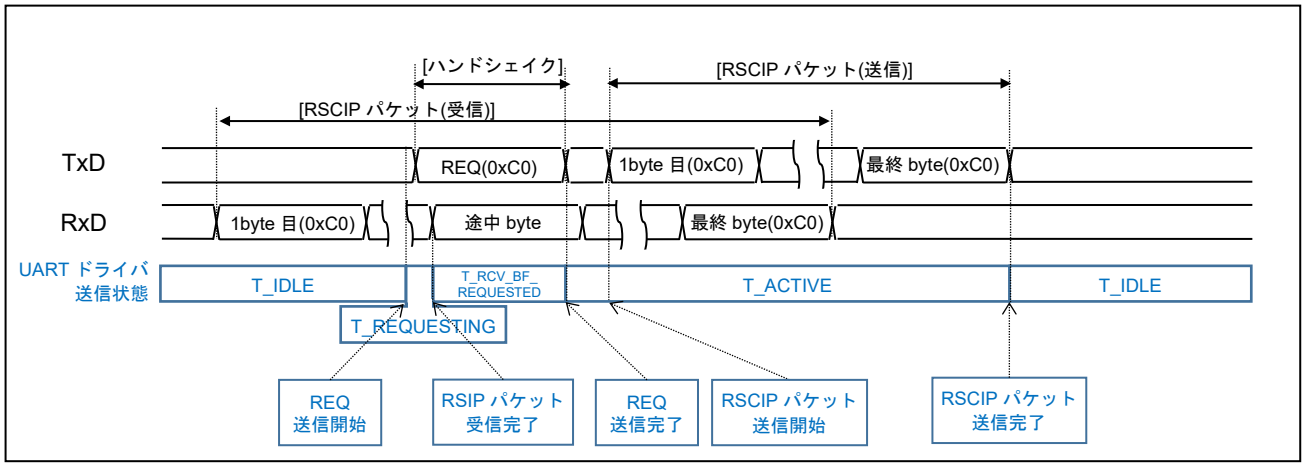

図 6-4　Host MCU が BLE MCU からのデータを受信している場合の動作

(3)　ACK バイト受信タイムアウト

REQ バイトを送信した後 Host MCU は、タイムアウトタイマを動作させます。一定時間 ACK バイトを受信できなかった場合、REQ バイトを再送します。

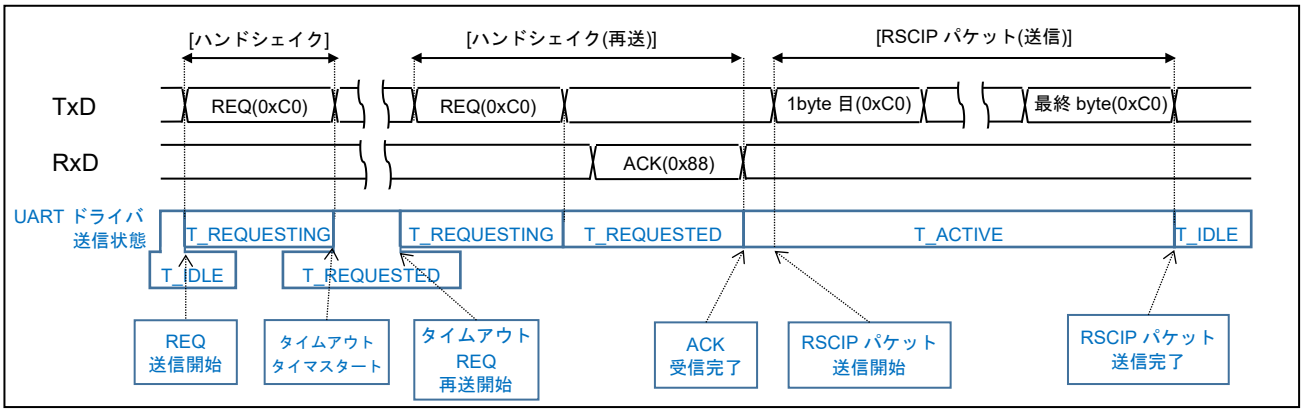

図 6-5　ACK バイトの受信がタイムアウトした場合の動作

6.4.2　受信動作

受信時に UART ドライバの状態遷移はありません。BLE MCU からのデータを受信するために、rBLE_Host から指定されたバイト数で BLE MCU からの RSCIP パケットを待ち受けます。

6.4.3　応用回路

RL78/G1D モジュール RY7011 と Host MCU をモデム構成で接続する応用回路を示します。

(1)　RL78/G14 Fast Prototyping Board と RL78/G1D BLE Module Expansion Board の接続

Host MCU に RL78/G14 Fast Prototyping Board、BLE MCU に RL78/G1D BLE Module Expansion Board を使用する構成の UART 2 線分岐接続方式図を示します。2 つのボードは Pmod™ インタフェースで接続されているため Host MCU の TxD ラインを分岐して RY7011 の 2 ピンへ入力することができません。代わりに Host MCU のポート機能で RY7011 の 2 ピンへ Low レベルを入力することにより、疑似的に UART 2 線分岐接続方式で通信できるようになります。

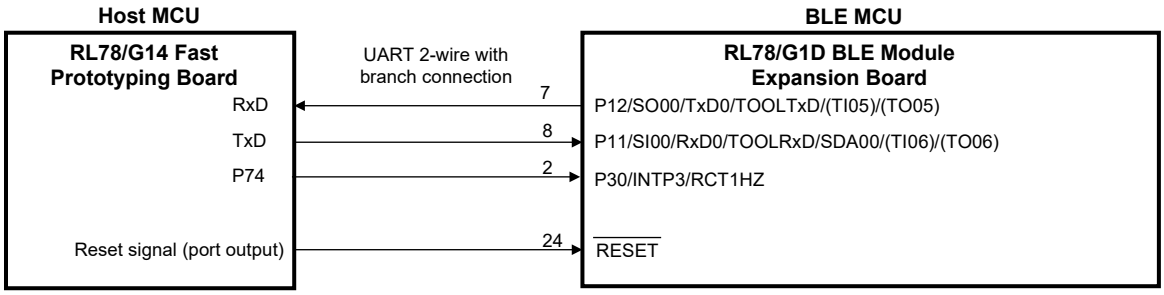

図 6-6　UART 2 線分岐接続方式(1)

(2)　RL78/G1D モジュール RY7011 との接続

BLE MCU に RY7011 を使用する構成の UART 2 線分岐接続を示します。

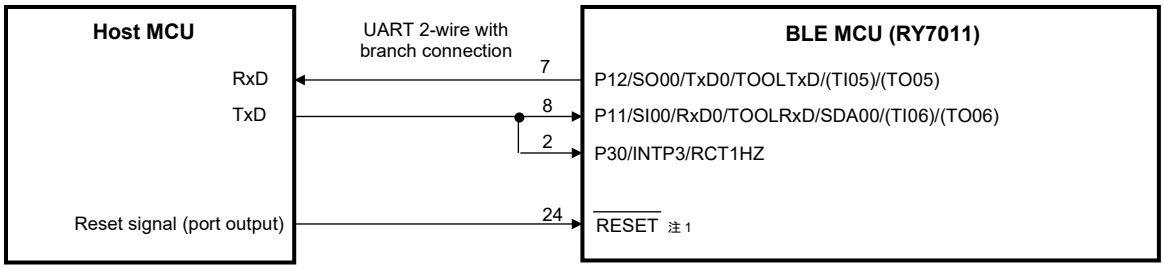

図 6-7　UART 2 線分岐接続方式(2)

【注】　1. /RESET 端子は、必要に応じてプルアップ／プルダウン抵抗を追加してください（「RL78/G1D ユーザーズマニュアル ハードウェア編」(R01UH0515)参照）。

6.5　センサの無効化

センサ(Renesas HS3001 Humidity and Temperature Sensor Module)処理を無効にする方法を示します。センサ処理のソースコードをプロジェクトから外すため、スマートフォンとコネクションし Indication を許可した後は、アルファベット 1 文字をスマートフォンへ送信します。

(1)　定義マクロの変更

CS+ for CC または e² studio のプロジェクトで、定義マクロを下記のように変更します。

USE_SENSOR　-> noUSE_SENSOR

[CS+ for CC]

メニューの「表示(V)」-「プロパティ(P)」を選択します。「プロジェクト・ツリー」内で「CC-RL(ビルドツール)」を選択します。「プロパティ」タブ-「共通オプション」タブ内の「定義マクロ」を編集します。

[e² studio]

「プロジェクト・エクスプローラー」内で「RL78G14_Fast_Prototyping_Board_HostSample」を選択します。メニューの「プロジェクト(P)」-「プロパティ(P)」を選択します。「C/C++ビルド」の「設定」を選択します。「ツール設定」タブの「Compiler」-「ソース」を選択し、「プリプロセッサ・マクロの定義」を編集します。

(2)　ソースファイルの取り外し

CS+ for CC または e² studio のプロジェクトから、下記ソースファイルを取り外します。

[CS+ for CC]

「プロジェクト・ツリー」内で下記ファイルを選択し、右クリックのメニューから「プロジェクトから外す」を選択します。

RL78G14_Fast_Prototyping_Board_HostSample¥File¥Platform¥driver¥hs3001¥

r_hs3001.c

r_hs3001.h

RL78G14_Fast_Prototyping_Board_HostSample¥File¥rBLE¥sample_app¥

r_hs3001_app.c

r_hs3001_app.h

[e² studio]

「プロジェクト・エクスプローラー」内で下記ファイルを選択し、右クリックのメニューから「削除」を選択します。

RL78G14_Fast_Prototyping_Board_HostSample¥src¥HostSample¥Platform¥driver¥hs3001¥

r_hs3001.c

r_hs3001.h

RL78G14_Fast_Prototyping_Board_HostSample¥src¥HostSample¥rBLE¥sample_app¥

r_hs3001_app.c

r_hs3001_app.h

7. 通信シーケンス

本章では、機器の間や、ソフトウェアブロック間の通信シーケンスを示します。機器は、Local Device の Host MCU と BLE MCU、Remote Device のスマートフォンで構成されます。Local Device は、Host MCU の APP、SecLib、Data Flash、rBLE_Host と、BLE MCU の rBLE_Core のソフトウェアで構成されます。

SecLib 内部の通信シーケンスは「Bluetooth® Low Energy プロトコルスタック Security Library」(R01AN3777)を参照して下さい。

7.1 メインシーケンス

メインシーケンス図では、処理ブロックとして Step1～10 までを定義し、処理ブロックの順序と関連するデバイスまたはソフトウェアの範囲を示します。処理ブロック Step1～10 の詳細は次節以降に記載します。

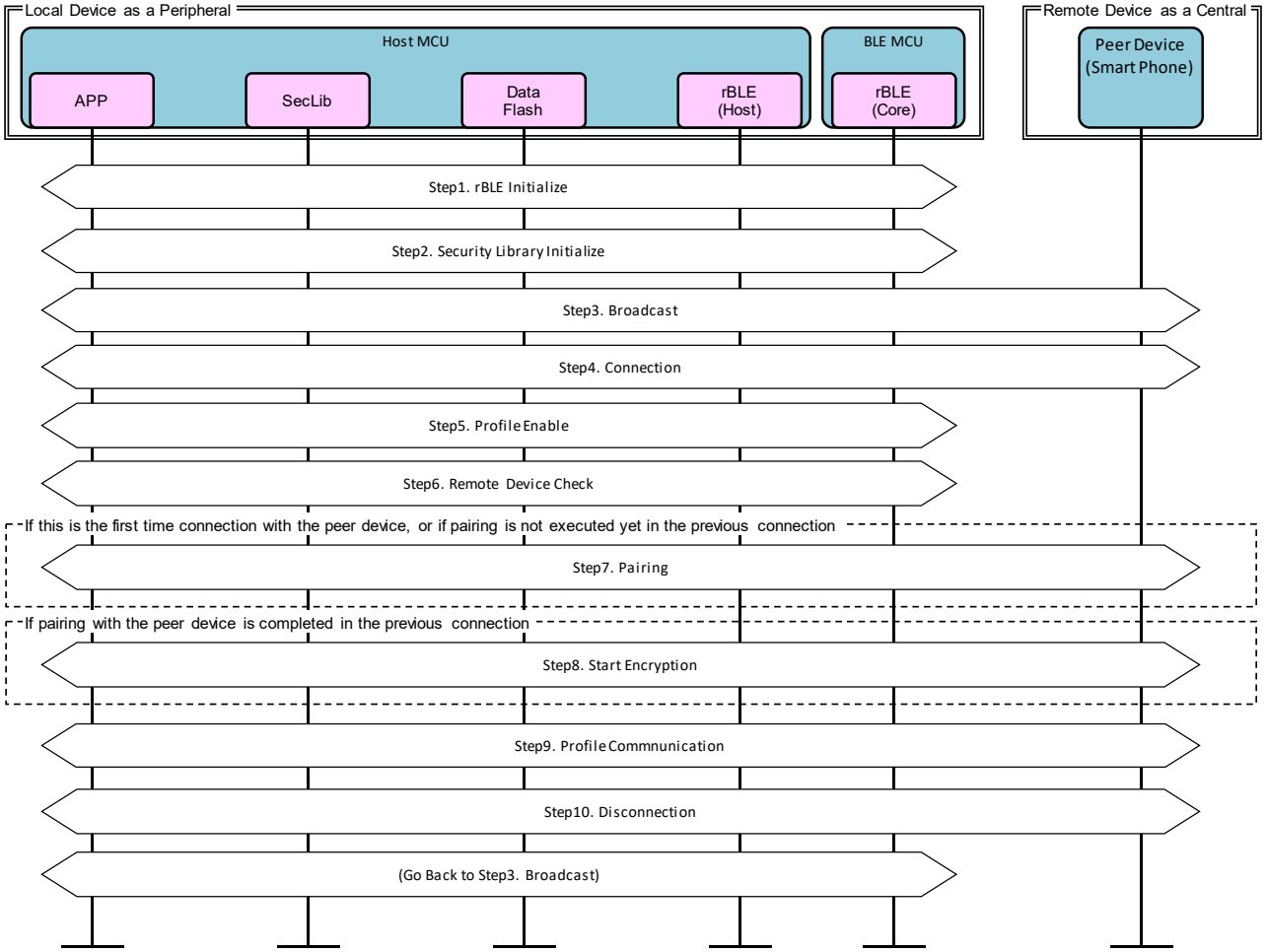

図 7-1　メインシーケンス

7.2　Step1. rBLE Initialize シーケンス

APP は RBLE_Init 関数をコールし、rBLE（rBLE_Host/rBLE_Core）を初期化します。rBLE の初期化が完了し BLE MCU とのシリアル通信が確立されると、rBLE から RBLE_MODE_ACTIVE イベントが通知されます。

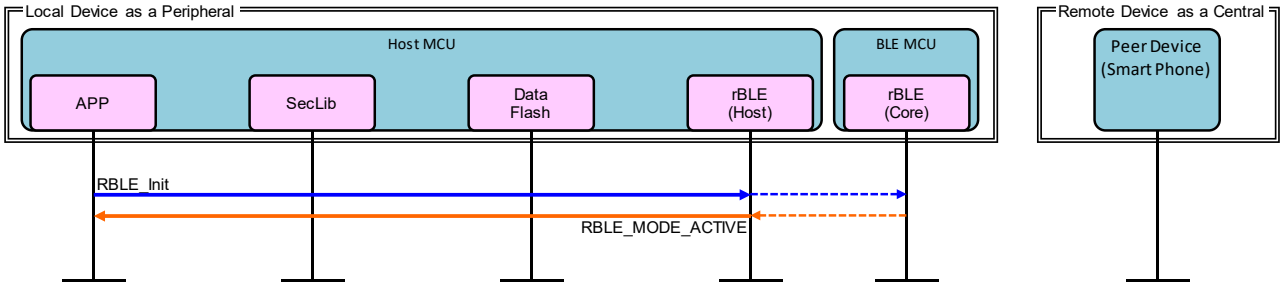

図 7-2　rBLE Initialize シーケンス

7.3　Step2. Security Library Initialize シーケンス

APP は SecLib_Init 関数をコールし、SecLib を初期化します。GAP の初期化は、SecLib がライブラリ内部で RBLE_GAP_Reset 関数を呼び出して初期化します。初期化が完了すると、SecLib から SECLIB_EVENT_INIT_COMP イベントが通知されます。

APP は SecLib_Set_Param 関数をコールしペアリングやプライバシーの実施に使用するセキュリティパラメータを設定します。設定が完了すると、SecLib から SECLIB_EVENT_SET_PARAM_COMP イベントが通知されます。

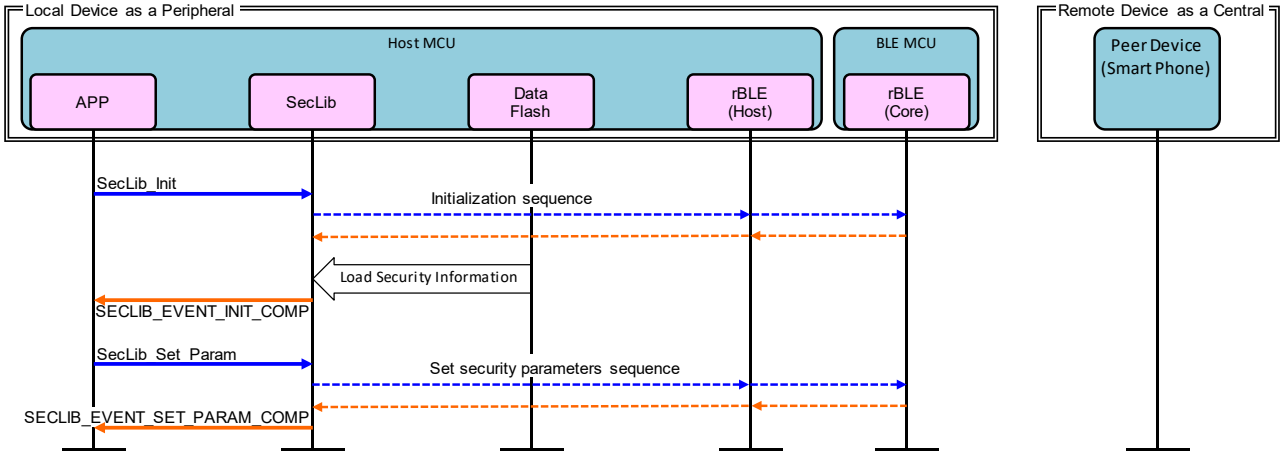

図 7-3　Security Library Initialize シーケンス

7.4　Step3. Broadcast シーケンス

Local Device を Peripheral として接続するための Broadcast を開始します。

APP は RBLE_GAP_Broacast_Enable 関数をコールし、Broadcast を開始します。開始が完了すると、rBLE から RBLE_GAP_EVENT_BROADCAST_ENABLE_COMP イベントが通知されます。

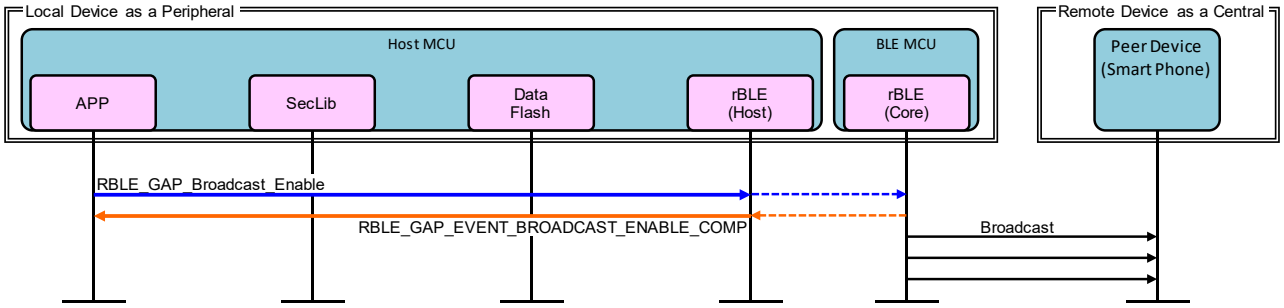

図 7-4　Broadcast シーケンス

7.5　Step4. Connection シーケンス

Local Device からの Broadcast を受信した Remote Device は、接続を要求します。

Remote Device から Connection Request が送信され、Local Device と Remote Device の接続が確立されると、rBLE から RBLE_GAP_EVENT_CONNECTION_COMP イベントが通知されます。

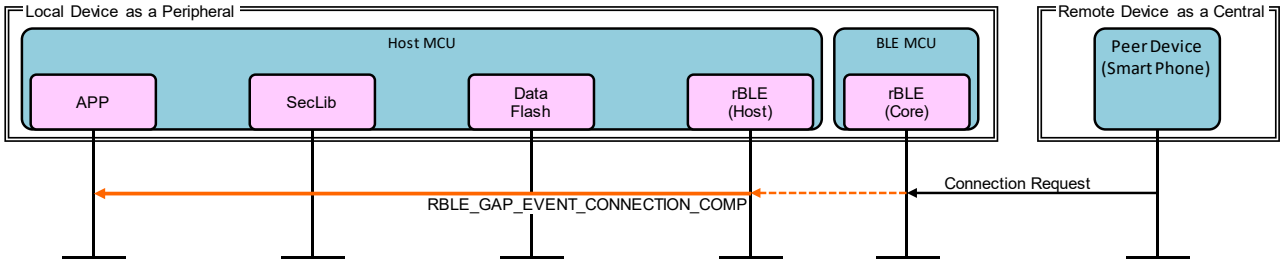

図 7-5　Connection シーケンス

7.6　Step5. Profile Enable シーケンス

データ送信に利用する GPCP（General Purpose Communication Profile）の Server を有効化します。

APP は RBLE_VUART_Server_Enable 関数をコールし、GPCP を有効化します。有効化は、Remote Device から Indication を許可する Write Client Characteristic Configuration が送信されると完了します。「図 7-10　Profile Communication シーケンス」を参照してください。

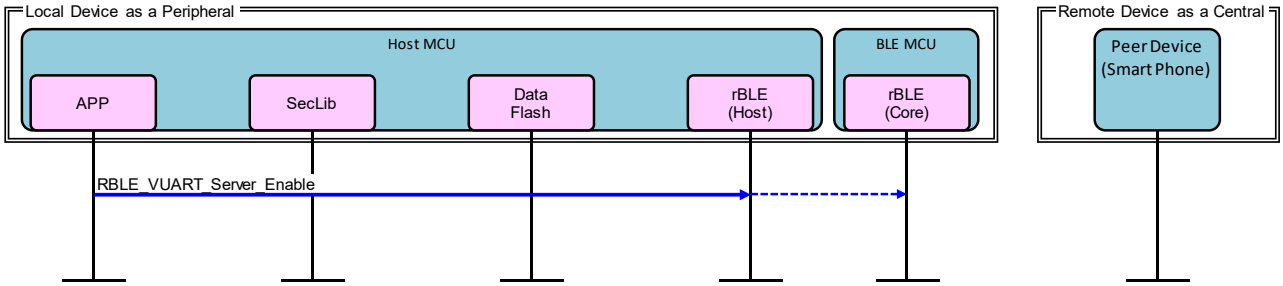

図 7-6　Profile Enable シーケンス

7.7　Step6. Remote Device Check シーケンス

Remote Device とのペアリングの状態が、完了か未完了かの確認結果である SECLIB_EVENT_CHK_ADDR_COMP イベントが SecLib から通知されます。

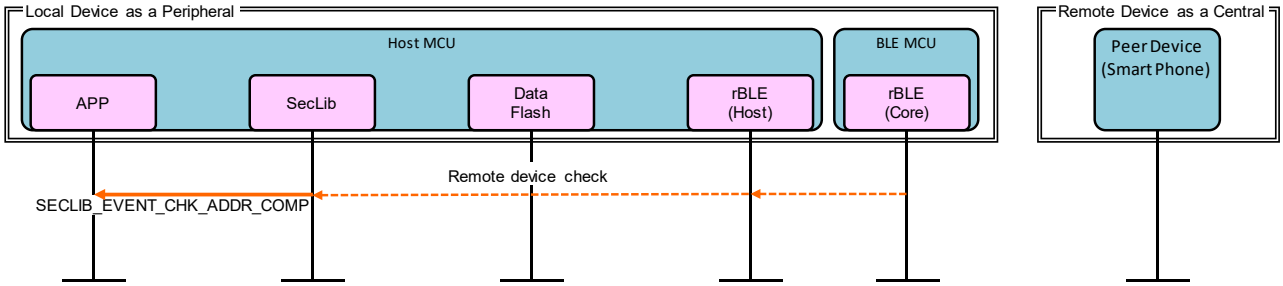

図 7-7　Remote Device Check シーケンス

7.8　Step7. Pairing シーケンス

Remote Device との初回接続または以前の接続にて Pairing を行っていない場合、Remote Device からの Pairing 要求により Pairing シーケンスを開始します。

ペアリングの要求は SecLib から SECLIB_EVENT_PAIRING_REQ イベントが通知されます。APP はペアリングの受諾・拒否のいずれかを SecLib_Pairing_Req_Resp 関数をコールして応答します。

Remote Device とのペアリングが完了すると、SecLib はセキュリティ情報を Data Flash に保存し、SECLIB_EVENT_PAIRING_COMP イベントを通知します。

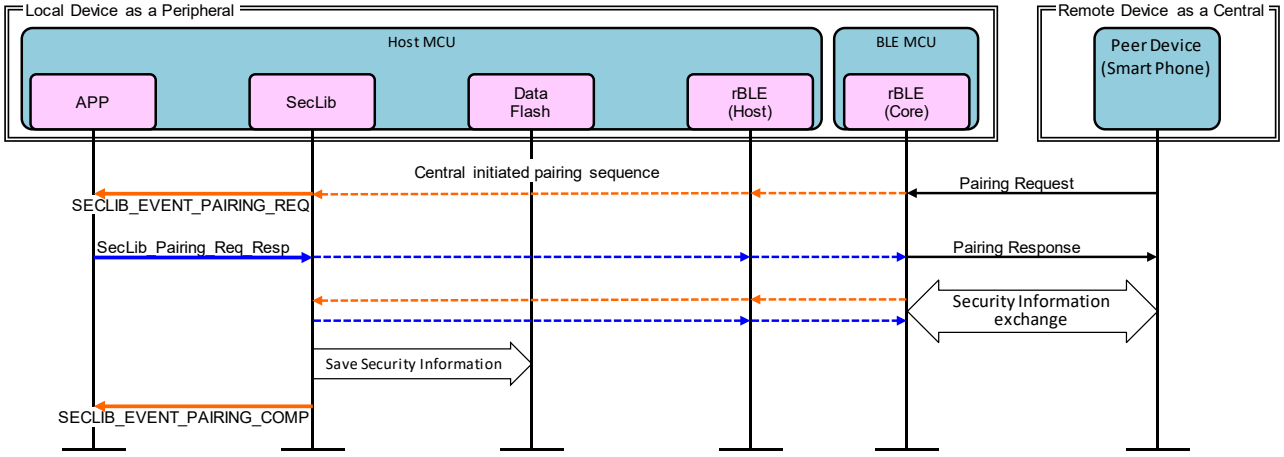

図 7-8　Pairing シーケンス

7.9　Step8. Start Encryption シーケンス

以前の接続で Pairing が完了している場合、セキュリティ情報を使用して暗号化を開始します。

Remote Device から Start Encryption 送信されると、SecLib は LTK を使用して暗号化を開始します。Remote Device との暗号化が完了すると、SecLib はセキュリティ情報を更新し、SECLIB_EVENT_ENC_COMP イベントを通知します。

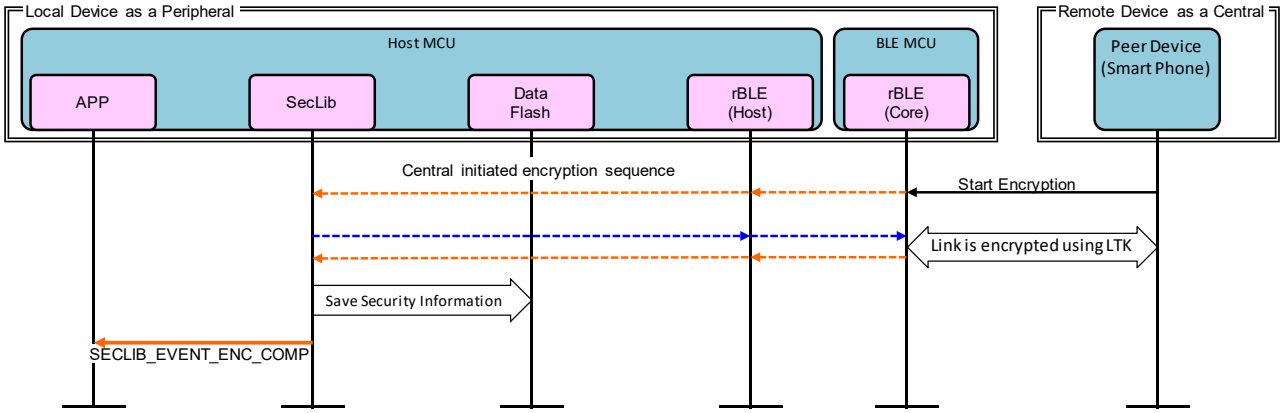

図 7-9　Start Encryption シーケンス

7.10 Step9. Profile Communication シーケンス

GPCP（General Purpose Communication Profile）を利用し Indication によるデータ送信を開始します。

Remote Device から Indication を許可する Write Client Characteristic Configuration が送信されると、rBLE から RBLE_VUART_EVENT_SERVER_ENABLE_COMP イベントが通知されます。

Remote Device から'S'または`s'の文字を受信すると、rBLE から RBLE_VUART_EVENT_SERVER_WRITE_REQ イベントが通知され、センサで湿度、温度の測定を開始します。rBLE Timer を使用して繰り返し測定し、RBLE_VUART_Server_Send_Indication で測定したデータを Indication データとして Remote Device へ送信します。

Remote Device は Indication を受信すると Confirmation を送信します。Local Device が Confirmation を受信すると RBLE_VUART_EVENT_SERVER_INDICATION_CFM イベントが通知されます。

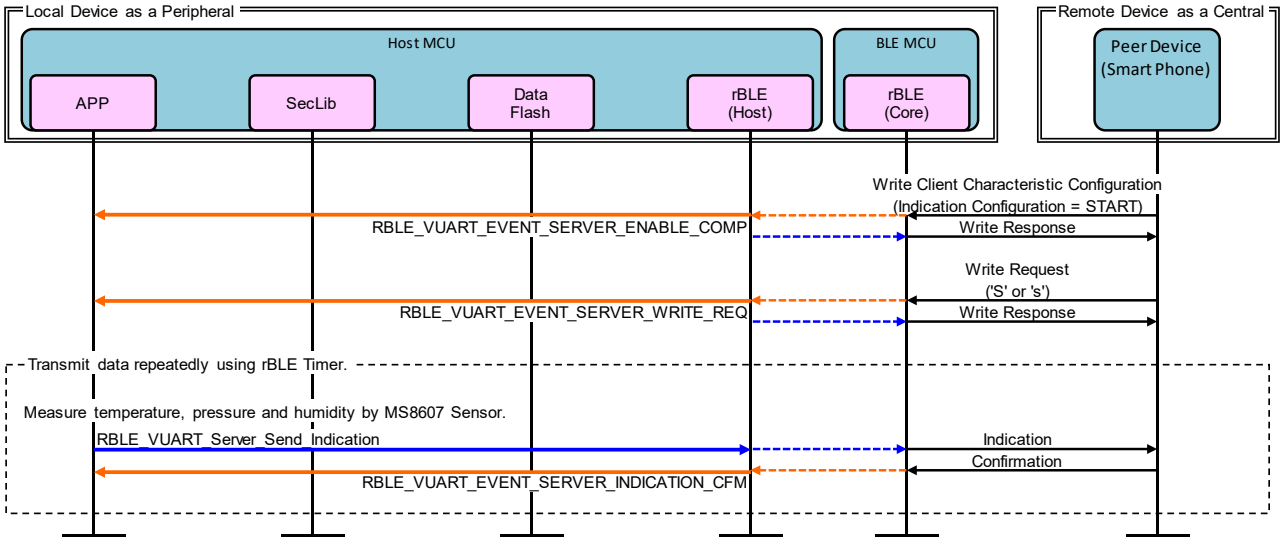

図 7-10　Profile Communication シーケンス

7.11 Step10. Disconnection シーケンス

Remote Device から接続の切断のための Disconnect が送信されると、切断が完了し RBLE_GAP_EVENT_DISCONNECT_COMP イベントが通知されます。

APP は RBLE_VUART_Server_Disable 関数をコールし、GPCP を無効化します。

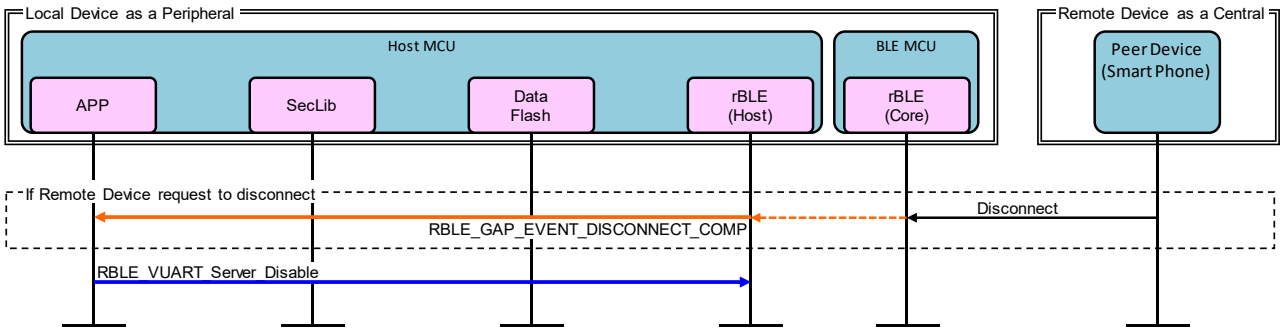

図 7-11　Disconnection シーケンス

8. 付録

8.1 ROM・RAM サイズ

「表 8-1 ROM・RAM サイズ」にホストサンプルが使用する ROM・RAM サイズを示します。

表 8-1 ROM・RAM サイズ

コンパイラ	ROM (bytes)	RAM (bytes)
CC-RL V1.10	43,474	4,823

8.2 参考文献

1. Bluetooth Core Specification v4.2, Bluetooth SIG
2. Bluetooth SIG Assigned Numbers
3. Services UUID
4. Characteristics UUID

8.3 用語説明

用語	英語	説明
サービス	Service	サービスは GATT サーバから GATT クライアントへ提供され、GATT サーバはインタフェースとしていくらかの特性を公開します。 サービスは公開された特性へのアクセス手順について規定します。
プロファイル	Profile	1 つ以上のサービスを使用してユースケースの実現を可能にします。使用するサービスは各プロファイルの仕様にて規定されます。
特性	Characteristic	特性はサービスを識別する値で、各サービスにて公開する特性やそのフォーマットが定義されます。
ロール	Role	役割。それぞれのデバイスが、プロファイルやサービスで規定される役割を果たすことで、ユースケースの実現が可能になります。
コネクションハンドル	Connection Handle	リモートデバイスとの接続を識別するための Controller スタックによって決定されるハンドルです。ハンドルの有効範囲は 0x0000～0x0EFF です。
UUID	Universally Unique Identifier	一意に識別するための識別子です。BLE 規格ではサービスや特性等を識別するために 16bit の UUID が定義されています。
BD アドレス	Bluetooth Device Address	Bluetooth デバイスを識別するための 48bit のアドレスです。BLE 規格ではパブリックアドレスとランダムアドレスが規定されており、少なくともどちらか一方をサポートする必要があります。
パブリックアドレス	Public Address	IEEE に登録し割り当てられた 24bit の OUI(Organizationally Unique Identifier)を含むアドレスです。
ランダムアドレス	Random Address	乱数を含むアドレスで、以下の 3 つに分類されます。 スタティックアドレス Non-resolvable private アドレス Resolvable private アドレス
スタティックアドレス	Static Address	上位 2bit は共に 1 で、残 46bit は全てが 1 または 0 ではない乱数からなるアドレスです。電源断まではそのスタティックアドレスを変更できません。
Non-resolvable private アドレス	Non-resolvable private Address	上位 2bit は共に 0 で、残 46bit は全てが 1 または 0 ではない乱数からなるアドレスです。スタティックおよびパブリックアドレスと等しくてはなりません。 短い期間でアドレスを変更することで攻撃者からの追跡を困難にする目的で使用されます。

Resolvable private アドレス	Resolvable private Address	IRK と 24bit の乱数から生成されるアドレスです。上位 2bit は 0 と 1、上位の残 22bit は全てが 1 または 0 ではない乱数で、下位 24bit は IRK と上位の乱数を元に計算されます。 短い期間でアドレスを変更することで攻撃者からの追跡を困難にする目的で使用されます。 IRK を対向機に配布することで、対向機はその IRK を使用してデバイスを特定することが可能です。
Broadcaster	Broadcaster	GAP のロールで、Advertising データを送信します。
Observer	Observer	GAP のロールで、Advertising データを受信します。
Central	Central	GAP のロールで、物理リンクの確立を行います。
Peripheral	Peripheral	GAP のロールで、物理リンクの確立を受け入れます。
Advertising	Advertising	接続確立や、データ送信の目的のために特定チャネル上でデータを送信します。
Scan	Scan	Advertising データを受信します。Scan には、ただ受信するのみの Passive Scan と、SCAN_REQ を送信することで追加情報を要求する Active Scan があります。
White List	White List	接続済みやボンディング済みなどの既知デバイスを White List に登録しておくことで、Advertising データや接続要求を受け取ることを許可するデバイスをフィルタリングすることが可能です。
デバイス名	Device Name	Bluetooth デバイスに任意につけられたデバイスを識別するためのユーザフレンドリーな名前です。 BLE 規格では、GAP の特性として GATT サーバによって対向機に公開されます。
Reconnection Address	Reconnection Address	Non-resolvable private アドレスを使用して、短い期間でアドレスを変更する場合、攻撃者だけでなく対向機もデバイスの特定が困難になります。そのため対向機の公開する Reconnection Address 特性に新しい Reconnection Address を設定することで再接続時のアドレスを通知します。
コネクションインターバル	Connection Interval	接続確立後に定期的にデータの送受信を行う間隔です。
コネクションイベント	Connection Event	コネクションインターバルごとにデータの送受信を行う期間です。
スーパービジョンタイムアウト	Supervision Timeout	対向機からの応答がなく、リンクが切断されたとみなすタイムアウト時間です。
Passkey Entry	Passkey Entry	ペアリング方式の一つで、互いのデバイスで 6 桁の数値入力または、一方で 6 桁の数値表示、もう一方でその数値入力を行います。

Just Works	Just Works	ペアリング方式の一つで、ユーザアクションを必要としません。
OOB	OOB	ペアリング方式の一つで、Bluetooth 以外の通信方式で取得したデータを使用してペアリングを行います。
IRK	Identity Resolving Key	Resolvable private アドレスの生成や解決に用いる 128bit のキーです。
CSRK	Connection Signature Resolving Key	データ署名の作成および、受信データの署名の確認に使用される 128bit のキーです。
LTK	Long Term Key	暗号化に使用される 128bit のキーです。使用するキーサイズはペアリング時に同意されたサイズになります。
STK	Short Term Key	キー交換時に暗号化するために使用される 128bit のキーです。TK を用いて生成されます。
TK	Temporary Key	STK 生成に必要となる 128bit のキーです。Just Works の場合は 0、Passkey Entry は入力された 6 桁の数値、OOB は OOB データが TK の値となります。

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	2019.06.30	ー	初版発行
1.01	2020.01.10	16	「4.3 Renesas Flash Programmer を使用した HEX ファイルの書き込み」を追加。
		19	「5.3.1 Android デバイスでの通信動作確認」の図 A6 で、スマートフォンへ送信するセンサデータをバイナリからテキストに変更。
		21	「5.3.2 iOS デバイスでの通信動作確認」の図 B6 で、スマートフォンへ送信するセンサデータをバイナリからテキストに変更。
		24	「6.3 GPCP 送受信関数」を追加。
		29	「6.5 センサの無効化」を追加。
1.10	2020.12.18	4	「1 概要」を更新。
		5	「2.2 ソフトウェア環境」の統合環境/コンパイラバージョンを更新。
		7	「3.1.2 ユーザスイッチ(SW_USR)」を追加。
		8	「3.2 ソフトウェア構成」に Security Library の説明を追加。
		10	「3.3 周辺機能構成」にユーザスイッチで使用する割り込み機能を追加。
		13	「3.4 ファイル構成」を更新。
		32	「7 通信シーケンス」を Security Library を使用した通信シーケンスに変更。
		38	「8.1 ROM・RAM サイズ」を更新。
		-	下記の記載を変更。 ・Master を Central ・Slave を Peripheral
1.20	2021.03.12	ー	センサを TE Connectivity 社製から Renesas HS3001 Humidity and Temperature Sensor Module に変更。

製品ご使用上の注意事項

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

1. 静電気対策
   CMOS 製品の取り扱いの際は静電気防止を心がけてください。CMOS 製品は強い静電気によってゲート絶縁破壊を生じることがあります。運搬や保存の際には、当社が出荷梱包に使用している導電性のトレーやマガジンケース、導電性の緩衝材、金属ケースなどを利用し、組み立て工程にはアースを施してください。プラスチック板上に放置したり、端子を触ったりしないでください。また、CMOS 製品を実装したボードについても同様の扱いをしてください。
2. 電源投入時の処置
   電源投入時は、製品の状態は不定です。電源投入時には、LSI の内部回路の状態は不確定であり、レジスタの設定や各端子の状態は不定です。外部リセット端子でリセットする製品の場合、電源投入からリセットが有効になるまでの期間、端子の状態は保証できません。同様に、内蔵パワーオンリセット機能を使用してリセットする製品の場合、電源投入からリセットのかかる一定電圧に達するまでの期間、端子の状態は保証できません。
3. 電源オフ時における入力信号
   当該製品の電源がオフ状態のときに、入力信号や入出力プルアップ電源を入れないでください。入力信号や入出力プルアップ電源からの電流注入により、誤動作を引き起こしたり、異常電流が流れ内部素子を劣化させたりする場合があります。資料中に「電源オフ時における入力信号」についての記載のある製品は、その内容を守ってください。
4. 未使用端子の処理
   未使用端子は、「未使用端子の処理」に従って処理してください。CMOS 製品の入力端子のインピーダンスは、一般に、ハイインピーダンスとなっています。未使用端子を開放状態で動作させると、誘導現象により、LSI 周辺のノイズが印加され、LSI 内部で貫通電流が流れたり、入力信号と認識されて誤動作を起こす恐れがあります。
5. クロックについて
   リセット時は、クロックが安定した後、リセットを解除してください。プログラム実行中のクロック切り替え時は、切り替え先クロックが安定した後に切り替えてください。リセット時、外部発振子（または外部発振回路）を用いたクロックで動作を開始するシステムでは、クロックが十分安定した後、リセットを解除してください。また、プログラムの途中で外部発振子（または外部発振回路）を用いたクロックに切り替える場合は、切り替え先のクロックが十分安定してから切り替えてください。
6. 入力端子の印加波形
   入力ノイズや反射波による波形歪みは誤動作の原因になりますので注意してください。CMOS 製品の入力がノイズなどに起因して、V_{IL}（Max.）から V_{IH}（Min.）までの領域にとどまるような場合は、誤動作を引き起こす恐れがあります。入力レベルが固定の場合はもちろん、V_{IL}（Max.）から V_{IH}（Min.）までの領域を通過する遷移期間中にチャタリングノイズなどが入らないように使用してください。
7. リザーブアドレス（予約領域）のアクセス禁止
   リザーブアドレス（予約領域）のアクセスを禁止します。アドレス領域には、将来の拡張機能用に割り付けられている リザーブアドレス（予約領域）があります。これらのアドレスをアクセスしたときの動作については、保証できませんので、アクセスしないようにしてください。
8. 製品間の相違について
   型名の異なる製品に変更する場合は、製品型名ごとにシステム評価試験を実施してください。同じグループのマイコンでも型名が違うと、フラッシュメモリ、レイアウトパターンの相違などにより、電気的特性の範囲で、特性値、動作マージン、ノイズ耐量、ノイズ幅射量などが異なる場合があります。型名が違う製品に変更する場合は、個々の製品ごとにシステム評価試験を実施してください。

ご注意書き

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

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

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

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

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

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

(Rev.5.0-1　2020.10)

本社所在地

〒135-0061 東京都江東区豊洲 3-2-24（豊洲フォレシア）

www.renesas.com

お問合せ窓口

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

www.renesas.com/contact/

商標について

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